

**Цифровая обработка сигналов**  
Практическое руководство для инженеров  
и научных работников



DEMYSTIFYING TECHNOLOGY SERIES

---

# Digital Signal Processing

A Practical Guide for Engineers and Scientists

by Steven W. Smith



---

Стивен Смит

# Цифровая обработка сигналов

**Практическое руководство  
для инженеров и научных работников**

*Перевод с английского*



Москва  
Издательский дом «Додэка-XXI»  
2011

УДК 621.391.8

ББК 32.811.3+32.871

C50

**Смит, Стивен**

**C50** Цифровая обработка сигналов. Практическое руководство для инженеров и научных работников / Стивен Смит ; пер. с англ. А. Ю. Линовича, С. В. Витязева, И. С. Гусинского. — М. : Додэка-XXI, 2011. — 720 с. +CD : ил. — Доп. тит. л. англ. — ISBN 978-5-94120-145-7.

В книге изложены основы теории цифровой обработки сигналов. Акцент сделан на доступности изложения материала и объяснении методов и алгоритмов так, как они понимаются при практическом использовании. Цель книги — практический подход к цифровой обработке сигналов, позволяющий преодолеть барьер сложной математики и абстрактной теории, характерных для традиционных учебников. Изложение материала сопровождается большим количеством примеров, иллюстраций и текстов программ (которые вы также можете найти на прилагаемом CD).

Книга предназначена научным работникам и инженерам, желающим применять методы цифровой обработки в различных технических сферах. Рекомендуется аспирантам и студентам, изучающим цифровую обработку сигналов.

УДК 621.391.8

ББК 32.811.3+32.871

***Стивен Смит***

**Цифровая обработка сигналов**  
**Практическое руководство для инженеров**  
**и научных работников**

Подписано в печать 28.10.2010. Формат 70×100/16. Бумага офсетная.

Гарнитура «NewtonC». Печать офсетная.

Объём 45 п. л. Усл. п. л. 58,3

Дополнительный тираж 1000 экз. Код DSP(2). Заказ №

Издательский дом «Додэка-XXI»

105318 Москва, а/я 70

Тел./факс: (495) 366-04-56, 365-26-95

E-mail: red@dodeca.ru

Отпечатано с готовых диапозитивов в типографии «Тиль»  
123557, Москва, Тишинский Б. пер, д. 26

Книга «Цифровая обработка сигналов. Практический подход» Стивена Смита подготовлена и издана по договору с Elsevier INC, 200 Wheeler Road, 6th Floor, Burlington, MA01803, USA.

ISBN 978-5-94120-145-7 (рус.)

ISBN 0-750674-44-X (англ.)

© Steven W. Smith

© Издательский дом «Додэка-XXI», 2010

# Содержание

<b>Предисловие .....</b>	11
<b>Благодарности .....</b>	13
<b>Глава 1. Ширина и глубина распространения цифровой обработки сигналов .....</b>	14
1.1. Истоки ЦОС .....	14
1.2. Связь .....	17
1.2.1. Мультиплексирование .....	17
1.2.2. Сжатие .....	18
1.2.3. Эхоподавление .....	18
1.3. Обработка звуковых сигналов .....	19
1.3.1. Музыка .....	19
1.3.2. Синтез речи .....	19
1.3.3. Распознавание речи .....	20
1.4. Эхолокация .....	21
1.4.1. Радиолокация .....	21
1.4.2. Гидролокация .....	22
1.4.3. Сейсморазведка методом отражённых волн .....	22
1.5. Обработка изображений .....	23
1.5.1. Изображения в медицине .....	23
1.5.2. Изображения, получаемые в космосе .....	24
1.5.3. Коммерческие продукты .....	25
<b>Глава 2. Математическая статистика и случайные сигналы .....</b>	26
2.1. Сигналы и их графическое отображение .....	26
2.2. Среднее значение и среднеквадратическое отклонение .....	28
2.3. Сигналы и процессы .....	33
2.4. Гистограмма, распределение вероятностей, функция плотности вероятности .....	35
2.5. Нормальное распределение .....	43
2.6. Генерация цифрового шума .....	47
2.7. Точность и погрешность .....	50
<b>Глава 3. АЦП и ЦАП .....</b>	53
3.1. Теорема отсчётов .....	57
3.2. Цифро-аналоговое преобразование .....	64
3.3. Аналоговые фильтры преобразования данных .....	67
3.4. Выбор антиэлайзингового фильтра .....	75
3.5. Многоскоростная обработка в процессе аналого-цифрового преобразования .....	79
3.6. Однобитные аналого-цифровое и цифро-аналоговое преобразование .....	81
<b>Глава 4. Программное обеспечение ЦОС .....</b>	88
4.1. Представление чисел в компьютере .....	88
4.2. Числа в формате с фиксированной точкой (целочисленный формат) .....	89
4.3. Числа в формате с плавающей точкой (вещественный формат) .....	91
4.4. Точность представления чисел .....	94
4.5. Скорость вычислений: влияние языка программирования .....	98
4.6. Скорость вычислений: влияние аппаратной платформы .....	104

4.7. Скорость вычислений: советы для программистов . . . . .	109
<b>Глава 5. Линейные системы . . . . .</b>	<b>112</b>
5.1. Сигналы и системы . . . . .	112
5.2. Условия линейности системы . . . . .	114
5.3. Статическая характеристика и передача гармонических сигналов . . . . .	117
5.4. Примеры линейных и нелинейных систем . . . . .	120
5.4.1. Примеры линейных систем (процессов) . . . . .	120
5.4.2. Примеры нелинейных систем . . . . .	121
5.5. Особые свойства линейности . . . . .	121
5.6. Принцип суперпозиции — фундаментальное понятие ЦОС . . . . .	123
5.7. Наиболее распространённые виды декомпозиции . . . . .	126
5.7.1. Импульсная декомпозиция . . . . .	126
5.7.2. Ступенчатая декомпозиция . . . . .	128
5.7.3. Декомпозиция на основе сигналов с чётной и нечётной симметрией . . . . .	128
5.7.4. Декомпозиция с прореживанием . . . . .	129
5.7.5. Декомпозиция Фурье . . . . .	130
5.8. Если система нелинейна . . . . .	132
<b>Глава 6. Свёртка . . . . .</b>	<b>134</b>
6.1. Дельта-функция и импульсная характеристика . . . . .	134
6.2. Свёртка . . . . .	136
6.3. Описание свёртки со стороны входа системы . . . . .	139
6.4. Описание свёртки со стороны выхода системы . . . . .	143
6.5. Сумма взвешенных входных отсчётов . . . . .	149
<b>Глава 7. Свойства свёртки . . . . .</b>	<b>151</b>
7.1. Типовые импульсные характеристики . . . . .	151
7.1.1. Единичный импульс . . . . .	151
7.1.2. Процедуры дифференциального и интегрального исчисления . . . . .	153
7.1.3. Фильтры нижних и верхних частот . . . . .	156
7.1.4. Каузальные и некаузальные сигналы . . . . .	158
7.1.5. Сигналы с нулевой, линейной и нелинейной фазой . . . . .	159
7.2. Математические свойства свёртки . . . . .	161
7.2.1. Свойство коммутативности . . . . .	161
7.2.2. Свойство ассоциативности . . . . .	161
7.2.3. Свойство дистрибутивности . . . . .	162
7.2.4. Преемственность между входом и выходом . . . . .	163
7.2.5. Центральная предельная теорема . . . . .	164
7.3. Корреляция . . . . .	165
7.4. Скорость вычислений . . . . .	168
<b>Глава 8. Дискретное преобразование Фурье . . . . .</b>	<b>170</b>
8.1. Преобразование Фурье . . . . .	170
8.2. Действительное ДПФ: терминология и обозначения . . . . .	176
8.3. Независимая переменная при описании в частотной области . . . . .	178
8.4. Базисные функции ДПФ . . . . .	180
8.5. Синтез сигнала с помощью обратного ДПФ . . . . .	182
8.6. Анализ сигналов на основе ДПФ . . . . .	187
8.6.1. Вычисление ДПФ решением системы уравнений . . . . .	187
8.6.2. Вычисление ДПФ с помощью свёртки . . . . .	188
8.7. Дуальность . . . . .	192
8.8. Форма представления в полярных координатах . . . . .	192
8.9. Проблемы представления в полярных координатах . . . . .	195

---

8.9.1. Проблема 1: радианы или градусы? . . . . .	195
8.9.2. Проблема 2: ошибка деления на ноль . . . . .	196
8.9.3. Проблема 3: неоднозначность значений арктангенса . . . . .	197
8.9.4. Проблема 4: неправильная фаза при очень маленьких значениях модуля . . . . .	197
8.9.5. Проблема 5: неоднозначность периода фазы $2\pi$ . . . . .	198
8.9.6. Проблема 6: модуль всегда положителен (неоднозначность половины периода фазы $\pi$ ) 199	
8.9.7. Проблема 7: скачки между $\pi$ и $-\pi$ . . . . .	200
<b>Глава 9. Применение ДПФ . . . . .</b>	201
9.1. Спектральный анализ сигналов. . . . .	201
9.2. Частотные характеристики систем . . . . .	210
9.3. Свёртка в частотной области . . . . .	213
<b>Глава 10. Свойства преобразования Фурье . . . . .</b>	219
10.1. Линейность преобразования Фурье . . . . .	219
10.2. Свойства фазовой характеристики . . . . .	222
10.3. Периодичность сигналов ДПФ . . . . .	229
10.4. Сжатие и расширение сигналов. Многоскоростная обработка . . . . .	235
10.5. Умножение сигналов (амплитудная модуляция) . . . . .	239
10.6. Преобразование Фурье дискретного времени. . . . .	241
10.7. Уравнение Парсеваля . . . . .	243
<b>Глава 11. Пары Фурье-преобразований . . . . .</b>	245
11.1. Единичный импульс . . . . .	245
11.2. Функция $\sin(x)/x$ . . . . .	248
11.3. Другие пары преобразований Фурье . . . . .	251
11.4. Колебания Гиббса . . . . .	254
11.5. Гармоники . . . . .	256
11.6. ЛЧМ-сигналы . . . . .	259
<b>Глава 12. Быстрое преобразование Фурье . . . . .</b>	262
12.1. Комплексное ДПФ для действительных сигналов . . . . .	262
12.2. Как работает алгоритм БПФ . . . . .	265
12.3. Примеры программной реализации БПФ . . . . .	270
12.4. Сравнение по точности и быстродействию . . . . .	275
12.5. Дополнительное повышение быстродействия . . . . .	277
<b>Глава 13. Аналоговая обработка сигналов . . . . .</b>	282
13.1. Дельта-функция . . . . .	282
13.2. Операция свёртки . . . . .	284
13.3. Интеграл Фурье . . . . .	290
13.4. Ряд Фурье . . . . .	293
<b>Глава 14. Введение в цифровую фильтрацию . . . . .</b>	299
14.1. Основные понятия . . . . .	299
14.2. Формы представления информации в сигнале . . . . .	303
14.3. Временные характеристики . . . . .	304
14.4. Частотные характеристики . . . . .	305
14.5. Высокочастотные, полосовые и режекторные фильтры . . . . .	310
14.6. Классификация фильтров . . . . .	314
<b>Глава 15. Однородные фильтры . . . . .</b>	316
15.1. Однородные нерекурсивные фильтры . . . . .	316
15.2. Переходная характеристика и подавление шума . . . . .	317

15.3. Частотная характеристика . . . . .	319
15.4. Модифицированные однородные фильтры . . . . .	320
15.5. Однородные рекурсивные фильтры . . . . .	322
<b>Глава 16. Оконные фильтры . . . . .</b>	<b>325</b>
16.1. Принципы построения оконных фильтров. . . . .	325
16.2. Расчёт оконного фильтра . . . . .	329
16.3. Примеры использования оконных фильтров . . . . .	332
16.4. Достижение сверхвысокой точности . . . . .	336
<b>Глава 17. Специальные фильтры . . . . .</b>	<b>338</b>
17.1. АЧХ произвольной формы . . . . .	338
17.2. Коррекция (выравнивание) частотной характеристики . . . . .	342
17.3. Оптимальная фильтрация . . . . .	349
<b>Глава 18. Быстрая свёртка . . . . .</b>	<b>353</b>
18.1. Свёртка с секционированием . . . . .	353
18.2. Быстрая свёртка . . . . .	355
18.3. Сокращение вычислительных затрат . . . . .	360
<b>Глава 19. Рекурсивные фильтры . . . . .</b>	<b>362</b>
19.1. Рекурсивный метод . . . . .	362
19.2. Однополюсный рекурсивный фильтр . . . . .	365
19.3. Узкополосный рекурсивный фильтр . . . . .	369
19.4. ФЧХ рекурсивных фильтров . . . . .	371
19.5. Применение целочисленной арифметики. . . . .	375
<b>Глава 20. Фильтры Чебышева . . . . .</b>	<b>377</b>
20.1. Частотные характеристики фильтров Чебышева и Баттерворта . . . . .	377
20.2. Расчёт фильтра . . . . .	378
20.3. Переходная характеристика и перерегулирование. . . . .	386
20.4. Устойчивость . . . . .	387
<b>Глава 21. Сравнительный анализ фильтров . . . . .</b>	<b>392</b>
21.1. Первый раунд: аналоговый фильтр против цифрового . . . . .	392
21.2. Второй раунд: оконный фильтр против фильтра Чебышева . . . . .	395
21.3. Третий раунд: однородный фильтр против однополюсного . . . . .	398
<b>Глава 22. Обработка звука . . . . .</b>	<b>400</b>
22.1. Слух человека . . . . .	400
22.2. Тембр . . . . .	404
22.3. Компромисс между качеством звука и частотой дискретизации. . . . .	407
22.4. Аудио высокого качества . . . . .	408
22.5. Компандирование . . . . .	412
22.6. Синтез и распознавание речи . . . . .	413
22.7. Нелинейная обработка звуковых сигналов . . . . .	418
<b>Глава 23. Формирование и демонстрация изображений . . . . .</b>	<b>422</b>
23.1. Структура цифрового изображения . . . . .	422
23.2. Фотоаппарат и глаз человека . . . . .	425
23.3. Телевизионный видеосигнал . . . . .	434
23.4. Другой способ формирования и демонстрации изображений. . . . .	436
23.5. Регулировка яркости и контрастности. . . . .	437
23.6. Преобразование шкалы серого . . . . .	441
23.7. Деформирование изображений. . . . .	443

---

<b>Глава 24. Линейная обработка изображений</b>	447
24.1. Свёртка	447
24.2. Модификация границ с помощью ФРТ размерностью $3 \times 3$ пикселя	452
24.3. Свёртка при выполнении условия сепарабельности	454
24.4. ФРТ большой размерности. Выравнивание освещённости	457
24.5. Фурье-анализ изображений	461
24.6. Быстрая свёртка	466
24.7. Более внимательный взгляд на свёртку изображений	469
<b>Глава 25. Особые методы обработки изображений</b>	473
25.1. Пространственное разрешение	473
25.2. Шаг и апертура выборки	480
25.3. Отношение сигнал/шум	483
25.4. Морфологическая обработка изображений	487
25.5. Компьютерная томография	493
<b>Глава 26. Нейронные сети</b>	503
26.1. Задача обнаружения	503
26.2. Архитектура нейронных сетей	510
26.3. Почему это работает?	515
26.4. Обучение нейронной сети	517
26.5. Оценка результатов	525
26.6. Проектирование рекурсивных фильтров	528
<b>Глава 27. Сжатие данных</b>	534
27.1. Стратегии сжатия данных	534
27.2. Кодирование длин серий	536
27.3. Кодирование Хаффмана	538
27.4. Дельта-кодирование	540
27.5. LZW-сжатие	541
27.6. Сжатие с преобразованием	548
27.7. MPEG	555
<b>Глава 28. Цифровые сигнальные процессоры</b>	557
28.1. Чем ЦСП отличаются от других микропроцессоров	557
28.2. Циклическая буферизация	560
28.3. Архитектура цифрового сигнального процессора	563
28.4. Процессоры с фиксированной и плавающей точкой	569
28.5. Си или Ассемблер?	576
28.6. Насколько быстры ЦСП?	582
28.7. Рынок цифровых сигнальных процессоров	588
<b>Глава 29. Начинаем работать с ЦСП</b>	592
29.1. Семейство ADSP-2106x	592
29.2. Стартовый набор SHARC EZ-KIT Lite	594
29.3. Пример проектирования: КИХ-фильтр обработки аудиосигналов	596
29.4. Аналоговые измерения с помощью цифровых систем	599
29.5. К вопросу о выборе фиксированной или плавающей точки	601
29.6. Более мощные программные инструменты	604
<b>Глава 30. Комплексные числа</b>	610
30.1. Основные понятия	610
30.2. Полярная форма записи комплексных чисел	614
30.2.1. Метод замещения комплексными числами	617

30.3. Комплексное представление синусоидальных функций . . . . .	619
30.4. Описание систем с использованием комплексных чисел . . . . .	621
30.5. Анализ электрических цепей . . . . .	622
<b>Глава 31. Комплексное преобразование Фурье . . . . .</b>	<b>627</b>
31.1. Вещественное ДПФ . . . . .	627
31.2. Математическая эквивалентность . . . . .	629
31.3. Комплексное ДПФ . . . . .	630
31.4. Семейство преобразований Фурье . . . . .	635
31.4.1. Четвёрка преобразований Фурье . . . . .	635
31.4.2. Вещественные и комплексные преобразования . . . . .	636
31.4.3. Анализ и синтез . . . . .	636
31.4.4. Система обозначений во временной области . . . . .	636
31.4.5. Система обозначений в частотной области . . . . .	636
31.4.6. Уравнения анализа . . . . .	637
31.4.7. Уравнения синтеза . . . . .	637
31.4.8. Масштабирование . . . . .	637
31.4.9. Другие формы записи уравнений . . . . .	638
31.5. Зачем использовать комплексное преобразование Фурье? . . . . .	638
<b>Глава 32. Преобразование Лапласа . . . . .</b>	<b>642</b>
32.1. Понятие $S$ -плоскости . . . . .	642
32.2. Стратегия преобразования Лапласа . . . . .	649
32.3. Анализ электрических схем . . . . .	654
32.4. Значение нулей и полюсов . . . . .	658
32.5. Расчёт фильтров в $S$ -области . . . . .	661
<b>Глава 33. Z-преобразование . . . . .</b>	<b>667</b>
33.1. Понятие $Z$ -плоскости . . . . .	667
33.2. Анализ рекурсивных систем . . . . .	672
33.3. Каскадное и параллельное соединения . . . . .	677
33.4. Инверсия АЧХ . . . . .	681
33.5. Нормировка коэффициента усиления . . . . .	683
33.6. Расчёт фильтров. Метод Чебышева–Баттерворт . . . . .	685
33.6.1. Основной цикл программы . . . . .	685
33.6.2. Объединение коэффициентов . . . . .	688
33.6.3. Вычисление координат полюсов фильтра на $S$ -плоскости . . . . .	688
33.6.4. Преобразование окружности в эллипс . . . . .	689
33.6.5. Переход от аналогового фильтра к цифровому . . . . .	691
33.6.6. Преобразование НЧ-фильтра в НЧ-фильтр . . . . .	693
33.6.7. Преобразование НЧ-фильтра в ВЧ-фильтр . . . . .	694
33.7. Две стороны ЦОС: хорошая и плохая . . . . .	694
<b>Глоссарий . . . . .</b>	<b>695</b>
<b>Предметный указатель . . . . .</b>	<b>713</b>

# **Предисловие**

## **Цель и стратегия книги**

Технический мир меняется очень быстро. Всего за 15 лет мощность персональных компьютеров возросла в тысячу раз, и, по всем прогнозам, возрастёт ещё в тысячу раз в течение следующих 15 лет<sup>1)</sup>. Громадные вычислительные мощности изменили пути развития науки и техники, и цифровая обработка сигналов (ЦОС) — прекрасное тому подтверждение.

В начале 1980-х годов ЦОС изучали в высших учебных заведениях при подготовке магистров электротехнических специальностей. Спустя десятилетие ЦОС вошла в состав стандартных дисциплин базовых университетских курсов. Сегодня ЦОС стала неотъемлемой частью основных навыков, необходимых учёным и инженерам разных специальностей. К сожалению, университетский курс по цифровой обработке сигналов не смог приспособиться к такой перемене. Почти все учебники по этой дисциплине написаны в традиционном электротехническом стиле с характерными для него математическими подробностями и скрупулёзностью. Возможности ЦОС просто огромны, но, не постигнув эту науку, вы не сможете ими воспользоваться.

Эта книга предназначена для широкого круга научных работников и инженеров различных областей: физики, биоинженерии, геологии, океанографии, машиностроения, электротехники и многих других сфер. Целью её написания является идея освещения практических методов и приёмов без привлечения математических подробностей и абстрактной теории. Для воплощения этой идеи в ходе написания книги были реализованы три следующих принципа.

Во-первых, технические приёмы не только доказываются с помощью математических выражений, но и подробно объясняются. Хотя в книгу включено много математики, она не используется как основное средство для подачи материала. Ничто не может сравниться с интересным, хорошо написанным разделом книги, сопровождаемым замечательными примерами и иллюстрациями.

Во-вторых, комплексное исчисление рассматривается в тематических разделах повышенной сложности, к изучению которых приступают только после того, как усвоены основные понятия. Вся базовая теория ЦОС объясняется в главах 1...29 с использованием простых алгебраических выражений, и только в редких случаях применяются элементарные операции интегрального и дифференциального исчислений. В главах 30...33 показано, как математика

---

<sup>1)</sup> Книга была опубликована в 2003 году, а написана автором в конце предыдущего столетия. — *Примеч. пер.*

комплексных чисел расширяет возможности цифровой обработки сигналов, предлагая приёмы, которые не могут быть реализованы с помощью действительных чисел. Многие могут назвать такой подход лженаучным, ведь классические учебники по цифровой обработке сигналов просто наводнены математикой комплексных чисел, зачастую начиная с первых страниц.

В-третьих, в книге используются совсем простые компьютерные программы. Большинство реальных программ цифровой обработки сигналов пишется на языках Си, Фортран или других, им подобных. Однако изучение ЦОС ставит совсем иные требования, нежели её практическое использование. Студентам необходимо сосредоточить своё внимание на алгоритмах и методах обработки, не отвлекаясь на особенности определённого языка. Простота программ в данном случае очень важный фактор. Программы в этой книге написаны таким образом, чтобы обучать методам цифровой обработки сигналов в наиболее доступной форме, при этом все остальные факторы воспринимаются как вторичные. Хорошим стилем программирования можно и пренебречь, если это позволит получить более понятную логику программы. Например:

- используется упрощённая версия языка программирования Бейсик;
- используется нумерация строк;
- из управляющих структур используются только циклы вида FOR–NEXT;
- не используются команды ввода/вывода.

Этот стиль программирования — самый простой из всех известных. Возможно, кто-то подумает, что язык программирования Си больше подошёл бы для данной книги. Я бы не стал участвовать в таком споре.

## Круг читателей

Данная книга задумывалась в первую очередь как одногодичный практический курс по цифровой обработке сигналов для студентов, обучающихся по различным специальностям. Предполагается предварительное знакомство читателей со следующими дисциплинами:

- практической электроникой (операционными усилителями,  $RC$ -цепями и т. д.);
- программированием (на языках Фортран или других);
- дифференциальным и интегральным исчислением.

Книга написана также с учётом интересов специалистов, уже использующих ЦОС в практической работе. В ней обсуждается целый ряд наиболее распространённых приложений цифровой обработки сигналов: цифровые фильтры, нейронные сети, сжатие данных, обработка звука, изображений и т. д. Данные темы обсуждаются по возможности в специальных главах, так, чтобы читателю не пришлось перелистывать всю книгу в поисках методов решения конкретной задачи.

## Благодарности

Хочу выразить особую благодарность многочисленным рецензентам, которые поделились своими комментариями и предложениями по поводу данной книги: Магнус Аронссон (факультет электротехники, университет штата Юта); Брюс Б. Азими (военно-морские силы США); Вернон Л. Чи (факультет вычислительной техники, университет Северной Каролины); Манохар Дас, доктор философии (факультет электротехники и системотехники, Окландский университет); Кэрол А. Дин (компания Analog Devices); Фред ДеПьери, доктор философии (факультет электротехники, Калифорнийский государственный политехнический университет); Джоуз Фридман, доктор философии (компания Analog Devices); Фредерик К. Дюеннебиер, доктор философии (факультет геологии и геофизики, университет Гаваи, Маноа); Д. Ли Фюгаль (компания Space & Signals Technologies); Филсон Х. Гланз, доктор философии (факультет электронной и компьютерной техники, университет г. Нью-Гемпшира); Кеннет Х. Джэкер (факультет вычислительной техники Аппалачский университет); Раджив Кападиа, доктор философии (факультет электротехники, университет г. Манката); Дэн Кинг (компания Analog Devices); Кевин Лири (компания Analog Devices); Дэйл Мэгоун, доктор философии (факультет вычислительной техники, университет северо-восточной Луизианы); Бен Мбуугуа (компания Analog Devices); Бернард Максум, доктор философии (факультет электротехники, университет Ламара); Пол Морган, доктор философии (факультет геологии, университет Северной Аризоны); Дейл Мицлер (факультет математической науки, университет Акрона); Кристофер Л. Муллен, доктор философии (факультет гражданской инженерии, университет Миссисипи); Синтия Л. Нельсон, доктор философии (Национальная лаборатория Сандия); Бранислава Перуничич-Дразенович, доктор философии (факультет электротехники, университет Ламара); Джон Шмельк, доктор философии (математический факультет, университет содружества Вирджинии); Ричард Р. Шульц, доктор философии (факультет электротехники, университет Северной Дакоты); Дэвид Скольник (компания Analog Devices); Джей Л. Смит, доктор философии (центр аэрокосмических технологий, университет Джорджии); Джеффри Смит, доктор философии (факультет компьютерных наук, университет штата Джорджия); Оскар Янез Суарес, доктор философии (факультет электротехники, университет метрополии, кампус Итапалапу, г. Мехико) и другие рецензенты, которые пожелали остаться неизвестными.

Теперь книга в руках последнего рецензента — в ваших руках. Пожалуйста, потратьте немного времени, чтобы поделиться со мной комментариями и предложениями. Благодаря этому следующее издание будет ещё лучше отвечать вашим потребностям. Всё, что нужно для этого сделать, — отправить небольшое электронное письмо на адрес: [Smith@DSPguide.com](mailto:Smith@DSPguide.com). Это займет у вас не более двух минут. Благодарю. Надеюсь, книга вам понравится.

Стив Смит, январь 1999

## ШИРИНА И ГЛУБИНА РАСПРОСТРАНЕНИЯ ЦИФРОВОЙ ОБРАБОТКИ СИГНАЛОВ

*Цифровая обработка сигналов (ЦОС)* представляет собой одну из наиболее мощных технологий, которая в XXI веке будет определять дальнейшее развитие науки и техники. Революционные изменения уже свершились в *широком* поле сфер деятельности — связи и телерадиовещании, радио- и гидролокации, высококачественном звуковоспроизведении и сжатии изображений и т. д. Каждой из этих областей присуще своё собственное, *глубоко* проработанное направление развития ЦОС со своими алгоритмами, математическим аппаратом и специализированными методами. Очевидно, что освоить все технологии цифровой обработки сигналов, которые существуют на сегодняшний день, практически невозможно. Поэтому изучение методов ЦОС целесообразно разбить на две задачи: изучение общих концепций, справедливых для всех направлений цифровой обработки сигналов, и изучение специализированных методов, применяемых в сфере ваших интересов. В этой главе мы начнём наше путешествие в мир цифровой обработки сигналов и увидим, какой весьма значительный эффект произвела она в различных областях науки и техники. Итак, вперёд!

### 1.1. Истоки ЦОС

Цифровая обработка сигналов отличается от других наук о вычислительных системах уникальным типом данных, с которым она работает, — *сигналами*. В большинстве случаев сигналы формируются на основе информации, получаемой от датчиков, которые фиксируют явления реального мира: сейсмические вибрации, оптические изображения, звуковые волны и т. д. ЦОС представляет собой совокупность математических методов, алгоритмов и технических приёмов, которые используются для управления сигналами после того, как они были преобразованы в цифровую форму. Обработка сигналов может решать множество задач, среди которых повышение качества изображений, распознавание и синтез речи, сжатие данных для их последующего хранения и передачи и т. д. Предположим, мы подключили аналогово-цифровой преобразователь к компьютеру и используем его для получения цифровых данных из реального мира. В этом случае ЦОС позволяет ответить на вопрос: «*Что дальше?*»

Цифровая обработка сигналов появилась в 1960—1970-е годы, во времена, когда впервые стали доступны цифровые компьютеры. Стоимость компьютеров в те годы была довольно высока, и применение ЦОС ограничивалось только некоторыми критически важными задачами. Первые попытки применения цифровой обработки были предприняты в четырёх ключевых областях: в радиолокации и гидролокации как важных составляющих национальной безопасности; при поиске новых нефтяных месторождений, т. к. эта сфера сулит большие доходы; при исследо-

вании космического пространства, где обрабатываются уникальные данные; в рентгенографии, используемой для спасения человеческих жизней. Революция в сфере персональных компьютеров 1980—1990-х годов стала причиной развития ЦОС в новых областях науки и техники. Несмотря на заинтересованность некоторых военных и государственных кругов в использовании ЦОС только для своих задач, цифровая обработка неожиданно стала развиваться по законам свободного рынка. Все, кто хотел заработать деньги в этой быстро развивающейся области, становились поставщиками ЦОС-продуктов. Цифровая обработка сигналов для широкого круга потребителей стала доступна в таких средствах, как мобильные телефоны, CD-плееры, электронная голосовая почта и др. На Рис. 1.1 показаны некоторые из приложений ЦОС.

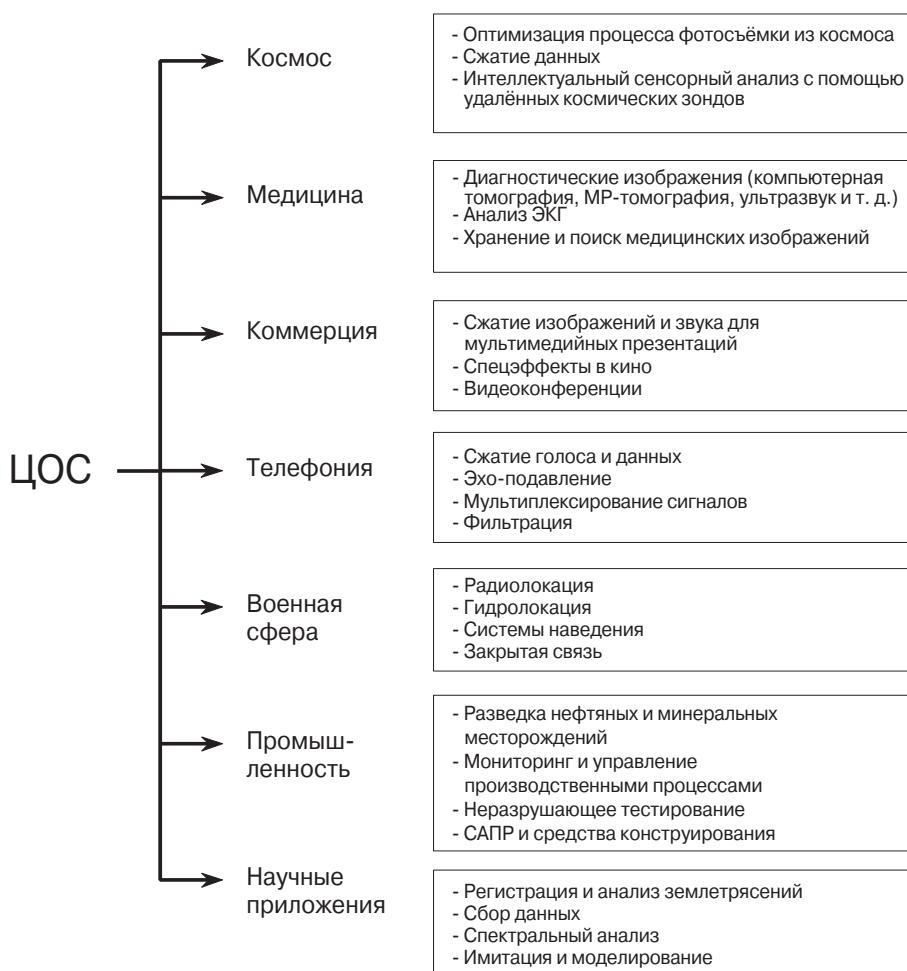
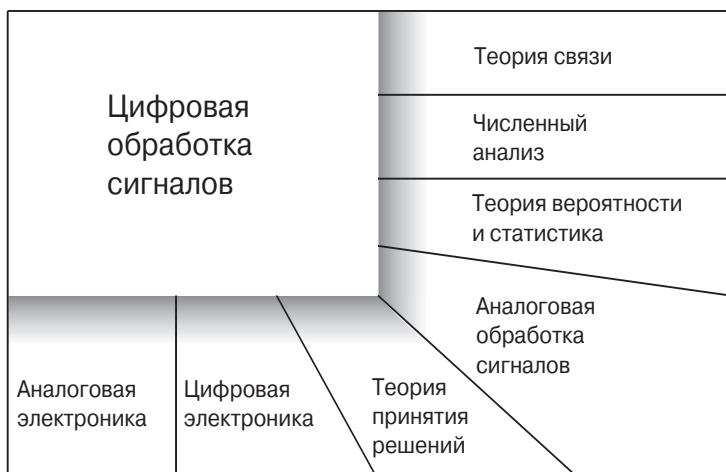


Рис. 1.1. ЦОС осуществила перелом во многих сферах науки и техники.  
Некоторые из них представлены на этом рисунке.

Эта технологическая революция развивалась по принципу «сверху вниз». В начале 1980-х годов ЦОС изучали в высших учебных заведениях при подготовке магистров электротехнических специальностей. Спустя десятилетие ЦОС вошла в состав стандартных дисциплин базовых университетских курсов. Сегодня ЦОС стала неотъемлемой частью базовых навыков, необходимых учёным и инженерам многих специальностей. Цифровую обработку сигналов можно сравнить с *электроникой*, которая совершила предыдущую техническую революцию. И в наши дни главенство электроники продолжается, т. к. почти каждый учёный и инженер обладает хотя бы начальными знаниями по разработке электрических схем. Без этих знаний они бы просто потерялись в современном техническом мире. Цифровой обработке сигналов предстоит такое же будущее.

Эта история не просто любопытна, она должна стать хорошим импульсом к изучению и использованию методов ЦОС. Допустим, вы столкнулись с какой-то трудностью в использовании цифровой обработки сигналов и в поисках решения обращаетесь к учебникам или другим публикациям. Обычно в этих пособиях можно найти целые страницы уравнений, непонятных математических символов и неизвестной терминологии. Это просто кошмар! Большая часть литературы по ЦОС тяжело воспринимается даже теми людьми, у которых есть опыт работы в данной сфере. Не то чтобы это плохой материал, просто он рассчитан на очень узкоспециализированную аудиторию. Такого рода подробная математика нужна современным исследователям для постижения теоретических результатов работы.

Основной идеей создания этой книги явилось то, что большинство методов ЦОС может быть изучено и в дальнейшем использовано без привлечения традиционного аппарата математической теории, становящегося барьером на пути инженера и исследователя. Эта книга написана для тех, кто желает использовать ЦОС в качестве *инструмента*, а не для тех, кто хочет *сделать научную карьеру* в этой области.



**Рис. 1.2.** Цифровая обработка сигналов имеет весьма размытые границы со многими смежными направлениями науки и техники, а также математикой.

Далее в этой главе рассматриваются те области науки и техники, в которых ЦОС произвела поистине революционные изменения. По мере перехода от одной сферы применения к другой вы можете заметить, что цифровая обработка сигналов является междисциплинарной, поскольку основана на теоретической базе многих смежных областей. Кроме того, как показано на Рис. 1.2, границы между ЦОС и другими техническими дисциплинами определены не чётко. Они довольно размыты и обуславливают взаимное проникновение дисциплин друг в друга. Если вы хотите специализироваться в ЦОС, вам придётся изучить и смежные дисциплины.

## 1.2. Связь

**Связь** представляет собой раздел науки и техники, посвящённый передаче информации на расстояние. Передаваться могут самые разные виды информации: телефонные разговоры, телевизионные сигналы, компьютерные файлы и другие типы данных. Для осуществления передачи информации необходимо организовать *канал связи*. Это может быть пара проводов, радиоканал, оптоволокно и т. д. Телекоммуникационные компании взимают плату за передачу информации со своих клиентов, и в то же время они должны сами оплачивать установку и поддержание каналов. Финансовая выгода очевидна: чем больше информации они смогут пропустить через один канал, тем больше денег заработают. ЦОС вызвала настоящую революцию в сфере телекоммуникационной индустрии. Новый уровень производительности и функциональности был достигнут в таких областях, как генерация и детектирование тональных сигналов, сдвиг частотных диапазонов, фильтрация шумов источника питания и многих других. Далее мы рассмотрим три примера применения ЦОС в телефонии: мультиплексирование, сжатие и эхоподавление.

### 1.2.1. Мультиплексирование

В мире насчитывается примерно 1 миллиард телефонов. Всего через несколько секунд после нажатия нескольких кнопок коммутационная схема позволяет соединить любой из этих аппаратов с любым другим. Сложность этой задачи просто поражает! До 1960-х годов соединение между двумя телефонами требовало прохождения аналогового звукового сигнала через механические переключатели и усилители. Для одного соединения необходимо было задействовать одну пару проводов. В современных системах цифровая обработка сигналов преобразует аудиосигналы в последовательный поток цифровых данных. Поскольку биты могут довольно легко объединяться в один поток и затем также легко разделяться, то по одному каналу можно передавать несколько телефонных разговоров. Например, телефонный стандарт под названием *T-carrier system (24-канальная ИКМ-система типа T)* позволяет передавать одновременно до 24 голосовых сигналов. Каждый речевой сигнал подвергается дискретизации со скоростью 8000 отсчётов в секунду посредством 8-битного компандированного (с логарифмическим сжатием) аналого-цифрового преобразования. В результате каждый голосовой сигнал

преобразуется в поток данных со скоростью передачи 64000 бит/с, а все 24 канала передают данные со скоростью 1.544 Мбит/с. Этот сигнал передаётся через обычный медный телефонный кабель с сечением стандарта AWG22 на расстояние около 1800 метров, что является типичной величиной для соединения типа точка–точка. В результате цифровая передача приносит огромную финансовую выгоду, позволяя заменить дорогие кабели и аналоговые коммутирующие устройства на дешёвые цифровые логические схемы.

## 1.2.2. Сжатие

При дискретизации голосового сигнала со скоростью 8000 отсчётов в секунду большая часть цифровой информации является *избыточной*. Происходит это потому, что информация, передаваемая одним отсчётом, в значительной степени дублируется соседними отсчётами. Для преобразования цифрового речевого сигнала в поток данных с минимальным отношением бит/с разработано большое количество алгоритмов ЦОС, которые называют *алгоритмами сжатия*. Для последующего получения сигнала в его первоначальном виде используются соответствующие *алгоритмы восстановления*. Алгоритмы различаются между собой степенью сжатия данных и качеством звука после восстановления. В общем случае уменьшение скорости передачи данных с 64 до 32 Кбит/с не приводит к ухудшению качества звука. При сжатии до скорости передачи 8 Кбит/с звук заметно меняется, но всё же годится для использования, например, в телефонных сетях дальней связи. При самой высокой степени сжатия скорость потока уменьшается до 2 Кбит/с, в результате чего звук сильно искажается, но всё же находит, например, применение в некоторых военных приложениях или для связи под водой.

## 1.2.3. Эхоподавление

Эхо-сигналы представляют собой серьёзную проблему в телефонных соединениях дальней связи. Когда вы говорите по телефону, сигнал, представляющий ваш голос, передаётся к приёмнику, от которого часть этого сигнала возвращается в виде эха. Если длина соединения не превышает нескольких сот километров, то время прихода эхо-сигнала составляет всего несколько миллисекунд. Человеческое ухо адаптировано к восприятию эха с таким малым временем задержки, поэтому качество звука не ухудшается. С увеличением расстояния эхо становится всё более заметным и вызывает неудобство. В случае межконтинентальной связи задержка может достигать нескольких сот миллисекунд, что весьма нежелательно. Системы с цифровой обработкой сигналов решают эту проблему с помощью измерения отражённого сигнала и генерирования соответствующего «антисигнала» для подавления эха. Этот же принцип позволяет использовать «громкую связь» одновременно для прослушивания и передачи голоса без аудиоэффекта обратной связи (свиста). Этот метод можно также использовать и для уменьшения уровня окружающих шумов с помощью цифровой генерации сигнала «антишума».

## 1.3. Обработка звуковых сигналов

Слух, наряду со зрением, является основным чувством восприятия человека. Поэтому неудивительно, что значительная часть ЦОС-систем ориентирована на обработку звуковых сигналов. Воспринимаемую на слух информацию можно разделить на музыку и речь. В обоих этих направлениях цифровая обработка сигналов произвела революционные изменения.

### 1.3.1. Музыка

Путь, ведущий от микрофона музыканта к колонкам меломана, долг и тернист. Цифровая обработка позволяет предотвратить возникающие на этом пути искажения сигнала, в основном связанные с использованием аналоговых устройств преобразования и хранения звукового сигнала. Результат знаком любому, кто сравнивал качество музыки, воспроизведенной с аудиокассет и с компакт-дисков. Как правило, в музыкальной студии музыкальные произведения записываются по нескольким каналам или дорожкам. Иногда производится раздельная запись отдельных инструментов и солистов, чтобы дать звукоинженеру больше возможностей для создания конечного продукта. Сложный процесс объединения звуков отдельных дорожек в окончательный вариант записи музыкального произведения называется *микшированием* (или *сведением фонограмм*). В этом процессе ЦОС может выполнять несколько важных функций, в том числе фильтрацию, суммирование и вычитание сигналов, монтаж и др.

Одним из самых интересных применений ЦОС в процессе создания записи музыкального произведения является *искусственная реверберация*. Если просто сложить вместе звуки отдельных каналов, то полученный таким образом музыкальный фрагмент будет звучать вяло и неярко, как если бы музыканты играли на улице. Это связано с тем, что на слушателей большое впечатление оказывает эхо, или реверберация в музыкальном произведении, которая обычно минимальна в условиях звуковой студии. Применение ЦОС в процессе смешивания звуковых дорожек позволяет добавлять искусственное эхо и имитировать различные условия для прослушивания музыки. Эхо с задержкой в несколько сот миллисекунд позволяет получить звучание, аналогичное тому, какое можно услышать в кафедральном соборе. Добавление эхо-сигналов с задержкой в 10...20 миллисекунд даёт ощущение прослушивания музыки в помещении более скромных размеров.

### 1.3.2. Синтез речи

*Синтез и распознавание речи* применяются для осуществления связи между людьми и машинами. Вместо рук и глаз вы можете использовать рот и уши. Это очень удобно, когда ваши руки и глаза заняты чем-либо другим, например вождением автомобиля, проведением хирургической операции или стрельбой. Для компьютерного синтеза речи используются два метода: *цифровая запись* и *эмulation речевого тракта*. При цифровой записи человеческий голос оцифровывается и хранится в памяти компьютера обычно в сжатом виде. Во время воспроизведения цифровые данные распаковываются и преобразовываются обратно в аналоговый сигнал. Для хранения записи речи длительностью один час потребуется

около 3 МБ памяти, что является весьма хорошим показателем даже для небольших компьютерных систем. Этот метод на сегодняшний день является наиболее распространённым в системах синтеза речи.

Эмуляция речевого тракта является более сложным методом, при котором имитируются физические механизмы, с помощью которых люди разговаривают. Человеческий речевой тракт представляет собой акустический резонатор с определёнными резонансными частотами, величина которых определяется размерами и формой его полостей. Звук в речевом тракте формируется одним из двух основных способов, соответствующих образованию двух типов звуков: вокализованных и фрикативных. При образовании вокализованных звуков вибрация голосовых связок производит в полостях речевого тракта колебания воздуха, близкие к периодическим. Фрикативные звуки образуются при прохождении воздушного потока через тонкие щели, образованные, например, губами и зубами. Эмуляция речевого тракта с помощью цифровой обработки сигналов подразумевает имитацию двух рассмотренных способов формирования звука. Характеристики резонансной полости эмулируются при помощи прохождения возбуждающих сигналов через цифровой фильтр с подобными резонансными характеристиками. Такой метод успешно использовался на раннем этапе развития ЦОС в популярном электронном учебном пособии для детей «Speak & Spell».

### 1.3.3. Распознавание речи

Автоматическое распознавание человеческой речи является существенно более сложным процессом, чем синтез. Распознавание речи — классический пример задачи, с которой легко справляется человеческий мозг, а самый современный цифровой компьютер часто оказывается бессильным. Компьютеры могут хранить и воспроизводить огромное количество данных, за считанные мгновения выполнять математические вычисления, эффективно и без усталости производить повторяющиеся операции. Но, к сожалению, современные компьютеры очень плохо справляются с работой, связанной с обработкой данных реального мира, поступающих непосредственно с датчиков. Довольно легко заставить компьютер ежемесячно присыпать вам счёт за электроэнергию, но научить его понимать вашу речь — задача куда более сложная.

Цифровая обработка сигналов обычно решает задачу распознавания речи в два этапа: *выделение особенностей* и *сопоставление особенностей*. Каждое слово в принимаемом аудиосигнале выделяется отдельно, а затем анализируется для определения типа возбуждений, которые присутствуют в нём, а также значений резонансных частот. Затем обнаруженные параметры сравниваются с ранее сохранёнными примерами произнесённых слов для поиска наилучшего соответствия. Часто в таких системах словарный запас ограничен несколькими сотнями слов, они могут воспринимать речь с обязательными чёткими паузами между словами и должны настраиваться на каждого индивидуального пользователя. Эти ограничения пока вполне подходят для многих коммерческих приложений, но если эти ограничения сравнить с возможностями человека, то компьютер будет выглядеть достаточно бледно. В этой сфере ещё многое предстоит сделать, и того, кто сумеет создать успешный коммерческий продукт, ждёт огромная финансовая прибыль.

## 1.4. Эхолокация

Для получения информации об удалённом объекте наиболее часто используется эффект отражения волн. Например, радиолокационная станция излучает импульсы радиоволн в окружающее воздушное пространство и затем принимает сигналы, отражённые от находящихся в воздухе объектов (самолетов). В гидролокации звуковые волны излучаются в толщу воды для обнаружения подводных лодок и других объектов. Долгое время геологи исследовали землю, формируя на поверхности взрывы и регистрируя эхо-сигналы, отражённые от глубоко лежащих слоёв породы. Несмотря на то что все эти приложения используют общие принципы, для каждого из них характерны свои специфические проблемы и требования. Цифровая обработка сигналов широко и весьма эффективно применяется во всех трёх указанных направлениях.

### 1.4.1. Радиолокация

В английском языке для обозначения радиолокационного устройства используется слово «radar» (радар). Это слово является аббревиатурой словосочетания *Radio Detection And Ranging* (*Радиообнаружение и измерение расстояния*). В простейшей радиолокационной системе передатчик формирует радиоимпульс длительностью несколько микросекунд. Этот импульс поступает в узконаправленную антенну, излучающую соответствующую радиоволну в пространство, где она распространяется со скоростью света. Самолёт, находящийся на пути распространения радиоволны, отражает некоторую часть падающего на него излучения обратно на приёмную антенну, расположенную вблизи передатчика или совмещённую с ним. В таком случае расстояние до объекта можно будет определить по времени, прошедшем от момента излучения импульса до момента регистрации эхо-сигнала. Направление на объект определяется по положению узконаправленной антенны в момент приёма эхо-сигнала.

Рабочая дальность радиолокационной системы определяется двумя параметрами: энергией излучаемого (зондирующего) импульса и уровнем шума радиоприёмника. К сожалению, увеличение энергии импульса обычно требует *увеличения длительности импульса*, а с другой стороны, импульсы большей длительности приводят к снижению точности измерения временного интервала. В результате наступает противоречие между двумя важными параметрами: способностью обнаруживать объекты на большом удалении и точностью определения расстояния до них.

ЦОС произвела революционные изменения в трёх направлениях решения этой проблемы. Во-первых, ЦОС-системы могут сжимать принимаемый импульс, обеспечивая высокое разрешение по дальности (точность измерения расстояния) без уменьшения рабочего диапазона системы. Во-вторых, ЦОС может производить фильтрацию принимаемого сигнала для уменьшения уровня шумов, что способствует увеличению рабочей дальности без ухудшения точности определения расстояния. В-третьих, ЦОС позволяет производить быстрый выбор и генерацию зондирующих импульсов различной формы и длительности, что обеспечивает оптимизацию параметров импульса в каждой конкретной ситуации. А

теперь самое удивительное: многие из этих операций осуществляются на частоте дискретизации, сравнимой с используемой радиочастотой и составляющей несколько сот мегагерц!

### 1.4.2. Гидролокация

Гидролокация в английском языке обозначается словом «*sonar*», которое образовано от слов *SOund NAVigation and Ranging* (звуконавигация и измерение расстояний). Различают *активную* и *пассивную* гидролокацию. При активной гидролокации звуковые импульсы частотой 2...40 кГц излучаются в толщу воды, после чего отражённые эхо-сигналы обнаруживаются и анализируются. Активную гидролокацию используют для обнаружения и определения местоположения подводных тел, навигации, связи и составления карт морского дна. Типичная максимальная рабочая дальность таких систем — 10...100 км. Пассивная гидролокация представляет собой обычное прослушивание подводных звуков, которые включают в себя естественную турбулентность, морскую жизнь, механические звуки подводных лодок и надводных судов. Поскольку при пассивной гидролокации не происходит излучения энергии, то она идеальна для проведения секретных операций, т. е. тогда, когда вы хотите обнаружить другого «парня» и чтобы при этом этот «парень» не обнаружил вас. Самое широкое применение пассивная гидролокация нашла в системах наблюдения военного назначения, предназначенных для обнаружения и отслеживания передвижения подводных лодок. Обычно в пассивной гидролокации применяются более низкие частоты, чем в активной гидролокации, так как звуковые волны проходят через толщу воды с меньшим поглощением. Дальность обнаружения при этом может достигать тысяч километров.

Так же как и в радиолокации, в гидролокации ЦОС произвела коренные изменения во многих сферах: при генерации импульсов, сжатии импульсов и обнаружении сигналов. На первый взгляд гидролокация должна быть проще, чем радиолокация, так как в ней используются более низкие частоты. В то же время гидролокация сложнее радиолокации, так как среда распространения сигнала в ней неоднородна и нестабильна. Гидролокационные системы обычно не являются одноканальными, а используют многоэлементные передающие и принимающие антенные решётки. С помощью управления и смешивания сигналов такая многоэлементная гидролокационная система способна излучать импульс в заданном направлении или регистрировать отражения с нужного курса. Для управления такой многоканальной системой гидролокатор требует применения не менее мощных методов ЦОС, чем радиолокатор.

### 1.4.3. Сейсморазведка методом отражённых волн

Ещё в начале 1920-х годов геофизики обнаружили, что структуру земной коры можно исследовать с помощью звука. Исследователи производили подрыв взрывного устройства и записывали эхо-сигналы, отражённые от граничных слоёв, находящихся на расстоянии более десяти километров ниже поверхности земли. Полученные эхо-сейсмограммы использовались для составления примерных карт структуры приповерхностной области земли. Впоследствии сейсмический метод

отражения стал основным при разведке месторождений нефти и других полезных ископаемых и остаётся таким по сей день.

В идеальном случае направленный в толщу земли звуковой импульс произведёт один эхо-сигнал на каждый пройденный им граничный слой. К сожалению, на практике обычно не всё так просто. Каждый возвращающийся на поверхность эхо-сигнал должен пройти сквозь другие слои, которые находятся выше места его возникновения. В результате может появиться эхо-сигнал, который отразится между слоями и спровоцирует возникновение другого эхо-сигнала от эхо-сигнала, который также будет обнаружен на поверхности. Из-за таких вторичных эхо-сигналов принимаемый сигнал может стать очень сложным и трудным для интерпретации. Поэтому начиная с 1960-х годов для отделения первичных и вторичных эхо-сигналов при составлении сейсмограмм с записью отражённых волн широко используется цифровая обработка сигналов. Как же обходились первые геофизики без ЦОС? Ответ довольно прост: они вели наблюдение в тех районах земли, где многочисленные переотражения сигналов были минимальны. ЦОС позволяет находить нефть в труднодоступных местах, например под толщей океана.

## 1.5. Обработка изображений

Изображения — это особый класс сигналов. Во-первых, они характеризуются изменением некоторого параметра в пространстве (зависимостью от расстояния), в то время как большинство сигналов описывается изменением параметра во времени (зависимостью от времени). Во-вторых, объём содержащейся в них информации очень велик. Например, для хранения одной секунды телевизионного сигнала может потребоваться более 10 МБ. Это более чем в 1000 раз превышает требования хранения речевого сигнала такой же длительности. В-третьих, часто при оценке качества изображения субъективное человеческое мнение одерживает верх над объективными критериями. Данные специфические признаки выделяют сферу обработки изображений в отдельную область ЦОС.

### 1.5.1. Изображения в медицине

В 1895 году Вильгельм Конрад Рентген обнаружил, что рентгеновские лучи способны проходить сквозь материю. Возможность заглянуть внутрь живого человеческого тела произвела революционные изменения в медицине. Всего за несколько лет медицинские рентгеновские системы распространились по всему миру. Несмотря на явный успех, до появления в 1970-х годах ЦОС и других смежных технологий развитие медицинской рентгенографии было ограничено четырьмя проблемами. Во-первых, перекрывающиеся структуры в теле человека могут быть скрыты друг за другом. Например, часть сердца может быть невидима за рёбрами. Во-вторых, не всегда можно определить различие между похожими тканями. Например, отличить кость от мягкой ткани ещё возможно, но довольно трудно отличить опухоль от печени. В-третьих, рентгеновские снимки передают *анатомию* (строение тела), а не *физиологию* (его функционирование). Рентгеновский снимок живого человека выглядит абсолютно так же, как рентгеновский снимок

мёртвого. В-четвёртых, воздействие рентгеновских лучей может вызвать рак, поэтому их использование ограничено.

Проблема перекрывающихся структур была решена в 1971 году с появлением *компьютерной томографии* (раньше называлась *компьютерной осевой томографией*). Компьютерная томография представляет собой классический пример применения цифровой обработки сигналов. С большого количества направлений рентгеновские лучи пропускаются сквозь зондируемый участок тела пациента. Вместо того чтобы просто формировать изображение с помощью принятых рентгеновских лучей, эти сигналы преобразуются в цифровую форму и сохраняются в компьютере. Затем эта информация используется для расчёта и формирования по-слойных изображений структуры человеческого тела. Полученные изображения намного подробнее тех, что получаются с помощью обычных технологий, и позволяют значительно улучшить диагностику и лечение больного. Появление компьютерной томографии было таким же грандиозным событием, как в своё время открытие рентгеновских лучей. Всего через несколько лет каждая крупная клиника мира имела доступ к компьютерному томографу. В 1979 году два основных изобретателя компьютерной томографии — Годфри Хоунсфильд и Аллан Кормак получили Нобелевскую премию в области медицины. *Это замечательный пример использования ЦОС!*

Последние три проблемы использования рентгеновских лучей были решены с помощью применения радио- и звуковых волн. ЦОС играет ключевую роль во всех этих методах. Например, в магнитно-резонансной томографии для зондирования внутренностей человеческого тела используются магнитные поля в сочетании с радиоволнами. Воздействие полей определённой силы и частоты вызывает резонанс квантовых энергетических состояний атомных ядер локального участка человеческой ткани. В результате резонанса излучается вторичная радиоволна, которая детектируется антенной, расположенной возле обследуемого участка тела. Величина и другие характеристики принятого сигнала несут информацию об исследуемом участке, подвергаемом резонансу. Регулируя параметры магнитного поля, можно просканировать всё тело, отображая таким образом внутреннюю его структуру. Обычно эта информация представляется в виде изображения, как в компьютерной томографии. Магнитно-резонансная томография передаёт информацию не только о различиях между разными типами мягкой ткани, но также и о физиологии, например о токе крови по артериям. Магнитно-резонансная томография полностью основана на методах цифровой обработки сигналов и не может быть реализована без их использования.

## 1.5.2. Изображения, получаемые в космосе

Иногда возникает необходимость по возможности максимально улучшить качество фотографического снимка. Довольно часто снимки невысокого качества получаются при фотографировании с автоматических спутников и космических исследовательских станций. Никто не будет отправлять на Марс ремонтную группу только лишь для того, чтобы отрегулировать камеру, нажав на пару кнопок! С помощью ЦОС можно улучшить качество изображений, полученных в крайне неблагоприятных условиях, несколькими способами: регулировкой яркости и контрастности, подчёркиванием границ, подавлением шума, регулировкой фокус-

сировки, уменьшением размытости изображения движущегося объекта и т. д. Могут быть скорректированы изображения с пространственными искажениями, например теми, которые получаются в результате отображения сферических поверхностей (планет) на плоскости. Изображения могут быть помещены в отдельную базу данных, что позволяет отображать информацию различными способами. Например, с помощью воспроизведения видеокадров можно смоделировать полёт летательного аппарата над поверхностью далёкой планеты.

### 1.5.3. Коммерческие продукты

Огромные объёмы информации, которые содержат изображения, представляют собой серьёзную проблему для коммерческих продуктов, ориентированных на широкие круги пользователей. Коммерческие системы должны быть дешёвы, что плохо сочетается с требуемыми большими объёмами памяти и высокими скоростями передачи данных. Одним из путей решения данной проблемы является сжатие изображений. Как и в случае со звуковыми сигналами, изображения содержат большое количество избыточной информации. Применив специальные алгоритмы сжатия информации, можно уменьшить количество битов, необходимых для представления изображения. Для сжатия особенно подходят телевизионные и другие изображения, так как большая часть информации в них переходит из одного кадра в другой без изменений. Подобная технология эффективно применяется в таких коммерческих видеосистемах, как видеотелефоны, компьютерные программы, использующие видеоданные, цифровое телевидение.

## МАТЕМАТИЧЕСКАЯ СТАТИСТИКА И СЛУЧАЙНЫЕ СИГНАЛЫ

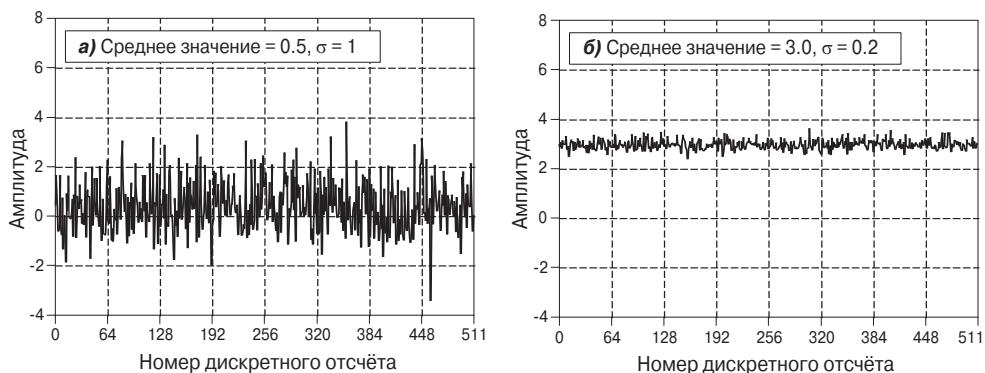
*Математическая статистика и теория вероятностей* используются в ЦОС для описания сигналов и процессов. Почему это необходимо? При решении задач цифровой обработки практически всегда возникает необходимость уменьшения различного рода помех, *шумов* или каких-либо иных нежелательных компонентов в регистрируемом сигнале. Эти компоненты могут являться неотъемлемой частью самого входного сигнала, возникая, например, из-за несовершенной работы системы сбора данных, или появляться как нежелательный побочный эффект в процессе выполнения одной из ЦОС-операций. Статистика и теория вероятностей позволяют измерять и классифицировать эти нежелательные компоненты, что является первым необходимым шагом на пути их устранения. В этой главе описаны наиболее важные статистические и вероятностные концепции с акцентом на то, какую практическую роль они играют в обработке регистрируемых сигналов.

### 2.1. Сигналы и их графическое отображение

*Сигналом* называется процесс изменения состояния некоторого объекта, служащий для отображения и передачи информации. Сигналы выражаются зависимостью одной величины от другой. Например, в аналоговой электронике наиболее типичным примером сигнала может служить изменение во времени уровня электрического напряжения. Поскольку обеим этим величинам (напряжению и времени) свойственна непрерывность принимаемых значений, такой сигнал называется *непрерывным*, или *аналоговым*. При прохождении непрерывного сигнала через аналого-цифровой преобразователь обе эти величины подвергнутся *квантованию*. Пусть, например, выполняется аналого-цифровое преобразование с 12-битной точностью и частотой выборки 1000 отсчётов в секунду. В этом случае диапазон значений напряжения будет ограничен 4096-ю ( $2^{12}$ ) возможными двоичными уровнями, а время сможет увеличиваться только на величину фиксированного шага, равного одной миллисекунде. Сигналы, сформированные в результате такой процедуры квантования по двум величинам одновременно, называют *цифровыми сигналами*. Как правило, аналоговые сигналы существуют в природе, в то время как цифровые сигналы способны существовать только внутри вычислительных машин (заметим, что для обоих случаев есть исключения). Встречаются также сигналы, у которых один параметр изменяется непрерывно, а второй — дискретно. Например, сигнал является *дискретным*, если он может принимать непрерывный ряд значений при дискретных значениях аргумента. Поэтому

му для точного определения типа сигнала необходимо детальное изучение природы описывающих его величин.

На Рис. 2.1 показаны примеры сигналов на входе цифровой системы. По вертикальной оси могут откладываться такие величины, как напряжение, интенсивность излучения, звуковое давление или любой другой параметр. В данном случае мы воспользуемся общим названием — *амплитуда*. Часто для обозначения этой величины используются и другие названия: *ось Y, зависимая переменная, множество значений, ось ординат*.



**Рис. 2.1.** Пример двух сигналов, принимаемых цифровой системой, имеющих различные средние значения и среднеквадратические отклонения.

По горизонтальной оси откладывается вторая величина, представляющая сигнал. Для её обозначения в общем случае также может использоваться ряд терминов: *ось X, независимая переменная, область определения или абсцисса*. Для сигналов, принимаемых системой обработки, наиболее распространённой величиной, откладываемой по горизонтальной оси, является время. В некоторых приложениях, однако, могут использоваться и другие параметры. Например, геофизики измеряют плотность породы через одинаковые расстояния вдоль поверхности земли. В этом случае по горизонтальной оси могут откладываться метры или километры. Чтобы сохранить общность суждений, по горизонтальной оси мы будем откладывать *номера дискретных отсчётов*. Это применимо, однако, только для дискретных сигналов. Если бы на Рис. 2.1 был изображён аналоговый сигнал, то для обозначения горизонтальной оси нужно было бы использовать иные величины, например, время, расстояние,  $x$  и т. д.

Две величины, участвующие в описании сигнала, в общем случае не являются равнозначными. Величина, отложенная по оси  $Y$  (зависимая переменная), представляет собой *функцию* величины, отложенной по оси  $X$  (независимой переменной). Другими словами, независимая переменная показывает, *когда, в какой момент времени* производится выборка каждого следующего отсчёта, а зависимая переменная несёт информацию о значении сигнала в момент выборки. Для каждого значения на оси  $X$  мы всегда можем найти соответствующее значение на оси  $Y$ , но, как правило, не наоборот.

Обратим особое внимание на термин «область определения», который широко используется в ЦОС. Например, сигнал, который в качестве независимой пе-

ременной (т. е. параметра по горизонтальной оси) использует время, называют представленным во *временной области*. Другой распространённый в ЦОС тип сигнала в качестве независимой переменной использует частоту. Такой сигнал называется представленным в *частотной области*. Аналогично сигналы, использующие в качестве независимого параметра расстояние, называются представленными в пространственной области. Таким образом, название области определения сигнала определяется типом параметра по горизонтальной оси. А как поступить, если ось  $X$  обозначена таким общим параметром, как *номер дискретного отсчёта*? Обычно авторы в таких случаях полагают, что сигнал представлен во временной области. Дело в том, что выборка через равные промежутки времени является самым распространённым способом получения сигналов, и если специально не оговорено, то дискретные отсчёты сигнала являются сформированными именно во временной области.

Хотя сигналы на **Рис. 2.1** — дискретные, они изображены в виде непрерывных кривых. Объяснить это можно наличием большого количества отсчётов, что не позволяет в ограниченном пространстве рисунка изобразить их в виде отдельных разделяемых точек. На графиках, изображающих короткие сигналы, скажем, менее 100 отсчётов, обычно используются отдельные точки. При этом соседние дискретные точки сигнала могут соединяться непрерывными линиями, а могут быть и не соединены, в зависимости от того, как автор пожелает представить данные. Соединение непрерывными линиями позволяет показать поведение сигнала между отсчётами и выявить, например, общие тенденции поведения сигнала. Таким образом, для того, чтобы определить, о каких сигналах идёт речь — дискретных или непрерывных, — необходимо проверить обозначения на горизонтальной оси. Полагаться только на умение автора рисовать точки не рекомендуется.

В ЦОС часто используется переменная  $N$ , которая обозначает общее количество отсчётов в сигнале. Например, для сигналов, изображённых на **Рис. 2.1**,  $N = 512$ . Для систематизации данных каждому отсчёту присваивается *номер* или *индекс*. Эти числа вы можете видеть вдоль горизонтальной оси. Для обозначения номеров, присвоенных отсчётам, обычно используют два способа. В первом случае номера идут от 1 до  $N$  (т. е. от 1 до 512), а во втором — номера отсчитываются от 0 до  $N - 1$  (т. е. от 0 до 511). Математики чаще используют первый способ, а в ЦОС большее применение находит второй. В этой книге мы также будем использовать второе обозначение. Не стоит рассматривать это замечание как несущественное: иногда эти правила сбивают с толку. Будьте внимательны!

## 2.2. Среднее значение и среднеквадратическое отклонение

*Среднее значение*, обозначаемое буквой  $\mu$  (маленькая греческая буква *му*), является статистическим термином, обозначающим среднее значение сигнала. Как вы уже догадались, чтобы найти среднее значение, необходимо сложить все отсчёты и полученную сумму разделить на  $N$ . В математической форме это выглядит следующим образом:

$$\mu = \frac{1}{N} \sum_{i=0}^{N-1} x_i. \quad (2.1)$$

Вычисление среднего значения сигнала. Сигнал представляет собой выборку отсчётов от  $x_0$  до  $x_{N-1}$ , где  $i$  — это номер отсчёта, а  $\mu$  — среднее значение.

Представленная формула означает следующее. Необходимо суммировать все значения сигнала  $x_i$ , изменяя индекс  $i$  от 0 до  $N - 1$ , а затем полученную сумму разделить на  $N$ . Эта запись идентична выражению:  $\mu = (x_0 + x_1 + x_2 + \dots + x_{N-1})/N$ . Если вы ещё не знакомы с символом  $\Sigma$  (большая греческая буква *сигма*), который используется для обозначения операции *суммирования*, то внимательно изучите приведенное выражение и проанализируйте реализующую его **Программу 2.1**. Операция суммирования очень часто встречается в ЦОС, и потому вам необходимо разобраться в этой системе обозначений.

В электронике *среднее значение* сигнала обычно называют значением *постоянного тока*. Соответственно *переменный ток* показывает, как сигнал колеблется относительно среднего значения. Если сигнал имеет простую форму, например является синусоидой или меандром, то отклонение его значения от среднего легко определяется амплитудой размаха. К сожалению, у большинства реально принимаемых сигналов невозможно точно определить амплитуду, так как они имеют случайный характер, подобно сигналам на **Рис. 2.1**. Для характеристики степени отклонения уровня сигнала от среднего значения в таких случаях используют параметр, называемый *среднеквадратическим отклонением (СКО)* и обозначаемый символом  $\sigma$  (маленькая греческая буква *сигма*).

Рассмотрим выражение  $|x_i - \mu|$ . Оно показывает, насколько  $i$ -й отсчёт *отклоняется (отличается)* от среднего значения сигнала. *Среднее отклонение* значений сигнала можно найти посредством суммирования отклонений каждого отдельного отсчёта и деления полученной суммы на количество отсчётов, т. е. на  $N$ . Затем, что для суммирования мы должны брать абсолютное значение каждого отклонения, поскольку при сложении положительных и отрицательных значений сумма может оказаться равной нулю. Среднее отклонение представляет собой число, показывающее, насколько в среднем отсчёты сигнала отклоняются от среднего значения. Но, несмотря на простоту и удобство, среднее отклонение почти никогда не используется в статистике, поскольку этот параметр очень плохо описывает физику поведения сигналов. В большинстве случаев намного важнее знать не *отклонение* от среднего значения, а *мощность* этого отклонения. Например, когда случайные сигналы помех объединяются в электрической цепи, результирующий шум определяется не суммой их *амплитуд*, а суммой *мощностей* отдельных сигналов.

*Среднеквадратическое отклонение* похоже на *среднее отклонение*, за исключением того, что усредняются не отклонения амплитуды, а мощности отклонений уровней сигнала во всех точках. Это достигается при помощи возвведения в квадрат каждого отклонения и получением среднего значения (напомним, что мощность прямо пропорциональна квадрату напряжения). После нахождения среднего значения для компенсации процедуры возвведения в квадрат необходимо из полученного числа извлечь *квадратный корень*. Формула для вычисления среднеквадратического отклонения имеет вид:

$$\sigma^2 = \frac{1}{N-1} \sum_{i=0}^{N-1} (x_i - \mu)^2. \quad (2.2)$$

Вычисление среднеквадратического отклонения сигнала. Сигнал представлен отсчетами  $x_i$ ;  $\mu$  — среднее значение, вычисляемое в соответствии с выражением (2.1);  $N$  — количество отсчетов,  $\sigma$  — среднеквадратическое отклонение.

Альтернативная запись этого выражения будет иметь вид

$$\sigma = \sqrt{[(x_0 - \mu)^2 + (x_1 - \mu)^2 + \dots + (x_{N-1} - \mu)^2] / (N-1)}.$$

Заметим, что при усреднении деление производится на  $N-1$ , а не на  $N$ . Эта особенность более подробно будет обсуждаться в следующем разделе. Величина  $\sigma^2$  довольно часто встречается в статистике и называется *дисперсией*. Среднеквадратическое отклонение показывает, насколько сильно сигнал флюкутирует относительно своего среднего значения, а дисперсия показывает мощность этих флюкутаций. В электронике также довольно часто используется понятие *среднеквадратичного значения*. По определению, среднеквадратическое отклонение характеризует только переменную составляющую сигнала, а среднеквадратичное значение — и переменную, и постоянную составляющие. Если сигнал не имеет постоянной составляющей, то среднеквадратичное значение равно среднеквадратическому отклонению. На Рис. 2.2 показана взаимосвязь между среднеквадратическим отклонением и амплитудой размаха сигнала для некоторых типовых форм сигналов. Для прямоугольных импульсов отношение амплитуды размаха к среднеквадратическому отклонению равно 2; для последовательности треугольных импульсов — 3.46; для синусоиды — 2.83. Амплитуду размаха случайного сигнала (шума) точно определить невозможно, она примерно в 6...8 раз превышает значение среднеквадратического отклонения.

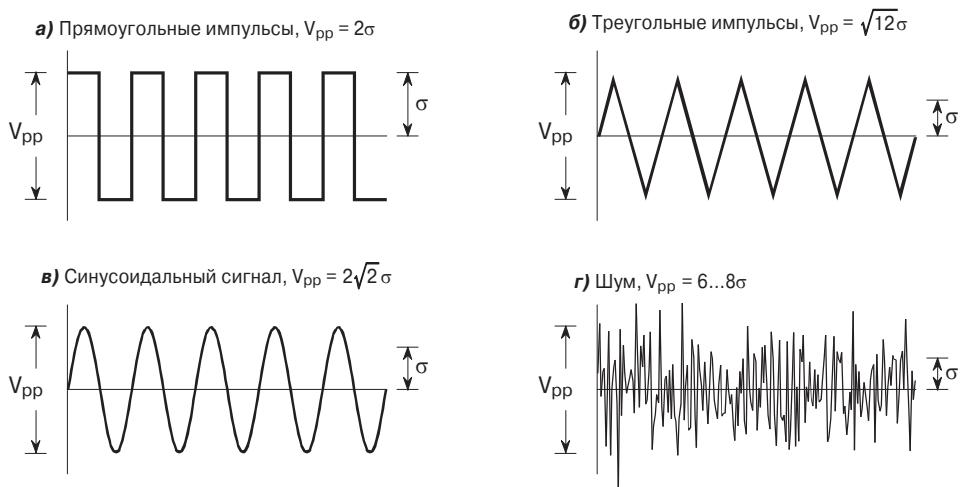


Рис. 2.2. Отношение амплитуды размаха к среднеквадратическому отклонению некоторых типовых сигналов.

**Программа 2.1** вычисляет среднее значение и среднеквадратическое отклонение согласно (2.1) и (2.2). Все программы, приведённые в этой книге, разрабатывались, исходя из принципа наилучшего соответствия алгоритму. Различные методы разработки программ, которые отвечают требованиям хорошего стиля программирования, здесь игнорируются в пользу упрощения структуры программы, делающей более понятной логику её работы. При написании программ используется упрощённая версия языка Бейсик, применяется нумерация строк, из управляющих структур задействуются только циклы вида FOR–NEXT, не применяются выражения для ввода/вывода и т. д. Такие программы следует рассматривать как альтернативный способ иллюстрации математических выражений, что позволяет лучше понять суть методов ЦОС. В языке Бейсик символ % в конце имени переменной обозначает целочисленный тип данных. Все другие переменные предназначены для описания чисел с плавающей точкой. Более подробно типы данных будут обсуждаться в Главе 4.

### Программа 2.1

```

100 'ВЫЧИСЛЕНИЕ СРЕДНЕГО ЗНАЧЕНИЯ И СРЕДНЕКВАДРАТИЧЕСКОГО ОТКЛОНЕНИЯ
110 '
120 DIM X[511] 'Значения сигнала хранятся в ячейках X[0]...X[511]
130 N% = 512 'N% - количество отсчётов сигнала
140 '
150 GOSUB XXXX 'Некоторая подпрограмма, которая загружает сигнал в массив X[ ]
160 '
170 MEAN = 0 'Вычисление среднего значения в соответствии с выражением (2.1)
180 FOR I% = 0 TO N%-1
190 MEAN = MEAN + X[I%]
200 NEXT I%
210 MEAN = MEAN/N%
220 '
230 VARIANCE = 0 'Вычисление среднеквадратического отклонения в соответствии с
 выражением (2.2)
240 FOR I% = 0 TO N%-1
250 VARIANCE = VARIANCE + ( X[I%] - MEAN )^2
260 NEXT I%
270 VARIANCE = VARIANCE/ (N%-1)
280 SD = SQR(VARIANCE)
290 '
300 PRINT MEAN SD 'Вывод среднего значения и среднеквадратического отклонения
310 '
320 END

```

Рассмотренный метод вычисления среднего значения и среднеквадратического отклонения подходит для большинства случаев, однако существует два ограничения. Во-первых, если среднее значение намного больше среднеквадратического отклонения, то выражение (2.2) будет содержать вычитание двух чисел, значения которых мало отличаются друг от друга, что может привести к большой ошибке округления. Более подробно эта проблема будет обсуждаться в Главе 4. Во-вторых, очень часто необходимо пересчитывать среднее значение и среднеквадратическое отклонение по мере поступления новых отсчётов сигнала. Этот тип вычислений мы будем называть «плавающей статистикой». В этом случае ис-

пользование выражений (2.1) и (2.2) подразумевает в каждом новом вычислении участие всех имеющихся на данный момент отсчётов сигнала, что очень неэффективно с точки зрения использования памяти и вычислительных ресурсов.

Решением этой проблемы может быть преобразование выражений (2.1) и (2.2) к новому виду, позволяющему повысить эффективность вычисления среднеквадратического отклонения:

$$\sigma^2 = \frac{1}{N-1} \left[ \sum_{i=0}^{N-1} x_i^2 - \frac{1}{N} \left( \sum_{i=0}^{N-1} x_i \right)^2 \right]$$

или в более простой форме:

$$\sigma^2 = \frac{1}{N-1} \left[ \text{сумма квадратов} - \frac{\text{сумма}^2}{N} \right]. \quad (2.3)$$

Вычисление среднеквадратического отклонения сигнала в случае плавающей статистики. Это выражение обеспечивает такой же результат, как и (2.2), но с меньшими ошибками округления и большей вычислительной эффективностью. Результат формируется с помощью трёх накопленных параметров: количества отсчётов  $N$ , суммы этих отсчётов и суммы квадратов отсчётов. По этим трём величинам вычисляются среднее значение и среднеквадратическое отклонение сигнала.

По мере приёма сигнала производится перерасчёт трёх величин: количества уже обработанных отсчётов, суммы этих отсчётов и суммы квадратов отсчётов (вычисляется квадрат каждого нового отсчёта, а его значение прибавляется к уже накопленному результату). Таким образом, среднее значение и среднеквадратическое отклонение могут быть вычислены при любом количестве поступивших отсчётов всего по трём текущим значениям параметров. **Программа 2.2** вычисляет таким способом среднее значение и среднеквадратическое отклонение сигнала по мере поступления каждого нового отсчёта. Этот метод используется, например, в карманных калькуляторах для расчёта статистических данных последовательности чисел. Каждый раз, когда вы вводите число и нажимаете кнопку «Σ» (суммирование), происходит обновление значений трёх параметров. При таком подходе среднее значение и среднеквадратическое отклонение могут быть быстро рассчитаны, так сказать, по первому требованию, без необходимости пересчёта всей последовательности.

### **Программа 2.2**

```

100 ' ВЫЧИСЛЕНИЕ СРЕДНЕГО ЗНАЧЕНИЯ И СКО В СЛУЧАЕ ПЛАВАЮЩЕЙ СТАТИСТИКИ
110 '
120 DIM X[511] 'Значения сигнала хранятся в ячейках X[0]...X[511]
130 '
140 GOSUB XXXX 'Некоторая подпрограмма, которая загружает сигнал в массив X[ ]
150 '
160 N% = 0 'Обнуление трёх текущих параметров
170 SUM = 0
180 SUMSQUARES = 0
190 '
200 FOR I% = 0 TO 511 'Цикл по каждому отсчёту сигнала
210 '
220 N% = N%+1 'Обновление значений трёх параметров

```

```

230 SUM = SUM + X[I%]
240 SUMSQUARES = SUMSQUARES + X[I%]^2
250 '
260 MEAN = SUM/N% 'Среднее значение и СКО в соответствии с (2.3)
270 IF N% = 1 THEN SD = 0: GOTO 300
280 SD = SQR( (SUMSQUARES - SUM^2/N%) / (N%-1) )
290 '
300 PRINT MEAN SD 'Вывод среднего значения и среднеквадратического отклонения
310 '
320 NEXT I%
330 '
340 END

```

И в завершение рассуждений о среднем значении и среднеквадратическом отклонении следует вспомнить ещё о двух понятиях. В некоторых задачах среднее значение выступает в качестве интересующего нас параметра, в то время как среднеквадратическое отклонение характеризует уровень шумов и помех. В таких случаях не так важно само среднеквадратическое отклонение, сколько его сравнение со средним значением. Отсюда возникло понятие *отношение сигнал/шум*, которое вычисляется делением среднего значения на среднеквадратическое отклонение. Также часто используется параметр под названием *коэффициент вариации*, который определяется обычным делением среднеквадратического отклонения на среднее значение и умножением на 100%. Например, у сигнала (или других измеряемых значений) с коэффициентом вариации 2% отношение сигнал/шум будет равно 50. Данные тем достовернее, чем выше отношение сигнал/шум и чем меньше коэффициент вариации.

## 2.3. Сигналы и процессы

*Статистика* — это наука, помогающая интерпретировать зарегистрированные числовые данные, например такие, как *случайный сигнал*. В отличие от неё теория вероятностей позволяет описывать *процессы*, в результате которых сигналы формируются. Несмотря на их тесную связь, *сигнал, принимаемый системой обработки, и процесс, сформировавший этот сигнал, всё же имеют существенные различия, и эти различия в цифровой обработке сигналов часто оказываются ключевыми*.

Например, представьте себе процесс генерации сигнала, состоящего из 1000 отсчётов, формируемого в результате подбрасывания монеты 1000 раз. Если выпадает «орёл», отсчёту сигнала присваивается значение единицы, если «решка» — нуля. Среднее значение *процесса*, который формирует сигнал, в данном случае точно равно 0.5, что определяется относительной вероятностью каждого возможного исхода: 50% — «орёл», 50% — «решка». Однако это не значит, что и среднее значение сигнала будет равно 0.5. Случайная природа событий приведёт к тому, что количество единиц и нулей будет всё время немного отличаться. Таким образом, вероятностные характеристики исходного процесса являются постоянными, а статистические характеристики принимаемого сигнала изменяются каждый раз при проведении нового эксперимента. Такая случайность, лежащая в основе реальных данных, получила название *статистической вариации, статистической флюктуации, или статистического шума*.

Здесь может возникнуть своего рода дилемма. Если в литературе встречается понятия *среднее значение* и *среднеквадратическое отклонение*, как вы узнаете, что автор имеет в виду — статистические параметры конкретного сигнала или вероятностные характеристики породившего его процесса? К сожалению, это можно выяснить только из контекста. Есть, однако, в статистике и теории вероятностей и такие понятия, которые, фактически обозначая одно и то же, имеют всё же разные названия. Например, термины *гистограмма* и *распределение вероятностей* (которые будут рассмотрены в следующем разделе), используемые соответственно в статистике и теории вероятностей, описывают совпадающие по смыслу понятия.

Теперь вернёмся к выражению (2.2), используемому для расчёта среднеквадратического отклонения. Как уже отмечалось, в этой формуле (для случая расчёта квадрата СКО) в знаменателе стоит  $(N - 1)$ , а не просто  $N$ . Для того чтобы понять, почему это так, давайте представим, что мы хотим найти среднее значение и среднеквадратическое отклонение некоторого *процесса*. Чтобы это сделать, необходимо взять одну реализацию этого процесса (один сигнал, сформированный процессом) длиной  $N$  отсчётов и вычислить среднее значение в соответствии с выражением (2.1). После этого можно использовать полученное число в качестве *оценки* среднего значения процесса, не забывая о том, что это значение будет содержать ошибку — *статистический шум*. В частности, для случайных сигналов типовая ошибка между средним значением сигнала, состоящего из  $N$  отсчётов, и средним значением исходного процесса определяется по формуле

$$\text{Типовая ошибка} = \frac{\sigma}{N^{1/2}}. \quad (2.4)$$

Типовая ошибка вычисления среднего значения процесса по ограниченному числу  $N$  отсчётов одной реализации. Параметр  $\sigma$  представляет собой среднеквадратическое отклонение.

Если число  $N$  мало, то значение статистического шума в вычислении среднего значения окажется достаточно большим. Другими словами, для достоверного описания процесса необходимо иметь большое количество данных. С увеличением числа  $N$  величина ошибки уменьшается. Один из основополагающих законов теории вероятностей — *закон больших чисел* — позволяет утверждать, что при  $N$ , приближающемуся к бесконечности, значение этой ошибки будет стремиться к нулю.

Рассчитаем теперь СКО принятого сигнала и используем его для оценки СКО исходного процесса. Здесь нас ожидает новая проблема. Перед тем как вычислить СКО в соответствии с выражением (2.2), необходимо знать среднее значение  $\mu$ . Однако нам не известно среднее значение процесса, а только его оценка, полученная по  $N$  отсчётам сигнала и содержащая ошибку из-за статистического шума. Такая ошибка приводит к уменьшению вычисляемого значения СКО. Для компенсации этого эффекта число  $N$  заменяется на  $(N - 1)$ . В результате если  $N$  достаточно большое, то разница в вычислениях от такой замены несущественна, а при малых  $N$  это обеспечивает более точную оценку СКО исходного процесса. Другими словами, выражение (2.2) описывает *оценку СКО исходного процесса*. Если в этом выражении в знаменателе вместо  $(N - 1)$  использовать число  $N$ , то мы получим выражение для СКО *принимаемого сигнала*.

В качестве иллюстрации таких рассуждений рассмотрим сигналы, изображённые на Рис. 2.3. Зададимся вопросом: являются ли изменения в этих сигналах результатом действия статистического шума, или они вызваны изменениями характеристик исходного процесса? Думаю, будет несложно убедить себя, что эти изменения слишком велики, чтобы быть случайными, и скорее всего они должны быть связаны с изменениями во времени вероятностных характеристик исходно-

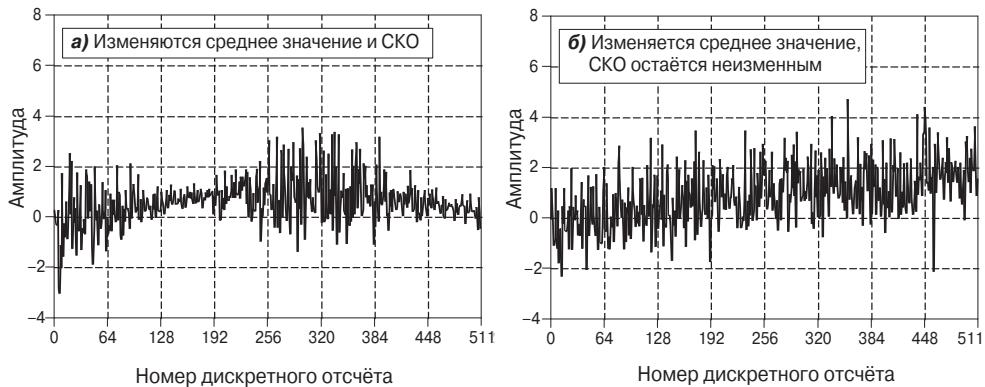


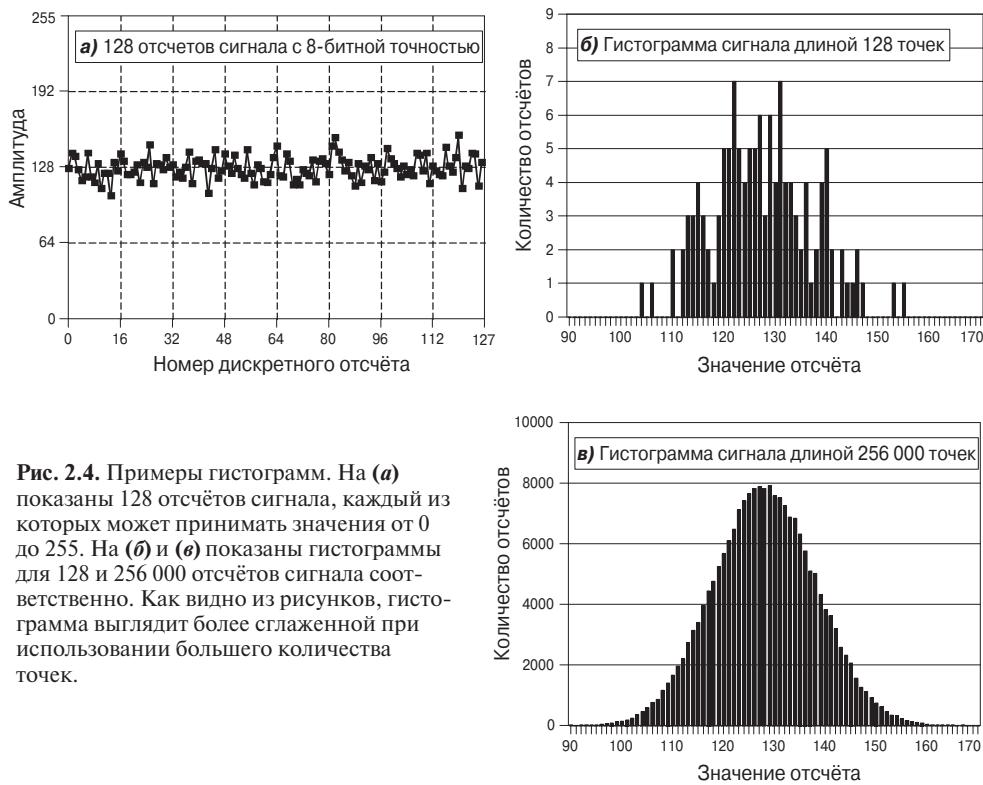
Рис. 2.3. Примеры сигналов, генерируемых нестационарными процессами.

го процесса. Такие процессы называются *нестационарными*. Для сравнения: сигналы, представленные на Рис. 2.1, генерированы *стационарным процессом*, а изменения в поведении сигналов являются результатом проявления статистического шума. На Рис. 2.3а изменяются и среднее значение, и СКО процесса. На (б) СКО остаётся неизменным и равно единице, а среднее значение изменяется от нуля до двух. На (б) проиллюстрирована общая для нестационарных сигналов проблема: медленное изменение *среднего значения* влияет на точность вычисления *среднеквадратического отклонения*. В этом примере СКО сигнала на небольшом интервале равно 1, хотя СКО всего сигнала составляет 1.16. Эта ошибка может быть почти полностью исключена путём разбиения сигнала на короткие сегменты и вычисления статистических показателей индивидуально для каждого фрагмента. При необходимости можно получить СКО всего сигнала путём усреднения среднеквадратических отклонений, рассчитанных для каждого фрагмента независимо.

## 2.4. Гистограмма, распределение вероятностей, функция плотности вероятности

Давайте предположим, что мы подключили 8-битный аналогово-цифровой преобразователь к компьютеру и получили с его помощью 256 000 отсчётов некоторого сигнала. Рис. 2.4а показывает 128 значений, являющихся частью этого сигнала. Значение каждого отсчёта может принимать одно из 256 возможных зна-

чений от 0 до 255. Гистограмма отображает количество присутствующих в сигнале отсчётов с заданным значением. На (б) показана гистограмма, соответствующая фрагменту сигнала на (а). Она показывает, что имеется, например, два отсчёта со значением 110, семь — со значением 131, ноль — со значением 170 и т. д. Для обозначения гистограммы мы будем использовать символ  $H_i$ , где индекс  $i$  меняется в диапазоне от 0 до  $(M - 1)$ ,  $M$  — количество возможных значений сигнала. Например,  $H_{50}$  обозначает число отсчётов сигнала, принимающих значение 50. На (в) показана гистограмма для полного набора данных, состоящего в нашем случае из 256 000 отсчётов. Можно отметить, что рассмотрение большего числа отсчётов сигнала даёт гистограмму более сглаженного вида. Так же как и в случае средних значений, величина статистического шума гистограммы обратно пропорциональна корню квадратному из количества используемых отсчётов.



**Рис. 2.4.** Примеры гистограмм. На (а) показаны 128 отсчётов сигнала, каждый из которых может принимать значения от 0 до 255. На (б) и (в) показаны гистограммы для 128 и 256 000 отсчётов сигнала соответственно. Как видно из рисунков, гистограмма выглядит более сглаженной при использовании большего количества точек.

Исходя из определения понятия гистограммы, сумма всех её значений должна быть равна длине сигнала:

$$N = \sum_{i=0}^{M-1} H_i. \quad (2.5)$$

Сумма всех значений гистограммы равна длине сигнала. В этом выражении  $H_i$  описывает гистограмму,  $N$  — это количество отсчётов в сигнале (его длина),  $M$  — число значений, которое может принимать сигнал.

Гистограмма может использоваться для эффективного вычисления среднего значения и СКО очень больших наборов данных, что весьма важно в первую очередь при обработке изображений, часто содержащих миллионы отсчётов. Гистограмма также позволяет производить вычисление статистических данных, опирая несколькими группами отсчётов с одинаковыми значениями, а не большим количеством отдельных отсчётов. В этом случае среднее значение и СКО могут быть получены на основе следующих выражений:

$$\mu = \frac{1}{N} \sum_{i=0}^{M-1} i \cdot H_i, \quad (2.6)$$

Вычисление среднего значения с помощью гистограммы. Это выражение можно рассматривать как объединение в группы всех отсчётов, имеющих одинаковые значения с последующим применением выражения (2.1) для каждой группы.

$$\sigma^2 = \frac{1}{N-1} \sum_{i=0}^{M-1} (i - \mu)^2 H_i. \quad (2.7)$$

Вычисление СКО с помощью гистограммы. Подход к вычислению СКО в этом выражении схож с выражением (2.2), однако здесь сразу обрабатываются все отсчёты, имеющие одинаковые значения, объединённые в группы.

**Программа 2.3** использует эти выражения для расчёта гистограммы, среднего значения и СКО. Расчёт гистограммы очень быстрая процедура, требующая использования лишь операций индексирования (изменения индексов) и инкремента. Вычисление же среднего значения и СКО требует более трудоёмких операций сложения и умножения. При разработке программы использована следующая стратегия: медленно выполняемые операции сложения и умножения используются только для работы с некоторыми точками гистограммы, а не с большим количеством отсчётов сигнала. Такой подход позволяет сделать алгоритм намного более быстрым по сравнению с рассмотренными ранее методами. (Для очень длинных сигналов выигрыш во времени вычисления при реализации на обычных персональных компьютерах может достигать десяти раз.)

### Программа 2.3

```

100 'ВЫЧИСЛЕНИЕ ГИСТОГРАММЫ, СРЕДНЕГО ЗНАЧЕНИЯ И СКО
110 '
120 DIM X%[25000] 'Ячейки с X%[0] по X%[25000] хранят значения обрабатываемого
   сигнала
130 DIM H%[255] ' В ячейки с H%[0] по H%[255] записываются значения гистограммы
140 N% = 25001 'Количество точек в сигнале
150 '
160 FOR I% = 0 TO 255 'Обнуление ячеек гистограммы, необходимое для накопления
170 H%[I%] = 0
180 NEXT I%
190 '
200 GOSUB XXXX 'Некоторая подпрограмма, которая загружает сигнал в массив X%[ ]
210 '
220 FOR I% = 0 TO 25000 'Вычисление гистограммы для 25001 точек
230 H%[ X%[I%] ] = H%[ X%[I%] ] + 1
240 NEXT I%
```

```

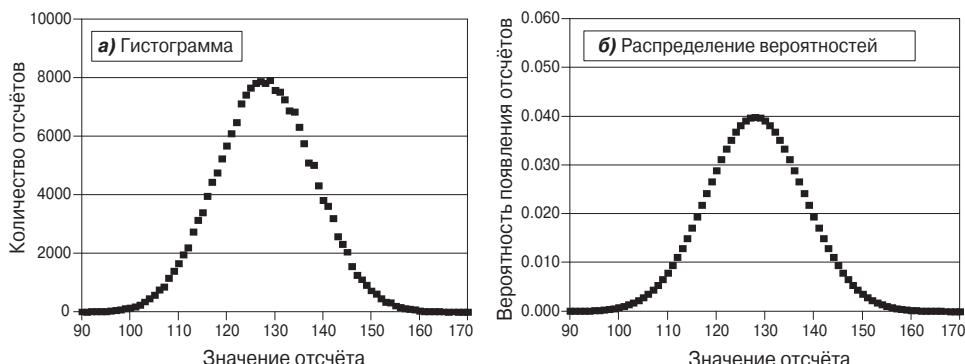
250 '
260 MEAN = 0 'Вычисление среднего значения в соответствии с выражением (2.6)
270 FOR I% = 0 TO 255
280 MEAN = MEAN + I% * H%[I%]
290 NEXT I%
300 MEAN = MEAN / N%
310 '
320 VARIANCE = 0 'Вычисление СКО в соответствии с выражением (2.7)
330 FOR I% = 0 TO 255
340 VARIANCE = VARIANCE + H%[I%] * (I%-MEAN)^2
350 NEXT I%
360 VARIANCE = VARIANCE / (N%-1)
370 SD = SQR(VARIANCE)
380 '
390 PRINT MEAN SD 'Вывод среднего значения и СКО
400 '
410 END

```

То, что регистрируемый нами *сигнал* всегда является лишь одной реализацией породившего его *процесса* и включает случайные ошибки, является очень важным фактом, настолько важным, что некоторые понятия, относящиеся к сигналу и к процессу, имеют даже разные названия. Так, *гистограмма* является характеристикой принятого сигнала, а аналогичная ей характеристика исходного процесса называется *распределением вероятностей*. Гистограмма всегда вычисляется с использованием конечного числа отсчётов, а распределение вероятностей может быть получено на основе бесконечного количества значений. Распределение вероятностей можно приблизительно рассчитать на основании гистограммы или получить с помощью некоторых математических приёмов, таких, как в примере с подбрасыванием монеты.

На Рис. 2.5 показан пример распределения вероятностей и одна из гистограмм, которая может ей соответствовать. Для нас ключевой информацией здесь являются единицы, отложенные по вертикальной оси. В случае гистограммы, как говорилось выше, по вертикальной оси откладывается число, показывающее, сколько раз то или иное значение встречается в сигнале. В случае распределения вероятностей по вертикальной оси откладывается по сути та же информация, но выраженная отношением количества заданных значений сигнала к общему числу его отсчётов. То есть для аппроксимации распределения вероятностей необходимо каждое значение гистограммы разделить на общее число отсчётов в сигнале. Это означает, что каждое значение распределения вероятностей должно лежать в диапазоне от нуля до единицы, а сумма всех значений должна быть равна единице.

Распределение вероятностей имеет большое значение: с её помощью определяется *вероятность* появления в сигнале отсчёта с тем или иным значением. Рассмотрим, например, сигнал, изображённый на Рис. 2.4а, с распределением вероятностей, показанным на Рис. 2.5б. Какова вероятность того, что случайно взятый отсчёт этого сигнала будет иметь значение 120? Ответ может дать Рис. 2.5б. Вероятность равна 0.03, или примерно 1 шанс из 34. А какова вероятность того, что случайно выбранный отсчёт будет иметь значение большее, чем 150? Сложение значений распределения вероятностей для номеров отсчётов 151, 152, 153, ..., 255 даёт ответ на этот вопрос: 0.0122, или примерно 1 шанс из 82. Та-



**Рис. 2.5.** Взаимосвязь гистограммы (а), распределения вероятностей (б) и функции плотности вероятности (в). Гистограмма вычисляется по конечному числу отсчётов одной реализации процесса. Распределение вероятностей характеризует сам исходный процесс, а функция плотности вероятности подобна распределению вероятностей, но используется в основном для описания непрерывных сигналов. Одинаковые значения на вертикальных осях (от 0 до 0.06) на (б) и (в) — это всего лишь совпадение в данном примере. Амплитуды трёх приведённых кривых определяются с помощью:

**а)** суммы значений гистограммы, равной количеству отсчётов в сигнале; **б)** суммы значений распределения вероятностей, равной единице; **в)** площади под кривой плотности вероятности, равной единице.

ким образом, ожидается, что сигнал будет иметь значение больше, чем 150, примерно через каждые 82 точки. Какова вероятность того, что значение любого отсчёта будет располагаться в диапазоне от 0 до 255? Если сложить все значения распределения вероятностей, то вероятность такого исхода составит 1, т. е. это событие обязательно произойдёт.

Гистограмма и распределение вероятностей могут использоваться только для дискретных данных, таких, как оцифрованные сигналы, хранящиеся в компьютере. Однако аналогичные понятия, безусловно, существуют и для непрерывных сигналов, с которыми работает аналоговая электроника. *Функция плотности вероятности*, или *функция распределения вероятности*, для аналоговых сигналов имеет такой же смысл, что и распределение вероятностей для дискретных сигналов. Например, представьте себе аналоговый сигнал, который, пройдя через аналого-цифровой преобразователь, становится цифровым сигналом, изображённым на Рис. 2.4а. Допустим, напряжение амплитудой от 0 до 255 мВ после оцифровки преобразуется в диапазон уровней цифрового сигнала от 0 до 255. Распределение вероятностей этого цифрового сигнала показано на Рис. 2.5б как совокупность отдельных дискретных точек. Функция плотности вероятности аналогового сигнала изображена на Рис. 2.5в. Эта кривая является непрерывной, что говорит о способности сигнала принимать непрерывный диапазон значений так, как это свойственно напряжению в электрической цепи.

По вертикальной оси в случае функции плотности вероятности откладывается не просто вероятность, а именно *плотность вероятности*. Например, плотность вероятности 0.3 в точке 120.5 не означает, что напряжение в 120.5 милливольт встретится в 3% случаев. На самом деле вероятность того, что значение непрерывного сигнала окажется точно равным 120.5 милливольта, ничтожно мала. Ведь существует бесконечное число возможных значений, расположенных в окрестности этой точки: 120.49997, 120.49998, 120.49999 и т. д. Шансов, что значение сигнала составит именно 120.50000, практически нет.

Чтобы рассчитать вероятность появления отсчёта со значением, лежащим в некотором диапазоне, необходимо функцию плотности вероятности умножить на этот диапазон значений. Например, для представленного рисунка вероятность того, что сигнал примет значение 120...121, будет равна:  $(121 - 120) \times 0.03 = 0.03$ , а вероятность того, что амплитуда сигнала будет находиться между значениями 120.4 и 120.5, оказывается равна:  $(120.5 - 120.4) \times 0.03 = 0.003$ . Если в рассматриваемом диапазоне плотность вероятности непостоянна, то умножение следует заменить интегрированием плотности вероятности на требуемом интервале, т. е. вероятность будет равна площади под кривой плотности вероятности, ограниченной заданными значениями. Общее пространство под кривой плотности вероятности, т. е. значение интеграла  $-\infty \dots +\infty$ , всегда будет равно единице, аналогично тому, как сумма всех значений распределения вероятностей равна единице, а сумма всех значений гистограммы равна  $N$ .

Гистограмма, распределение вероятностей и функция плотности вероятности представляют собой три очень похожих понятия. Математики всегда различают их, но многие учёные и инженеры, как вы ещё не раз убедитесь, часто подменяют эти понятия и соответственно неправильно их используют. На Рис. 2.6 изображены три непрерывных сигнала и их функции плотности вероятности. График функции плотности вероятности обычно изображают рядом с графиком сигнала, который она описывает, и располагают его под углом  $90^\circ$ . Функция плотности вероятности прямоугольного колебания, приведённая на (а), состоит из двух бесконечно узких пиков, что соответствует сигналу с двумя возможными значениями. Функция плотности вероятности треугольного колебания (б) имеет постоянное значение во всем диапазоне возможных значений сигнала и часто называется *равномерным распределением*. Функция плотности вероятности случайного сигнала (шума), изображённая на (в), наиболее интересна. Она представляет собой колоколообразную кривую, которая называется гауссовой. Если бы данные сигналы были дискретными, то в этом случае горизонтальная ось называлась бы «номер отсчёта» и для таких сигналов использовалось бы распределение вероятностей.

Расчёт гистограммы может оказаться затруднителен, если число значений, которые может принимать каждый отсчёт сигнала, значительно превышает число самих отсчётов в сигнале. Подобная ситуация всегда возникает с сигналами, представленными в *формате с плавающей точкой*, т. е. когда каждый отсчёт сигнала представляется в виде вещественного числа. При целочисленном представлении данных значение отсчёта может быть равно или 3, или 4, а плавающая точка позволяет иметь миллионы возможных дробных значений между 3 и 4. Рассмотренный ранее метод вычисления гистограммы основывался на подсчёте количества отсчётов на каждом из возможных уровней квантования, что невозможно для

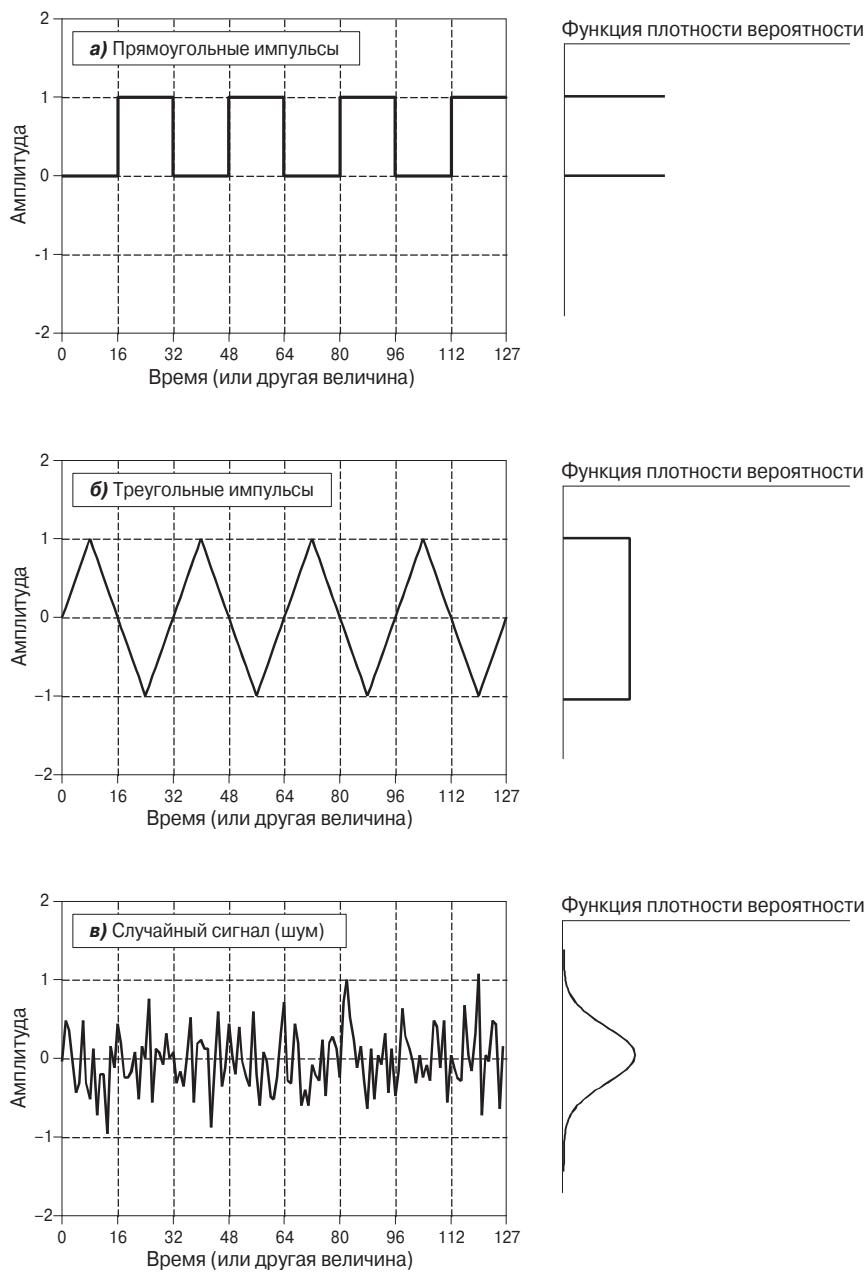


Рис. 2.6. Три типа распространённых сигналов и их функции плотности вероятности.

данных с плавающей точкой, так как пришлось бы иметь дело с миллиардами значений. Ещё хуже то, что почти все эти уровни не будут содержать точно соответствующих им отсчётов. Представьте себе сигнал длиной 10 000 отсчётов, каждый из которых может принимать миллиард возможных значений. Соответствующая гистограмма будет включать миллиард точек, только 10 000 из которых могут быть неравными нулю.

Уйти от указанной проблемы можно, выбирая длину гистограммы произвольным образом, так, чтобы она составляла приемлемое число, например 1000, а в каждом значении гистограммы учитывать все отсчёты, попадающие в некоторый диапазон. Возьмём, например, сигнал, представленный в формате с плавающей точкой, значения которого лежат в диапазоне 0.0...10.0. В качестве длины гистограммы выберем 1000 значений. Описанный подход позволяет нам вычислить нулевой отсчёт гистограммы как число отсчётов сигнала, значения которых лежат в диапазоне 0...0.01; первый отсчёт — как число отсчётов сигнала в диапазоне 0.01...0.02, и так далее до 999-го отсчёта гистограммы, содержащего число отсчётов сигнала со значением 9.99...10.0. В **Программе 2.4** рассчитываются гистограммы с использованием описанного метода.

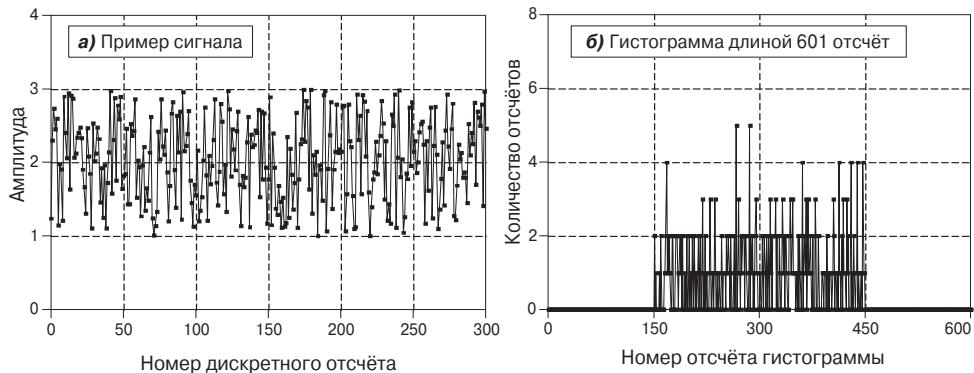
### **Программа 2.4**

```

100 'РАСЧЁТ ГИСТОГРАММЫ ДЛЯ ДАННЫХ В ФОРМАТЕ С ПЛАВАЮЩЕЙ ТОЧКОЙ
110 '
120 DIM X[25000] 'Ячейки X[0]...X[25000] хранят значения отсчётов сигнала,
130 ' Величина каждого отсчёта находится в пределах 0.0...10.0.
140 DIM H%[999] 'Ячейки H%[0]...H%[999] предназначены для хранения отсчётов
гистограммы
150 '
160 FOR I% = 0 TO 999 'Обнуление ячеек, выделенных под отсчёты гистограммы,
необходимое для использования их в качестве накопителей
170 H%[I%] = 0
180 NEXT I%
190 '
200 GOSUB XXXX Некоторая подпрограмма, загружающая сигнал в массив X%[ ]
210 '
220 FOR I% = 0 TO 25000 ' 'Вычисление гистограммы для 25001 точки
230 BINNUM% = INT( X[I%] * 100 )
240 H%[ BINNUM% ] = H%[ BINNUM% ] + 1
250 NEXT I%
260 '
270 END

```

Однако каким же значением следует ограничивать длину гистограммы? Для решения этого вопроса следует найти компромисс между двумя факторами. Как видно из **Рис. 2.7**, слишком большое количество отсчётов гистограммы затрудняет оценку уровня исходного распределения вероятностей, так как на каждый отсчёт гистограммы приходится малое число отсчётов сигнала, что приводит к повышению уровня статистического шума. С другой стороны, слишком маленькая длина гистограммы ухудшает разрешение исходной кривой распределения вероятностей по горизонтальной оси. Таким образом, задача выбора длины гистограммы сводится к подбору оптимального соотношения между разрешением по оси *Y* и разрешением по оси *X*.



**Рис. 2.7.** Примеры гистограмм, построенных с ограничением числа отсчётов. Как видно из (а) длина сигнала равна 300 отсчётам, значения которых равномерно распределены в диапазоне от 1 до 3 и представлены в формате с плавающей точкой. На (б) и (в) показаны гистограммы этого сигнала, составленные с ограничением их длины. Можно видеть, что большая длина гистограммы характеризуется низким разрешением по вертикальной оси, а малая длина — низким разрешением по горизонтальной оси. Использование большего числа отсчётов сигнала способно повысить разрешение и по вертикали, и по горизонтали.

## 2.5. Нормальное распределение

Сигналу, сформированному *случайным процессом*, как правило, свойственна колоколообразная функция плотности вероятности. Она получила название *нормального распределения*, или *кривой Гаусса*, — в честь великого немецкого математика Карла Фридриха Гаусса (1771–1855). Причины, по которым гауссово распределение часто встречается в природе, кратко будут рассмотрены при обсуждении *генерации цифрового шума*. Кривая Гаусса рассчитывается на основе следующего базового выражения:

$$y(x) = e^{-x^2}.$$

Из этой исходной кривой можно сформировать кривую Гаусса с любыми требуемыми параметрами, дополняя выражение величинами *среднего значения*  $\mu$  и *среднеквадратического отклонения*  $\sigma$ . Кроме того, необходимо так провести нормировку, чтобы общая площадь под кривой была равна единице. Это — требование для всех функций плотности вероятности. В результате мы получим типовую

форму нормального распределения, которое является одним из самых важных отношений в статистике и теории вероятностей:

$$P(x) = \frac{1}{\sqrt{2\pi} \cdot \sigma} e^{-(x-\mu)^2 / 2\sigma^2}. \quad (2.8)$$

Нормальное распределение.  $P(x)$  — функция плотности вероятности,  $\mu$  — среднее значение,  $\sigma$  — среднеквадратическое отклонение.

На Рис. 2.8 показано несколько примеров гауссовых кривых с разными средними значениями и среднеквадратическими отклонениями. Среднее значение центрирует кривую относительно определённого значения, а среднеквадратическое отклонение определяет ширину колокола.

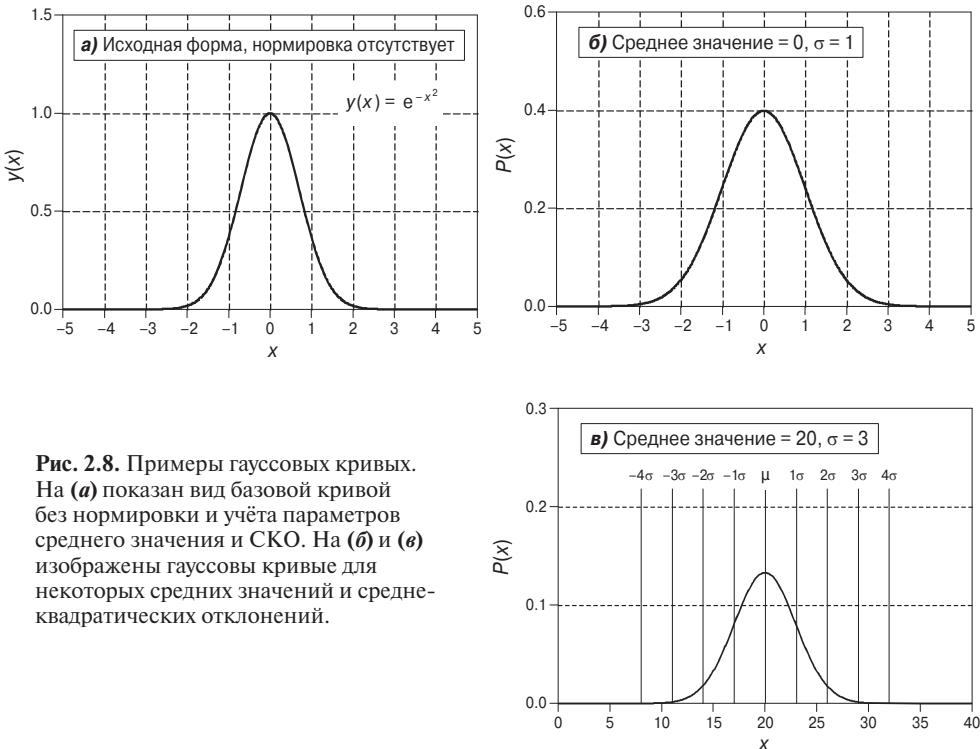


Рис. 2.8. Примеры гауссовых кривых. На (а) показан вид базовой кривой без нормировки и учёта параметров среднего значения и СКО. На (б) и (в) изображены гауссовые кривые для некоторых средних значений и среднеквадратических отклонений.

Интересно, что у гауссовой кривой «хвосты» слева и справа очень быстро спадают к нулю, значительно быстрее, чем у других обычных функций, таких, как убывающая экспонента или функция  $1/x$ . Например, в точках, которые отстоят от среднего значения на два, четыре и шесть  $\sigma$ , значение кривой Гаусса падает примерно до  $1/19$ ,  $1/7563$  и  $1/66666666$  соответственно. Вследствие этого, как видно из Рис. 2.6в, нормально распределённые сигналы, как правило, характеризуются ограниченным диапазоном значений амплитуды. В принципе их амплитуда мо-

жет стремиться и к бесконечности, но на практике экстремальные ситуации подобного рода почти не встречаются из-за резкого спада гауссовой функции распределения вероятностей. В результате форма таких сигналов выглядит ограниченной уровнем порядка 6...8 $\sigma$ .

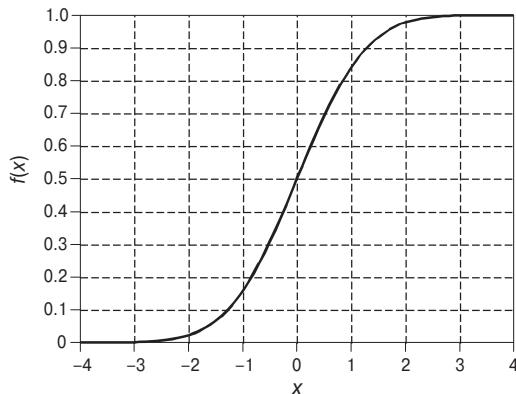
Как уже отмечалось, для нахождения вероятности того, что сигнал будет находиться в определённом диапазоне значений, используется интеграл функции плотности вероятности. По этой причине интеграл плотности вероятности представляет собой настолько важное понятие, что ему присвоили собственное название — *интегральная функция распределения*, или просто *функция распределения*. Однако гауссова кривая не может быть проинтегрирована с помощью обычных методов. Для неё необходимо использовать численное интегрирование. Оно сводится к квантованию непрерывной гауссовой кривой с малым шагом, дающим приблизительно несколько миллионов точек в диапазоне  $-10\sigma\dots+10\sigma$ , и к последующему сложению значений всех отсчётов полученного дискретного сигнала. Дискретная кривая Гаусса, рассчитанная указанным образом, сохраняется в виде таблицы и впоследствии используется для вычисления вероятностей.

На Рис. 2.9 показана интегральная функция нормального распределения, а её числовые значения приведены в Табл. 2.1. Данная кривая очень часто используется в теории вероятностей, и для её обозначения существует специальный символ  $\Phi(x)$  (большая греческая буква *фи*). Например,  $\Phi(-2)$  имеет значение 0.0228. Это означает следующее: вероятность того, что в произвольный момент времени значение сигнала будет находиться в диапазоне от  $-\infty$  до  $(\mu - 2\sigma)$ , составляет 2.28%. Аналогично значение  $\Phi(1) = 0.8413$  означает, что вероятность того, что в любой момент времени сигнал будет находиться между  $-\infty$  и  $(\mu + \sigma)$ , равна 84.13%. Для вычисления вероятности того, что сигнал будет находиться между двумя значениями, нужно вычесть соответствующие числа, которые находятся в таблице  $\Phi(x)$ . Например, вероятность того, что значение сигнала в любой момент времени будет между  $(\mu - 2\sigma)$  и  $(\mu + \sigma)$ , определяется как:  $\Phi(1) - \Phi(-2) = 0.8185$ , или 81.85%. На основании такого подхода можно утверждать, что сигнал с нормальным распределением приблизительно в 68% случаев принимает значения в диапазоне  $\pm 1\sigma$ ; в 95% случаев — значения в диапазоне  $\pm 2\sigma$ ; в 99.75% случаев — значения в диапазоне  $\pm 3\sigma$ . Вероятность равенства уровня сигнала значению, в 10 раз превышающему СКО (10 $\sigma$ ) относительно среднего значения, так ничтожна, что её можно сопоставить с длительностью в несколько микросекунд по сравнению со временем существования Вселенной (уже около 10 миллиардов лет!).

Значения, приведённые в Табл. 2.1, получены с помощью численного интегрирования нормального распределения, показанного на Рис. 2.8б. То есть  $\Phi(x)$  представляет собой вероятность того, что значение нормально распределённого сигнала в любой случайно выбранный момент времени будет меньше  $x$ . Значения  $x$  в данной таблице выражены с помощью числа СКО относительно среднего значения.

Выражение (2.8) также может использоваться для вычисления распределения вероятностей нормально распределённых дискретных сигналов. В этом случае в качестве аргумента  $x$  будет выступать значение одного из тех уровней квантования, которые может принять сигнал, т. е., например, одно из 4096 двоичных значений, появляющихся на выходе 12-битного аналогово-цифрового преобразователя. При этом можно пренебречь сомножителем  $1/\sqrt{2\pi}\sigma$ , так как он используется только для того, чтобы сделать значение общей площади под кри-

вой функции плотности вероятности равным *единице*. Вместо него необходимо использовать другое значение, делающее сумму всех значений распределения вероятностей равной *единице*. В большинстве случаев этот процесс выполняется следующим образом: строится кривая без учёта нормировки, производится суммирование всех ненормированных отсчётов, и значения всех отсчётов делят на полученное значение суммы.



**Рис. 2.9.**  $\Phi(x)$  — интегральная функция нормального распределения (среднее значение = 0, СКО = 1).

**Таблица 2.1. Числовые значения интегральной функции нормального распределения  $\Phi(x)$**

x	$\Phi(x)$	x	$\Phi(x)$	x	$\Phi(x)$	x	$\Phi(x)$
-3.4	0.0003	0.0	0.5000	-1.6	0.0548	1.8	0.9641
-3.3	0.0005	0.1	0.5398	-1.5	0.0668	1.9	0.9713
-3.2	0.0007	0.2	0.5793	-1.4	0.0808	2.0	0.9772
-3.1	0.0010	0.3	0.6179	-1.3	0.0968	2.1	0.9821
-3.0	0.0013	0.4	0.6554	-1.2	0.1151	2.2	0.9861
-2.9	0.0019	0.5	0.6915	-1.1	0.1357	2.3	0.9893
-2.8	0.0026	0.6	0.7257	-1.0	0.1587	2.4	0.9918
-2.7	0.0035	0.7	0.7580	-0.9	0.1841	2.5	0.9938
-2.6	0.0047	0.8	0.7881	-0.8	0.2119	2.6	0.9953
-2.5	0.0062	0.9	0.8159	-0.7	0.2420	2.7	0.9965
-2.4	0.0082	1.0	0.8413	-0.6	0.2743	2.8	0.9974
-2.3	0.0107	1.1	0.8643	-0.5	0.3085	2.9	0.9981
-2.2	0.0139	1.2	0.8849	-0.4	0.3446	3.0	0.9987
-2.1	0.0179	1.3	0.9032	-0.3	0.3821	3.1	0.9990
-2.0	0.0228	1.4	0.9192	-0.2	0.4207	3.2	0.9993
-1.9	0.0287	1.5	0.9332	-0.1	0.4602	3.3	0.9995
-1.8	0.0359	1.6	0.9452	0.0	0.5000	3.4	0.9997
-1.7	0.0446	1.7	0.9554				

## 2.6. Генерация цифрового шума

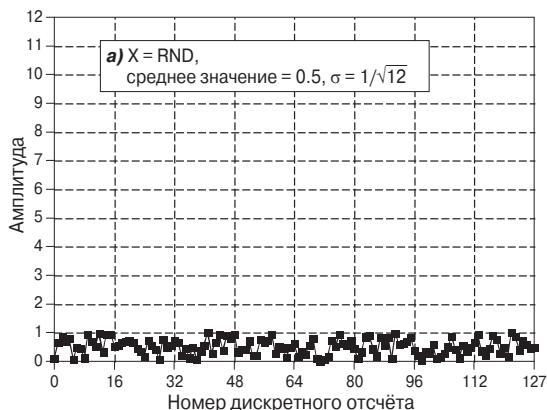
Случайные шумы являются очень важным типом сигналов как для сферы аналоговой электроники, так и для ЦОС. В частности, именно шум обычно определяет требования к минимальному уровню сигнала в системе, расстояние, на которое может быть осуществлена передача данных, или количество радиации, требуемое для получения рентгеновского снимка, и т. д. В цифровой обработке сигналов существует потребность в генерации шумов различных типов. Такая необходимость возникает при тестировании алгоритмов, которые должны оставаться работоспособными в присутствии шума заданного уровня.

Генерация цифрового шума строится на базе *генератора случайных чисел*. В большинстве языков программирования для выполнения этой задачи предусмотрена специальная функция. Так, в языке Бейсик выражение вида  $X = RND$  производит присвоение переменной  $X$  случайного значения при каждом новом выполнении такой команды. Это случайное значение будет находиться в диапазоне от нуля до единицы и иметь равную вероятность появления в пределах этих границ. На Рис. 2.10 $a$  показан сигнал, сформированный с помощью такого генератора случайных чисел для случая 128 отсчётов. Среднее значение исходного процесса при генерации данного сигнала составляет 0.5, а СКО равно  $1/\sqrt{2\pi} = 0.29$ . Распределение является равномерным в диапазоне от нуля до единицы.

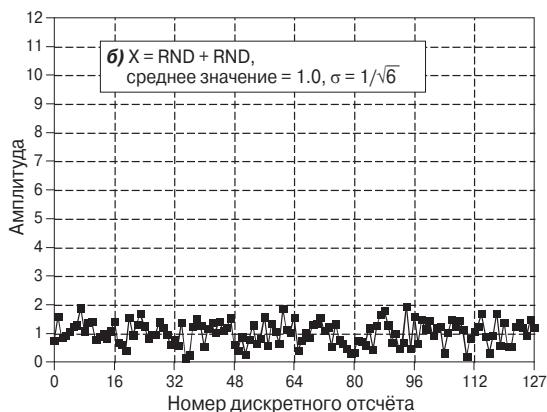
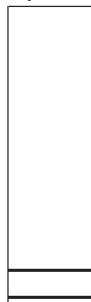
В процессе тестирования алгоритмов обычно используют сигналы того же типа, что будут встречаться в процессе реальной работы. Поэтому требуется генерировать цифровой шум не с равномерной, а с гауссовой функцией плотности вероятности. Существуют два метода формирования подобных сигналов с использованием описанного генератора случайных чисел. На Рис. 2.10 проиллюстрирован первый метод. На (б) изображён сигнал, полученный с помощью сложения двух случайных чисел для формирования каждого отсчёта, т. е.  $X = RND + RND$ . Так как каждое случайное число может принимать значение от нуля до единицы, то их сумма уже может находиться в пределах от нуля до двух. Среднее значение в данном случае будет равно единице, а СКО составит  $1/\sqrt{6}$  (запомните, что при сложении независимых случайных сигналов их *дисперсии* также складываются). При этом, как видно из графика функции плотности вероятности, равномерное распределение изменилось на треугольное. Объяснить это явление можно тем, что большинство значений сигнала теперь находится в области единицы, а меньшая часть отсчётов приближается к нулю и двум.

Далее перейдём к формированию каждого отсчёта сигнала как суммы 12 случайных чисел. В результате получим ещё большие изменения (в). Среднее значение стало равным *шести*, а СКО — *единице*. При этом самое главное то, что функция плотности вероятности стала практически гауссовой. Описанная процедура может использоваться для создания нормально распределённого шумового сигнала с произвольным требуемым средним значением и СКО. Для этого, рассчитывая каждый отсчёт сигнала, необходимо: 1) сложить 12 случайных чисел; 2) из полученной суммы вычесть шесть, делая среднее значение равным нулю; 3) умножить значение отсчёта на величину желаемого СКО; 4) добавить требуемое среднее значение.

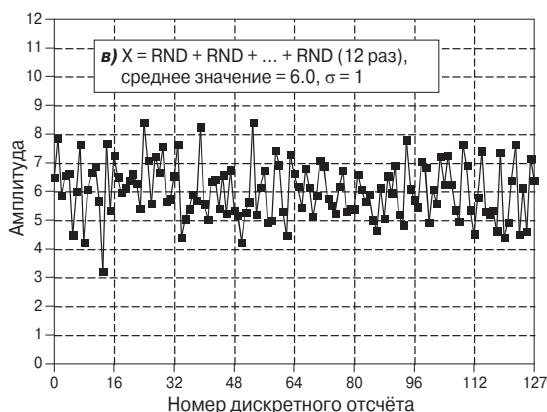
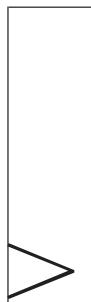
Математическое подтверждение справедливости описанной процедуры основывается на *центральной предельной теореме*, которая является одной из важней-



Функция плотности вероятности



Функция плотности вероятности



Функция плотности вероятности



**Рис. 2.10.** Генерация сигнала с нормальным распределением на основе генератора случайных чисел с равномерным распределением. На (а) показан сигнал, каждый отсчет которого формируется с помощью генератора случайных чисел. График функции плотности вероятности этого сигнала показывает, что значение каждого его отсчета имеет равномерное распределение в диапазоне от нуля до единицы. На (б) показан сигнал, каждый отсчет которого формируется как сумма двух значений, поступающих с генератора случайных чисел, а на (в) — сигнал, каждый отсчет которого равен сумме двенадцати значений с генератора случайных чисел. Функция плотности вероятности, характерная для последнего сигнала (в), имеет очень близкую к кривой Гаусса форму со средним значением, равным шести, и СКО, равным единице.

ших в теории вероятностей. В самом простом изложении теорема гласит, что *сумма* случайных чисел становится *нормально распределённой* с увеличением количества суммируемых элементов. При этом нет необходимости в том, чтобы отдельные случайные элементы были распределены по какому-то определённому закону или даже чтобы эти случайные числа имели один и тот же закон распределения. Центральная предельная теорема даёт объяснение факту широкого распространения нормально распределённых сигналов, в том числе в цифровой обработке сигналов. В реальном окружающем нас мире существует огромное множество различных сил, действующих на систему, в результате чего итоговая функция плотности вероятности такого совокупного влияния оказывается гауссовой.

Второй метод генерации нормально распределённого сигнала предлагает дважды запускать генератор случайных чисел, формируя два случайных числа —  $R_1$  и  $R_2$ . После этого может быть найдено нормально распределённое случайное число  $X$  согласно выражению

$$X = (-2 \log R_1)^{1/2} \cos(2\pi R_2). \quad (2.9)$$

Генерация нормально распределённых случайных чисел.  $R_1$  и  $R_2$  являются случайными числами с равномерным распределением, лежащими в диапазоне от нуля до единицы. В результате вычислений число  $X$  окажется нормально распределённым и будет иметь среднее значение, равное нулю, а СКО, равное единице. В этом выражении используется натуральный логарифм (логарифм по основанию  $e$ ); аргумент функции косинуса выражается в радианах.

Как и ранее, с помощью данного подхода можно генерировать нормально распределённые случайные сигналы с произвольным средним значением и произвольным СКО. Для этого возьмите каждое число, полученное с помощью данного выражения, умножьте на желаемое значение СКО и добавьте требуемое среднее значение.

Работа генератора случайных чисел начинается с некоторого *начального значения*, лежащего в диапазоне от нуля до единицы. Это число по определённому фиксированному алгоритму преобразуется в новое число, лежащее в том же диапазоне и рассматриваемое как случайное. Это новое число сохраняется и при следующем вызове генератора случайных чисел используется в качестве начального значения. Алгоритм формирования случайного числа на основе начального значения выглядит следующим образом:

$$R = (aS + b) \bmod c. \quad (2.10)$$

Алгоритм работы генератора случайных чисел, имеющих равномерное распределение на интервале от нуля до единицы.  $S$  — начальное значение;  $R$  — генерируемое случайное число;  $a$ ,  $b$  и  $c$  — выбранные некоторым образом константы. В простой формулировке: результат сложения  $aS + b$  делится на  $c$ , а остаток от деления присваивается числу  $R$ .

Таким образом, на основе одного начального значения формируется последовательность случайных чисел. Нетрудно видеть, что если при генерации новой последовательности случайных чисел начальное значение оставить прежним, то она будет точно совпадать с первой сформированной последовательностью. Для того чтобы заставить генератор формировать неповторяющиеся случайные последовательности, большинство языков программирования включают функции

переустановки начального значения генератора случайных чисел так, чтобы оно каждый раз оказывалось новым. Общепринятой практикой является использование системных часов в качестве начального значения показаний. Это при каждом новом запуске программы гарантирует выдачу новой случайной последовательности.

С математической точки зрения, числа, сгенерированные описанным образом, не являются абсолютно случайными, так как каждое полученное значение полностью определяется *предыдущим* числом. Поэтому для таких процессов принято использовать термин  *псевдослучайные*. Однако эта особенность выходит за рамки рассматриваемой нами темы. С достаточно высокой степенью достоверности можно утверждать, что последовательности, формируемые с помощью генераторов случайных чисел, являются случайными. Вероятность того, что вам встретится ситуация, в которой такие последовательности нельзя будет считать случайными, крайне мала.

## 2.7. Точность и погрешность

*Точность* и *погрешность* — термины, используемые при описании систем и методов, предназначенных для каких-либо *измерений*, *оценок* или *предсказаний*. Во всех этих случаях мы заинтересованы в получении значения некоторого параметра, которое называют *истинным значением* или просто *истиной*. Желательно, чтобы *измеренное значение* было, насколько это возможно, близко к истинному. *Точность* и *погрешность* являются показателями ошибки между истинным и измеренным значением.

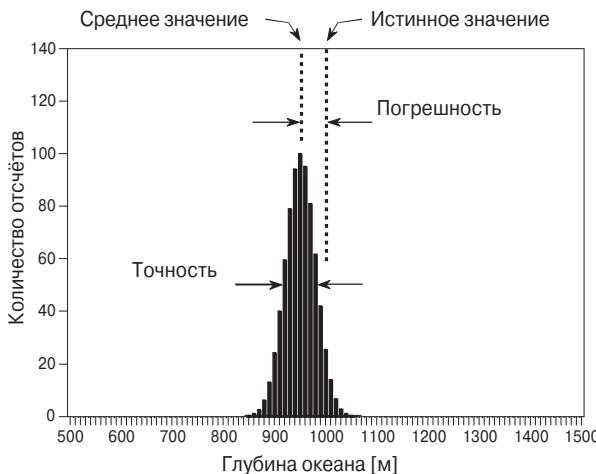
К сожалению, в нетехнических кругах понятия точности и погрешности часто считаются равнозначными. И в самом деле, их определения в словарях позволяют говорить о них как о синонимах! Однако в науке и технике существуют достаточно чёткие определения для каждого из этих терминов. Вам следует правильно использовать эти понятия, но спокойно относиться к ошибкам других людей.

В качестве примера представим океанографа, который измеряет глубину дна с помощью гидроакустического комплекса. Короткие звуковые импульсы излучаются с корабля, отражаются от дна океана и принимаются на поверхности в качестве эхо-сигнала. Звуковые волны проходят через водную среду со сравнительно постоянной скоростью, что позволяет определить глубину по разности времени приёма и передачи импульсов. Как и во всех эмпирических измерениях, существует определённая вероятность погрешности между измеренным и истинным значением. На результат данного измерения могут оказать влияние многие факторы, такие как шумы электронных цепей приёмника, волнение на поверхности океана, наличие растений на дне, перепады температуры воды и т. д.

Для исследования этих факторов океанограф производит многочисленные замеры в месте, где глубина составляет ровно 1000 метров (истинное значение). Затем его измерения представляются в виде гистограммы, как показано на **Рис. 2.11**. Как и предполагалось, исходя из центральной предельной теоремы, полученные данные будут распределены по нормальному закону. *Среднее значение* находится в центре распределения и является самой лучшей оценкой глубины, основанной на всех полученных данных. *Среднеквадратическое отклонение* опре-

деляет ширину кривой распределения и описывает степень различия результатов разных измерений.

В данном случае в системе возможны два основных типа ошибок. Во-первых, среднее значение может быть сдвинуто относительно истинного значения. Вели-



**Рис. 2.11.** Погрешность и точность. Погрешность — это разница между истинным значением и средним значением процесса, с помощью которого генерируются данные. Точность — это степень разброса значений, определяемая по СКО, отношению сигнал/шум или коэффициенту вариации.

чину сдвига называют *погрешностью измерения*. Во-вторых, отдельные измерения могут сильно отличаться друг от друга, что показывает ширину кривой распределения. Степень такого разброса данных и называют *точностью измерения*. Она выражается среднеквадратическим отклонением, *отношением сигнал/шум* или *коэффициентом вариации*.

Предположим, измерения выполняются с малой погрешностью, но низкой точностью. В этом случае гистограмма окажется сосредоточена около истинного значения, но она будет слишком широка. Получается, что совместное рассмотрение всей группы измерений может дать хороший результат, однако при выборке одного конкретного измерения вероятность того, что оно будет достаточно близко к истинному, мала. В таких случаях говорят, что имеет место плохая *повторяемость результатов измерений*; последовательно выполненные эксперименты плохо согласуются между собой. Низкая точность измерения является результатом *случайных ошибок*, т. е. таких искажающих факторов, которые меняются при каждом повторном измерении. При этом точность *всегда* можно повысить, если усреднить большее количество измерений. Можно сказать, что *точность является показателем уровня шума в системе*.

Теперь представим себе измерение, выполняемое с высокой точностью, но большой погрешностью. В этом случае гистограмма становится очень узкой, но она не концентрируется вокруг истинного значения. Последовательные замеры близки по значению, однако *все* они содержат грубые ошибки. Высокая погрешность является результатом *систематических ошибок*. Это такие ошибки, которые

одинаково повторяются при каждом измерении. Обычно величина погрешности зависит от того, как вы произвели *калибровку* системы. Например, при измерении глубины дна непосредственно измеряемым параметром является время. С помощью процедуры калибровки время преобразуется в глубину, при этом устанавливается соотношение между *миллисекундами* и *метрами*. Эта процедура может сводиться к простому умножению на постоянную скорость, а может включать десятки коррекций второго порядка. Усреднение результатов измерений никак не скажется на величине погрешности. Можно сказать, что *погрешность — это мера откалиброванности системы*.

На практике понятия точности и погрешности часто могут быть переплетены. Например, представьте себе электронный усилитель, схема которого собрана на резисторах с допуском номинала 1%. Такое допустимое отклонение означает, что значение сопротивления каждого резистора будет находиться в пределах 1% от номинального значения в зависимости от целого ряда факторов, таких, как температура, влажность, срок службы и т. д. Такой разброс значений сопротивлений вызовет соответствующий разброс коэффициента усиления устройства. На что будет влиять эта ошибка: на погрешность измерений или на их точность?

Ответ определяется тем, каким образом производятся измерения. Допустим, вы используете *один* усилитель и несколько раз тестируете его в течение нескольких минут. При этом ошибка в коэффициенте усиления остаётся неизменной в каждом тесте, тогда вы делаете вывод, что проблема имеет отношение к *погрешности*. В другом случае вы собираете *тысячу* усилителей. Коэффициент усиления случайным образом меняется от устройства к устройству, и в этом случае проблема, вероятно, относится к *точности*. Аналогично если один из усилителей будет демонстрировать флюктуацию коэффициента усиления в зависимости от температуры или других условий внешней среды, то проблема также будет связана с *точностью*.

При решении вопроса, к какому типу ошибок отнести неточность измерений, следует ответить на два вопроса. Первый: обеспечит ли усреднение нескольких полученных значений улучшение измерения? Если да, то эта ошибка относится к точности; если нет — к погрешности. Второй вопрос: уменьшится ли ошибка с помощью калибровки? Если да, то она относится к погрешности; если нет — к точности. В этом случае, возможно, потребуется решать вопрос о том, каким образом производить калибровку устройства и как часто её следует повторять.

# Глава 3

## АЦП И ЦАП

В большинстве случаев *сигналы*, с которыми можно встретиться в научных и инженерных разработках, в их непосредственной форме являются *непрерывными*. Например, это могут быть зависимость силы света от расстояния, изменение напряжения во времени, влияние температуры на скорость химической реакции и т. д. *Анало-цифровое преобразование* и *цифро-аналоговое преобразование* — это операции, позволяющие цифровым компьютерам взаимодействовать с миром реальных непрерывных сигналов. *Цифровые сигналы* отличаются от своих непрерывных прообразов двумя основными особенностями: они *дискретизированы* во времени и *квантованы* по уровню. Обе эти особенности ограничивают максимальное количество информации, которое может содержать цифровой сигнал. Данная глава посвящена управлению информацией при переходе из одной формы представления сигнала к другой: пониманию того, какую часть информации необходимо при этом сохранить, а какую можно позволить себе потерять. Такое понимание даёт возможность правильно выбрать *частоту дискретизации*, необходимую *разрядность* чисел (или *количество бит*) и вид *аналоговой фильтрации*, используемой в процессе преобразования.

В английском языке термины «цифровой компьютер» и «цифровые данные» записываются как «*digital computer*» и «*digital data*», но не «*digit computer*» и «*digit data*». В то же время процесс перевода сигнала из непрерывной формы в цифровую называется «*digitize*» и «*digitization*» (оцифровка), а не «*digitalize*» и «*digitalization*». Почему? Ответ оказывается простым, но неожиданным. Термины «*digitalize*» или «*digitalization*» (дигитализация) означают насыщение сердечной мышцы препаратами дигиталиса и используются в медицине уже около ста лет.

На Рис. 3.1 показаны графики, описывающие типичную процедуру анало-цифрового преобразования. На (а) изображён исходный непрерывный сигнал, который требуется оцифровать. Сигнал описывается зависимостью напряжения от времени. Для удобства будем считать, что напряжение меняется от 0 до 4.095 В и может быть представлено числами от 0 до 4095, генерируемыми 12-битным анало-цифровым преобразователем. Схема преобразования разбита на два блока: устройство *выборки-хранения* и собственно *аналого-цифровой преобразователь (АЦП)*. Как вам, возможно, известно из курса электроники, устройство выборки-хранения необходимо, чтобы на время преобразования зафиксировать входное напряжение АЦП на постоянном уровне. Но это не является причиной, по которой данный блок выделен нами в схеме. Разбиение процесса оцифровки на две стадии позволяет сформировать важную для понимания модель анало-цифрового преобразования. То, что это оказалось напрямую связано с известными из электроники понятиями, — просто удачное совпадение.

На (б) показан сигнал на выходе устройства выборки-хранения. Это сигнал, изменяющий своё значение только в фиксированные моменты времени, следу-

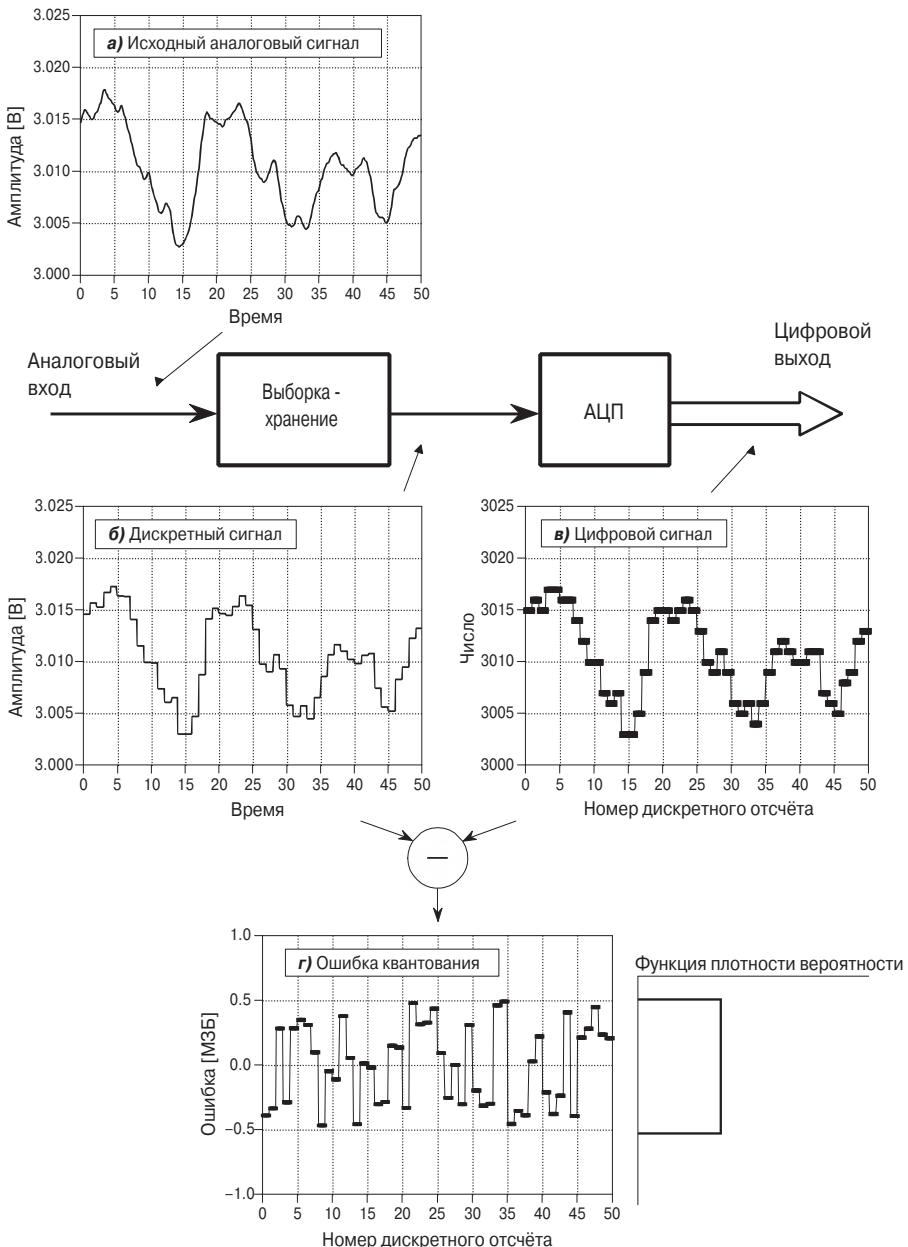
ющие с постоянным периодом, и равный в эти моменты мгновенному значению сигнала на входе. Поведение входного сигнала между моментами выборки полностью игнорируется. Таким образом, процедура выборки-хранения реализует *дискретизацию* — переход от непрерывного закона изменения независимой переменной (в данном примере — времени) к дискретному.

На (в) иллюстрируется работа АЦП, который осуществляет *квантование* — генерирует целое число от 0 до 4095 для каждой ступеньки сигнала, показанного на (б). При этом в сигнал вносится ошибка, так как уровень входного напряжения может принимать любое значение от 0 до 4.095 В. Например, и 2.56000 В, и 2.56001 В на входе АЦП будут на его выходе представлены одним и тем же числом 2560. Процедура квантования осуществляет переход от непрерывного закона изменения зависимой переменной (в данном случае — напряжения) к дискретному.

Отметим, что мы избегаем сопоставления (а) и (в), так как это заставило бы нас рассматривать совместно процедуры дискретизации и квантования. Для нас важен раздельный анализ этих этапов, поскольку каждый из них по-своему влияет на результат преобразования. Кроме того, на практике параметры дискретизации и квантования задаются различными механизмами. Бывают также ситуации, когда используется только один из этапов. Например, в *фильтрах с переключаемыми конденсаторами* квантование сигнала по уровню не производится.

Начнём с процедуры квантования. В результате квантования любой отсчёт цифрового сигнала может содержать ошибку, максимальное значение которой равно  $\pm 1/2$  значения младшего значащего бита («расстояния» между соседними уровнями квантования). На (г) показан пример сигнала *ошибки квантования*. Этот сигнал рассчитывается как разность между сигналом на выходе АЦП (в) и сигналом на выходе устройства выборки-хранения (б) с учётом необходимых преобразований. С другой стороны, можно сказать, что цифровой сигнал на выходе АЦП (в) равен сумме сигнала на входе АЦП (б) и сигнала ошибки квантования (г). Сделаем важное замечание: ошибку квантования можно рассматривать как *белый шум*. На основании вышесказанного приходим к следующему утверждению: в большинстве случаев квантование сигнала равносильно простому добавлению к нему аддитивного белого шума определённого уровня мощности. Шум равномерно распределён на интервале  $\pm 1/2$  значения младшего бита, имеет нулевое *математическое ожидание* и *среднеквадратическое отклонение*, равное  $1/\sqrt{12}$  значения младшего бита ( $\approx 0.29$  значения младшего бита). Так, например, прохождение аналогового сигнала через 8-битный аналого-цифровой преобразователь добавляет к сигналу шум со среднеквадратическим отклонением  $0.29/256$  или  $1/900$  максимального значения, представляемого разрядной сеткой. 12-битное преобразование добавляет шум с СКО  $0.29/4096$  или  $1/14000$  максимального значения, а 16-битное преобразование — шум с СКО  $0.29/65536$  или  $1/227000$  максимального значения. Так как ошибка квантования является *случайным процессом (шумом)*, то число битов (двоичных разрядов) определяет *точность* представления данных. Например, следующее выражение будет звучать грамотно: «Мы повысили точность измерения с 8 до 12 бит».

Предложенная модель ошибки квантования имеет чрезвычайно важное значение. Она позволяет рассматривать результат этой процедуры как простое добавление белого шума к шумам, присутствующим в исходном аналоговом сигнале. В качестве примера рассмотрим непрерывный сигнал с максимальной амплитудой 1 В.



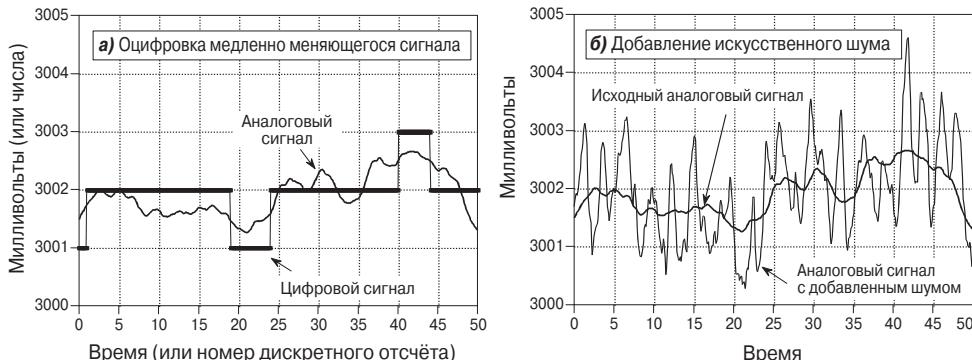
**Рис. 3.1.** Электрические зависимости, иллюстрирующие процесс аналого-цифрового преобразования. Процедура разбита на два этапа, что позволяет раздельно рассматривать эффекты, связанные с дискретизацией и квантованием. Первый этап — выборка-хранение. Он оставляет в сигнале только мгновенные значения входного сигнала, взятые с периодом дискретизации. Второй этап — непосредственно аналого-цифровое преобразование. Уровень сигнала принимает целочисленные значения, ближайшие к реальному уровню сигнала. Эта процедура приводит к появлению ошибки представления каждого отсчета цифрового сигнала, лежащей в диапазоне  $\pm 1/2$  значения младшего значащего бита (МЗБ) (см. Рис. 3.1г). Обычно результат квантования можно рассматривать как простое добавление к сигналу белого шума.

Пусть в этом сигнале присутствует белый шум со среднеквадратическим отклонением 1 мВ. Процесс аналого-цифрового преобразования с 8-битным представлением сигнала переведёт 1 В в 255, а 1 мВ — в 0.255 значения младшего значащего бита. Как было указано в Главе 2, дисперсия суммы сигналов, представляющих собой белый шум, равна сумме дисперсий этих сигналов. Тогда для вычисления СКО

суммарного сигнала используем формулу  $\sqrt{A^2 + B^2} = C$ . То есть СКО шума на выходе аналого-цифрового преобразования в данном примере будет составлять

$\sqrt{0.2552^2 + 0.292^2} = 0.386$  значения младшего значащего бита. Это означает более чем 50%-е увеличение уровня шума в результате оцифровки. В то же время 12-битное аналого-цифровое преобразование практически не даст нарастания шума и оцифровка почти не внесёт потерь. Таким образом, при решении проблемы выбора разрядности аналого-цифрового преобразования, требуемой для реализации той или иной системы, следует ответить на два вопроса: каков уровень шума в исходном аналоговом сигнале и какой уровень шума допустим для цифрового сигнала?

Всегда ли справедлива выбранная модель ошибки квантования? Нет. Исключением является ситуация, когда значение аналогового сигнала на входе АЦП мало изменяется от одного *периода дискретизации* к другому в течение длительного интервала времени (**Рис. 3.2а**). В этом случае значение выходного сигнала остаётся на постоянном уровне для большого числа идущих подряд дискретных отсчётов, несмотря на то, что аналоговый сигнал на входе может колебаться в пределах



**Рис. 3.2.** Искусственное добавление шума. На (а) показано, как аналоговый сигнал, амплитуда которого меняется в диапазоне не более  $\pm 1/2$  значения младшего значащего бита, может оказаться при оцифровке зафиксированным на одном уровне квантования. Искусственное зашумление позволяет этого избежать путём «подмешивания» к сигналу небольшого шума (б). В показанном примере добавляемый шум имеет нормальное распределение и СКО, равное  $2/3$  значения младшего значащего бита. На (в) показано, что добавление шума вызвало скачки в цифровом сигнале между соседними уровнями квантования. Это позволяет сохранить большее количество информации об исходном аналоговом сигнале.

$\pm 1/2$  значения младшего значащего бита. Сигнал ошибки квантования нельзя при этом считать белым шумом. Он становится похож скорее на результат пороговой обработки.

Общепринятой методикой повышения качества аналого-цифрового преобразования медленно меняющихся сигналов является *искусственное добавление шума (dithering)*. Эта процедура проиллюстрирована на Рис. 3.2б. К исходному аналого-вому сигналу «подмешивается» небольшой белый шум. В рассматриваемом примере этот шум является *нормально распределённым* и имеет среднеквадратическое отклонение, равное  $2/3$  значения младшего значащего бита; полный размах амплитуды — 3 значения. На Рис. 3.2в показано, какое влияние оказывает искусственное внесение шума на цифровой сигнал. Даже если уровень аналогового сигнала колеблется в диапазоне менее  $\pm 1/2$  значения младшего значащего бита, добавление шума вызывает в выходном цифровом сигнале случайные скачки между двумя соседними уровнями квантования.

Чтобы понять, почему это улучшает ситуацию, представим себе, что в качестве входного сигнала используется постоянное напряжение 3.0001 В. Будем считать, что это соответствует значению, превышающему уровень квантования 3000 на 0.1 шага квантования. Если не использовать искусственное добавление шума, то по прошествии 10 000 периодов дискретизации все 10 000 выходных чисел примут значение 3000. Продолжая эксперимент, добавим к сигналу небольшой по мощности шум. В результате получим 10 000 значений, колеблющихся между двумя (или более) уровнями квантования, так что в 90% случаев будем иметь число 3000 и в 10% случаев — 3001. Вычислив среднее значение для всех 10 000 дискретных отсчётов, получим приближенно 3000.1. Точность одного измерения всегда ограничена  $\pm 1/2$  значения младшего значащего бита. Однако статистические данные, собираемые большим количеством дискретных отсчётов, позволяют добиться гораздо лучших результатов. Ситуация кажется странной: добавление шума повышает точность!

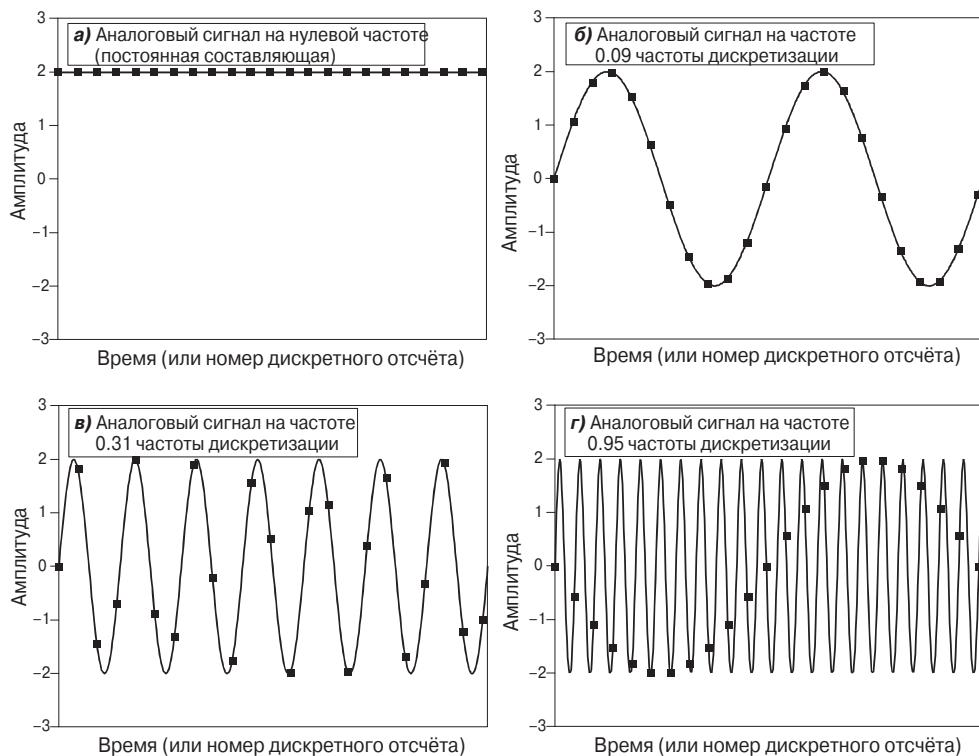
Практическая реализация описанной методики оказывается весьма сложной. Например, можно использовать специальный процессор, генерирующий *последовательность случайных чисел*, поступающую на *цифро-аналоговый преобразователь* и суммируемую с входным сигналом. После оцифровки процессор вычитает известную ему случайную последовательность из цифрового сигнала. Этот изящный подход называется *искусственным зашумлением с вычитанием (subtractive dither)*. Он, однако, используется только в сложных системах. Простейшие же методы основаны на использовании того шума, который уже присутствует в сигнале. Но они не всегда оказываются возможными.

## 3.1. Теорема отсчётов

Как правильно выполнить *дискретизацию*? Допустим, мы каким-либо образом провели дискретизацию *непрерывного сигнала*. Если теперь по *дискретным отсчётам* мы сможем восстановить исходный *аналоговый сигнал*, то это будет означать, что дискретизация выполнена правильно. Данные в дискретной форме могут выглядеть неправдоподобными или неполными, но если мы можем восстановить исходный сигнал, значит, ключевая информация в них всё-таки содержится.

На Рис. 3.3 показано несколько гармонических сигналов до и после оцифровки. Непрерывной линией показан аналоговый сигнал на входе АЦП, а квадратными отметками — цифровой сигнал на выходе АЦП. Входной сигнал на (а) — это постоянный сигнал, формируемый как косинус нулевой частоты. Из рисунка видно, что аналоговый сигнал можно восстановить из цифровых отсчётов, просто соединив их прямыми линиями. Таким образом, вся информация для получения исходного непрерывного сигнала имеется в цифровом сигнале. То есть дискретизация выполнена правильно.

На (б) показана синусоида с частотой, равной 0.09 частоты дискретизации. Например, это может соответствовать слушаю гармонического сигнала частотой 90 Гц, дискретизируемого со скоростью 1000 отсчётов в секунду (1000 Гц). Это



**Рис. 3.3.** Правильно и неправильно выполненная дискретизация. Дискретизация считается правильной, если в дискретных отсчётах содержится достаточная для восстановления исходного аналогового сигнала информация. Случай правильной дискретизации для трёх синусоидальных сигналов различной частоты проиллюстрируют (а, б и в). Этот факт, однако, не является очевидным: дискретный сигнал на (в), например, по своей форме даже не напоминает исходный сигнал. И всё-таки для каждого из этих рисунков характерно строгое и единственное возможное соответствие аналоговых сигналов их дискретным эквивалентам. А это гарантирует нам возможность восстановления непрерывного сигнала. На (г) частота непрерывного сигнала оказывается выше частоты Найквиста (половины частоты дискретизации). Это приводит к эффекту заворачивания спектра, когда частота дискретного сигнала отличается от частоты непрерывного сигнала. Заворачивание спектра искажает информацию, и исходный аналоговый сигнал уже не может быть восстановлен по дискретным отсчётом.

также означает, что на одном периоде синусоиды укладывается 11.1 дискретных отсчёта. Данная ситуация сложнее предыдущего примера, так как исходный аналоговый сигнал уже не может быть восстановлен простым соединением дискретных отсчётов прямыми линиями. Адекватно ли эти отсчёты отражают исходный сигнал? Ответ — «да», поскольку ни один другой гармонический сигнал или их сумма не может дать такой же последовательности отсчётов (в рамках разумных ограничений, описываемых ниже). По данным отсчётам может быть построен только один аналоговый сигнал, и, следовательно, исходный аналоговый сигнал может быть полностью восстановлен. Таким образом, дискретизация вновь выполнена правильно.

На (б) показана ещё более сложная ситуация: частота аналогового сигнала увеличена до 0.31 частоты дискретизации. На одном периоде синусоиды теперь укладывается только 3.2 дискретных отсчёта. Отсчёты идут настолько редко, что даже не повторяют форму непрерывного сигнала. Содержит ли теперь дискретный сигнал всю информацию об исходном непрерывном? Ответ — вновь «да», по тем же причинам. По этим отсчётам может быть построен единственный аналоговый сигнал. Вся информация, необходимая для восстановления непрерывного сигнала, содержится в цифровых данных. Почему это так, вы узнаете дальше. Конечно, доказательство данного факта является более сложным, чем соединение отсчётов линиями. Итак, как бы странно это ни казалось, мы вновь рассмотрели случай правильно выполненной дискретизации.

Аналоговый сигнал на (г) имеет ещё более высокую частоту, составляющую 0.95 частоты дискретизации, так что на период синусоиды приходится всего лишь 1.05 отсчёта. Адекватно ли в этом случае дискретные отсчёты представляют непрерывный сигнал? Нет. Они отображают сигнал, отличный от исходного. Синусоида с частотой 0.95 преобразуется в синусоиду с частотой 0.05 частоты дискретизации. Это явление изменения частоты сигнала в процессе дискретизации, называемое *заворачиванием спектра*, приводит к *наложению спектральных составляющих* дискретного сигнала и, как следствие, неоднозначному восприятию его спектра. Подобно тому как преступник может скрываться под вымышленными именами (прозвищами), аналоговые сигналы при неправильной дискретизации «скрываются» в вымышленных частотах<sup>1)</sup>. Поскольку цифровые данные теперь неоднозначно отражают исходный сигнал, восстановить его становится затруднительно. В цифровом сигнале нет никакой информации о том, какой из двух частот, 0.95 или 0.05, он соответствует. Сигнал полностью скрыл от нас своё лицо. Опознать «преступника» стало невозможно. Идеальное преступление! О дискретизации в этом случае можно сказать, что она выполнена неправильно.

Эта линия рассуждений приводит нас к одному из основополагающих положений в теории цифровой обработки сигналов — *теореме отсчётов*. Часто её называют также *теоремой отсчётов Шеннона* или *теоремой отсчётов Найквиста*, по именам авторов, опубликовавших в 1940-х годах свои работы по этой теме<sup>2)</sup>. Тео-

<sup>1)</sup> В английском языке слову «прозвище» соответствует слово «alias», а термин «наложение спектров при дискретизации» звучит как «aliasing». — Примеч. пер.

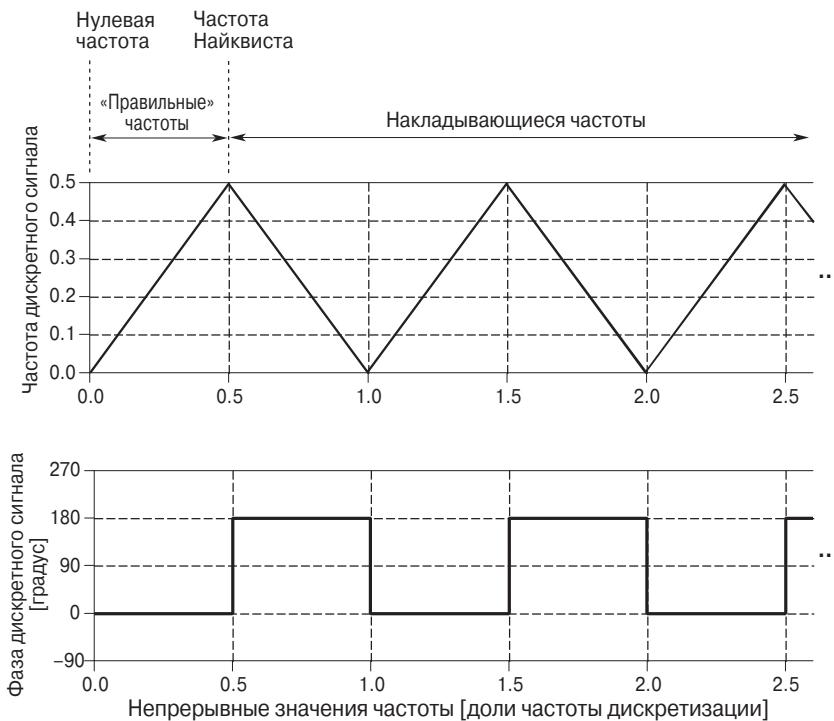
<sup>2)</sup> В российской научной литературе эта теорема чаще фигурирует как теорема Котельникова. Академик В.А. Котельников сформулировал эту важную в теории связи теорему в 1938 году. — Примеч. пер.

рема отсчётов утверждает, что непрерывный сигнал может быть адекватно представлен в дискретной форме только в случае, если он не содержит частот выше  $1/2$  частоты дискретизации. Например, частота дискретизации 2000 Гц требует, чтобы спектр непрерывного сигнала был ограничен частотой 1000 Гц. Если в сигнале будут присутствовать компоненты с частотами выше 1000 Гц, произойдёт их наложение на частоты в диапазоне 0..1000 Гц, и информация, расположенная в этой полосе, будет искажена.

В зарубежной литературе при обсуждении вопросов дискретизации часто используются два термина: «Nyquist Frequency» и «Nyquist Rate» (*частота Найквиста*). При этом значение этих терминов строго не оговорено. Разные авторы используют их по-разному. Рассмотрим такой пример. Полоса частот аналогового сигнала лежит в пределах от 0 до 3 кГц. В соответствии с теоремой отсчётов частота дискретизации в этом случае должна быть 6 кГц или выше. Предположим, мы выбрали частоту дискретизации, равную 8 кГц, позволяя аналоговому сигналу включать частоты до 4 кГц. Таким образом, мы имеем 4 важные величины: граница спектра аналогового сигнала — 3 кГц; удвоенное значение границы спектра — 6 кГц; частота дискретизации — 8 кГц;  $1/2$  частоты дискретизации — 4 кГц. Указанные выше термины могут применяться для обозначения всех этих величин, что может вносить путаницу. К счастью, обычно авторы оговаривают, что они понимают под «частотой Найквиста». В нашем случае будем использовать термин частота Найквиста для обозначения  $1/2$  частоты дискретизации.

**Рис. 3.4** иллюстрирует эффект *наложения спектров* при дискретизации. Самое главное — помнить, что цифровой сигнал не может содержать частот, больших половины частоты дискретизации (частоты Найквиста). Если частоты аналогового сигнала лежат ниже частоты Найквиста, они представляются адекватно. Если же в спектре сигнала присутствуют частоты, превышающие частоту Найквиста, то в дискретном сигнале они должны перейти в разрешённый диапазон от 0 до  $1/2$  частоты дискретизации, вызывая наложение спектров. На **(а)** показано, что любой частоте выше частоты Найквиста в дискретном сигнале будет соответствовать определённая частота из диапазона от 0 до  $1/2$  частоты дискретизации. Если на этой «нижней» частоте уже содержится информация в исходном сигнале, то при дискретизации произойдёт сложение двух частотных компонент, и данные окажутся искажёнными. Заворачивание приводит не только к искажению спектра ниже частоты Найквиста, но и вызывает потерю информации о частотах, лежащих выше этой частоты. Предположим, мы имеем гармонический дискретный сигнал, частота которого равна  $0.2$  частоты дискретизации. Если считать, что дискретизация была выполнена правильно, то можно говорить, что исходный аналоговый сигнал содержал единственную гармонику на частоте  $0.2$  частоты дискретизации. Однако если в процессе дискретизации было допущено наложение спектров, то непрерывный сигнал с тем же дискретным представлением мог содержать бесконечное число частотных компонент:  $0.2, 0.8, 1.2, 1.8, 2.2, \dots$  частоты дискретизации.

Кроме изменения частоты сигнала, эффект заворачивания спектра может изменять также и фазу сигнала. Для примера вернёмся к **Рис. 3.3г** Дискретный сигнал, полученный с наложением спектров, является инверсным по отношению к исходному непрерывному сигналу: если первый представлен синусоидой, то второй — синусоидой с обратным знаком. Эффект наложения спектров привёл не

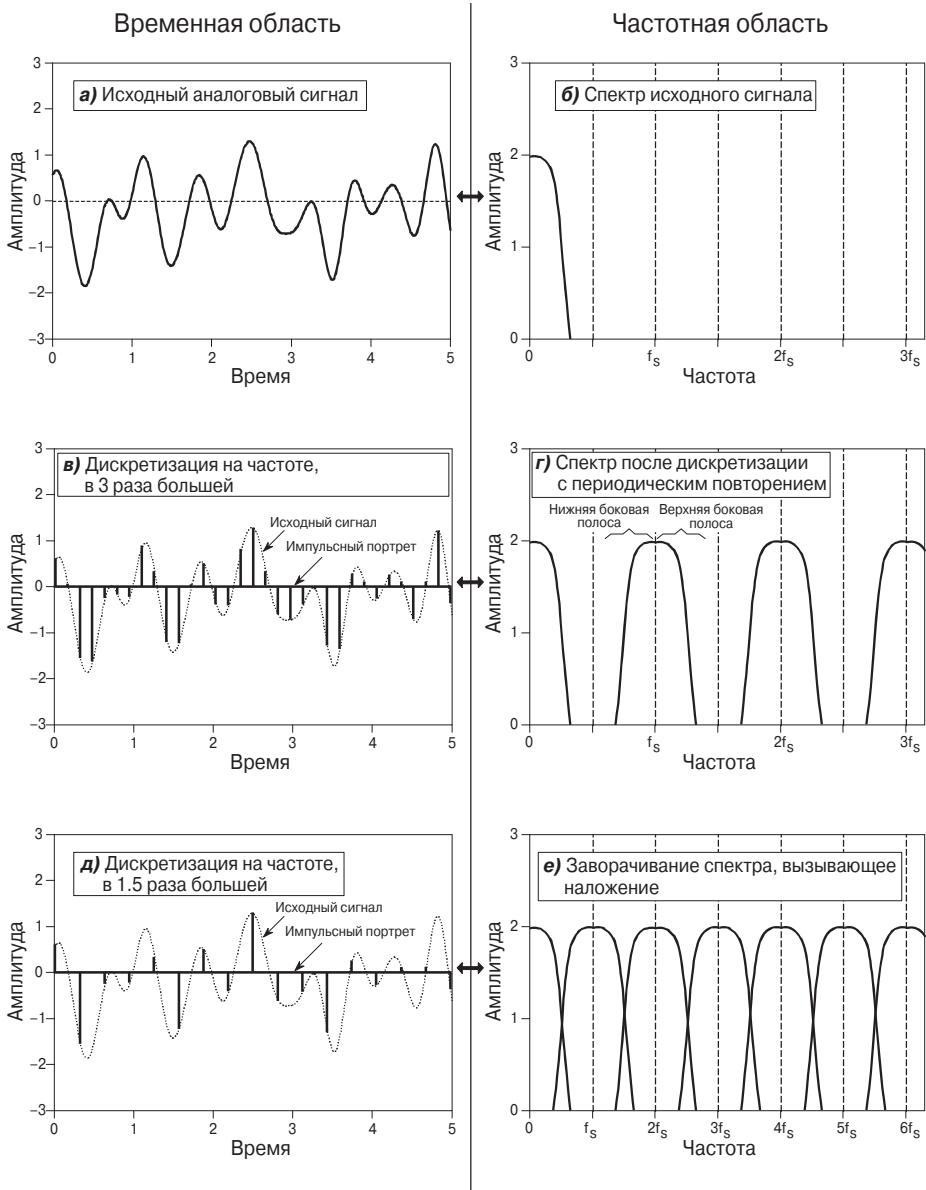


**Рис. 3.4.** Преобразование частотных компонент непрерывного сигнала в процессе дискретизации. Частоты непрерывного сигнала, лежащие ниже половины частоты дискретизации, переходят непосредственно в частоты дискретного сигнала без искажений. Для частот, лежащих выше половины частоты дискретизации, имеет место заворачивание спектра и наложение на нижние частоты, что приводит к неправильному его представлению. В процессе дискретизации любые частоты, входящие в состав аналогового сигнала, оказываются в диапазоне от 0 до  $1/2$  частоты дискретизации. Кроме того, наложение спектров может приводить к сдвигу фазы сигнала на  $180^\circ$ .

только к переносу частоты, но и к *сдвигу фазы* на  $180^\circ$ . Возможны, однако, только два значения поворота фазы: на  $0^\circ$  (нет сдвига) и на  $180^\circ$  (инверсия). Нулевой сдвиг фазы характерен для частот  $0\dots0.5, 1\dots1.5, 2\dots2.5$  частоты дискретизации и т. д. Поворот на  $180^\circ$  происходит для частот  $0.5\dots1, 1.5\dots2, 2.5\dots3$  частоты дискретизации и т. д.

Рассмотрим более детально процесс дискретизации и то, как происходит наложение спектров. Наша цель при этом — понять, что происходит с информацией, заключённой в сигнале, при его преобразовании из непрерывного в дискретный. Здесь, однако, возникает проблема. Дискретная и непрерывная формы представления сигнала совершенно различны: одна — это непрерывный колебательный процесс, другая — массив чисел. Сравнивать их между собой невозможно. Чтобы разрешить данное противоречие, введём понятие *импульсного портрета* (*impulse train*).

На Рис. 3.5 приведён пример аналогового сигнала, показана его дискретизация и изображён соответствующий ему *импульсный портрет* (*e*). Он представляет собой непрерывный сигнал, состоящий из последовательности бесконечно коротких импульсов, следующих с периодом дискретизации и равных по уровню



**Рис. 3.5.** Теорема отсчётов во временной и частотной областях. На (а и б) показан аналоговый сигнал с полосой частот, лежащей в пределах 0...0.33 частоты дискретизации  $F_d$ . На (в) показана дискретизация аналогового сигнала и представление его в виде импульсного портрета. В частотной области (г) это приводит к появлению бесконечного числа копий исходного спектра, включающих нижние и верхние боковые полосы. Поскольку частоты исходного сигнала (б) остались при этом без изменений (в), дискретизацию можно считать выполненной правильно. В другой ситуации (д) аналоговый сигнал содержит частоты до 0.66 частоты дискретизации, т. е. выше частоты Найквиста. Результатом является наложение спектров, показанное на (е) в виде перекрытия соседних полос.

значениям исходного сигнала в эти же моменты времени. Каждый импульс имеет бесконечно малую длительность. Подробно такие сигналы будут рассмотрены в Главе 13. Значения импульсного портрета в моменты между импульсами равны нулю. Следует помнить, что понятие импульсного портрета является чисто теоретическим и не имеет физической реализации. С его помощью мы приходим к возможности описания исходного аналогового, и получаемого дискретного сигналов в непрерывной форме, избегая таким образом сравнения несопоставимых величин.

Установим взаимосвязь между импульсным портретом и дискретным сигналом. Это достаточно просто, т. к. с точки зрения содержащейся в них информации они идентичны. Дискретный сигнал и импульсный портрет — это два конца моста, соединяющего мир дискретных (**в**) и мир аналоговых (**а**) сигналов.

На (**а**, **в** и **д**) изображены сигналы, представленные в непрерывной форме, а на (**б**, **г** и **е**) показаны соответствующие им спектры. Понятие *спектра*, возможно, известно вам из курса электроники: любой сигнал может быть представлен набором гармонических компонентов различной амплитуды и частоты. Представление сигналов в частотной области будет подробно рассматриваться в последующих главах. Возможно, вам будет полезно вернуться к данному разделу после ознакомления с понятием спектра.

На (**а**) показан аналоговый сигнал, дискретизацию которого мы хотим осуществить. Его спектр (**б**) содержит частоты от 0 до приблизительно  $0.33f_d$ , где  $f_d$  — частота дискретизации, которую мы намерены использовать. В качестве конкретного примера можно рассмотреть речевой сигнал, пропущенный через фильтр, подавляющий все частоты выше 3.3 кГц. Частота дискретизации в этом примере будет составлять 10 кГц (10 000 дискретных отсчётов в секунду).

Получаемый дискретный сигнал, представленный в форме импульсного портрета, показан на (**в**). Его спектр (**г**) получен из спектра непрерывного сигнала дополнением составляющей, симметричной относительно вертикальной оси, и копированием на все частоты, кратные частоте дискретизации:  $f_d$ ,  $2f_d$ ,  $3f_d$  и т. д. При этом правые и левые симметричные части каждой копии спектра называют соответственно *верхней* и *нижней боковыми полосами* (*upper sideband* и *lower sideband*). Таким образом, в результате дискретизации в спектре сигнала оказываются новые частоты. Следует убедиться, что дискретизация выполнена правильно. В данном случае это так, потому что спектр исходного сигнала может быть восстановлен из спектра дискретного сигнала простым отбрасыванием частотных компонент, лежащих выше  $1/2$  частоты дискретизации. Эта процедура может выполняться, например, аналоговым фильтром нижних частот.

Если вы уже знакомы с основами цифровой обработки сигналов, вам, возможно, будет полезно узнать объяснение периодического повторения спектра дискретного сигнала. Во *временной области* процесс дискретизации может рассматриваться как умножение непрерывного сигнала на так называемую *дискретизирующую последовательность* — импульсный портрет, состоящий только из импульсов единичной амплитуды. Спектр такого импульсного портрета также представляет собой импульсный портрет единичной амплитуды с импульсами, повторяющимися на частотах  $f_d$ ,  $2f_d$ ,  $3f_d$  и т. д. Известно, что процедура умножения во временной области эквивалентна операции свёртки в *частотной области*. В результате свёртки и происходит размножение спектра непрерывного сиг-

нала на все частоты. При рассмотрении спектра исходного сигнала не только в области положительных, но и в области отрицательных частот появляются верхняя и нижняя боковые полосы. Можно провести аналогию указанного эффекта с *амплитудной модуляцией*, которая будет рассмотрена в Главе 10.

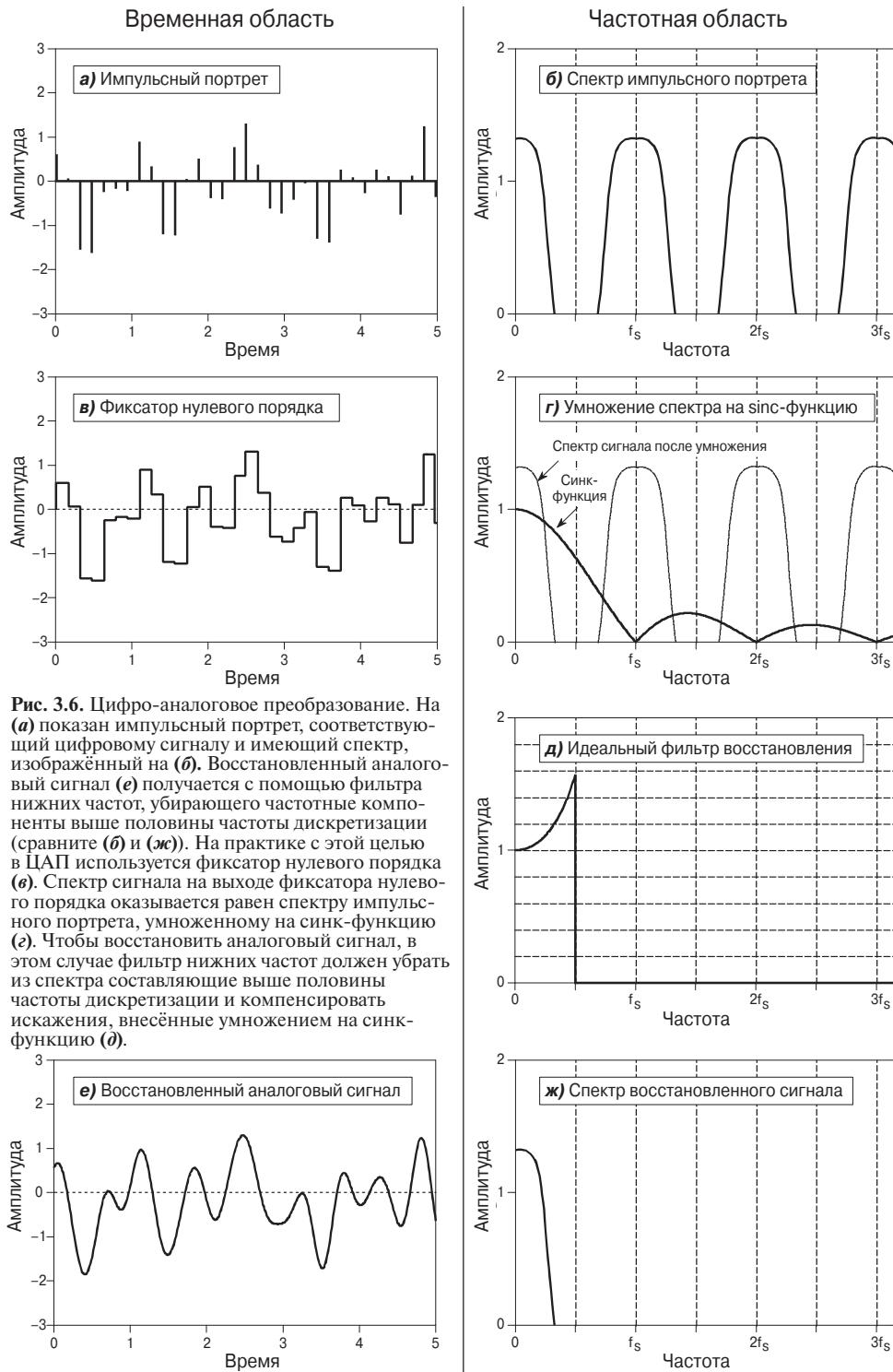
На (д) показан случай неправильной дискретизации в результате использования недостаточно высокой частоты дискретизации. Аналоговый сигнал по-прежнему лежит в полосе до 3.3 кГц, но частота дискретизации понижена до 5 кГц. Обратите внимание, что отметки  $f_d$ ,  $2f_d$ ,  $3f_d$ , ... на (е) идут чаще, чем на (д). На (е) продемонстрирован негативный эффект наложения спектров. Копия спектра исходного сигнала частично попала в полосу  $0 \dots 1/2 f_d$ . На (е) перекрывающиеся полосы частот показаны как бы раздельно, однако в действительности они суммируются, образуя общую искажённую смесь. Способов разделения наложившихся частотных полос нет<sup>1)</sup>. Поэтому информацию об исходном сигнале можно считать утерянной, а его восстановление — невозможным. Перекрытие спектров происходит тогда, когда аналоговый сигнал содержит частоты выше половины частоты дискретизации.

## 3.2. Цифро-аналоговое преобразование

Теоретически простейшим методом *цифро-аналогового преобразования* является чтение цифровых отсчётов (элементов массива) из памяти и преобразование их в *импульсный портрет* (Рис. 3.6). Получаемый при таком подходе импульсный портрет показан на (а), а его *спектр* — на (б). В этом случае исходный *аналоговый сигнал*, как уже было сказано, может быть полностью восстановлен путём пропускания импульсного портрета через *фильтр нижних частот с частотой среза, равной 1/2 частоты дискретизации*. Другими словами, спектр исходного сигнала и спектр импульсного портрета ниже *частоты Найквиста* (равной половине частоты дискретизации) являются идентичными. Выше этой частоты в спектре импульсного портрета содержатся периодически повторяющиеся копии, а в исходном аналоговом сигнале спектр на этих частотах равен 0 (предполагаем, что наложения допущено не было).

Описанный подход математически безупречен, однако на практике оказывается, что генерировать требуемые короткие импульсы, из которых состоит импульсный портрет, очень сложно. Чтобы решить эту проблему, практически все цифро-анalogовые преобразователи просто «задерживают» значение последнего *дискретного отсчёта* до прихода следующего. Эта процедура называется *фиксацией нулевого порядка* (фиксатор первого порядка соединяет соседние отсчёты прямыми линиями, фиксатор второго порядка — параболами и т. д.). При цифро-анalogовом преобразовании фиксация нулевого порядка служит аналогом процедуры *выборки-хранения*, используемой в *аналого-цифровом преобразовании*. Сигнал на выходе фиксатора нулевого порядка имеет вид ступеней (в).

<sup>1)</sup> В действительности такие способы есть, но они относятся к более сложной теории ЦОС, не рассматриваемой в этой книге. Это методы, предложенные Вайдянатханом. — Примеч. пер.



**Рис. 3.6.** Цифро-аналоговое преобразование. На (а) показан импульсный портрет, соответствующий цифровому сигналу и имеющий спектр, изображенный на (б). Восстановленный аналоговый сигнал (е) получается с помощью фильтра нижних частот, убирающего частотные компоненты выше половины частоты дискретизации (сравните (б) и (ж)). На практике с этой целью в ЦАП используется фиксатор нулевого порядка (в). Спектр сигнала на выходе фиксатора нулевого порядка оказывается равен спектру импульсного портрета, умноженному на sinc-функцию (г). Чтобы восстановить аналоговый сигнал, в этом случае фильтр нижних частот должен убрать из спектра составляющие выше половины частоты дискретизации и компенсировать искажения, внесенные умножением на sinc-функцию (д).

Процедура фиксации нулевого порядка в *частотной области* эквивалентна умножению спектра импульсного портрета на спектр прямоугольного импульса, показанный на (e) жирной линией и описываемый выражением

$$H(f) = \frac{\sin(\pi f / F_d)}{(\pi f / F_d)}. \quad (3.1)$$

Ослабление высокочастотных компонент в процессе фиксации нулевого порядка. Описываемая выражением (3.1) кривая показана на Рис. 3.6г  $f_d$  — частота дискретизации, при  $f=0$ ,  $H(0)=1$ .

В общем виде это выражение выглядит как  $\sin(\pi x)/(\pi x)$  и называется *sinc-функцией* —  $\text{sinc}(x)$ . Sinc-функция (или функция «синус икс на икс») очень часто встречается в ЦОС, поэтому она будет обсуждаться более детально в последующих главах. Если у вас уже есть представление о данном материале, вы, вероятно, знаете, что процедура фиксации нулевого порядка может рассматриваться как операция *свёртки* импульсного портрета с прямоугольным импульсом, длительность которого равна периоду дискретизации. В *частотной области* это эквивалентно умножению на спектр прямоугольного импульса, т. е. на sinc-функцию. На (e) тонкой линией показан спектр импульсного портрета (спектр сигнала), а толстой линией — sinc-функция. Спектр сигнала на выходе фиксатора нулевого порядка равен произведению этих двух спектров.

Для восстановления исходного сигнала после операции фиксации нулевого порядка используется *аналоговый фильтр*, который должен выполнить два действия: 1) подавить все частотные компоненты, лежащие выше половины частоты дискретизации; 2) компенсировать влияние фиксатора нулевого порядка с помощью обратного сигнала  $1/\text{sinc}(x)$ . При этом происходит усиление частотных компонент, в частности, на половине частоты дискретизации почти на 36%. На Рис. 3.6д показана *частотная характеристика* такого идеального аналогового фильтра.

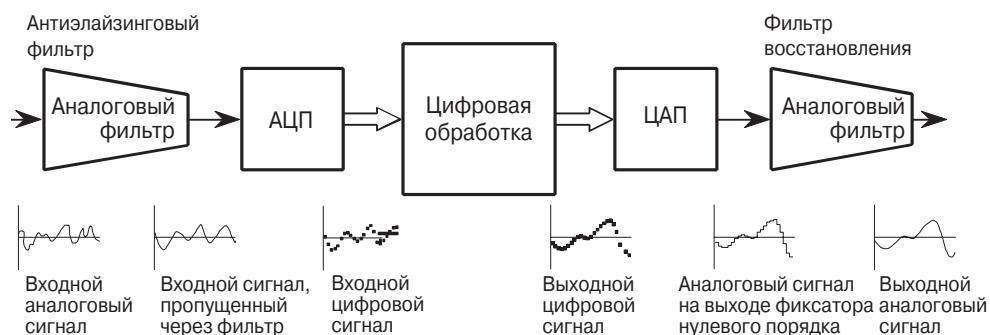
При решении проблемы компенсации эффекта, вносимого фиксатором нулевого порядка, можно поступить одним из следующих способов: 1) проигнорировать проблему и принять вызываемые ею последствия; 2) спроектировать аналоговый фильтр, который бы включал поправку на  $1/\text{sinc}(x)$  в своей частотной характеристике; 3) использовать оригинальную методику *многоскоростной обработки*, описанную далее в этой главе; 4) осуществлять коррекцию программным способом до цифро-аналогового преобразования (см. Главу 24).

Перед тем как закончить изложение этой части материала, посвящённого *дискретизации*, хотелось бы развеять широко распространённый миф о преимуществах аналогового сигнала перед цифровым. Как было показано выше, количество информации, которая может содержаться в *цифровом сигнале*, ограничено двумя факторами. Во-первых, *разрядность* представления чисел ограничивает разрешающую способность по зависимой переменной. То есть небольшие изменения амплитуды сигнала теряются в *шумах квантования*. Во-вторых, *частота дискретизации* ограничивает разрешающую способность по независимой переменной: информация о поведении сигнала между соседними моментами выборки оказывается неизвестной. Это, кстати, другой способ объяснения, почему частоты, лежащие выше *частоты Найквиста*, теряются.

На этой почве возникает миф: раз аналоговые сигналы характеризуются непрерывными параметрами, значит, они имеют бесконечно высокое разрешение как по независимой, так и по зависимой переменной. Это не так! Аналоговые сигналы имеют те же ограничения, что и цифровые, — *шум и ширина полосы частот* (наивысшая частота, которая может присутствовать в сигнале). Шум в аналоговом сигнале ограничивает *точность* измерения его формы точно так же, как шум квантования ограничивает точность в цифровом сигнале. А возможность измерения поведения аналогового сигнала на коротком интервале времени определяется содержащейся в нём максимальной частотой, так же, как в цифровом сигнале, т. е. частотой дискретизации. Чтобы понять это, представим себе аналоговый сигнал в виде двух импульсов, следующих друг за другом с коротким промежутком времени. Если пропустить такой сигнал через фильтр нижних частот (удаляющий все высокочастотные компоненты), то два импульса сольются в один. Аналоговый сигнал, полоса частот которого лежит в пределах 0...10 кГц, будет иметь точно такое же разрешение, как цифровой сигнал, полученный на частоте дискретизации 20 кГц. Так и должно быть, ведь теорема отсчётов гарантирует, что такие два сигнала содержат одну и ту же информацию.

### 3.3. Аналоговые фильтры преобразования данных

На Рис. 3.7 показана блок-схема системы ЦОС, построенной в соответствии с теоремой отсчётов. Перед тем как поступить на АЦП, непрерывный сигнал обрабатывается *аналоговым фильтром* нижних частот, подавляющим все частотные компоненты, лежащие выше частоты Найквиста (половины частоты дискретизации). Этот фильтр необходим, чтобы избежать *наложения спектров* в процессе дискретизации, и называется *фильтром защиты от наложения спектров* или *антиэлайзинговым фильтром*. На другой стороне блок-схемы цифровой сигнал проходит



**Рис. 3.7** Использование аналоговых фильтров при построении системы в соответствии с теоремой отсчётов. Аналоговый фильтр, включённый до АЦП, называется фильтром защиты от наложения спектров или антиэлайзинговым фильтром. Он используется для подавления частотных компонент выше частоты Найквиста, которые бы вызвали наложение спектров при дискретизации. Аналоговый фильтр, расположенный после ЦАП, называется фильтром восстановления. Он тоже предназначен для подавления частотных компонент выше половины частоты дискретизации и может компенсировать эффекты применения фиксатора нулевого порядка.

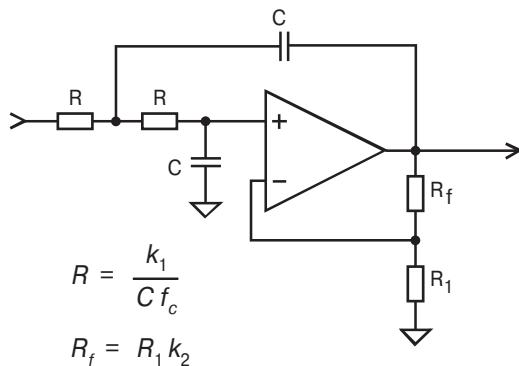
через *цифро-аналоговый преобразователь (ЦАП)* и ещё один аналоговый фильтр нижних частот, настроенный на частоту Найквиста. Этот выходной фильтр называется *фильтром восстановления* и может компенсировать последствия использования фиксатора нулевого порядка, о чём было сказано выше. К сожалению, эта простая модель системы имеет существенный недостаток: проблемы применения аналоговых фильтров оказываются не меньше тех проблем, которые эти фильтры пытаются решать.

Если вас интересует в первую очередь разработка программного обеспечения ЦОС, то вы можете подумать, что материал данного параграфа вам не нужен. И будете неправы. Даже если вы дали клятвенное обещание никогда не прикасаться к осциллографу, знание свойств аналоговых фильтров будет вам необходимо для успешного освоения ЦОС. Во-первых, характеристики любого цифрового сигнала, с которым вы будете работать, напрямую зависят от свойств антиэлайзингового фильтра, использованного при оцифровке сигнала. Не зная, как работает этот фильтр, вы не сможете понять структуру цифрового сигнала. Во-вторых, в своём развитии цифровая обработка идёт по пути замены аппаратных узлов программными средствами. Например, методика *многоскоростной обработки сигналов (передискретизации)* в процессе аналого-цифрового преобразования, описываемая далее в этой главе, уменьшает необходимость в антиэлайзинговом фильтре и фильтре восстановления за счёт использования оригинальной обработки, выполняемой программным методом. Не понимая, как работают аппаратные средства, вы не сможете спроектировать заменяющее их программное обеспечение. В-третьих, широко распространённым подходом к проектированию *цифровых фильтров* — одному из базовых вопросов ЦОС — является разработка эквивалентного аналогового фильтра, который затем реализуется программно. Изложение материала в последующих главах будет предполагать, что вам известны основы аналоговой фильтрации.

Наиболее распространены три типа аналоговых фильтров: *фильтры Чебышева, Баттерворта и Бесселя* (также известный как фильтр Томсона). Фильтры различаются между собой тем, на оптимизацию какого параметра они ориентированы. Сложность каждого фильтра определяется числом *полюсов* и *нулей* — математических понятий, которые будут обсуждаться в последующих главах. Чем больше полюсов имеет фильтр, тем сложнее его техническая реализация и тем выше качество его работы. То, к какому типу относится фильтр, говорит о характеристиках его работы, а не о том, какие резисторы и конденсаторы входят в его состав. Например, фильтр Бесселя с шестью полюсами может быть построен по многим различным схемам, однако все они будут одинаковы с точки зрения функционирования. Для цифровой обработки сигналов представляют интерес именно характеристики функционирования фильтра, а не то, каким способом он реализован. Тем не менее мы начнём с рассмотрения вопросов схемотехнической реализации этих фильтров, чтобы получить целостный взгляд на проблему.

На Рис. 3.8 представлена модифицированная схема *Саллена—Ки*, широко используемая при построении аналоговых фильтров. Своё название она получила по именам авторов опубликованной в 1950-х годах работы, описывающей принципы её применения. Представленная схема является фильтром нижних частот с двумя полюсами и может использоваться для разработки более сложных фильтров любого из перечисленных типов. Необходимые данные для выбора номина-

лов резисторов и конденсаторов для проектирования фильтров приведены в **Табл. 3.1**. Так, например, для построения фильтра Баттервортса с двумя полюсами и полосой 1 кГц необходимо, в соответствии с данными **Табл. 3.1**, использовать параметры  $k_1 = 0.1592$  и  $k_2 = 0.586$ . Установив значения  $R_1 = 10$  кОм и  $C = 0.01$  мкФ (общепринятые значения для операционных усилителей), можем вычислить величины  $R$  и  $R_f$ :  $R = 15.95$  кОм;  $R_f = 5.86$  кОм. Выбирая ближайшие номинальные значения по шкале с допуском 1%, получаем  $R = 15.8$  кОм;  $R_f = 5.90$  кОм. Все компоненты схемы должны быть прецизионными с допуском номиналов не более 1%.



**Рис. 3.8.** Модифицированная схема Саллена–Ки — базовый компонент, используемый при построении активных фильтров. Показанная структура соответствует фильтру нижних частот с двумя полюсами. Более сложные фильтры (с большим числом полюсов) могут быть получены каскадным соединением нескольких таких схем. При построении фильтров используйте значения  $k_1$  и  $k_2$  из **Табл. 3.1**, задавшись значениями  $R_1$  и  $C$  (например, типовыми  $R_1 = 10$  кОм и  $C = 0.01$  мкФ) и затем рассчитайте значения  $R$  и  $R_f$  по формулам, приведённым на рисунке. Величина  $f_c$  — это желаемая частота среза фильтра, выраженная в герцах.

**Таблица 3.1. Параметры для расчёта фильтров Бесселя, Баттервортса и Чебышева с 6%-й неравномерностью**

Число полюсов	Фильтр Бесселя		Фильтр Баттервортса		Фильтр Чебышева	
	$k_1$	$k_2$	$k_1$	$k_2$	$k_1$	$k_2$
2	цепь 1	0.1251	0.268	0.1592	0.586	0.1293
4	цепь 1	0.1111	0.084	0.1592	0.152	0.2666
	цепь 2	0.0991	0.759	0.1592	1.235	0.1544
6	цепь 1	0.0990	0.040	0.1592	0.068	0.4019
	цепь 2	0.0941	0.364	0.1592	0.586	0.2072
	цепь 3	0.0834	1.023	0.1592	1.483	0.1574
8	цепь 1	0.0894	0.024	0.1592	0.038	0.5359
	цепь 2	0.0867	0.213	0.1592	0.337	0.2657
	цепь 3	0.0814	0.593	0.1592	0.889	0.1848
	цепь 4	0.0726	1.184	0.1592	1.610	0.1582

Конкретный тип операционного усилителя, используемого в схеме, не имеет существенного значения при условии, что частота единичного усиления в 30...100 раз выше частоты среза фильтра. Это условие легко выполнимо, когда частоты среза фильтров лежат не выше 100 кГц.

Фильтры с числом полюсов 4, 6 и 8 образуются путём каскадного соединения 2, 3 и 4 таких цепей. На Рис. 3.9 показан пример соединения, реализующего фильтр Бесселя с шестью полюсами и образованного последовательным включением трёх цепей.

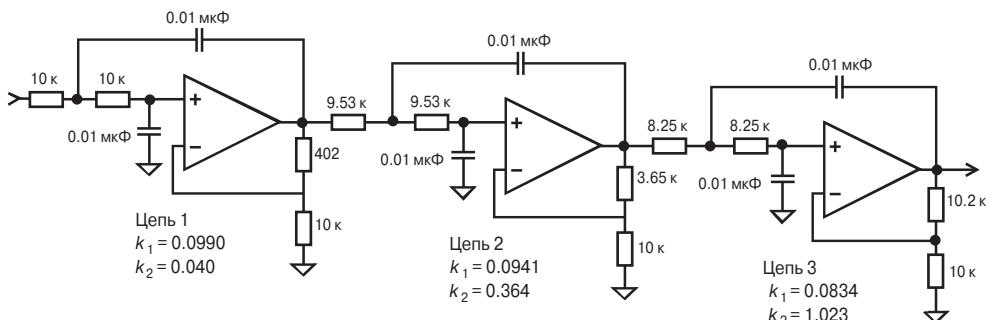


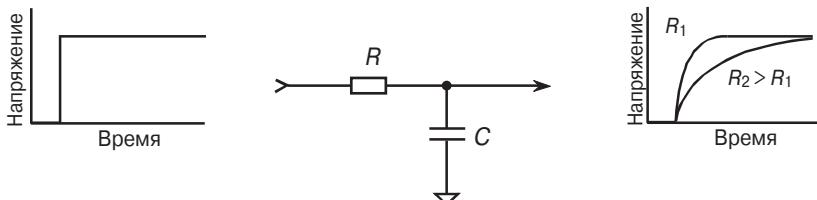
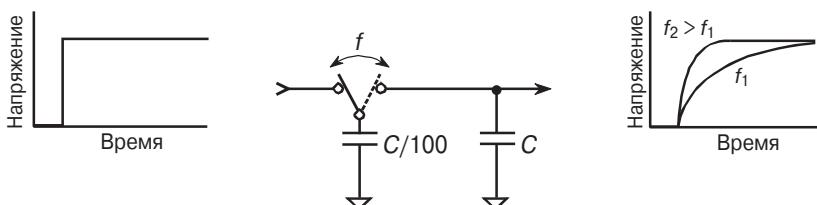
Рис. 3.9. Фильтр Бесселя с шестью полюсами, образованный каскадным соединением трёх базовых цепей. Это фильтр низких частот, рассчитанный на частоту среза 1 кГц.

В каждой цепи каскадного соединения используются, в соответствии с Табл. 3.1, свои параметры  $k_1$  и  $k_2$ , а следовательно, и значения сопротивлений и ёмкостей. Если вам нужен высокочастотный фильтр, просто поменяйте местами компоненты  $R$  и  $C$ , оставляя без изменений  $R_1$  и  $R_f$ .

Описанный способ реализации фильтров широко применяется при малоносерийном производстве и при проведении научно-исследовательских работ. Однако для крупномасштабного производства требуется изготовление устройств в виде интегральных схем. Здесь возникает проблема: разместить резисторы непосредственно на кристалле весьма затруднительно. Решение состоит в использовании фильтров на переключаемых конденсаторах. Рис. 3.10 иллюстрирует работу такого фильтра в сопоставлении с обычной  $RC$ -цепью. Если на вход  $HЧ$ -фильтра, представленного  $RC$ -цепью, подать ступенчатую функцию, на выходе будем наблюдать экспоненциальное возрастание сигнала до уровня входного напряжения. Напряжение на конденсаторе не может изменяться мгновенно, так как резистор ограничивает поток электрических зарядов.

В фильтре на переключаемых конденсаторах традиционная  $RC$ -цепь заменяется цепью с двумя конденсаторами и электронным переключателем. Добавленный конденсатор имеет намного меньшую ёмкость по сравнению с конденсатором, присутствовавшим в исходной  $RC$ -цепи (скажем, 1% его ёмкости). Переключатель замыкает этот конденсатор поочерёдно на вход и выход цепи, работая с большой частотой, обычно в 100 раз большей требуемой частоты среза фильтра. При замыкании на вход происходит быстрый заряд конденсатора малой ёмкости до значения входного напряжения. При замыкании на выход заряд с малого конденсатора «стекает» через конденсатор большей ёмкости. В случае использования резистора скорость разряда определялась его сопротивлением. При использовании схемы с переключаемыми конденсаторами она определяется ёмкостью малого конденсатора и частотой переключения. Это наделяет данный тип фильтров важным свойством: частота среза фильтра прямо пропорциональна тактовой частоте переключателей. Фильтры на переключаемых конденсаторах иде-

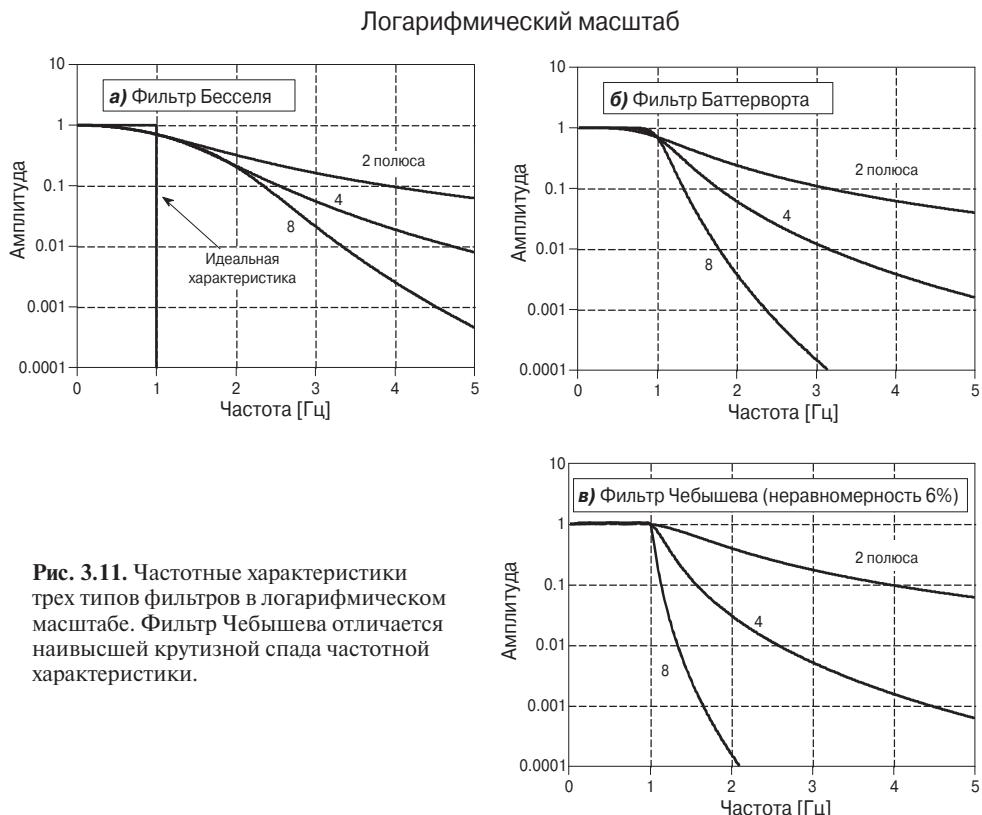
альны для применения в системах сбора данных, оперирующих несколькими частотами дискретизации. Это простые в использовании, дешёвые устройства, позволяющие, например, реализовать фильтр с восемью полюсами в одной микросхеме с восемью выводами.

*RC-цепь**Схема на переключаемых конденсаторах*

**Рис. 3.10.** Фильтр на переключаемых конденсаторах. Фильтры на переключаемых конденсаторах используют вместо резисторов  $RC$ -цепи переключатели и конденсаторы. Представленные на рисунке переходные характеристики схем показывают, что, в частности, схема на двух конденсаторах и одном переключателе эквивалентна  $RC$ -цепи.

Перейдём теперь к очень важному материалу: рассмотрим характеристики трёх базовых типов фильтров. Первым параметром, характеризующим эффективность фильтра, является  *крутизна спада частотной характеристики на частоте среза*. Фильтр низких частот должен подавлять частотные компоненты, лежащие выше частоты среза (в *полосе заграждения*), и пропускать частоты, лежащие ниже этой частоты (в *полосе пропускания*). На Рис. 3.11 в логарифмическом масштабе (в децибелах) показаны *частотные характеристики* трёх классических типов фильтров. Частота среза для всех фильтров взята равной 1 Гц, но характеристики будут выглядеть аналогично и для любой другой выбранной частоты, пропорционально растягиваясь или сжимаясь. Как можно оценить эти фильтры, исходя из представленных графиков? Фильтры Чебышева, безусловно, оказались лучше всех, фильтры Баттервортса существенно им уступают, а фильтры Бесселя совсем никуда не годятся! Как вы, возможно, уже догадались, фильтры Чебышева в данном случае имеют явное преимущество, поскольку они проектировались именно для обеспечения максимальной крутизны спада частотной характеристики.

К сожалению, даже фильтр Чебышева с восемью полюсами не имеет достаточно высоких показателей эффективности, чтобы использовать его в качестве *антиэлайзингового фильтра*. Покажем это на примере системы, использующей 12-битное представление чисел и частоту дискретизации входного сигнала 10 кГц. В соответствии с теоремой отсчётов все частотные компоненты входного сигнала



**Рис. 3.11.** Частотные характеристики трех типов фильтров в логарифмическом масштабе. Фильтр Чебышева отличается наивысшей крутизной спада частотной характеристики.

выше 5 кГц будут заворачиваться, вызывая наложение спектров, которого мы хотим избежать. Вспоминая выше изложенный материал, приходим к выводу, что нам необходимо использовать антиэлайзинговый фильтр, который бы уменьшал амплитуду всех частотных компонент выше 5 кГц не менее чем в 100 раз, обеспечивая остаточный уровень частот наложения не более 1%. Анализируя Рис. 3.11в, отмечаем, что фильтр Чебышева с восемью полюсами и частотой среза 1 Гц обеспечивает требуемое подавление лишь на частоте, приблизительно равной 1.35 Гц. Переводя график в масштаб, подходящий для рассматриваемого примера, получим, что для подавления частотных компонент, лежащих выше 5 кГц, в 100 раз необходимо, чтобы частота среза фильтра была равна 3.7 кГц. А это означает, что в полосе частот от 3.7 до 5 кГц будет наблюдаться нежелательное ослабление частотных компонент сигнала.

Здесь есть тонкий момент. Подавление заворачивающихся частотных компонент в 100 раз является достаточным для представления сигнала даже при использовании всех 12 бит. Как видно из Рис. 3.4, частота 5100 Гц будет заворачиваться в частоту 4900 Гц, а частота 6000 Гц — в частоту 4000 Гц и т. д. Можно не заботиться о том, каков уровень сигнала в полосе частот 5000...6300 Гц, т. к. эти частоты переходят при дискретизации в полосу неиспользуемых частот от 3700 до 5000 Гц. В полосу пропускания фильтра (0...3.7 кГц) при заворачивании будут попадать только частоты, лежащие выше 6300 Гц, что соответствует 1.7 частоты среза фильтра.

тра (3700 Гц). Из Рис. 3.11 $\nu$  следует, что затухание, обеспечиваемое фильтром Чебышева с восемью полюсами на частоте, равной 1.7 частоты среза, составляет порядка 1300, что обеспечивает гораздо большую точность, чем подавление, в 100 раз. Все эти рассуждения приводят нас к важному выводу: в большинстве систем полоса частот приблизительно 0.4...0.5 частоты дискретизации не используется и представляет собой переходную зону. Это является прямым следствием ограниченности возможностей аналоговых фильтров.

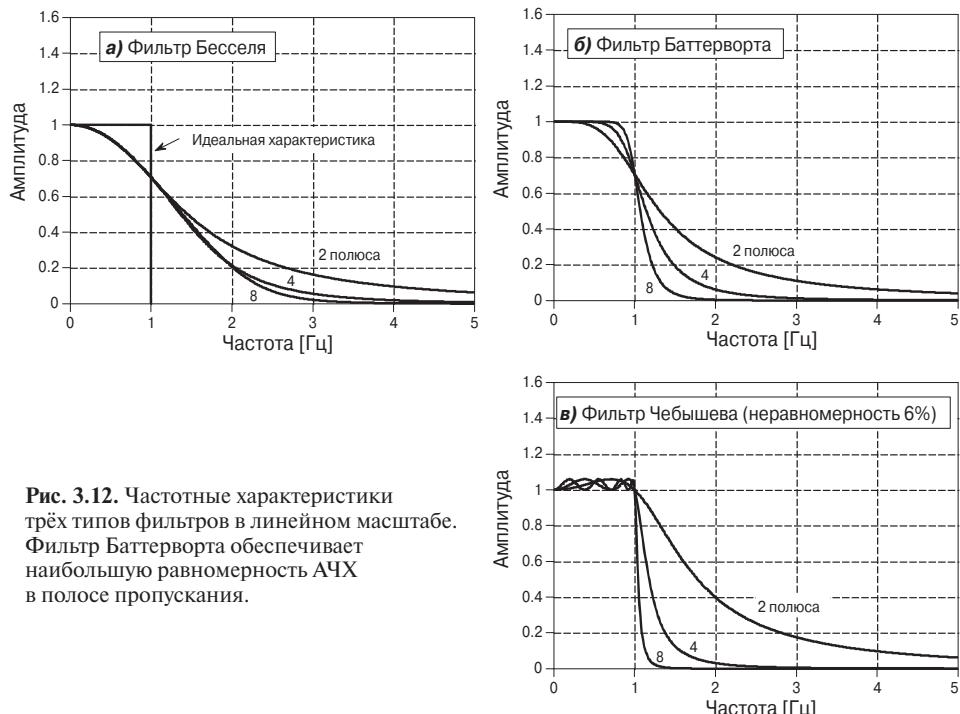
Идеальный фильтр низких частот имеет абсолютно плоскую амплитудно-частотную характеристику во всей полосе пропускания. На Рис. 3.11 все фильтры с этой точки зрения выглядят идеальными, однако только потому, что приводимые характеристики отображены в логарифмическом масштабе. Иная картина может наблюдаться при переходе к линейному масштабу отображения по вертикальной оси (см. Рис. 3.12). В этом случае в характеристике фильтра Чебышева ярко выражена *неравномерность в полосе пропускания* (колебания коэффициента передачи в зависимости от частоты). В действительности фильтр Чебышева обладает столь замечательными характеристиками крутизны спада за счёт того, что допускает большую неравномерность в полосе пропускания. Чем больше неравномерность в полосе пропускания фильтра, тем большей скорости спада можно достичь. Данные для расчёта фильтров Чебышева, приведённые в Табл. 3.1, допускают 6%-ю неравномерность в полосе пропускания (0.5 дБ), что является неплохим компромиссом и обычно используется на практике. Проектируемый похожим образом *эллиптический фильтр* допускает наличие неравномерности АЧХ и в полосе пропускания, и в полосе подавления. Этот фильтр позволяет достичь более удачного сочетания неравномерности и крутизны спада, однако проектирование его сложнее.

Сравнивая три типа фильтров, отметим, что фильтр Баттервортя является наилучшим с точки зрения обеспечения максимальной крутизны спада АЧХ при отсутствии неравномерности в полосе пропускания. Его часто называют *фильтром с максимально плоской характеристикой*. Он идентичен фильтру Чебышева, построенному для случая нулевой неравномерности в полосе пропускания. Фильтр Бесселя также не допускает неравномерности АЧХ в полосе пропускания, однако по крутизне спада частотной характеристики он существенно проигрывает фильтру Баттервортя.

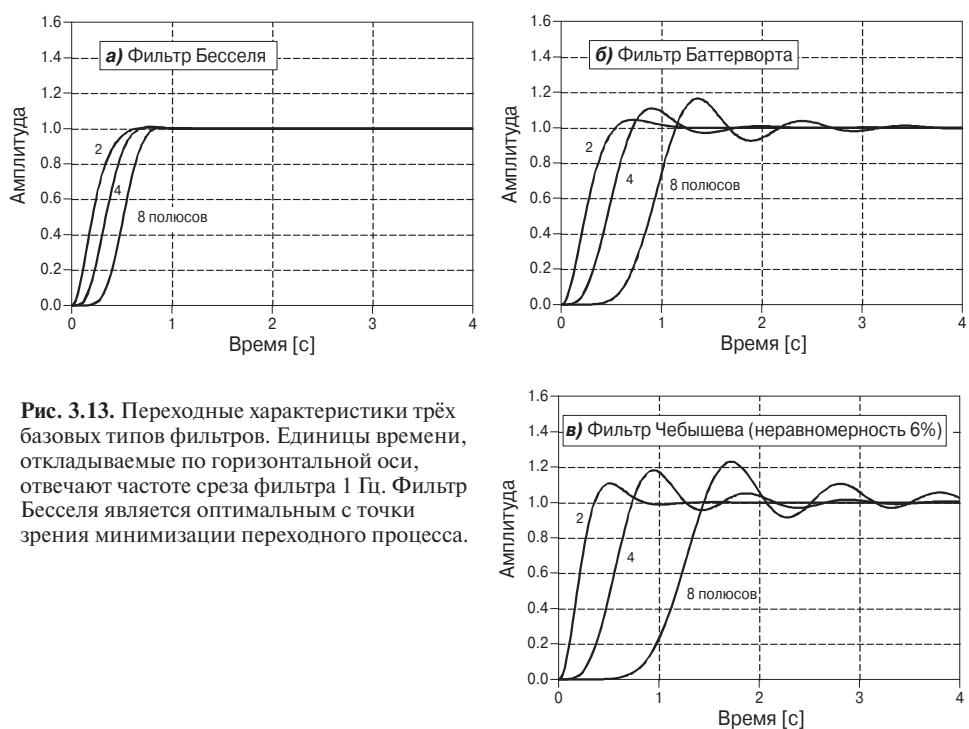
Еще одной характеристикой, которую нам следует рассмотреть, является *переходная характеристика* фильтра — реакция на ступенчатое воздействие, т. е. быстрый скачок уровня входного сигнала с одного значения на другое. Рис. 3.13 иллюстрирует переходные характеристики всех трёх типов фильтров. Рассматриваются фильтры с частотой среза 1 Гц, однако, как и ранее, приведённые характеристики будут справедливы и для других частот среза, растягиваясь или сжимаясь обратно пропорционально этим частотам. Например, для случая частоты среза 1000 Гц на том же графике оцифровка по горизонтальной оси просто будет проводиться в миллисекундах, а не в секундах. Фильтры Баттервортя и Чебышева характеризуются большим *перерегулированием* и наличием ярко выраженного *колебательного процесса*. В отличие от них, фильтр Бесселя не имеет этих негативных особенностей.

Рис. 3.14 ещё раз иллюстрирует достоинства временных характеристик фильтра Бесселя. На (а) показан импульсный сигнал, который может рассматриваться

## Линейный масштаб

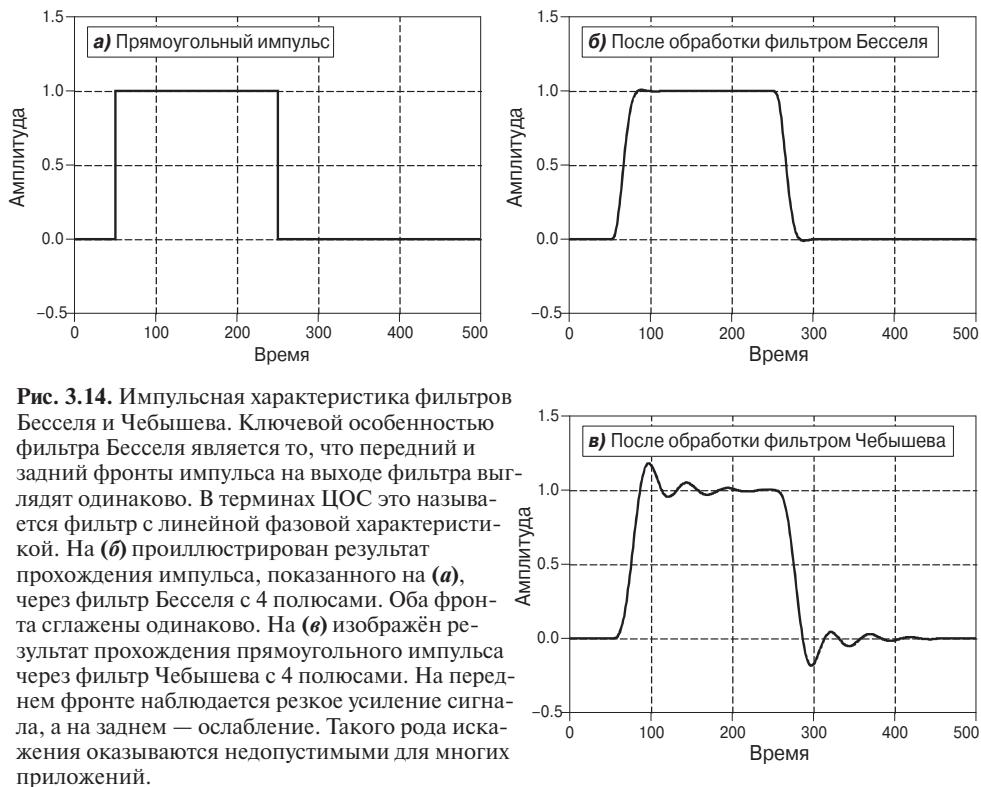


**Рис. 3.12.** Частотные характеристики трёх типов фильтров в линейном масштабе. Фильтр Баттервортса обеспечивает наибольшую равномерность АЧХ в полосе пропускания.



**Рис. 3.13.** Переходные характеристики трёх базовых типов фильтров. Единицы времени, откладываемые по горизонтальной оси, отвечают частоте среза фильтра 1 Гц. Фильтр Бесселя является оптимальным с точки зрения минимизации переходного процесса.

как пара ступенчатых воздействий: скачок уровня входного сигнала вверх и обратный скачок вниз. На (б и в) показаны результаты прохождения такого сигнала через фильтры Бесселя и Чебышева. Если бы, например, это был видеосигнал, то искажения, вносимые фильтром Чебышева, оказались бы разрушительными! Резкое усиление переднего фронта сигнала сильно изменило бы яркость контура объекта по сравнению с центральной частью. Более того, левый край объекта выглядел бы ярче, в то время как правый край — тусклее. Для многих приложений недопустимы такие искажения и требуется высокое качество переходной характеристики фильтра. И здесь фильтр Бесселя оказывается на высоте: отсутствуют колебания переходного процесса; края импульсной характеристики симметричны.



**Рис. 3.14.** Импульсная характеристика фильтров Бесселя и Чебышева. Ключевой особенностью фильтра Бесселя является то, что передний и задний фронты импульса на выходе фильтра выглядят одинаково. В терминах ЦОС это называется фильтр с линейной фазовой характеристикой. На (б) проиллюстрирован результат прохождения импульса, показанного на (а), через фильтр Бесселя с 4 полюсами. Оба фронта слажены одинаково. На (в) изображён результат прохождения прямоугольного импульса через фильтр Чебышева с 4 полюсами. На переднем фронте наблюдается резкое усиление сигнала, а на заднем — ослабление. Такого рода искажения оказываются недопустимыми для многих приложений.

## 3.4. Выбор антиэлайзингового фильтра

В Табл. 3.2 приводятся обобщённые характеристики фильтров трёх основных типов. Из таблицы видно, что, оптимизируя каждый из фильтров по одному из параметров, другие параметры при этом ухудшаются. *Фильтр Чебышева* имеет наилучшую крутизну спада частотной характеристики, *фильтр Баттерворта* — максимально плоскую АЧХ в полосе пропускания, а *фильтр Бесселя* обладает оптимальной переходной характеристикой.

Таблица 3.2. Характеристики трёх основных типов фильтров

Тип фильтра	Коэффициент передачи на 0-й частоте	Переходная характеристика			Частотная характеристика		
		Перегулирование [%]	Время установки перерегулирования 1%	Время установки перерегулирования 0.1%	Неравномерность в полосе пропускания [%]	Частота подавления в 100 раз	Частота подавления в 1000 раз
<b>Бесселя</b>	<b>2 полюса</b>	1.27	0.4	0.60	1.12	0	12.74
	<b>4 полюса</b>	1.91	0.9	0.66	1.20	0	4.74
	<b>6 полюсов</b>	2.87	0.7	0.74	1.18	0	3.65
	<b>8 полюсов</b>	4.32	0.4	0.80	1.16	0	3.35
<b>Баттерворт</b>	<b>2 полюса</b>	1.59	4.3	1.06	1.66	0	10.0
	<b>4 полюса</b>	2.58	10.9	1.68	2.74	0	3.17
	<b>6 полюсов</b>	4.21	14.3	2.74	3.92	0	2.16
	<b>8 полюсов</b>	6.84	16.4	3.50	5.12	0	1.78
<b>Чебышева</b>	<b>2 полюса</b>	1.84	10.8	1.10	1.62	6	12.33
	<b>4 полюса</b>	4.21	18.2	3.04	5.42	6	2.59
	<b>6 полюсов</b>	10.71	21.3	5.86	10.4	6	1.63
	<b>8 полюсов</b>	28.58	23.0	8.34	16.4	6	1.34

Фильтр Бесселя имеет наилучшую переходную характеристику, что делает его предпочтительным для обработки сигналов с временным кодированием. Фильтры Чебышева и Баттерворта используются для подавления частот в зоне заграждения и лучше подходят для обработки сигналов с частотным кодированием. Значения, приведённые в таблице, измеряются в секундах и герцах. Рассматривается случай фильтров с частотой среза 1 Гц.

Выбор того или иного типа для реализации *антиэлайзингового фильтра* практически полностью зависит от того, каким образом информация представлена в обрабатываемом сигнале. Несмотря на то что существует множество способов преобразования информации в *сигнал*, можно выделить два основных метода: *кодирование во временной области* и *кодирование в частотной области*. Различия этих двух методов играют большую роль в ЦОС, поэтому их обсуждение будет повторяться не раз на страницах этой книги.

В случае кодирования в частотной области информация закладывается в частоты гармонических колебаний, которые суммируются, образуя сигнал. Аудиосигналы являются отличным примером такого кодирования. Когда человек слышит речь или музыку, воспринимаемый им звук определяется набором частот, входящих в его состав, а не формой суммарного сигнала. Это можно доказать, пропустив такой сигнал через устройство, изменяющее фазы отдельных гармоник, не меняя их частоты и амплитуды. Сигнал на выходе, наблюдаемый на экране осциллографа, окажется абсолютно другим по своей форме, однако звучать он будет по-прежнему. Несмотря на то что внешний вид сигнала сильно изменился, заключенная в нём информация осталась нетронутой. Так как при *наложении спектров* в процессе *дискретизации* происходит искажение частотных компонентов, то эффекты *заворачивания частот* особенно опасны для сигналов с кодированием в частотной области. Следовательно, для таких сигналов особенно важно при *аналого-цифровом преобразовании* использовать антиэлайзинговый фильтр с крутой частотной характеристикой в переходной зоне, т. е. *фильтр Чебышева, эллиптический фильтр* или *фильтр Баттерворт*. Вы спросите: «А как же ужасные переходные характеристики этих фильтров?» А они не имеют значения: информация, зашифрованная в частотной области, не подвержена такого рода искажениям.

При кодировании во *временной области* информация, напротив, закладывается в форму сигнала. К примеру, в медицине электрическая активность сердца контролируется с помощью электродов, прикладываемых к груди и рукам человека (электрокардиограмма, или ЭКГ). Форма сигналов ЭКГ несёт интересующую врачей информацию о функционировании разных отделов сердца. Другой пример подобных сигналов — *изображения*. Если обычный сигнал представляется формой колебаний, изменяющихся во времени, то изображение можно рассматривать как сигнал, изменяющийся в пространстве от точки к точке (от пикселя к пикселю). Изображение формируется именно формой этого колебания — отдельными участками определённой яркости и цвета и переходами одних участков в другие. Глядя на «Мону Лизу», вы не станете говорить: «Ух ты, смотри какой замечательный набор гармоник!»

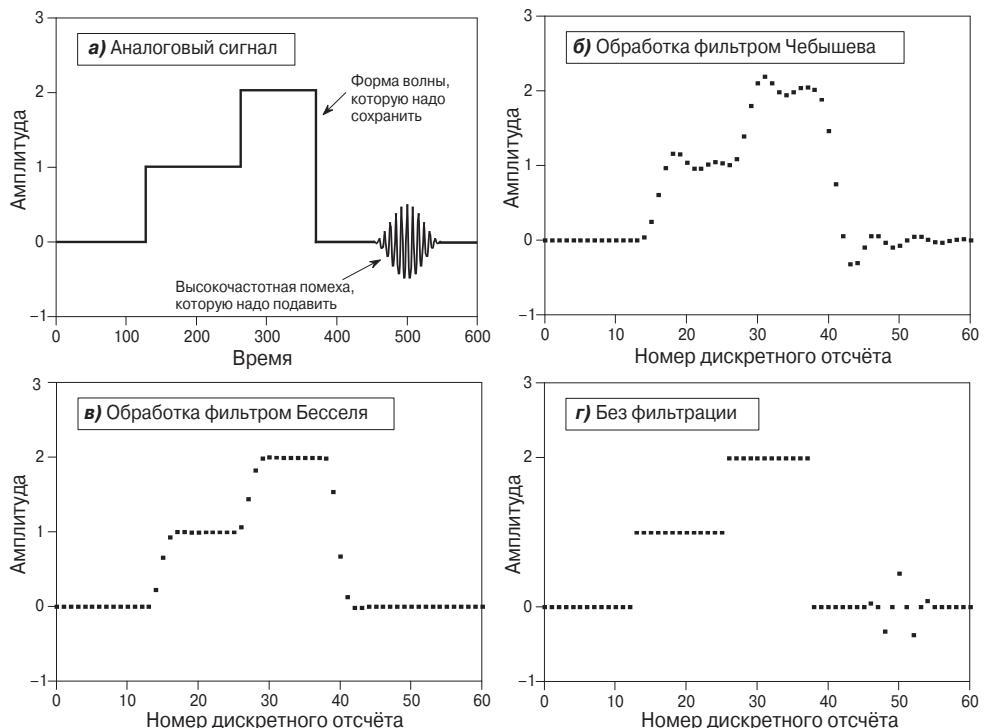
*Теорема отсчётов* отвечает на вопрос, что происходит с сигналом в *частотной области* при его дискретизации. Она идеальна для анализа процесса аналого-цифрового преобразования сигналов с кодированием в частотной области. В то же время она ничего не говорит о правилах дискретизации сигналов, которые несут информацию во временной области. Рассмотрим эту проблему более детально.

**Рис. 3.15** иллюстрирует варианты аналого-цифрового преобразования сигнала с кодированием во временной области. На **(а)** представлен аналоговый сигнал, дискретизацию которого мы осуществляем. Нас будет интересовать информация, заложенная в форме прямоугольных импульсов. В сигнал включено короткое по длительности высокочастотное гармоническое колебание. Такие колебания всегда присутствуют в аналоговых сигналах, являясь следствием аддитивных широкополосных шумов. На других рисунках показан вид цифрового сигнала, полученного при использовании антиэлайзинговых фильтров Чебышева, Бесселя и в ситуации, когда антиэлайзинговый фильтр отсутствует.

Важно отметить, что независимо от типа антиэлайзинговой фильтрации восстановление исходного сигнала по дискретным отсчётам в данном примере будет невозможно. Аналоговый сигнал содержит частотные компоненты, лежащие выше половины частоты дискретизации. Цифровой сигнал не может включать эти частоты, а следовательно, они не будут входить и в состав восстановленного непрерывного сигнала. Частоты, о которых идёт речь, присутствуют в исходном аналоговом сигнале, вследствие двух факторов: 1) первый источник высокочастотных компонент — это шум, от которого нам было бы полезно избавиться; 2) второй источник — это скачкообразное изменение формы волны на краях прямоугольных импульсов — информация, которую мы хотели бы сохранить.

Фильтр Чебышева **(б)** подавляет все частотные компоненты, лежащие выше частоты Найквиста. В результате мы получаем сигнал, который может быть оцифрован и впоследствии полностью восстановлен. Однако этот восстановленный сигнал окажется идентичен аналоговому сигналу на выходе антиэлайзингового фильтра, а не исходному сигналу. Информация не была искажена при дискретизации, однако она была потеряна в процессе фильтрации. Лекарство оказалось страшнее болезни! Не следует его применять!

На решение рассматриваемой задачи полностью ориентирован *фильтр Бесселя* **(в)**. Сигнал на его выходе повторяет форму входного сигнала, отличаясь несколько сглаженными краями прямоугольных импульсов. Регулируя частоту среза фильтра, можно подобрать оптимальное соотношение степени сглаживания на



**Рис. 3.15.** Три варианта антиэлайзинговой фильтрации сигналов с кодированием во временной области. Целью фильтрации является подавление высокочастотных компонент, заворачивающихся при дискретизации, и одновременное сохранение ступенчатой формы сигнала, в которой заключена информация. На (а) изображён аналоговый сигнал ступенчатой структуры, который также содержит высокочастотный шумовой выброс. На (б) показан результат аналого-цифрового преобразования в случае использования Чебышевского антиэлайзингового фильтра. Шум на высоких частотах полностью удалён, однако края ступеней оказались сильно искажёнными. Такое решение нельзя считать приемлемым. Фильтр Бесселя (в), обеспечивает в определённой степени, одновременное сглаживание ступеней и подавление высоких частот. В случае, когда фильтрация не используется (г), форма сигнала оказывается полностью сохранённой, однако неудалённые высокие частоты вызывают появление искажённых отсчётов в цифровом сигнале.

краях и амплитуды высокочастотных колебаний. Увеличение числа полюсов фильтра позволит добиться лучшего компромисса между этими двумя параметрами. Общим правилом здесь является выбор частоты среза фильтра, равной  $1/4$  частоты дискретизации. В этом случае ступенчатые перепады формы сигнала представляются двумя дискретными отсчётами. Заметим, что как фильтр Чебышева, так и фильтр Бесселя позволяют избавиться от высокочастотного колебания-шума.

Отсутствие какой-либо фильтрации для защиты от наложения спектров также можно рассматривать как один из вариантов построения системы (г). Бесспорным достоинством при этом окажется идентичность дискретных отсчётов значениям аналогового сигнала. То есть крутизна перепадов на краях ступенек будет сохранена, поскольку изменение амплитуды непрерывного сигнала мгновенно вызывает изменение уровня дискретного сигнала. Недостатком метода является

возможность искажения сигнала при заворачивании спектра. Эффекты наложения спектров здесь могут принимать две различные формы. При первой помехи, присутствующие на высоких частотах, такие как рассматриваемое в данном примере высокочастотное колебание, заворачиваются и формируют новые, не имеющие смысла отсчёты (*ε*). Другими словами, весь высокочастотный шум аналого-вого сигнала будет присутствовать в частотном диапазоне цифрового сигнала. При более общем подходе этот эффект нарастания шума должен рассматриваться не как следствие *дискретизации*, а как следствие некачественной предварительной аналоговой обработки. В задачи АЦП не входит борьба с помехами и шумами. За это отвечают аналоговые устройства, стоящие до АЦП. Если для решения этой проблемы используется, например, фильтр Бесселя, то он будет рассматриваться как одна из процедур аналоговой обработки, а не как этап аналого-цифрового преобразования.

Вторая форма проявления эффектов *наложения спектров* при дискретизации является менее очевидной. Изменения формы аналогового сигнала, такие как ступенчатые скачки амплитуды, отражаются в цифровом сигнале уже в следующем дискретном отсчёте. При этом в цифровом сигнале нет информации о поведении непрерывного сигнала в моменты между двумя соседними отсчётами. Сравним теперь вариант использования фильтра Бесселя с ситуацией, когда фильтрация не применяется. Мысленно соединим соседние отсчёты на (*ε*) прямыми линиями. Момент, в который такая линия пересекает уровень половины высоты ступени, даёт нам оценку времени (промежуточный отсчёт), когда скачок происходит в аналоговом сигнале. Если фильтрация не используется, то промежуточных отсчётов нет, и оценить момент начала ступени оказывается сложно. Чтобы решить, насколько существенен этот момент, необходимо хорошо себе представлять, что вы собираетесь делать с входными данными.

## 3.5. Многоскоростная обработка в процессе аналого-цифрового преобразования

В современной электронике наблюдается устойчивая тенденция замены аналоговых узлов программными решениями. Преобразования формы сигнала из непрерывной в цифровую и обратно являются отличной иллюстрацией этого. Рассмотрим пример цифрового устройства воспроизведения голоса (цифрового диктофона) — системы, которая осуществляет *аналого-цифровое преобразование* входного голосового сигнала, записывает его в цифровой форме в память и впоследствии воспроизводит. Чтобы получить более или менее приемлемое качество восстановленного речевого сигнала, необходимо иметь информацию в полосе частот от 100 до 3000 Гц. В то же время входной *анalogовый сигнал* с микрофона лежит в гораздо более широком диапазоне частот до 40 кГц. Решая проблему «в лоб», можно пропустить сигнал через низкочастотный фильтр Чебышева 8-го порядка с частотой среза 3 кГц и произвести *дискретизацию* на частоте 8 кГц. Затем *цифро-аналоговый преобразователь* с помощью фиксатора нулевого порядка сформирует непрерывный сигнал, а ещё один фильтр Чебышева с частотой среза 3 кГц восстановит исходный аналоговый сигнал на выходе системы.

Однако можно использовать другой подход — так называемую *передискретизацию*, т. е. дискретизацию сигнала с частотой, большей, чем того требует *теорема отсчётов*. Этот подход имеет много достоинств. Пусть, например, в рассматриваемой системе *частота дискретизации* выбрана равной 64 кГц. Задача антиэлайзингового фильтра в этом случае сразу же упрощается. Теперь он должен пропускать частоты ниже 3 кГц и подавлять частоты выше 32 кГц. Аналогично упрощается и фильтр восстановления. Использование передискретизации позволяет заменить сложные аналоговые фильтры простыми *RC*-цепочками. Однако на вход системы цифровой обработки теперь поступает огромный поток данных на высокой частоте.

Дальнейшим уровнем усовершенствования системы является применение методов *многоскоростной обработки сигналов*, основанных на использовании нескольких различных частот дискретизации в одной системе. Для нашего примера это будет означать следующее. Голосовой сигнал проходит через фильтр нижних частот, представленный *RC*-цепочкой, и дискретизируется с частотой 64 кГц. Полученный цифровой сигнал, кроме требуемой полосы частот 100...3000 Гц, включает неинформативную полосу 3...32 кГц. Далее цифровой фильтр нижних частот, реализованный программно, выполняет подавление частот, лежащих выше 3 кГц. На следующем этапе происходит понижение частоты дискретизации с 64 до 8 кГц путём простого отбрасывания каждого 7 из 8 последовательно идущих дискретных отсчётов. Эта процедура называется *децимацией*. Формируемый таким образом цифровой сигнал полностью идентичен сигналу, получаемому с применением сложного антиэлайзингового фильтра и дискретизируемому непосредственно на частоте 8 кГц.

Многоскоростная обработка применима в нашем примере и для формирования выходного сигнала. Данные, характеризуемые частотой дискретизации 8 кГц, извлекаются из памяти и преобразуются в сигнал с частотой дискретизации 64 кГц. Эта процедура повышения частоты дискретизации называется *интерполяцией* и состоит в дополнении сигнала семью элементами, например нулевыми, между каждыми двумя соседними отсчётами. В результате получаем импульсный портрет сигнала, в спектре которого содержатся полезные частоты в диапазоне 100...3000 Гц и их периодически повторяющиеся копии в полосе 3...32 кГц. Рис. 3.6а и в поясняют такой результат. Все частотные компоненты, лежащие выше 3 кГц, далее подавляются цифровым низкочастотным фильтром. После перехода к непрерывному сигналу с помощью ЦАП в качестве устройства аналоговой обработки используется простая *RC*-цепь, на выходе которой получаем восстановленный речевой сигнал.

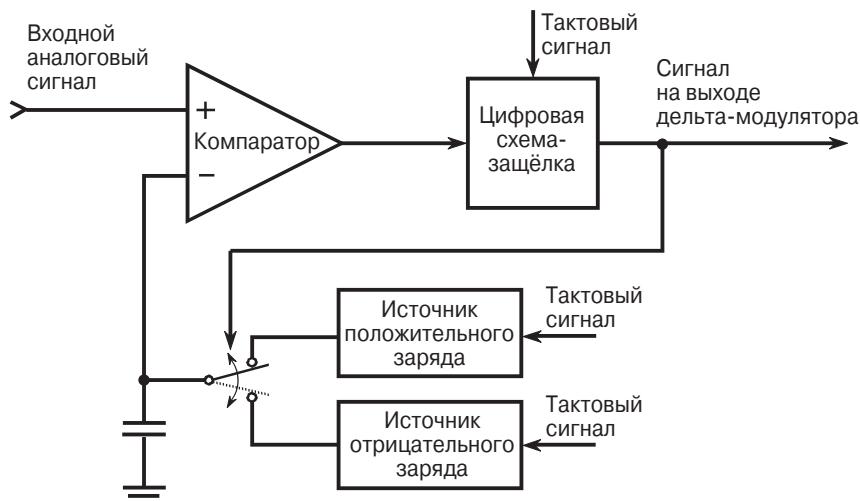
Применение многоскоростной обработки сигналов имеет два преимущества: во-первых, она позволяет заменять аналоговые компоненты программными алгоритмами — очевидная экономическая выгода для случая массового производства; во-вторых, она открывает возможность достижения нового уровня производительности системы. Так, например, аудиопроигрыватели компакт-дисков используют многоскоростную обработку для повышения качества звучания. Это оказывается возможным за счёт замены аналоговых компонентов (с 1%-м разбросом номиналов) на алгоритмы цифровой обработки (с 0.0001%-й точностью, ограниченной ошибками округления). Как будет показано в последующих главах, цифровые фильтры в сотни раз превосходят аналоговые по своим основным характеристикам.

## 3.6. Однобитные аналого-цифровое и цифро-аналоговое преобразования

В телекоммуникационном оборудовании и аудиосистемах высокого качества широкое распространение получили так называемые *однобитные АЦП* и *ЦАП*. Их функционирование основано на принципах *многоскоростной обработки* и состоит в уменьшении разрядности в пределе до одного бита за счёт *передискретизации* на повышенной частоте. Существует много различных вариантов реализации таких систем, большинство из которых использует *дельта-модуляцию*. Рассмотрим три примера, чтобы проиллюстрировать суть метода. Все три схемы в действительности реализуются в виде интегральных схем (ИС), и задаваться вопросом, где и какие транзисторы или операционные усилители надо размещать, не стоит. Вам не придётся собирать эти схемы из набора электронных компонентов.

На Рис. 3.16 показана типовая схема дельта-модулятора. Пусть аналоговый речевой сигнал на входе схемы имеет амплитуду в несколько вольт. Цифровой сигнал на выходе представляет собой поток нулей и единиц. Входящий в систему компаратор сравнивает уровень входного сигнала с напряжением на конденсаторе и выдает решение в виде цифровых нуля или единицы, поступающих на схему-защёлку. Последняя осуществляет передачу состояния со своего входа на свой выход на частоте тактирования, обычно равной нескольким сотням килогерц. Схема-защёлка нужна для того, чтобы обеспечить синхронизацию выходного сигнала с тактовыми импульсами, т. е. чтобы сформировать выходной сигнал на заданной частоте дискретизации. С этой частотой может изменяться состояние выходного бита данных.

В схеме присутствует петля обратной связи: цифровой сигнал с выхода системы подается на электронный переключатель. Если сигнал на выходе соответствую-



**Рис. 3.16.** Схема дельта-модулятора. Компаратор осуществляет сравнение уровня входного сигнала с напряжением на конденсаторе и выдает решение в виде двоичного нуля или единицы на схему-защёлку. Выход этой схемы синхронизируется с тактовыми импульсами и подаётся в контур обратной связи. Обратная связь заставляет напряжение на конденсаторе «следить» за уровнем входного сигнала.

ет уровню двоичной единицы, то конденсатор подключается к схеме источника положительного заряда, увеличивающей напряжение на конденсаторе на определённую величину, например 1 мВ, за период тактирования. В простейшем случае таким источником может быть резистор, на который подано большое напряжение. Если уровень сигнала на выходе соответствует уровню двоичного нуля, то конденсатор подключается к источнику отрицательного заряда, уменьшающему напряжение на конденсаторе на ту же фиксированную величину.

На Рис. 3.17 показаны сигналы, иллюстрирующие работу схемы. В нулевой момент времени уровень входного аналогового сигнала и напряжение на конденсаторе равны нулю. На восьмом такте уровень входного сигнала поднимается скачком до 9.5 В (а). Так как входное напряжение начинает превышать напряжение на конденсаторе, то уровень цифрового сигнала на выходе схемы устанавливается в единицу (б). Это, в свою очередь, заставляет переключатель подсоединить конденсатор к источнику положительного заряда, и напряжение на нём постепенно увеличивается на определённую величину за каждый такт. На Рис. 3.17а для иллюстрации показан случай, когда скорость нарастания напряжения составляет 1 В за такт. Более типичным является увеличение напряжения на 1 мВ за такт. Ступенчатое нарастание напряжения на конденсаторе продолжается

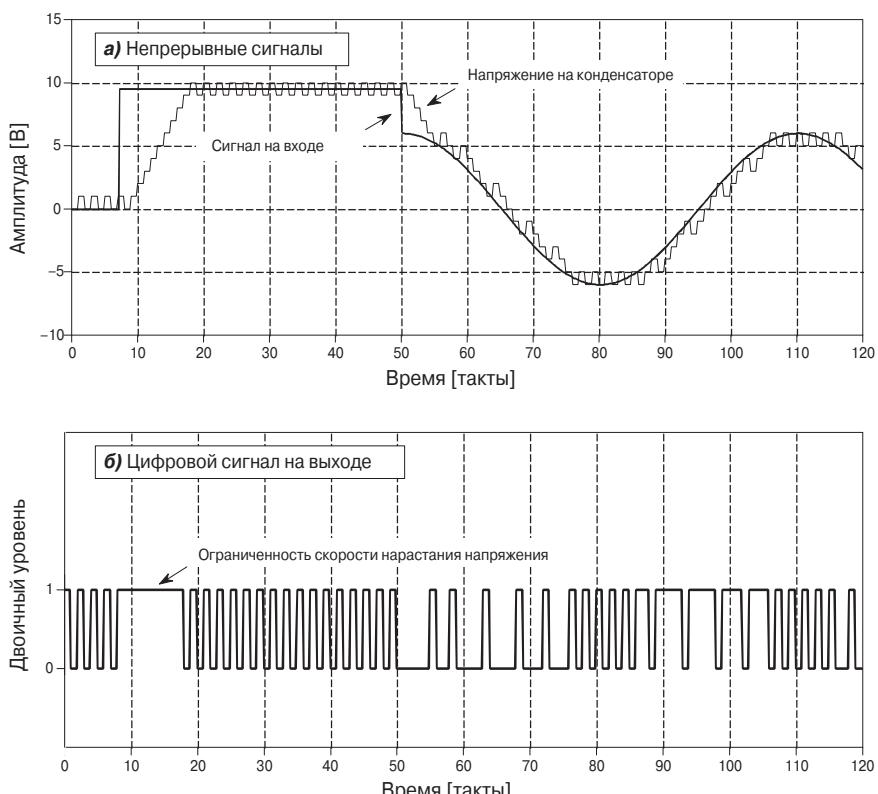


Рис. 3.17. Пример сигналов, иллюстрирующих работу схемы дельта-модулятора, представленного на Рис. 3.16. Вид непрерывного входного сигнала и сигнала на конденсаторе показан на (а). Цифровой сигнал на выходе дельта-модулятора, представляющий собой поток нулей и единиц, изображён на (б).

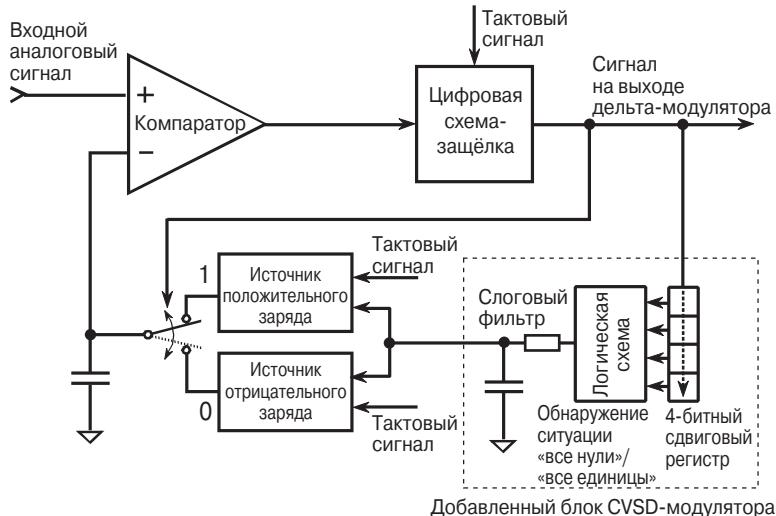
до тех пор, пока оно не превысит напряжение на входе схемы. Система входит в уравновешенное состояние. Выходной сигнал поочерёдно принимает значения нуля и единицы, заставляя напряжение на конденсаторе колебаться между 9 и 10 В. Таким образом, благодаря петле обратной связи напряжение на конденсаторе осуществляет «сложение» за уровнем входного сигнала. Скорость изменения напряжения на конденсаторе является постоянной величиной, и если напряжение на входе меняется скачком, то системе требуется время, чтобы перейти в уравновешенное состояние. Скорость, с которой меняется напряжение на конденсаторе, называется *скоростью нарастания выходного напряжения* аналогично другим электронным устройствам, например операционным усилителям.

Рассмотрим теперь характеристики сигнала на выходе дельта-модулятора. Если уровень аналогового сигнала увеличивается, в цифровом сигнале преобладают единицы. Если уровень сигнала уменьшается, начинают преобладать нули. В случае, когда напряжение на входе имеет постоянное значение, цифровой выходной сигнал переключается между нулём и единицей, количество которых в сигнале оказывается одинаковым. В целом можно сказать, что отношение числа единиц к числу нулей в выходном цифровом сигнале прямо пропорционально скорости изменения входного аналогового сигнала (его производной).

Рассмотренный метод преобразования аналогового сигнала в поток двоичных нулей и единиц в случае применения для систем передачи и хранения цифровых данных отличается малыми затратами на реализацию. Существенным достоинством является то, что все биты оказываются равнозначными в отличие от традиционного формата представления последовательных данных: *старт-бит*, *младший бит*, ..., *старший бит*, *стоп-бит*. Схема приёмника в части петли обратной связи строится аналогично схеме передатчика. Напряжение на конденсаторе в приёмнике осуществляет сложение за входным сигналом точно так же, как и в передатчике. Таким образом, показанный на Рис. 3.17 $a$  график напряжения на конденсаторе также иллюстрирует процесс восстановления аналогового сигнала.

Ключевым недостатком данной схемы является неизбежность поиска компромисса между противоречавшими друг другу параметрами — максимальной скоростью нарастания выходного напряжения, числом уровней квантования и тактовой частотой. В частности, если максимальная скорость нарастания напряжения и число уровней квантования выбраны удовлетворяющими требованиям систем связи, то тактовая частота должна будет лежать в мегагерцовой области, что является слишком большой величиной, экономически не оправданной для этого типа приложений. При обычном подходе к дискретизации речевого сигнала скорость преобразования составляет порядка 64 000 бит в секунду (тактовая частота — 64 кГц).

Схема, в которой эта проблема решена, показана на Рис. 3.18. Это *дельта-модулятор с непрерывным изменением крутизны* (*Continuously Variable Slope Delta (CVSD) modulator*), используемый, например, в семействе операционных усилителей MC3518 производства фирмы Motorola. В такой схеме выбираются приемлемые значения тактовой частоты и количества уровней квантования, например 30 кГц и 2000. Для коррекции скорости нарастания выходного напряжения используют дополнительную схему. Принцип её действия состоит в следующем. С помощью регистра сдвига осуществляется постоянное сложение за состоянием последних четырёх битов на выходе. Если система входит в состояние, когда начинает сказываться ограниченность скорости нарастания напряжения, все по-



**Рис. 3.18.** Схема дельта-модулятора с непрерывным изменением крутизны. По сравнению с базовым дельта-модулятором добавлена логическая схема, позволяющая изменять скорость нарастания выходного напряжения на конденсаторе.

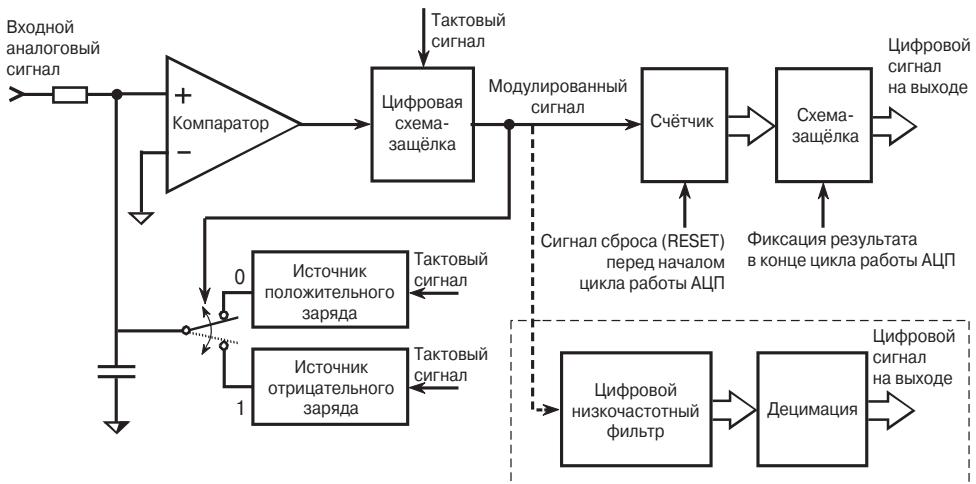
следние четыре бита на выходе будут либо единицами (в случае нарастания уровня входного сигнала), либо нулями (в случае падения уровня входного сигнала). Логическая схема детектирует данное состояние и генерирует аналоговый сигнал, увеличивающий величину напряжения, подаваемого на конденсатор с источника положительного или отрицательного напряжения. Это повышает скорость нарастания выходного напряжения, увеличивая шаг, с которым напряжение на конденсаторе изменяется на каждом тактовом импульсе.

Между логической схемой и источниками положительного и отрицательного зарядов обычно помещают аналоговый фильтр. Он используется для регулирования шага нарастания выходного напряжения в зависимости от того, как долго находится система в состоянии «погони» за изменившимся уровнем входного сигнала. Пока система находится в этом состоянии, скорость нарастания напряжения всё больше увеличивается. Часто этот фильтр называют *слоговым фильтром*, потому что его характеристики зависят от средней длины слогов, составляющих речевые сигналы. Путём соответствующей оптимизации параметров системы (это делают фирмы-производители микросхем в соответствии со своими требованиями) можно получить приемлемое качество речи на частотах в полосе 16...32 кГц. Постоянное изменение шага нарастания/спада напряжения на конденсаторе делает цифровые данные трудно читаемыми, однако это и не нужно. На стороне приёмника, восстанавливающего исходный аналоговый сигнал, в схему включается слоговой фильтр, полностью идентичный фильтру, используемому в схеме передатчика. При хорошей согласованности этих фильтров искажения, вносимые дельта-модулятором с непрерывным изменением крутизны, оказываются незначительными. Этот тип схемы дельта-модулятора, вероятно, является наиболее удобным для передачи речевых сигналов в цифровом виде.

Хотя идеально подходит для кодирования речи, дельта-модулятор с непрерывным изменением крутизны оказывается непригодным для реализации анало-

го-цифровых преобразователей общего назначения. Несмотря на то что удаётся избежать зависимости от скорости изменения сигнала на входе, происходит искашение информации, обусловленное переменной скоростью нарастания напряжения. Восстановить информацию становится невозможным. Кроме того, данная схема не преобразует в цифровую форму постоянную величину, присутствующую в сигнале.

Этих недостатков лишён *сигма-дельта АЦП*, структура которого показана на Рис. 3.19. В его основе лежит сочетание аналоговой электроники и цифровых алгоритмов. Напряжение на конденсаторе сравнивается теперь с потенциалом земли. Петля обратной связи вызывает уменьшение напряжения на конденсаторе при единичном состоянии выхода схемы и увеличение напряжения — при нулевом состоянии. Увеличение или уменьшение уровня входного сигнала вызывает повышение или понижение напряжения на конденсаторе. Изменение напряжения детектируется компаратором, который инициирует подключение конденсатора к источнику соответствующего отрицательного или положительного заряда. Конденсатор подвергается компенсирующему влиянию, и напряжение на нём остаётся нулевым.



**Рис. 3.19.** Схема сигма-дельта АЦП. В простейшем случае реализации на выходе дельта-модулятора подсчитывается число импульсов в сигнале. Подсчёт ведётся в течение заранее установленного числа тактов. Выход счётчика затем фиксируется и формирует результат преобразования. В случае более сложной реализации импульсы проходят через цифровой низкочастотный фильтр. На выходе фильтра ставится блок децимации для понижения частоты дискретизации выходного сигнала.

В случае положительного входного напряжения в выходном цифровом сигнале преобладают двоичные единицы. Они необходимы, чтобы компенсировать разность потенциалов на конденсаторе с помощью источника отрицательного заряда. Их преобладание продолжается до тех пор, пока входной сигнал не поменяет знак. Аналогично в случае отрицательного напряжения на входе схемы в выходном цифровом сигнале наблюдается большое количество двоичных нулей, подключающих конденсатор к источнику положительного заряда. Если уровень входного сигнала равен нулю, то число единиц и нулей на выходе оказывается

одинаковым. Поочерёдное подключение конденсатора к источнику положительного и отрицательного зарядов приводит к тому, что результирующее напряжение на нём вновь оказывается равным нулю.

Отношение числа единиц к числу нулей на выходе схемы в данном случае пропорционально самому сигналу, а не его производной, как в предыдущем примере. Такое решение оказывается намного удобнее. Например, такую схему легко можно использовать для формирования 12-битного АЦП. Для этого достаточно подключить к её выходу счётчик, подсчитывающий число двоичных единиц, появляющихся в течение каждого 4096 тактов. Максимальному положительному напряжению на выходе в этом случае будет соответствовать число 4095, максимальному отрицательному напряжению — число 0, а нулевому уровню входного сигнала — число 2048. Это также указывает на происхождение самого названия сигма-дельта модулятора: это дельта-модулятор с последующим подсчётом, т. е. суммированием (поэтому в название добавлено «сигма»).

Цифровой сигнал в виде потока нулей и единиц, полученный с помощью сигма-дельта АЦП, очень просто преобразовать обратно в непрерывный сигнал. Достаточно пропустить его через аналоговый фильтр нижних частот, реализованный в виде простой одиночной  $RC$ -цепи. Высокий и низкий уровни напряжения, соответствующие двоичным единицам и нулям, усредняются и формируют восстановленный непрерывный сигнал. Допустим, единице соответствует напряжение 5 В, а нулю — 0 В. Если цифровой сигнал на 80% состоит из двоичных единиц и на 20% из двоичных нулей, то на выходе низкочастотного фильтра будем иметь сигнал напряжением 4 В.

Описанный метод восстановления непрерывного сигнала из потока двоичных нулей и единиц имеет важное значение по нескольким причинам. Главная из них состоит в том, что появляется интересная возможность заменить счётчик в схеме сигма-дельта АЦП процедурой децимации. Двоичный сигнал с выхода дельта-модулятора поступает на цифровой низкочастотный фильтр, после чего децимируется — переносится на пониженную частоту дискретизации. Данная процедура может начинаться с представления каждого двоичного нуля или единицы в виде 12-битного отсчёта. Единицам будет соответствовать число 4095, а нулям — число 0. Пропуская такой сигнал через цифровой фильтр нижних частот, получим цифровой эквивалент исходного аналогового сигнала, точно так же как аналоговый низкочастотный фильтр дал бы нам непрерывный восстановленный сигнал. Следующая далее процедура децимации понижает частоту дискретизации сигнала за счёт отбрасывания большинства отсчётов. В результате получается цифровой сигнал, такой же, как если бы мы непосредственно осуществляли оцифровку входного аналогового сигнала.

Описанный метод применяется во многих промышленно выпускаемых АЦП, ориентированных на оцифровку речевых и звуковых сигналов. Примером может служить микросхема ADC16071 фирмы National Semiconductor. Это 16-битный аналого-цифровой преобразователь с частотой дискретизации до 192 кГц. При частоте дискретизации 100 кГц дельта-модулятор работает на тактовой частоте 6.4 МГц. Цифровой низкочастотный фильтр реализуется в виде фильтра с конечной импульсной характеристикой 246-го порядка. Описание таких фильтров можно найти в Главе 16. Фильтрация удаляет из сигнала все частотные компоненты, лежащие выше 50 кГц, т. е. половины частоты дискретизации. Теоретически

это эквивалентно формированию цифрового сигнала на частоте 6.4 МГц с 16-битным представлением каждого отсчёта и последующим понижением частоты дискретизации с 6.4 МГц до 100 кГц. Понижение частоты дискретизации выполняется путём отбрасывания каждого 63 из 64 последовательно идущих отсчётов. На самом деле внутри устройства происходят гораздо более сложные процессы, чем те, которые обсуждаются в нашем простом примере.

Сигма-дельта конверторы могут применяться также для цифро-аналогового преобразования речевых и звуковых сигналов. При этом цифровой сигнал, хранящийся в памяти, поступает на дельта-модулятор и преобразуется в поток двоичных нулей и единиц. Полученный однобитный сигнал, как было сказано выше, может легко быть преобразован в восстановленный непрерывный сигнал путём пропускания его через аналоговый низкочастотный фильтр. Этот фильтр прост в реализации и обычно представляет собой  $RC$ -цепь. Упрощение структуры аналого-вального фильтра достигается благодаря цифровому фильтру, который берёт на себя основную нагрузку по предварительной обработке сигнала .

Сигма-дельта АЦП имеют некоторые особенности, которые делают их применение ограниченным несколькими специфическими сферами применения. Например, вызывает затруднение мультиплексирование входных сигналов таких АЦП. Дело в том, что при переключении входа схемы с одного сигнала на другой цифровому фильтру, входящему в состав устройства, необходимо время для сброса данных предыдущего сигнала. Другой причиной ограниченности применения сигма-дельта АЦП является невозможность точного определения того, какому моменту времени соответствует тот или иной отсчёт цифрового сигнала. Отсчёты генерируются на основе информации, заложенной в целом фрагменте однобитных данных. Для сигналов с кодированием в частотной области, в частности аудиосигналов, это не будет проблемой, однако для сигналов с кодированием во временной области это оказывается неприемлемо. Часто, чтобы правильно детектировать форму сигнала, точное знание моментов выборки является необходимым. И наконец, ещё одна особенность сигма-дельта АЦП состоит в том, что в большинстве случаев их производство ориентировано на аудиоприложения, что отражается на приводимых производителями характеристиках устройств. Например, указание, что аналого-цифровой преобразователь речевого сигнала является 16-битным, совсем не обязательно означает, что каждый отсчёт цифрового сигнала имеет точность до 16-го бита. Скорее компания-производитель утверждает, что речевые сигналы могут быть преобразованы в цифровой код с 16-битным динамическим диапазоном. Не следует ожидать, что при обработке другого типа сигналов все 16 бит каждого отсчёта будут полностью информативны.

Все приведённые примеры и объяснения являются лишь поверхностным описанием принципов работы АЦП и ЦАП. Следует помнить, что функционирование реальных устройств является более сложным. Все производители имеют свои секреты организации АЦП и ЦАП. Они не хотят делиться этой информацией с конкурентами, поэтому не будут делиться и с вами.

## ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЦОС

Для реализации алгоритмов Цифровой Обработки Сигналов (ЦОС) обычно используются те же языки программирования, что и для решения других научных и инженерных задач: Си, Бейсик, Ассемблер и др. Язык Си, благодаря его мощности и гибкости, предпочитают большинство профессиональных программистов и специалистов по вычислительной технике. Бейсик из-за его простоты с удовольствием используют многие научные работники и инженеры, не являющиеся профессиональными программистами. В действительности же большинство важных для ЦОС особенностей разработки *программного обеспечения* лежит на более низком уровне и не зависит от того, какой язык программирования вы используете. К этим особенностям относятся: способы представления чисел ограниченным числом битов, влияние ошибок округления на результаты арифметических операций, скорость вычислений, характерная для различных типов процессоров, и др. В этой главе будет рассказано о том, что необходимо сделать на высоком уровне программирования, чтобы процессы, происходящие на низком уровне, не разрушили ваш замысел.

### 4.1. Представление чисел в компьютере

Цифровые компьютеры весьма эффективно осуществляют операции записи чисел в память и последующую их загрузку, однако при этом они допускают ошибки. Предположим, вы даёте компьютеру команду сохранить число 1.41421356. Всё, что он сможет сделать, это записать число 1.41421354 – ближайшее по значению к заданному и допустимое в его разрядной сетке. В некоторых случаях возникающая при этом ошибка является несущественной, однако в других случаях она может приводить к катастрофическим результатам. Приведём другой пример. Классическая ошибка в процессе вычислений возникает при сложении двух чисел, сильно отличающихся друг от друга по значению. Например, суммируя числа 1 и 0.00000001, мы ожидаем в качестве результата увидеть 1.00000001, однако компьютер вместо этого даст ответ 1. Понимание того, как процессор хранит данные и манипулирует ими, позволит вам предугадать связанные с представлением чисел ошибки и исправить их раньше, чем ваша программа начнёт выдавать неправильные результаты.

Ошибки представления чисел в компьютере связаны с ограниченностью выделенного для этого числа бит. Обычно это 8, 16, 32 или 64 бита. Рассмотрим ситуацию, когда для представления значений некоторой переменной выделено 8 бит. Такое число битов позволяет записать  $2^8 = 256$  различных двоичных комбинаций, т. е. переменная может принимать только 256 различных значений. Именно здесь и возникает проблема, и мы ничего не можем с этим поделать. Единс-

твенное, на что мы способны повлиять, — это выбор способа представления чисел с помощью заданного числа бит. Например, в простейшем случае с помощью 8 бит мы можем представлять целые числа от 0 до 255, или от 1 до 256, или от -127 до 128. В другом, мало применяемом на практике, случае числа могут представляться в виде экспоненциальной зависимости: 1, 10, 100, ...,  $10^{254}$ ,  $10^{255}$ . Разумеется, оперируя данными, необходимо знать, по какому правилу представляются числа на данной разрядной сетке. Обычно это обеспечивается алгоритмом или формулой преобразования числа из обычного математического представления в слово данных процессора и наоборот.

Несмотря на то что существует много различных схем кодировки чисел в двоичном виде, общее распространение получили два способа: представление чисел в *формате с фиксированной точкой* (целочисленный формат) и представление чисел в *формате с плавающей точкой* (вещественный формат). В программах на языке Бейсик, приводимых в данной книге, для обозначения переменных в формате с фиксированной точкой используется символ «%» в качестве последнего знака в имени переменной. Например: I%, N%, SUM% и т. д. Все остальные переменные — это переменные в формате с плавающей точкой: X, Y, MEAN и др. Ниже следует описание этих двух основных форматов. Знакомясь с ними, пострайтесь оценить их с точки зрения двух основных параметров: *динамического диапазона* (наименьшего и наибольшего чисел, которые могут быть представлены в данном формате) и *точности* (интервала между ближайшими представляемыми числами).

## 4.2. Числа в формате с фиксированной точкой (целочисленный формат)

*Формат с фиксированной точкой* (*fixed-point*) используется для представления целых чисел: ..., -3, -2, -1, 0, 1, 2, 3, ... . В языках программирования высокого уровня, таких как Си и Бейсик, для представления числа в формате с фиксированной точкой обычно отводится 16 бит. В простейшем случае 16 бит позволяют использовать  $2^{16} = 65536$  различных двоичных кодовых комбинаций, что соответствует возможности представления чисел в диапазоне 0...65535. Это *беззнаковый целочисленный формат* (*unsigned integer*). Упрощённый пример записи чисел в этом формате (используются только 4 бита) показан на Рис. 4.1. Переход от обычного представления числа к представлению в беззнаковом целочисленном формате равносителен переходу от десятичного основания к двоичному. Недостатком формата является невозможность представления отрицательных чисел.

*Смещённый двоичный формат* (*offset binary*) аналогичен беззнаковому целочисленному формату, однако десятичные числа в нём «смещаются», оставляя место отрицательным числам. В 4-битном примере, показанном на Рис. 4.1, десятичные числа сдвинуты на 7. В результате диапазон представляемых чисел сужается до 7...8. Аналогично при 16-битном представлении чисел смещение производится на 32767, а диапазон чисел лежит в пределах -32767...32768. Смещённый двоичный формат не является полностью стандартизованным, и вы можете встретить другие величины смещения, например 32768. Наиболее важной областью применения смещённого двоичного формата являются *аналого-цифровое и цифро-аналоговые преобразователи*.

Беззнаковый целочисленный формат		Смешённый двоичный формат		Формат с разделением поля знака и поля значения		Представление в дополнительном коде	
Десятичное число	Двоичный код	Десятичное число	Двоичный код	Десятичное число	Двоичный код	Десятичное число	Двоичный код
15	1111	8	1111	7	0111	7	0111
14	1110	7	1110	6	0110	6	0110
13	1101	6	1101	5	0101	5	0101
12	1100	5	1100	4	0100	4	0100
11	1011	4	1011	3	0011	3	0011
10	1010	3	1010	2	0010	2	0010
9	1001	2	1001	1	0001	1	0001
8	1000	1	1000	0	0000	0	0000
7	0111	0	0111	0	1000	-1	1111
6	0110	-1	0110	-1	1001	-2	1110
5	0101	-2	0101	-2	1010	-3	1101
4	0100	-3	0100	-3	1011	-4	1100
3	0011	-4	0011	-4	1100	-5	1011
2	0010	-5	0010	-5	1101	-6	1010
1	0001	-6	0001	-6	1110	-7	1001
0	0000	-7	0000	-7	1111	-8	1000
Диапазон чисел при 16 битах: 0...65535		Диапазон чисел при 16 битах: -32767...+32768		Диапазон чисел при 16 битах: -32767...+32767		Диапазон чисел при 16 битах: -32768...32767	

**Рис. 4.1.** Общепринятые способы представления данных в формате с фиксированной точкой (целочисленном формате). Беззнаковый целочисленный формат соответствует простому переводу чисел из десятичного представления в двоичное. Этот формат не предназначен для представления отрицательных чисел. Смешённый двоичный код и формат с выделением поля знака позволяют представлять как положительные, так и отрицательные числа. Однако арифметические операции над данными, представленными в этих форматах, трудно реализовать аппаратно. Использование чисел в дополнительном коде очень удобно для аппаратной реализации. Этот формат нашёл широкое распространение в вычислительной технике.

**логовое преобразования.** Например, непрерывный сигнал, колеблющийся в диапазоне  $-5 \text{ В} \dots +5 \text{ В}$ , может быть представлен в виде чисел 0...4095 в случае 12-битного АЦП.

Другим способом представления и положительных, и отрицательных чисел является использование *формата с разделением поля знака и поля значения*. Крайний левый бит отводится под знак и называется *знакомым битом*. Ноль в этом бите означает, что число положительное, единица — что число отрицательное. Другие биты используются так же, как и в предыдущих случаях. Особенностью такого формата является сужение диапазона представляемых чисел на единицу из-за наличия двух комбинаций, соответствующих нулю: 0000 (положительный ноль) и 1000 (отрицательный ноль). При 16-битном формате диапазон представляемых чисел  $-32767 \dots 32767$ .

Первые три рассмотренных формата представления чисел в двоичном коде просты в теории, но трудно реализуемы на практике. Не забывайте, что выражение типа  $A = B + C$  в компьютерной программе фактически означает, что разработчик архитектуры процессора должен придумать, как сложить числа  $B$  и  $C$ , представленные в двоичном коде, чтобы получить число  $A$  в том же формате.

*Дополнительный код* наилучшим образом отвечает требованиям аппаратной реализации. Именно в дополнительном коде обычно представляются целые числа в компьютерах. Чтобы понять, как осуществляется перевод чисел в дополнительный код, выберем за начало отсчёта десятичный ноль, которому в двоичном коде соответствует комбинация битов 0000 (**Рис. 4.1**). Если мы начнём считать в сторону увеличения, то получим простое соответствие десятичных чисел двоичным ( $0 = 0000$ ,  $1 = 0001$ ,  $2 = 0010$ ,  $3 = 0011$  и т. д.). Теперь вспомним, что наши двоичные числа хранятся в 4-битных регистрах, состоящих из 4 бистабильных ячеек (ячеек с двумя возможными состояниями). Если мы начнём считать от 0000 в сторону уменьшения, то арифметическое устройство, выполняющее вычитание, автоматически начнёт использовать дополнительный код:  $0 = 0000$ ;  $-1 = 1111$ ;  $-2 = 1110$ ;  $-3 = 1101$  и т. д. Ситуация аналогична автомобильному одометру, используемому в современных машинах. При движении вперёд его показания меняются в порядке: 00000, 00001, 00002, 00003 и далее. А при движении назад: 00000, 99999, 99998, 99997 и далее.

При наличии 16 бит для представления чисел дополнительный код даёт возможность использовать диапазон чисел  $-32768\dots32767$ . Крайний левый бит при этом принимает значение 0 для положительных чисел и нуля и 1 — для отрицательных. Поэтому этот бит также называется знаковым, как и в формате с разделением поля знака и поля значения. Преобразование чисел из десятичного формата в дополнительный код является очевидным для положительных чисел: простой переход от десятичной системы счисления к двоичной. Для отрицательных чисел обычно используется следующий алгоритм: 1) взять модуль десятичного числа; 2) преобразовать его в двоичное число; 3) проинвертировать все биты (единицы заменить нулями, а нули — единицами); 4) добавить 1 к полученному двоичному числу. Например:  $-5 \rightarrow 5 \rightarrow 0101 \rightarrow 1010 \rightarrow 1011$ . Дополнительный код менее понятен для человека, но зато гораздо более удобен для вычислительной техники.

## 4.3. Числа в формате с плавающей точкой (вещественный формат)

Представление чисел в *формате с плавающей точкой* сложнее, чем в формате с фиксированной точкой. Основная идея аналогична той, которая используется при экспоненциальной записи чисел в виде совокупности *мантицы* и *порядка*. Например, в записи  $5.4321 \times 10^6$  мантиссой выступает число 5.4321, а порядком — число 6. Экспоненциальная запись великолепно подходит для одновременного представления и очень больших, и очень маленьких чисел. Например,  $1.2 \times 10^{50}$  — это количество атомов, из которых состоит Земля, а  $2.6 \times 10^{-23}$  — это расстояние, которое проползает черепаха за одну секунду, отнесённое к диаметру нашей Галактики. Заметим, что числа в экспоненциальной форме являются нормализованными: слева от десятичной точки всегда остаётся только одна отличная от нуля цифра. Это достигается соответствующим выбором порядка.

Представление чисел в формате с плавающей точкой основывается на том же принципе, что и представление в экспоненциальной форме, с той лишь разницей, что вместо десятичной записи и основания 10 используется двоичная запись

и основание 2. Существует несколько похожих, но всё же отличающихся друг от друга форматов с плавающей точкой. Наиболее распространённым среди них является стандарт ANSI/IEEE Std. 754-1985. Этот стандарт определяет формат записи чисел с одинарной (*single precision*) и с двойной точностью (*double precision*). Числа с одинарной точностью являются 32-битными. Под числа с двойной точностью отводится 64 бита. Запись чисел в формате с плавающей точкой с одинарной точностью иллюстрирует Рис. 4.2. Предназначенные для этого формата 32 бита делятся на 3 части: 0...22 отводятся под мантиссу; 23...30 хранят порядок числа; а 31 бит является знаковым. Формирование числа на основе такой комбинации битов осуществляется в соответствии с выражением

$$v = (-1)^S \times M \times 2^{E-127}. \quad (4.1)$$

Преобразование двоичной комбинации в вещественное число  $v$ .  
 $S$  — это значение знакового бита,  $M$  — значение мантиссы,  $E$  — порядок.

Множитель  $(-1)^S$  в выражении (4.1) означает, что знаковый бит  $S$  принимает значение 0 для положительных чисел и 1 — для отрицательных. Под порядок числа  $E$  отводится 8 битов, позволяя ему принимать значения 0...255. Вычитание 127 из числа  $E$  устанавливает диапазон значений степени  $2^{-127} \dots 2^{128}$ . То есть для представления порядка используется смещённый двоичный формат с величиной смещения, равной 127.

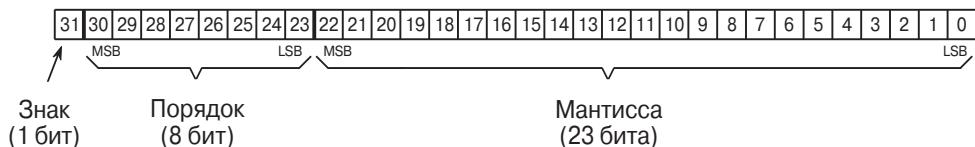
Мантисса числа  $M$  формируется из выделенных под неё 23 битов как двоичная дробь. Например, десятичная дробь 2.783 может быть разложена в ряд:  $2 + 7/10 + 8/100 + 3/1000$ . Аналогично двоичная дробь 1.0101 представляется в виде:  $1 + 0/2 + 1/4 + 0/8 + 1/16$ . Для формата с плавающей точкой характерна та же нормализация, что и для чисел в экспоненциальной форме: только один ненулевой бит остаётся слева от двоичной точки (двоичная точка в двоичной системе счисления — это эквивалент десятичной точки, используемой в десятичной системе счисления). Однако в двоичной системе счисления, в отличие от десятичной, этот ненулевой бит может принимать единственное значение 1. То есть первый бит мантиссы всегда равен 1, а следовательно, хранить его не требуется. Благодаря этому удается использовать ещё один дополнительный бит для младших битов мантиссы, увеличивая точность представления чисел. Таким образом, выделенные под мантиссу 23 бита  $m_{22}, m_{21}, m_{20}, \dots, m_0$  преобразуются в её значение по следующей формуле:

$$M = 1. m_{22}m_{21}m_{20}m_{19} \dots m_2m_1m_0. \quad (4.2)$$

Формирование мантиссы  $M$ , используемой в выражении (4.1),  
из последовательности выделенных под неё бит.

Другой формой записи выражения (4.2) является формула:  $M = 1 + m_{22}2^{-1} + m_{21}2^{-2} + m_{20}2^{-3} \dots$ . Если значения всех битов, выделенных под мантиссу, равны 0, то  $M$  принимает значение 1. Если значения всех битов равны 1, то мантисса будет равна примерно двум, а именно:  $2 - 2^{-23}$ .

Самые большие по модулю числа, представляемые в двоичном коде по приведённой схеме, — это  $\pm(2 - 2^{-23}) \times 2^{128} = \pm6.8 \times 10^{38}$ . Самыми маленькими по модулю числами являются  $\pm1.0 \times 2^{-127} = \pm5.9 \times 10^{-39}$ . Стандарт ANSI/IEEE Std. 754-1985 немного сужает этот диапазон, освобождая некоторые кодовые комбинации для специальных значений. Самыми большими и самыми маленькими



## Пример 1

$$0\ 00000111\ 110000000000000000000000000000 + 1.75 \times 2^{(7 - 127)} = +1.316554 \times 10^{-36}$$

↓      ↓      ↓  
 +      7      0.75

## Пример 2

$$1\ 10000001\ 011000000000000000000000000000 - 1.375 \times 2^{(129 - 127)} = -5.500000$$

↓      ↓      ↓  
 -      129      0.375

**Рис. 4.2.** Представление чисел в формате с плавающей точкой с одинарной точностью. 32 бита разбиты на три части: знак, порядок и мантисса. Выражения (4.1) и (4.2) описывают, каким образом эти части формируют вещественное число.

по модулю значениями становятся  $\pm 3.4 \times 10^{38}$  и  $\pm 1.2 \times 10^{-38}$  соответственно. Освободившиеся комбинации битов используются для представления специальных значений трёх типов: 1) значению  $\pm 0$  соответствует комбинация, когда и мантисса, и порядок состоят из всех нулей; 2) значению  $\pm \infty$  соответствует мантисса, состоящая из всех нулей, и порядок, состоящий из всех единиц; 3) очень маленькие по величине ненормализованные числа в диапазоне  $\pm 1.2 \times 10^{-38}$  и  $\pm 1.4 \times 10^{-45}$ . Эти числа имеют меньшую точность и получаются за счёт снятия требования равенства единице крайней левой цифры мантиссы. Кроме этих трёх типов специальных значений, употребляются комбинации, обозначающие нечисловые значения — NaN (не число).

Представление чисел в формате с плавающей точкой с двойной точностью отличается выделением большего числа битов для мантиссы и порядка. Из 64 битов, предусмотренных для чисел с двойной точностью, биты с 0-го по 51-й используются под мантиссу, биты с 52-го по 62-й — под порядок, 63-й бит является битом знака. Как и прежде, мантисса может принимать значения от 1 и почти что до 2:  $M = 1 + m_{51}2^{-1} + m_{50}2^{-2} + m_{49}2^{-3} \dots$ . Одиннадцать битов, выделенных под порядок, позволяют ему принимать значения 0...2047 и, с учётом сдвига на 1023, использовать умножение на  $2^{-1023} \dots 2^{1024}$ . Минимальное и максимальное представляемые значения составляют соответственно  $\pm 1.8 \times 10^{308}$  и  $\pm 2.2 \times 10^{-308}$ . Это действительно чрезвычайно большое и чрезвычайно малое значения. Приложений, в которых одинарной точности чисел оказывается недостаточно, крайне мало, а приложения, в которых и двойная точность окажется недостаточной, вам вообще едва ли удастся найти.

## 4.4. Точность представления чисел

Ошибки представления чисел в компьютере очень похожи на ошибки квантования, возникающие в процессе аналого-цифрового преобразования. Вам бы хотелось оперировать с непрерывным рядом значений переменной, однако в компьютере может быть представлено лишь ограниченное количество уровней квантования. Каждое новое число, появляющееся, например, в результате арифметических действий, должно быть округлено до ближайшего значения, которое может быть представлено в используемом числовом формате.

Рассмотрим пример. Допустим, для представления чисел вы используете 32 бита. Это позволяет записать  $2^{32} = 4294967296$  различных кодовых комбинаций, т. е. 4294967296 различных чисел. В некоторых языках программирования существует тип *длинное целое* (*long integer*), который хранит 32-битные данные в *формате с фиксированной точкой*, представляемые в *дополнительном коде*. Диапазон значений переменных этого типа лежит от  $-2147483648$  до  $2147483647$ . В то же время при использовании *формата с плавающей точкой* одинарной точности те же 4294967296 кодовые комбинации позволяют работать в гораздо более широком диапазоне: от  $-3.4 \times 10^{38}$  до  $3.4 \times 10^{38}$ .

При использовании формата с фиксированной точкой шаг между соседними представляемыми числами всегда одинаков и равен 1. В формате с плавающей точкой этот шаг существенно варьируется в зависимости от порядка чисел. Если выбрать случайным образом некоторое число в формате с плавающей точкой, то шаг между ним и соседним с ним числом окажется приблизительно в 10 миллионов раз ( $2^{-24} \dots 2^{-23}$ ) меньше значения самого числа. Здесь проявляется одна из ключевых особенностей формата с плавающей точкой: большие числа имеют меньшую точность, а меньшие числа — большую точность. Эта особенность проиллюстрирована на **Рис. 4.3**, на котором показаны последовательно идущие числа в различных диапазонах значений и приведены соответствующие шаги между соседними числами.

**Программа 4.1** иллюстрирует, как *ошибки округления* (аналог ошибок квантования в математических операциях) могут явиться причиной неправильной работы вычислительного устройства. В этой программе переменная *X* инициализируется значением 1.000000, а затем участвует в циклически повторяющихся арифметических действиях, результат которых не должен менять значений переменной *X*. Программа включает цикл, в теле которого два случайным образом сгенерированных числа — *A* и *B* добавляются к переменной *X*, представленной в формате с плавающей точкой, а затем вычтываются из полученного результата. Таким образом, выполнение цикла не должно менять величину *X*. В действительности, однако, вычисления будут сопровождаться ошибками округления, и значение *X* будет по-немногу уходить от своего первоначального значения с каждой итерацией цикла. Этот уход может носить различный характер в зависимости от того, как ошибки округления будут компенсировать друг друга. Если случайные их значения появляются как с положительным, так и с отрицательным знаком, то уход величины *X* от правильного значения будет то нарастать, то уменьшаться. Если же ошибки округления постоянно или преимущественно оказываются с одним и тем же знаком, то общая ошибка результата будет быстро и неуклонно расти.

0.00001233862713	Шаг: 0.0000000000091
0.00001233862804	(1/13-миллионная часть)
0.00001233862895	
0.00001233862986	
⋮	
1.0000000000	Шаг: 0.000000119
1.000000119	(1/8-миллионная часть)
1.000000238	
1.000000358	
⋮	
1.996093750	Шаг: 0.000000119
1.996093869	(1/17-миллионная часть)
1.996093988	
1.996094108	
⋮	
636.0312500	Шаг: 0.0000610
636.0313110	(1/10-миллионная часть)
636.0313720	
636.0314331	
⋮	
217063424.0	Шаг: 16.0
217063440.0	(1/14-миллионная часть)
217063456.0	
217063472.0	

**Рис. 4.3.** Примеры интервалов между соседними числами в формате с плавающей точкой с одинарной точностью для различных диапазонов значений. Шаг между числами лежит в пределах от 1/8-миллионной до 1/17-миллионной части этих чисел.

#### **Программа 4.1. Программа, иллюстрирующая накопление ошибки округления в процессе вычислений с плавающей точкой**

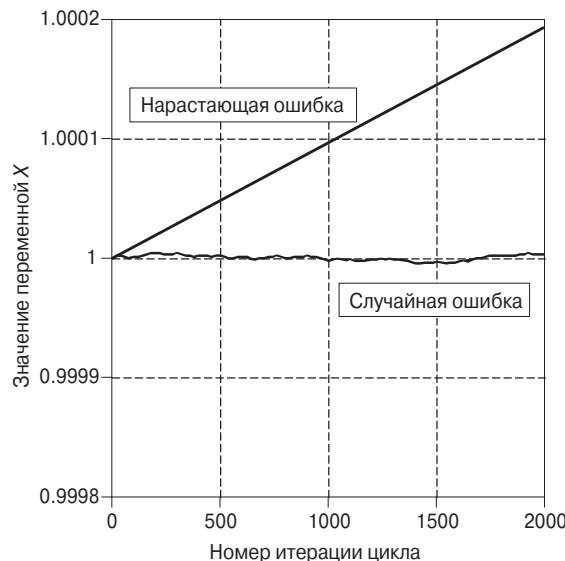
```

100 X = 1           'инициализация X
110
120 FOR I% = 0 TO 2000
130 A = RND          'запись в А и В
140 B = RND          'случайных значений
150
160 X = X + A      'добавление к X значений А и В
170 X = X + B      'обратное вычитание А и В - отмена
180 X = X - A      'предыдущих действий
190 X = X - B
200
210 PRINT X         'в идеале X должен быть равен 1
220 NEXT I%
230 END

```

На **Рис. 4.4** проиллюстрирован процесс ухода значения переменной  $X$  от начального значения. Очевидным выводом является то, что непрерывно нарастающая ошибка проявляется гораздо сильнее, чем ошибка случайная. Случайные ошибки частично компенсируют друг друга. Нарастающая ошибка просто накапливает все ошибки округления, возникающие при выполнении каждой арифме-

тической операции. Эту суммарную ошибку можно грубо оценить величиной, равной ошибке округления при выполнении одной операции, умноженной на число таких операций. В отличие от нарастающей ошибки, случайная ошибка увеличивается пропорционально квадратному корню из числа арифметических операций, т. е. гораздо медленнее. Как показывает приведённый пример, в алгоритмах цифровой обработки сигналов непрерывно нарастающая ошибка может оказываться в сотни раз больше случайной.



**Рис. 4.4.** Накопление ошибки округления в процессе вычислений в формате с плавающей точкой. Представленные кривые генерированы **Программой 4.1**. Если переменная в формате с плавающей точкой участвует в повторяющихся вычислительных операциях, то накопление ошибок округления приводит к отклонению её значения от истинного. Когда ошибка округления в разных итерациях появляется то с положительным, то с отрицательным знаком, уход значения переменной от истины будет то увеличиваться, то уменьшаться случайным образом. Если же ошибки округления в подавляющем большинстве итераций оказываются с одним и тем же знаком, то суммарная ошибка будет нарастать гораздо быстрее.

К сожалению, проследить или предсказать, как поведёт себя ошибка округления в том или ином конкретном алгоритме обработки сигнала, практически невозможно. **Программа 4.1**, например, генерирует нарастающую ошибку. Однако небольшое изменение случайных чисел  $A$  и  $B$  приведёт к тому, что ошибка станет носить случайный характер. В частности, это может быть сделано заменой строк  $A = \text{RND}$  и  $B = \text{RND}$  на  $A = \text{EXP}(\text{RND})$  и  $B = \text{EXP}(\text{RND})$ . Именно эта замена была использована для получения кривой случайной ошибки, приведённой на **Рис. 4.4**. Равномерное распределение значений чисел  $A$  и  $B$  на интервале  $0\dots 1$  заменяется экспоненциальным распределением на интервале  $1\dots 2.718$ . Даже такого незначительного изменения в программе достаточно, чтобы кардинально поменять характер поведения суммарной ошибки округления.

Так как мы не можем предугадать характер поведения ошибки округления, то приходится ориентироваться на худший случай. Будем ожидать, что каждое число в формате с плавающей точкой с одинарной точностью содержит ошибку, равную

одной 40-миллионной части числа, умноженной на количество арифметических действий, в которых это число участвовало. Такая оценка суммарной ошибки округления сделана в предположении, что она является постоянно нарастающей, и её среднее значение для одной операции составляет одну четвёртую часть шага квантования. Рассуждая аналогичным образом, получим, что для чисел с удвоенной точностью ошибка представления числа с плавающей точкой составляет  $1/40$ -квадрилионную часть числа, умноженную на количество арифметических операций.

**Таблица 4.1. Сравнение форматов с плавающей и с фиксированной точкой для управления циклом**

Программа А	Программа Б
100 'Счётчик цикла с плавающей точкой	100 'Счётчик цикла с фиксированной точкой
110 FOR X = 0 TO 10 STEP 0.01	110 FOR I% = 0 TO 1000
120 PRINT X	120 X = I%/100
130 NEXT X	130 PRINT X
	140 NEXT I%
Результат выполнения программы А:	Результат выполнения программы Б:
0.00	0.00
0.0	0.0
1	1
0.0	0.0
2	2
0.0	0.03
3	.
.	.
.	.
.	.
9.960132	9.96
9.9	9.9
70133	7
9.9	9.9
80133	8
9.9	9.9
90133	9
	10.00

**Табл. 4.1** иллюстрирует особенно неприятное следствие ошибок округления. Обе приведённые программы выполняют одну и ту же задачу — выводят на печать 1001 число с одинаковым шагом в диапазоне 0...10. Программа в левой колонке использует в качестве счётчика цикла переменную  $X$  в формате с плавающей точкой. Перед выполнением цикла счётчику присваивается начальное значение (в данном примере — ноль). В конце каждой итерации цикла счётчик увеличивается на величину шага (в рассматриваемой программе — 0.01). После этого принимается решение, следует ли повторять итерации цикла, или его выполнение завершено. Решение принимается путём сравнения содержимого счётчика цикла с его конечным значением  $X = 10$ . Приведённый в таблице результат выполнения программы показывает, что за время выполнения цикла из-за нарастающей суммарной ошибки округления переменная  $X$  существенно отклоняется

от правильного значения. Накопленная ошибка приводит даже к тому, что последняя итерация цикла не выполняется, поскольку значение переменной  $X$  вместе со десяти оказывается равным 10.000133. А раз счётчик цикла принял значение, большее требуемого конечного числа, то цикл считается выполненным. Такая потеря крайнего значения является достаточно распространённой ошибкой, присущей в компьютерных программах.

Программа в правой колонке использует в качестве счётчика цикла переменную в формате с фиксированной точкой 1%. Сложение, вычитание или умножение двух целых чисел всегда дают целое число. Поэтому для формата чисел с фиксированной точкой абсолютно не характерны ошибки округления (при достаточной длине разрядной сетки). Целочисленный формат великолепно подходит для представления переменных, используемых в качестве счётчика цикла, и других переменных, подвергающихся многочисленным арифметическим преобразованиям. В нашем примере использование формата данных с фиксированной точкой для управления циклом гарантирует, что последняя итерация цикла не пропадёт! Страйтесь всегда, если есть возможность, применять этот формат для переменных, используемых в качестве индексов или счётчиков.

В случае, если для счётчика цикла вам всё же необходимо представление в формате с плавающей точкой, попытайтесь вместо обычных дробей, таких, как 0.1, 0.6, 1.4, 2.3, использовать дроби, у которых в знаменателе стоит степень двух, например,  $1/2$ ,  $1/4$ ,  $3/8$ ,  $27/16$ . То есть от цикла типа FOR  $X = 1$  TO 10 STEP 0.1 следует переходить к циклу вида FOR  $X = 1$  TO 10 STEP 0.125. Это позволяет уменьшить ошибку округления за счёт того, что счётчик цикла всегда оказывается представлен точно двоичным кодом. Например, десятичное число 1.125 имеет в двоичной форме точное представление:  $1.00100000000000000000000000000000 \times 2^0$ . В то же время число 1.1 в формате с плавающей точкой оказывается между двумя значениями: 1.0999999046 и 1.1000000238 (в двоичной форме это числа  $1.0001100110011001100 \times 2^0$  и  $1.0001100110011001101 \times 2^0$ ). Это приводит к неизбежной погрешности в вычислениях каждый раз, когда число 1.1 встречается в программе.

Полезно будет запомнить, что для формата с плавающей точкой с одинарной точностью характерно точное двоичное представление всех целых чисел в диапазоне  $\pm 16.8$  миллионов (или  $\pm 2^{24}$ ). Вне этого диапазона шаг между соседними числами становится больше единицы, что приводит к невозможности точного представления некоторых целых чисел. Целые числа в формате с плавающей точкой из указанного диапазона значений участвуют в операциях сложения, вычитания и умножения без потери точности.

## 4.5. Скорость вычислений: влияние языка программирования

В программировании систем цифровой обработки сигналов условно можно выделить три уровня, отличающихся по своей сложности: уровень Ассемблера, уровень компилируемого языка и уровень приложений. Чтобы понять различие между ними, нам необходимо вспомнить основы цифровой электроники. Вся микропроцессорная техника построена на использовании набора внутренних

двоичных регистров, т.е. наборов триггеров, способных хранить состояния логических нулей и единиц. К примеру, микропроцессор серии 8088, на котором были построены компьютеры IBM PC, включает четыре 16-битных регистра общего назначения. Им присвоены имена AX, BX, CX и DX. Кроме них, предусмотрено девять регистров специального назначения: SI, DI, SP, BP, CS, DS, SS, ES и IP. В частности, регистр IP — указатель команды (Instruction Pointer) — «отслеживает», где в памяти процессора располагается следующая выполняемая команда.

Предположим, вы пишите программу сложения двух чисел: 1234 и 4321. В начале выполнения программы регистр IP содержит адрес сегмента памяти, в котором записаны коды, приведённые в **Табл. 4.2**. Эти коды для обычного человека выглядят совершенно бессмысленно. Однако в них содержится вся информация, необходимая компьютеру для выполнения поставленной задачи, включая команды и данные. Встречая двоичную комбинацию 00000011 11000011, процессор понимает её как команду выборки 16-битного операнда из регистра BX, сложения его с другим 16-битным операндом, хранящимся в регистре AX, и записи результата в регистр AX. Эти комбинации нулей и единиц называются *машинным кодом* и являются самым низким уровнем программирования.

**Таблица 4.2. Машинный код программы сложения чисел 1234 и 4321**

10111001	00000000
11010010	10100001
00000100	00000000
10001001	00000000
00001110	10001011
00000000	00011110
00000000	00000010
10111001	00000000
11100001	00000011
00010000	11000011
10001001	10100011
00001110	00000100
00000000	00000001

**Примечание.** Правая колонка в таблице является продолжением левой.

Работа с двоичным кодом способна свести с ума даже самого терпеливого программиста. Поэтому различным комбинациям двоичных кодов присвоили имена в соответствии с теми функциями, которые они выполняют. Так появился язык *Ассемблер*, который был использован при написании **Программы 4.2**. Ассемблерный код гораздо проще для понимания, чем машинный код, но по своей сути эти два уровня программирования являются идентичными, так как в том и другом случае соответствие команд и выполняемых микропроцессором действий лежит на самом низком уровне: каждая команда напрямую управляет определёнными аппаратными узлами микропроцессора. Команда Ассемблера ADD AX, BX транслируется в машинный код 00000011 11000011. Перевод ассемблерного кода, называемого *исходным кодом* (*source code*), в машинный код, называемый *объектным* (*object code*) или *исполняемым кодом* (*executable code*), осуществляется программа, также называемая Ассемблером. Исполняемый код может быть непосредственно выполнен микропроцессором. Очевидно, что для программирования на

языке Ассемблера необходимо хорошее знание внутренней архитектуры конкретного микропроцессора, который вы собираетесь использовать.

### Программа 4.2. Ассемблерный код программы сложения чисел 1234 и 4321

```
MOV CX,1234          ;загрузка числа 1234 в регистр CX  
MOV DS:[0],CX        ;запись этого числа в память по адресу DS:[0]  
MOV CX,4321          ;загрузка числа 4321 в регистр CX  
MOV DS:[2],CX        ;запись этого числа в память по адресу DS:[2]  
MOV AX,DS:[0]          ;загрузка данных, хранящихся в памяти по адресам  
MOV BX,DS:[2]          ;DS:[0] и DS:[2] в пару регистров AX, BX  
ADD AX,BX            ;сложение AX и BX с загрузкой результата в AX  
MOV DS:[4],AX          ;запись суммы в память по адресу DS:[4]
```

Итак, команды Ассемблера позволяют напрямую манипулировать аппаратными ресурсами микропроцессора: регистрами, ячейками памяти, битами состояния и т. д. Следующий более сложный уровень программирования оперирует абстрактными переменными, не привязанными к конкретным аппаратным узлам. Используемые на этом уровне языки программирования называются *компилируемыми языками* или *языками высокого уровня*. Более десятка различных языков высокого уровня получили общее распространение. Вот некоторые из них: Си, Бейсик, Фортран, Паскаль, АПЛ, КОБОЛ, Лисп и другие. **Программа 4.3** для сложения чисел 1234 и 4321 написана на языке Бейсик. Программист имеет дело только с переменными A, B и C и может ничего не знать об аппаратных средствах процессора, выполняющего эту программу.

### Программа 4.3. Программа сложения чисел 1234 и 4321 на языке Бейсик

```
100 A = 1234  
110 B = 4321  
120 C = A+B  
130 END
```

Для перевода исходного кода, написанного на языке высокого уровня, в машинный код служит программа, называемая *компилятором*. Компилятор должен выполнять привязку абстрактных переменных, используемых в программе, к конкретным ячейкам памяти используемой аппаратуры. В рассматриваемой программе, там где переменная A встречается первый раз (строка с номером 100), компилятор определяет её как переменную в формате с плавающей точкой с одинарной точностью и в соответствии с типом переменной выделяет под неё в памяти процессора четыре байта. Эта область памяти будет использоваться при выполнении программы исключительно для хранения значения переменной A. Каждый последующий раз, когда переменная A будет встречаться в программе, содержимое этих четырёх байтов будет изменяться соответствующим образом. Другой задачей компилятора является замена сложных математических выражений, таких, например, как  $Y = \text{LOG}(\text{X}_{\text{COS}(Z)})$ , рядом базовых арифметических действий. Ведь микропроцессоры умеют только складывать, вычитать, умножать и делить. Любые более сложные вычисления должны быть представлены комбинацией этих элементарных действий.

Языки высокого уровня избавляют программиста от необходимости работать непосредственно с аппаратным обеспечением. Процесс разработки программы

существенно упрощается, а написанные исходные коды оказываются легко переносимыми с одних аппаратных платформ на другие. Самым же важным достоинством использования языков высокого уровня является то, что программист может не знать внутреннюю архитектуру микропроцессора, для которого он пишет программу. Задачу досконального изучения микропроцессора и интерпретации высокоуровневых программ в машинные коды берёт на себя другой программист — разработчик компилятора.

Большинство компиляторов осуществляют трансляцию в машинные коды всей программы перед тем, как её выполнять. Однако есть и ряд компиляторов, называемых *интерпретаторами*, которые осуществляют трансляцию построчно: преобразуют в машинный код одну строку исходной программы, организуют выполнение этого кода и переходят к трансляции следующей строки. Наиболее типичным примером интерпретатора может служить Бейсик. Интерпретатор предоставляет разработчику очень удобную интерактивную среду программирования, однако скорость вычислений в этом режиме компиляции существенно ниже (скажем, в 100 раз).

Наиболее высоким уровнем программирования систем ЦОС является уровень приложений. Предположим, вы покупаете некоторый современный процессор для встраивания в разрабатываемую вами систему цифровой обработки сигналов. Такие специализированные микропроцессоры обычно содержат набор встроенных аппаратных средств поддержки систем ЦОС: например, аналоговые входы, аналоговые выходы, цифровой ввод/вывод, антиэлайзинговый и восстанавливающий фильтры и др. Перед вами стоит задача разработки программы для этого устройства. В худшем случае производитель снабдит вас только Ассемблером процессора, и вам придётся разбираться с внутренней архитектурой устройства. В более благоприятном и типовом случае вы получите также компилятор языка Си, позволяющий вести разработку программ на языке высокого уровня, не вдаваясь в подробности работы микропроцессора.

В самом же лучшем случае производитель обеспечит вас сложными программными пакетами, оказывающими существенную поддержку при разработке программного обеспечения. Это могут быть библиотеки функций обработки сигналов, программные пакеты управления устройствами ввода/вывода, инструментальные средства отладки программ и т. д. Всё, что от вас требуется, — это построить в графической среде нужную вам систему, просто соединив требуемым образом выбранный вами набор функциональных блоков. Вы манипулируете такими объектами, как каналы обработки сигнала, алгоритмы обработки, параметры ввода/вывода цифровых и аналоговых сигналов и т. д. Сконструировав систему должным образом, вы даёте команду построения проекта, и структура вашей системы автоматически переводится из графической формы в машинный код цифрового процессора. Могут быть и другие инструментальные программные средства, ориентированные непосредственно на приложения: например, обработку изображений, спектральный анализ, измерительные и управляющие системы, проектирование цифровых фильтров и т. д. Все эти средства постоянно разрабатываются и совершенствуются.

Границы между тремя описанными уровнями разработки программного обеспечения весьма расплывчаты. Например, в большинстве компилируемых языков предусмотрена возможность прямого управления аппаратными ресурсами про-

цессора. Одновременно с этим компиляторы языков высокого уровня часто сопровождаются библиотеками типовых функций обработки, что позволяет говорить о программировании на уровне приложения. Поэтому основу разделения уровней программирования следует искать в том, какими объектами вы оперируете: аппаратными ресурсами, абстрактными данными или целыми процессами.

Существует также другой аспект выделения различных уровней программирования. При использовании языков высокого уровня в задачах оптимального управления внутренними ресурсами микропроцессора вы полагаетесь на программиста, разработавшего компилятор. Аналогично, работая на уровне приложений, вы считаете, что требуемые вам функции, например сигнальной обработки, были написаны программистом, правильно и оптимально разработавшим используемый вами инструментальный пакет. Однако эти не известные вам программисты решали поставленные задачи с точки зрения универсальности и ничего не знали о вашей конкретной системе. Поэтому обычно они не способны обеспечить вас действительно оптимальным решением. Чем более высокий уровень программирования вы используете для разработки, тем менее эффективным окажется конечный машинный код с точки зрения затрат памяти, скорости выполнения и точности вычислений.

Какой же язык программирования выбрать? Ответ на этот вопрос зависит от того, кем вы являетесь и какую задачу решаете. Большинство профессиональных программистов и специалистов по вычислительной технике используют язык Си (или его усовершенствованную версию Си++). Язык Си отличают мощность, гибкость, модульная организация. Язык Си настолько популярен, что невольно возникает вопрос: «А зачем вообще использовать для программирования ЦОС-систем какой-либо другой язык, кроме Си?» Можно предложить три ответа на этот вопрос. Во-первых, ЦОС-технологии пришли в нашу жизнь столь стремительно, что многие фирмы и отдельные разработчики просто не успели перестроиться — «зависли» в режиме использования таких языков, как Фортран или Паскаль. Особенно это касается военных и правительственные организаций с их консерватизмом. Во-вторых, некоторые приложения требуют крайне высокой вычислительной эффективности, достижимой только с помощью Ассемблера. Это такие приложения, в которых оправдано значительное усложнение работы программиста ради небольшого выигрыша в скорости вычислений. И в-третьих, язык Си не так уж прост в освоении и едва ли подходит для тех, кто занимается программированием лишь от случая к случаю. А инженеров и научных работников, нуждающихся в использовании цифровой обработки сигналов именно время от времени, существует достаточно много. Эти специалисты вынуждены обращаться, например, к Бейсику благодаря его простоте.

Почему язык Бейсик взят за основу в этой книге? Дело в том, что она рассказывает об алгоритмах, а не о языках программирования. Читателю следует сконцентрировать своё внимание на методах ЦОС и не отвлекаться на понимание конструкций того или иного языка программирования. Например, все строки программ в этой книге пронумерованы. Это очень удобно для их описания: «строка с номером 100 делает то-то, строка 110 работает так-то». Вы конечно же едва ли будете использовать нумерацию строк при разработке своих собственных программ. Но изучение цифровой обработки сигналов требует от оформления программных кодов одного, а использование полученных знаний — другого. Су-

ществует большое число книг по данной тематике, где можно найти готовые примеры программных кодов для алгоритмов ЦОС. Если вы хотите просто найти уже кем-то написанную программу и вставить её в свой проект, вы идёте неправильным путём.

Сравнение эффективности языков программирования низкого и высокого уровней — это весьма неблагодарная задача. Вне зависимости от результата таких «соревнований» проигравший всегда сможет пожаловаться на несправедливое судейство. Программисты, предпочитающие языки высокого уровня (обычно специалисты по вычислительной технике широкого профиля) утверждают, что ассемблерный код только на 50% быстрее кода компилятора, но в 5 раз сложнее для разработки. Те же, кто предпочитает Ассемблер (ЦОС-программисты и специалисты по аппаратному обеспечению), наоборот, говорят, что ассемблерный код в 5 раз быстрее и лишь на 50% сложнее в использовании. И конечно же, как и бывает в спорах, обе стороны могут подтвердить свою точку зрения рядом своих примеров.

Как показывает практика, скорость выполнения программы, написанной на Ассемблере, в 1.5...3 раза выше скорости выполнения той же программы, написанной на языке высокого уровня. Однако эта величина весьма условна. Единственный способ оценить выигрыш в скорости для конкретного случая — это написать соответствующие программные коды и протестировать результаты. Если учитывать, что скорости персональных компьютеров возрастают каждый год примерно на 40%, то получается, что переход на язык низкого уровня эквивалентен двухгодичному скачку в развитии аппаратных технологий.

Большинство профессиональных программистов скорее всего будут чувствовать себя оскорбленными, если им предложат программировать на Ассемблере, и просто осмеянными, если им предложат Бейсик. Дело в том, что языки Ассемблер и Бейсик «заставляют» писать программы без соблюдения хорошего стиля программирования. Хороший стиль программирования подразумевает портируемость (возможность переноса с одной аппаратной платформы на другую), модульный принцип построения (организация программы в виде совокупности самостоятельных модулей) и удобство восприятия (сопровождение комментариями и использование имён переменных, говорящих об их назначении). Обеспечить эти характеристики программного обеспечения при использовании Бейсика или Ассемблера очень непросто. Кроме того, Ассемблер и Бейсик чаще всего используют люди, не имеющие достаточных знаний о правилах разработки программного обеспечения и документации к нему.

Любители Ассемблера отвечают на подобные атаки собственными выпадами. Представьте, что вы разработали некоторую программу на языке Си, а ваш конкурент такую же программу на Ассемблере. Конечный пользователь скажет о вашем программном продукте, что это ненужный хлам, так как он работает вдвое медленнее. Кстати, никто не заставляет вас писать всю программу на Ассемблере — только те фрагменты кода, в которых скорость вычислений особенно важна. К примеру, многие функции из программных библиотек цифровой обработки сигналов написаны на Ассемблере и адаптированы к вызову из Си-проектов, в состав которых они включаются. Даже самый ярый борец за соблюдение хорошего стиля программирования будет использовать ассемблерные коды, в случае если ему не нужно разрабатывать их самому.

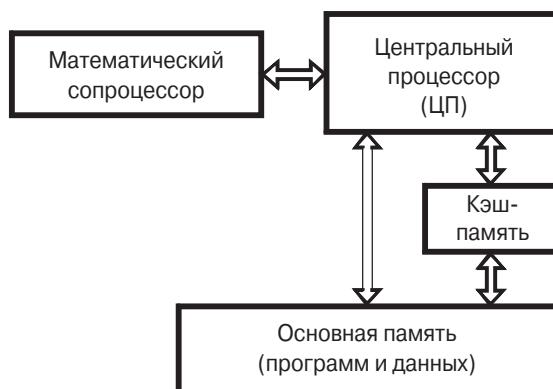
## 4.6. Скорость вычислений: влияние аппаратной платформы

Вычислительная мощность современных компьютеров возрастает так стремительно, что посвящённые им книги устаревают, ещё не успев выйти в свет. Данная книга, к сожалению, не исключение. Для автора — это просто кошмар! Первый персональный компьютер IBM PC появился в 1981 году. В его основе лежал микропроцессор 8088 с тактовой частотой 4.77 МГц и 8-битной шиной данных. За ним последовал ряд новых семейств ПК, которые появлялись каждые 3...4 года: 8088 → 80286 → 80386 → 80486 → 80586 (Pentium). Каждый новый процессор обладал производительностью, в среднем в 5 раз большей по сравнению с его предшественником. К 1996 г. тактовые частоты достигли 200 МГц, а ширина шины данных — 32 бита. Наряду с другими усовершенствованиями это привело к общему увеличению вычислительной производительности процессоров в 1000 раз за 15 лет! Можно ожидать дальнейшего возрастания вычислительной мощности компьютеров в последующие годы.

Единственный способ получить самую последнюю информацию в этой области — это обратиться непосредственно к производителю и посмотреть рекламные материалы, спецификации, прайс-листы и т. д. Искать данные о производительности процессоров в книгах не имеет смысла. Ищите их в журналах и ежедневных информационных сообщениях. Можно ожидать примерно двухкратного увеличения скорости вычислений каждые два года. Однако просто интересоваться уровнем вычислительной производительности современных компьютеров недостаточно. Требуется отслеживать и изучать технологии, применяемые для новых достижений.

Имея это в виду, рассмотрим, как аппаратная архитектура компьютера ограничивает предельную скорость вычислений. В состав вычислительной системы входит большое число подсистем. Поэтому время, требуемое для выполнения задачи, будет определяться двумя факторами: скоростью работы каждой из подсистем в отдельности и временем передачи данных между подсистемами. На Рис. 4.5 показана в упрощённой форме структурная схема типового персонального компьютера с выделенными базовыми блоками, в основном определяющими скорость работы системы. *Центральный процессор (Central Processing Unit — CPU)* — это ядро компьютера. Как было сказано выше, в его состав входит более десяти регистров, каждый из которых служит для хранения 32-битных операндов. Кроме регистров в состав центрального процессора входят элементы, выполняющие базовые операции, такие как перемещение бит и целочисленная арифметика.

Для выполнения более сложных вычислительных процедур данные направляются в специальный аппаратный блок, называемый *математическим сопроцессором* или *арифметико-логическим устройством (АЛУ)*. АЛУ и центральный процессор могут быть реализованы на одном кристалле или выполнены в виде разных микросхем. Приведём пример. Сложение двух чисел в формате с плавающей точкой потребует от центрального процессора передать 8 байт данных (каждое число представляется 4 байтами) в математический сопроцессор. Также потребуются несколько байтов, несущих информацию о том, какие действия с переданными операндами необходимо осуществить. Спустя небольшое время, требуемое для реализации вычислений, АЛУ возвращает в центральный процессор 4 байта —



**Рис. 4.5.** Структура типовой вычислительной системы. Скорость вычислений ограничена, во-первых, скоростью работы составных функциональных блоков и, во-вторых, скоростью обмена данными между блоками.

число в формате с плавающей точкой, представляющее собой результат сложения. Обычно недорогие модели компьютеров не включают математического сопроцессора или предлагают его опционально. К примеру, процессор 80486DX имеет встроенный сопроцессор, а в процессоре 80486SX он отсутствует. Такие устройства выполняют программно те задачи, которые при наличии сопроцессора решаются аппаратно. Каждое математическое действие разбивается на ряд элементарных операций, которые могут быть непосредственно выполнены центральным процессором. Уровень производительности при этом, естественно, снижается. При том же конечном результате обработки время вычислений увеличивается ориентировочно в 10...20 раз.

Большинство программ для персонального компьютера способны выполнятьсь независимо от того, присутствует в системе математический сопроцессор или нет. Это достигается с помощью компилятора, генерирующего машинный код для обоих случаев и сохраняющего его в выходном исполняемом файле. Если сопроцессор присутствует в конкретной системе, на которой организуется выполнение программы, то используется один участок кода. Если сопроцессора нет — используется другой участок. Есть, однако, возможность дать компилятору указание генерировать исполняемый код только для одного требуемого случая. Поэтому иногда можно столкнуться с программами, которые требуют наличия математического сопроцессора и приводят к сбоям системы, если сопроцессор отсутствует. Такие программы, как текстовые редакторы, обычно мало выигрывают от наличия встроенного математического сопроцессора. В их основе лежат операции с перемещением данных в памяти, а не вычисления сложных математических выражений. Аналогично программы, работающие с целыми числами, не нуждаются в наличии сопроцессора, поскольку вычисления в формате с фиксированной точкой могут выполняться и центральным процессором. С другой стороны, скорость вычислений в алгоритмах цифровой обработки сигналов и других приложениях, требующих использования формата с плавающей точкой, может различаться на порядок для компьютеров, включающих сопроцессор и не имеющих его.

Центральный процессор и основная память обычно выполняются в виде двух разных микросхем. По очевидным причинам желательно иметь быстродействую-

щую память очень большого объёма. Однако эти желания ограничиваются высокой стоимостью микросхем. Скорость обмена данными между ЦП и основной памятью — это всегда одно из самых слабых мест системы с точки зрения скорости вычислений. Центральный процессор запрашивает данные, лежащие в основной памяти по указанному им адресу, и входит в режим ожидания. Чтобы сократить потери времени, связанные с ожиданием данных из основной памяти, общепринятым решением является использование промежуточной *кэш-памяти*. Кэш-память представляет собой небольшую по объёму, очень быстродействующую память, используемую в качестве буфера между ЦП и основной памятью. Ёмкость кэш-памяти обычно составляет несколько сот килобайт. При поступлении запроса от ЦП на загрузку данных из основной памяти, расположенных по некоторому адресу, быстро действующая электроника вместе с загрузкой требуемых данных осуществляет копирование фрагмента данных вокруг этого адреса из основной памяти в кэш-память. Вероятность того, что при следующем запросе ЦП будет требовать данные, расположенные в пределах этого фрагмента, довольно высока. Скорость загрузки данных при последующих обращениях существенно возрастает. Принцип использования кэш-памяти основан на том факте, что обычно на обработку поступают массивы данных, расположенных в памяти друг за другом. Этот приём для типовых программных решений способен повысить скорость вычислений в целом в несколько раз. Кэш-память выполняется на одном кристалле с центральным процессором или может быть реализована в виде автономного устройства.

Скорость, с которой происходит обмен данными между отдельными подсистемами компьютера, определяется числом независимых каналов, предназначенных для передачи, и максимально возможной скоростью обмена цифровыми сигналами в каждом из этих каналов. Обычно скорость передачи цифровых сигналов внутри кристалла существенно выше скорости обмена с внешними микросхемами. Аналогично скорость передачи данных между разными микросхемами одной печатной платы обычно выше скорости обмена между микросхемами разных плат, соединённых с помощью разъёмов (взаимодействие нескольких печатных плат может реализовываться, например, с помощью шин). Это является серьёзным основанием для стремления как можно больше аппаратных узлов реализовать в рамках одного кристалла центрального процессора.

Одной из особенно неприятных проблем, ограничивающих скорость компьютеров, является требование так называемой обратной совместимости. Когда некоторая фирма, работающая на рынке вычислительных систем, выпускает новый продукт, например аппаратный модуль сбора данных или программное обеспечение, в её интересах заинтересовать в покупке этого продукта как можно большее число потребителей. Это заставляет производителя обеспечивать совместимость нового продукта с подавляющим большинством используемых в данный момент компьютерных систем, которые могут на два-три поколения отставать от передовых технологий. Необходимость соблюдения обратной совместимости может существенно ограничивать производительность аппаратных и программных продуктов, снижая её до уровня наиболее старых охватываемых моделей. Представьте, например, что вы приобрели плату ввода/вывода, устанавливаемую в шинный разъём вашего компьютера Pentium (1 ГГц), которая обеспечивает приём и передачу данных по восьми цифровым каналам со скоростью 1 байт за такт. Вы разрабатываете программу на языке Ассемблера, организующую быстрый обмен

данными между вашим ПК и некоторым внешним устройством, например испытательным или научным оборудованием или просто другим компьютером. К своему большому удивлению, вы обнаруживаете, что максимальная скорость передачи данных оказывается не более 100 000 байт в секунду, что в 10 000 раз меньше тактовой частоты процессора! Виновницей вашего разочарования является шина стандарта ISA, которая была разработана в начале 1980-х годов и до сих пор поддерживается по причине обратной совместимости.

**В Табл. 4.3** приведены значения времени выполнения ряда операций, характерные для нескольких типов процессоров. Естественно, эти данные следует рассматривать как достаточно грубые оценки. Время вычислений на вашем компьютере будет зависеть от конкретного используемого аппаратного и программного обеспечения. Если вы хотите получить подобные характеристики для своей системы, вам следует проводить измерения именно в своей системе. Делается это очень просто. Напишите программный цикл, выполняющий миллион раз интересующее вас действие, и с помощью обычных часов определите время его выполнения. Первые три системы — это процессоры 80286, 80486 и Pentium для обычных персональных компьютеров, появившихся в 1986, 1993 и 1996 годах соответственно. Четвертый микропроцессор — это ADSP-TS101 фирмы Analog Devices, разработанный в 2000-м году специально для решения задач цифровой обработки сигналов.

Процессор Pentium оказывается производительнее процессора 80286 за счёт четырёх факторов: большая тактовая частота, большая разрядность шины данных, наличие кэш-памяти и более эффективная внутренняя организация, позволяющая тратить меньшее число тактов на выполнение одной команды.

Таблица 4.3. Оценка времени вычислений на различных моделях процессоров

Наименование операции	80286 (12 МГц)	80486 (33 МГц)	PENTIUM (100 МГц)	ADSP-TS101 (250 МГц) [нс]
Фиксированная точка				
$A\% = B\% + C\%$	1.6	0.12	0.04	
$A\% = B\% - C\%$	1.6	0.12	0.04	
$A\% = B\% \times C\%$	2.7	0.59	0.13	
$A\% = B\% \div C\%$	64	9.2	1.5	
Плавающая точка				
$A = B + C$	33	2.5	0.50	4
$A = B - C$	35	2.5	0.50	4
$A = B \times C$	35	2.5	0.50	4
$A = B \div C$	49	4.5	0.87	32
$A = \text{SQR}(B)$	45	5.3	1.3	56
$A = \text{LOG}(B)$	186	19	3.4	196
$A = \text{EXP}(B)$	246	25	5.5	152
$A = B^C$	311	31	5.3	148
$A = \text{SIN}(B)$	262	30	6.6	152
$A = \text{ARCTAN}(B)$	168	21	4.4	244

1) Время приведено в микросекундах.

2) Все устройства включают математический сопроцессор.

Если процессор Pentium можно считать «Кадиллаком» компьютерного мира, то ADSP-TS101 — это «Феррари»: меньше комфорта, но потрясающая скорость. Эта микросхема является представителем класса микропроцессоров, специально разработанных для выполнения задач цифровой обработки сигналов с наименьшими затратами времени. Другими известными процессорами этой категории являются, например, TIC55xx фирмы Texas Instruments и DSP56800E фирмы

Motorola. Эти специализированные микропроцессоры известны под названием *цифровые сигнальные процессоры (ЦСП)* или *процессоры цифровой обработки сигналов (ПЦОС)*. В англоязычной литературе распространён термин *DSP*, употребляющийся как для обозначения цифрового сигнального процессора (*Digital Signal Processor*), так и для самой цифровой обработки сигналов как научного направления (*Digital Signal Processing*). К подобным микросхемам иногда применяют также название *RISC-процессоры* или *процессоры с сокращённым набором команд (Reduced Instruction Set Computer)*. Это название отражает тот факт, что увеличение скорости в этих системах достигается, в частности, за счёт уменьшения числа ассемблерных команд, используемых для программирования. В противовес *RISC-процессорам* традиционные процессоры, такие как Pentium, называются *CISC-процессорами* или *процессорами с полным набором команд (Complex Instruction Set Computer)*.

Цифровые сигнальные процессоры могут применяться для построения вспомогательных модулей обработки сигналов, работающих под управлением компьютера общего назначения, либо использоваться в качестве самостоятельного процессора, встраиваемого в специализированное оборудование, такое как, например, сотовый телефон. Некоторые модели ЦСП работают только с числами в формате с фиксированной точкой. Другие модели поддерживают и целочисленную арифметику, и арифметику с плавающей точкой. К характерным чертам аппаратной организации цифровых сигнальных процессоров, наделяющим их беспрецедентным быстродействием, относятся: размещение внутри кристалла набора быстродействующих блоков кэш-памяти, разделение шин программ и данных, которое обеспечивает одновременный доступ к командам и операндам (так называемая *гарвардская архитектура*), включение во внутреннюю архитектуру блоков быстрых математических вычислений (ускорителей или акселераторов), использование принципа *конвейерного выполнения команд*.

При использовании конвейерной обработки процесс выполнения команды разбивается на ряд последовательных этапов. Например, сложение двух чисел может выполняться в три этапа. На первом этапе происходит выборка чисел из памяти. На втором этапе непосредственно выполняется сложение. На третьем этапе полученный результат записывается в память процессора. Если каждый этап обработки выполняется за один такт, то общее время выполнения команды составляет три такта. Принцип конвейерной обработки основан на том, что следующая команда может начать выполняться раньше, чем полностью выполнится предыдущая команда. В нашем примере мы можем начать сложение двух других чисел сразу после того, как команда сложения первых двух чисел пройдёт этап выборки операндов из памяти, т. е. по окончании первого такта. При большом числе последовательно выполняемых команд мы получим, что новый результат сложения чисел появляется на каждом такте, несмотря на то, что в действительности время выполнения каждой команды сложения составляет три такта. Использование принципа конвейерной обработки существенно повышает быстродействие процессора, однако оно затрудняет его программирование. Алгоритм, реализуемый в программе, должен позволять начинать новые вычисления до того, как станет известен результат предыдущих, ещё находящихся в конвейере вычислений.

Более детальному описанию цифровых сигнальных процессоров посвящены Главы 28 и 29. Это удивительные устройства. Их высокая производительность и одновременно небольшая цена обеспечивают проникновение технологии цифро-

вой обработки сигналов в широкий ряд разнообразных сфер применения, включая потребительские и научные приложения. Цифровая обработка сигналов — это одна из ключевых технологий, которая будет определять развитие электроники в XXI веке.

## 4.7. Скорость вычислений: советы для программистов

Аппаратная организация вычислительной системы и используемый язык *программирования*, безусловно, оказывают огромное влияние на скорость обработки данных. Однако, выбрав однажды тип процессора и язык программирования, вы не скоро вновь вернётесь к этому вопросу. Иначе обстоит дело с тем, как вы программируете. Те или иные приёмы программирования вы можете изменить в любое время, что также во многом определяет быстродействие системы. Здесь есть три рекомендации.

1. *Используйте целочисленные переменные вместо переменных в формате с плавающей точкой* везде, где это возможно. Микропроцессоры общего назначения, например те, что применяются в персональных компьютерах, работают с целочисленными переменными в 10...20 раз быстрее, чем с вещественным форматом. Для систем, не имеющих математического сопроцессора, это соотношение может составлять 200 к 1. Исключением является только деление целочисленных переменных, при котором оба операнда переводятся в *формат с плавающей точкой*. Целочисленное деление выполняется намного дольше других операций в *формате с фиксированной точкой* (см. **Табл. 4.3**).

2. *Избегайте использования таких функций, как  $\sin(x)$ ,  $\log(x)$ ,  $y^x$  и др.* Это трансцендентные функции, вычисляемые как комбинация большого числа простейших арифметических действий: сложения, вычитания и умножения. Например, с помощью разложения в ряд Макларена вычисляют следующие функции:

$$\begin{aligned}\sin(x) &= x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \frac{x^{11}}{11!} + \dots \\ \cos(x) &= 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \frac{x^{10}}{10!} + \dots \\ e^x &= 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \dots\end{aligned}\tag{4.3}$$

Разложение трансцендентных функций в ряд Макларена. Таким образом подобные функции вычисляются в компьютерах. Это объясняет, почему их выполнение требует много времени.

Хотя приведённые ряды являются бесконечными, они быстро убывают, так что их младшими членами можно пренебречь, например:

$$\sin(1) = 1 - 0.16666 + 0.008333 - 0.000198 + 0.000002 - \dots$$

Такие функции вычисляются примерно в 10 раз дольше, чем обычное сложение или умножение (см. **Табл. 4.3**). Можно использовать некоторые хитрости, чтобы исключить или упростить сложные вычисления. Например:  $x^3 = x \cdot x \cdot x$ ;

$\sin(x) \approx x$  при  $x << 1$ ;  $\sin(-x) = -\sin(x)$ . Последнее выражение эффективно, если известно одно из значений и требуется вычислить другое. Можно привести и другие примеры. В большинстве языков программирования непосредственно реализовано лишь небольшое число трансцендентных функций. Для вычисления других функций можно воспользоваться **Табл. 4.4**. Естественно, такие производные функции окажутся ещё медленнее.

Таблица 4.4. Вычисление редко используемых функций на основе базовых

Функция	Формула для расчёта
Секанс ( $X$ ) =	$1/\text{COS}(X)$
Косеканс ( $X$ ) =	$1/\text{SIN}(X)$
Котангенс ( $X$ ) =	$1/\text{TAN}(X)$
Арксинус ( $X$ ) =	$\text{ATN}(X/\text{SQR}(1-X^2))$
Арккосинус ( $X$ ) =	$-\text{ATN}(X/\text{SQR}(1-X^2)) + \text{PI}/2$
Аркsecанс ( $X$ ) =	$\text{ATN}(\text{SQR}(X^2-1)) + (\text{SGN}(X)-1) * \text{PI}/2$
Арккосеканс ( $X$ ) =	$\text{ATN}(1/\text{SQR}(X^2-1)) + (\text{SGN}(X)-1) * \text{PI}/2$
Арккотангенс ( $X$ ) =	$-\text{ATN}(X) + \text{PI}/2$
Гиперболический синус ( $X$ ) =	$(\text{EXP}(X)-\text{EXP}(-X))/2$
Гиперболический косинус ( $X$ ) =	$(\text{EXP}(X)+\text{EXP}(-X))/2$
Гиперболический тангенс ( $X$ ) =	$(\text{EXP}(X)-\text{EXP}(-X))/(\text{EXP}(X)+\text{EXP}(-X))$
Гиперболический секанс ( $X$ ) =	$1/\text{Гиперболический косинус}$
Гиперболический косеканс ( $X$ ) =	$1/\text{Гиперболический синус}$
Гиперболический котангенс ( $X$ ) =	$1/\text{Гиперболический тангенс}$
Гиперболический арксинус ( $X$ ) =	$\text{LOG}(X+\text{SQR}(X^2+1))$
Гиперболический арккосинус ( $X$ ) =	$\text{LOG}(X+\text{SQR}(X^2-1))$
Гиперболический арктангенс ( $X$ ) =	$\text{LOG}((1+X)/(1-X))/2$
Гиперболический аркsecанс ( $X$ ) =	$\text{LOG}((\text{SQR}(1-X^2)+1)/X)$
Гиперболический арккосеканс ( $X$ ) =	$\text{LOG}(1+\text{SGN}(X)*\text{SQR}(1+X^2))/X$
Гиперболический арккотангенс ( $X$ ) =	$\text{LOG}((X+1)/(X-1))/2$
Десятичный логарифм ( $X$ ) =	$\text{LOG}10(X) = \text{LOG}(X)/\text{LOG}(10) = 0.4342945 \text{ LOG}(X)$
Число Пи	$\text{PI} = 4*\text{ATN}(1) = 3.141592653589794$

1) Углы измеряются в радианах.

2) Обозначение  $\text{ATN}(X)$  соответствует функции арктангенса,  $\text{LOG}(X)$  — натуральному логарифму, функция  $\text{SGN}(X)$  вычисляет знак переменной  $X$ :  $\text{SGN}(X) = -1$  при  $X \leq 0$  и  $\text{SGN}(X) = 1$  при  $X > 0$ ,  $\text{EXP}(X) = e^x$ .

Другим подходом к вычислению сложных функций является предварительный расчёт и формирование в памяти таблиц значений этих функций — *справочных таблиц (look-up table)*. Рассмотрим пример 8-битной системы сбора данных, измеряющей падение напряжения на некотором резисторе. Пусть требуется определить рассеиваемую этим резистором мощность. Интересующий нас параметр может рассчитываться на основе измеряемой величины напряжения по формуле  $P = V^2/R$ . Однако более быстрым способом определения величины  $P$  будет использование табличных значений функции  $P(V)$ . Рассеиваемая мощность заранее рассчитывается по приведённой формуле для всех 256 возможных измеряемых значений напряжения и хранится в памяти в виде таблицы. В процессе работы системы измеренное напряжение принимает целые значения от 0 до 255, которые становятся индексами элементов таблицы и используются для считывания из неё

соответствующего значения мощности. Такой метод позволяет в сотни раз ускорить процесс вычислений по сравнению с непосредственным расчётом функции.

*3. Изучите, какие процедуры выполняются в вашей конкретной системе быстро, а какие медленно.* Обычно это приходит с опытом работы с определённой аппаратурой или программной средой и часто оказывается сюрпризом. Особое внимание уделяйте функциям работы с графикой и командам ввода/вывода. Существует несколько способов реализации таких процедур, которые могут сильно отличаться друг от друга по скорости выполнения. Например, в языке Бейсик есть команда **BLOAD**, реализующая непосредственную запись файла данных в заданную область памяти. Более традиционная передача данных байт за байтом (реализуемая в цикле) из того же файла в память оказывается в 100 раз медленнее. Другим примером может служить команда языка Бейсик **LINE**, отображающая на экране прямоугольник заданного цвета. Рисование того же прямоугольника пиксель за пиксели также будет осуществляться в 100 раз дольше. Простая вставка команды **PRINT** в тело цикла (позволяющая отслеживать, на какой стадии выполнения находится программа) может снизить скорость в тысячи раз!

# Глава 5

## ЛИНЕЙНЫЕ СИСТЕМЫ

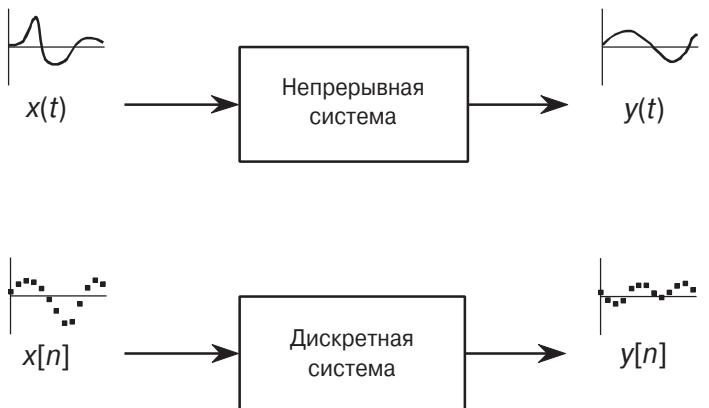
Широкое распространение в цифровой обработке сигналов получил принцип суперпозиции — декомпозиции (т. е. разложения) сложных сигналов на ряд простых компонентов. Полученные компоненты обрабатываются по отдельности, а результаты их обработки объединяются, формируя общий результат. Разбиение одной сложной задачи на ряд простых даёт колossalный эффект. Принцип суперпозиции, однако, справедлив только для линейных систем — таких систем, для которых выполняются определённые математические законы. К счастью, большинство приложений, с которыми на практике имеют дело ученые и инженеры, попадают в категорию линейных систем. Данная глава посвящена фундаментальным понятиям цифровой обработки сигналов: описанию линейных систем, различным видам декомпозиции, методам обработки сигналов, использующим принцип суперпозиции.

### 5.1. Сигналы и системы

Сигнал в общем случае представляет собой изменение состояния некоторого объекта, служащее для отображения и передачи информации. Описываются сигналы зависимостью одной величины от другой. Например, сигналом может являться изменение напряжения во времени в электрической цепи или зависимость яркости точки от её координат на изображении. Системой обработки сигналов называется процесс, преобразующий один сигнал (входной) в другой (выходной) (см. Рис. 5.1). *Непрерывные системы* принимают и генерируют *непрерывные сигналы*, характерные для аналоговой электроники. *Цифровые системы* имеют на входе и выходе *цифровые сигналы*, такие как массивы данных, с которыми оперируют компьютерные программы.

При обозначении сигналов используется ряд правил. В практике ЦОС этим правилам следуют не всегда, но, поскольку они являются общепринятыми, следует их запомнить. Математика обычно очень строго относится к обозначениям. Во-первых, при обозначении непрерывных сигналов используются круглые скобки, например  $x(t)$  или  $y(t)$ , а при обозначении дискретных сигналов — квадратные скобки:  $x[n]$  или  $y[n]$ . Во-вторых, сигналы обозначаются строчными буквами. Прописные буквы зарезервированы для представления сигналов в частотной области, что будет обсуждаться в последующих главах. В-третьих, для обозначения сигналов обычно используются «говорящие» символы. Например, зависимость напряжения (voltage) от времени (time) записывают как  $v(t)$ , а, скажем, изменение рыночной стоимости (price) товара изо дня в день (day) обозначают  $p(d)$ .

Понятиями сигналов и систем часто оперируют, не имея точной информации об их параметрах. Аналогично в математике используют переменные  $x$  и  $y$ , не под-



**Рис. 5.1.** Иллюстрация понятий сигналов и систем. Система обработки — это любой процесс, генерирующий выходной сигнал как реакцию на входное воздействие. При обозначении непрерывных сигналов обычно используют круглые скобки, а при обозначении дискретных сигналов — квадратные. Для записи сигналов используют строчные буквы, резервируя прописные буквы для представления в частотной области, что будет обсуждаться в последующих главах. В случае, если нельзя использовать более удобные обозначения, входные сигналы называют  $x(t)$  и  $x[n]$ , а выходные —  $y(t)$  и  $y[n]$ .

разумевая под ними конкретных физических величин. Здесь возникает четвёртое правило обозначения сигналов: если в их обозначении не удаётся отразить явный физический смысл, то принято использовать символы  $x[n]$  и  $x(t)$  для входного сигнала дискретной и непрерывной систем и  $y[n]$  и  $y(t)$  — для выходного сигнала дискретной и непрерывной систем соответственно.

Существует много причин, по которым необходимо уметь сформировать аналитическое описание системы. Такое описание нужно в первую очередь при проектировании (создании) системы, например системы подавления шума при снятии электрокардиограммы, или устройства коррекции расфокусировки изображения, или системы подавления сигналов эха в процессе звукозаписи. Математическое описание необходимо также в случаях, если мы хотим оценить влияние системы на входной сигнал. Такая ситуация возникает, например, в телефонных сетях. Линия передачи рассматривается как система, оказывающая определённое искажающее воздействие на передаваемый сигнал, в результате чего воспроизведимый на принимающей стороне голос может существенно отличаться от произнесённого на передающей стороне. Если уметь определять влияние системы на сигнал, то можно научиться компенсировать его негативные последствия. Ещё одним примером необходимости аналитического описания систем может служить ситуация, когда возникает задача анализа некоторого физического процесса, представляемого как система обработки. Характерным примером этого могут служить радиолокация и гидролокация. Они используют сравнение переданного сигнала с принятым, отражённым от удалённого объекта, чтобы определить параметры этого объекта. В терминах теории систем это означает, что требуется найти систему, переводящую заданный входной сигнал в известный выходной сигнал (задача идентификации).

На первый взгляд может показаться, что перед нами стоит необъятная задача математического описания огромного числа всего множества встречающихся в

мире систем. Однако большинство систем, имеющих практическое значение, относится к категории *линейных систем*. Это очень важный факт. Он позволяет нам уйти от необходимости индивидуального анализа множества разнообразных, не связанных между собой типов систем и оперировать понятиями, характерными одновременно для всего класса линейных систем. Нашей первой задачей становится научиться определять, какие системы являются линейными, какую роль они играют в современной электронике и разработке программного обеспечения, а также какое место они занимают среди других систем обработки сигналов.

## 5.2. УСЛОВИЯ ЛИНЕЙНОСТИ СИСТЕМЫ

Система называется линейной, если она обладает двумя свойствами: свойством *однородности* (гомогенности) и свойством *аддитивности*. Для доказательства линейности системы достаточно показать, что эти два свойства ей присущи. Если система не обладает хотя бы одним из указанных свойств, она не является линейной. Ещё одно свойство — свойство *инвариантности к сдвигу* — хотя и не является неотъемлемым свойством линейных систем, но в большинстве методов ЦОС оказывается обязательным. Если вы встретите термин «линейная система» применительно к цифровой обработке сигналов, то считайте, что свойство инвариантности к сдвигу ей присуще, даже если это специально не оговорено, хотя из определения линейной системы этого не следует. Перечисленные свойства линейных систем позволяют построить математическую теорию линейных систем. Ниже в этой главе будет описан интуитивно более понятный смысл линейности. Сейчас же рассмотрим перечисленные свойства с позиции их математического описания.

Свойство однородности, означающее, что изменение амплитуды сигнала на входе системы приводит к аналогичному изменению амплитуды на выходе, иллюстрирует Рис. 5.2. На языке математики это звучит так: если входному сигналу  $x[n]$  соответствует выходной сигнал  $y[n]$ , то сигналу  $kx[n]$  на входе будет соответствовать сигнал  $ky[n]$  на выходе для любых сигналов  $x[n]$  и постоянной величины  $k$ .

Обычный резистор может служить хорошим примером как однородной, так и неоднородной системы. Если рассматривать его как систему, входом которой является сигнал напряжения на резисторе  $v(t)$ , а выходом — ток, протекающий через резистор  $i(t)$ , то такая система будет являться однородной. Это следует из закона Ома: если напряжение увеличивается или уменьшается на некоторую величину, то ток будет увеличиваться или уменьшаться пропорционально. Если же рассматривать тот же резистор как систему, на вход которой подан, как и прежде, сигнал напряжения  $v(t)$ , а выходом будет являться рассеиваемая мощность  $p(t)$ , то такая система будет уже неоднородной. Мощность пропорциональна квадрату напряжения. Увеличение входного сигнала вдвое приведет к четырёхкратному возрастанию рассеиваемой мощности. Такая система не является однородной, а следовательно, не может рассматриваться как линейная.

Свойство аддитивности иллюстрирует Рис. 5.3. Рассмотрим систему, которая переводит входной сигнал  $x_1[n]$  в выходной сигнал  $y_1[n]$ , а сигнал  $x_2[n]$ , отличный от  $x_1[n]$ , — в сигнал  $y_2[n]$ . О такой системе говорят, что она обладает свойством аддитивности, если для любых сигналов  $x_1[n]$  и  $x_2[n]$  сумме сигналов  $x_1[n] + x_2[n]$  на

ЕСЛИ

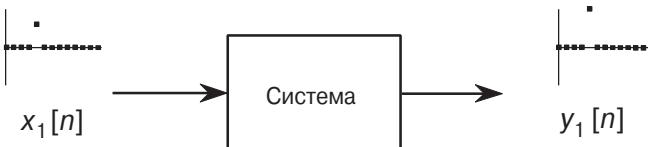


ТО



**Рис. 5.2.** Определение однородности. О системе говорят, что она однородная, если изменение амплитуды сигнала на входе вызывает аналогичное изменение амплитуды сигнала на выходе. Другими словами, если входному сигналу  $x[n]$  соответствует выходной сигнал  $y[n]$ , то сигналу  $kx[n]$  на входе будет соответствовать сигнал  $ky[n]$  на выходе для любых сигналов  $x[n]$  и постоянной величины  $k$ .

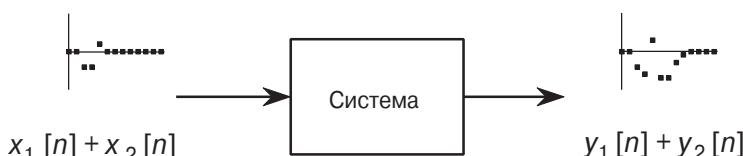
ЕСЛИ



И ЕСЛИ



ТО



**Рис. 5.3.** Определение аддитивности. О системе говорят, что она обладает свойством аддитивности, если суммируемые на входе сигналы проходят через неё без взаимного влияния друг на друга. Другими словами, если входному сигналу  $x_1[n]$  соответствует выходной сигнал  $y_1[n]$ , а входному сигналу  $x_2[n]$  — выходной сигнал  $y_2[n]$ , то сумме сигналов на входе  $x_1[n] + x_2[n]$  будет соответствовать сумма сигналов на выходе  $y_1[n] + y_2[n]$ .

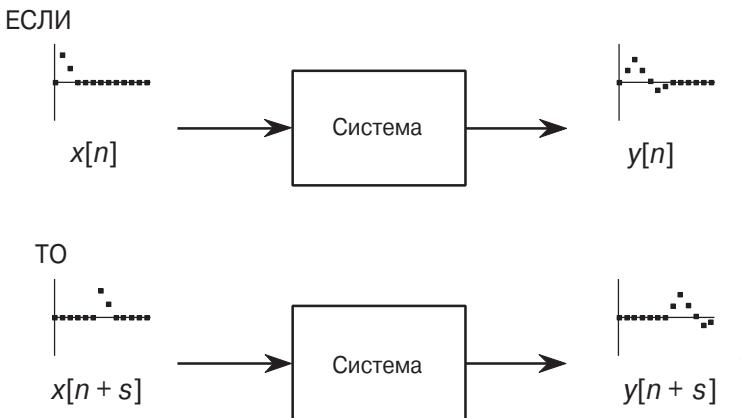
входе будет соответствовать сигнал  $y_1[n] + y_2[n]$  на выходе. Другими словами, сумма сигналов на входе приводит к суммированию выходных сигналов.

Ключевой особенностью систем, обладающих свойством аддитивности, является то, что сигналы, суммируемые на входе, проходят через них без взаимного влияния друг на друга. Что это означает на практике, рассмотрим на примере. Допустим, вы разговариваете по телефону с вашей тётушкой Эдной и дядюшкой Берни. Тётушка Эдна рассказывает вам свою длинную историю о том, какую замечательную редиску она вырастила в этом году. За её голосом вы слышите, как дядюшка Берни кричит на свою собаку, которая допустила оплошность в его любимом кресле. Оба голоса складываются и в виде электромагнитной волны передаются по телефонной линии связи. Эта система обладает свойством аддитивности. Поэтому на её выходе вы слышите два голоса так, как будто они передавались по отдельности. Вы слышите тётушку Эдну и дядюшку Берни, а не господина Эднаберни.

Хорошим примером системы, не обладающей свойством аддитивности, может служить смеситель, осуществляющий преобразование частоты в радиопередатчике. На его вход поступают два сигнала: аудиосигнал (речь или музыка) и несущее высокочастотное колебание, способное при излучении антенной распространяться в пространстве на большие расстояния. Эти два сигнала объединяются и поступают на нелинейный элемент, например диод. Происходит слияние сигналов так, что они образуют новый сигнал — модулированную радиоволну, способную переносить информацию на огромные расстояния.

Свойство инвариантности к сдвигу иллюстрирует **Рис. 5.4**. Оно означает, что сдвиг сигнала на входе системы приводит к такому же сдвигу сигнала на выходе. Более строго: если входному сигналу  $x[n]$  соответствует выходной сигнал  $y[n]$ , то сигналу  $x[n + s]$  на входе будет соответствовать сигнал  $y[n + s]$  на выходе для любых сигналов  $x[n]$  и постоянной величины  $s$ . Обратите внимание на математическую запись операции сдвига. Она ещё будет встречаться вам в последующих главах. Добавление постоянной величины  $s$  к аргументу  $n$  вызывает сдвиг сигнала вправо или влево вдоль оси времени, т. е. вызывает запаздывание или опережение сигнала. Если, например,  $s = 2$ , сигнал сдвигается влево, что вызывает опережение на два периода дискретизации. Если  $s = -2$ , сигнал сдвигается вправо — задерживается на два дискретных отсчёта.

Свойство инвариантности к сдвигу имеет большое значение, так как говорит о неизменности характеристик системы во времени (или с изменением другой какой-либо величины, представляющей аргумент). Если скачок сигнала на входе вызвал ответный скачок на выходе, то можно быть уверенным, что другой такой же скачок вызовет точно такую же реакцию системы. Большинство систем, с которыми вам придётся иметь дело, будут инвариантны к сдвигу. Это очень хорошо, ведь работать с системами, параметры которых меняются со временем, очень тяжело. Представьте себе, что вы разработали систему цифровой фильтрации, компенсирующую искажения, возникающие в телефонной линии связи. Ваш фильтр улучшает качество передачи голоса, делая его более естественным и разборчивым. К вашему немалому удивлению, с наступлением зимы качество работы системы понизилось. А дело в том, что из-за понижения температуры изменились параметры телефонной линии связи и произошла расстройка системы. В этих условиях желательно использовать другой, более сложный алгоритм обработки, адаптирующийся к изменяющимся условиям окружающей среды.



**Рис. 5.4.** Определение инвариантности к сдвигу. Система является инвариантной к сдвигу, если сдвиг сигнала на её входе вызывает такой же сдвиг сигнала на её выходе. Другими словами, если входному сигналу  $x[n]$  соответствует выходной сигнал  $y[n]$ , то сигналу на входе  $x[n+s]$  будет соответствовать сигнал  $y[n+s]$  на выходе для любых сигналов  $x[n]$  и постоянной величины  $s$ .

Однако почему свойства однородности и аддитивности являются обязательными для линейных систем, в то время как свойство инвариантности к сдвигу как бы вспомогательное? Это потому, что линейность является очень широким понятием, охватывающим далеко не только сигналы и системы. Рассмотрим, например, такую ситуацию. Некий фермер торгует апельсинами по 2 доллара за ящик и яблоками по 5 долларов за ящик. Если фермер будет продавать только апельсины, он будет получать 20 долларов за 10 ящиков и 40 долларов за 20 ящиков, что соответствует свойству однородности. Если фермер продаст 20 ящиков апельсинов и 10 ящиков яблок, он получит  $20 \times 2 + 10 \times 5 = 90$  долларов. Абсолютно столько же он заработал бы, если бы продавал апельсины и яблоки по отдельности. Это соответствует свойству аддитивности. Обладание свойствами однородности и аддитивности делает рассмотренный процесс линейным. Однако понятие сигналов в данном примеру не применимо, и системой его назвать нельзя. Свойство инвариантности к сдвигу в данном случае не имеет смысла. Таким образом, инвариантность к сдвигу следует считать дополнительным свойством, вводимым в рассмотрение в тех случаях, когда речь идёт о сигналах и системах.

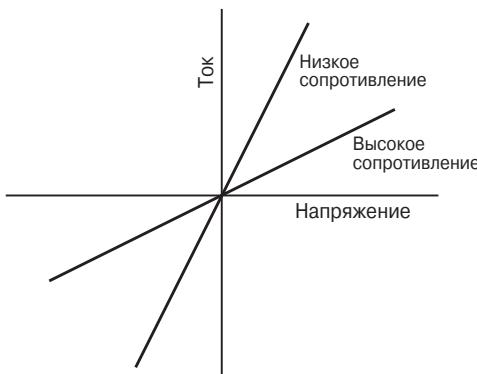
## 5.3. Статическая характеристика и передача гармонических сигналов

*Свойства однородности, аддитивности и инвариантности к сдвигу* являются основополагающими в теории линейных систем, так как они позволяют определить условия линейности математически. В то же время эти свойства не дают интуитивно понятного представления о линейности, приемлемого для инженерных приложений. Здесь приходят на помощь понятия линейности статической характеристики и неискажаемой передачи гармонических сигналов. С точки зрения математики эти понятия не имеют сколько-нибудь существенного значения, но они

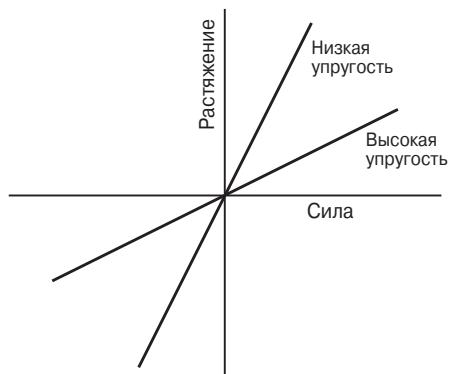
полезны на практике. Рекомендуем уделить особое внимание нижеизложенному материалу.

*Статическая характеристика* определяет, как линейная система реагирует на постоянное входное воздействие, например на постоянный ток или на статический сигнал. Отклик линейной системы на постоянный сигнал очень легко выражается математически: сигнал на выходе системы равен входному сигналу, умноженному на постоянную величину. При графическом изображении зависимость сигнала на выходе системы от возможных постоянных значений на входе представляет собой прямую линию, проходящую через начало координат. **Рис. 5.5** отображает два общеизвестных примера линейных зависимостей: закон Ома — для электрических цепей и закон Гука — для упругих тел. Для сравнения на **Рис. 5.6** показана статическая характеристика двух нелинейных систем: вольтамперная характеристика диода и кривая намагничивания железа.

а) Закон Ома



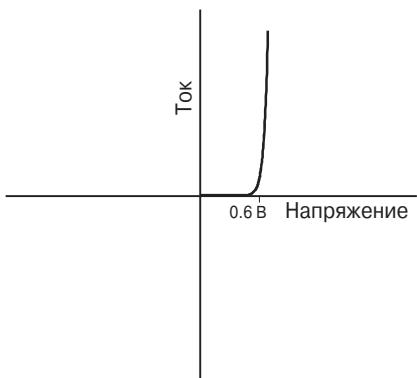
б) Закон Гука



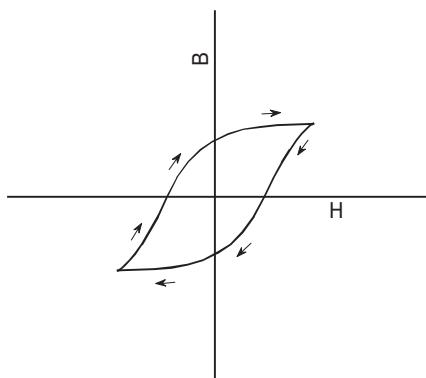
**Рис. 5.5.** Два примера линейности статической характеристики. На (а) отображен закон Ома: ток, протекающий через резистор, равен напряжению на резисторе, деленному на сопротивление. Закон Гука иллюстрирует (б): удлинение (укорочение) пружины равно произведению коэффициента упругости пружины и растягивающей (сжимающей) её силы.

Свойством линейности статической характеристики обладают все линейные системы. Обратное верно в большинстве случаев, но не всегда. Есть системы, которые ведут себя как линейные при постоянных воздействиях и перестают быть линейными в случае подачи на них сигналов другого типа. Тем не менее есть достаточно широкий класс систем, линейность которых полностью доказывается свойством линейности статической характеристики. Для таких систем не имеет значения, какие сигналы поданы на вход: постоянные или нет. Эти системы называются *системами без памяти*, поскольку их выход определяется только текущим значением сигнала на входе и не зависит от истории. Так, например, мгновенное значение тока, протекающего через резистор, определяется только мгновенным значением напряжения на резисторе и не зависит от того, какие значения они принимали до этого. Если система обладает свойством линейности статической характеристики и является системой без памяти, то она обязательно линейная. Это важное утверждение позволяет понять и доказать линейность систем, относящихся к рассмотренному классу.

а) Полупроводниковый диод



б) Железо



**Рис. 5.6.** Два примера нелинейного преобразования постоянного входного воздействия. На (а) показана вольтамперная характеристика полупроводникового диода, представляющая собой экспоненциальную взаимосвязь напряжения и тока. На (б) изображена петля гистерезиса: взаимосвязь напряжённости магнитного поля  $H$  и магнитной индукции  $B$  железа. Напряжённость магнитного поля определяется не только текущим значением индукции, но и предыдущим состоянием свойств материала.

Если на вход линейной системы подать гармонический сигнал, то на её выходе будет наблюдаться также гармонический сигнал с той же частотой. Это важное свойство линейных систем мы назовём свойством *неискажаемой передачи гармонических сигналов*. Свойство неискажаемой передачи линейных систем распространяется только на гармонические сигналы. Не стоит ожидать, что, например, при подаче прямоугольного импульса на вход линейной системы на её выходе также окажется прямоугольный импульс. Гармонический сигнал на выходе гарантирует появление гармонического сигнала на выходе, который, однако, может отличаться от входного сигнала амплитудой и фазой. Это может быть знакомо вам из электроники: электрическая цепь описывается частотной характеристикой — графическим отображением зависимости коэффициента усиления и поворота фазы от частоты.

Возникает вопрос: верно ли обратное? Если система всегда переводит гармоническое воздействие в гармонический сигнал на выходе, означает ли это, что она обязательно линейна? Ответ — нет, однако из этого правила есть исключения, их очень немного, и обычно они очевидны. Представьте себе, что в вашу систему вселился «злой дух», который хочет ввести вас в заблуждение и выдать нелинейную систему за линейную. Он имеет при себе осциллограф для наблюдения за входными сигналами и генератор гармонических сигналов, выход которого служит выходом системы. Вы подаёте на вход синусоидальный сигнал, и он тут же измеряет его частоту и генерирует на выходе системы соответствующий гармонический сигнал. Но эта система, конечно же, нелинейна. Она не обладает свойством *аддитивности*. Чтобы доказать это, подайте на вход сумму двух гармонических воздействий. Выходом системы может быть сигнал только одной частоты. Этот пример не такой надуманный, как вам могло показаться. Подобным образом работают широко распространённые системы фазовой автоподстройки частоты.

Чтобы лучше понять, какие системы являются линейными, рассмотрим практическую задачу. Вы работаете с некоторым устройством и хотите ответить на воп-

рос: это *система линейная* или нет? Первое, что вам нужно будет сделать, — это подключить ко входу системы генератор гармонических сигналов, а к выходу — осциллограф. Подайте на вход синусоидальный сигнал и убедитесь, что выходной сигнал также является синусоидальным. Он не должен быть ограничен по амплитуде, нижняя и верхняя половины должны быть симметричны, не должно быть искажений в точках пересечения нуля и т. д. Теперь изменяйте амплитуду входного воздействия и наблюдайте за выходной реакцией. Если система линейна, то амплитуда сигнала на выходе должна изменяться аналогично амплитуде сигнала на входе. И наконец, вам следует провести наблюдения при различных частотах входного сигнала. Выходной сигнал должен менять свою частоту так же, как меняется частота на входе. При переменной частоте входного сигнала реакция системы, повторяя входную частоту, может существенно отличаться амплитудой и фазой, но это не говорит о нелинейности системы. Некоторые частоты на входе могут вообще давать ноль на выходе, что может соответствовать гармоническим сигналам с требуемой частотой и нулевой амплитудой. Если все перечисленные особенности поведения наблюдаются вами для данной системы, то вам следует сделать вывод о её линейности. Это, конечно, не строгое математическое доказательство, но вероятность правильности такого решения будет достаточно высока.

## 5.4. Примеры линейных и нелинейных систем

Примеры наиболее распространённых линейных и нелинейных систем приведены ниже. Изучая их, помните о математическом определении линейности (свойства однородности, аддитивности и инвариантности к сдвигу) и об инженерном практическом подходе (свойства линейности статической характеристики и неискажаемой передачи гармонических сигналов).

### 5.4.1. Примеры линейных систем (процессов)

**Распространение волн**, например звуковых и электромагнитных.

**Электрические цепи**, состоящие из резисторов, конденсаторов и индуктивностей.

**Электронные схемы**, такие как усилители и фильтры.

**Механическое движение**, обусловленное взаимодействием инертных тел, действием сил растяжения или сжатия.

**Системы, описываемые дифференциальными уравнениями**, например цепи, включающие резисторы, конденсаторы и индуктивности.

**Умножение на постоянную величину**: усиление или затухание сигнала.

**Трансформация сигнала**: эхо-сигналы, явление резонанса, размытие изображения.

**Единичная система** (unity system): система, в которой выход всегда равен входу.

**Нулевая система**: система, в которой выход всегда равен нулю независимо от входа.

**Дифференцирование и интегрирование**, а также операции вычисления первой разности и текущего значения суммы для дискретных сигналов.

**Малые входные воздействия** обычно в нелинейных системах, например усиление слабого сигнала в транзисторе, настроенном соответствующим образом.

**Свёртка** — математическая операция, при которой каждый отсчёт выходного сигнала получается как сумма произведений входных отсчётов и набора весовых коэффициентов.

**Рекурсия** — похожая на свёртку процедура, отличающаяся, однако, тем, что при формировании текущего выходного отсчёта наряду с набором входных отсчётов используются также ранее полученные выходные.

## 5.4.2. Примеры нелинейных систем

Системы, не обладающие свойством линейности статической характеристики, например мощность рассеивания на резисторе, зависящая от падения напряжения:  $P = V^2 R$ , или излучение энергии нагретым телом, значение которой определяется температурой:  $R = kT^4$ , или прохождение света через прозрачный слой, в результате чего интенсивность светового потока выражается формулой  $I = e^{-\alpha t}$ .

Системы, не обладающие свойством неискажаемой передачи гармонических воздействий, например электрические цепи, реализующие пиковый детектор, квадратичный детектор, преобразователь гармонических сигналов в прямоугольные импульсы, множитель частоты и др.

**Типовые для электроники искажения:** ограничение амплитуды, искажения типа «ступенька».

**Перемножение сигналов**, например, в процессе амплитудной модуляции или автоматической регулировки усиления.

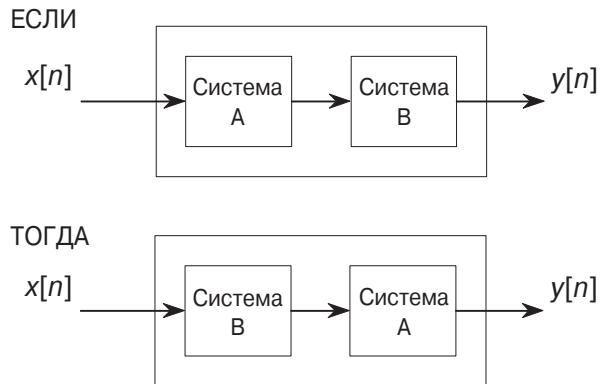
**Петля гистерезиса**, характерная для процессов намагничивания железа, вулканизации резины и др.

**Насыщение**, возникающее при подаче чрезмерно высоких уровней сигналов на вход, например, усилителя или трансформатора.

**Пороговые системы**, такие как цифровые логические элементы или детекторы сигналов по превышению заданного уровня мощности.

## 5.5. Особые свойства линейности

Линейность характеризуется *коммутативностью* — свойством, распространяющимся на комбинации двух и более систем. Рис. 5.7 иллюстрирует суть коммутативности. Представьте себе две системы, соединённые *каскадно* (*последовательно*), т. е. так, что выход первой системы является входом второй. Если каждая из этих систем линейна, то их комбинация также будет линейной системой. Коммутативность означает, что изменение порядка следования систем в каскадном соединении не влияет на характеристики всей комбинации в целом. Возможно, вам приходилось пользоваться этим принципом в электронике. Например, представим себе цепь, состоящую из двух звеньев — усиления и фильтрации. Как лучше реализовать такую схему: сначала включить усилитель, а потом фильтр или, наоборот, сначала провести фильтрацию, а затем усилить сигнал? Если оба этих звена линейны, то порядок осуществления операций не имеет значения и результат всей обработки будет одинаков в обоих случаях. Конечно, надо помнить, что все реальные электронные схемы нелинейны, и поэтому порядок следования этапов обработки сигнала часто имеет важное значение. К нелинейности реальных

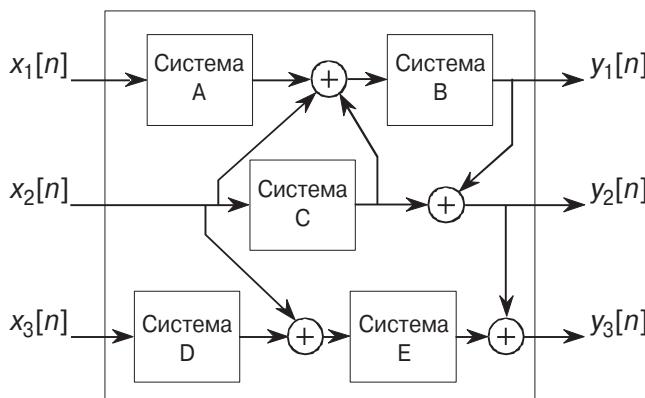


**Рис. 5.7.** Свойство коммутативности линейных систем. Если две или более линейные системы соединены каскадно, то порядок их следования не влияет на характеристики всего соединения в целом.

устройств приводит ряд распространённых эффектов: наложения, смещение по постоянному току, внутренние шумы, ограниченность скорости нарастания выходного напряжения и др.

**Рис. 5.8** иллюстрирует следующее свойство линейных систем, относящееся к системам со многими входами и выходами. Система со многими входами и/или выходами является линейной, если она образована соединением линейных подсистем и элементами сложения сигналов. Уровень сложности не имеет значения, лишь бы все процессы внутри системы были линейными.

Чтобы лучше понять значение свойства линейности для систем со многими входами и выходами, рассмотрим следующий пример. Пусть только на один из входов такой системы подан сигнал, в то время как остальные входные воздействия являются нулевыми. На выходах системы будем наблюдать набор реакций, вызванных таким воздействием. Подадим новое входное воздействие на другой вход системы, по-прежнему оставляя остальные входы, включая первый, нулевыми. На



**Рис. 5.8.** Любая система со многими входами и/или выходами является линейной, если она образована соединением линейных подсистем, взаимодействие между которыми сводится к сложению сигналов.

выходах системы вновь получим набор соответствующих реакций. Наконец, подадим оба входных воздействия на использованные в первых двух экспериментах входы системы одновременно. Наблюдаемая на выходе реакция окажется суперпозицией (суммой) реакций, наблюдавшихся нами в двух первых экспериментах.

Использование операций умножения, применительно к линейным системам, часто вносит некоторую неясность. Дело в том, что умножение может быть как линейным, так и нелинейным в зависимости от того, на что умножается сигнал. На Рис. 5.9 показаны оба случая. Система, в которой входной сигнал умножается на константу, является линейной. Такая система реализует усиление или ослабление сигнала, в зависимости от значения постоянного множителя: больше он единицы или меньше. В отличие от первой, система, в которой происходит умножение одного сигнала на другой, оказывается нелинейной. Представьте себе умножение синусоидального сигнала с одной частотой на синусоидальный сигнал с другой частотой. Очевидно, что на выходе такой системы синусоиды в чистом виде мы не увидим.

Другим распространённым фактором, часто затрудняющим понимание линейности, является добавление к сигналам паразитных составляющих, например постоянной составляющей или теплового шума. Делает ли наличие таких сигналов систему нелинейной? Ответ зависит от источника этих паразитных сигналов. Если они возникают внутри самой системы, то её следует считать нелинейной. Синусоидальное колебание, поданное на вход такой системы, на выходе не будет чистой синусоидой. В другом случае можно считать, что паразитное воздействие поступает извне наряду с входным сигналом по другому входу системы. Такую систему можно рассматривать как линейную, считая, что добавление паразитной составляющей — это сложение сигналов, разрешённое для линейных систем.

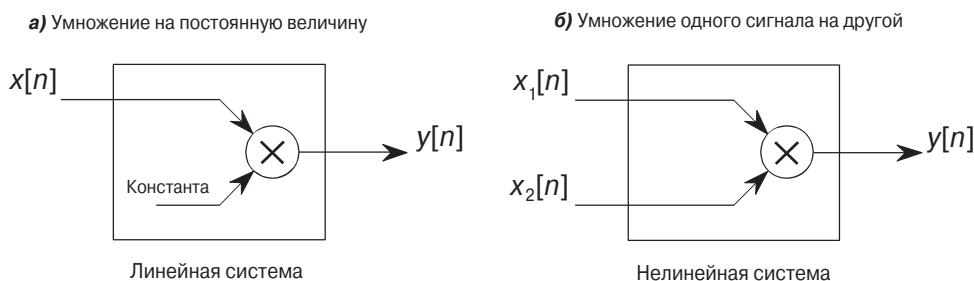


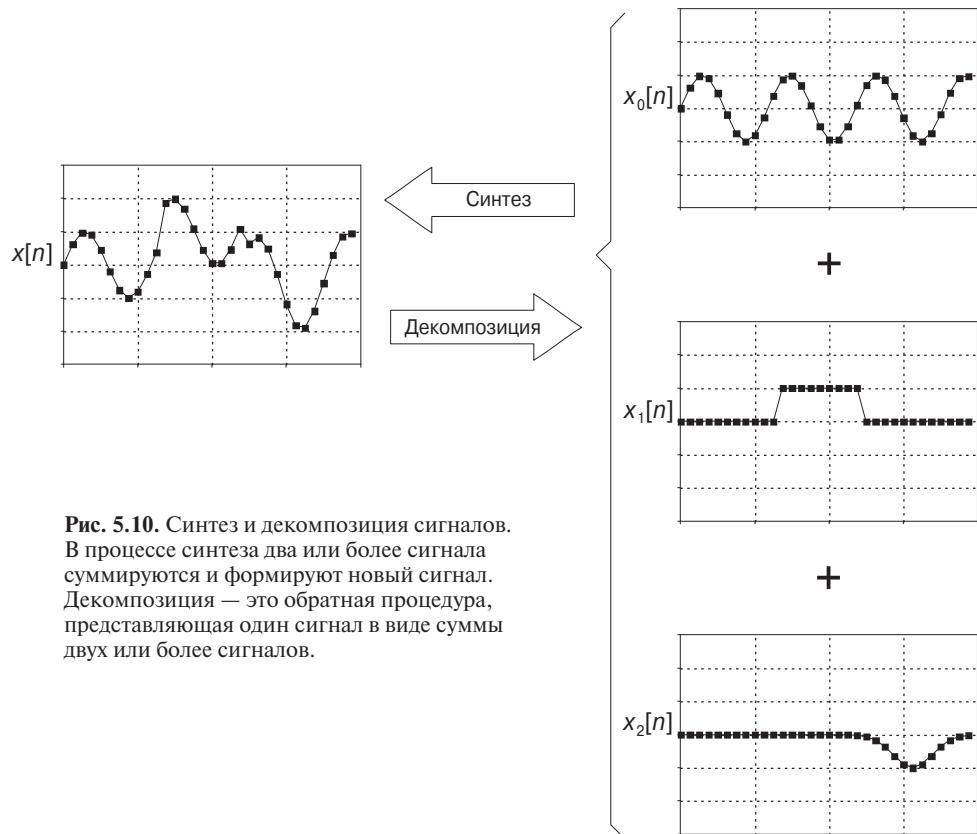
Рис. 5.9. Линейность операции умножения. Процедура умножения сигнала на постоянную величину является линейной. Умножение одного сигнала на другой делает систему нелинейной.

## 5.6. Принцип суперпозиции — фундаментальное понятие ЦОС

При работе в рамках теории линейных систем все взаимодействия сигналов сводятся исключительно к процедурам масштабирования (умножения сигналов на постоянную величину) и сложения. Другие преобразования, например умножение одного сигнала на другой, применять нельзя. На Рис. 5.10 приведён следующий пример: три сигнала —  $x_0[n]$ ,  $x_1[n]$  и  $x_2[n]$  суммируются и образуют четырёх

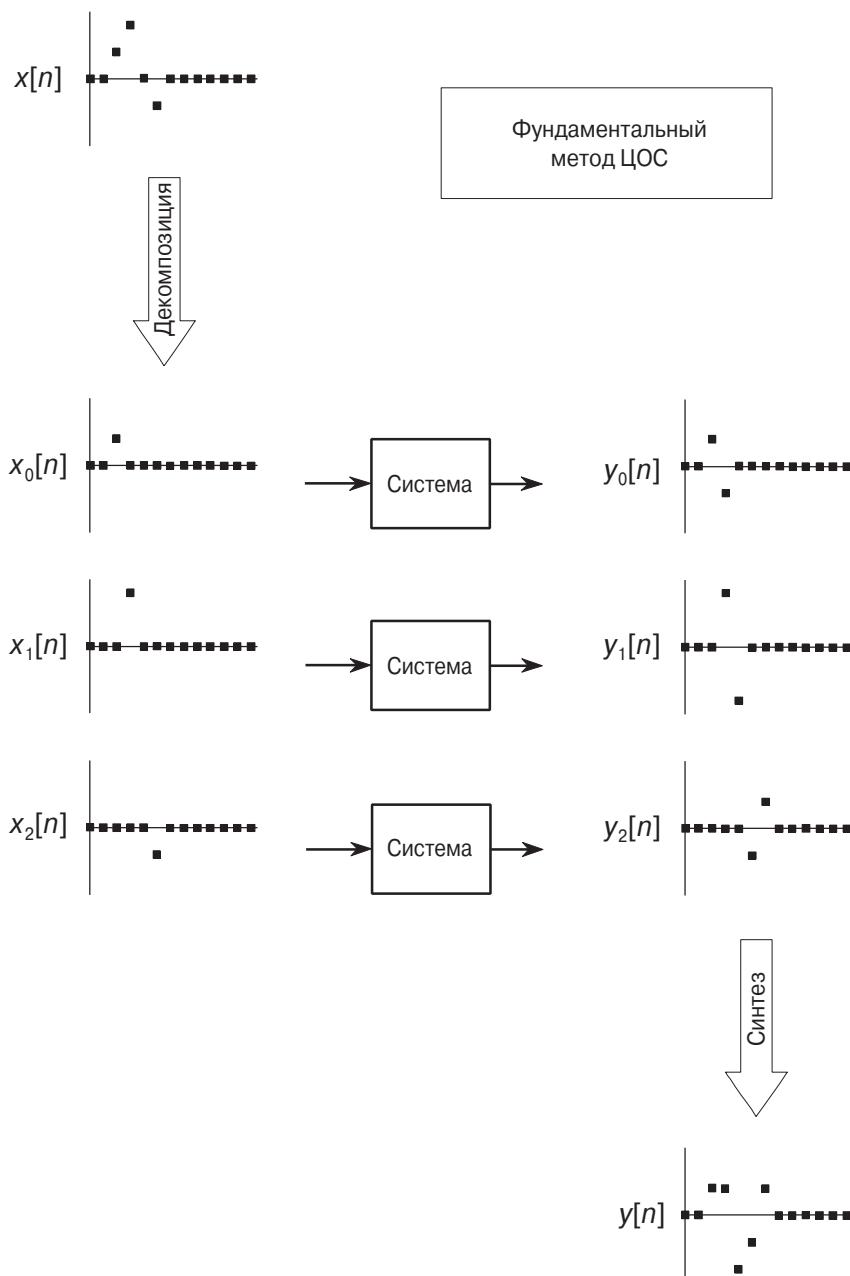
тый сигнал  $x[n]$ . Такой процесс формирования сигнала путём масштабирования и сложения других сигналов называется *синтезом*.

Обратная процедура представления одного сигнала в виде суммы двух или более компонент называется *декомпозицией* (разложением). Она является более сложной по сравнению с синтезом, поскольку для любого сигнала можно найти бесконечное множество возможных разложений. Например, на базе двух чисел — 15 и 25 — может быть получена (синтезирована) только одна сумма — число 40. В то же время число 40 может быть представлено в виде целого ряда разложений (декомпозиций):  $1 + 39$ ,  $2 + 38$ ,  $-30.5 + 60 + 10.5$  и т. д.



**Рис. 5.10.** Синтез и декомпозиция сигналов.  
В процессе синтеза два или более сигнала суммируются и формируют новый сигнал. Декомпозиция — это обратная процедура, представляющая один сигнал в виде суммы двух или более сигналов.

Итак, перейдём к рассмотрению фундаментального для ЦОС *принципа суперпозиции*, на использовании которого построен весь анализ *сигналов и систем*, применяемых в цифровой обработке сигналов. Рассмотрим входной сигнал  $x[n]$ , прохождение которого через линейную систему приводит к появлению на её выходе сигнала  $y[n]$ . Как показано на **Рис. 5.11**, сигнал  $x[n]$  может быть представлен в виде комбинации (или декомпозиции) набора более простых сигналов  $x_0[n]$ ,  $x_1[n]$ ,  $x_2[n]$ , ... . Назовём их *компонентами входного сигнала*. Каждый из компонентов входного сигнала, проходя через систему, вызывает на её выходах набор реакций в  $y_0[n]$ ,  $y_1[n]$ ,  $y_2[n]$ , ..., которые мы назовём *компонентами выходного сигнала*. Процедура синтеза позволяет получить выходной сигнал  $y[n]$  как комбинацию компонентов выходного сигнала.



**Рис. 5.11.** Фундаментальный метод ЦОС. Произвольный сигнал  $x[n]$  может быть представлен как декомпозиция двух или более сигналов, в данном примере:  $x_1[n]$ ,  $x_2[n]$  и  $x_3[n]$ . Прохождение этих сигналов через систему приводит к появлению на её выходе сигналов  $y_1[n]$ ,  $y_2[n]$  и  $y_3[n]$ . Сложение последних формирует выходную реакцию системы  $y[n]$ , точно такую же, как если бы на вход системы был подан сам сигнал  $x[n]$ , а не его декомпозиция.

А теперь очень важное утверждение: выходной сигнал  $y[n]$ , полученный описанным методом, полностью идентичен сигналу, который оказался бы на выходе системы при подаче на её вход самого сигнала  $x[n]$ . Это действительно замечательный факт. Вместо того чтобы пытаться описать прохождение через систему сложных входных воздействий, нам достаточно знать, как преобразуются в ней простые сигналы. В терминах цифровой обработки сигналов это означает, что любые входные и выходные воздействия могут быть представлены *суперпозицией* простых сигналов. Данная концепция лежит в основе практически всех подходов, применяемых в цифровой обработке сигналов.

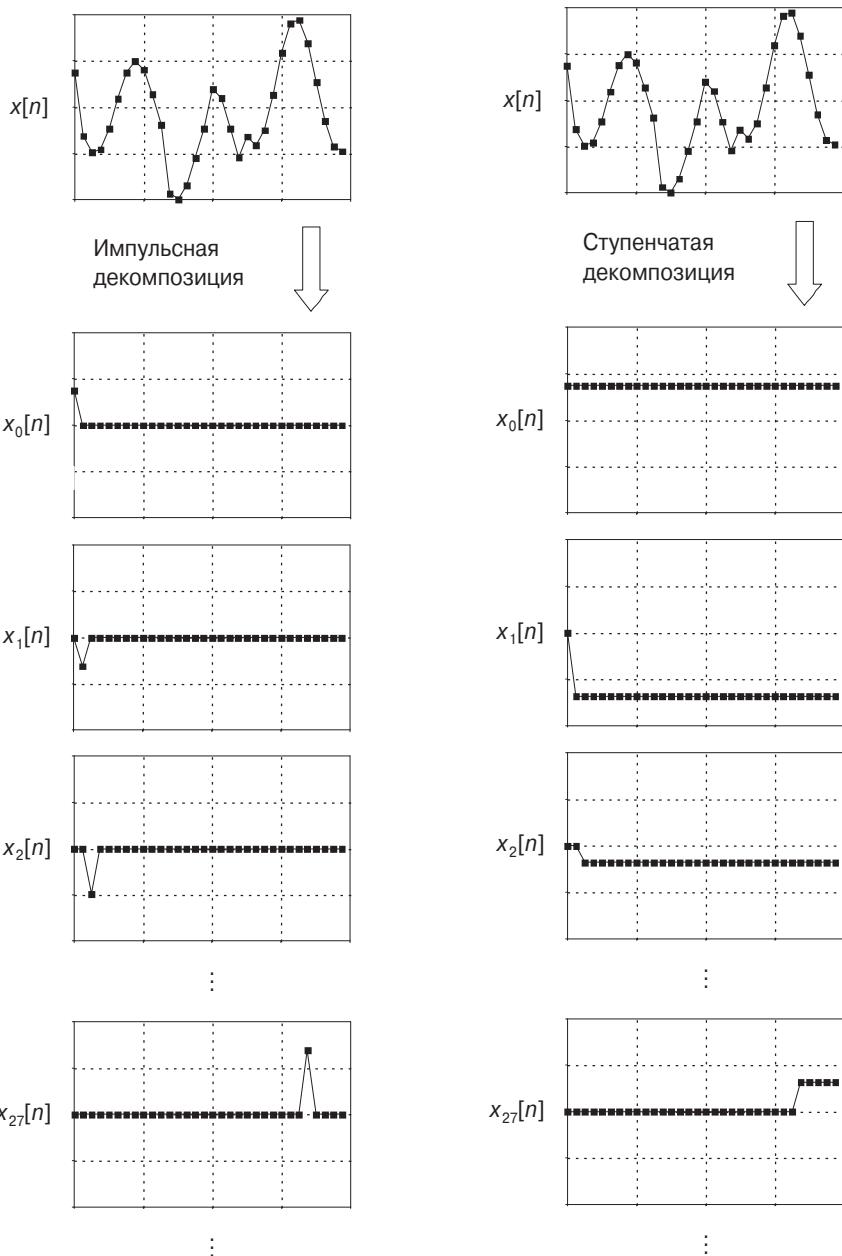
Простым примером применения принципа суперпозиции может служить умножение числа 2041 на число 4 так, как мы выполняем его в уме. Как мы реализуем вычисление? Возможно, кто-то представит себе 2041 счётную палочку, затем мысленно учтёт количество единиц и затем всё пересчитает. Большинство же упростит эту задачу, используя принцип суперпозиции. Число 2041 может быть разложено с помощью декомпозиции в сумму:  $2000 + 40 + 1$ . Каждое из этих чисел достаточно легко умножить на 4. Искомый результат мы получим из частных произведений путём синтеза:  $8000 + 160 + 4 = 8164$ .

## 5.7. Наиболее распространённые виды декомпозиции

Следует помнить о том, что целью рассмотренного подхода является упрощение задачи за счёт перехода от одной сложной проблемы к нескольким простым. Если декомпозиция не приводит в той или иной степени к упрощению, то она становится бессмысленна. В теории обработки сигналов наибольшее распространение получили два варианта декомпозиции: декомпозиция с использованием единичных импульсов и декомпозиция на основе разложения в ряд Фурье. Их детальному описанию посвящены следующие несколько глав. Существует и ряд других способов декомпозиции, используемых достаточно редко. Приведём краткое описание двух основных и трёх реже используемых методов декомпозиции.

### 5.7.1. Импульсная декомпозиция

*Импульсная декомпозиция* представляет сигнал, состоящий из  $N$  дискретных отсчётов, в виде совокупности  $N$  компонентов по  $N$  отсчётов каждый (Рис. 5.12), один из них взят из исходного сигнала, а все остальные равны нулю. Эти сигналы тесно связаны с понятием единичного импульса — дискретного сигнала, у которого нулевой отсчёт равен единице, а остальные равны нулю. Фактически компонентные сигналы импульсной декомпозиции представляют собой единичный импульс, сдвинутый во времени и умноженный на константу. Импульсная декомпозиция имеет огромное значение для теории цифровой обработки сигналов. Она позволяет работать с сигналами не целиком, а обрабатывать их отсчёт за отсчётом. Кроме того, она позволяет описывать системы с помощью реакции на единичный импульс — импульсной характеристики. Если известна реакция системы на единичный импульс, то с помощью импульсной декомпозиции можно получить реакцию на любое входное воздействие. Этот процесс осуществляется с помощью свёртки, о которой мы расскажем в следующих двух главах.



**Рис. 5.12.** Пример импульсной декомпозиции. Исходный сигнал, состоящий из  $N$  дискретных отсчётов, разбивается на  $N$  сигналов, каждый из которых имеет единственный ненулевой отсчёт.

**Рис. 5.13.** Пример ступенчатой декомпозиции. Исходный сигнал, состоящий из  $N$  дискретных отсчётов, разбивается на  $N$  сигналов, каждый из которых представлен ступенчатой функцией.

## 5.7.2. Ступенчатая декомпозиция

*Ступенчатая декомпозиция* также разбивает исходный сигнал длиной  $N$  дискретных отсчётов на  $N$  сигналов, по  $N$  отсчётов каждый (**Рис. 5.13**). Компонентные сигналы декомпозиции в этом случае представлены ступенчатыми функциями — функциями, у которых все  $K$  первых значений равны нулю, а все последующие — единице. Рассмотрим декомпозицию сигнала  $x[n]$ , состоящего из  $N$  дискретных отсчётов, с использованием компонентных сигналов  $x_0[n], x_1[n], x_2[n], \dots, x_{N-1}[n]$ .  $k$ -й сигнал декомпозиции  $x_k[n]$  будет иметь все отсчёты с 0-го по  $(k-1)$ -й равными нулю, а все последующие отсчёты равными  $x[k] - x[k-1]$ . Например, 5-й компонентный сигнал декомпозиции  $x_5[k]$  будет содержать отсчёты с 0-го по 4-й, равные нулю, и остальные отсчёты, равные  $x[5] - x[4]$  (разность между четвёртым и пятым отсчётом исходного сигнала). Сигнал  $x_0[k]$  является исключением. Все его отсчёты имеют значение  $x[0]$ . Если импульсная декомпозиция позволяет выделять отдельные отсчёты сигнала, то ступенчатая декомпозиция характеризует различие между значениями соседних отсчётов сигнала. С помощью ступенчатой декомпозиции вводится ещё одна характеристика системы — реакция на ступенчатое входное воздействие, или переходная характеристика.

## 5.7.3. Декомпозиция на основе сигналов с чётной и нечётной симметрией

*Декомпозиция на основе сигналов с чётной и нечётной симметрией* (**Рис. 5.14**) использует для разложения два компонента: сигнал с чётной симметрией и сигнал с нечётной симметрией. Об  $N$ -точечном сигнале говорят, что он обладает чётной симметрией, если относительно вертикальной оси, проведённой через точку  $N/2$ , его правая часть является зеркальным отражением его левой части. То есть значение  $x[N/2 + 1]$  равно значению  $x[N/2 - 1]$ ; значение  $x[N/2 + 2]$  равно значению  $x[N/2 - 2]$ , и т. д. Сигнал является сигналом с нечётной симметрией, если относительно оси  $N/2$  значения его отсчётов в правой части равны значениям отсчётов в левой части по модулю и противоположны по знаку. То есть значение  $x[N/2 + 1]$  равно значению  $-x[N/2 - 1]$ ; значение  $x[N/2 + 2]$  равно значению  $x[N/2 - 2]$ , и т. д. Такое определение предполагает, что число  $N$  — чётное и что отсчёты сигнала нумеруются с 0-го по  $(N-1)$ -й. Для декомпозиции сигнала используется следующее выражение:

$$\begin{aligned} x_E[n] &= \frac{x[N] + x[N-n]}{2}, \\ x_0[n] &= \frac{x[N] - x[N-n]}{2}. \end{aligned} \tag{5.1}$$

Формула декомпозиции на основе сигналов с чётной и нечётной симметрией. Исходный сигнал представляется в виде совокупности двух компонентных сигналов — сигнала с чётной симметрией  $x_E[n]$  и сигнала с нечётной симметрией  $x_0[n]$ . Основой декомпозиции служит круговая симметрия, поэтому нулевые отсчёты компонентных сигналов рассчитываются по формулам  $x_E[0] = x[0]$  и  $x_0[0] = 0$ . Исходный сигнал и оба компонента декомпозиции содержат по  $N$  отсчётов с 0-го по  $(N-1)$ -й.

Такое определение симметричности левой и правой сторон может показаться странным, поскольку действительным центром является точка  $N/2 - 1/2$ , попадающая между двумя соседними отсчётами сигнала, а не точка  $N/2$ . При использовании такой симметрии нулевой отсчёт выделяется из совокупности остальных отсчётов и определяется по-иному. Однако в чём суть использования этих сигналов?

Декомпозиция на основе указанных сигналов тесно связана с имеющим большое значение в теории ЦОС понятием *круговой симметрии*. Круговая симметрия рассматривает сигналы конечной длины как круговые массивы отсчётов: конец массива соединён с его началом. Так же как следующим для отсчёта  $x[4]$  является отсчёт  $x[5]$ , для отсчёта  $x[N - 1]$  следующим будет отсчёт  $x[0]$ . Вообразите себе змею, кусающую собственный хвост. Если рассматривать сигналы с чётной и нечётной симметрией как круговые сигналы, то можно выделить две оси симметрии: в точке  $x[N/2]$  и в точке  $x[0]$ . Так, например, для сигнала с чётной симметрией отсчёт  $x[1]$  будет равен отсчёту  $x[N - 1]$ , отсчёт  $x[2]$  — отсчёту  $x[N - 2]$  и т. д. В сигнале с нечётной симметрией значения отсчётов  $x[0]$  и  $x[N/2]$  всегда нулевые, а в сигнале с чётной симметрией они равны соответствующим значениям отсчётов исходного сигнала.

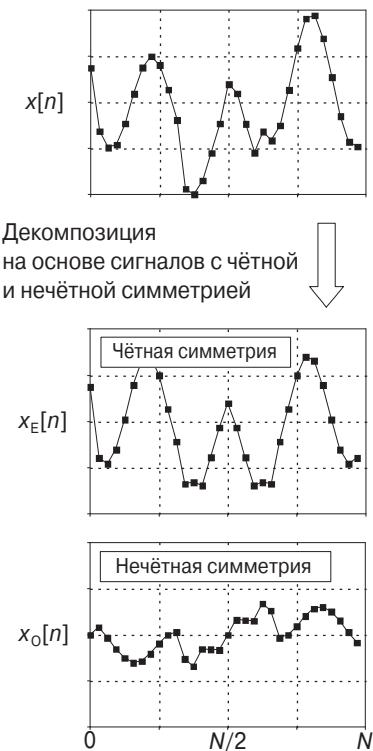
Однако зачем нам рассматривать начальные данные в массиве сигнала как продолжение его конечных данных? Обычные способы приёма сигналов не дают никаких оснований использовать круговые массивы. Напротив, первый и последний отсчёты сигнала имеют наименьшую взаимосвязь по сравнению с любыми другими двумя отсчётами. Это же очевидно! Разгадка кроется в особенностях применения известного в ЦОС метода гармонического анализа, или анализа Фурье. Рассмотрение сигналов в круговой форме присуще Фурье-анализу и является для него математически обоснованным. Физическая природа сигнала при этом, т. е. то, каким образом сигнал получен, может не иметь к этому никакого отношения. Более детально этот вопрос будет обсуждаться в Главе 10. В данном параграфе для нас имеет значение лишь декомпозиция сигнала с использованием выражения (5.1). Доказать правомочность декомпозиции на основе сигналов с чётной и нечётной симметрией очень просто — их суммирование всегда даёт восстановление исходного сигнала.

#### 5.7.4. Декомпозиция с прореживанием

*Декомпозиция с прореживанием* (Рис. 5.15) представляет сигнал в виде совокупности двух компонентов — сигнала с чётными отсчётами и сигнала с нечётными отсчётами (не надо путать с сигналами с чётной и нечётной симметрией). Чтобы получить сигнал с чётными отсчётами, следует взять исходный сигнал и приравнять нулю все его нечётные отсчёты. Для получения сигнала с нечётными отсчётами чётные отсчёты исходного сигнала приравниваются нулю. Довольно просто.

На первый взгляд такое разложение может показаться очевидным и не представляющим интереса. Однако, как бы странно это ни казалось, именно декомпозиция с прореживанием лежит в основе одного из основополагающих алгоритмов цифровой обработки сигналов — алгоритма *быстрого преобразования Фурье* (БПФ). Ниже будет рассмотрена *декомпозиция Фурье*, известная уже несколько сотен лет. Она, однако, требует большого количества времени выполнения на обычных компьютерах. Алгоритмы БПФ, которых существует достаточно большой на-

бор, появились ещё в 1960-х годах и дали возможность значительно сократить вычислительные затраты. Идея, лежащая в их основе, характерна для ЦОС: сигнал за счёт последовательного использования декомпозиций с прореживанием представляется в виде совокупности простых компонентов; рассчитывается декомпозиция Фурье для этих простых сигналов; на базе полученных результатов осуществляется синтез выходного сигнала. Результат такого подхода потрясающий! Время вычислений сокращается в сотни и тысячи раз.



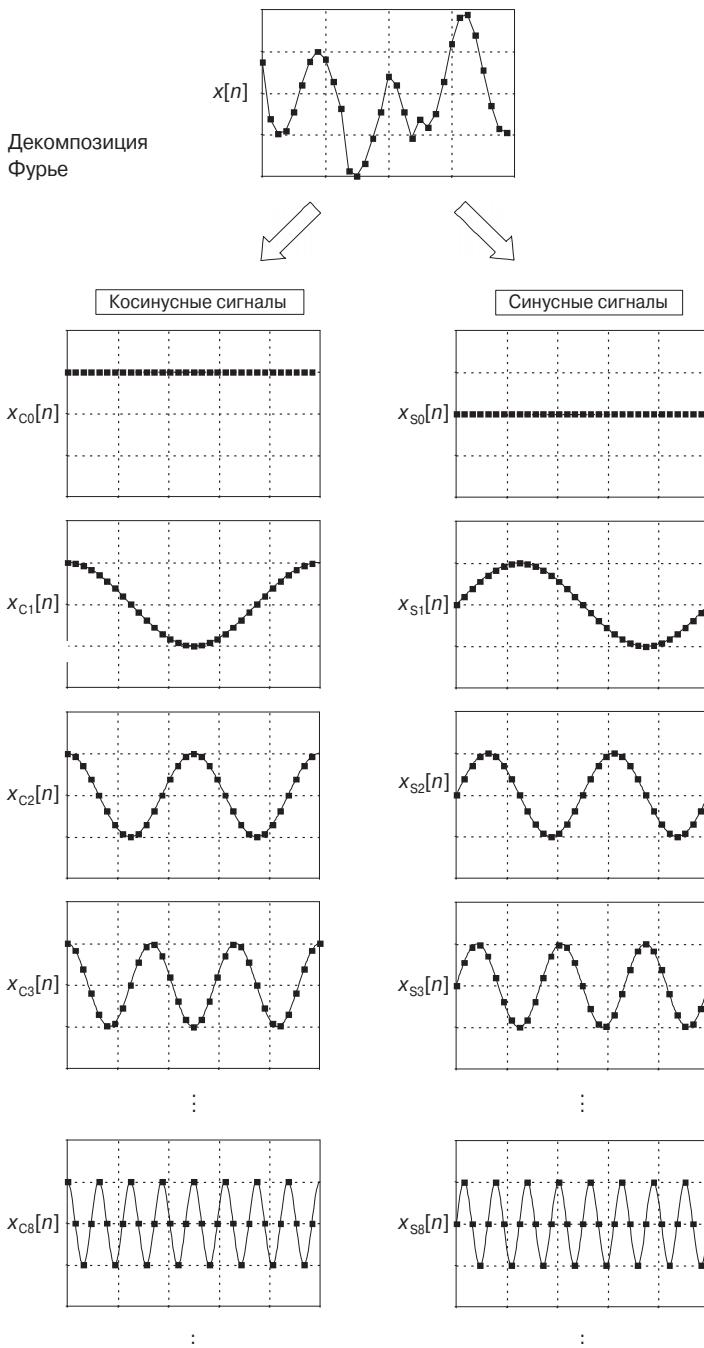
**Рис. 5.14.** Пример декомпозиции на основе сигналов с чётной и нечётной симметрией. Исходный сигнал, состоящий из  $N$  дискретных отсчётов, разбивается на два сигнала, по  $N$  отсчётов каждый, один из которых является сигналом с чётной симметрией, а другой — сигналом с нечётной симметрией.



**Рис. 5.15.** Пример декомпозиции с прореживанием. Исходный сигнал, состоящий из  $N$  дискретных отсчётов, разбивается на два сигнала, по  $N$  отсчётов каждый, в одном из которых все нечётные отсчёты приравнены нулю, а в другом все чётные отсчёты приравнены нулю.

## 5.7.5. Декомпозиция Фурье

**Декомпозиция Фурье** носит достаточно глубокий математический характер и неочевидна для восприятия. **Рис. 5.16** иллюстрирует данную процедуру. Любой сигнал, включающий  $N$  дискретных отсчётов, может быть представлен в виде совокупности  $(N + 2)$ -х сигналов, половина из которых синусоиды, а вторая половина — косинусоиды. Косинусный сигнал с наименьшей частотой (обозначенный на рисунке  $x_{C0}[n]$ ) представляет собой постоянный сигнал, включающий  $N$  отсчётов. Последующие косинусные компоненты разложения  $x_{C1}[n]$ ,



**Рис. 5.16.** Пример декомпозиции Фурье. Исходный сигнал, состоящий из  $N$  дискретных отсчётов, разбивается на  $(N + 2)$  сигнала, по  $N$  отсчётов каждый. Одна половина этих сигналов является косинусоидами, другая — синусоидами. Гармонические сигналы, лежащие в основе декомпозиции, имеют фиксированные частоты. Для разных исходных сигналов они отличаются только соотношениями амплитуд.

$x_{C2}[n]$  и  $x_{C3}[n]$  на длительности  $N$  периодов дискретизации имеют соответственно 1, 2 и 3 периода повторения. Аналогично можно предложить остальные косинусные, а также синусные компоненты разложения. Частоты базовых компонентов декомпозиции фиксированы для любых исходных сигналов. Изменяться могут только амплитуды синусоид и косинусоид, входящих в состав разложения.

Декомпозиция Фурье играет большую роль в цифровой обработке сигналов, благодаря следующим трём факторам. Во-первых, многие сигналы, встречающиеся в практике ЦОС, по своей природе образованы суммой набора гармонических сигналов. Хорошим примером здесь служат аудиосигналы. Использование декомпозиции Фурье очевидно для анализа структуры таких сигналов. Во-вторых, гармонические сигналы играют особую роль для анализа линейных систем, так как не подвергаются искажению формы, а лишь изменяют свою амплитуду и начальную фазу. Если известно, как гармонические сигналы проходят через линейную систему, то декомпозиция Фурье позволяет рассчитать, как через эту систему пройдет произвольный сигнал. И в-третьих, декомпозиция Фурье лежит в основе мощного и широко применяемого аппарата Фурье-анализа, а также фундаментальных для ЦОС преобразования Лапласа и Z-преобразования. Большинство современных алгоритмов цифровой обработки сигналов в той или иной степени используют этот аппарат.

Почему произвольный сигнал может быть представлен набором синусных и косинусных колебаний? Как определить амплитуды гармонических сигналов декомпозиции для конкретных входных сигналов? Какие типы систем могут строиться на базе этой методики? На все эти вопросы мы ответим в последующих главах. Декомпозиция Фурье — слишком сложное понятие, чтобы рассмотреть его в кратком обзоре, представленном в этой главе. Сейчас для нас важен лишь факт, что если все гармонические сигналы декомпозиции Фурье сложить, то исходный сигнал будет полностью восстановлен. Более детальное описание может быть найдено в Главе 8.

## 5.8. Если система нелинейна

Чтобы оценить значение линейных систем, отметим, что единственный способ анализа нелинейных систем сводится к приведению их в той или иной степени к линейному виду. При этом наиболее распространены три способа:

1. Нелинейность игнорируется. Этот способ хорош, если нелинейность проявляется слабо и ею можно пренебречь. Возникающие при таком допущении ошибки рассматриваются как шум или вовсе игнорируются.

2. Используются сигналы малых уровней. Многие нелинейные системы ведут себя как линейные, если входные сигналы имеют малую амплитуду. Например, транзистор — это нелинейное устройство, если рассматривать его характеристики во всём рабочем диапазоне. Однако он обеспечивает линейное усиление сигналов, уровень которых не превышает нескольких милливольт. Этот эффект используется, например, в операционных усилителях. Маленькие входные сигналы обуславливают абсолютно линейное поведение системы, которая в общем случае может оказаться явно нелинейной.

3. Применяется линеаризация. Рассмотрим, например, сигнал, генерируемый как произведение двух других сигналов:  $a[n] = b[n] \times c[n]$ . Если перейти к логарифмам сигналов, то вместо произведения мы получим сумму:  $\log(a[n]) = \log(b[n]) + \log(c[n])$ . Для этого подхода используется причудливое название *гомоморфная обработка сигналов*. Она широко применяется, например, при обработке изображений. Представьте, что моделью изображения может являться произведение двумерного сигнала, идущего от источника освещения, и двумерной матрицы коэффициентов переотражения освещаемой сцены. Гомоморфная обработка сигналов позволяет привести «освещдающий» сигнал к известному виду, позволяя тем самым применять общераспространённые методы оптимальной фильтрации изображений.

В следующих главах мы рассмотрим два основных метода обработки сигналов: свёртку и преобразование Фурье. И тот, и другой методы базируются на изложенных в данной главе принципах: декомпозиции сигнала в ряд простых компонентов, обработке этих простых сигналов независимо друг от друга и синтезе общего результата. На этом построена вся классическая ЦОС.

# Глава 6

---

## СВЁРТКА

*Свёртка* — это математическая процедура, которая из двух исходных сигналов формирует по определённому правилу третий сигнал. В цифровой обработке она играет важнейшую роль. Дело в том, что цифровые системы принято описывать их импульсными характеристиками, что оказывается возможным благодаря использованию *импульсной декомпозиции*. Свёртка устанавливает взаимосвязь сигнала на входе и на выходе системы через её импульсную характеристику. В этой главе процедура свёртки рассматривается с двух позиций: со стороны входа системы и со стороны её выхода. Свёртка лежит в основе всей цифровой обработки сигналов. Пожалуй, нет ничего более важного.

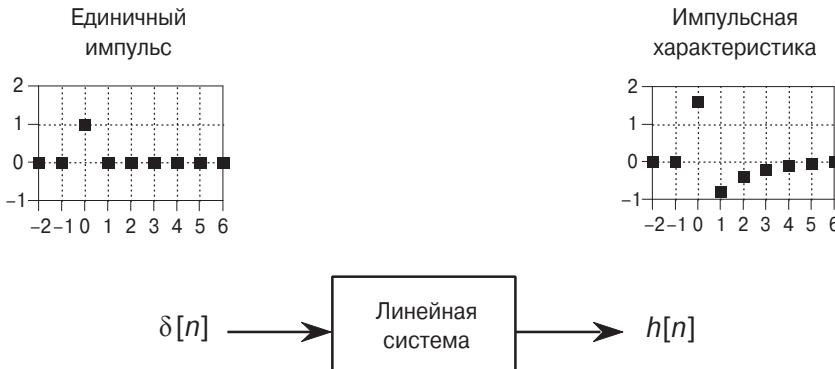
### 6.1. Дельта-функция и импульсная характеристика

В предыдущей главе мы рассмотрели *декомпозицию* сигналов, использующую в качестве компонентов разложения совокупность импульсных сигналов с единственным ненулевым отсчётом. Там же был дан очень важный в теории ЦОС подход к системам обработки: входной сигнал представляется совокупностью простых компонентов — сигналов разложения (декомпозиция). Прохождение компонентов через *линейную систему* рассматривается независимо для каждого сигнала декомпозиции, а реакции системы на их воздействия суммируются на выходе (синтез). Результирующий выходной сигнал оказывается полностью идентичным сигналу, который оказался бы на выходе системы при подаче на её вход исходного сигнала без декомпозиции.

Может быть найдено достаточно большое количество различных вариантов декомпозиции. Однако в ЦОС наибольшее значение имеют два из них: *импульсная декомпозиция* и *декомпозиция Фурье*. При использовании первой математическое описание работы системы осуществляется с помощью операции *свёртки*. В этой и в большинстве последующих глав мы будем иметь дело только с *дискретными сигналами*. Свёртка, однако, не ограничена только дискретными сигналами. В Главе 13 мы рассмотрим свёртку непрерывных сигналов.

Огромное значение для ЦОС имеют следующие два понятия. Первое из них — это дискретная дельта-функция, обычно обозначаемая символом  $\delta[n]$  (Рис. 6.1). Для дискретных систем она называется *единичным импульсом*, или *единичным отсчётом*, и представляет собой сигнал, в котором нулевой отсчёт имеет значение единицы, а все остальные отсчёты равны нулю.

Второе понятие — это *импульсная характеристика* системы (Рис. 6.1). Как можно догадаться по самому названию, импульсная характеристика — это реак-



**Рис. 6.1.** Иллюстрация понятий единичного импульса и импульсной характеристики. Единичный импульс — это дискретный сигнал, все отсчёты которого, кроме нулевого, равны нулю. Значение нулевого отсчёта равно единице. Единичный импульс является дискретным аналогом дельта-функции, которую принято обозначать символом  $\delta(t)$ . Импульсную характеристику линейной системы — это реакция системы на единичный импульс. Импульсную характеристику обычно обозначают  $h[n]$ .

ция системы на единичный импульс. Если две любые системы имеют хотя бы какие-то различия, их импульсные характеристики будут обязательно отличаться. Если сигналы на входе и выходе системы традиционно обозначают  $x[n]$  и  $y[n]$  соответственно, то импульсную характеристику системы принято обозначать  $h[n]$ . Конечно, это не является обязательным, и можно использовать другие обозначения, если они кажутся вам более удобными: например, импульсную характеристику фильтра можно обозначить  $f[n]$ .

Любой импульсный сигнал, являющийся компонентом разложения при импульсной декомпозиции, может быть представлен как единичный импульс, сдвинутый на соответствующую величину по временной оси и умноженный на значение отсчёта исходного сигнала в этот же момент времени. Для примера рассмотрим сигнал  $a[n]$ , все значения которого равны нулю, кроме 8-го отсчёта, принимающего значение -3. Этот сигнал может рассматриваться как единичный импульс, сдвинутый (задержанный) на 8 периодов дискретизации и умноженный на -3, или в форме математического выражения:  $a[n] = -3\delta[n - 8]$ . Советуем вам запомнить эту запись, поскольку она используется практически во всех выражениях, употребляемых в ЦОС.

Допустим, на вход системы подан сигнал  $a[n] = -3\delta[n - 8]$ . Каким окажется её выход? Чтобы ответить на этот вопрос, придётся вспомнить ранее рассмотренные свойства однородности и инвариантности к сдвигу. Умножение и задержка сигнала на входе приводят к аналогичным преобразованиям выходного сигнала. Таким образом, если входному воздействию  $\delta[n]$  соответствует реакция на выходе  $h[n]$  (импульсная характеристика системы), то сигналу  $-3\delta[n - 8]$  на входе системы будет соответствовать выходной сигнал  $-3h[n - 8]$ . Другими словами, сигналом на выходе окажется импульсная характеристика системы, сдвинутая во времени и усиленная по амплитуде, так же, как сдвинут и усилен единичный импульс на выходе. Итак, если вам известна импульсная характеристика системы, то вычислить её реакцию на произвольный импульсный сигнал с единственным ненулевым отсчётом не составляет труда.

## 6.2. Свёртка

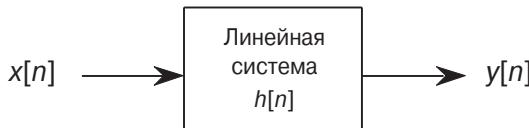
Нам осталось подвести итог материалу, посвящённому определению реакции на выходе системы при произвольных входных воздействиях. Во-первых, входной сигнал путём декомпозиции представляется в виде совокупности импульсных сигналов с единственным ненулевым отсчётом, каждый из которых может рассматриваться как *единичный импульс*, умноженный на некоторую величину и сдвинутый во времени. Во-вторых, реакция системы на каждый из входных импульсных сигналов может быть получена путём соответствующего усиления и сдвига *импульсной характеристики* системы. И наконец, в-третьих, реакция системы на исходный входной сигнал рассчитывается суммированием реакций системы на все компоненты декомпозиции в отдельности. Таким образом, если нам известна импульсная характеристика системы, то мы можем определить её реакцию на любое входное воздействие. Фактически это означает, что нам известно всё о системе. При описании *линейной системы* импульсная характеристика является исчерпывающей (заметим, однако, что в последующих главах мы рассмотрим и другие характеристики систем, которые не несут новой информации, а просто представляют её в другой форме).

Импульсная характеристика может обозначаться и другими терминами в зависимости от сферы применения. Например, при описании фильтров возможно использование понятия *ядра фильтра*, *ядра свёртки* или просто *ядра*. В приложениях цифровой обработки изображений вместо импульсной характеристики используется термин *функция рассеяния точки*. Несмотря на небольшие различия в использовании указанных терминов, все они обозначают одно и то же: реакцию системы на воздействие в виде дельта-функции или единичного импульса.

Операция свёртки является обычной математической операцией, такой, как, например, умножение, сложение или интегрирование. Аналогично любой другой операции при свёртке двух сигналов (дискретных последовательностей) получают единственный третий сигнал. Процедура свёртки применяется в различных разделах математики, в частности в теории вероятностей и математической статистике. Теория линейных систем использует свёртку для установления взаимосвязи между входом системы, её импульсной характеристикой и выходным сигналом.

**Рис. 6.2** иллюстрирует понятие свёртки применительно к линейным системам. Сигнал  $x[n]$  поступает на вход линейной системы, описываемой импульсной характеристикой  $h[n]$ , и вызывает на выходе системы реакцию  $y[n]$ . В форме уравнения это записывается так:  $x[n] * h[n] = y[n]$ . То есть сигнал на выходе линейной системы равен свёртке сигнала на входе системы и её импульсной характеристики. Подобно тому как сложение обозначается плюсом (+), а умножение — крестиком ( $\times$ ), свёртка обозначается звёздочкой (\*). К сожалению, в большинстве языков программирования символ \* принято использовать для обозначения умножения. Таким образом, звёздочка в тексте программы означает умножение, а та же звёздочка в математических выражениях означает свёртку.

На **Рис. 6.3** проиллюстрировано использование свёртки для реализации низкочастотной и высокочастотной фильтрации. В этом примере входным сигналом является сумма медленно возрастающего сигнала (спектр лежит в области низких частот) и синусоиды, у которой на интервале наблюдения укладывается несколь-

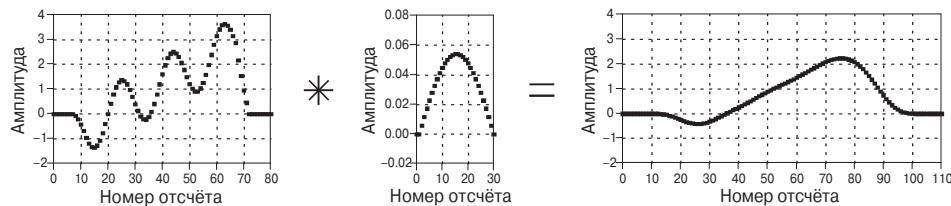


$$x[n] * h[n] = y[n]$$

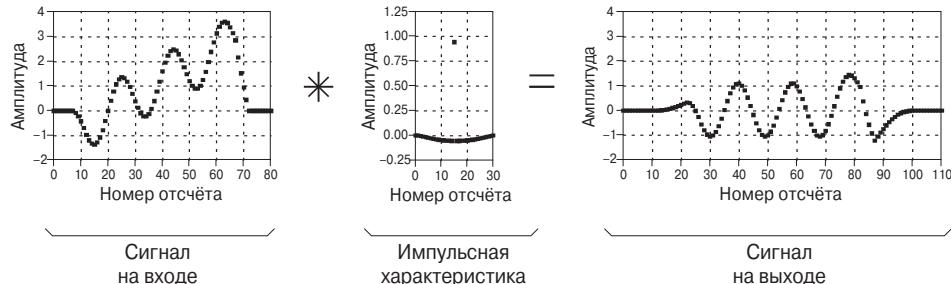
**Рис. 6.2.** Использование операции свёртки в цифровой обработке сигналов. Сигнал на выходе линейной системы равен свёртке входного сигнала и импульсной характеристики системы. В математических выражениях операция свёртки обозначается звёздочкой (\*).

ко периодов повторения (спектр относительно высокочастотный). В случае низкочастотного фильтра (**а**) импульсная характеристика в форме дуги позволяет проходить на выход системы только медленно меняющейся составляющей. В случае высокочастотного фильтра (**б**), напротив, импульсная характеристика системы такова, что на выходе системы присутствует только относительно быстро меняющийся компонент входного сигнала.

**a)** Фильтр нижних частот

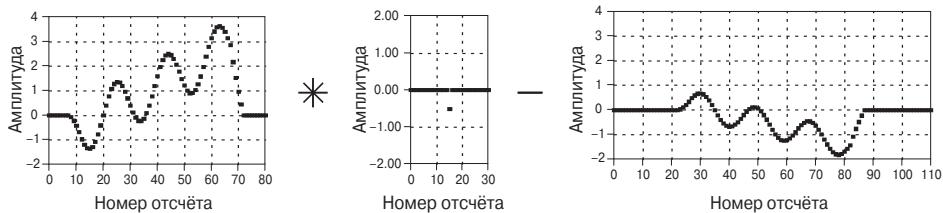
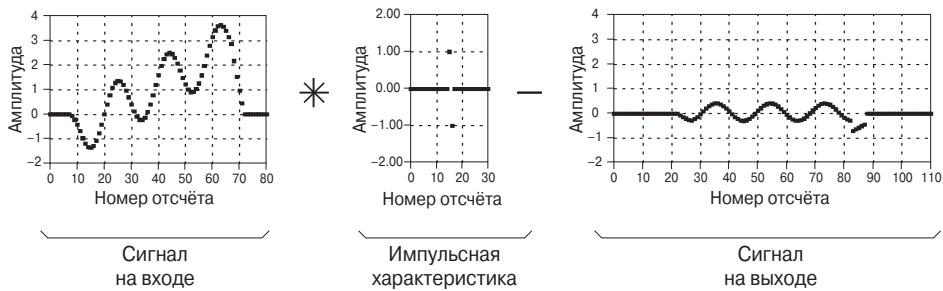


б) Фильтр верхних частот



**Рис. 6.3.** Пример использования свёртки для описания систем низкочастотной и высокочастотной фильтрации. В приведённом примере на вход системы поступает сумма медленно возрастающего сигнала синусоиды, у которой на интервале наблюдения укладывается несколько периодов повторения. Эти два компонента входного сигнала удается разделить за счёт использования выбранных соответствующим образом импульсных характеристик.

**Рис. 6.4** демонстрирует другие два примера использования свёртки для описания процесса обработки сигналов. Инвертирующий аттенюатор (**а**) «переворачивает» сигнал «вверх ногами» и уменьшает его амплитуду. Взятие дискретной производной (вычисление первой разности) даёт на выходе системы сигнал, пропорциональный скорости изменения входного сигнала (**б**).

**а) Инвертирующий аттенюатор****б) Дискретная производная**

**Рис. 6.4.** Использование свёртки для описания процесса обработки сигналов. Большинство систем ЦОС имеют достаточно простые импульсные характеристики. Приведённые примеры показывают, что даже система с импульсной характеристикой, имеющей лишь несколько ненулевых отсчётов, может осуществлять значительные преобразования сигналов.

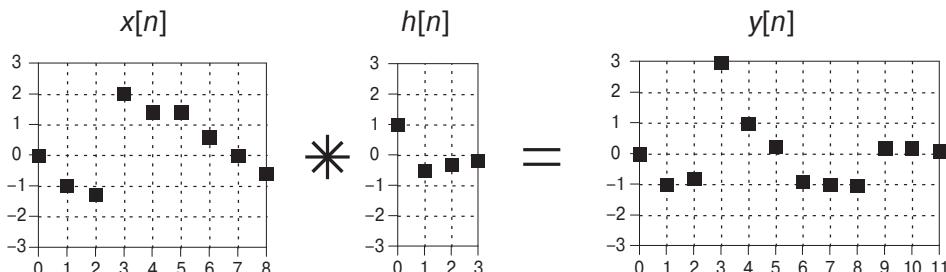
Обратите внимание на длины сигналов, приведённых на Рис. 6.3 и 6.4. Входные сигналы включают 81 дискретный отсчёт, в то время как импульсные характеристики состоят только из 31 отсчёта. Дело в том, что в большинстве приложений цифровой обработки входные сигналы рассматриваются на интервалах в сотни, тысячи и даже миллионы отсчётов. Импульсная же характеристика значительно короче, скажем, в сотни раз. Математическое определение операции свёртки не накладывает никаких ограничений на длительность этих сигналов. Оно, однако, позволяет определить длину сигнала на выходе, равную сумме длины входного сигнала и длины импульсной характеристики минус один. Так для примеров, показанных на Рис. 6.3 и 6.4, длительность выходных сигналов составляет  $81 + 31 - 1 = 111$ . Отсчёты входных сигналов нумеруются с 0-го по 80-й, отсчёты импульсных характеристик — с 0-го по 30-й, а отсчёты выходных сигналов — с 0-го по 110-й.

Приступим теперь к подробному рассмотрению математического описания свёртки. В том качестве, в котором свёртка используется в ЦОС, её следует рассматривать с двух позиций: по отношению к входному сигналу и по отношению к сигналу на выходе. Представление операции *свёртки со стороны входа* системы подразумевает анализ того, как каждый отсчёт входного сигнала влияет на общую реакцию системы на её выходе. Представление операции *свёртки со стороны выхода*, напротив, означает понимание того, как каждый отсчёт выходного сигнала зависит от совокупности входных отсчётов.

Следует помнить, что оба указанных способа анализа в действительности являются двумя сторонами одного и того же процесса — математической операции свёртки. Представление свёртки со стороны входного сигнала позволяет понять огромную роль этой процедуры в теории ЦОС. В то же время представление свёртки со стороны выходного сигнала даёт детальное понимание её математической сути. В этом проявляется одна из типичных для ЦОС проблем: интуитивно понятные принципы цифровой обработки должны быть описаны сложным математическим языком, чтобы ваши идеи могли стать доступны другим.

## 6.3. Описание свёртки со стороны входа системы

На Рис. 6.5 показан простой пример реализации свёртки: входной сигнал  $x[n]$ , состоящий из 9 дискретных отсчётов, проходит через линейную систему, имеющую импульсную характеристику  $h[n]$  длиной 4 дискретных отсчёта, и вызывает появление на выходе системы сигнала  $y[n]$ , длина которого составляет  $9 + 4 - 1 = 12$  отсчётов. На языке математики это означает, что сигнал  $x[n]$  сворачивается с  $h[n]$  и вызывает реакцию системы  $y[n]$ . Представление свёртки со стороны входа использует базовую для ЦОС концепцию: входной сигнал подвергается декомпозиции, простые компоненты декомпозиции проходят через систему независимо и на выходе системы процедура синтеза восстанавливает общую реакцию системы на исходный входной сигнал. В рассматриваемом примере каждый из 9 входных отсчётов даёт на выходе системы сдвинутую по временной оси и умноженную на постоянный коэффициент копию импульсной характеристики системы (Рис. 6.6). Выходной сигнал  $y[n]$  образуется суммированием этих 9 сигналов.

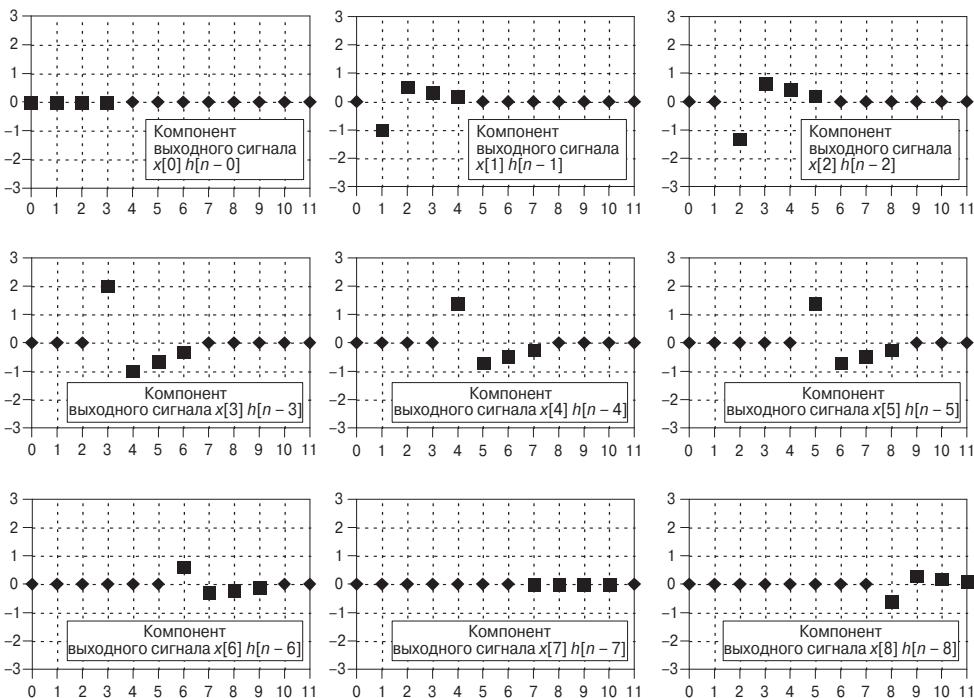


**Рис. 6.5.** Пример реализации свёртки. 9-точечный входной сигнал сворачивается с импульсной характеристикой системы длиной в 4 отсчёта и вызывает появление на выходе системы сигнала  $y[n]$  длительностью 12 отсчётов. Каждый отсчёт входного сигнала вызывает на выходе системы реакцию, рассчитываемую сдвигом и умножением на постоянную величину импульсной характеристики системы. Девять реакций системы на девять отсчётов входного сигнала показаны на девяти графиках на Рис. 6.6.

Рассмотрим более подробно некоторые из сигналов на выходе системы, являющиеся реакциями на отдельные отсчёты входного сигнала. Начнём с четвёртого отсчёта сигнала на входе системы, т. е. с  $x[4]$ . Значение этого отсчёта составляет 1.4. Базовый сигнал декомпозиции для  $x[4]$  описывается выражением  $1.4\delta[n - 4]$ .

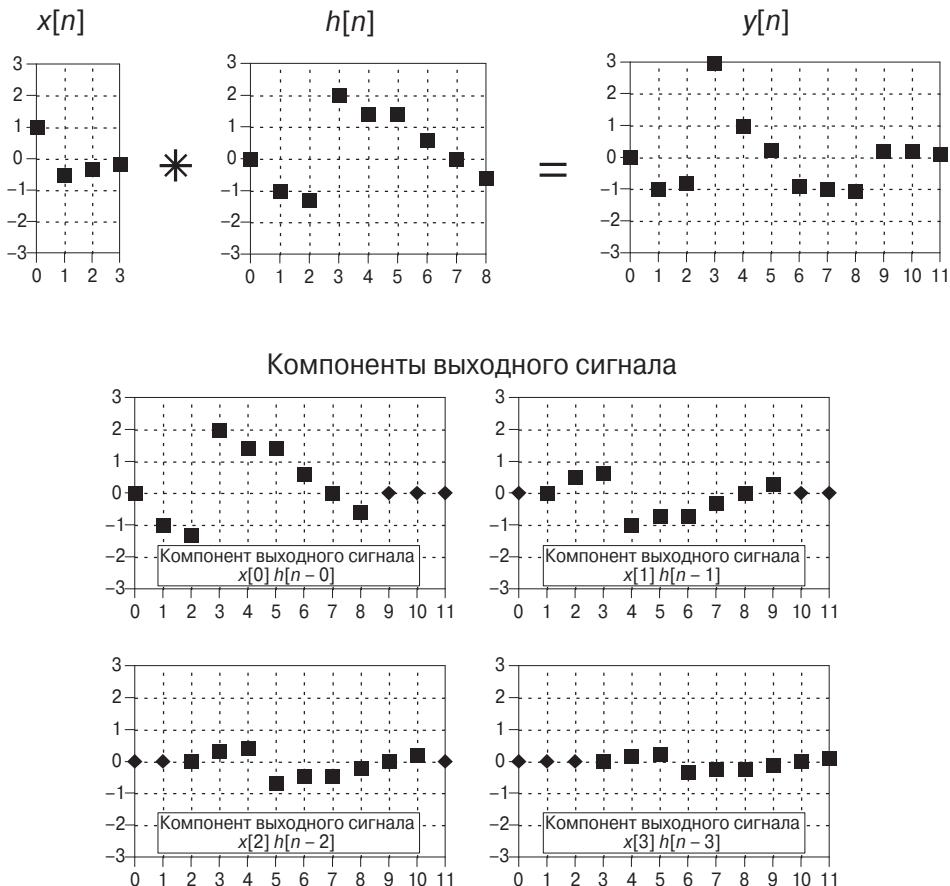
А реакция системы на такой сигнал:  $1.4h[n - 4]$ . На Рис. 6.6 этот сигнал показан на графике, расположенном по центру. Видно, что выходной сигнал представляет собой импульсную характеристику системы, умноженную на 1.4 и сдвинутую вправо на 4 отсчёта. Отсчёты 0...3 и 8...11 заполняются нулями, делая длину выходного сигнала равной рассчитанным 12 отсчётам. Для большей наглядности отсчёты, полученные на основе отсчётов импульсной характеристики, обозначены на графике квадратами, а дополнительные нулевые отсчёты — ромбами.

Посмотрим теперь на последний, 8-й отсчёт входного сигнала  $x[8]$ . Его значение равно  $-0.5$ . Реакция системы на соответствующий ему компонент декомпозиции, показанная на правом нижнем графике Рис. 6.6, представляет собой импульсную характеристику системы, умноженную на  $-0.5$  и сдвинутую вправо на 8 отсчётов. Нулями заполнены отсчёты с 0-го по 7-й. И наконец, рассмотрим входные отсчёты  $x[0]$  и  $x[7]$ . Их значения равны нулю. Поэтому реакция системы на них представляет собой полностью нулевой сигнал.



**Рис. 6.6.** Компоненты выходного сигнала, получаемые при реализации свёртки, показанной на Рис. 6.5. Отсчёты сигнала, полученные путём сдвига и масштабирования импульсной характеристики системы, отмечены на графиках квадратами. Оставшиеся отсчёты, являющиеся дополнительными (равными нулю), обозначены ромбами.

В рассмотренном примере входной сигнал  $x[n]$  имел длину 9 отсчётов, а импульсная характеристика  $h[n]$  — 4 отсчёта. В следующем примере, проиллюстрированном на Рис. 6.7, ситуация противоположна: входной сигнал ограничен 4 отсчётами, а импульсная характеристика системы включает 9 отсчётов. Формы сигналов сохранены прежними. Входной сигнал и импульсная характеристика



**Рис. 6.7.** Второй пример реализации свёртки. По сравнению с первым примером (Рис. 6.5) импульсная характеристика и входной сигнал «поменялись местами». Благодаря свойству коммутативности, характерному для свёртки, выходные сигналы в обоих примерах идентичны.

просто «поменялись местами». По графикам компонентов выходного сигнала видно, что четыре отсчёта входного сигнала, как и в предыдущем примере, вызвали реакцию системы в виде сдвинутой и промасштабированной 9-точечной импульсной характеристики, дополненной слева и справа нулями.

Посмотрим повнимательнее. Сигналы, изображённые на Рис. 6.7 и 6.5, оказались идентичными! Это не случайность, а одно из важных свойств свёртки — свойство коммутативности:  $a[n] * b[n] = b[n] * a[n]$ . С математической точки зрения не имеет значения, какой из сигналов является входным, а какой представляет импульсную характеристику. Имеет место просто свёртка двух сигналов. Однако с точки зрения реальных физических процессов такая перестановка сигналов не имеет смысла в теории систем. Входной сигнал и импульсная характеристика — это совершенно разные понятия. Поэтому указанное математическое свойство свёртки полезно исключительно для использования в математических преобразованиях.

**Программа 6.1** реализует свёртку так, как она представляется со стороны входа системы. Не забывайте, что программы в этой книге используются просто для на-

иболее наглядной иллюстрации алгоритмов и могут не отличаться хорошим стилем программирования. Все процедуры, связанные с вводом и выводом данных, реализуются в воображаемых подпрограммах (строки 160 и 280). Для нас не важно, как конкретно реализуются эти действия, и мы не станем загромождать ими программу. Рекомендуем детально ознакомиться с текстом программы: она отражает ключевые моменты рассматриваемой темы.

### Программа 6.1

```

100          'СВЁРТКА, ПРЕДСТАВЛЕННАЯ СО СТОРОНЫ ВХОДА
110
120 DIM X[80]          'Сигнал на входе, 81 отсчёт
130 DIM H[30]           'Импульсная характеристика, 31 отсчёт
140 DIM Y[110]          'Сигнал на выходе, 111 отсчётов
150
160 GOSUB XXXX          'Вызов подпрограммы загрузки X[ ] и H[ ]
170
180 FOR I% = 0 TO 110   'Обнуление массива выходного сигнала
190 Y(I%) = 0
200 NEXT I%
210
220 FOR I% = 0 TO 80    'Цикл по всем элементам массива X[ ]
230 FOR J% = 0 TO 30    'Цикл по всем элементам массива H[ ]
240 Y[I%+J%] = Y[I%+J%] + X[I%*H[J%]]  '(не забудьте, в программе «*» - это знак
250 NEXT J%              'умножения!)
260 NEXT I%
270
280 GOSUB XXXX          'Вызов подпрограммы вывода результата
290                         'из массива Y[ ]
300 END

```

В программе реализуется свёртка входного сигнала, включающего 81 дискретный отсчёт и расположенного в массиве  $X[ ]$ , с импульсной характеристикой, состоящей из 31 отсчёта и расположенной в массиве  $H[ ]$ . Результатом обработки является массив  $Y[ ]$ , содержащий 111 дискретных отсчётов. Длины сигналов взяты аналогичными примерам, проиллюстрированным на Рис. 6.3 и 6.4. Отметим, что для обозначения массивов используются заглавные буквы, что является нарушением правил обозначения, о которых мы договаривались раньше. Напомним, что заглавные буквы были зарезервированы нами для обозначения сигналов в частотной области. Дело в том, что, к сожалению, язык Бейсик в своей базовой форме не разрешает использование строчных букв для обозначения переменных. Другое замечание будет касаться использования символа «\*» (строка 240). Дело в том, что в тексте программ этот символ означает умножение, в то время как в математических выражениях им обозначается свёртка. Если же символ «звёздочка» встречается в обычном тексте, например в документации или в комментариях к программе, то он может означать и то, и другое в зависимости от контекста.

Воображаемая подпрограмма, вызываемая в строке 160, загружает отсчёты входного сигнала в массив  $X[ ]$ , а отсчёты импульсной характеристики — в массив  $H[ ]$ . Строки 180...200 обнуляют элементы массива  $Y[ ]$ . Обнуление в данном случае является необходимым, так как массив  $Y[ ]$  используется как акку-

мулятор для суммирования всех компонентов выходного сигнала, получаемых в процессе вычислений. Строки 220...260 являются «сердцем» программы. Команда FOR в строке 220 организует цикл поэлементной обработки массива  $X[ ]$ . Внутри него находится другой, вложенный цикл (строки 230...250), который для каждого отсчёта входного сигнала рассчитывает выходной компонент свёртки путём смещения и масштабирования импульсной характеристики системы. Получаемый результат добавляется к текущему значению массива  $Y[ ]$ . Структура программы с вложенным циклом (внутри одного цикла находится другой цикл) является ключевой для реализации свёртки. Обратите на неё внимание.

Анализ строки 240 в непосредственной её записи может свести с ума, поэтому попробуем разложить её на более простые части. Допустим, половина программы уже выполнена, и мы приступаем к обработке отсчёта  $X[40]$ , т. е. индекс  $I\% = 40$ . Выполнение команд внутреннего цикла для этого отсчёта будет подразумевать следующие три действия. Во-первых, все элементы импульсной характеристики умножаются на входной отсчёт. Если бы это действие было единственным, требующим выполнения в строке 240, она бы выглядела так:  $Y[J\%] = X[40] * H[J\%]$ . Во-вторых, полученный результат сдвигается вправо на 40 отсчётов за счёт прибавления этой величины к значению индекса элемента выходного массива:  $Y[40+J\%] = X[40*H[J\%]]$ . И в-третьих, результаты обработки отдельных входных отсчётов должны накапливаться в выходном массиве  $Y[ ]$  (процедура синтеза). То есть результат, полученный на текущем шаге, должен быть добавлен к содержимому выходного массива. В результате получаем команду  $Y[40+J\%] = Y[40+J\%] + X[40*H[J\%]]$ . Рекомендуем подробно изучить данный фрагмент программы. Он немного запутан, но принципиально важен.

## 6.4. Описание свёртки со стороны выхода системы

При представлении свёртки со стороны входа системы мы анализировали, каким образом каждый отсчёт входного сигнала формирует выходную реакцию системы, включающую много отсчётов. Описание свёртки со стороны выхода системы меняет ситуацию на противоположную. Требуется проанализировать, как каждый отсчёт на выходе системы получается из совокупности входных отсчётов. Такой анализ имеет большое значение и с точки зрения математического описания, и с позиции практики применения свёртки. Предположим, нам известны входной сигнал и импульсная характеристика системы, и мы хотим вычислить их свёртку. Наиболее логичным в данной ситуации будет написать программу, рассчитывающую выходной массив данных в цикле отсчёт за отсчётом. В этой программе, подобно записи уравнений в форме:  $y[n] = \text{‘некоторая комбинация переменных’}$ , отсчёт с индексом  $n$  выходного сигнала будет вычисляться как некоторая комбинация отсчётов входного сигнала и импульсной характеристики. Таким образом, необходимо найти взаимосвязь каждого отсчёта выходного сигнала в отдельности от входных отсчётов и импульсной характеристики. Эту взаимосвязь устанавливает описание свёртки со стороны выхода системы.

Рассмотрим на примере, как влияет набор входных отсчётов на один из отсчётов выходного сигнала. В качестве такого выходного отсчёта выберем значение

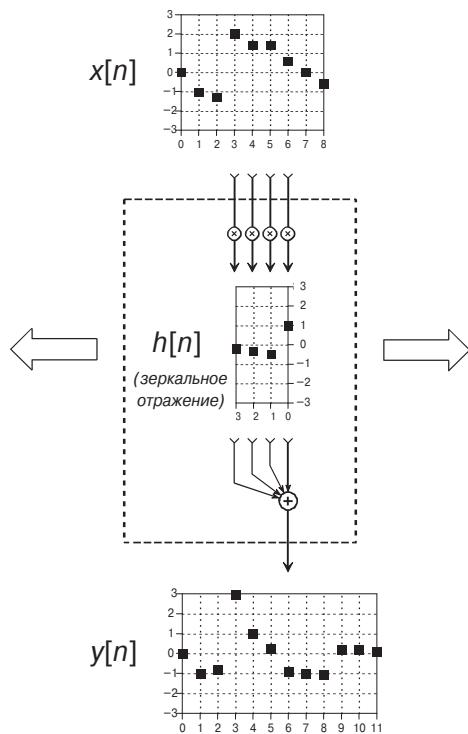
$y[6]$  сигнала, изображённого на Рис. 6.5. Значение этого отсчёта равно сумме всех отсчётов с индексом 6 девяти компонентов выходного сигнала, показанных на Рис. 6.6. Посмотрим теперь повнимательнее на эти девять выходных компонентов и определим, какие из них определяют значение  $y[6]$ . Это будут те сигналы, у которых отсчёт с индексом 6 имеет ненулевое значение. Итак, пять из девяти сигналов имеют отсчёт с индексом 6, равным 0 (это нулевые отсчёты, которыми выходной сигнал был дополнен до теоретически рассчитанной длины; на графиках они обозначены ромбами). Эти компоненты можно проигнорировать. Остальные четыре сигнала могут содержать ненулевой 6-й отсчёт. К этим сигналам относятся компоненты выходного сигнала, полученные при обработке входных отсчётов:  $x[3], x[4], x[5]$  и  $x[6]$ . Суммированием отсчётов компонентов выходного сигнала с указанным индексом значение  $y[6]$  определяется так:  $y[6] = x[3]h[3] + x[4]h[2] + x[5]h[1] + x[6]h[0]$ . То есть четыре отсчёта входного сигнала умножаются на четыре отсчёта импульсной характеристики, и произведения складываются.

Рис. 6.8 иллюстрирует механизм реализации свёртки со стороны выхода системы в наглядной форме. Представьте, что графики входного  $x[n]$  и выходного  $y[n]$  сигналов зафиксированы на своих местах на странице, а блок реализации свёртки (всё, что обведено пунктирной линией) может перемещаться влево-вправо. Фиксация блока свёртки в определённом положении означает вычисление того отсчёта выходного сигнала, с которым совмещён выход блока. Четыре отсчёта входного сигнала поступают на вход блока реализации свёртки. Их значения умножаются на обозначенные отсчёты импульсной характеристики; получаемые произведения складываются, а результат направляется в соответствующий элемент выходного сигнала. На рисунке показан пример вычисления  $y[6]$  на основе отсчётов  $x[3], x[4], x[5]$  и  $x[6]$ .

Чтобы вычислить выходной отсчёт  $y[7]$ , надо сдвинуть блок, представляющий свёртку, на один отсчёт вправо. При этом на вход процедуры свёртки поступают другие четыре входных отсчёта  $x[4]...x[7]$ . Результат свёртки выводится в соответствующий элемент выходного массива  $y[7]$ . Процедура повторяется аналогично для всех отсчётов выходного сигнала.

Очень важным является порядок следования отсчётов импульсной характеристики, изображённой внутри блока реализации свёртки. Импульсная характеристика оказывается зеркально перевёрнутой. Её нулевой отсчёт является крайним правым, а остальные отсчёты идут по возрастанию их индекса справа налево. Чтобы лучше понять, что значит «зеркально перевёрнутая», сравните вид импульсной характеристики на Рис. 6.8 с «нормальной» импульсной характеристикой на Рис. 6.5. Почему используется зеркальное отражение? Просто этого требует математика процедуры свёртки. Импульсная характеристика говорит нам о том, какую реакцию даёт каждый отдельный отсчёт входного сигнала на выходе системы. Чтобы узнать, как каждый отдельный отсчёт выходного сигнала получается из совокупности входных отсчётов, требуется взвешивать входной сигнал зеркально перевёрнутой копией импульсной характеристики.

На Рис. 6.9 показан механизм реализации свёртки для некоторых отсчётов выходного сигнала. При этом становится видна одна из важных проблем, возникающих при реализации свёртки. На (а) блок свёртки находится в крайнем левом положении, и вычисляется выходной отсчёт  $y[0]$ . При этом в процессе вычислений должны участвовать входные отсчёты  $x[-3], x[-2], x[-1]$  и  $x[0]$ . Однако от-

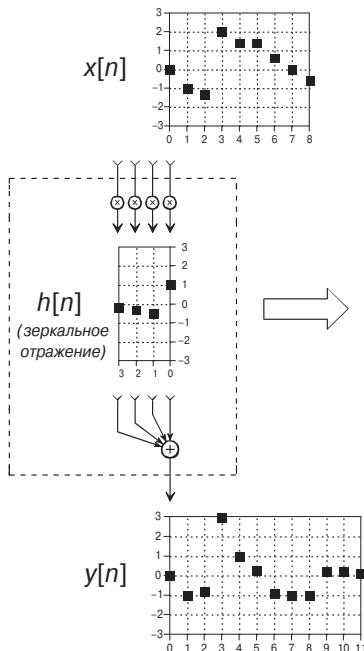
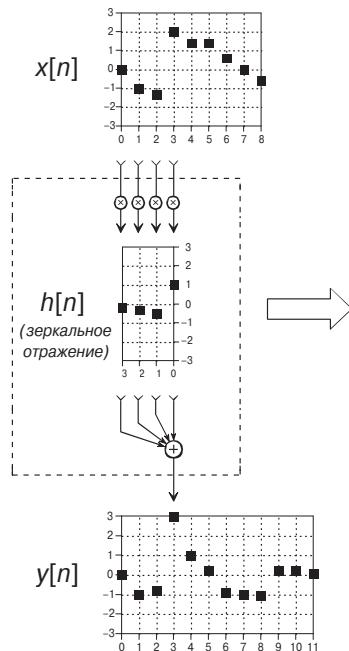
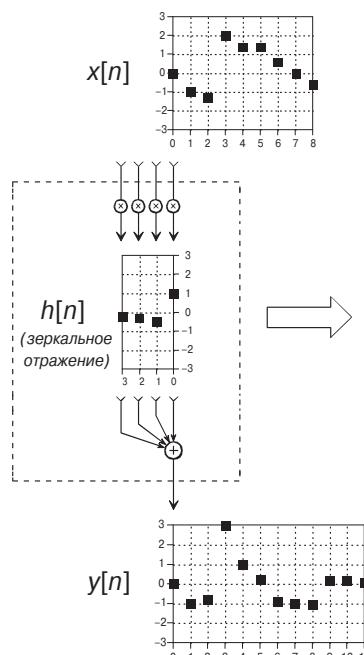
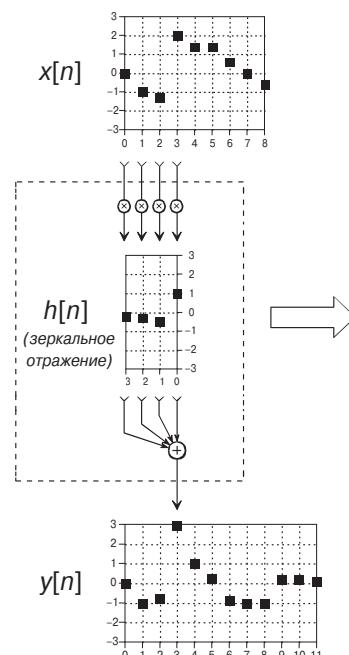


**Рис. 6.8.** Механизм реализации свёртки. Данная схема иллюстрирует, как каждый отдельный отсчёт выходного сигнала получается из отсчётов входного сигнала и импульсной характеристики системы.

счётов  $x[-3]$ ,  $x[-2]$  и  $x[-1]$  не существует! Такая же проблема возникает при вычислении отсчёта  $y[11]$ , требующего использования трёх входных отсчётов —  $x[9]$ ,  $x[10]$  и  $x[11]$  с индексами, превышающими длину входного сигнала ( $\varepsilon$ ).

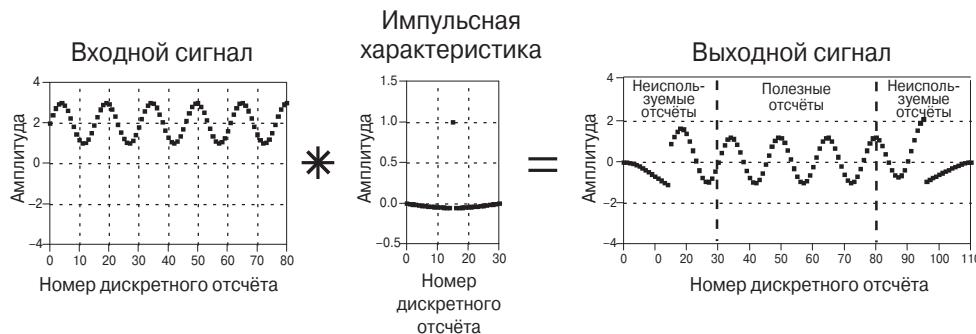
Решение может состоять в том, чтобы сформировать недостающие входные отсчёты каким-либо образом и дополнить ими сигнал слева и справа. Очевидным шагом является *дополнение нулями*. При этом программа реализации свёртки вместо того, чтобы обращаться к несуществующим входным элементам, будет загружать нули. Поскольку нулевой элемент, участвуя в операции умножения, не вносит изменений в результат обработки, свёртка, выполненная указанным образом, будет математически эквивалентна свёртке, выполненной с учётом модификации механизма вычислений на краях.

Есть, однако, одно важное замечание. Крайние правые и крайние левые отсчёты выходного сигнала формируются на основе неполной информации. На языке ЦОС говорят, что *импульсная характеристика не является полностью согласованной с входным сигналом*. Если импульсная характеристика включает  $M$  отсчётов, то первые и последние ( $M - 1$ ) отсчёты выходного сигнала будут вычислены на основе меньшего числа входных данных, чем остальные. Здесь можно провести аналогию с электрическими цепями, которые после подачи питания требуют некоторого времени для стабилизации, входа в установленный режим. Различие, однако, состоит в том, что в случае аналоговой электроники влияние пере-

**a) Вычисление выходного отсчёта  $y[0]$** **б) Вычисление выходного отсчёта  $y[3]$** **в) Вычисление выходного отсчёта  $y[8]$** **г) Вычисление выходного отсчёта  $y[11]$** **Рис. 6.9.** Механизм свёртки «в действии». Реализация свёртки при вычислении четырёх различных отсчётов выходного сигнала:  $y[0], y[3], y[8]$  и  $y[11]$ .

ходного процесса можно иногда проигнорировать, а в случае цифровой обработки это явление гораздо более заметно.

На Рис. 6.10 показан пример того, как описанные краевые эффекты могут приводить к неправильной работе системы. Входной сигнал представлен суммой синусоидального колебания и постоянного смещения. Желательно в процессе обработки избавиться от постоянной составляющей и сохранить без изменений



**Рис. 6.10.** Краевые эффекты при реализации свёртки. Когда входной сигнал сворачивается с  $M$ -точечной импульсной характеристикой, первые и последние  $(M - 1)$  отсчёты выходного сигнала следует отбрасывать. На рисунке приведён пример системы, имеющей импульсную характеристику фильтра высоких частот и применяемую для удаления из входного сигнала постоянной составляющей.

гармонический сигнал. Это требует применения высокочастотного фильтра, импульсная характеристика которого приведена на рисунке. Проблема в том, что 30 первых и 30 последних выходных отсчётов неверны! Как они будут выглядеть, можно рассчитать, если представить, что обрабатывается входной сигнал, дополненный 30 нулевыми отсчётами  $x[-1]...x[-30]$  слева и 30 нулевыми отсчётами  $x[81]...x[110]$  справа. Сигнал на выходе получается при обработке всего такого «удлинённого» входного сигнала. Проблема «краевого эффекта» широко распространена в ЦОС. Можно принять за общее правило, что начальные и конечные отсчёты обрабатываемых сигналов следует игнорировать.

Теперь перейдём к математике. Взяв за основу рассмотренную схему, запишем уравнение свёртки. Если сигнал  $x[n]$  имеет длину  $N$  отсчётов с 0-го по  $(N - 1)$ -й, а сигнал  $h[n]$  — длину  $M$  отсчётов с 0-го по  $(M - 1)$ -й, то сигнал  $y[n]$ , равный свёртке этих двух сигналов  $y[n] = x[n] * h[n]$ , будет иметь длину  $N + M - 1$  и вычисляться по формуле

$$y[i] = \sum_{j=0}^{M-1} h[j]x[i-j]. \quad (6.1)$$

**Вычисление свёртки.** Это строгое математическое определение свёртки, кратко записываемое как  $y[n] = x[n] * h[n]$ . В этом выражении  $h[n]$  имеет длину  $M$  отсчётов с 0-го по  $(M - 1)$ -й.

Это выражение называется свёрткой сигналов (последовательностей отсчётов). Формула свёртки позволяет вычислять отдельные отсчёты выходного сигна-

ла независимо от остальных. Номер вычисляемого выходного отсчёта определяет индекс  $i$ . В схеме, показанной на Рис. 6.8, этот индекс задаёт положение блока реализации свёртки относительно массива входного сигнала. В программной реализации этот индекс определяет состояние счётчика цикла, в каждой итерации которого осуществляется расчёт одного отсчёта выходного сигнала. При вычислении очередного выходного отсчёта «внутри» механизма реализации свёртки действует другой индекс — индекс  $j$ . Он меняется от 0 до  $M - 1$ , и каждому его значению соответствует умножение очередного отсчёта импульсной характеристики  $h[n]$  на соответствующий отсчёт входного сигнала  $x[i - j]$ . Все полученные таким образом произведения суммируются, образуя новый отсчёт выходного сигнала. Рекомендуем детально изучить выражение (6.1), так, чтобы было полностью понятно его соответствие механизму реализации свёртки, проиллюстрированному на Рис. 6.8. Огромная часть всей ЦОС построена на использовании этого выражения. (Пусть вас не смущает употребление символа  $n$  в выражении  $y[n] = x[n] * h[n]$ . Он используется просто для того, чтобы показать, что некоторая переменная должна присутствовать в качестве индекса массива. Иногда уравнения записывают и в виде  $y[ ] = x[ ] * h[ ]$ , избегая употребления лишней переменной.)

**Программа 6.2** реализует свёртку в соответствии с её описанием со стороны выхода системы, т. е. в прямом соответствии с выражением (6.1). Результат выполнения этой программы полностью совпадает с результатом программы, вычисляющей свёртку в соответствии с её описанием со стороны входа системы (см. Программу 6.1). Отметим основное различие между двумя этими программами. В первой из них, построенной в соответствии с описанием свёртки со стороны входа, цикл обработки организован по всем отсчётам входного сигнала (строка 220 в Программе 6.1), а во второй — по всем отсчётам выходного сигнала (строка 180 в Программе 6.2).

### Программа 6.2

```

100          'СВЁРТКА, ПРЕДСТАВЛЕННАЯ СО СТОРОНЫ ВЫХОДА
110          '
120 DIM X[80]          'Сигнал на входе, 81 отсчёт
130 DIM H[30]          'Импульсная характеристика, 31 отсчёт
140 DIM Y[110]         'Сигнал на выходе, 111 отсчётов
150          '
160 GOSUB XXXX         'Вызов подпрограммы загрузки X[ ] и H[ ]
170          '
180 FOR I% = 0 TO 110   'Цикл по всем элементам массива Y[ ]
190 Y(I%) = 0           'Обнуление массива выходного сигнала
200 FOR J% = 0 TO 30    'Цикл по всем элементам массива H[ ]
210 IF (I%-J% < 0) THEN GOTO 240
220 IF (I%-J% > 80) THEN GOTO 240
230 Y(I%) = Y(I%) + H(J%) *
X(I%-J%)
240 NEXT J%
250 NEXT I%
260          '
270 GOSUB XXXX         'Вызов подпрограммы записи результата Y[ ]
280          '
290 END

```

Рассмотрим в деталях новую программу. Цикл FOR-NEXT, расположенный в строках 180...250, «пробегает» по всем элементам выходного массива с использованием индекса I%. Для каждого значения индекса в строках 200...230 организуется ещё один внутренний цикл, в котором происходит вычисление выходного отсчёта Y[I%]. В строке 190 элемент Y[I%] обнуляется, позволяя использовать его для накопления результатов умножений. Цикл FOR-NEXT в строках 200...240 непосредственно реализует выражение (6.1). Индекс J% пробегает все элементы массива импульсной характеристики. В строке 230 происходит умножение отсчётов импульсной характеристики H[J%] на соответствующие входные отсчёты X[I% - J%] и добавление результата к текущему значению аккумулятора.

В строке 230 происходит выборка отсчёта входного сигнала X[I%-J%]. Строки 210 и 220 защищают программу от выхода за пределы массива X[ ], т. е. от обращений к элементам с индексом, меньшим 0 или большим 80. Это означает, что данная программа игнорирует вычисления, связанные с обращениями к неизвестным краевым значениям входного сигнала. Альтернативой этому могло бы быть расширение массива слева и справа нулевыми элементами, так чтобы были разрешены обращения к входным отсчётам с X[-30] по X[110]. Есть также третий вариант вычислений: изменять счётчик цикла FOR-NEXT в строке 180 от 30 до 80, а не от 0 до 110. В этом случае программа вычисляла бы значения только тех отсчётов выходного сигнала, для которых известны все входные данные. Один из этих трёх вариантов обязательно должен быть реализован в программе. Если этого не предусмотреть, то результат выполнения программы может оказаться непредсказуемым из-за обращений к несуществующим данным.

## 6.5. СУММА ВЗВЕШЕННЫХ ВХОДНЫХ ОТСЧЁТОВ

Линейная система полностью описывается своей импульсной характеристикой. Это является основой описания свёртки со стороны входа: каждый входной отсчёт формирует реакцию на выходе системы в виде сдвинутой во времени и умноженной на постоянный множитель импульсной характеристики. Развитие этого утверждения с помощью математических выкладок приводит к описанию свёртки со стороны выхода: каждый отсчёт выходного сигнала получается при умножении входных отсчётов на зеркальную копию импульсной характеристики. Всё это так, однако это ещё не даёт полного представления о вариантах использования операции свёртки, обуславливающих её широчайшее распространение в теории и практике ЦОС.

Обратимся вновь к механизму реализации свёртки, изображённому на Рис. 6.8. Забудем о том, что сигнал внутри пунктирного прямоугольника — это импульсная характеристика. Будем рассматривать её просто как набор *весовых коэффициентов*. В этом случае можно говорить, что каждый отсчёт выходного сигнала получается как сумма взвешенных входных отсчётов. Значение каждого из них определяется частью отсчётов сигнала на входе и выбором значений весовых коэффициентов. Например, если используются 10 одинаковых весовых коэффициентов, равных 0.1, то отсчёты выходного сигнала будут представлять собой средние значения десяти отсчётов сигнала на входе.

Развивая далее эту идею, мы приходим к тому, что в случае массива весовых коэффициентов выравнивание весового окна по номеру вычисляемого отсчёта

становится необязательным. Например, на **Рис. 6.8** при использовании импульсной характеристики отсчёт выходного сигнала  $y[6]$  вычисляется с использованием входных отсчётов  $x[3], x[4], x[5]$  и  $x[6]$ . В то же время, если рассматривать вычисление свёртки с использованием набора весовых коэффициентов, весовую функцию можно выбирать симметричной относительно вычисляемого выходного отсчёта. То есть отсчёт  $y[6]$  может рассчитываться на основе входных отсчётов  $x[4], x[5], x[6], x[7]$  и  $x[8]$ . Тогда, не меняя обозначений **Рис. 6.8**, получаем, что для этих пяти входных отсчётов потребуется знание следующих пяти весовых коэффициентов:  $h[2], h[1], h[0], h[-1]$  и  $h[-2]$ . Таким образом, при рассмотрении импульсной характеристики как весового окна, симметричного относительно вычисляемого отсчёта, требуется знание её отрицательных значений. К этому вопросу мы ещё вернёмся в следующей главе.

В математике есть только одно определение свёртки, отражённое в выражении **(6.1)**. Однако при решении различных практических задач часто возникает необходимость рассматривать её с двух позиций. Либо вы используете её при описании системы с помощью импульсной характеристики, либо вам удобнее описывать систему набором весовых коэффициентов. Оба этих подхода, используемых в ЦОС, необходимо знать, а также уметь переходить от одного к другому.

# Глава 7

## СВОЙСТВА СВЁРТКИ

Характеристики линейной системы полностью определяются её импульсной характеристикой, что математически определяется процедурой свёртки. Это является основой многих методов обработки сигналов. Например, цифровые фильтры проектируются путём расчёта импульсной характеристики с желаемыми свойствами. В радиолокации обнаружение воздушных целей осуществляется путём анализа регистрируемой импульсной характеристики. Подавление эхо-сигналов в протяжённых линиях телефонной связи выполняется путём формирования импульсной характеристики, компенсирующей импульсную характеристику канала связи, создающего отражённые сигналы. Эти примеры можно продолжать и продолжать. Данная глава посвящена описанию свойств свёртки и вопросам её использования. В начале главы рассматривается несколько обще-распространённых видов импульсных характеристик. Затем приводятся методы описания последовательных и параллельных соединений линейных систем. Далее вводится понятие корреляции. В заключение обсуждается серьёзная проблема, возникающая при реализации свёртки: время вычислений при использовании обычных алгоритмов и вычислительных средств часто оказывается неприемлемо большим.

### 7.1. Типовые импульсные характеристики

#### 7.1.1. Единичный импульс

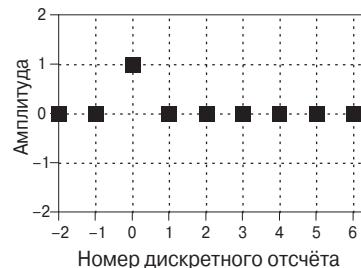
Простейшей импульсной характеристикой может при определённых условиях оказаться сам *единичный импульс* (**Рис. 7.1а**). В системе с такой импульсной характеристикой единичный импульс на входе проходит на выход без изменений. Более того, произвольный сигнал на входе также пройдёт на выход, не меняясь, потому что свёртка любого сигнала с единичным импульсом равна самому сигналу. Математическая запись этого свойства выглядит следующим образом:

$$x[n] * \delta[n] = x[n]. \quad (7.1)$$

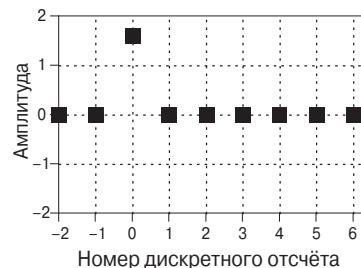
Единичный импульс инвариантен к свёртке: сворачивание произвольного сигнала с единичным импульсом оставляет этот сигнал без изменений.

Данное свойство делает единичный импульс инвариантным к свёртке точно так же, как ноль инвариантен к сложению ( $a + 0 = a$ ), а единица инвариантна к умножению ( $a \times 1 = a$ ). На первый взгляд системы с такой импульсной характеристикой кажутся слишком простыми и не представляющими интереса. Однако это не так! Данный класс систем является идеальным для многих приложений

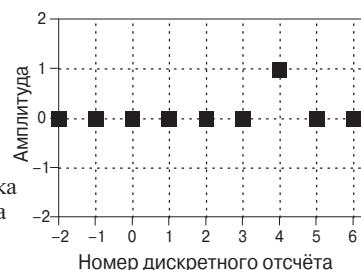
**a) Инвариантность.** Единичный импульс инвариантен к свёртке. Свёртка произвольного сигнала с единичным импульсом оставляет сигнал без изменений. Системы с такой импульсной характеристикой идеальны для систем хранения и передачи данных.



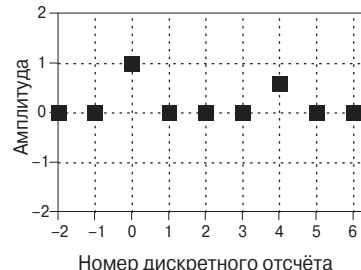
**б) Усиление и ослабление.** Умножив единичный импульс на постоянный множитель, получим импульсную характеристику системы, являющейся усилителем или аттенюатором сигнала. Так, система с импульсной характеристикой, показанной на данном рисунке, будет усиливать сигналы в 1.6 раза.



**в) Сдвиг.** Система, имеющая импульсную характеристику в виде единичного импульса, сдвинутого вдоль временной оси, будет осуществлять аналогичный сдвиг сигнала на выходе по отношению к входному сигналу. В зависимости от направления сдвига такая система будет являться системой задержки или предсказания. Импульсная характеристика на данном рисунке вызывает задержку входного сигнала на 4 периода дискретизации.



**г) Эхо.** Импульсная характеристика в виде суммы единичного импульса и импульса, сдвинутого и умноженного на постоянный коэффициент, вызывает на выходе системы появление входного сигнала и его эха. В приведённом примере эхо-сигнал задержан на четыре отсчета, а его амплитуда составляет 60% уровня исходного сигнала.



**Рис. 7.1.** Импульсные характеристики простейшего типа, образованные на основе единичного импульса с использованием сдвига и умножения на постоянную величину.

цифровой обработки сигналов, в которых требуется, чтобы информация проходила через систему без изменений. К таким приложениям относятся, например, системы записи и хранения данных, системы передачи информации, измерительные системы и др.

Немного видоизменив единичный импульс, а именно умножив его на постоянный множитель, получим другую импульсную характеристику (**б**). В зависимости от значения множителя система окажется усилителем или аттенюатором.

Если описать такую систему с помощью свёртки (7.2), то усиление сигналов будет происходить при  $k$ , *большем единице*, а ослабление сигналов — при  $k$ , *меньшем единице*:

$$x[n] * k\delta[n] = kx[n]. \quad (7.2)$$

Система усиления или ослабления сигнала имеет импульсную характеристику в виде единичного импульса, умноженного на постоянный коэффициент. Характер усиления или ослабления определяется множителем  $k$ .

Если единичный импульс сдвинуть вдоль временной оси, получим третью импульсную характеристику (в). Система с такой импульсной характеристикой осуществляет сдвиг выходного сигнала на соответствующую величину относительно входного. В зависимости от направления сдвига система будет задерживать сигнал или предсказывать его. Обозначим величину сдвига параметром  $s$ . Тогда работа системы будет описываться выражением

$$x[n] * \delta[n + s] = x[n + s]. \quad (7.3)$$

Сдвиг сигнала на выходе по отношению к входному сигналу осуществляется системой, импульсная характеристика которой представлена сдвинутым по временной оси единичным импульсом. Параметр  $s$  определяет величину и направление сдвига.

В научных и инженерных приложениях встречается множество примеров, когда рассматриваются сигналы, смешённые друг относительно друга. Это, например, радиосигнал, излучаемый космическим зондом, и его копия, регистрируемая на Земле и характеризуемая задержкой, обусловленной временем распространения радиоволны в пространстве. Другой пример из биологии: электрические импульсы в соседних нервных клетках являются сдвинутыми друг относительно друга копиями. Сдвиг сигналов вызван прохождением через синапс, соединяющий клетки.

Наконец, на (г) изображена импульсная характеристика системы, которая представляет собой сумму единичного импульса и его копии, сдвинутой во времени и умноженной на постоянный коэффициент. Используя приведённые выше рассуждения и *принцип суперпозиции*, можно сделать вывод, что на выходе такой системы будет наблюдаться копия входного сигнала плюс другая копия входного сигнала, отличающаяся временной задержкой, т. е. эхо. Эхо-сигналы широко распространены в ЦОС. Например, в аудиоприложениях эхо-сигналы искусственно вводятся в систему, чтобы сделать звучание более естественным и приятным. Радиолокация и гидролокация используют отражённые эхо-сигналы для обнаружения воздушных и подводных объектов. В геофизике сигналы эха несут информацию о составе грунта и месторождениях полезных ископаемых. В системах связи к эхо-сигналам проявляется интерес другого рода. Здесь их стараются подавить.

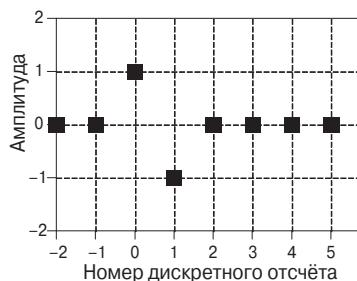
## 7.1.2. Процедуры дифференциального и интегрального исчисления

С помощью *свёртки* над *дискретными сигналами* могут быть выполнены преобразования, аналогичные *дифференцированию* и *интегрированию*. Однако, поскольку термины «производная» и «интеграл» относятся к непрерывным сигна-

лам, для дискретных сигналов были предложены свои обозначения. В дискретной области аналогом первой производной является *первая разность*. Аналогом интеграла служит *текущая сумма*. Эти операции также называют *дискретной производной* и *дискретным интегралом*, что, однако, является неофициальной терминологией и может заставить математиков нахмурить брови.

На Рис. 7.2 показаны импульсные характеристики систем, вычисляющих первую разность и текущую сумму. Рис. 7.3 иллюстрирует примеры использования

**а)** Первая разность. Первая разность — это дискретный аналог первой производной. Каждый отсчёт выходного сигнала равен разности двух соседних входных отсчётов. Другими словами, выходной сигнал отражает скорость изменения входного сигнала.



**б)** Текущая сумма. Текущая сумма — это дискретный аналог интеграла (первообразной). Каждый отсчёт выходного сигнала равен сумме всех входных отсчётов, предшествующих текущему. Примечательно, что импульсная характеристика при этом является бесконечной, что не очень-то хорошо.

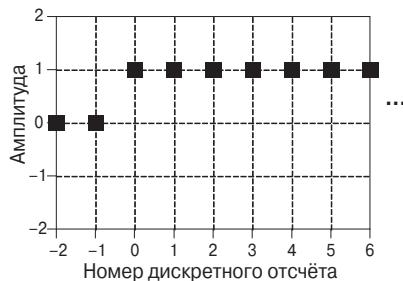


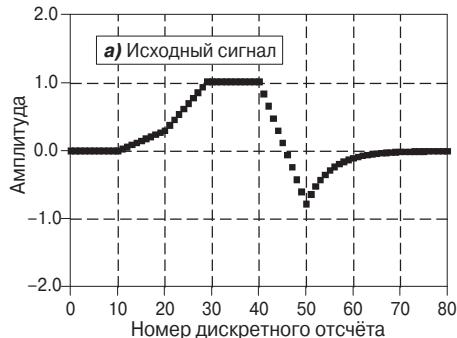
Рис. 7.2. Импульсные характеристики систем, реализующих дискретные аналоги дифференциального и интегрального исчисления.

таких импульсных характеристик. На (а) показан сигнал, состоящий из нескольких фрагментов с различной крутизной наклона. Свёртка такого сигнала с импульсной характеристикой системы, вычисляющей первую разность, даёт на выходе сигнал, представленный на (б). Так же как в случае взятия первой производной, амплитуда каждого отсчёта сигнала первой разности определяется крутизной входного сигнала в соответствующий момент времени. Вычисление текущей суммы — это процедура, обратная нахождению первой разности. Поэтому можно сказать, что свёртка входного сигнала, изображённого на (б), с импульсной характеристикой системы, вычисляющей текущую сумму, даёт на выходе сигнал, показанный на (а).

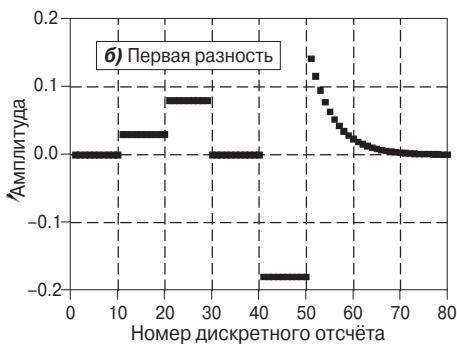
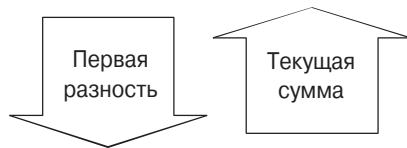
Рассматриваемый тип импульсных характеристик является достаточно простым и не требующим при реализации свёртки выполнения всех действий классической процедуры. Здесь удобнее представлять свёртку как расчёт отсчётов выходного сигнала на основе суммы взвешенных отсчётов входного. При этом, например, первая разность может быть вычислена как

$$y[n] = x[n] - x[n - 1]. \quad (7.4)$$

Вычисление первой разности. Здесь  $x[n]$  — входной сигнал, а  $y[n]$  — первая разность.



**Рис. 7.3.** Пример реализации дискретных операций дифференциального и интегрального исчисления. Сигнал, изображенный на (б), является первой разностью сигнала, изображённого на (а). Соответственно сигнал на (а) является текущей суммой сигнала на (б). Вычисление первой разности и текущей суммы являются дискретными аналогами процедур дифференцирования и интегрирования.



То есть каждый отсчёт выходного сигнала равен разности двух соседних входных отсчётов. Например,  $y[40] = x[40] - x[39]$ . Следует заметить, что это не единственный способ расчёта дискретной производной. Другим распространённым подходом является использование крутизны фрагмента сигнала, симметричного относительно точки наблюдения, т. е.  $y[n] = (x[n+1] - x[n-1])/2$ .

Аналогично каждый отсчёт текущей суммы вычисляется суммированием всех отсчётов входного сигнала, расположенных левее элемента, для которого выполняется расчёт. Например, если  $y[n]$  является текущей суммой  $x[n]$ , то элемент  $y[40]$  рассчитывается как сумма элементов  $x[0] \dots x[40]$ , а элемент  $y[41]$  — как сумма элементов  $x[0] \dots x[41]$ . Конечно, вычисление текущей суммы непосредственно в представленном виде было бы малоэффективно. Например, если отсчёт  $y[40]$  уже рассчитан, то для вычисления  $y[41]$  достаточно выполнить всего одно сложение:  $y[41] = y[40] + x[41]$ . Или в форме уравнения

$$y[n] = x[n] + y[n-1]. \quad (7.5)$$

Вычисление текущей суммы. Здесь  $x[n]$  — входной сигнал, а  $y[n]$  — текущая сумма.

Выражения такого типа называются *рекурсивными разностными уравнениями*. Мы вновь обратимся к ним в Главе 19. Сейчас достаточно запомнить, что выражения данного типа являются частным случаем свёртки, имеющим место при использовании в качестве импульсных характеристик сигналов, изображённых на Рис. 7.2. Программы 7.1 и 7.2 реализуют вычисления в соответствии с дискретными процедурами дифференцирования и интегрирования. Исходный сигнал считается загруженным в массив  $X[ ]$ . Результат обработки (первая разность или текущая сумма) записывается в массив  $Y[ ]$ . Индексы элементов обоих массивов меняются в диапазоне  $0\dots(N\% - 1)$ .

### **Программа 7.1. Программа вычисления первой разности**

```
100 'Вычисление первой разности
110 Y[0] = 0
120 FOR I% = 1 TO N%-1
130 Y[I%] = X[I%] - X[I%-1]
140 NEXT I%
```

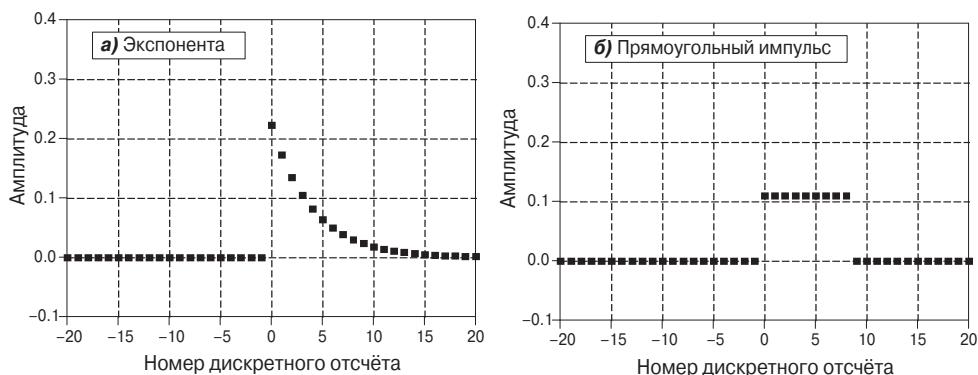
### **Программа 7.2. Программа вычисления текущей суммы**

```
100 'Вычисление текущей суммы
110 Y[0] = X[0]
120 FOR I% = 1 TO N%-1
130 Y[I%] = Y[I%-1] + X[I%]
140 NEXT I%
```

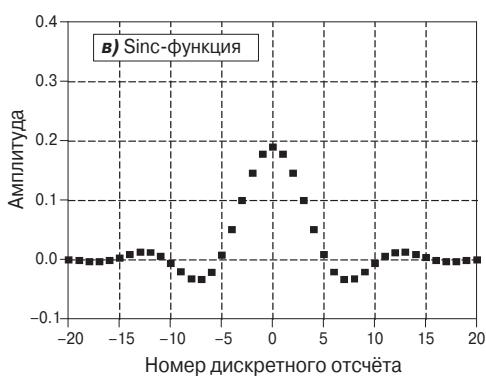
## **7.1.3. Фильтры нижних и верхних частот**

Вопросы проектирования *цифровых фильтров* подробно рассматриваются в последующих главах. Сейчас для нас будет достаточно познакомиться с общим видом ядра *фильтра нижних и верхних частот* (ядро фильтра — это ещё один термин, используемый для обозначения *импульсной характеристики*). На Рис. 7.4 приведён ряд типовых форм ядра фильтра нижних частот. В общем случае ядро низкочастотного фильтра строится на основе группы положительных отсчётов, следующих непосредственно друг за другом. Это обуславливает формирование каждого выходного отсчёта как взвешенное среднее группы соседних входных отсчётов. Усреднение отсчётов входного сигнала приводит к сглаживанию формы сигнала, т. е. к устранению высокочастотных компонентов. В некоторых случаях ядро фильтра нижних частот может включать и ряд отрицательных отсчётов на краях, как, например, в случае ядра фильтра в форме *sinc-функции* ( $\sin(x)/x$ ), показанной на (в). Точно так же как и в аналоговой электронике, в цифровой обработке сигналов низкочастотные фильтры используются для подавления шума, формирования и селекции сигналов и для других задач.

Частота среза частотной характеристики фильтра регулируется остротой формы его ядра. Если на нулевой частоте фильтр нижних частот имеет коэффициент усиления, равный единице, то это означает, что сумма всех отсчётов его импульсной характеристики должна быть равна единице. Некоторые типы импульсных характеристик, такие как, например, изображённые на (а и в), теоретически продолжаются бесконечно, никогда не устанавливаясь в ноль. На практике приходится ограничивать такие импульсные характеристики конечным числом

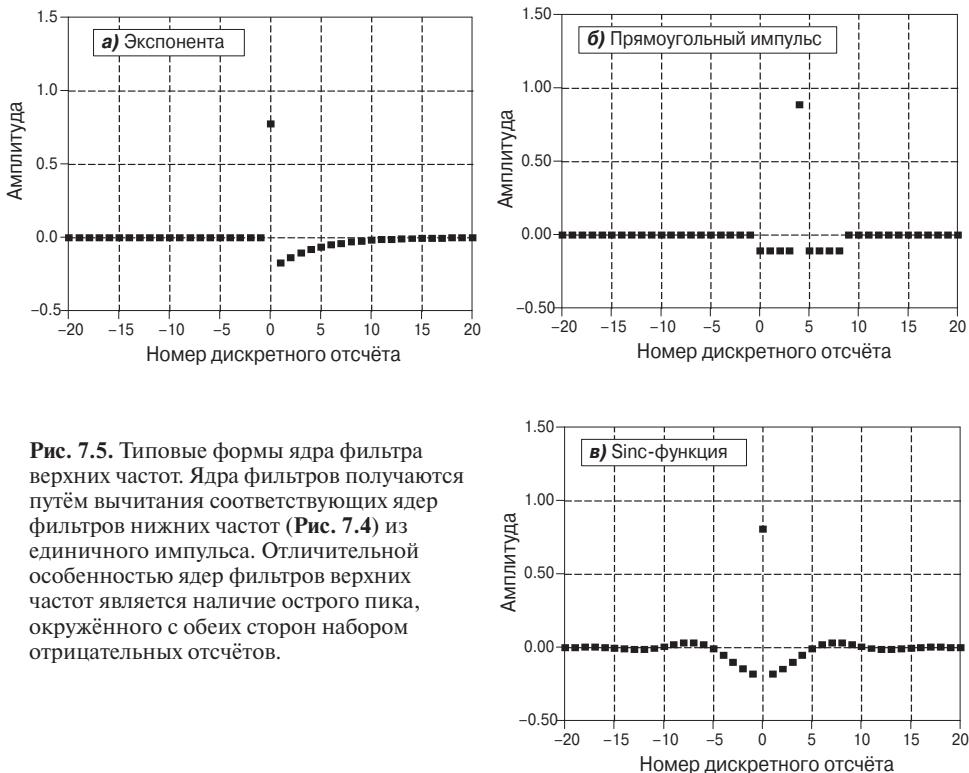


**Рис. 7.4.** Типовые формы ядра фильтра низких частот. Ядра фильтров низких частот характеризуются наличием группы соседних положительных отсчётов, обуславливающих усреднение и сглаживание сигнала на выходе. Как будет показано в следующих главах, каждое из приведённых ядер будет иметь преимущество при решении тех или иных задач. Экспоненциальное ядро (а) представляет простейшую форму рекурсивного фильтра. Прямоугольное ядро (б) характеризуется хорошим подавлением шумов при сохранении крутизны фронтов. Синк-функция — кривая, описываемая выражением  $\sin(x)/x$  (в), — используется для разделения частотных полос сигнала.



дискретных отсчётов, отбрасывая часть информации на краях. А как же иначе можно представить их в вычислительном устройстве?

На Рис. 7.5 показаны три типовых ядра фильтров верхних частот, полученные из соответствующих ядер фильтров низких частот, приведённых на Рис. 7.4. Такой подход к проектированию фильтров является общераспространённым: сначала создаётся низкочастотный фильтр, а затем он трансформируется в тот тип фильтра, который нужно получить: высокочастотный, полосовой, режекторный и т. д. Чтобы понять переход от фильтра низких частот к фильтру верхних частот, необходимо вспомнить, что импульсная характеристика в виде *единичного импульса* пропускает на выход системы все частотные компоненты без изменений, а импульсная характеристика фильтра низких частот — только низкочастотные компоненты. С помощью *принципа суперпозиции* можно получить, что система с импульсной характеристикой в виде разности единичного импульса и импульсной характеристики фильтра низких частот будет пропускать на выход все частотные компоненты минус компоненты, расположенные на низких частотах. То есть мы получаем высокочастотный фильтр! При вычитании единичный импульс обычно сдвигается в центр симметрии импульсной характеристики низкочастотного фильтра или в ноль, если импульсная характеристика не симметрична (Рис. 7.5). На нулевой частоте фильтры верхних частот имеют нулевой коэффициент передачи, что обусловлено равенством нулю суммы всех отсчётов ядра фильтра.

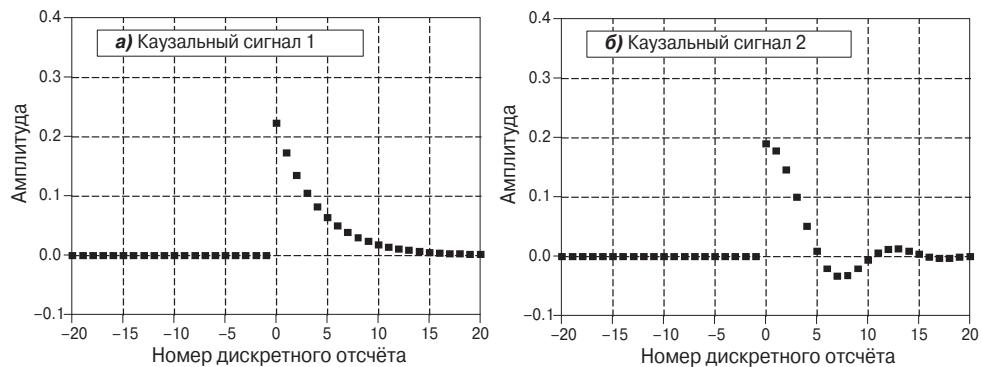


**Рис. 7.5.** Типовые формы ядра фильтра верхних частот. Ядра фильтров получаются путём вычитания соответствующих ядер фильтров нижних частот (Рис. 7.4) из единичного импульса. Отличительной особенностью ядер фильтров верхних частот является наличие острого пика, окружённого с обеих сторон набором отрицательных отсчётов.

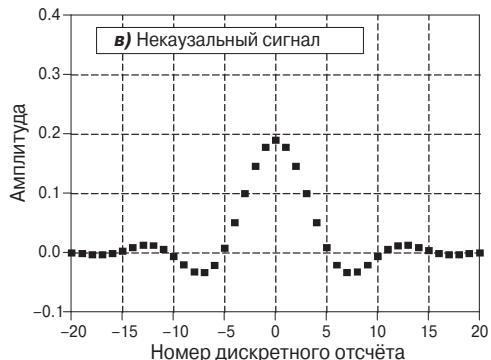
#### 7.1.4. Каузальные и некаузальные сигналы

Представьте себе простую аналоговую электрическую цепь. Если подать на её вход электрический импульс, на выходе появится сигнал реакции цепи. Это очевидно и следует из простых причинно-следственных связей, на которых строится весь наш мир. В основу положена аксиома: любое явление имеет причину, благодаря которой и после которой оно возникает. В этом проявляется основное свойство понятия, которое мы называем временем. Теперь попробуем перенести этот принцип на цифровую систему, которая преобразует входной сигнал в выходной, причём оба этих сигнала представлены в вычислительном устройстве массивами чисел. Если считать, что цифровая система полностью повторяет работу реальных непрерывных цепей, тот же принцип причинно-следственной связи должен действовать в ней так же, как он действует в реальном мире. То есть, например, входной отсчёт с номером 8 может влиять только на выходные отсчёты с 8-го и выше. Системы, оперирующие в соответствии с этим принципом, называются *каузальными*. Однако является очевидным, что цифровые системы могут и не соблюдать это правило. Поскольку как входной, так и выходной сигнал представлены массивами чисел, одновременно хранящимися в вычислительном устройстве, расчёт любого отсчёта выходного сигнала можно производить на основе любых входных отсчётов.

Как показано на Рис. 7.6, для каузальных систем характерно, что все отсчёты импульсной характеристики в отрицательные моменты времени равны нулю. Обосновать данное утверждение лучше всего с позиции рассмотрения *свёртки* со стороны входа. Чтобы система была каузальной, необходимо, чтобы каждый  $n$ -й отсчёт входного сигнала влиял только на отсчёты выходного сигнала с индексом  $n$  и более. В общем случае понятие каузальности используется для любого сигнала, у которого все отсчёты в отрицательные моменты времени равны нулю, независимо от того, является он импульсной характеристикой или нет.

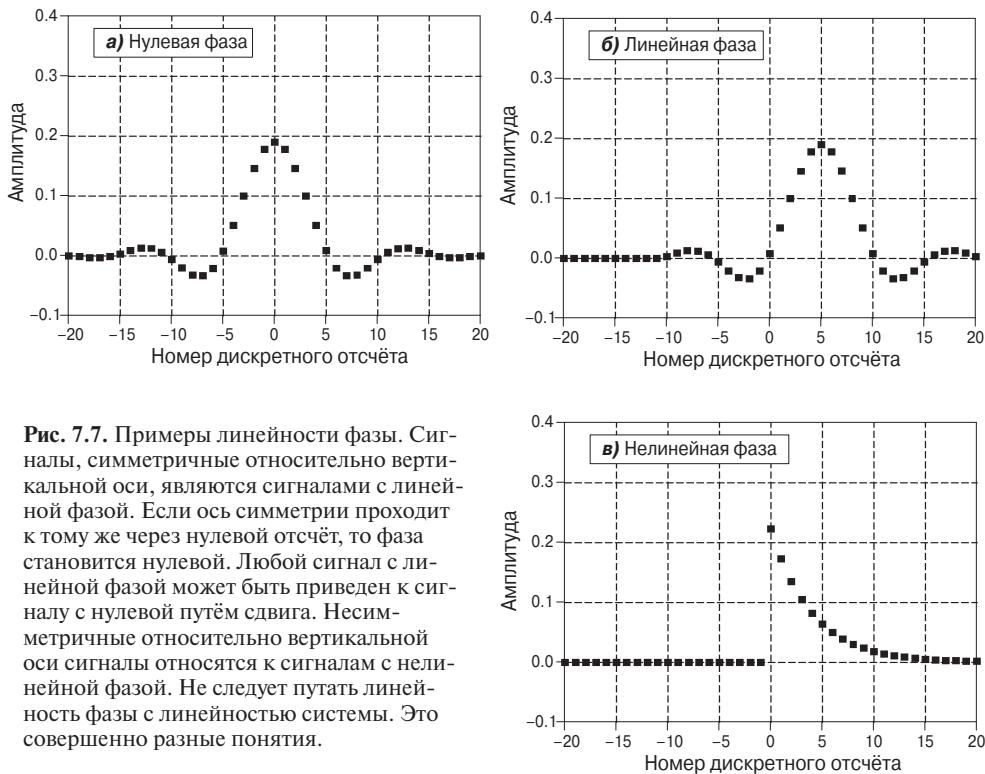


**Рис. 7.6.** Примеры каузальных сигналов. Импульсная характеристика (или сигнал) называется каузальной, если все её значения в отрицательные моменты времени равны нулю. На рисунке показаны три примера. Любой сигнал конечной длины, являющийся некаузальным, может быть приведён к каузальному простым сдвигом вдоль временной оси.



### 7.1.5. Сигналы с нулевой, линейной и нелинейной фазой

Сигнал называют *сигналом с нулевой фазой*, если он является симметричным относительно вертикальной оси, проходящей через нулевой отсчёт (Рис. 7.7). Если сигнал симметричен относительно вертикальной оси, проходящей через какой-либо другой отсчёт, его называют *сигналом с линейной фазой*. Сигнал с линейной фазой может быть приведён к сигналу с нулевой фазой с помощью простого сдвига вдоль временной оси. Наконец, сигнал называют *сигналом с нелинейной фазой*, если он не обладает симметрией относительно вертикальной оси.



**Рис. 7.7.** Примеры линейности фазы. Сигналы, симметричные относительно вертикальной оси, являются сигналами с линейной фазой. Если ось симметрии проходит к тому же через нулевой отсчёт, то фаза становится нулевой. Любой сигнал с линейной фазой может быть приведен к сигналу с нулевой путём сдвига. Несимметричные относительно вертикальной оси сигналы относятся к сигналам с нелинейной фазой. Не следует путать линейность фазы с линейностью системы. Это совершенно разные понятия.

Вам может показаться, что названия сигналов никак не связаны с их определениями. Действительно, какое отношение фаза сигнала может иметь к его симметричности? Однако связь есть, и проявляется она при анализе *частотных характеристик* сигналов, с которыми мы будем подробно знакомиться в последующих главах. Здесь попробуем дать короткое объяснение. Частотная характеристика любого сигнала включает две составляющие: *амплитудную* и *фазовую*. Фазовая характеристика сигнала, симметричного относительно вертикальной оси, проходящей через ноль, принимает нулевое значение для всех частот. Фазовая характеристика сигнала, симметричного относительно вертикальной оси, проходящей через какой-либо другой отсчёт, является линейной. И наконец, для сигнала, не обладающего симметрией относительно вертикальной оси, фазовая характеристика становится нелинейной.

Отдельно нужно осветить вопрос использования терминов линейности и нелинейности по отношению к фазе сигнала и к описанию свойств систем. Как связано понятие линейной фазы с понятием *линейной системы*, которое мы обсуждали в предыдущих главах? Ответ: никак! Линейность системы — это очень широкое понятие, на котором основана почти вся цифровая обработка сигналов (принцип суперпозиции, свойства однородности и аддитивности и т. д.). Линейность и нелинейность фазы говорят только о том, является фазовый спектр сигнала прямой линией или нет. В действительности система должна быть линейной, чтобы вообще можно было вести речь о линейности или нелинейности фазы.

## 7.2. Математические свойства свёртки

### 7.2.1. Свойство коммутативности

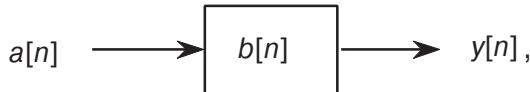
*Свойство коммутативности*, которым обладает операция *свёртки*, математически записывается так:

$$a[n] * b[n] = b[n] * a[n]. \quad (7.6)$$

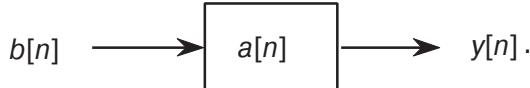
Свойство коммутативности свёртки. В соответствии с данным выражением порядок следования сигналов, участвующих в процедуре свёртки, не имеет значения.

То есть порядок следования двух сворачиваемых сигналов не имеет значения: в обоих случаях результат будет один и тот же. В теории систем это приводит к немного странному выводу (**Рис. 7.8**). Входной сигнал и *импульсную характеристику* системы можно поменять местами, и результат обработки останется прежним. Это интересный факт, который на практике, однако, обычно не имеет физического смысла. Ведь входной сигнал и импульсная характеристика по своей природе совершенно различны. То, что математика позволяет нам осуществлять какие-либо преобразования, ещё не значит, что эти преобразования будут иметь смысла. Например, представьте себе, что вы зарабатываете 10 долларов в час и соответственно  $10 \text{ долларов/час} \times 2000 \text{ часов/год} = 20000 \text{ долларов/год}$ . Свойство коммутативности, которым обладает операция умножения, позволяет вам получать в год ту же сумму, работая всего 10 часов/год за 2000 долларов/час. Попробуйте доказать своему боссу, что это имеет смысл! Тем не менее свойство коммутативности свёртки играет в ЦОС большую роль, позволяя преобразовывать уравнения так же, как в обычной алгебре.

ЕСЛИ



ТО



**Рис. 7.8.** Свойство коммутативности свёртки в теории систем. Данное свойство свёртки с математической точки зрения позволяет поменять местами сигнал на входе системы и импульсную характеристику. Результат обработки при этом не изменится. Несмотря на то, что это интересный факт, на практике такая замена обычно не имеет физического смысла. (На рисунке сигнал, показанный внутри блока, т. е.  $b[n]$  или  $a[n]$ , — это импульсная характеристика системы.)

### 7.2.2. Свойство ассоциативности

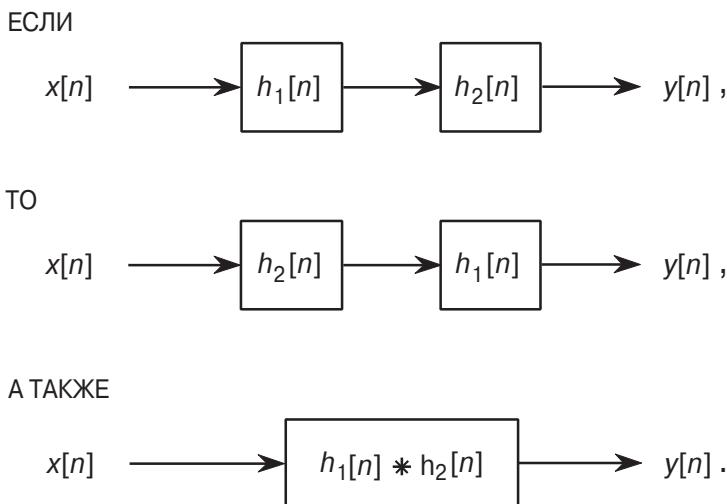
Могут ли участвовать в свёртке три (и более) сигнала? Ответ: да. Как это осуществляется, описывает *свойство ассоциативности*: сначала сворачиваются два сигнала и получается некоторый промежуточный результат; затем промежуточ-

ный результат сворачивается с третьим сигналом. Ассоциативность позволяет сворачивать сигналы в произвольном порядке. То есть в виде уравнения

$$(a[n] * b[n]) * c[n] = a[n] * (b[n] * c[n]). \quad (7.7)$$

Свойство ассоциативности описывает, как осуществляется свёртка трёх и более сигналов.

В теории систем свойство ассоциативности используется для описания поведения *последовательного соединения систем*. Две (или более) системы называются соединёнными последовательно, если выход одной системы является входом другой (Рис. 7.9). В соответствии со свойством ассоциативности порядок следования систем в последовательном соединении может быть изменён без изменения общего результата обработки. Более того, любое число систем в каскадном соединении может быть заменено одной системой. Импульсная характеристика такой эквивалентной системы рассчитывается как свёртка импульсных характеристик всех исходных систем.



**Рис. 7.9.** Свойство ассоциативности свёртки в теории систем. Свойство ассоциативности свёртки позволяет сформировать два утверждения, имеющих большое значение при описании последовательных соединений нескольких систем. Во-первых, порядок следования последовательно соединённых систем может меняться без изменения общего результата обработки на выходе. Во-вторых, две (или более) системы в последовательном соединении могут заменяться одной эквивалентной. Импульсная характеристика этой эквивалентной системы рассчитывается как свёртка импульсных характеристик исходных систем.

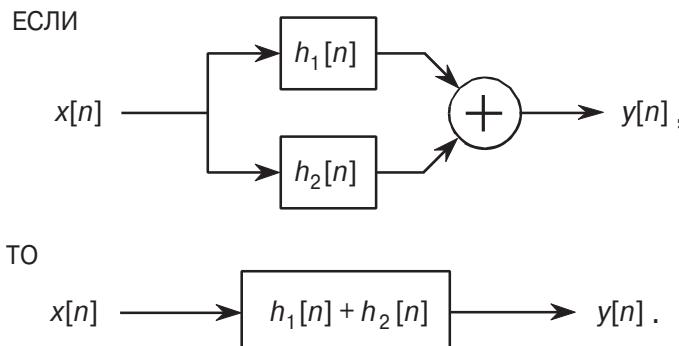
### 7.2.3. Свойство дистрибутивности

Математическая запись *свойства дистрибутивности* выглядит следующим образом:

$$a[n] * b[n] + a[n] * c[n] = a[n] * (b[n] + c[n]). \quad (7.8)$$

Свойство дистрибутивности свёртки используется при анализе параллельных соединений нескольких систем.

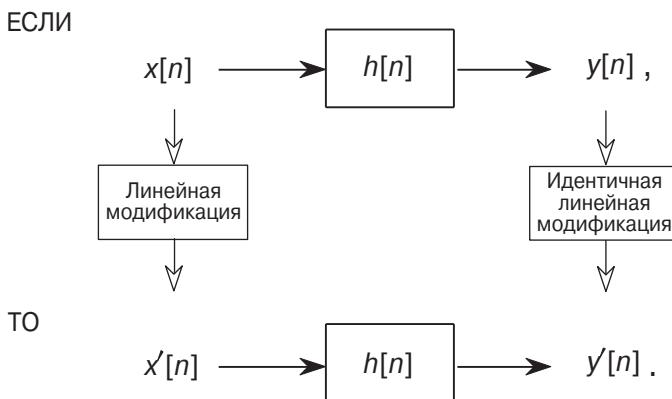
Свойство дистрибутивности свёртки описывает *параллельное соединение нескольких систем с суммируемыми выходами*. Такое соединение показано на Рис. 7.10. На вход двух (или более) систем подаётся один сигнал  $x[n]$ . Результаты обработки суммируются, образуя выходной сигнал  $y[n]$ . Использование свойства дистрибутивности позволяет заменить параллельное соединение систем одной эквивалентной, импульсная характеристика которой определяется как сумма импульсных характеристик исходных систем.



**Рис. 7.10.** Свойство дистрибутивности свёртки в теории систем. Свойство дистрибутивности свёртки показывает, что параллельное соединение нескольких систем с суммируемыми выходами может быть заменено одной. Импульсная характеристика этой эквивалентной системы определяется как сумма импульсных характеристик исходных систем.

## 7.2.4. Преемственность между входом и выходом

Это не строгое математическое правило, а скорее наше интуитивное понимание того, что происходит в большинстве цифровых систем. Пусть на вход линейной системы подаётся сигнал  $x[n]$ , вызывающий на выходе системы реакцию  $y[n]$  (Рис. 7.11). Пусть входной сигнал изменился по линейному закону и образовался

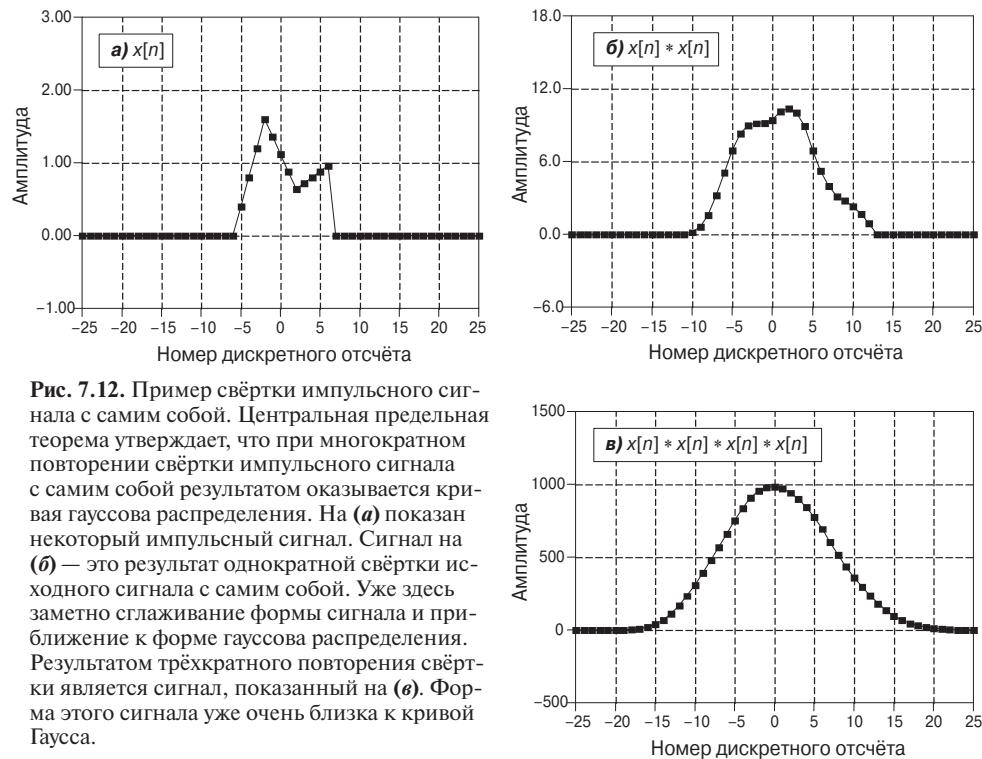


**Рис. 7.11.** Преемственность между входом и выходом. Это свойство обусловлено интуитивным пониманием того, что происходит в большинстве систем обработки сигналов. Линейная модификация сигнала на входе приводит к точно такой же линейной модификации сигнала на выходе.

новый —  $x'[n]$ . Получаемую при этом новую реакцию системы обозначим  $y'[n]$ . Вопрос: как изменение формы входного сигнала влияет на вид сигнала на выходе? Ответ: сигнал на выходе меняется точно по тому же линейному закону, что и сигнал на входе. Например, если входной сигнал был усилен в 2 раза, то и выходной сигнал окажется также усиленным в 2 раза. Если было произведено дифференцирование входного сигнала, то и на выходе системы окажется первая разность исходного выходного сигнала. Если входной сигнал прошел процедуру фильтрации, то и выходной сигнал окажется отфильтрованным аналогичным образом. Доказать это достаточно просто на основе свойства ассоциативности.

## 7.2.5. Центральная предельная теорема

*Центральная предельная теорема* играет огромную роль в теории вероятностей, так как именно она позволяет математически доказать возможность использования *гауссова закона распределения вероятностей* для описания большинства процессов, происходящих в природе. Например, амплитуда теплового шума в электрической цепи имеет гауссово распределение; интенсивность лазерного луча в поперечном направлении меняется по тому же закону; и даже точки попаданий при игре в дартс оказываются распределены вокруг «яблочка» по закону Гаусса. В простейшей своей формулировке центральная предельная теорема утверждает: сумма большого числа случайных процессов имеет гауссово распределение вероятностей независимо от того, по какому закону эти процессы распределены.

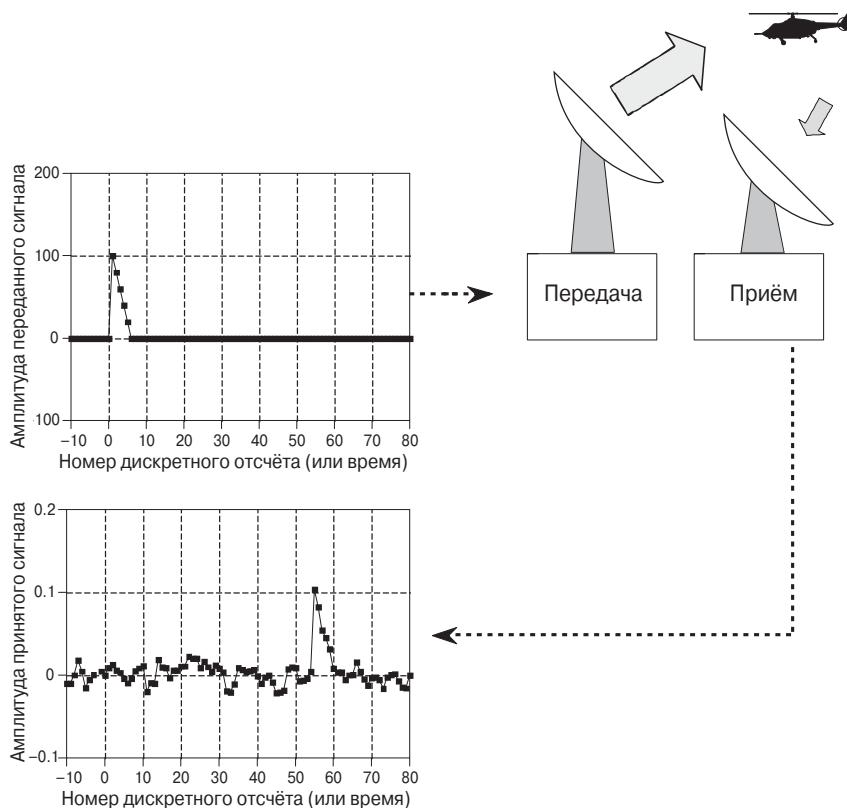


Интересным образом центральная предельная теорема находит отражение применительно к свёртке. Оказывается, если свёртку произвольного импульсного сигнала с самим собой произвести многократно, то результатом будет функция, напоминающая функцию распределения Гаусса. Это проиллюстрировано на Рис. 7.12. На (а) показан импульсный сигнал сложной формы, специально выбранный сильно отличающимся от кривой гауссова распределения. На (б) изображён результат однократной свёртки этого импульса с самим собой, а на (в) — результат выполнения трёх таких процедур свёртки. Хорошо видно, что полученный результат имеет сильное сходство с гауссовой кривой, даже несмотря на такое малое число повторений процедуры свёртки. В математике в таких случаях говорят: процедура имеет высокую скорость сходимости. Ширина полученной таким образом гауссовой кривой (величина  $\sigma$  в выражениях (2.7) и (2.8)) равна ширине исходного импульса (обозначена через  $\sigma$  в выражении (2.7)), умноженной на корень квадратный из числа произведённых процедур свёртки.

## 7.3. Корреляция

Понятие *корреляции* удобнее всего ввести на примере. На Рис. 7.13 изображена радиолокационная станция с её основными элементами. Специальным образом сконструированная антенна излучает в пространство в требуемом направлении короткую электромагнитную волну — радиоимпульс. Если эта волна, распространяясь в пространстве, встречает некоторый объект, например вертолёт, как в рассматриваемом случае, часть энергии волны отражается в обратном направлении и достигает приёмника радиолокатора, в качестве которого обычно выступает та же антенна. Излучаемый сигнал имеет определённую форму, выбираваемую, исходя из ряда требований. В данном примере используется треугольный импульс. Принимаемый отражённый сигнал будет состоять из двух частей: задержанной и уменьшенной по амплитуде копии переданного сигнала и случайного шума, являющегося результатом действия мешающих радиоволн, тепловых шумов электрических цепей приёмника и т. д. Поскольку скорость распространения радиоволн известна — она равна скорости света, — то по задержке отражённого сигнала можно точно найти расстояние до объекта. Возникает задача определения момента появления сигнала известной формы в другом, неизвестном сигнале. Эта задача решается с помощью корреляции.

*Корреляция* — это математическая процедура, имеющая достаточно сильное сходство со *свёрткой*. Так же как и в случае свёртки, корреляция задает правило преобразования двух сигналов в третий. Этот третий сигнал называется *взаимной корреляцией* двух исходных сигналов или *автокорреляцией* (в случае, если сигнал коррелируется с самим собой). Для объяснения процедуры свёртки в предыдущей главе мы специальной схемой проиллюстрировали механизм её реализации. Аналогично поступим с корреляцией. *Механизм реализации корреляции* продемонстрирован на Рис. 7.14. Принимаемый (входной) сигнал  $x[n]$  и сигнал взаимной корреляции  $y[n]$  (выходной) считаем «захваченными» на странице в одном положении. Сигнал  $t[n]$ , который мы хотим обнаружить в принимаемой последовательности (назовём его *отпорный сигнал*), содержится внутри блока корреляции, «перемещающегося» влево-вправо. Для вычисления выходного отсчёта  $y[n]$  необходимо установить блок корреляции в положение, когда его выход указывает на

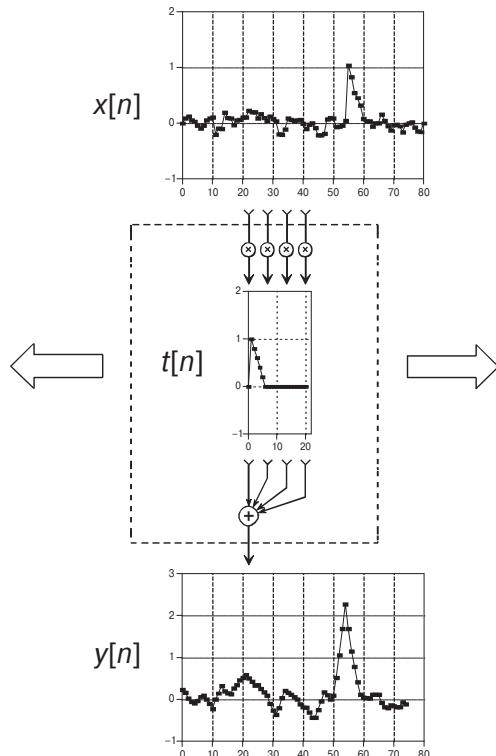


**Рис. 7.13.** Основные элементы радиолокационной станции. Так же как и в других системах эхо-локации, радиолокационная станция излучает в пространство электромагнитную волну, которая отражается от цели и возвращается к радару. Отраженный сигнал является суммой сдвинутой во времени (задержанной) копии излученного сигнала и шума. Проблема обнаружения момента прихода сигнала известной формы на фоне шумов является основополагающей в радиолокации. Решение этой проблемы напрямую связано с корреляцией.

требуемый элемент. Затем производится умножение соответствующих входных отсчётов на элементы опорного сигнала с суммированием получаемых результатов. Значение суммы записывается в выходной массив.

Каждый отсчёт сигнала взаимной корреляции является мерой степени схожести входного и опорного сигналов в этой точке. Это означает, что каждый раз, когда во входной последовательности будет присутствовать фрагмент, совпадающий по форме с опорным сигналом, в сигнале взаимной корреляции будут наблюдаться пиковые значения амплитуды. Когда искомый фрагмент входного сигнала поравняется с опорным, взаимная корреляция достигнет своего максимального значения.

А что, если в опорном сигнале имеются отрицательные значения отсчётов? Ничего не изменится. Представьте себе, что блок корреляции находится в положении, когда опорный сигнал строго выровнен по точно такому же фрагменту входной последовательности. В этом случае в процессе корреляции отсчёты входного сигнала будут умножаться на точно такие же отсчёты опорного (если не учитывать наличие шума). Как положительные, так и отрицательные отсчёты, умно-



**Рис. 7.14.** Механизм реализации корреляции. Показана схема вычисления взаимной корреляции двух сигналов. Сигнал  $y[n]$  является результатом корреляции сигналов  $x[n]$  и  $t[n]$ . Предполагается, что блок корреляции, выделенный пунктирной линией, может перемещаться влево-вправо так, что его выход указывает на текущий вычисляемый отсчёт выходного сигнала. При этом из входной последовательности  $x[n]$  выбираются соответствующие текущему положению блока корреляции отсчёты, и происходит поэлементное умножение этих отсчётов на отсчёты сигнала  $t[n]$  с последующим суммированием результатов. Сам механизм реализации корреляции абсолютно идентичен механизму свёртки (Рис. 6.8 и 6.9), только опорный сигнал в данном случае не «переворачивается». На рисунке показан пример, когда вычисление выходных отсчётов происходит только при условии наличия всех входных элементов (вычисления на краях не производятся).

женные на самих себя, всегда дают результат больше нуля. Таким образом, даже если все отсчёты опорного сигнала будут иметь отрицательные значения, результат корреляции всё равно даст положительное число, максимальное среди результатов корреляции сигналов в других положениях.

Если во входном сигнале присутствует шум, то он окажется и в сигнале взаимной корреляции. Каждый фрагмент сигнала шума будет в той или иной степени походить на опорный сигнал, и это неизбежно. Шум в сигнале взаимной корреляции является мерой такого сходства. Если не принимать во внимание шум, то пик в сигнале взаимной корреляции имеет симметричные правую и левую части. Это будет так, даже если искомый опорный сигнал не является симметричным. Длительность пика равна удвоенной длительности опорного сигнала. Следует не забывать, что процедура взаимной корреляции ставит своей целью обнаружение сигнала, а не его восстановление. То есть не стоит ожидать, что пик в сигнале взаимной корреляции имеет симметричные правую и левую части.

имной корреляции, появляющийся при обнаружении, будет сколько-нибудь напоминать по форме опорный сигнал.

Корреляция является оптимальным методом обнаружения сигнала известной формы на фоне белого шума. То есть пик, появляющийся на выходе процедуры корреляции, будет максимально выше уровня шума по сравнению с любой другой линейной системой обнаружения. Или по-другому: корреляционный приёмник обеспечивает наивысшее *отношение сигнал/шум*. Процедура использования корреляции для обнаружения сигналов известной формы часто называется *согласованной фильтрацией*. Более детально мы познакомимся с этим в Главе 17.

Механизмы реализации свёртки и корреляции являются идентичными, за одним небольшим исключением. Как было сказано в предыдущей главе, при реализации свёртки используется зеркально отражённая копия сигнала внутри блока вычислений. Отсчёты сигнала с номерами 1, 2, 3 и т. д. берутся справа налево. При реализации корреляции этого переворота не требуется, и отсчёты опорного сигнала берутся в нормальном порядке.

Поскольку порядок выборки отсчётов опорного сигнала является единственным различием между механизмами реализации свёртки и корреляции, то существует возможность описания процедуры корреляции с использованием той же математики, что и для свёртки. Всё, что для этого требуется, — это предварительно изменить порядок следования отсчётов опорного сигнала, и тогда при реализации свёртки в действительности будет осуществляться корреляция. Пусть, например,  $c[n]$  является результатом свёртки  $a[n]$  и  $b[n]$ . Математически это записывается в виде уравнения  $c[n] = a[n] * b[n]$ . Тогда взаимная корреляция  $a[n]$  и  $b[n]$  будет записываться так:  $c[n] = a[n] * b[-n]$ . То есть «переворот» сигнала  $b[n]$  достигается путём взятия его аргумента с противоположным знаком  $b[-n]$ .

Мнимая схожесть процедур свёртки и корреляции не должна ввести вас в заблуждение. Это две очень разные с точки зрения цифровой обработки сигналов процедуры. Если свёртка описывает зависимость между входом, выходом и импульсной характеристикой линейной системы, то корреляция — это метод обнаружения сигнала известной формы на фоне шумов. Идентичность математической реализации этих двух процедур — просто совпадение.

## 7.4. Скорость вычислений

Операция *свёртки* в виде программы представляет собой достаточно простой код — всего несколько строк. Однако выполнение этой программы может оказаться весьма затратным. Дело в том, что реализация свёртки требует большого количества операций умножения и сложения. Это существенно увеличивает время вычисления результата. На примерах программ, приведённых нами в предыдущем разделе, можно говорить, что наиболее трудоёмкая процедура при реализации свёртки — это умножение двух чисел с накоплением результата в аккумуляторе. Другие фрагменты кода, такие как, например, изменение индекса массива, выполняются очень быстро. Умножение с накоплением — это один из базовых элементов всех вычислений в ЦОС, встречающийся, как будет показано далее, и во многих других важных алгоритмах. Не секрет, что скорость выполнения умножения с накоплением является одной из основных характеристик *вычислительной производительности* процессоров ЦОС.

Чтобы осуществить свёртку сигнала, состоящего из  $N$  отсчётов, с сигналом, включающим  $M$  отсчётов, необходимо выполнить  $N \times M$  операций умножения с накоплением. Это можно видеть на примерах программ, приведённых в предыдущей главе. Если взять за основу персональные компьютеры середины 90-х годов, то они тратили на выполнение одного умножения с накоплением около микросекунды (в качестве примера выбран процессор Pentium с частотой 100 МГц, использующий формат с плавающей точкой с одинарной точностью (Табл. 4.3)). Таким образом, для реализации свёртки сигнала длиной 10 000 дискретных отсчётов с импульсной характеристикой, включающей 100 отсчётов, необходимо около секунды, а для обработки сигнала из миллиона отсчётов с использованием импульсной характеристики длиной 3000 отсчётов требуется час. Для сравнения, за десять лет до этого (процессор 80286 с частотой 12 МГц) на выполнение этой же задачи потребовалось бы три дня.

Решение проблемы сокращения времени вычислений может идти по трём направлениям. Первое: рассматривайте сигналы на максимально коротких интервалах и используйте арифметику с фиксированной, а не плавающей точкой. Этот вариант будет, пожалуй, оптимальен с точки зрения соотношения качества получаемого результата (скорости выполнения свёртки) и затрат на модификацию программного обеспечения. Он пригоден для использования в случаях, если вам требуется выполнить свёртку всего несколько раз. Если же свёртка должна повторяться многократно, как, например, в системах обработки сигналов, разрабатываемых для коммерческого использования, необходим второй вариант: использование вычислительных устройств, специально ориентированных на ЦОС. Эти специализированные цифровые устройства способны выполнять умножение с накоплением всего за несколько десятков наносекунд<sup>1)</sup>.

Третьим вариантом решения проблемы является использование более эффективного в вычислительном отношении алгоритма реализации свёртки. В Главе 17 мы рассмотрим очень эффективный алгоритм реализации *свёртки в частотной области* с использованием *быстрого преобразования Фурье (БПФ)* — *быструю свёртку*. Результат выполнения быстрой свёртки полностью совпадает с результатом свёртки, выполняемой традиционным методом, описанным в предыдущей главе. Однако время вычислений существенно сокращается. Например, для сигналов длиной в тысячи отсчётов алгоритм быстрой свёртки окажется в сотни раз быстрее. Недостатком метода является сложность программной реализации. Даже если сам алгоритм вам известен и понятен, вы потратите не один час, прежде чем заставите программу правильно работать.

---

<sup>1)</sup> В 2005 году компания Texas Instruments выпустила процессор цифровой обработки сигналов с частотой 1 ГГц, выполняющий одну 16-битную операцию умножения с накоплением в формате с фиксированной точкой за 1/8 наносекунды. — Примеч. пер.

## ДИСКРЕТНОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ

Анализ Фурье — это семейство математических методов, в основе которых лежит разложение сигналов в ряд тригонометрических функций. Дискретное преобразование Фурье (ДПФ) входит в это семейство и используется при работе с цифровыми сигналами. Данная глава является первой из четырёх, посвящённых действительному ДПФ — дискретному преобразованию Фурье, использующему представление сигналов в действительной форме (действительными числами). Комплексное ДПФ, работающее с сигналами в комплексной форме, является более мощным аппаратом. О нём пойдёт речь в Главе 31. В этой главе мы рассмотрим математические основы и методику разложения в ряд Фурье, что является основой ДПФ.

### 8.1. Преобразование Фурье

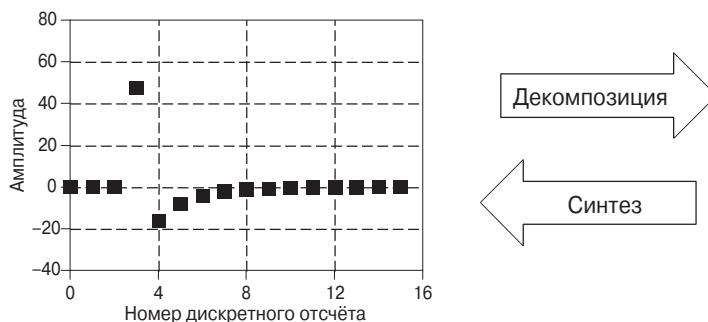
Анализ Фурье своим названием обязан французскому математику и физику Жану Батисту Жозефу Фурье (1768—1830). Фурье внёс весомый вклад в развитие современной науки и снискал огромное признание за свои математические открытия и предвидение того, как значимы они окажутся в практическом использовании. Фурье исследовал вопросы теплопроводности и в 1807 г. представил в Институте Франции (*Institut de France*) свою работу по использованию тригонометрических рядов для представления уравнений теплопроводности. В работе высказывалось вызвавшее тогда много споров утверждение о том, что любой непрерывный периодический сигнал может быть представлен суммой гармонических сигналов с правильно подобранными параметрами. Среди рецензентов было два именитых математика: Жозеф Луи Лагранж (1736—1813) и Пьер Симон Лаплас (1749—1827).

Лаплас и другие рецензенты были за опубликование статьи. Однако Лагранж решительно протестовал. Около 50 лет Лагранж утверждал, что такой подход не применим для представления сигналов, рассматривая при этом сигналы со скачкообразным изменением формы. К таким сигналам, например, относятся прямоугольные колебания. Институт Франции «склонил голову» перед именем Лагранжа и отклонил работу Фурье. Только спустя 15 лет после смерти Лагранжа работа всё-таки была опубликована. К счастью, Фурье не сидел всё это время сложа руки. Он принимал активное участие в политических событиях, участвовал в экспедициях в Египет, организованных Наполеоном, а после Французской революции пытался избежать гильотины (без преувеличения!).

Но кто же всё-таки был прав? Ответ неоднозначен. Лагранж не ошибался, утверждая, что суммой гармонических сигналов невозможно представить сигнал со скачкообразным изменением формы. Но можно получить очень близкое пред-

ставление, настолько близкое, что разностный сигнал окажется сигналом с нулевой энергией. С этой точки зрения Фурье прав в своём утверждении, но в науке 18-го столетия концепция энергии была ещё слабо развита. Сейчас это явление известно как *эффект Гиббса*. О нём пойдет речь в Главе 11.

**Рис. 8.1** иллюстрирует разложение сигнала в тригонометрический ряд синусоид и косинусоид. Исходный сигнал, включающий 16 отсчётов — с 0-го по 15-й, показан на **(а)**. Разложение Фурье этого сигнала представлено набором 9 синусоидальных и 9 косинусоидальных сигналов, каждый со своей амплитудой и частотой **(б)**. Это пока далеко не очевидно, но сумма представленных 18 гармонических сигналов даёт исходное колебание, изображённое на **(а)**. Следует отметить, что протест Лагранжа справедлив исключительно для непрерывных сигналов. Для сигналов дискретных такое разложение является математически точным. Нет никакой разницы между исходным сигналом **(а)** и суммой сигналов разложения **(б)**, как нет разницы между числом 7 и суммой  $3 + 4$ .

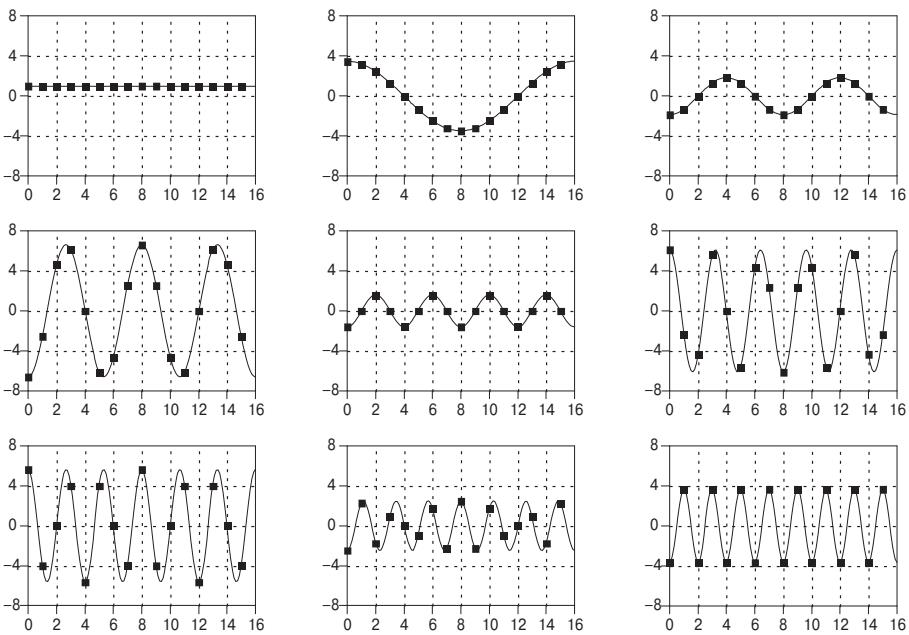


**Рис. 8.1а.** Исходный сигнал, включающий 16 отсчётов  
(см. продолжение на следующей странице).

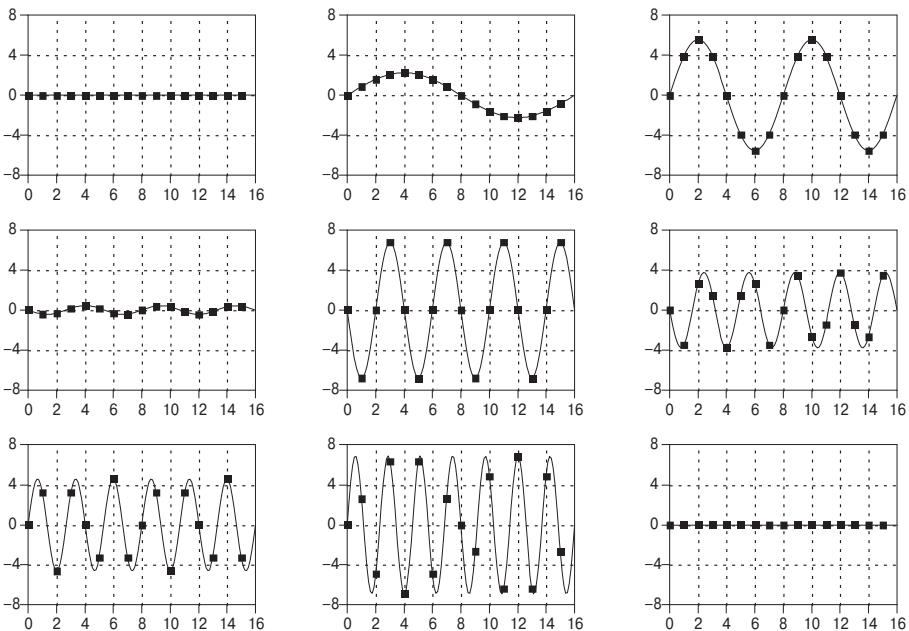
Почему в основу ряда положены именно гармонические колебания, а скажем, не прямоугольные или пилообразные? Вспомним, что способов *декомпозиции* сигнала существует бесконечное множество. Среди них надо выбрать тот, который приводит к переходу от сложного исходного сигнала к набору компонент, с которыми проще работать. Например, импульсная декомпозиция даёт возможность по отсчёту анализа сигнала, что в конечном счёте приводит к фундаментальному методу свёртки. Гармонические сигналы удобно использовать в качестве компонентов разложения, потому что они обладают свойством, которым не обладают исходные сигналы: они являются *собственными функциями линейных систем*. Как уже было сказано в Главе 5, гармонический сигнал на входе линейной системы всегда приводит к появлению гармонического сигнала на её выходе. Меняются только амплитуда и фаза колебания, а частота и форма волны остаются неизменными. Эта важная особенность характерна исключительно для гармонических сигналов. Разложение с помощью прямоугольных или пилообразных колебаний возможно, но нет никаких оснований полагать, что оно окажется сколько-нибудь целесообразным.

Общее понятие *преобразования Фурье* охватывает 4 класса преобразований, выделяемых в соответствии с типами сигналов, которыми они оперируют. Сигна-

### Косинусоиды



### Синусоиды



**Рис. 8.16.** Пример разложения в ряд Фурье. Исходный сигнал, включающий 16 отсчётов (Рис. 8.1а на предыдущей странице), раскладывается на 9 косинусоидальных и 9 синусоидальных сигналов. Частота каждого колебания постоянна для любых исходных сигналов, в зависимости от формы сигнала меняется только амплитуда компонентов разложения.

лы могут быть *непрерывными и дискретными, периодическими и апериодическими*. Различные сочетания этих характеристик дают 4 класса сигналов, которые описаны ниже и проиллюстрированы на Рис. 8.2.

#### **Апериодические непрерывные сигналы.**

К этой категории сигналов относятся, например, убывающая экспонента и кривая Гаусса. Они уходят в бесконечность и со стороны положительных, и со стороны отрицательных аргументов и при этом не имеют периодически повторяющихся фрагментов. Преобразование Фурье для таких сигналов не носит какого-либо специального наименования, а так и называется: *преобразование Фурье*.

#### **Периодические непрерывные сигналы.**

Примерами сигналов, попадающих в данную категорию, могут служить: синусоида, меандр, а также любые другие сигналы, формируемые периодическим повторением одного и того же фрагмента и простирающиеся от минус бесконечности до плюс бесконечности. Когда речь идёт о сигналах такого рода говорят не о преобразовании Фурье, а о разложении в ряд Фурье.

#### **Апериодические дискретные сигналы.**

Сигналы этого типа определены только в дискретные моменты времени на всей временной оси от минус до плюс бесконечности и не имеют периодических повторений. О преобразовании Фурье для таких сигналов говорят: *преобразование Фурье дискретного времени*.

#### **Периодические дискретные сигналы.**

Это дискретные сигналы, для которых характерно периодическое повторение формы на всей временной оси от минус до плюс бесконечности. В данном случае преобразование Фурье иногда называют рядом Фурье, но более распространённым термином является *дискретное преобразование Фурье*.

Вы можете сказать, что названия этих четырёх типов преобразований Фурье не имеют никакой логической организации и в них легко запутаться. Это действительно так. Термины появились около 200 лет назад, и в их формировании не было никакой системы. Но мы ничего не можем с этим поделать. Нужно просто их запомнить и двигаться дальше.

О всех перечисленных классах сигналов мы говорили, что во времени они продолжаются от минус до плюс бесконечности. «Погодите-ка! — скажете вы. — А что, если у нас есть только конечное число отсчётов сигнала, хранящихся в массиве данных в памяти компьютера, например сигнал из 1024 отсчётов? Разве нет преобразования Фурье для сигналов конечной длины?» Нет, такого преобразования не существует. Синусоида и косинусоида определены на бесконечном интервале от минус до плюс бесконечности. Невозможно использовать сигналы бесконечной длины для формирования сигнала с ограниченным числом отсчётов. Однако есть выход: можно искусственно рассматривать сигнал, состоящий из конечного числа отсчётов, как сигнал бесконечной длины. Для этого достаточно мысленно дополнить сигнал конечной длины воображаемыми отсчётами слева и справа до минус и плюс бесконечности соответственно. Если в качестве этих дополнительных отсчётов выбрать все нули, то мы получим дискретный апериодический сигнал, для которого применимо преобразование Фурье дискретного времени. Другим вариантом формирования сигнала слева и справа от имеющегося фрагмента является использование бесконечного числа его копий (1024 отсчётов в данном случае). При этом мы приходим к дискретному периодическому сигна-

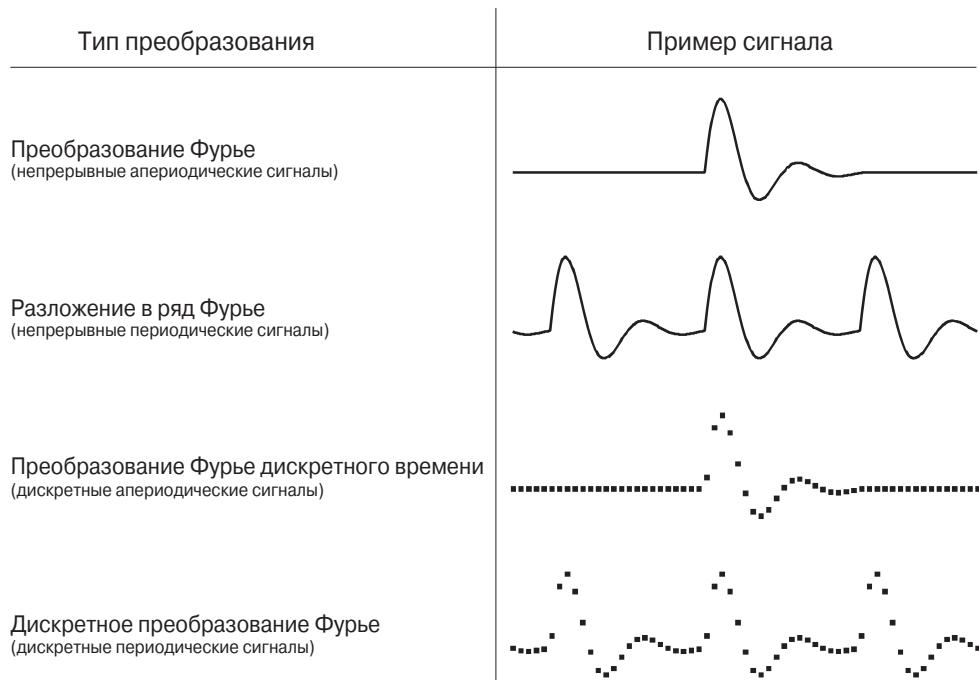


Рис. 8.2. Четыре типа преобразование Фурье.

лу с периодом, равным 1024 дискретным отсчётом, и можем использовать дискретное преобразование Фурье.

Чтобы получить разложение апериодического сигнала, требуется бесконечное число синусных и косинусных компонентов. Это делает невозможным выполнение преобразования Фурье дискретного времени на компьютере. Таким образом, единственным типом Фурье-преобразования, применяемым в ЦОС, является дискретное преобразование Фурье. Другими словами, вычислительные устройства способны работать только с дискретными сигналами конечной длины. Если вы решаете какую-то научную теоретическую задачу, или выполняете студенческое домашнее задание, или просто размышляете о тайнах математики, вам, возможно, будет интересно использовать первые три типа преобразования Фурье. Но когда вы садитесь за компьютер, вы можете использовать только ДПФ. С первыми тремя категориями преобразования мы кратко познакомимся в последующих главах. Сейчас давайте сконцентрируемся на дискретном преобразовании Фурье.

Вернёмся к Рис. 8.1, на котором приведён пример декомпозиции Фурье. Не вдаваясь в подробности, мы видим, что 16-точечный сигнал представляется в виде совокупности 18 гармонических сигналов, каждый из которых состоит из 16 отсчётов. Однако при строгом подходе мы должны рассматривать 16-точечный сигнал, показанный на (а), как один период сигнала бесконечной длины. Аналогично каждая из показанных 16-точечных синусоид (косинусоид), приведенных на (б), — это только фрагмент бесконечного гармонического колебания. Возникает вопрос: есть ли разница между тем, рассматриваем мы 16-точечный сигнал как формируемый из набора 16-точечных синусоид (косинусоид) или как бесконечный периодический сигнал, получаемый из совокупности гармонических колеба-

ний бесконечной длины? Ответ: обычно нет, но иногда разница существует. В следующих главах мы рассмотрим свойства ДПФ, которые кажутся непонятными, если считать сигналы конечными, и становятся абсолютно ясны, если учесть бесконечное периодическое повторение. Несмотря на то что рассмотрение сигнала как периодического обычно не имеет смысла с позиции действительной природы сигнала и его регистрации системой обработки, оно, однако, оказывается необходимым, чтобы использовать математический аппарат дискретного преобразования Фурье.

В каждом из четырёх перечисленных типов преобразования Фурье, в свою очередь, можно выделить преобразование *действительное и комплексное*. Действительное преобразование Фурье проще — оно работает с обычными числами и обычной арифметикой декомпозиции и синтеза. Например, на Рис. 8.1 показано именно *действительное ДПФ*. Комплексное преобразование Фурье существенно сложнее. Оно имеет дело с комплексными числами. Это такие числа, как, например,  $3 + 4j$ , где  $j$  равно  $\sqrt{-1}$  (в электротехнике для записи комплексных чисел используют символ  $j$ , в то время как в математике обычно применяют символ  $i$ ). Математика комплексных чисел может быстро утомить даже того, кто специализируется на ЦОС. Наша же книга рассчитана на широкую аудиторию инженеров и научных работников и ставит своей целью описание фундаментальных основ ЦОС, не затрудняя восприятие математическими сложностями. Комплексное преобразование Фурье — это прерогатива специалистов ЦОС, желающих по горло погрузиться в трясину математики. Если вы относите себя к этой категории, для вас предназначены Главы 30...33.

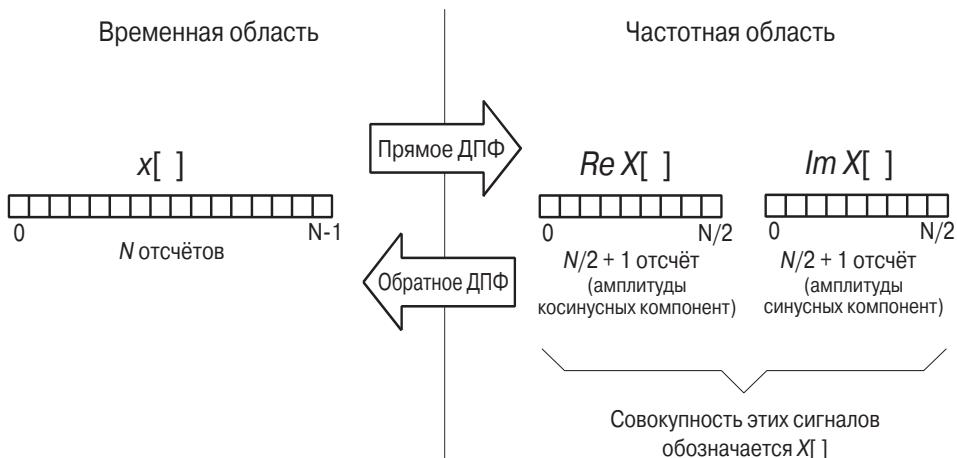
Понятие «*преобразование*» является математическим. Оно очень широко используется в теории ЦОС, например: *преобразование Фурье, преобразование Лапласа, Z-преобразование, преобразование Гильберта, дискретное косинусное преобразование* и т. д. Но что такое преобразование? Чтобы ответить на этот вопрос, давайте сначала вспомним, что такое функция. Функция — это алгоритм или процедура, которая переводит значения одной величины в значения другой величины. Например,  $y = 2x + 1$  — это функция. Вы задаёте каким-либо значением  $x$ , подставляете его в приведённую формулу, и она выдаёт вам значение  $y$ . Есть также функции, которые переводят несколько значений разных величин в одно значение функции, например:  $y = 2a + 3b + 4c$ . Здесь значения величин  $a$ ,  $b$  и  $c$  переводятся в значение переменной  $y$ .

Понятие преобразования является дальнейшим развитием этих рассуждений, позволяющим использовать несколько величин и в качестве входных, и в качестве выходных переменных. Предположим, вы имеете дело с сигналом, представленным 100 дискретными отсчётыми. Если вы сформируете какое-либо уравнение, алгоритм или процедуру, переводящую эти отсчёты в другие 100 отсчётов, вы таким образом получите преобразование. Если при этом у вас есть основания полагать, что это преобразование является полезным, вы имеете все основания добавить к нему свою фамилию и идти разъяснять его полезность коллегам (ещё лучше, если при этом вы являетесь известным французским математиком XVIII века). Само понятие преобразования не ограничено только определённым типом или количеством данных. Например, возможно преобразование 100 входных дискретных отсчётов в 200 выходных дискретных отсчётов или преобразование непрерывного входного сигнала в непрерывный выходной сигнал. Смешан-

ное использование сигналов — дискретный вход и непрерывный выход или наоборот — также возможно. Таким образом, преобразование — это любая чётко определённая процедура, переводящая один набор данных в другой. Давайте посмотрим, что это означает применительно к нашей теме — дискретному преобразованию Фурье.

## 8.2. Действительное ДПФ: терминология и обозначения

Как показано на Рис. 8.3, дискретное преобразование Фурье переводит  $N$ -точечный входной сигнал в два выходных сигнала, каждый длиной  $N/2 + 1$  дискретных отсчёта. Сигнал на входе — это сигнал, подлежащий декомпозиции. Два сигнала на выходе сформированы значениями амплитуд косинусных и синусных компонентов декомпозиции (как они получаются, мы рассмотрим вкратце немного позже). О входном сигнале говорят, что он представлен *во временной области*, если его описание выражается функцией времени. Дискретное преобразование Фурье применяется обычно именно к таким сигналам, хотя в действительности любой набор дискретных данных может быть подвержен ДПФ независимо от способа регистрации этих данных. Когда вы встречаете термин «временная область», относящийся к Фурье-анализу, то это может действительно указывать на сигнал, отсчёты которого взяты в дискретные моменты времени, но также может и подразумевать любой другой дискретный сигнал, подлежащий декомпозиции.



**Рис. 8.3.** Терминология ДПФ. Во временной области сигнал  $x[ ]$  представлен  $N$  дискретными отсчётаами с 0-го по  $(N - 1)$ -й. Дискретное преобразование Фурье переводит сигнал в частотную область, где формирует два сигнала: действительную часть  $Re X[ ]$  и мнимую часть  $Im X[ ]$ . Каждый из этих сигналов, определённых в частотной области, имеет длину  $N/2 + 1$  отсчётов, нумеруемых с 0-го по  $N/2$ . ДПФ, переводящее сигналы из временной области в частотную, называется *прямым*, а ДПФ, переводящее сигнал из частотной области во временную, называется *обратным*. (Имейте в виду: на рисунке представлено *действительное ДПФ*. *Комплексное ДПФ*, обсуждаемое в Главе 31, преобразует комплексный  $N$ -точечный сигнал в другой комплексный  $N$ -точечный сигнал.)

Термин «частотная область» используется для обозначения набора амплитуд синусных и косинусных компонентов Фурье-декомпозиции (с учётом небольшой модификации, о которой мы уже обещали поговорить чуть позже).

Представление сигнала в частотной области содержит абсолютно ту же информацию, что и представление во временной области, но просто в иной форме. Если известна одна из этих форм сигнала, всегда можно вычислить другую. При известном представлении во временной области переход в частотную область называется *декомпозицией, анализом, прямым ДПФ* или просто *ДПФ*. Наоборот, переход к временному представлению сигнала при известном частотном представлении называется *синтезом или обратным ДПФ*. Обе процедуры анализа и синтеза могут быть выражены в форме уравнения или алгоритма для компьютерных вычислений.

Обычно число отсчётов сигнала, представленного во временной области, обозначается через  $N$ . Величина  $N$  может принимать любые целые положительные значения, но чаще её стараются брать равной степени двух, т. е., например, 128, 256, 512, 1024 и т. д. Это происходит по двум причинам. Во-первых, цифровые данные хранятся в памяти компьютеров по двоичным адресам, и использование в качестве размера массива степени двух оказывается наиболее естественным. Во-вторых, наиболее эффективным вариантом вычисления ДПФ является алгоритм *быстрого преобразования Фурье (БПФ)*, а он обычно оперирует с массивами, длина которых равна степени двух. Чаще  $N$  принимает значение в диапазоне 32...4096. Отсчёты в большинстве случаев нумеруются не с 1 до  $N$ , а с 0 до  $N - 1$ .

Сигналы, представленные во временной области, принято обозначать строчными буквами, например:  $x[ ]$ ,  $y[ ]$  или  $z[ ]$ . Те же обозначения, выполненные заглавными буквами, относятся к представлению тех же сигналов в частотной области:  $X[ ]$ ,  $Y[ ]$  и  $Z[ ]$ . Пусть, например, имеется  $N$ -точечный сигнал  $x[ ]$ , представленный во временной области. Его частотное представление будет обозначаться  $X[ ]$  и включать две составляющие, каждая длиной ( $N/2 + 1$ ) отсчёт: *действительную часть*  $X[ ]$ , обозначаемую  $Re X[ ]$ , и *мнимую часть*  $X[ ]$ , обозначаемую  $Im X[ ]$ . Действительная часть  $Re X[ ]$  содержит значения амплитуд косинусных сигналов разложения, а мнимая часть  $Im X[ ]$  — значения амплитуд синусных сигналов разложения (мы пока не учитываем нормировку, о которой пойдёт речь совсем скоро). Точно так же, как сигнал во временной области представлен отсчётами  $x[0]...x[N - 1]$ , сигнал в частотной области включает отсчёты  $Re X[0]...Re X[N/2]$  и  $Im X[0]...Im X[N/2]$ . Рекомендуем хорошо запомнить эти обозначения. Это необходимо, чтобы понимать формулы цифровой обработки сигналов. К сожалению, некоторые языки программирования не различают строчные и прописные буквы, заставляя программиста самостоятельно решать, какие правила использовать для обозначения переменных. В нашей книге в текстах программ мы будем использовать массив  $XX[ ]$  для размещения отсчётов сигнала, представленного во временной области, а для частотного представления выделим массивы  $REX[ ]$  и  $IMX[ ]$ .

Понятия действительной и мнимой частей взяты из теории комплексного ДПФ, где они используются для выделения действительной и мнимой частей комплексных чисел. Но для действительного ДПФ лишние сложности ни к чему. Пока вы не дошли до Главы 31, понимайте «действительную часть» просто как совокупность амплитуд косинусных компонентов разложения, а «мнимую часть» — как совокупность амплитуд синусных компонентов разложения. Пусть эти термины не сбивают вас с толку: пока мы используем только обычные действительные числа.

Пусть также вас не вводит в заблуждение понятие длины сигнала, используемое применительно к представлению в частотной области. В литературе по цифровой обработке сигналов очень часто можно встретить такие выражения, как, например: «ДПФ переводит  $N$ -точечный сигнал во временной области в  $N$ -точечный сигнал в частотной области». Эти выражения относятся к комплексному ДПФ, при котором каждый отсчёт сигнала представляется комплексным числом, состоящим из действительной и мнимой частей. Сейчас мы говорим о действительном ДПФ, а сложная математика начнется уже достаточно скоро.

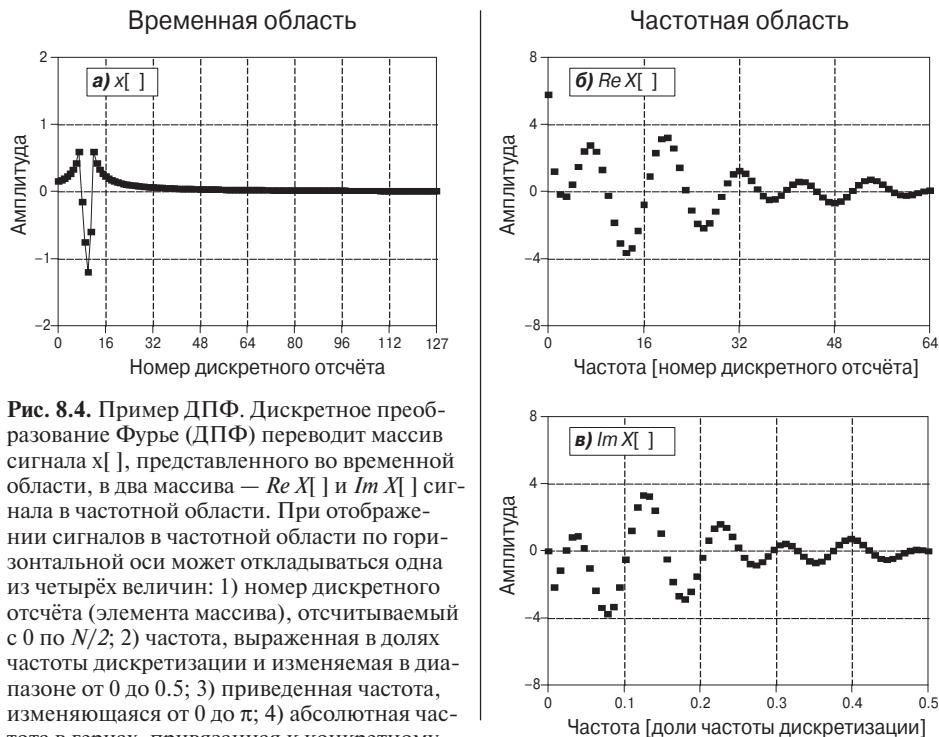
### 8.3. Независимая переменная при описании в частотной области

На Рис. 8.4 показан пример ДПФ сигнала длиной  $N = 128$  отсчётов. Во временной области сигнал представлен отсчётами  $x[0]...x[127]$  (массив чисел). Представление в частотной области требует использования двух массивов  $Re X[0]...Re X[64]$  и  $Im X[0]...Im X[64]$ . Обратите внимание на то, что 128 отсчётов во временной области переходят в 65 отсчётов в каждом из двух массивов сигнала в частотной области, нумеруемых с 0-го по 64-й. То есть  $N$ -точечный сигнал во временной области заменяется в частотной двумя  $(N/2 + 1)$ -точечными сигналами (а не  $N/2$ ). Потеря этого дополнительного отсчёта является общераспространённой ошибкой при написании программных реализаций ДПФ.

При графическом отображении частотного представления сигнала в качестве независимой переменной, откладываемой по оси абсцисс, может выступать одна из четырёх величин. Все четыре варианта употребляемого аргумента являются широко применяемыми в ЦОС. Первая величина — это номер отсчёта сигнала в частотной области, т. е.  $0...N/2$  (в нашем примере  $0...64$ ). При таком подходе аргумент функции, описывающей сигнал в частотной области, представлен целыми числами:  $Re X[k]$ ,  $Im X[k]$ , где  $k$  изменяется от 0 до  $N/2$  с шагом 1. Этот вариант записи аргумента удобен для программистов, потому что они непосредственно в такой форме употребляют индексацию массивов сигнала при разработке кода программы ДПФ. Иллюстрация такого подхода показана на (б).

Второй метод, проиллюстрированный на (в), измеряет аргумент частотного представления сигнала в долях от *частоты дискретизации*. При этом значение независимой переменной, откладываемой по оси абсцисс, изменяется от 0 до 0,5, так как цифровой сигнал может содержать только частоты от нуля до половины частоты дискретизации. Обозначается аргумент символом  $f$ , что значит «частота» (*frequency*). Действительная и мнимая части сигнала записываются соответственно как  $Re X[f]$  и  $Im X[f]$ , где  $f$  принимает  $N/2 + 1$  различных значений, равномерно распределённых на интервале  $[0; 0.5]$ . Чтобы перейти от первого варианта аргумента  $k$  ко второму  $f$ , достаточно выполнить деление на  $N$ , т. е.  $f = k/N$ . В данной книге большинство графиков использует именно этот второй аргумент, как бы подчёркивая, что в дискретном сигнале содержатся лишь частоты от нуля до половины частоты дискретизации.

Третий вариант аргумента имеет сходство со вторым, отличаясь только домножением на  $2\pi$ . Обозначается он символом  $\omega$  — строчной греческой буквой *омега*. В этом случае действительная и мнимая части частотного представления



**Рис. 8.4.** Пример ДПФ. Дискретное преобразование Фурье (ДПФ) переводит массив сигнала  $x[n]$ , представленного во временной области, в два массива —  $\operatorname{Re} X[\omega]$  и  $\operatorname{Im} X[\omega]$  сигнала в частотной области. При отображении сигналов в частотной области по горизонтальной оси может откладываться одна из четырёх величин: 1) номер дискретного отсчёта (элемента массива), отсчитываемый с 0 по  $N/2$ ; 2) частота, выраженная в долях частоты дискретизации и изменяющаяся в диапазоне от 0 до 0.5; 3) приведенная частота, изменяющаяся от 0 до  $\pi$ ; 4) абсолютная частота в герцах, привязанная к конкретному значению частоты дискретизации. В приведенном примере (б) иллюстрирует использование первого метода, а (в) — второго.

записываются как  $\operatorname{Re} X[\omega]$  и  $\operatorname{Im} X[\omega]$ , где  $\omega$  принимает  $N/2 + 1$  различных значений, равномерно распределённых на интервале  $[0; \pi]$ . Параметр  $\omega$  именуется *приведённой круговой частотой* и измеряется в радианах. Идея использования этого аргумента основывается на том, что при движении по кругу угол изменяется от 0 до  $2\pi$  радиан. Данный тип аргумента очень удобно использовать в математических выражениях, которые становятся при этом компактнее. Например, сравните варианты записи одной и той же формулы с использованием всех трёх перечисленных аргументов:

- при использовании  $k$ :  $c[n] = \cos(2\pi kn/N)$ ;
- при использовании  $f$ :  $c[n] = \cos(2\pi fn)$ ;
- при использовании  $\omega$ :  $c[n] = \cos(\omega n)$ .

Четвёртый вариант представления независимой переменной использует в качестве аргумента абсолютные значения частот, характерные для конкретных задач. Например, если система использует частоту дискретизации 10 кГц (10 000 дискретных отсчётов в секунду), то при графическом отображении её частотных характеристик по оси абсцисс откладываются частоты от 0 до 5 кГц. Достоинством такого подхода является то, что используемые значения аргумента отражают действительные частоты реальных сигналов. Недостаток состоит в жёсткой привязке к конкретной частоте дискретизации, что не позволяет использовать данный метод для проектирования ЦОС-систем в общем, например для проектирования цифровых фильтров.

Все четыре описанных варианта независимой переменной широко используются в ЦОС, и вам следует знать, что они означают и как выполняется переход от одного из них к другому. При этом необходимо уметь пользоваться и графической формой отображения сигналов, и уравнениями. Чтобы понять, какой из вариантов используется в конкретной ситуации, нужно обратить внимание на запись независимой переменной и на диапазон её изменения. Здесь вы можете найти один из четырёх вариантов: аргумент  $k$  (или любой другой аргумент, принимающий только целые значения), лежащий в диапазоне от 0 до  $N/2$ ; аргумент  $f$ , принимающий значения от 0 до 0.5; аргумент  $\omega$ , изменяющийся от 0 до  $\pi$ ; и частота в герцах, принимающая значения от 0 Гц до половины конкретной используемой в системе частоты дискретизации.

## 8.4. Базисные функции ДПФ

Синусные и косинусные функции, выступающие в качестве компонентов *Фурье-декомпозиции*, называют *базисными функциями ДПФ*. Результат преобразования ДПФ представляет собой набор чисел-амплитуд, а базисные функции разложения — совокупность синусных и косинусных сигналов единичной амплитуды. Если каждому значению амплитуды (отсчёту сигнала в частотной области) поставить в соответствие по определённому правилу синусный или косинусный сигнал (базисную функцию), получим набор синусных и косинусных компонентов разной амплитуды, которые в сумме дадут нам исходный сигнал, представленный во временной области.

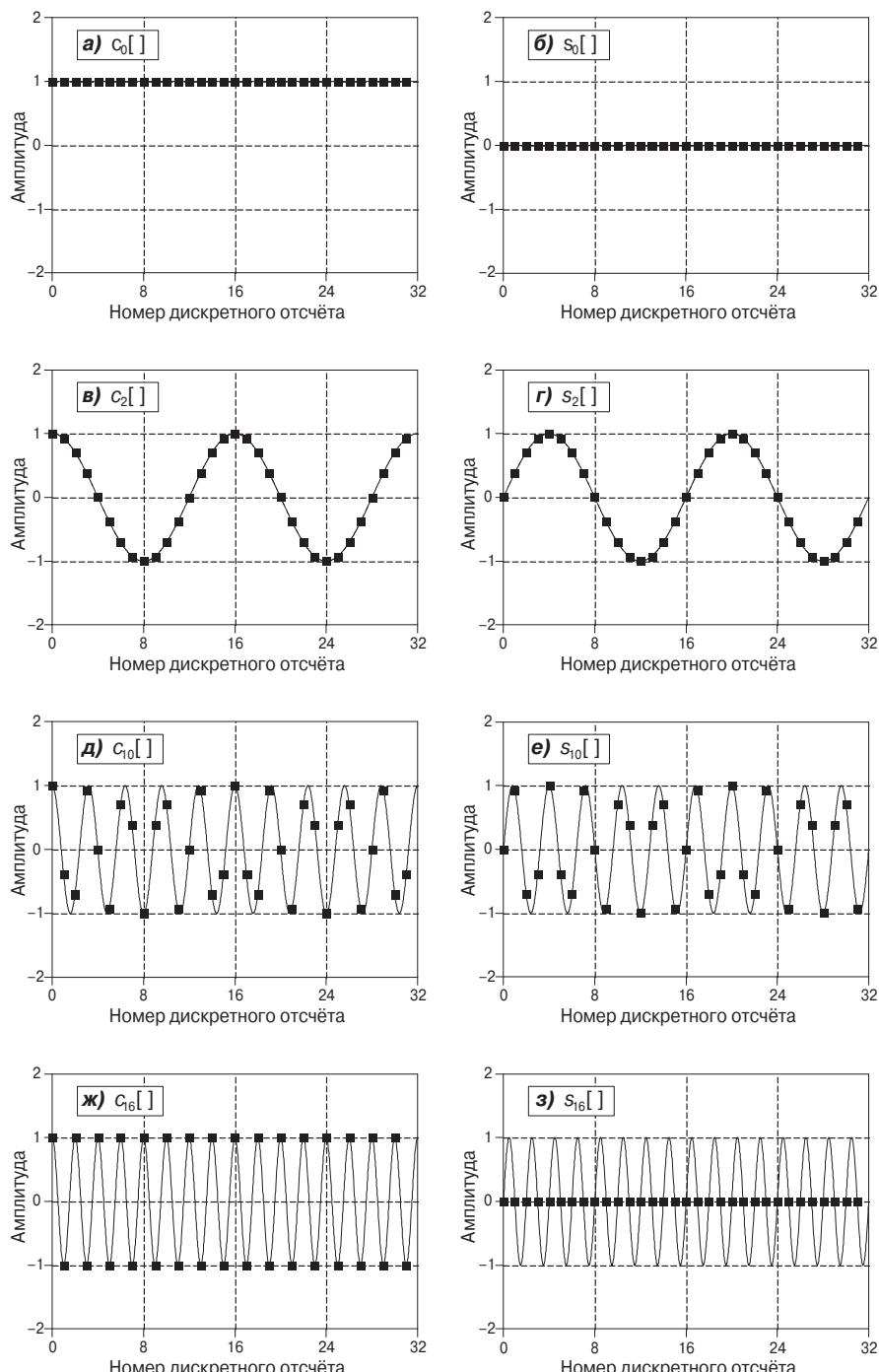
Базисные функции ДПФ формируются на основе уравнений

$$\begin{aligned} c_k[i] &= \cos(2\pi ki/N) \\ \text{и} \quad s_k[i] &= \sin(2\pi ki/N). \end{aligned} \tag{8.1}$$

Выражения для базисных функций ДПФ. В этих формулах  $c_k[i]$  и  $s_k[i]$  — это соответственно косинусный и синусный сигналы длиной  $N$  дискретных отсчётов, нумеруемых с 0-го по  $(N - 1)$ -й. Параметр  $k$  определяет частоту гармонического сигнала. В случае  $N$ -точечного ДПФ  $k$  принимает значения от 0 до  $N/2$ .

$c_k[ ]$  — это косинусный сигнал, амплитуда которого равна  $\operatorname{Re} X[k]$ ;  $s_k[ ]$  — синусный сигнал с амплитудой, равной  $\operatorname{Im} X[k]$ . Например, на Рис. 8.5 показаны некоторые из 17 косинусных и 17 синусных сигналов, являющихся компонентами декомпозиции 32-точечного ДПФ ( $N = 32$ ). Так как исходный сигнал во временной области равен сумме всех компонентов разложения, то все косинусные и синусные сигналы должны иметь одинаковую длину, равную длине исходного сигнала. В нашем примере это 32 дискретных отсчёта, нумеруемых с 0-го по 31-й. Параметр  $k$  определяет частоту косинусоиды или синусоиды. Так, запись  $c_1[ ]$  описывает косинусный сигнал, фаза которого делает один полный оборот за  $N$  периодов дискретизации;  $c_5[ ]$  описывает косинусный сигнал, фаза которого проходит пять полных оборотов за  $N$  периодов дискретизации, и т. д. Это очень важный момент для понимания базисных функций: частотный параметр  $k$  равен числу полных оборотов фазы за  $N$  периодов дискретизации, что соответствует длине сигнала.

Рассмотрим некоторые из этих базисных функций более детально. На (а) показан косинусный сигнал  $c_0[ ]$ . Он имеет нулевую частоту и представляет собой



**Рис. 8.5.** Базисные функции ДПФ. Набор базисных функций 32-точечного ДПФ состоит из 17 косинусных и 17 синусных дискретных сигналов. Восемь из них показаны на рисунке. Все сигналы дискретные; они изображены непрерывной линией только для того, чтобы лучше читались графики.

постоянный сигнал — константу, равную 1. Это означает, что значение  $\operatorname{Re} X[0]$  равно среднему значению всех отсчётов сигнала во временной области. В терминах электроники можно сказать, что  $\operatorname{Re} X[0]$  содержит *смещение по постоянному току*. Синусный сигнал нулевой частоты  $s_0[ ]$  показан на (б). Все его отсчёты равны нулю, и на результат синтеза он не влияет. Следовательно, значение  $\operatorname{Im} X[0]$  не играет никакой роли и поэтому всегда приравнивается нулю. Но об этом чуть позже.

На (в) и (г) показаны косинусный и синусный сигналы  $c_2[ ]$  и  $s_2[ ]$ , имеющие два периода на длине сигнала из  $N$  отсчётов. Этим сигналам в частотной области соответствуют отсчёты  $\operatorname{Re} X[2]$  и  $\operatorname{Im} X[2]$ . Аналогично на (д) и (е) показаны косинусный и синусный сигналы  $c_{10}[ ]$  и  $s_{10}[ ]$ , имеющие десять периодов на длине сигнала из  $N$  отсчётов. Им в частотной области будут соответствовать отсчёты  $\operatorname{Re} X[10]$  и  $\operatorname{Im} X[10]$ . Заметим, что дискретные сигналы на (д) и (е) по своей форме мало напоминают косинусоиду и синусоиду. Если бы отдельные отсчёты сигналов не были соединены непрерывными кривыми, вам бы пришлось немало потрудиться, чтобы просто установить форму этих колебаний. Это, конечно, немного неудобно, но беспокоиться в данном случае не следует. Гармоническая форма дискретных сигналов является математически обусловленной, несмотря на то, что человеческий глаз этот факт детектирует не в состоянии.

Базисные функции с наивысшей частотой показаны на (ж) и (з). Это сигналы  $c_{N/2}[ ]$  и  $s_{N/2}[ ]$  или в нашем примере:  $c_{16}[ ]$  и  $s_{16}[ ]$ . Дискретный косинусный сигнал колебляется между двумя принимаемыми им значениями: 1 и  $-1$ , что может интерпретироваться как дискретизация гармонического колебания с выборкой пиковых значений. Дискретный синусный сигнал, напротив, содержит только нулевые отсчёты, т. е. моменты выборки совпадают с точками пересечения нуля. Это делает частотный отсчёт  $\operatorname{Im} X[N/2]$  равным отсчёту  $\operatorname{Im} X[0]$  и равным нулю. Этот сигнал также не оказывает влияния на результат синтеза.

Возникает интересный вопрос: если на вход ДПФ поступают  $N$  отсчётов, а на выходе ДПФ получается  $N + 2$  отсчётов, то откуда берётся дополнительная информация в виде этих двух добавленных элементов? Ответ состоит в том, что два из выходных отсчётов ДПФ не несут информации. Как вы уже догадались, это всегда равные нулю отсчёты  $\operatorname{Im} X[0]$  и  $\operatorname{Im} X[N/2]$ .

## 8.5. Синтез сигнала с помощью обратного ДПФ

Обобщив всё выше сказанное, мы можем теперь записать *уравнение синтеза* сигнала в следующем виде:

$$x[i] = \sum_{k=0}^{N/2} \operatorname{Re} \bar{X}[k] \cos(2\pi k i / N) + \sum_{k=0}^{N/2} \operatorname{Im} \bar{X}[k] \sin(2\pi k i / N). \quad (8.2)$$

Уравнение синтеза. В этом уравнении  $x[i]$  — это синтезируемый сигнал, представленный в виде массива элементов с индексом  $i$ , принимающим значения от 0 до  $N - 1$ .  $\operatorname{Re} \bar{X}[k]$  и  $\operatorname{Im} \bar{X}[k]$  — это массивы амплитуд соответственно косинусных и синусных компонентов с индексом  $k$ , меняющимся от 0 до  $N/2$ . Выражение (8.3) описывает уравнения нормализации, дополняющие выражение (8.2) при его использовании для обратного ДПФ.

На словах это выражение означает следующее. Любой сигнал  $x[i]$  длительностью  $N$  дискретных отсчётов может быть представлен суммой косинусных сигналов длиной  $N/2 + 1$  отсчётов и синусных сигналов длиной  $N/2 + 1$  отсчётов. Амплитуды косинусных и синусных сигналов содержатся в массивах  $\operatorname{Re} \bar{X}[k]$  и  $\operatorname{Im} \bar{X}[k]$  соответственно. В уравнении синтеза эти амплитуды умножаются на соответствующие базисные функции, формируя, таким образом, набор косинусных и синусных сигналов разных уровней. Сумма этого набора компонентов декомпозиции даёт исходное представление сигнала  $x[i]$  во временной области.

Обратите внимание, что в выражении (8.2) для массивов амплитуд использованы обозначения  $\operatorname{Re} X[k]$  и  $\operatorname{Im} X[k]$ , а не  $\operatorname{Re} \bar{X}[k]$  и  $\operatorname{Im} \bar{X}[k]$ . Дело в том, что массивы амплитуд, участвующие в процедуре синтеза и обозначенные нами  $\operatorname{Re} X[k]$  и  $\operatorname{Im} X[k]$ , немного отличаются от частотного представления сигнала  $\operatorname{Re} \bar{X}[k]$  и  $\operatorname{Im} \bar{X}[k]$ . Отличие состоит в нормализации, на что мы ссылались раньше. Несмотря на то что нормализация сводится к простому делению на постоянный коэффициент, встречается немало ошибок в компьютерных реализациях ДПФ. Будьте внимательнее с этим вопросом! Уравнение нормализации выглядит следующим образом:

$$\begin{aligned}\operatorname{Re} \bar{X}[k] &= \frac{\operatorname{Re} X[k]}{N/2}, \\ \operatorname{Im} \bar{X}[k] &= -\frac{\operatorname{Im} X[k]}{N/2}; \\ \text{исключение для двух элементов: } & \\ \operatorname{Re} \bar{X}[0] &= \frac{\operatorname{Re} X[0]}{N}, \\ \operatorname{Re} \bar{X}[N/2] &= \frac{\operatorname{Re} X[N/2]}{N}.\end{aligned}\tag{8.3}$$

Взаимосвязь частотного представления сигнала и массивов амплитуд синусно-косинусных функций синтеза. Здесь обозначения  $\operatorname{Re} \bar{X}[k]$  и  $\operatorname{Im} \bar{X}[k]$  относятся к массивам амплитуд косинусных и синусных сигналов, участвующих в процедуре синтеза.  $\operatorname{Re} X[k]$  и  $\operatorname{Im} X[k]$  — это соответственно действительная и мнимая части представления сигнала в частотной области.  $N$ , как обычно, представляет длину сигнала во временной области, выраженную числом дискретных отсчётов, а  $k$  — индекс частоты, меняющийся от 0 до  $N/2$ .

Предположим, вам известно представление сигнала в частотной области и необходимо получить его временное представление. Начать необходимо с расчёта массивов амплитуд синусно-косинусных функций синтеза. То есть, зная  $\operatorname{Re} X[k]$  и  $\operatorname{Im} X[k]$ , вам необходимо найти  $\operatorname{Re} \bar{X}[k]$  и  $\operatorname{Im} \bar{X}[k]$ . Выражение (8.3) описывает этот переход в математической форме. При программной реализации от вас потребуется осуществить три действия:

1. Разделите все отсчёты частотного представления на  $N/2$ .
2. Поменяйте знак у мнимой части.
3. Разделите на 2 первый и последний отсчёты действительной части, т. е. элементы  $\operatorname{Re} X[0]$  и  $\operatorname{Re} X[N/2]$ .

Таким образом вы получите массивы амплитуд, необходимые для синтеза сигнала в соответствии с выражением (8.2). Вместе выражения (8.2) и (8.3) описывают обратное ДПФ.

Процедура обратного ДПФ реализована в виде **Программы 8.1**. Можно сформировать два способа реализации процедуры синтеза (**8.2**), и они оба учтены в данной программе. В первом случае базисные гармонические сигналы генерируются по очереди, умножаются на соответствующий коэффициент и накапливаются в массиве, в котором по окончании обработки всех гармоник получается исходный сигнал во временной области. Второй подход подразумевает формирование за один цикл вычислений одного очередного отсчёта временного представления сигнала как суммы всех соответствующих отсчётов синусно-косинусных сигналов синтеза. Естественно, оба способа дают один и тот же результат. Различие программных реализаций очень небольшое: внутренний и внешний циклы просто меняются местами.

### **Программа 8.1**

```

100 'ОБРАТНОЕ ДИСКРЕТНОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ
110 'Сигнал во временной области, записываемый в массив XX[ ], рассчитывается
120 'на основе частотного представления, размещённого в массивах REX[ ] и IMX[ ]. 
130 '
140 DIM XX[511]    'XX[ ] - Сигнал, представленный во временной области
150 DIM REX[256]   'REX[ ] - Действительная часть частотного представления сигнала
160 DIM IMX[256]   'IMX[ ] - Мнимая часть частотного представления сигнала
170 '
180 PI = 3.14159265 'ЧИСЛО ПИ
190 N% = 512          'N% - количество элементов в массиве XX[ ]
200 '
210 GOSUB XXXX      'Подпрограмма загрузки данных в массивы REX[ ] и IMX[ ]
220 '
230
240 'Вычисление амплитуд синусных и косинусных сигналов по формуле 8-3
250 FOR K% = 0 TO 256
260 REX[K%] = REX[K%] / (N%/2)
270 IMX[K%] = -IMX[K%] / (N%/2)
280 NEXT K%
290 '
300 REX[0] = REX[0] / 2
310 REX[256] = REX[256] / 2
320 '
330 '
340 FOR I% = 0 TO 511 'Обнуление массива XX[ ], используемого для накопления
350 XX[I%] = 0
360 NEXT I%
370 '
380 ' Формула 8-2. СИНТЕЗ. СПОСОБ 1. Цикл по номеру частоты в каждой своей итерации
390 ' рассчитывает пару синусного и косинусного сигналов на всей их длине, которые
400 ' затем накапливаются в аккумуляторе XX [ ]
410 '
420 FOR K% = 0 TO 256 'K% - индекс, пробегающий по всем элементам REX[ ] и IMX[ ]
430 FOR I% = 0 TO 511 'I% - индекс, пробегающий по всем элементам XX[ ]
440 '
450 XX[I%] = XX[I%] + REX[K%] * COS(2*PI*K%*I%/N%)
460 XX[I%] = XX[I%] + IMX[K%] * SIN(2*PI*K%*I%/N%)

```

```

470 '
480 NEXT I%
490 NEXT K%
500 '
510 END

```

### **Другая версия кода в строках 380...510**

```

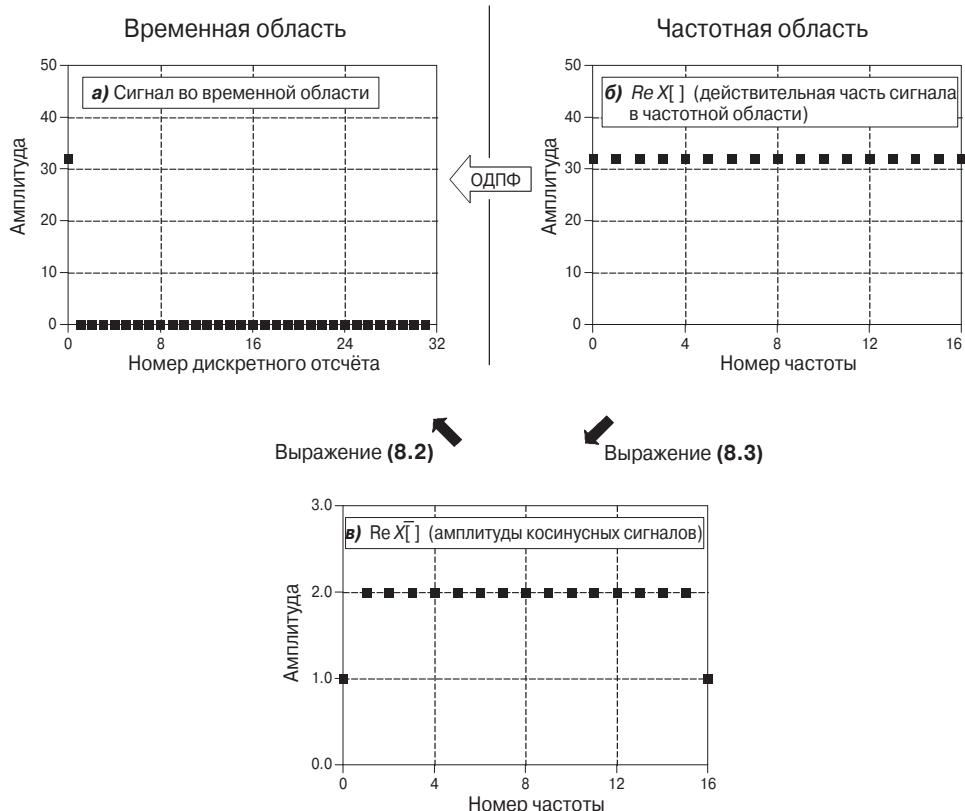
380 ' Формула 8-2. СИНТЕЗ. СПОСОБ 2. В каждой итерации цикла рассчитывается
390 ' один отсчёт сигнала во временной области путём суммирования соответствующих
400 ' отсчётов синусных и косинусных сигналов
410 '
420 FOR I% = 0 TO 511 'I% - индекс, пробегающий по всем элементам массива XX[ ]
430 FOR K% = 0 TO 256 'K% - индекс, пробегающий по всем элементам REX[] и IMX[]
440 '
450 XX[I%] = XX[I%] + REX[K%] * COS(2*PI*K%*I%/N%)
460 XX[I%] = XX[I%] + IMX[K%] * SIN(2*PI*K%*I%/N%)
470 '
480 NEXT K%
490 NEXT I%
500 '
510 END

```

**Рис. 8.6** иллюстрирует процедуру обратного ДПФ, а также то небольшое различие между частотным представлением сигнала и амплитудами косинусных и синусных функций синтеза, о котором мы говорили выше. На **(а)** приведён пример сигнала во временной области. Это единичный импульс, умноженный на константу 32 (нулевой отсчёт равен 32, а все остальные отсчёты равны 0). На **(б)** показано частотное представление этого сигнала. Действительная часть частотного представления — это постоянный сигнал, все отсчёты которого равны 32. Мнимая часть (не показанная на рисунке) имеет только нулевые отсчёты. В следующей главе будет показано, что эта пара представлений сигнала имеет важное значение: единичный импульс во временной области представляется постоянным сигналом в частотной области. Сейчас нам необходимо запомнить только следующее: на **(б)** показан результат ДПФ сигнала **(а)**, а сигнал **(а)** — это результат обратного ДПФ сигнала **(б)**.

Переход от частотного представления сигнала **(б)** к набору амплитуд тригонометрических функций **(в)** осуществляется в соответствии с выражением (8.3). В данном примере все амплитуды косинусных сигналов оказались равны двум, кроме амплитуд 0-й и 16-й косинусоид, равных единице. Амплитуды синусных сигналов не рассмотрены в этом примере, поскольку все они равны нулю и не влияют на результаты преобразований. Используя *уравнение синтеза* (8.2), можно теперь перейти от набора амплитуд тригонометрических функций разложения **(б)** к представлению сигнала во временной области **(а)**.

Итак, мы рассмотрели отличие частотного представления сигнала и набора амплитуд тригонометрических функций (*коэффициентов Фурье*). Однако мы не сказали, откуда возникает это различие. Дело в том, что частотное представление сигнала основано на использовании *спектральной плотности*. Что это такое, иллюстрирует **Рис. 8.7**. На этом рисунке показан пример действительной части частотного представления сигнала, длина которого составляет 32 дискретных отсчёта. Как обычно, частотные отсчёты нумеруются с 0-го по 16-й, образуя 17



**Рис. 8.6.** Пример обратного ДПФ. На (а) показан сигнал во временной области — единичный импульс, умноженный на константу 32 (нулевой отсчёт равен 32, а остальные отсчёты равны 0). На (б) изображена действительная часть представления этого сигнала в частотной области — постоянный сигнал, равный 32. Мнимая часть частотного представления (не показанная на рисунке) имеет все отсчёты, равные 0. На (в) представлен сигнал, отсчёты которого являются амплитудами косинусных сигналов, восстанавливающих в соответствии с выражением (8.2) исходный сигнал (а). Сигнал на (в) получается из сигнала (б) в соответствии с выражением (8.3).

отсчётов, равномерно распределённых на частотной оси в диапазоне от 0 до  $1/2$  частоты дискретизации. Спектральная плотность показывает, какая часть мощности (амплитуды) сигнала приходится на единицу полосы частот в том или ином частотном интервале. Поэтому, чтобы перейти от набора амплитуд тригонометрических функций разложения к спектральной плотности, необходимо разделить каждую амплитуду на ширину частотной полосы, которую она представляет. При этом, однако, возникает вопрос: что подразумевать под шириной полосы частот, если рассматривается дискретное частотное представление сигнала?

На рисунке проведены дополнительные вертикальные линии, проходящие по середине между соседними отсчётаами и позволяющие определить ширину полосы частот как интервал между ними. Например, отсчёту с номером 5 соответствует полоса от 4.5 до 5.5; отсчёту с номером 6 — полоса от 5.5 до 6.5 и т. д. Если ширину полосы частот, приходящуюся на каждый отсчёт, выразить в долях от всей

полосы дискретного сигнала (т. е. от  $N/2$ ), то получим, что она равна  $2/N$ . При этом граничные отсчёты (первый и последний) оказываются исключением. Для них ширина полосы вдвое меньше и равна  $1/N$ . Отсюда и возникает коэффициент  $2/N$  перехода от частотного представления к набору амплитуд (коэффициентам Фурье) с особыми первым и последним отсчётами, рассчитываемыми с коэффициентом  $1/N$ .

Почему мы игнорируем в этих примерах мнимую часть? Это сделано исключительно для того, чтобы рассматриваемое действительное ДПФ было идентично своему «старшему брату» — комплексному ДПФ. Математику комплексного ДПФ мы рассмотрим в Главе 29. При описании только действительного ДПФ многие авторы не говорят об игнорировании мнимой части. Более того, они не рассматривают даже и деление на коэффициент  $2/N$ . Имейте в виду, что вам могут повстречаться такие работы, в которых оба этих момента опущены. В нашей книге мы считаем необходимым описание этих особенностей по одной важнейшей причине. Наиболее эффективным методом выполнения ДПФ является алгоритм *быстрого преобразования Фурье (БПФ)*, который будет рассмотрен в Главе 12. Алгоритм БПФ формирует частотное представление сигнала в соответствии с выражениями (8.2) и (8.3). Если вы допустите ошибки с использованием коэффициентов нормализации, ваши программы, включающие БПФ, могут работать некорректно.

## 8.6. Анализ сигналов на основе ДПФ

Вычисление ДПФ может быть реализовано тремя совершенно разными способами. Первый из них — это решение системы уравнений. Он удобен с точки зрения прозрачности преобразования, но малоэффективен в практическом применении. Второй способ использует рассмотренный в предыдущей главе аппарат *свёртки*. Основой при этом является детектирование колебания известной формы во входном сигнале. Третий способ, известный под названием *быстрое преобразование Фурье (БПФ)*, — это оригинальный алгоритм, в основе которого лежит переход от одного  $N$ -точечного ДПФ к  $N/2$  двухточечных ДПФ. Алгоритм БПФ обычно в сотни раз быстрее других способов реализации ДПФ. В данном разделе мы рассмотрим первые два подхода. Быстрому преобразованию Фурье будет посвящена Глава 12. Обращаем ваше внимание, что результаты ДПФ, реализованного любым из трёх способов, должны быть идентичны. То, какой подход применять, зависит от условий задачи. Обычно на практике, если длина сигнала менее 32, используют свёртку. Если же длина сигнала больше 32, используют БПФ.

### 8.6.1. Вычисление ДПФ решением системы уравнений

Рассмотрим задачу реализации ДПФ в следующей трактовке. Имеется  $N$  значений сигнала во временной области; требуется найти  $N$  значений его представления в частотной области (игнорируя два всегда равных нулю отсчёта частотного представления, о которых мы ранее говорили). Из курса общей алгебры известно: чтобы найти  $N$  неизвестных, требуется составить  $N$  независимых линейных уравнений. Составить систему уравнений можно следующим образом.

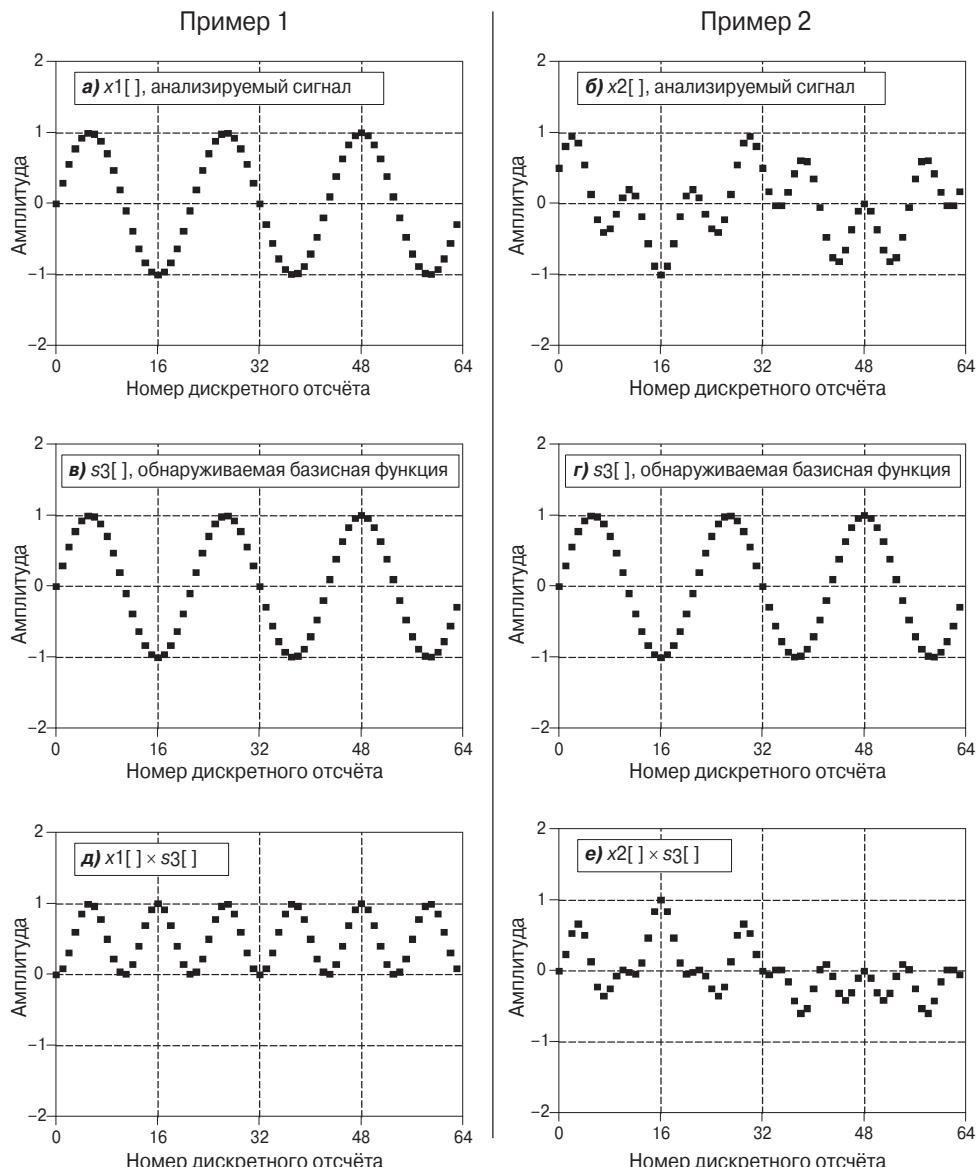
Первый отсчёт сигнала во временной области равен сумме первых отсчётов всех гармонических сигналов разложения. Получаем первое уравнение. Составляя аналогично уравнения для всех оставшихся отсчётов, получаем систему  $N$  уравнений. Её решение может быть найдено одним из известных методов решения системы линейных алгебраических уравнений, например методом исключений Гаусса. Этот способ реализации ДПФ, к сожалению, требует огромного количества вычислений, поэтому в ЦОС он практически не используется. Однако этот способ важен по другой причине. Он хорошо иллюстрирует, почему представление сигнала набором тригонометрических функций оказывается возможным, каким числом функций представляется сигнал, а также то, что базисные функции должны быть линейно независимы (об этом немного позже).

## 8.6.2. Вычисление ДПФ с помощью свёртки

Перейдём к более эффективному и общепринятому способу реализации ДПФ. Суть метода рассмотрим на примере. Предположим, нам необходимо вычислить ДПФ 64-точечного входного сигнала. Это означает, что мы должны найти 33 отсчёта действительной части и 33 отсчёта мнимой части частотного представления сигнала. Мы проиллюстрируем вычисление только одного отсчёта  $\text{Im } X[3]$ , что соответствует амплитуде синусного сигнала, у которого на длине входного сигнала (от 0-го до 63-го отсчёта) укладывается три периода повторения. Все остальные отсчёты частотного представления могут быть найдены аналогично.

Использование свёртки для вычисления отсчёта  $\text{Im } X[3]$  иллюстрирует Рис. 8.8. На (а и б) показаны два сигнала, представленные во временной области, выбранные в качестве примера. Сигналы обозначены  $x1[ ]$  и  $x2[ ]$ . Сигнал  $x1[ ]$  — это простая синусоида, у которой на длительности 64 периода дискретизации укладывается 3 периода повторения. Сигнал  $x2[ ]$  противопоставлен  $x1[ ]$  в том смысле, что представляет собой сумму большого числа синусных и косинусных сигналов, имеющих различные частоты, но не равные частоте сигнала  $x1[ ]$ . То есть сигнал  $x2[ ]$  не имеет в своём составе частоты с номером 3. На примере этих двух сигналов рассмотрим алгоритм вычисления  $\text{Im } X[3]$ . Если на входе имеется сигнал  $x1[ ]$ , то результат выполнения этого алгоритма должен дать число 32 — амплитуду синусоиды, представляющей сигнал  $x1[ ]$  (умноженную на коэффициент пропорциональности в соответствии с выражением (8.3)). Если же на вход подан сигнал  $x2[ ]$ , то на выходе должен оказаться ноль, означающий, что частота с таким номером в сигнале отсутствует.

С понятием *свёртки* мы познакомились в Главе 7. Напомним, что свёртка позволяет детектировать в произвольном входном сигнале наличие или отсутствие колебания известной формы. Входной и искомый сигналы по отсчёту перемножаются, а результаты умножений суммируются. Результатом является число, которое служит мерой схожести двух сигналов. Применимость данного подхода к реализации ДПФ иллюстрирует Рис. 8.8. На (в и г) показан один и тот же искомый сигнал — синусоида, имеющая на длительности в 64 отсчёта 3 периода повторения. На (д и е) показаны результаты перемножения сигналов, изображённых на (а, в) и (б, г) соответственно. Сумма всех отсчётов сигнала, представленного



**Рис. 8.8.** Пример анализа двух сигналов (**а** и **б**) на содержание в них определённой базисной функции (**в** и **г**). На (**д** и **е**) показаны результаты перемножений входных сигналов на обнаруженную базисную функцию. Среднее значение сигнала, представленного на (**д**), равно 0.5. Это означает, что в сигнале  $x1[ ]$  содержится данная базисная функция, и её амплитуда в сигнале равна 1. Среднее значение сигнала, показанного на (**е**), равно 0. В сигнале  $x2[ ]$  данная базисная функция не содержится.

на (**д**), равна 32, а сумма отсчётов сигнала, изображённого на (**е**), равна 0. Таким образом, мы показали, что алгоритм на основе свёртки является искомым.

Все другие отсчёты представления сигнала в частотной области могут быть получены по этому же правилу. Формализация описанного алгоритма вычисле-

ний приводит нас к *уравнению анализа* — математической записи правила перехода от временной формы представления сигнала к частотной:

$$\begin{aligned} \operatorname{Re} X[k] &= \sum_{i=0}^{N-1} x[i] \cdot \cos(2\pi k i / N), \\ \operatorname{Im} X[k] &= -\sum_{i=0}^{N-1} x[i] \cdot \sin(2\pi k i / N). \end{aligned} \quad (8.4)$$

Уравнение анализа для вычисления ДПФ. В представленных уравнениях  $x[i]$  — это сигнал во временной области, анализ которого производится.  $\operatorname{Re} X[k]$  и  $\operatorname{Im} X[k]$  — это частотное представление сигнала, которое мы хотим рассчитать. Индекс  $i$  меняется в диапазоне  $0\dots N-1$ , а индекс  $k$  — в диапазоне  $0\dots N/2$ .

Таким образом, уравнение анализа подразумевает следующие действия. Для вычисления каждого отсчёта частотного представления сигнала необходимо умножить исходный сигнал во временной области на синусный или косинусный сигнал, соответствующий номеру искомого отсчёта, и суммировать результаты умножений. Если кто-то вас спросит, что это вы делаете, можете с уверенностью ответить: «Я нахожу корреляцию входного сигнала с набором базисных функций». Описанный метод вычисления ДПФ реализует **Программа 8.2**.

В отличие от уравнения синтеза уравнение анализа не требует каких-либо специальных вычислений для первого и последнего отсчётов. Однако обращаем ваше внимание, что мнимая часть частотного представления сигнала, вычисляемая в соответствии с выражением (8.4), берётся с противоположным знаком. Как и в прежних ситуациях, эта смена знака необходима, чтобы действительное ДПФ соответствовало комплексному ДПФ. Иногда она может не производиться.

Для того чтобы описанный корреляционный алгоритм правильно работал, используемые базисные функции должны обладать одним интересным свойством: каждая из них должна быть некоррелирована со всеми другими. Это означает, что если вы по отсчёту перемножите любые две базисные функции между собой и просуммируете результаты умножений, то в результате получите ноль. Базисные функции, обладающие этим свойством, называются *ортогональными*. Существует много различных типов ортогональных функций. Это, например, прямоугольные колебания, пилообразные сигналы, импульсные сигналы и др. На основе этих базисных функций может быть произведена декомпозиция сигналов с использованием свёртки точно так же, как в случае использования тригонометрических базисных функций. Обратите внимание, мы не говорим, что это полезно, а просто утверждаем, что это возможно.

### **Программа 8.2**

```
100 'ДИСКРЕТНОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ
110 'Отсчёты частотного представления сигнала, помещаемые в массивы REX[] и
IMX[],
120 'рассчитываются по отсчётам сигнала во временной области, записанным в
XX[].
130 '
140 DIM XX[511] 'XX[ ] содержит сигнал во временной области
150 DIM REX[256] 'REX[ ] содержит действительную часть частотного
представления
```

```

160 DIM IMX[256] 'IMX[ ] содержит мнимую часть частотного представления
170 '
180 PI = 3.14159265 'Определение константы пи
190 N% = 512 'N% - это число отсчётов в массиве XX[ ]
200 '
210 GOSUB XXXX 'Предполагаемая подпрограмма загрузки данных в массив XX[ ]
220 '
230 '
240 FOR K% = 0 TO 256 'Обнуление массивов REX[ ] и IMX[ ], чтобы их можно было
250 REX[K%] = 0 'использовать в качестве аккумуляторов
260 IMX[K%] = 0
270 NEXT K%
280 '
290 ' Корреляция XX[ ] с косинусной и синусной базисными функциями (8.4)
300 '
310 FOR K% = 0 TO 256 'K% пробегает по всем отсчётам в массивах REX[ ] и IMX[ ]
320 FOR I% = 0 TO 511 'I% пробегает по всем отсчётам массива XX[ ]
330 '
340 REX[K%] = REX[K%] + XX[I%] * COS(2*PI*K%*I%/N%)
350 IMX[K%] = IMX[K%] - XX[I%] * SIN(2*PI*K%*I%/N%)
360 '
370 NEXT I%
380 NEXT K%
390 '
400 END

```

Ранее мы говорили о том, что есть два варианта программной реализации *обратного преобразования Фурье* (см. **Программу 8.1**). При переходе от одной реализации к другой внутренний и внешний циклы просто меняются местами. Это не влияет на результат выполнения программы, а только меняет ваш взгляд на то, что в ней происходит. Программа реализации *прямого ДПФ* (**Программа 8.2**) характеризуется той же особенностью. В ней внешний и внутренний циклы в строках 310 и 380 можно поменять местами. Точно так же результат выполнения программы останется тем же, а изменится только ваше представление о том, какие действия выполняются. (Эти два разных взгляда на реализацию прямого и обратного ДПФ можно расценивать как описание со стороны входа и описание со стороны выхода, точно так же, как мы делали, когда говорили о свёртке.)

В том виде, в котором написана **Программа 8.2**, она вычисляет каждый отдельный отсчёт в *частотной области* на основе всех отсчётов во *временной области*. То есть программа вычисляет отсчёты частотного представления последовательно друг за другом, а не одновременно группу отсчётов. Если внутренний и внешний циклы поменять местами, программа, пробегая последовательно по всем отсчётам сигнала во временной области, будет вычислять вклад каждого отсчёта в общий результат. После выполнения всех итераций цикла отдельные результаты суммируются, и на выходе появляются сразу все отсчёты представления сигнала в частотной области. Здесь возникает наш следующий вопрос: как влияет каждый отдельный отсчёт сигнала во временной области на результат представления сигнала в частотной области? Чтобы ответить на этот вопрос, необходимо рассмотреть характерную для Фурье-анализа особенность, называемую *дualityстью*.

## 8.7. Дуальность

Вы, наверное, уже обратили внимание на то, что *уравнения синтеза и анализа* (8.2) и (8.4) удивительно похожи. Для перехода от одной области представления сигнала к другой известные значения отсчётов перемножаются с отсчётами базисных функций, а результаты перемножений суммируются. Тот факт, что и *прямое*, и *обратное ДПФ* используют практически одну и ту же математику, весьма поразителен, особенно если учесть, что приходим мы к этим двум процедурам совершенно различными путями. Фактически единственным существенным отличием двух уравнений является то, что во временной области мы получаем один сигнал длиной  $N$  отсчётов, а в частотной области получаем два сигнала длиной  $N/2 + 1$  отсчётов каждый. В последующих главах мы рассмотрим *комплексное ДПФ*, при котором сигнал и во временной, и в частотной области содержит  $N$  отсчётов. Это делает частотное и временное представления сигнала «симметричными», а уравнения для перехода от одной области к другой — практически идентичными.

Эта «симметричность» частотного и временного представлений называется *дуальностью*. Она обуславливает целый ряд интересных свойств преобразований. Например, если в частотной области имеется лишь единственный отсчёт, то ему соответствует синусоида во временной области. Благодаря дуальности, справедливо и обратное. Если имеется единственный отсчёт во временной области, то в частотной области ему соответствует синусоида. Другим примером является *свёртка* во временной области, которой в частотной области соответствует умножение. Дуальность обуславливает то, что свёртке в частотной области соответствует умножение во временной области. Эти и другие следствия дуальности более подробно обсуждаются в Главах 10 и 11.

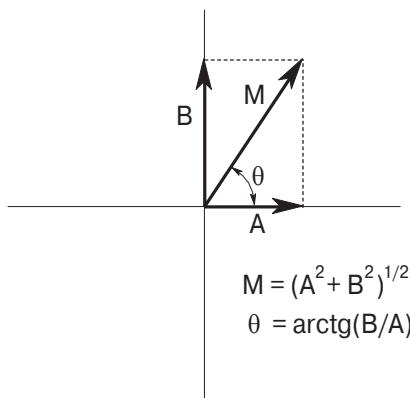
## 8.8. Форма представления в полярных координатах

До сих пор под частотным представлением сигнала мы понимали набор амплитуд синусных и косинусных сигналов (с учётом коэффициента масштабирования). Это форма представления в *прямоугольной системе координат*. Другой формой представления частотных характеристик сигнала является представление в *полярной системе координат*. В этой форме массивы  $\text{Re } X[\cdot]$  и  $\text{Im } X[\cdot]$  заменяются двумя другими массивами —  $\text{Mag } X[\cdot]$  и  $\text{Phase } X[\cdot]$  (*модулем* и *фазой*  $X[\cdot]$ ). Значения модуля и фазы попарно заменяют значения действительной и мнимой частей. Например, величины  $\text{Mag } X[0]$  и  $\text{Phase } X[0]$  могут быть вычислены только на основе значений  $\text{Re } X[0]$  и  $\text{Im } X[0]$ . Аналогично  $\text{Mag } X[14]$  и  $\text{Phase } X[14]$  рассчитываются с использованием  $\text{Re } X[14]$  и  $\text{Im } X[14]$ . И т. д. Чтобы понять этот переход, попробуем представить, что происходит при сложении косинусного и синусного сигналов одной частоты. Сумма оказывается равна косинусу той же частоты, однако с иной амплитудой и начальной фазой. Взаимосвязь двух форм представления в виде уравнения выглядит следующим образом:

$$A \cos(x) + B \sin(x) = M \cos(x + \theta). \quad (8.5)$$

Сумма косинусного и синусного сигналов равна косинусному сигналу с новой амплитудой и начальной фазой. Информация, содержавшаяся в параметрах  $A$  и  $B$ , перешла в другие две величины —  $M$  и  $\theta$ .

При таком преобразовании вся информация сохраняется: если вам известно одно представление, всегда можно перейти к другому. Информация, которая содержалась в амплитудах  $A$  и  $B$ , теперь оказалась заключена в амплитуде  $M$  и фазе  $\theta$ . Несмотря на то что в уравнении используются функции синуса и косинуса, сам переход от одних параметров к другим осуществляется по законам, характерным для векторной геометрии. **Рис. 8.9** иллюстрирует геометрическую интерпретацию взаимосвязи параметров  $A$  и  $B$  в прямоугольной и  $M$  и  $\theta$  — в полярной системах координат.



**Рис. 8.9.** Преобразование прямоугольной в полярную систему координат. Операция сложения функций синуса и косинуса (одной частоты) использует ту же математику, что и при сложении векторов.

В полярных координатах  $Mag X[ ]$  — это амплитуда косинусного сигнала (обозначенная  $M$  в выражении (8.5) и на **Рис. 8.9**), а  $Phase X[ ]$  — его фаза (обозначенная  $\theta$  в выражении (8.5) и на **Рис. 8.9**). Ниже приведены уравнения, которые позволяют перейти от частотного представления сигнала в прямоугольной системе координат к частотному представлению в полярных координатах и наоборот.

$$\begin{aligned} Mag X[k] &= (Re X[k])^2 + (Im X[k])^2)^{1/2}, \\ Phase X[k] &= \arctg(Im X[k] / Re X[k]). \end{aligned} \quad (8.6)$$

Переход от прямоугольной к полярной системе координат. Частотное представление сигнала в прямоугольных координатах  $Re X[k]$  и  $Im X[k]$  заменяется формой представления в полярной системе координат  $Mag X[k]$  и  $Phase X[k]$ .

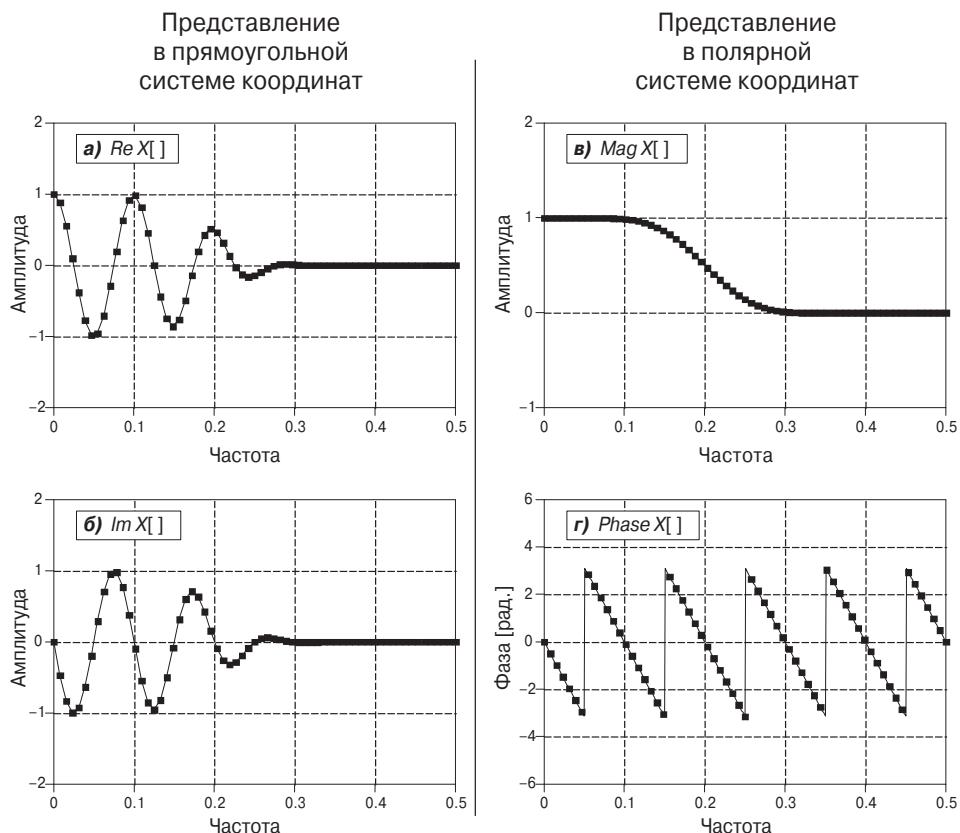
$$\begin{aligned} Re X[k] &= Mag X[k] \cos(Phase X[k]), \\ Im X[k] &= Mag X[k] \sin(Phase X[k]). \end{aligned} \quad (8.7)$$

Переход от полярной к прямоугольной системе координат. Два массива —  $Mag X[k]$  и  $Phase X[k]$  заменяются массивами  $Re X[k]$  и  $Im X[k]$ .

Использование прямоугольной и полярной систем координат позволяет рассматривать ДПФ с двух сторон. В прямоугольных координатах ДПФ — это декомпозиция  $N$ -точечного сигнала с помощью набора  $(N/2 + 1)$  косинусного и  $(N/2 + 1)$  синусного сигналов с определёнными амплитудами. При использовании полярной системы ДПФ подразумевает декомпозицию  $N$ -точечного сигнала

с помощью набора ( $N/2 + 1$ ) косинусных сигналов с определёнными амплитудами и фазами. Почему представление в полярных координатах использует именно косинусный, а не синусный сигнал? Дело в том, что синусоида не может представлять постоянный сигнал, поскольку на нулевой частоте все её значения всегда равны нулю (см. Рис. 8.5 $\alpha$  и  $\beta$ ).

Несмотря на то что и прямоугольная, и полярная формы представления с точки зрения информативности являются идентичными, во многих ситуациях использовать одну из этих форм оказывается гораздо удобнее, чем другую. В качестве примера, на Рис. 8.10 показано частотное представление сигнала и в прямоугольной, и в полярной системе координат. Понять, что означает форма кривой действительной и мнимой частей сигнала в прямоугольной форме, лучше даже не пытайтесь. Пожалуйте голову. А вот в полярной форме представления всё достаточно прозрачно: в сигнале присутствуют только частоты, амплитуда которых не превышает приблизительно 0.25, а фазовый сдвиг пропорционален частоте. Это соответствует частотной характеристике фильтра нижних частот.



**Рис. 8.10.** Представление одного и того же сигнала в частотной области в прямоугольной и полярной системе координат. Этот пример иллюстрирует, что обычно полярные координаты удобнее для восприятия и анализа частотных свойств сигнала. В свою очередь прямоугольная система используется для записи алгоритма вычислений и его реализации. Обратите внимание, что первый и последний отсёты фазы сигнала должны быть нулевыми, так же как отсёты мнимой части в прямоугольной системе координат.

Когда следует использовать прямоугольную систему координат, а когда полярную? Прямоугольная система удобна при вычислениях, т. е. при записи алгоритма и при написании программ. А вот графическая иллюстрация частотных свойств сигналов лучше представляется в полярной форме. На предыдущем примере мы убедились, что судить о частотных свойствах сигнала по форме действительной и мнимой частей частотного представления очень и очень сложно. Обычно в программной реализации для расчёта частотных характеристик используется прямоугольная форма представления, а для вывода их на экран — полярная, формируемая в результате специального преобразования.

Почему частотная характеристика оказывается нагляднее в полярной форме? Этот вопрос напрямую связан с вопросом о полезности декомпозиции на основе гармонических сигналов. Вспомним о свойстве *неискажаемой передачи гармонических сигналов*, характерном для *линейных систем*. Мы говорили о нём в Главе 5. Если на вход линейной системы подан гармонический сигнал, на выходе системы также окажется гармонический сигнал, причём той же частоты. Измениться могут только амплитуда и начальная фаза. Полярная форма непосредственно отражает эти два параметра гармонического сигнала — амплитуду и фазу. Любая система может быть описана законом изменения амплитуд и фаз гармонических сигналов.

А теперь посмотрим, что получится, если вышесказанное попробовать применить к прямоугольной форме представления. На вход системы поступает сумма синусных и косинусных сигналов. На выходе также наблюдается сумма синусных и косинусных сигналов. При этом косинусный сигнал на входе может переходить как в косинусный, так и в синусный сигнал на выходе. Аналогично синусный сигнал на входе может продуцировать как синусный, так и косинусный сигнал на выходе. Несмотря на то что эта неоднозначность может быть разрешена, всё же в целом данный метод расходится с тем основным свойством гармонических сигналов, из-за которого мы используем их в качестве компонентов декомпозиции Фурье.

## 8.9. Проблемы представления в полярных координатах

Возникает достаточно много затруднений, связанных с использованием частотного представления сигнала в полярной системе координат. Они не являются непреодолимыми, но сильно раздражают. **Программа 8.3** выполняет переход между прямоугольной и полярной системами и демонстрирует способы борьбы с некоторыми из таких негативных особенностей.

### 8.9.1. Проблема 1: радианы или градусы?

Фаза сигнала может быть выражена как в радианах, так и в градусах. Если используются градусы, то значение фазы может лежать в диапазоне  $-180\dots180$  градусов. При использовании радиан фаза принимает значения  $-\pi\dots\pi$ , т. е. приблизительно  $-3.1415926\dots3.1415926$ . В большинстве языков программирования тригонометрические функции, такие как синус, косинус, арктангенс и др., требуют измерения аргумента в радианах. Однако работать с такими числами и интер-

претировать получаемые результаты оказывается неудобно. Например, чтобы внести в сигнал фазовый сдвиг на 90 градусов, требуется к его фазе прибавить число 1.570796. Это, конечно, нельзя считать очень большим затруднением, но всё время сталкиваться с такой арифметикой утомительно. Лучший выход в данной ситуации — определить в начале вашего программного кода число  $\pi = 3.1415926$ . Тогда фазовый сдвиг на 90 градусов может быть записан как  $\pi/2$ . В ЦОС широко используют как радианы, так и градусы. Поэтому и те, и другие единицы должны быть вам хорошо знакомы.

### **Программа 8.3**

```

100 'ПЕРЕХОД ОТ ПРЯМОУГОЛЬНОЙ ФОРМЫ К ПОЛЯРНОЙ И ОТ ПОЛЯРНОЙ К ПРЯМОУГОЛЬНОЙ
110 '
120 DIM REX[256] 'Массив REX[ ] содержит действительную часть
130 DIM IMX[256] 'Массив IMX[ ] содержит мнимую часть
140 DIM MAG[256] 'Массив MAG[ ] содержит модуль
150 DIM PHASE[256] 'Массив PHASE[ ] содержит фазу
160 '
170 PI = 3.14159265
180 '
190 GOSUB XXXX 'Предполагаемая подпрограмма загрузки данных в массивы REX[] и IMX[]
200 '
210 '
220 ' 'Переход от прямоугольной к полярной форме представления; выражение (8.6)
230 FOR K% = 0 TO 256
240 MAG[K%] = SQR( REX[K%]^2 + IMX[K%]^2 ) 'см. выражение (8.6)
250 IF REX[K%] = 0 THEN REX[K%] = 1E-20 'защита от деления на ноль (проблема 2)
260 PHASE[K%] = ATN( IMX[K%] / REX[K%] ) 'см. выражение (8.6)
270 ' 'корректировка значений арктангенса (проблема 3)
280 IF REX[K%] < 0 AND IMX[K%] < 0 THEN PHASE[K%] = PHASE[K%] - PI
290 IF REX[K%] < 0 AND IMX[K%] >= 0 THEN PHASE[K%] = PHASE[K%] + PI
300 NEXT K%
310 '
320 '
330 ' 'Переход от полярной к прямоугольной форме представления; выражение (8.7)
340 FOR K% = 0 TO 256
350 REX[K%] = MAG[K%] * COS( PHASE[K%] )
360 IMX[K%] = MAG[K%] * SIN( PHASE[K%] )
370 NEXT K%
380 '
390 END

```

### **8.9.2. Проблема 2: ошибка деления на ноль**

При переходе от частотного представления сигнала в прямоугольной системе координат к полярным координатам часто появляется ситуация, когда действительная часть равна нулю, а мнимая часть не равна нулю. Такая ситуация означает, что фаза точно равна 90 или  $-90$  градусов. Однако попробуйте объяснить это компьютеру. Когда в подобной ситуации вычисление фазы производится компьютерной программой, в выражении  $Phase X[k] = \arctg(Im X[k]/Re X[k])$  возникает ошибка деления на ноль. Даже если выполнение программы при этом не пре-

рвётся, результат, который она выдаст для данной частоты, окажется ошибочным. Чтобы этого избежать, требуется перед вычислением фазы проверить, не равна ли нулю действительная часть. Если равна, то далее следует определить, положительной или отрицательной является мнимая часть, и приравнять фазу значению  $\pi/2$  или  $-\pi/2$  соответственно. Кроме того, в данной ситуации само деление необходимо пропустить. Ничего сложного в реализации всех этих действий нет. Необходимо только, чтобы была возможность немного усложнить программу. Другой вариант решения указанной проблемы демонстрирует код в строке 250 (см. **Программу 8.3**). Если действительная часть оказалась равной нулю, просто приравняйте её какому-нибудь препенебрежимо маленькою, но не нулевому значению, и компьютер не выдаст ошибки деления на ноль.

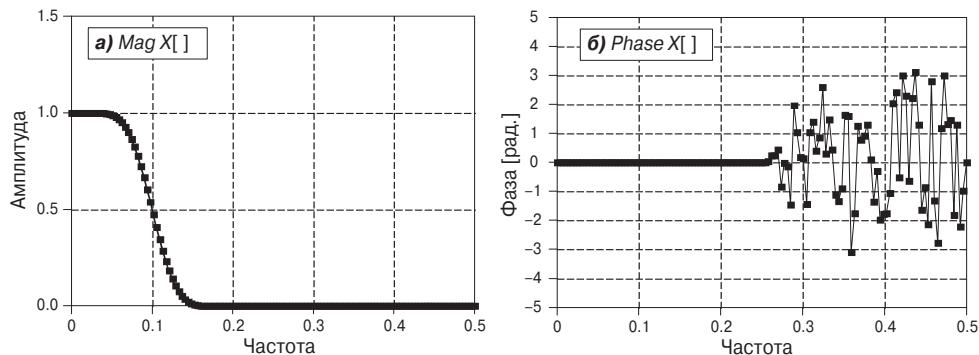
### **8.9.3. Проблема 3: неоднозначность значений арктангенса**

Представим себе ситуацию, когда действительная часть частотного представления сигнала  $Re X[k] = 1$  и одновременно мнимая часть частотного представления  $Im X[k] = 1$ . Если в соответствии с выражением (8.6) перейти от прямоугольной к полярной системе координат, то получим  $Mag X[k] = 1.414$  и  $Phase X[k] = 45$  градусов. Рассмотрим теперь другой случай:  $Re X[k] = -1$  и  $Im X[k] = -1$ . Используя выражение (8.6), в полярной форме снова получим те же значения:  $Mag X[k] = 1.414$  и  $Phase X[k] = 45$  градусов. Проблема в том, что фаза рассчитана нами неправильно. В действительности она должна быть равна  $-135$  градусам. Данная ошибка будет допускаться при расчёте фазы в соответствии с выражением (8.6) всякий раз, когда действительная часть частотного представления отрицательна. Чтобы решить эту проблему, необходимо после расчёта фазы провести проверку знака действительной и мнимой частей. Если обе они отрицательны, из получившегося значения фазы следует вычесть  $180$  градусов (или  $\pi$  радиан). Если действительная часть является отрицательной, а мнимая положительной, необходимо прибавить  $180$  градусов ( $\pi$  радиан). В **Программе 8.3** проверка знаков и коррекция фазы выполняется в строках 280 и 290. Если вы проигнорируете данную проблему, значение фазы, рассчитываемое по приведённому алгоритму, будет лежать только в диапазоне  $-\pi/2 \dots \pi/2$ , а не в диапазоне  $-\pi \dots \pi$ . Не забывайте об этом! Если вы заметите, что фаза принимает значения только на интервале  $\pm 1.5708$ , это будет означать, что вы забыли устраниТЬ неопределённость вычисления арктангенса.

### **8.9.4. Проблема 4: неправильная фаза при очень маленьких значениях модуля**

Представьте себе следующую ситуацию. Вы усердно решаете некоторую задачу и вдруг, анализируя сигналы, замечаете, что фаза иногда принимает странные значения. Например, она может выглядеть как шум, включать дребезг или как-то иначе сигнализировать вам, что её значения неправильны. В течение часа вы анализируете сотни строк кода вашей программы и наконец-то находите ответ. В те моменты, когда фаза сигнала кажется неправильной, модуль принимает очень маленькие значения, теряющиеся в ошибках округления. Если модуль сигнала очень мал, его фаза уже не имеет значения и может выглядеть странно. Пример

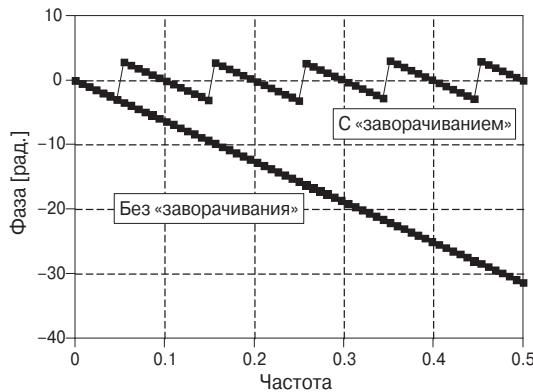
такой ситуации приведён на Рис. 8.11. Моменты, когда модуль сигнала оказывается очень мал, обнаруживаются достаточно просто: его амплитуда принимает очень маленькие значения и сигнал теряется в шумах. Сложнее обстоит дело с фазой. Если сигнал смешан с шумом и представлен в полярной системе, то в рассматриваемой ситуации фаза будет принимать случайные значения между  $-\pi$  и  $\pi$ .



**Рис. 8.11.** Фаза сигналов с малым значением модуля. На тех частотах, на которых модуль сигнала падает до очень низкого уровня, ошибки округления вызывают большие отклонения фазы от правильных значений. Будьте внимательны, чтобы не принять такой сигнал за истинный.

### 8.9.5. Проблема 5: неоднозначность периода фазы $2\pi$

Вернёмся к Рис. 8.10г и отметим, что изображённая на нём фаза сигнала имеет прерывистый характер (периодически скачком изменяет своё значение). Каждый раз, когда значение фазы должно уменьшиться ниже уровня  $-3.141592$ , оно «заворачивается», т. е. приравнивается значению  $3.141592$ . Это напрямую связано с периодическим характером гармонических сигналов. Например, фазовый сдвиг сиг-



**Рис. 8.12.** Пример фазы без «заворачивания». Верхняя кривая представляет типичный вид фазы, получаемой при программной реализации перехода от прямоугольной к полярной системе координат. Все значения фазы должны лежать в диапазоне  $-3.141592...3.141592$ ). Нижняя кривая представляет «развернутую» фазу. «Разворачивание» осуществляется путём добавления к фазе или вычитания из неё величины  $2\pi k$ , где  $k$  — целое число, выбираемое, исходя из условия обеспечения плавного (не скачкообразного) изменения формы сигнала.

нала на величину  $\theta$  равносителен сдвигу на величину  $\theta + 2\pi$ ,  $\theta + 4\pi$ ,  $\theta + 6\pi$  и т. д. Прерывистый характер фазы объясняется тем, что компьютерная программа должна выбрать только один из бесконечного числа таких вариантов. Выбирается всегда наименьшее значение, что ограничивает фазу диапазоном значений  $-\pi \dots \pi$ .

В действительности работать с фазой без этих скачков часто бывает удобнее, несмотря на то, что её значения могут при этом оказываться больше  $\pi$  или меньше  $-\pi$ . Переход к плавно изменяющейся фазе назовём «разворачиванием» фазы. Этот процесс проиллюстрирован на Рис. 8.12. В Программе 8.4 к вычисленному значению фазы прибавляется или вычитается величина  $2\pi$ , умноженная на целое число. При этом множитель выбирается так, чтобы минимизировать разницу между двумя соседними значениями фазы.

#### **Программа 8.4**

```

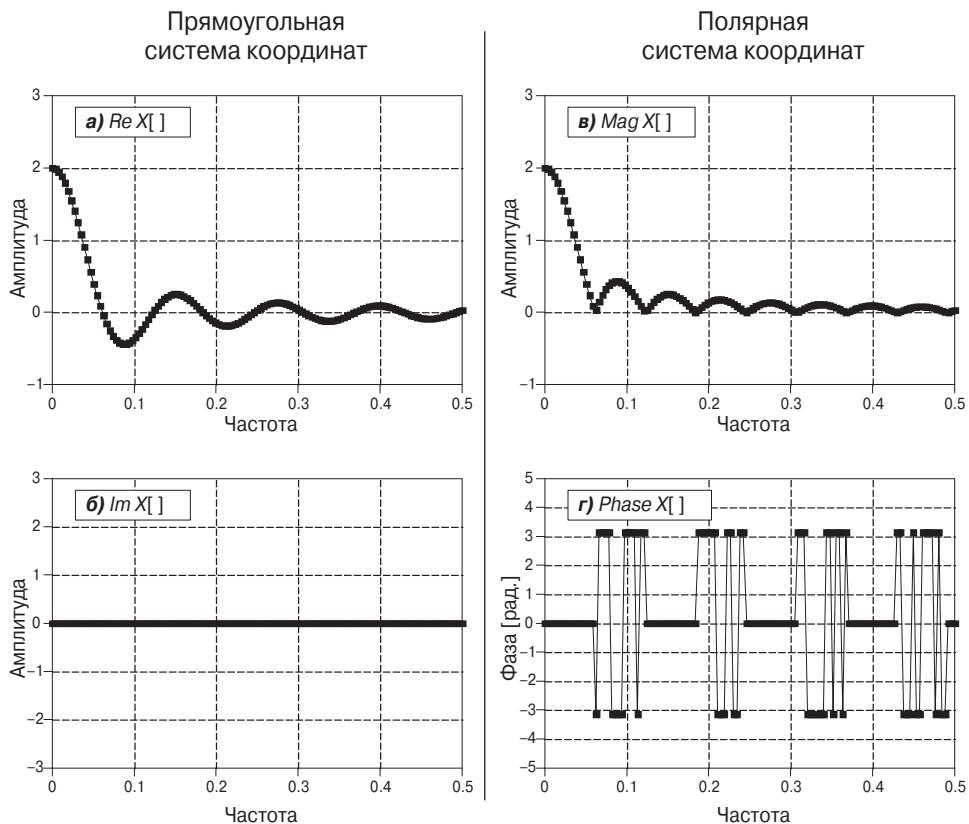
100 ' РАЗВОРАЧИВАНИЕ ФАЗЫ
110 '
120 DIM PHASE[256] 'Массив PHASE[ ] содержит исходные значения фазы
130 DIM UWPHASE[256] 'Массив UWPHASE[ ] содержит значения фазы после
разворачивания
140 '
150 PI = 3.14159265
160 '
170 GOSUB XXXX 'Предполагаемая подпрограмма загрузки данных в массив PHASE[]
180 '
190 UWPHASE[0] = 0 '0-й элемент массива фазы всегда равен нулю
200 '
210 ' 'Цикл непосредственной реализации разворачивания
220 FOR K% = 1 TO 256
230 C% = CINT( (UWPHASE[K%-1] - PHASE[K%]) / (2 * PI) )
240 UWPHASE[K%] = PHASE[K%] + C%*2*PI
250 NEXT K%
260 '
270 END

```

### **8.9.6. Проблема 6: модуль всегда положителен (неоднозначность половины периода фазы $\pi$ )**

На Рис. 8.13 изображено частотное представление сигнала в прямоугольной и полярной системах координат. Действительная часть сигнала — гладкая кривая с понятной структурой. Мнимая часть вообще равна нулю на всей длине сигнала. В то же время сигналы, представляющие полярную форму, характеризуются скачкообразным изменением значений и наличием острых углов. Причиной этого является то, что модуль по определению всегда должен быть положителен. В моменты, когда действительная часть становится отрицательной, модуль остаётся больше нуля за счёт скачкообразного изменения фазы на  $\pi$  радиан (или на  $-\pi$  радиан, что даёт тот же результат). Несмотря на то что для математики данная ситуация не является проблематичной, на практике резкие скачки в сигнале делают его сложным для анализа.

Как выход, можно разрешить модулю условно принимать и отрицательные значения. На Рис. 8.13 это заставило бы модуль выглядеть точно так же, как вы-



**Рис. 8.13.** Примеры сигналов, представленных в прямоугольной и полярной системах координат. Из-за того, что модуль всегда должен быть больше нуля (по определению), и модуль, и фаза включают точки скачкообразного изменения и характеризуются наличием острых углов. (г) вместе с тем иллюстрирует другую проблему: небольшой шум в сигнале способен вызвать резкий скачок в фазе сигнала на величину  $-\pi$  или  $\pi$ .

глядит действительная часть. Фаза при этом должна быть равна нулю во всех точках. Такой подход правомочен. Только не стоит использовать термин «модуль» по отношению к величине, которая принимает и отрицательные значения. Это противоречит определению модуля. В этой книге для обозначения модуля, принимающего отрицательные значения, мы будем использовать такое нестрогое понятие, как «модуль без заворачивания».

### 8.9.7. Проблема 7: скачки между $\pi$ и $-\pi$

Так как сдвиг фазы на  $-\pi$  даёт тот же результат, что и сдвиг на  $\pi$ , наличие ошибок округления может привести к быстрому переключению фазы между этими двумя значениями. Как показано на Рис. 8.13г, это приводит к наличию большого числа резких скачков в форме сигнала, там, где их быть не должно. Не поддавайтесь этому обману. На самом деле фаза на этих интервалах имеет гладкую форму.

# Глава 9

## ПРИМЕНЕНИЕ ДПФ

*Дискретное преобразование Фурье (ДПФ)* — это один из важнейших инструментов цифровой обработки сигналов. В данной главе будут обсуждаться три наиболее распространённых варианта использования ДПФ. Первый из них — это вычисление *частотного спектра* сигнала. Вычисление спектра непосредственно позволяет раскрыть информацию, заложенную в частоте, фазе и амплитуде гармонических компонентов, на которые раскладывается сигнал. Примером такого представления информации и системы, её распознающей, могут служить речь и слух человека. Второй вариант использования ДПФ связан с расчётом *частотной характеристики* системы по её импульсной характеристике или наоборот. Переход к частотной характеристике позволяет вести анализ и описание систем в частотной области, что является альтернативой описанию во временной области и применению свёртки. И третий вариант использует ДПФ как промежуточный этап более сложных преобразований сигнала. Классическим примером этого является реализация свёртки в частотной области — *быстрая свёртка*, на выполнение которой затрачивается в сотни раз меньше времени, чем на традиционную.

### 9.1. Спектральный анализ сигналов

Очень часто информация, переносимая *сигналом*, содержится в параметрах гармонических компонентов, на которые он может быть разложен. Это однаково касается и сигналов, существующих в природе, и сигналов, искусственно генерируемых человеком. Очень многие объекты в нашем мире совершают колебательное движение. Например, источником речи являются колебания голосовых связок; изменение яркости звезд и планет возникает вследствие их вращения вокруг своей оси и вокруг друг друга; винт корабля создает периодические колебания водных масс и т. д. Во всех этих случаях закон изменения формы сигнала во времени не играет роли. Ключевая информация содержится в частоте, фазе и амплитуде гармонических компонентов, представляющих сигнал. Чтобы выявить эту информацию, используется *ДПФ-преобразование*.

Как это происходит, рассмотрим на примере. Предположим, мы хотим исследовать звуковые колебания, распространяющиеся в толще океана. Для этого поместим микрофон в воду. Электрический сигнал с микрофона усилим до приемлемого уровня, скажем, в несколько вольт. Далее подключим аналоговый фильтр, подавляющий все частотные компоненты, лежащие выше 80 Гц, так, чтобы принимаемый сигнал мог быть оцифрован с частотой 160 отсчётов в секунду. Мы зарегистрируем с помощью такого устройства несколько тысяч отсчётов сигнала. И что же дальше?

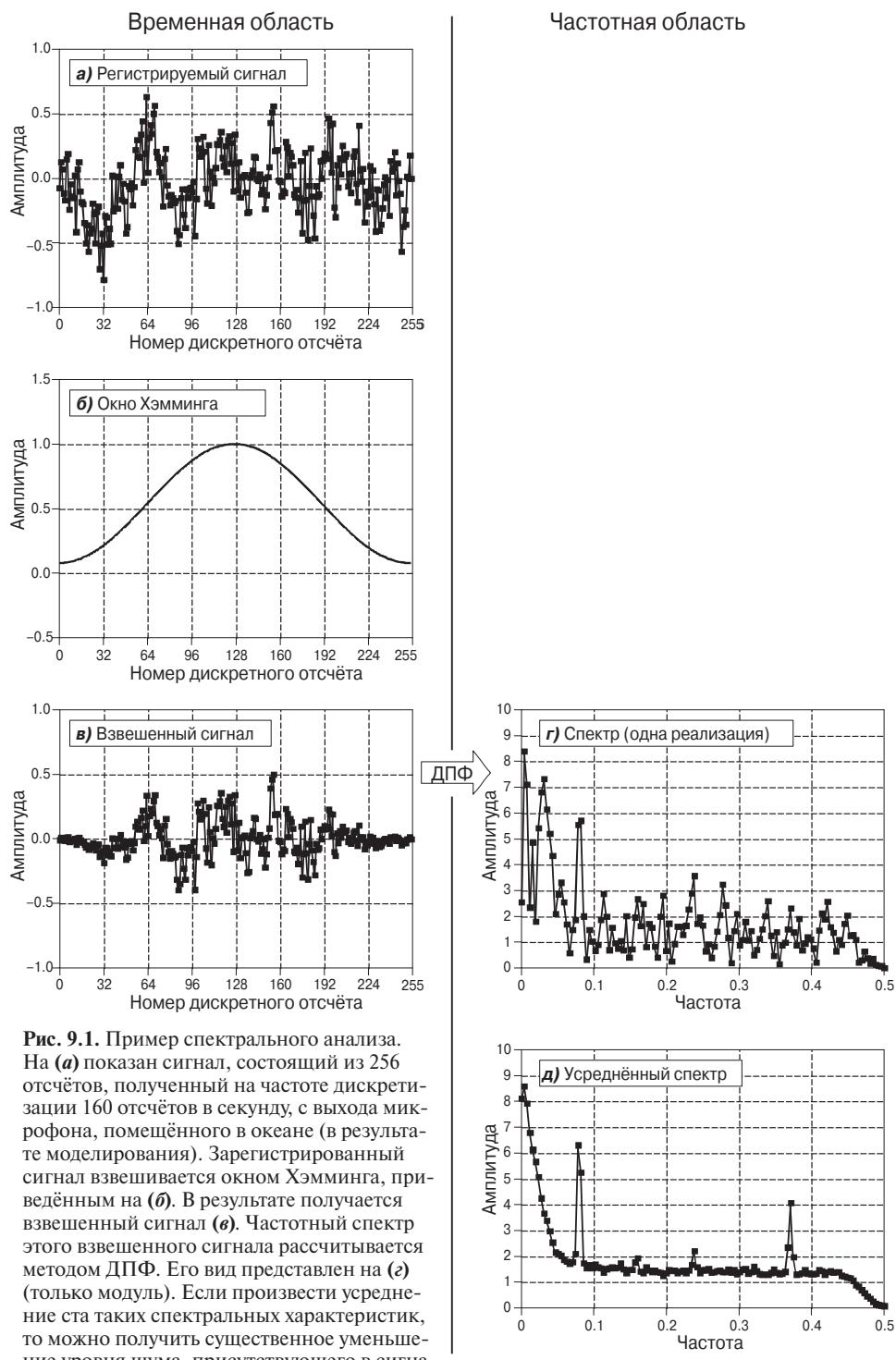
Ну, во-первых, посмотрим на данные. На Рис. 9.1 $\alpha$  показаны 256 отсчётов сигнала нашего воображаемого эксперимента. Он имеет вид реализации *случайного процесса*, и его форма не несёт для нас никакой информации. Теперь умножим этот сигнал на гладкую кривую, называемую *окном Хэмминга* ( $\delta$ ). Зачем это нужно, объясним чуть позже. Математическая запись окна Хэмминга, так же как и других окон, может быть найдена в Главе 16 (см. (16.1) и (16.2), а также Рис. 16.2 $\alpha$ ). Умножение на кривую Хэмминга даёт новый 256-точечный сигнал, у которого на краях наблюдается сильный спад амплитуды ( $\epsilon$ ).

ДПФ полученного сигнала и переход к *полярной системе координат* дают нам 129-точечный частотный *спектр* ( $\varepsilon$ ). К сожалению, спектр тоже носит случайный характер. Причиной этого является недостаточное количество информации, содержащееся в исходном 256-точечном сигнале, неспособное дать нам сведения о структуре сигнала. Если использовать ДПФ большей размерности, это всё равно не решит проблему. Например, если взять 2048-точечное преобразование, на выходе получим спектр из 1025 отсчётов. Несмотря на то что исходные 2048 отсчётов содержат больше информации, чем в предыдущем случае, *отношение сигнал/шум* остаётся на прежнем уровне, поэтому увеличение размерности ДПФ повышает разрешение по частоте, но уровень шума не снижается.

Выходом является использование большего числа отсчётов входного сигнала и меньшей размерности ДПФ. Для этого разобьём входной сигнал на сегменты по 256 дискретных отсчётов. Каждый такой сегмент умножим на окно Хэмминга, выполним 256-точечные ДПФ и представим результаты в полярной системе координат. Полученный таким образом набор спектральных характеристик сигнала можно усреднить и сформировать результирующий 129-точечный частотный спектр. На ( $\delta$ ) показан пример спектра, рассчитанного усреднением 100 выходных сигналов ДПФ, один из которых приведён на ( $\varepsilon$ ). Выигрыш предложенного подхода очевиден: уровень *шума* уменьшился настолько, что стало возможно легко выявить интересующие нас особенности структуры сигнала. Усреднение производится только для *модулей* выходов ДПФ. *Фазы* не усредняются, поскольку обычно не несут существенной информации. Уровень шума в сигнале уменьшается пропорционально корню квадратному из числа сегментов обработки. Наиболее употребимым числом сегментов является сто. Тем не менее бывают ситуации, когда усреднение приходится производить на миллионах сегментов, чтобы выявились «слабые» элементы структуры сигнала.

Есть ещё один метод уменьшения уровня шумов в частотной области. Начинают с того, что реализуют ДПФ очень большого размера, скажем, 16 384 точек. В результате получают спектр с очень хорошим частотным разрешением (8193 отсчёта), но с высоким уровнем шума. Этот сигнал обрабатывают *низкочастотным фильтром*, который сглаживает спектральную характеристику и подавляет шумы ценой снижения частотного разрешения. Например, в простейшем случае цифровой фильтр усредняет 64 соседних отсчёта исходной частотной характеристики и формирует таким образом один отсчёт результирующего частотного спектра. Выполнив все вычисления, мы получим примерно те же уровень шума и частотное разрешение, что и при использовании первого подхода. При этом 16 384 входных отсчёта разбиваются на 64 сегмента, по 256 отсчётов каждый.

Какой метод выбрать? Первый проще, поскольку он не использует фильтрации. Зато второй потенциально является более эффективным, так как цифровой

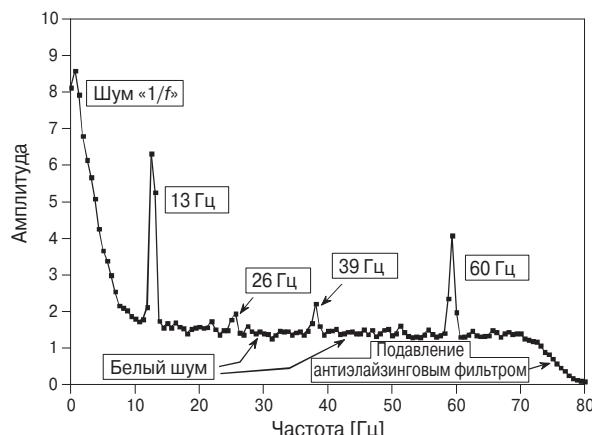


**Рис. 9.1.** Пример спектрального анализа.  
На (а) показан сигнал, состоящий из 256 отсчётов, полученный на частоте дискретизации 160 отсчётов в секунду, с выхода микрофона, помещённого в океане (в результате моделирования). Зарегистрированный сигнал взвешивается окном Хэмминга, приведённым на (б). В результате получается взвешенный сигнал (в). Частотный спектр этого взвешенного сигнала рассчитывается методом ДПФ. Его вид представлен на (г) (только модуль). Если произвести усреднение ста таких спектральных характеристик, то можно получить существенное уменьшение уровня шума, присутствующего в сигнале. Результат усреднения представлен на (д).

фильтр можно рассчитать таким образом, чтобы обеспечить оптимальное сочетание уровня шума и частотного разрешения. В то же время большая эффективность, достигаемая при использовании второго метода, редко бывает востребована. Рассмотрение большего числа входных отсчётов позволяет улучшить характеристики спектра и с точки зрения уровня шума, и с позиции разрешения по частоте. Действительно, если разбить входной сигнал на 10 000 сегментов, по 16 384 отсчёта в каждом, рассчитанный частотный спектр будет обладать высоким разрешением (8193 отсчёта) и одновременно содержать шум малого уровня (10000 усреднений). Итак, проблема решена! Мы будем использовать первый описанный метод — метод усреднения выходов ДПФ.

На Рис. 9.2 приведён пример частотного спектра сигнала, полученного в воображаемом нами эксперименте с выхода микрофона, «слушающего» океан. На этом примере проиллюстрированы некоторые особенности спектров, характерные в целом для сигналов, регистрируемых подобными методами. Сначала не будем обращать внимания на острые пики в спектре сигнала. Тогда можно сказать, что на интервале 10...70 Гц сигнал имеет достаточно ровный плоский спектр. Сигнал с таким спектром называется *белым шумом*, поскольку в его состав входят все частоты в равной степени (название «белый шум» выбрано по аналогии с белым светом). Во временной области белый шум характеризуется некоррелированностью соседних отсчётов. Это означает, что информация о сигнале в одной точке измерения не даёт никакой информации о его значении в другой точке. Примером источника белого шума может являться, например, случайное движение электронов в электрической цепи. Более простой пример: поток воды в душевой кабине, падая на пол, вызывает звук, являющийся белым шумом. Тот шум, который показан на Рис. 9.2, может быть сформирован различными источниками, в том числе входной аналоговой электроникой, а также самим океаном.

Выше 70 Гц уровень белого шума быстро падает. Это результат работы выбранного нами *антиэлайзингового фильтра*. Если бы фильтр был идеальным, он про-



**Рис. 9.2.** Пример частотного спектра. Для спектра сигнала из внешней среды характерны следующие три составляющие: белый шум и шум  $1/f$ ; наводки электрических линий передачи, импульсных источников питания, телерадиостанций, микрофонные наводки и др.; собственно сигналы, обычно состоящие из основной и второстепенных гармоник. Приведенный пример иллюстрирует некоторые из составляющих регистрируемого сигнала.

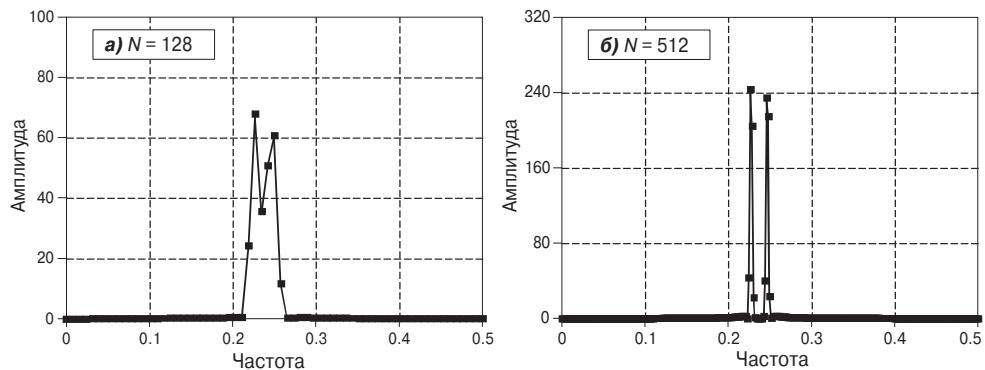
пускал бы без искажений все частоты, лежащие ниже 80 Гц, и полностью подавлял остальные. Однако практическая реализация фильтра с прямоугольной частотной характеристикой невозможна, и уровень подавления частот изменяется плавно, что и проиллюстрировано на рисунке. Будьте готовы к тому, что проблема, связанная с неидеальностью антиэлайзинговой фильтрации, всегда существует.

Ниже 10 Гц уровень шума, напротив, резко возрастает. Это связано с явлением, называемым *шумом «1/f»*. Шум «1/f» — это загадка. Он присутствует в самых разных системах, например он характерен для плотности движения автомобилей на автостраде и для шумов транзисторов. Вполне вероятно, что он может быть обнаружен во всех системах, если вести анализ на достаточно низких частотах. Однако, несмотря на широкую распространённость этого явления, теоретическое его описание ускользнуло от внимания учёных. Источник шума «1/f» может быть установлен в случае некоторых конкретных систем, однако это не даёт ответа на вопрос, почему этот шум присутствует везде. Обычно для аналоговой электроники и большинства физических процессов характерен переход от полосы белого шума к шуму «1/f» на частотах 1...100 Гц.

А теперь перейдём к острым пикам в спектре сигнала, показанного на Рис. 9.2. Наиболее просто объясняется наличие пика на частоте 60 Гц: это результат влияния электромагнитного излучения, источником которого является промышленная электросеть. Этим же могут быть обусловлены меньшие по уровню всплески на частотах, кратных 60 Гц (например, на 120, 180, 240 Гц и т. д.). Они возникают из-за того, что промышленная частота не является идеальной синусоидой. К этой же группе следует отнести возможные пики на частотах 25...40 кГц, которые используются разработчиками импульсных источников электропитания. Если идти выше по оси частот, то далее идут радио- и телевизионные станции, занимающие полосы частот в области мегагерц. Пики в области низких частот могут быть обусловлены вибрациями каких-либо составляющих системы. Они называются микрофонными и лежат в полосе 10...100 Гц.

Переходим, наконец, к интересующим нас сигналам. Явно виден острый пик высокого уровня на частоте 13 Гц и два пика поменьше на частотах 26 и 39 Гц. Как будет показано в следующей главе, такой спектр характерен для несинусоидальных периодических сигналов. Частота 13 Гц называется основной, а частоты 26 и 39 Гц — второй и третьей гармоникой соответственно. В спектре должны присутствовать и другие, более высокие, гармоники, кратные 13 Гц (например 52, 65, 78 Гц и т. д.). На Рис. 9.2 их не видно из-за того, что они скрыты в шумах. Обнаруженный нами сигнал 13 Гц мог, например, возникнуть вследствие работы трёхлопастного двигателя подводной лодки, вращающегося со скоростью 4.33 оборота в секунду. Рассмотренный нами подход лежит в основе пассивной гидролокации, которая идентифицирует источники подводных звуковых волн по их спектральному составу.

Представим теперь, что два пика в спектре расположены очень близко друг к другу, так, как это показано на Рис. 9.3. То, как близко могут находиться два пика, не сливаясь в один, называется разрешающей способностью по частоте. Разрешающая способность ограничивается двумя факторами. Первый — это размерность ДПФ. Частотный спектр, полученный в результате  $N$ -точечного ДПФ, состоит из  $(N/2 + 1)$  отсчёта, равномерно распределённых на частотной оси от нуля до половины частоты дискретизации. Чтобы два близко расположенных пика были раз-



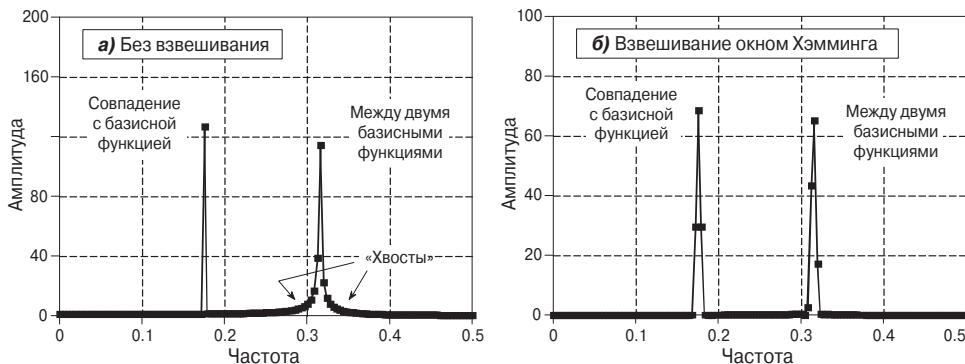
**Рис. 9.3.** Разрешение частотного спектра. Чем выше размерность ДПФ, тем выше способность разделять элементы спектра, расположенные на частотной оси близко друг к другу. В приведенном примере 128-точечное ДПФ не позволяет разделить два близко расположенных пика, а 512-точечное — позволяет.

делимы, необходимо, чтобы частотный интервал между ними был больше шага по частоте. Например, для сигнала, показанного на Рис. 9.3, 512-точечного ДПФ оказалось достаточно, чтобы разрешить пики, а 128-точечного — нет.

Второй фактор, ограничивающий частотное разрешение, менее очевиден. Представьте себе сигнал, равный сумме двух синусоид, частоты которых отличаются очень слабо. При рассмотрении на относительно коротком интервале времени, скажем, в несколько периодов гармонического колебания, сигнал будет выглядеть так, как будто синусоида только одна. Чем меньше различие частот, тем больший интервал наблюдения необходим, чтобы сказать, что в состав сигнала входит более чем одна гармоника. Другими словами, разрешение по частоте ограничивается длиной сигнала. Это не то же самое, что ограничение размерности ДПФ, так как длина входного сигнала необязательно должна быть равна размерности ДПФ. Например, сигнал, включающий 256 отсчетов, может быть дополнен нулями до 2048 отсчетов, и размерность ДПФ будет составлять 2048. На выходе такого преобразования мы получим 1025 отсчетов. Добавленные нулевые отсчеты не могут изменить форму спектра; они только позволят размещать больше отмечек на частотной оси. Различие же двух близко расположенных пиков будет улучшено лишь незначительно. Если длина сигнала и размерность ДПФ совпадают, то влияние указанных двух факторов окажется приблизительно одинаковым. Вскоре мы вернемся к обсуждению этого вопроса.

Рассмотрим теперь такой вопрос: каким окажется спектр сигнала, если он представлен синусоидой с частотой, попадающей между двумя соседними частотами базисных функций? Взгляните на Рис. 9.4а. Здесь изображен частотный спектр сигнала, включающего две гармоники. Одна точно совпадает с базисной функцией, а другая находится между двумя соседними базисными функциями. Как и следовало ожидать, спектр первой гармоники представлен одной-единственной точкой. Спектр второй гармоники имеет более сложную форму. Вместо одной точки мы получили пик, у которого слева и справа имеются «хвосты» («боковые лепестки»), существенно увеличивающие его ширину.

Как избавиться от этих «хвостов»? Необходимо умножить сигнал на окно Хэмминга, перед тем как вычислять ДПФ. Мы уже делали это раньше. Результат пока-



**Рис. 9.4.** Иллюстрация использования взвешивания при спектральном анализе. На (а) показан спектр сигнала (только модуль), состоящего из двух синусоидальных колебаний. Одна синусоида точно совпадает с одной из базисных функций ДПФ и поэтому представляется одним единственным отсчётом. Другая синусоида попадает между двумя соседними базисными функциями, поэтому её спектр сопровождается «хвостами». На (б) показан спектр того же сигнала, но только взвешенного окном Хэмминга перед выполнением ДПФ. Взвешивание привело к давлению уровня «хвостов» и к близкой схожести двух пиков, однако их ширина увеличилась.

зан на Рис. 9.4б. Можно отметить три изменения спектра сигнала, появившихся вследствие взвешивания окном Хэмминга. Во-первых, оба пика стали практически одинаковыми. Это хорошо. Во-вторых, «хвосты», от которых мы хотели избавиться, стали существенно меньше. Это тоже хорошо. В-третьих, увеличилась ширина пиков, тем самым снизилась разрешающая способность по частоте. Это плохо. В терминах ЦОС это означает, что применение оконных функций обеспечивает компромисс между частотным разрешением (шириною пиков) и уровнем «хвостов», обуславливающих их перекрытие.

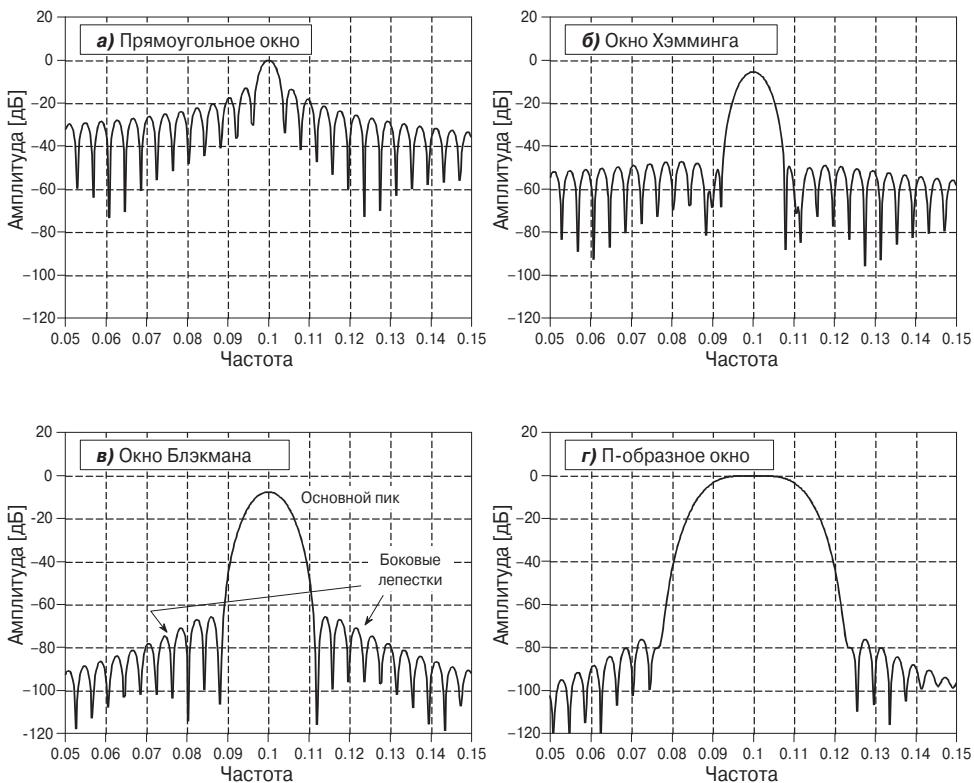
Попробуем найти теоретическое обоснование этому явлению. Представим себе дискретный синусоидальный сигнал бесконечной длительности с частотой, равной 0.1 частоты дискретизации. Частотный спектр такого сигнала представляет собой бесконечно короткий всплеск в одной-единственной точке, а на всех остальных частотах он равен нулю. Разумеется, ни сигнал с бесконечной длительностью, ни спектр с бесконечно малой шириной не подходят для представления в компьютере. Поэтому приходится модифицировать сигнал с помощью двух действий, каждое из которых изменяет исходную структуру его спектра.

Во-первых, мы «обрезаем» сигнал по длительности и искажаем содержащуюся в нём информацию, умножая его на весовое окно. Например, умножая сигнал на прямоугольное окно длиной 256 дискретных отсчётов, мы сохраним неизменными 256 точек этого сигнала бесконечной длины, а все остальные отсчёты обнуляем. Аналогично окно Хэмминга формирует группу отсчётов сигнала, в которых сохраняется информация и обнуляет все отсчёты вне окна. Сигнал по-прежнему остаётся бесконечным, но теперь только конечное число его отсчётов имеет ненулевое значение.

Как же влияет такое взвешивание на спектральный состав сигнала? Как будет показано в Главе 10, умножение *во временной области* равносильно свёртке *в частотной области*. Так как спектр исходного сигнала фактически представляет со-

бой единичный импульс, сдвинутый на частоту гармонического сигнала, то спектр взвешенного сигнала будет равен спектру весового окна, сдвинутому по частотной оси на ту же величину. На Рис. 9.5 показано, как будет изменяться спектр рассматриваемого нами синусоидального сигнала при умножении на весовое окно четырёх различных типов (если для вас удобнее представление в логарифмическом масштабе, вы найдёте его в главе 14). На (а) представлен результат взвешивания **прямоугольным окном**. Использование двух других наиболее широко применяемых типов окон — окна Хэмминга и окна Блэкмана — иллюстрируют (б) и (в) (описание этих окон дают выражения (16.1), (16.2) и Рис. 16.2а).

**Рис. 9.5** показывает, что использование окна любого типа искажает спектральный состав сигнала, расширяя основной пик и дополнняя его слева и справа бесконечными «хвостами», состоящими из многочисленных так называемых бо-



**Рис. 9.5.** Увеличенное изображение основного пика в частотном спектре гармонического сигнала, взвешенного различными типами окон. Каждый такой пик характеризуется наличием основного лепестка, окружённого справа и слева набором боковых лепестков. Изменением формы весового окна можно понизить уровень боковых лепестков ценой расширения основного. Прямоугольное окно (а) даёт наиболее узкий основной пик, но максимальный уровень боковых лепестков. Окно Хэмминга (б) и окно Блэкмана (в) позволяют уменьшить амплитуду боковых лепестков, но основной пик при этом расширяется. П-образное окно (г) используется в тех случаях, когда важно точно измерить уровень основного лепестка. Приведённые кривые построены для случая весовых окон длиной 255 отсчётов. Если длины окон выбирать большими, то ширина основного пика будет пропорционально уменьшаться.

ковых лепестков. Это неизбежное явление, возникающее при ограничении бесконечного сигнала конечным временным интервалом. Здесь для нас интересен выбор между тремя типами окон. Окно Блэкмана имеет наиболее широкий основной пик (это плохо), но наименьшую амплитуду боковых лепестков (это хорошо). Прямоугольное окно характеризуется наименьшей шириной основного пика (это хорошо) и наибольшей амплитудой боковых лепестков (это плохо). Окно Хэмминга имеет промежуточные характеристики.

Следует отметить, что частотные спектры на **Рис. 9.5** являются непрерывными, а не дискретными. Дело в том, что после процедуры взвешивания сигнал по-прежнему остаётся бесконечным, несмотря на то, что большинство его отсчётов равно нулю. Это, в свою очередь, означает, что спектр сигнала содержит  $(\infty/2 + 1)$  значений частотных компонентов между 0-й частотой и половиной частоты дискретизации. То есть спектр является непрерывным.

Отсюда становится понятной вторая модификация сигнала, необходимая для того, чтобы с ним можно было работать вычислительным средствам: *выберем N отсчётов сигнала, отбросив все остальные*. В эти  $N$  отсчётов должны входить все ненулевые отсчёты сигнала, определяемые длиной окна, но также может входить произвольное число нулевых отсчётов. Это фактически приводит к дискретизации непрерывного частотного спектра, если периодически продолжить выделенный окном сегмент сигнала. Например, если  $N$  выбрано равным 1024, спектр сигнала будет содержать 513 отсчётов, равномерно распределённых между нулём и половиной частоты дискретизации. Если  $N$  выбрано намного большим длины весового окна, то дискретные выборки спектра будут располагаться очень близко друг к другу, точно передавая форму спектра оконной функции — его максимумы и минимумы. Если  $N$  выбрано равным длине окна, то относительно небольшое число дискретных выборок спектра приведет к тому, что вместо исходного регулярного чередования максимумов и минимумов спектра сигнала (**Рис. 9.5**) будет наблюдаться нерегулярное их чередование в форме *боковых лепестков*, зависящее от момента выборки частотных отсчётов. Это объясняет, почему два пика в спектре сигнала, изображённом на **Рис. 9.4a**, отличаются друг от друга. Оба пика имеют одинаковый исходный вид, показанный на **Рис. 9.5a**. Однако в результате выборки  $N$  отсчётов их форма изменилась, причём не одинаково. Наличие или отсутствие боковых лепестков зависит от того, как берутся частотные выборки по отношению к положениям максимумов и минимумов спектра окна. Если частота гармонического сигнала точно совпадает с частотой одной из базисных функций, то частотные выборки будут попадать в точки минимумов окна, и боковых лепестков мы практически не увидим. Если же частоты сигнала и базисной функции не совпадут, то это и приведёт к появлению боковых лепестков различного уровня и формы.

Это приводит нас к необходимости использования *П-образного спектрального окна*, показанного на **Рис. 9.5g**. Есть приложения, в которых при анализе частотного спектра требуется очень точно измерять уровни имеющихся всплесков. В то же время дискретное преобразование Фурье даёт нам дискретный спектр, и нет никаких гарантий, что его отсчёты точно совпадут с интересующими нас пиками и будут адекватно отражать ситуацию. Скорее можно предположить, что в большинстве случаев дискретные моменты выборки попадут в окрестности точки максимума и дадут меньшие значения отсчётов. Чтобы уйти от этой проблемы, можно использовать окно, которое в частотной области имеет П-образный вид,

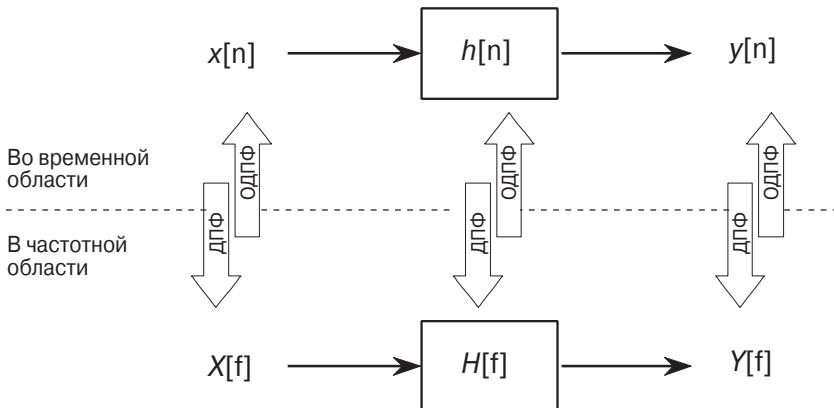
т. е. плоскую вершину. В результате один или несколько отсчётов дискретного спектра будут более точно передавать истинное значение. Из Рис. 9.5г видно, что платой за это свойство П-образного окна является сильное расширение основного лепестка, что приводит к низкой разрешающей способности.

Форма временного окна, П-образного в частотной области, полностью совпадает с формой импульсной характеристики *низкочастотного фильтра (НЧФ)*. Теоретическое обоснование этого утверждения будет рассмотрено в последующих главах. Сейчас для нас важно узнать, как этот приём применять. В Главе 16 описывается фильтр низких частот, называемый *оконным фильтром*. Методика проектирования его *импульсной характеристики* (которую мы хотим использовать в качестве весового окна) описывается выражением (16.4), а Рис. 16.4а иллюстрирует её вид. Чтобы воспользоваться приведенным выражением, необходимо задать два параметра:  $M$  и  $f_c$ . Они определяются по следующим формулам:  $M = N - 2$  и  $f_c = s/N$ , где  $N$  — размерность применяемого ДПФ, а  $s$  — число отсчётов, формирующих плоскую вершину кривой (обычно 3...5). Программа 16.1 производит расчёт желаемой импульсной характеристики НЧФ (или П-образного спектрального окна в данном случае). В программе учтено два тонких момента: нормализация с использованием масштабирующего коэффициента  $K$  и предотвращение ошибки деления на ноль. При использовании данного метода следует помнить, что постоянная величина, входящая в состав сигнала и имеющая значение единицы, в частотной области даст единицу на нулевой частоте. В то же время гармонический сигнал с единичной амплитудой в частотной области будет представлен гармоникой с уровнем 1/2 (это обсуждается в Главе 8 в разделе «Синтез сигнала с помощью обратного ДПФ»).

## 9.2. Частотные характеристики систем

При анализе *систем* во *временной области* используется операция *свёртки*. Подобный анализ может быть проведен и в *частотной области*. С использованием преобразования Фурье произвольный входной сигнал представляется в виде совокупности простых гармонических сигналов с определёнными амплитудами и фазами. Аналогично преобразование Фурье можно применить и к выходному сигналу. Любая *линейная система* может быть полностью описана тем, как она изменяет амплитуды и фазы гармонических сигналов, поданных на её вход. Этот характер изменения гармонических сигналов называется *частотной характеристикой*. Так как и импульсная, и частотная характеристики системы содержат о ней полную информацию, между ними должна существовать однозначная взаимосвязь. Если известна одна из характеристик, всегда можно получить другую. Связь между импульсной и частотной характеристиками основана на одном из фундаментальных положений обработки сигналов: *частотная характеристика системы является преобразованием Фурье её импульсной характеристики*. Эта взаимосвязь проиллюстрирована на Рис. 9.6.

В соответствии с общими правилами, принятыми в ЦОС, для обозначения импульсных характеристик используются строчные буквы, а для обозначения частотных характеристик — соответствующие прописные. Если импульсная характеристика обычно записывается как  $h[ ]$ , то традиционным обозначением час-



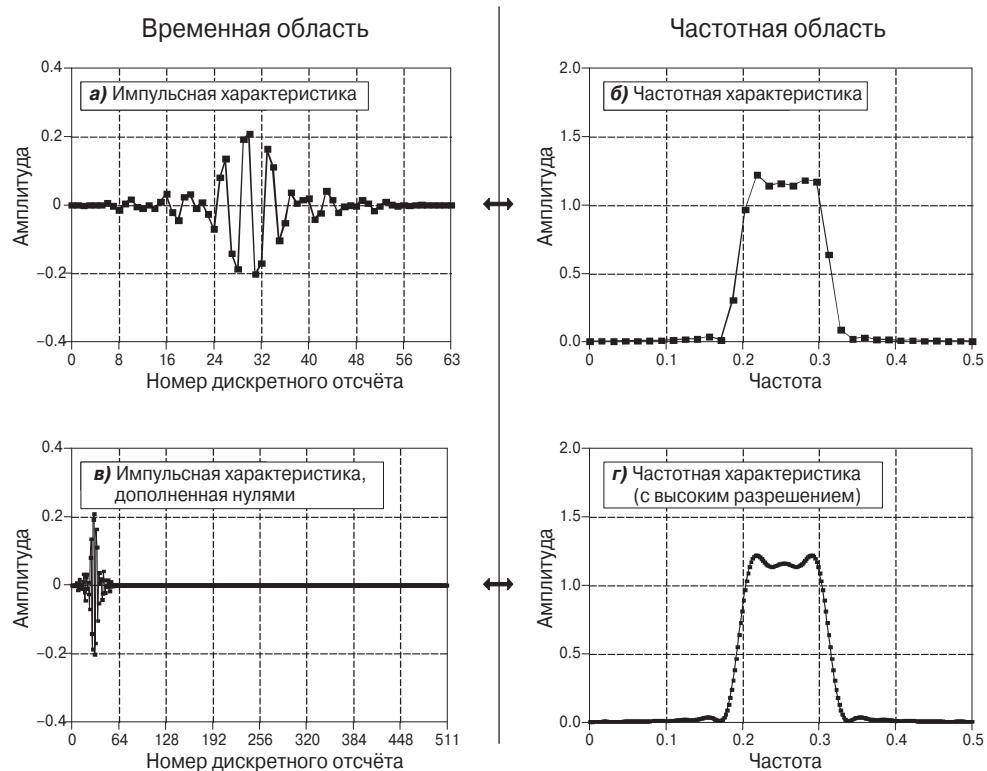
**Рис. 9.6.** Сопоставление функционирования системы во временной и частотной областях. Во временной области входной сигнал сворачивается с импульсной характеристикой системы, формируя выходной сигнал в соответствии с формулой  $x[n] * h[n] = y[n]$ . В частотной области спектр входного сигнала умножается на частотную характеристику системы, в результате чего получается выходной спектр:  $X[f] \times H[f] = Y[f]$ . Представление сигналов во временной и частотной областях связано прямым и обратным ДПФ.

тотной характеристики является  $H[ ]$ . Во временной области поведение системы описывается свёрткой:  $x[n] * h[n] = y[n]$ . В частотной области спектр входного сигнала умножается на частотную характеристику, формируя выходной спектр в соответствии с формулой  $X[f] \times H[f] = Y[f]$ . То есть свёртка во временной области равносильна умножению в частотной.

На Рис. 9.7 показан пример использования ДПФ для перехода от импульсной характеристики системы к частотной характеристике. Импульсная характеристика системы приведена на (а). Вид этой кривой не дает вам даже намека на то, что данная система делает. Применив 64-точечное дискретное преобразование Фурье и вычислив модуль, мы получаем амплитудно-частотную характеристику, приведенную на (б). Теперь назначение системы становится очевидным: она пропускает частотные компоненты сигнала, лежащие между 0.2 и 0.3, и подавляет все остальные компоненты. Это фактически полосовой фильтр. Может быть легко построена также фазо-частотная характеристика сигнала, но её вид мало для нас интересен и его сложнее проанализировать. Мы поговорим об этом в следующих главах.

Кривая на (б) очень неровная, что связано с относительно малым представляющим её числом отсчётов. Можно улучшить вид этой кривой, дополнив импульсную характеристику системы нулями перед выполнением ДПФ. Например, продлив таким образом импульсную характеристику до 512 отсчётов, как показано на (в), мы получим гораздо лучшее частотное разрешение и более гладкую частотную характеристику (г).

Какое максимальное разрешение мы можем получить в частотной области указанным способом? Ответ: бесконечно высокое, если дополнить импульсную характеристику нулями до бесконечной длины. Другими словами, частотное разрешение ограничивается только размерностью ДПФ, и больше ничем. Мы приходим к очень важному утверждению. Даже несмотря на то, что импульсная характеристика



**Рис. 9.7.** Расчёт частотной характеристики по известной импульсной характеристики. С помощью дискретного преобразования Фурье можно перейти от импульсной характеристики системы (*а*) к её частотной характеристики (*б*). Если предварительно дополнить импульсную характеристику нулевыми отсчётыами (*в*), можно получить частотную характеристику с более высоким частотным разрешением (*г*). На рисунке приведён только модуль частотной характеристики; обсуждение фазы отложено до следующей главы.

системы дискретна, её частотная характеристика непрерывна.  $N$ -точечное ДПФ импульсной характеристики даёт нам  $(N/2 + 1)$  отсчёт этой непрерывной кривой. Если увеличить размерность ДПФ, повысится частотное разрешение, и мы получим больше информации о том, как выглядит частотная характеристика системы. Не забывайте, что представляет собой частотная характеристика: это изменения, вносимые системой в амплитуды и фазы гармонических сигналов, проходящих через неё. Поскольку входной сигнал может содержать любые частоты в диапазоне от 0 до половины частоты дискретизации, частотная характеристика на этом интервале должна быть непрерывной.

Эта идея может быть более понятна, если рассматривать другой тип Фурье-преобразования — преобразование Фурье дискретного времени. Рассмотрим  $N$ -точечный сигнал, который путём ДПФ переводится в  $(N/2 + 1)$ -точечный сигнал в частотной области. Вспомним (мы говорили об этом в предыдущей главе), что ДПФ считает сигнал во временной области бесконечным и периодическим. То есть известные нам  $N$  точек считаются повторяющимися вновь и вновь вправо и влево до плюс и минус бесконечности. А теперь подумаем, что происходит, когда мы начинаем дополнять сигнал во временной области всё большим числом ну-

лей, стараясь всё более и более повышать частотное разрешение. Добавляя к сигналу нулевые отсчёты, мы увеличиваем период повторения сигнала, одновременно сближая отсчёты в частотной области.

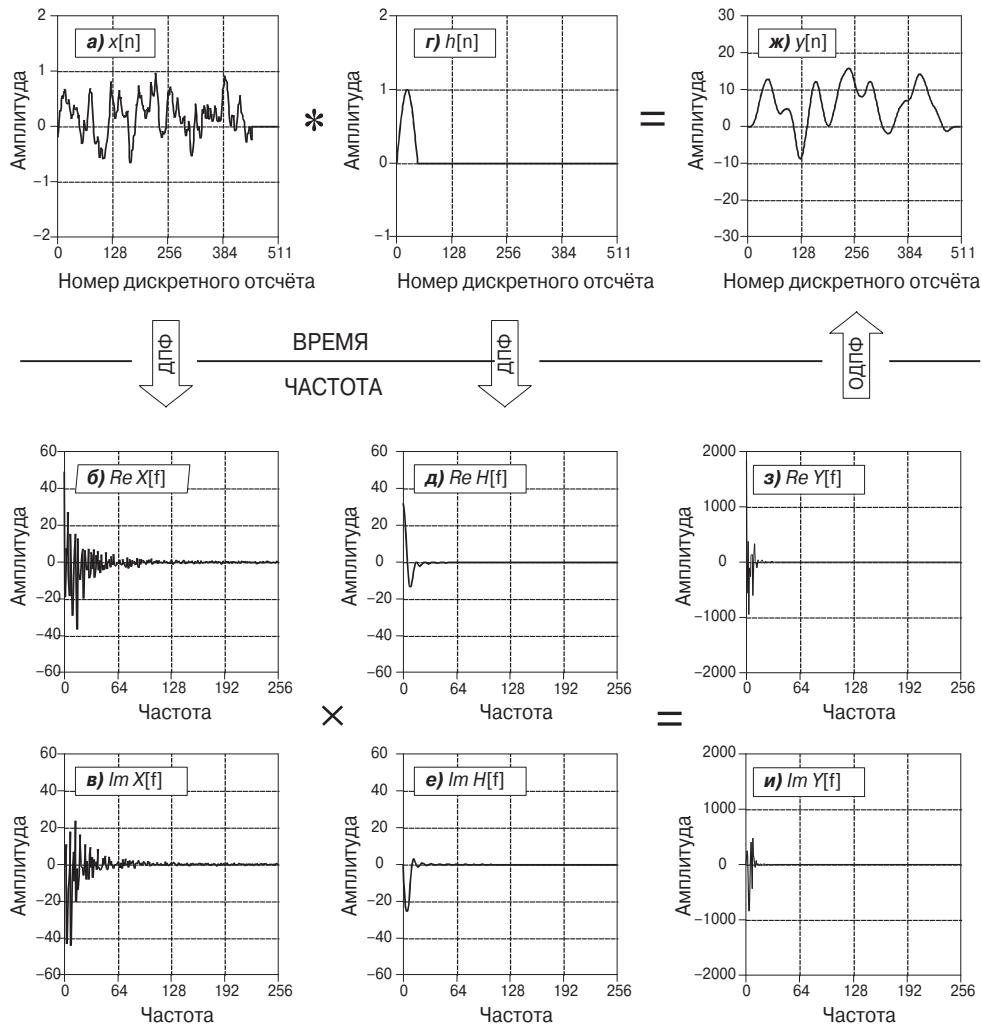
А теперь рассмотрим крайний случай: сигнал во временной области дополним бесконечным числом нулей. Мы получим иную ситуацию, отличающуюся двумя особенностями. Во-первых, период повторения сигнала стал бесконечно большим, т. е. мы перешли к апериодическому сигналу. Во-вторых, в частотной области мы получили бесконечно малое расстояние между соседними отсчётами частотной характеристики. То есть сигнал стал непрерывным. Это и есть преобразование Фурье дискретного времени — процедура преобразования дискретного апериодического сигнала, заданного во временной области, в непрерывный сигнал в частотной области. Более строго можно сказать, что частотная характеристика системы рассчитывается путём взятия преобразования Фурье дискретного времени от её импульсной характеристики. Поскольку вычислительные машины не работают с бесконечными последовательностями, на практике используется ДПФ-преобразование и рассчитывается дискретный вариант истинной непрерывной частотной характеристики. В этом проявляется разница между тем, что мы можем реализовать на компьютере (ДПФ), и тем, что мы можем написать в виде математических выражений (преобразование Фурье дискретного времени).

## 9.3. Свёртка в частотной области

Предположим, вы недолюбливаете *свёртку*. Что же вам делать, если имеется входной сигнал, *импульсная характеристика* системы и требуется найти сигнал на выходе? Выход есть: переведите сигнал и импульсную характеристику в *частотную область*, выполните умножение и представьте результат вновь во *временной области* (Рис. 9.8). При этом вы перейдёте от одной свёртки к двум ДПФ, процедуре перемножения и обратному ДПФ-преобразованию. Несмотря на существенное различие методик, результаты такого подхода и традиционной свёртки должны быть абсолютно одинаковыми.

Однако стоит ли так «ненавидеть» свёртку, чтобы решиться на выполнение этих, казалось бы, гораздо более сложных процедур? Ответ: да. Свёртка имеет два существенных недостатка. Во-первых, она описывается достаточно сложной математикой. Например, если вам известен сигнал на выходе системы и её импульсная характеристика, то найти входной сигнал (эта процедура именуется *обращением свёртки*) окажется вовсе не просто во временной области. В то же время обращение свёртки может быть перенесено в частотную область, где оно сводится к простому делению Фурье-образов. Переход в частотную область целесообразен в тех случаях, когда преобразование Фурье оказывается менее сложным в реализации, чем свёртка. С этой точки зрения два подхода оцениваются не тем, какой из них вам больше нравится, а тем, какой для вас менее проблематичен.

Второй фактор, определяющий ваше отношение к свёртке, — это скорость вычислений. Представьте, что вы спроектировали фильтр с *ядром* (импульсной характеристикой), состоящим из 512 отсчётов. При реализации его на персональном компьютере с частотой 200 МГц и арифметикой с плавающей точкой потребуется около одной миллисекунды для формирования каждого выходного отсчёта



**Рис. 9.8.** Свёртка в частотной области. Во временной области сигнал  $x[n]$  сворачивается с импульсной характеристикой  $h[n]$  и формирует выходной сигнал  $y[n]$  (а, г, ж). Та же процедура выполняется в частотной области. Для получения частотного спектра входного сигнала и частотной характеристики системы используется ДПФ, что изображено на (б, в) и (д, е) соответственно. Перемножение этих двух сигналов в частотной области даёт спектр выходного сигнала, что показано на (з и и). Для получения сигнала на выходе системы (ж) используется обратное ДПФ-преобразование.

с использованием стандартной процедуры свёртки. То есть пропускная способность системы всего 1000 отсчётов в секунду. Это в 40 раз меньше, чем требуется для приложений высококачественного аудио, и в 10 000 раз меньше, чем требует качественное телевизионное вещание!

Причиной низкой скорости вычисления свёртки по традиционному алгоритму является необходимость выполнения огромного количества операций умножения и сложения. Если просто перейти к реализации свёртки в частотной области с использованием ДПФ, это не принесёт никакого положительного эффекта. Вы-

полнение ДПФ потребует такого же числа арифметических операций, которое требует свёртки. Прорыв в решении этой проблемы был сделан в начале 60-х годов XX века, когда был разработан алгоритм *быстрого преобразования Фурье (БПФ)*.

БПФ — это оригинальный алгоритм быстрого вычисления ДПФ. С использованием БПФ свёртка в частотной области становится в сотни раз быстрее обычной. Те задачи, которые выполнялись часами, теперь решаются за минуты. Вот почему использование БПФ и работа в частотной области так привлекательны. О БПФ пойдёт речь в Главе 12, а о свёртке с использованием БПФ — в Главе 18. В этой главе мы рассмотрим, как происходит свёртка сигналов при умножении их спектров.

Для начала нам необходимо определить, что означает перемножение двух сигналов в частотной области, т. е. как интерпретируется запись:  $X[f] \times H[f] = Y[f]$ . В полярных координатах это выражение сводится к умножению модулей:  $Mag Y[f] = Mag X[f] \times Mag H[f]$  и сложению фаз  $Phase Y[f] = Phase X[f] + Phase H[f]$ . Проанализируем эти выражения на примере. Пусть на вход системы поступает косинусоидальный сигнал с некоторой амплитудой и фазой. На выходе системы тоже окажется косинусоидальный сигнал с другими значениями амплитуды и фазы. Полярная форма записи частотной характеристики системы напрямую описывает взаимосвязь значений амплитуды и фазы входного и выходного сигналов.

Если для умножения спектров используется прямоугольная форма записи, то между действительной и мнимой частями появляются перекрёстные члены. Поэтому, например, синусоида, поданная на вход системы, может дать на выходе как синусоидальный, так и косинусоидальный сигнал. Запись для умножения сигналов в частотной области в прямоугольной форме выглядит так:

$$\begin{aligned} Re Y[f] &= Re X[f] \times Re H[f] - Im X[f] \times Im H[f], \\ Im Y[f] &= Im X[f] \times Re H[f] + Re X[f] \times Im H[f]. \end{aligned} \quad (9.1)$$

Умножение сигналов в частотной области, записанное в прямоугольной форме:  $Y[f] = X[f] \times H[f]$ .

Для анализа умножения сигналов в частотной области будем использовать полярную форму записи и идею прохождения через систему простых косинусных колебаний. При этом будем подразумевать, что использование в данном случае менее наглядной прямоугольной формы даёт точно такие же результаты. Например, процедура деления сигналов в частотной области при использовании полярной формы записи выглядит достаточно просто. Выполняются действия, обратные тем, что необходимо выполнить при умножении сигналов. А именно: для вычисления  $H[f] = Y[f] / X[f]$  следует выполнить деление модулей и вычитание фаз, т. е.  $Mag H[f] = Mag Y[f] / Mag X[f]$ ,  $Phase H[f] = Phase Y[f] - Phase X[f]$ . В то же время в прямоугольной форме то же действие описывается следующим выражением:

$$\begin{aligned} Re H[f] &= \frac{Re Y[f] Re X[f] + Im Y[f] Im X[f]}{Re X[f]^2 + Im X[f]^2}, \\ Im H[f] &= \frac{Im Y[f] Re X[f] - Re Y[f] Im X[f]}{Re X[f]^2 + Im X[f]^2}. \end{aligned} \quad (9.2)$$

Деление сигналов в частотной области, записанное в прямоугольной форме:  $H[f] = Y[f] / X[f]$ .

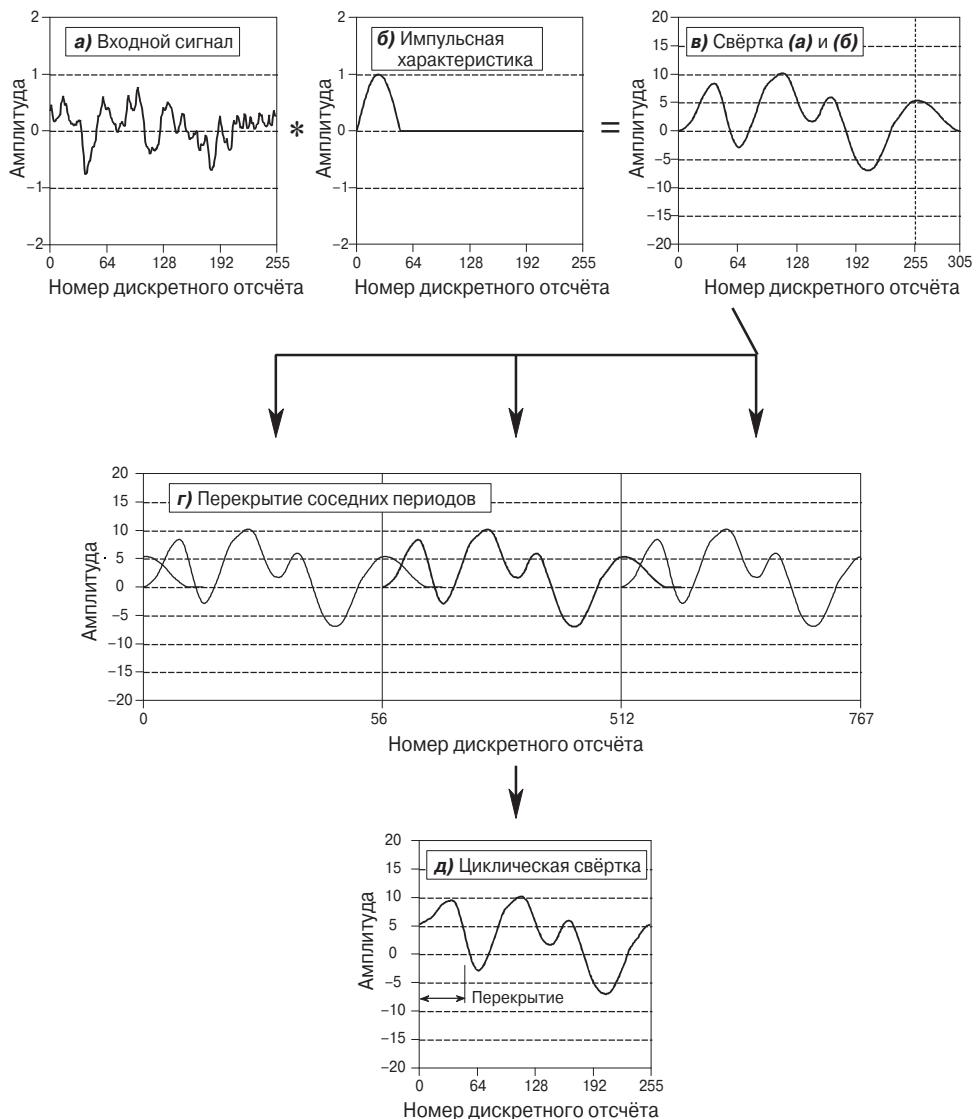
Но вернёмся к свёртке в частотной области. Вы, возможно, заметили, что на Рис. 9.8 мы пошли на некоторые уловки. Вспомним, что свёртка входного сигнала длиной  $N$  отсчётов с импульсной характеристикикой длиной  $M$  отсчётов формирует выходной сигнал, состоящий из  $N + M - 1$  отсчётов. Чтобы такое «удлинение» сигнала было возможно, нам пришлось дополнить входной сигнал нулевыми отсчётыами. Сигнал, показанный на (а), имеет 453 ненулевых отсчёта, а сигнал на (г) — 60. Свёртка этих сигналов даёт требуемые 512 отсчётов в выходном сигнале (ж).

Рассмотрим более общий случай, проиллюстрированный на Рис. 9.9. Входной сигнал (а) имеет длину 256 дискретных отсчётов, а импульсная характеристика системы (б) содержит только 51 ненулевой отсчёт. Результатом свёртки таких сигналов должен быть новый сигнал длиной 306 дискретных отсчётов (в). Однако при реализации свёртки в частотной области с использованием ДПФ длина выходного сигнала может состоять только из 256 отсчётов. Для перевода входного сигнала и импульсной характеристики в частотную область используются 256-точечные преобразования Фурье. После перемножения спектров получается сигнал, включающий 256 отсчётов. Обратное преобразование Фурье такого сигнала также даёт 256 дискретных отсчётов выходного сигнала во временной области. Каким же образом 306 отсчётов, получаемых при реализации свёртки во временной области, могли «сжаться» до 256, получаемых при использовании частотной области? Ответ: никак! Этот 256-точечный результат является искажённой версией правильного сигнала. Этот процесс называется *круговая (циклическая) свёртка*. Понимание принципа круговой свёртки очень важно, чтобы уметь избавиться от возможных негативных последствий.

Чтобы разобраться с циклической свёрткой, необходимо вспомнить, что  $N$ -точечное ДПФ рассматривает сигналы как бесконечные и периодические с периодом  $N$  дискретных отсчётов. На (г) показано три периода того сигнала, который с точки зрения ДПФ поступает в данном примере на вход системы. Поскольку  $N = 256$ , каждый период состоит из 256 отсчётов: с 0-го по 255-й; с 256-го по 511-й и с 512-го по 767-й. Свёртка в частотной области получает правильные 306 выходных отсчётов (в) для каждого периода повторения. В результате 49 крайних правых выходных отсчётов оказываются в соседнем периоде повторения, где они накладываются на отсчёты, «законно» находящиеся в этом же периоде. Перекрывающиеся отсчёты суммируются, что приводит к искажению выходного сигнала (д) и выполнению круговой свёртки.

Поняв суть циклической свёртки и её природу, становится ясно, как избежать её негативных последствий. Достаточно дополнить каждый из сворачиваемых сигналов необходимым числом нулевых отсчётов, чтобы в выходном сигнале уместились правильные  $N + M - 1$  отсчётов. Так, например, сигналы, показанные на (а) и (б), следует дополнить нулями до 512 отсчётов и использовать 512-точечное ДПФ. Выходом свёртки в частотной области в этом случае будет сигнал, включающий 306 правильных отсчётов и 206 отсчётов с нулевым значением. Детально данная процедура рассматривается в Главе 18.

Почему такая свёртка получила название циклической? Вернитесь к (г) и взгляните на средний период сигнала (отсчёты 256...511). Так как все периоды абсолютно одинаковы, те отсчёты, которые уходят вправо за границу текущего периода, окажутся в этом же периоде в левой его части. Если рассматривать только



**Рис. 9.9.** Циклическая свёртка. Входной сигнал (**а**) длиной 256 дискретных отсчётов сворачивается с импульсной характеристикой (**б**) длиной 51 отсчёт, формируя выходной сигнал (**в**), состоящий из 306 отсчётов. Если эта свёртка реализуется в частотной области с использованием 256-точечных ДПФ, длина выходного сигнала правильной свёртки (306 дискретных отсчётов) не согласуется с 256-точечным выходом ДПФ. Отсчёты выходного сигнала 256...305 оказываются в следующем периоде повторения выходного сигнала, где они накладываются на крайние левые отсчёты этого периода (**г**). На (**д**) показан один период выходного сигнала с перекрытием периодов.

один период сигнала (**д**), получается, что правый его край как бы соединён с левым краем. Это похоже на змею, кусающую себя за хвост. Следующим за 255-м является 0-й отсчёт, точно так же, как за 100-м отсчётом следует 101-й. Если отсчёты сигнала начинают выходить за его правую границу, они тут же будут посту-

пать с левого его конца. Таким образом, фрагмент из  $N$  отсчётов во временной области оказывается «зацикленным».

В предыдущей главе мы задались вопросом, имеет ли в действительности значение как рассматривается входной сигнал ДПФ: как конечный сигнал, состоящий из  $N$  отсчётов, или как один  $N$ -точечный период сигнала бесконечной длины? Циклическая свёртка — это один из примеров ситуации, когда такое различие в представлении сигнала играет значительную роль. Если рассматривать сигнал как бесконечный периодический, искажение, вызванное циклической свёрткой, можно лишь объяснить как «расширение» сигнала от одного периода к другому. Если же сигнал считается ограниченным  $N$  отсчётами, интерпретация циклической свёртки оказывается гораздо более интересной. Свёртка в частотной области представляет входной сигнал «свёрнутым в кольцо» так, что за его  $(N - 1)$ -м отсчётом следует 0-й.

## СВОЙСТВА ПРЕОБРАЗОВАНИЯ ФУРЬЕ

Описание *во временной и в частотной областях* — это две альтернативные формы представления сигналов. *Преобразование Фурье* является математической процедурой, позволяющей установить связь между ними. Если сигнал во временной области претерпевает некоторые изменения, то и в частотной области он будет модифицирован, однако обычно несколько иным образом. Так, например, мы знаем, что свёртка во временной области оказывается эквивалентна умножению в частотной области. Сложению, умножению на постоянный множитель, сдвигу и другим подобным процедурам также можно поставить в соответствие определённые математические действия, реализуемые в частотной области. Правила перехода от одной математической процедуры, выполняемой во временной области, к другой эквивалентной процедуре, выполняемой в частотной области, или наоборот, называются *свойствами Фурье-преобразования*.

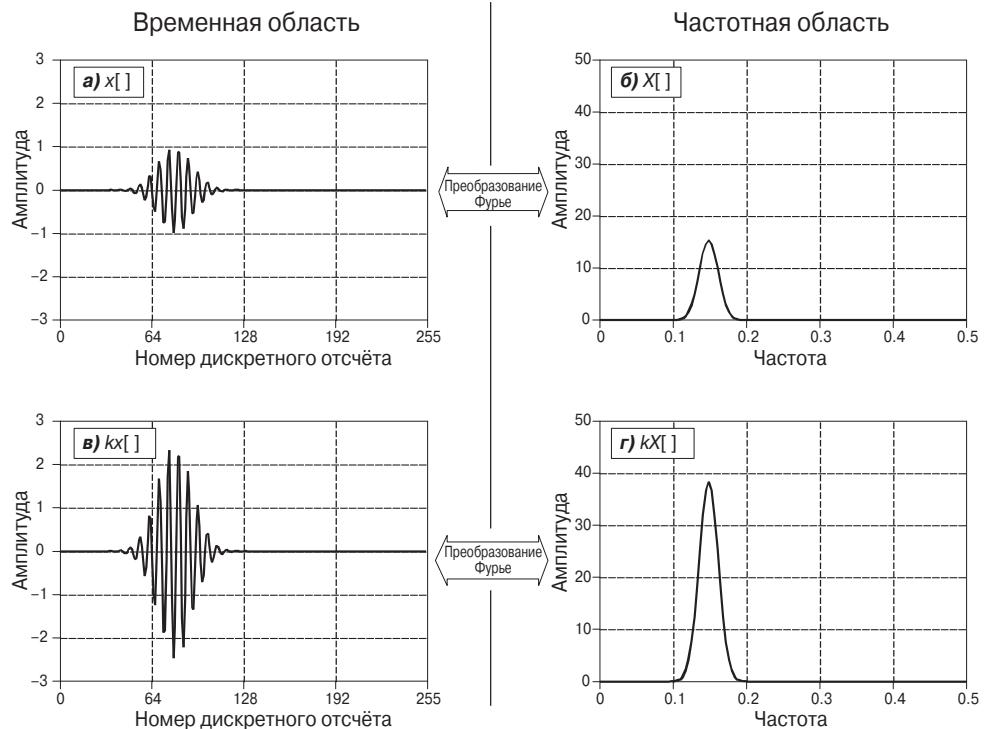
### 10.1. Линейность преобразования Фурье

*Преобразование Фурье* является *линейным*. Это означает, что оно обладает свойствами *однородности* и *аддитивности*. Данное утверждение справедливо для всех четырёх видов Фурье-преобразования (преобразование Фурье, ряд Фурье, ДПФ и преобразование Фурье дискретного времени).

**Рис. 10.1** иллюстрирует свойство однородности Фурье-преобразования. На **(а)** показан сигнал, представленный во *временной области*. Спектральное представление сигнала приведено на **(б)**. Обозначим сигнал и его спектр соответственно  $x[n]$  и  $X[k]$ . Однородность означает, что если амплитуда сигнала  $x[n]$  (временного представления) изменилась на некоторую величину, то амплитуда Фурье-образа  $X[k]$  (частотного представления) также изменится на ту же величину. Это должно быть понятно интуитивно: за изменением амплитуды временного сигнала на определённую величину должно стоять изменение амплитуд представляющих его синусных и косинусных сигналов, дающих в совокупности ту же величину.

Более строго это свойство записывается следующим образом. Если сигнал  $x[n]$  и его спектр  $X[k]$  — это пара Фурье-преобразований, то сигнал  $kx[n]$  и спектр  $kX[k]$  также являются парой Фурье-преобразований при любых значениях постоянной величины  $k$ . Если сигнал в *частотной области* представлен в прямоугольной системе координат, запись  $kX[k]$  означает, что на величину  $k$  умножаются действительная и мнимая части. В случае полярной системы координат на  $k$  умножается только модуль, а фаза остаётся без изменений.

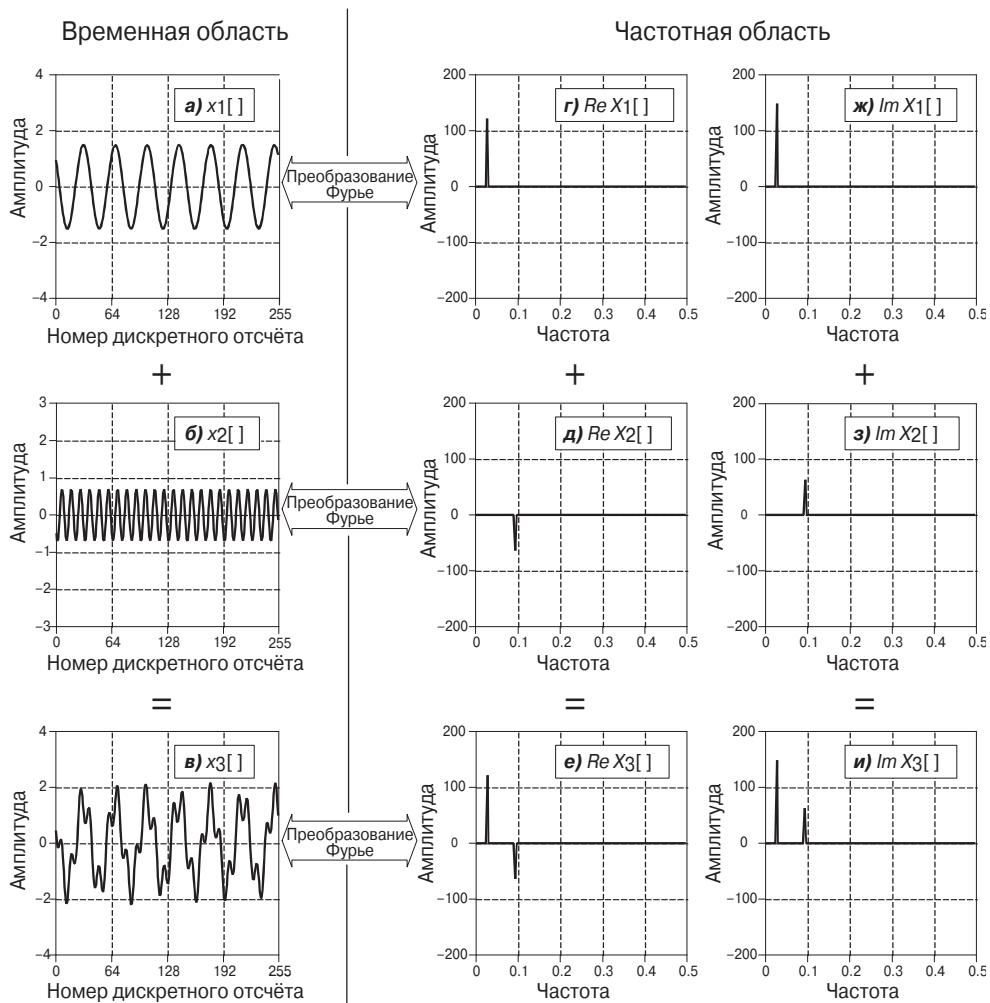
*Аддитивность* Фурье-преобразования означает, что сложение в одной области представления сигнала эквивалентно сложению в другой области. Данное свой-



**Рис. 10.1.** Свойство однородности преобразования Фурье. Если в одном представлении (временном или частотном) амплитуда сигнала изменяется на некоторую величину, то на ту же величину она должна измениться и в другом представлении. То есть умножение во временной области на постоянный множитель эквивалентно умножению на тот же постоянный множитель.

ство проиллюстрировано на **Рис. 10.2**. На (а и б) показаны два сигнала во временной области —  $x_1[ ]$  и  $x_2[ ]$ . Сумма этих сигналов даёт новый сигнал —  $x_3[ ]$  (в). В частотной области спектр каждого из этих трёх сигналов представлен действительной и мнимой частями, показанными на (г...и). Если во временной области два сигнала суммируются и дают третий, то в частотной области суммируются их спектры, формируя спектр третьего сигнала. В *прямоугольных координатах* сложение спектров означает суммирование их действительных и мнимых частей. То есть  $x_1[n] + x_2[n] = x_3[n]$  эквивалентно  $\operatorname{Re} X_1[f] + \operatorname{Re} X_2[f] = \operatorname{Re} X_3[f]$  и  $\operatorname{Im} X_1[f] + \operatorname{Im} X_2[f] = \operatorname{Im} X_3[f]$ . Вспомним, что сигнал представляется набором синусных и косинусных колебаний. Все косинусоиды (действительные части) и все синусоиды (мнимые части) суммируются независимо.

*Частотные спектры*, представленные в *полярных координатах*, в непосредственном виде не могут быть сложены. Их следует перевести в *прямоугольную* систему координат, просуммировать, а затем вновь вернуться к полярным координатам. Почему это так, тоже можно пояснить на основе синусно-косинусных компонентов Фурье-декомпозиции. Представьте, что суммируются два синусоидальных сигнала с одинаковой частотой, но произвольными амплитудами ( $A_1$  и



**Рис. 10.2.** Свойство аддитивности преобразования Фурье. Сложение двух или более сигналов в одной области (временной или частотной) эквивалентно их сложению в другой области. На (а и б) показаны два сигнала —  $x_1[n]$  и  $x_2[n]$ , которые суммируются и формируют третий сигнал —  $x_3[n]$  (в). Это эквивалентно сложению действительных и мнимых частей спектров этих сигналов.

$A_2$ ) и фазами ( $\phi_1$  и  $\phi_2$ ). Если фазы окажутся равными ( $\phi_1 = \phi_2$ ), то при сложении сигналов амплитуды суммируются ( $A_1 + A_2$ ). Если же фазы противоположны по знаку ( $\phi_1 = -\phi_2$ ), то амплитуды необходимо вычитать ( $A_1 - A_2$ ). Уже отсюда следует, что в полярных координатах сложение частотных спектров не может быть выполнено простым суммированием их модулей и фаз.

Несмотря на то что преобразование Фурье обладает свойством линейности, оно не является инвариантным к сдвигу. То есть сдвиг сигнала во временной области не эквивалентен сдвигу в частотной. Рассмотрим этот вопрос более детально.

## 10.2. Свойства фазовой характеристики

Если сигнал  $x[n]$  в частотной области описывается в полярной системе координат модулем *Mag X[f]* и фазой *Phase X[f]*, то сигналу  $x[n + s]$ , сдвинутому во времени на  $s$  отсчётов относительно исходного положения, в частотной области будут соответствовать модуль *Mag X[f]* и фаза *Phase X[f] + 2πsf*, где  $f$  — приведённая частота, выраженная в долях от *частоты дискретизации* и лежащая в диапазоне 0...0.5. Другими словами, если сигнал во временной области сдвинут на  $s$  дискретных отсчётов, то в частотной области модуль остаётся без изменений, а к фазе добавляется линейный член  $2πsf$ . Рассмотрим это на примере.

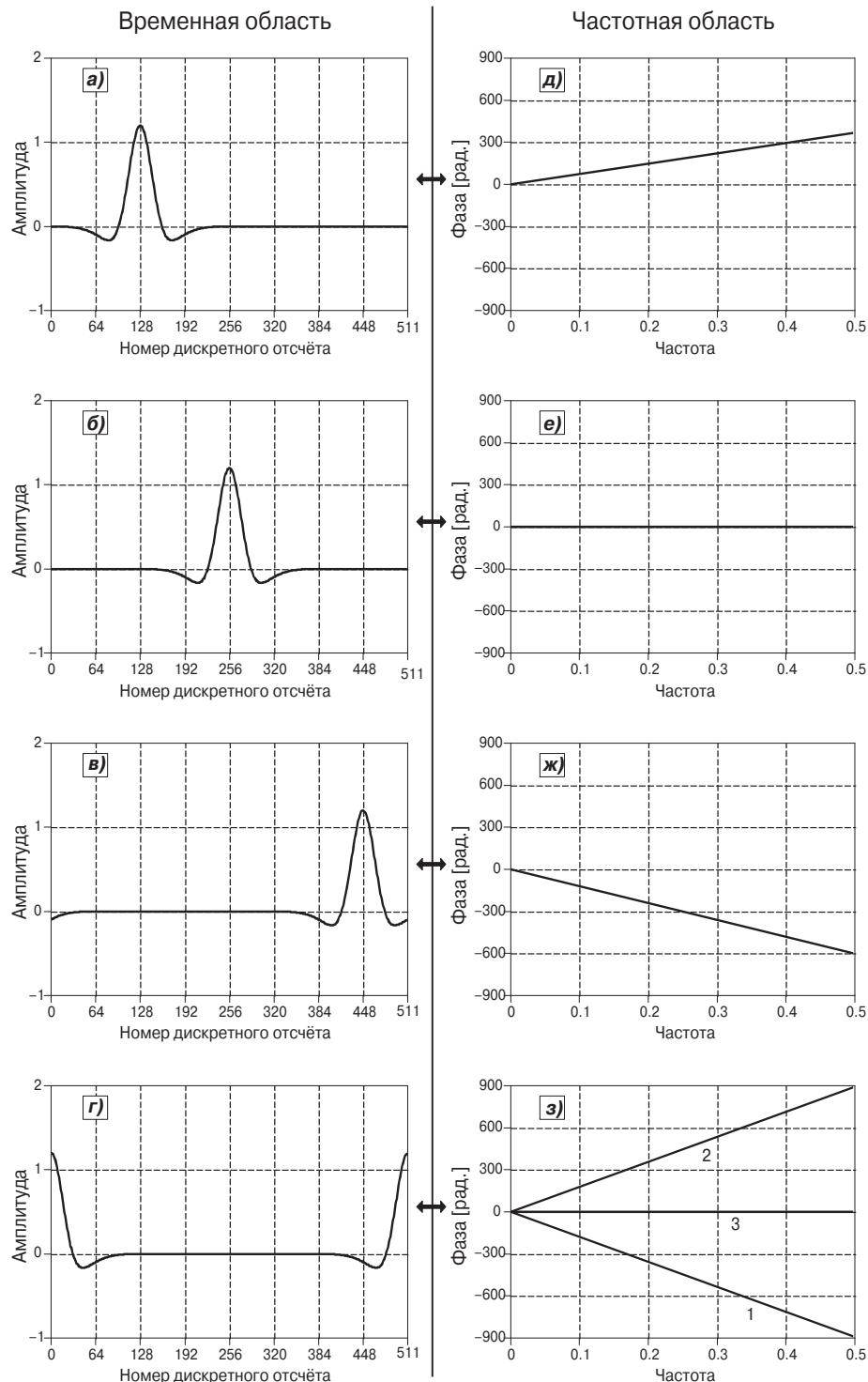
**Рис. 10.3** иллюстрирует, какое влияние на фазу оказывает сдвиг сигнала во временной области влево и вправо. Модуль на рисунке не показан, так как он в данном случае не представляет интереса: сдвиг во времени на него не влияет. На **(а...г)** изображён сигнал во временной области с различным положением на временной оси. От рисунка к рисунку сигнал последовательно сдвигается, так, что основной его пик перемещается со 128-го на 0-й отсчёт. Здесь учитывается, что ДПФ «рассматривает» сигнал во временной области как циклический: когда часть сигнала выходит за правую границу интервала наблюдения, она появляется с левой стороны.

Рассматриваемый сигнал симметричен относительно вертикальной оси: его правая и левая части являются зеркальными отражениями друг друга. Как было сказано в Главе 7, сигналы с такой симметрией являются сигналами с *линейной фазой*: фазовая составляющая частотного спектра представляет собой прямую линию. В свою очередь сигналы, не обладающие данным свойством симметричности, оказываются сигналами с *нелинейной фазой* и их фазовая характеристика отлична от прямой. На **(д...з)** показаны фазы сигналов, изображённых на **(а...г)**. В Главе 7 говорилось, что фазовая характеристика сигнала может быть «развёрнута», т. е. идти непрерывно без скачкообразных изменений, связанных с ограничением значения фазы диапазоном  $-\pi \dots \pi$ .

При сдвиге сигнала во временной области вправо фаза сигнала остаётся прямой линией, но угол её наклона уменьшается. Если сигнал во временной области сдвигается влево — угол наклона увеличивается. Это основное положение, которое вам следует запомнить: *сдвиг во временной области приводит к изменению угла наклона фазовой характеристики*.

На **(б и г)** показана особая ситуация, когда фазовая характеристика оказывается равной нулю. Такая ситуация наблюдается при полной симметричности сигнала во временной области относительно нулевого отсчёта. Симметричность сигнала на **(б)** относительно нуля не является очевидной на первый взгляд. Кажется, что сигнал симметричен относительно точки 256 (т. е.  $N/2$ ). Однако вспомним, что ДПФ считает сигналы циклическими, и их 0-й отсчёт непосредственно следует за  $(N - 1)$ -м отсчётом. Поэтому любой сигнал, симметричный относительно нуля, будет также всегда симметричен относительно 256-го отсчёта, и наоборот. Если же вы имеете дело с такими членами семейства Фурье-преобразований, которые не считают сигналы периодическими (как, например, преобразование Фурье дискретного времени), фаза оказывается нулевой в случае симметричности только относительно нулевого отсчёта.

Ситуация, проиллюстрированная на **(г и д)**, на первый взгляд непонятна. Если считать, что сигнал, изображённый на **(г)**, получен из сигнала **(б)** путём дополнени-



**Рис. 10.3.** Влияние сдвига сигнала во временной области на его фазовую характеристику.

тельного сдвига вправо, то фазовая характеристика на (з) должна иметь ещё более крутой отрицательный наклон по сравнению с (ж). Этому случаю соответствует кривая 1 на (з). Если же считать, что сигнал (г) получен из сигнала (а) сдвигом влево, то фазовая характеристика на (з) должна иметь положительный наклон, более крутой по сравнению с (д). Этой ситуации соответствует кривая 2 (з). Кроме того, можно отметить, что сигнал на (г) является симметричным относительно отсчёта  $N/2$ . А это означает, что он имеет нулевую фазовую характеристику (кривая 3 (з)). Какая же из этих трёх фазовых характеристик является правильной? Ответ связан с разрешением  $\pi$ - и  $2\pi$ -неопределённости фазы, о чём шла речь в Главе 8. Значения фазовой характеристики 2 отличаются от значений фазовой характеристики 1 в моменты частотных выборок ровно на целое число периодов  $2\pi$ . Аналогично кривая 3 связана с кривыми 1 и 2  $\pi$ -неопределённостью.

Чтобы понять описанное поведение фазы сигнала, рассмотрим сдвиг сигнала вправо на один дискретный отсчёт. Такой сдвиг означает, что все гармонические сигналы, на которые можно разложить исходный сигнал, также сдвигаются на один отсчёт вправо. Две синусоиды, которые могут входить в состав такого разложения, показаны на Рис. 10.4. Частота синусоиды, изображённой на (а), невелика, и один дискретный отсчёт составляет достаточно малую часть её периода. Синусоида на (б), напротив, имеет максимально возможную для дискретного сигнала частоту, равную половине частоты дискретизации. Сдвиг такого сигнала на один дискретный отсчёт означает смещение на половину периода, т. е. на  $\pi$  радиан. Таким образом, если величину сдвига рассматривать с точки зрения изменения фазы, то она становится пропорциональна частоте сдвигаемого синусоидального сигнала.

Рассмотрим, например, сигнал, имеющий симметричную относительно нулевого отсчёта форму, т. е. обладающий нулевой фазовой характеристикой. Рис. 10.5а иллюстрирует, как меняется фаза такого сигнала, когда он сдвигается во временной области вправо или влево. На максимальной частоте, равной половине частоты дискретизации, сдвиг сигнала влево на один дискретный отсчёт

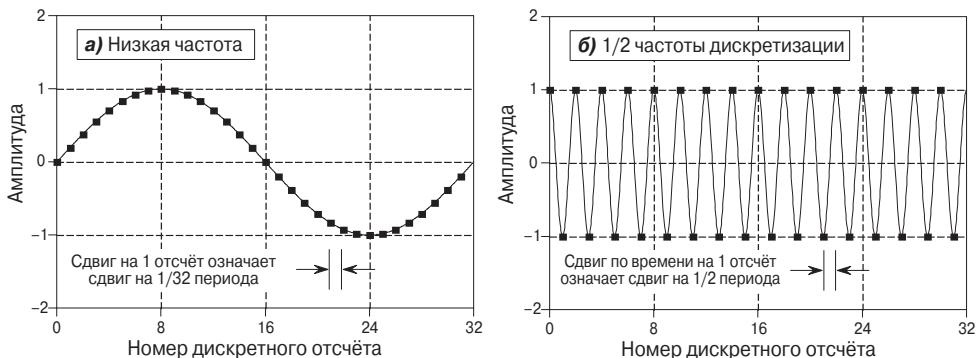
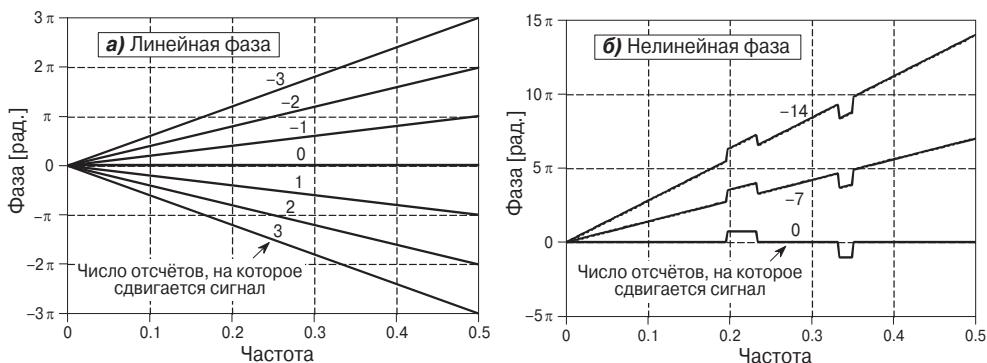


Рис. 10.4. Соотношения между моментами выборки и фазой сигнала. На (а) и (б) показаны низко- и высокочастотные синусоидальные сигналы соответственно. Для сигнала с низкой частотой сдвиг на один дискретный отсчёт означает смещение на  $1/32$  периода (а), а для сигнала с высокой частотой сдвиг на один отсчёт равносителен смещению на  $1/2$  периода (б). Это объясняет, почему сдвиг сигнала во временной области изменяет фазу сигнала на больших частотах на большую величину, чем на малых частотах.



**Рис. 10.5.** Фазовые характеристики, получаемые в результате сдвига сигнала во временной области. Сдвиг сигнала на каждый следующий дискретный отсчёт вправо приводит к уменьшению фазы на  $\pi$  радиан на частоте, равной половине частоты дискретизации. Сдвиг сигнала на каждый следующий дискретный отсчёт назад (влево) приводит к увеличению фазы на  $\pi$  радиан на этой максимальной частоте. (а) иллюстрирует пример линейной фазы (фазовая характеристика — прямая линия), а (б) — пример нелинейной фазы.

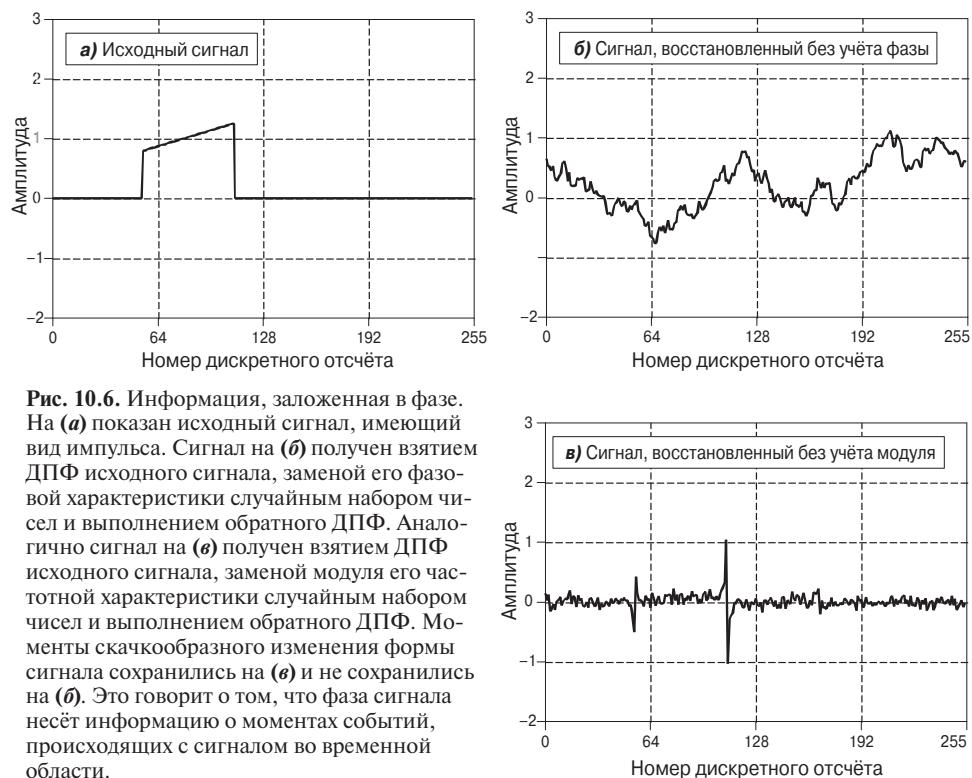
увеличивает фазу на  $\pi$  радиан, а сдвиг сигнала вправо на один отсчёт уменьшает фазу на  $\pi$  радиан. На нулевой частоте сдвиг не имеет смысла: фаза равна нулю. Значения фазы, лежащие на частотах между минимальной и максимальной, выстраиваются в прямую линию.

Все примеры, рассмотренные нами до сих пор, относились к сигналам с *линейной фазой*. Рис. 10.5б показывает, что в случае *нелинейной фазы* реакция фазовой характеристики на сдвиг во временной области остаётся такой же. В этом примере нелинейная фазовая характеристика имеет вид прямой с двумя прямоугольными всплесками. При сдвиге сигнала во времени нелинейные элементы фазовой характеристики сохраняются, просто располагаясь под другим углом.

А что же происходит с действительной и мнимой частями частотного представления сигнала в прямоугольных координатах при сдвиге во временной области? Напомним, что описание сигнала в частотной области в прямоугольной системе координат делает его очень сложным для человеческого восприятия. Обычно действительная и мнимая части выглядят как случайный сигнал малопонятной формы. При сдвиге во временной области эти и без того малопонятные колебания приобретают ещё более сложную для интерпретации форму. Не стоит искать в графиках действительной и мнимой частей какого-либо смысла и пытаться выявить влияние на их форму сдвига во временной области.

Посмотрим на Рис. 10.6. Он даёт интересную иллюстрацию того, какая информация закладывается в фазу сигнала, а какая — в его модуль. Сигнал на (а) имеет две характерные точки: на 55-й дискретный отсчёт приходится резкий скачок уровня сигнала, а на 110-й — спад. Такие перепады уровня сигнала имеют огромное значение в тех случаях, когда информация заложена в форме сигнала (*кодирование во временной области*). Они указывают на момент некоторого события, и весь сигнал делится на то, что было до него (слева от точки скачка), и то, что будет после него (справа от точки скачка). Это кодирование во временной области в чистом виде. Чтобы ответить на интересующий нас вопрос об информативности фазы и модуля сигнала, применим к сигналу на (а) ДПФ-преоб-

разование и представим его спектр в полярной системе координат. Заменим рассчитанную фазовую характеристику случайной функцией, значения которой лежат в диапазоне  $-\pi \dots \pi$ . Таким образом мы как бы «стёрли» информацию о фазе. Взяв обратное ДПФ, восстановим сигнал во временной области (**б**). Этот сигнал получен на основе только той информации, которая была заложена в модуле частотного представления. Аналогичным образом «стирая» модуль в частотном спектре сигнала, заменяя его набором случайных значений малого уровня и применяя затем обратное ДПФ, получим сигнал во временной области, восстановленный на основе только той информации, которая была заложена в фазе частотного представления (**в**).

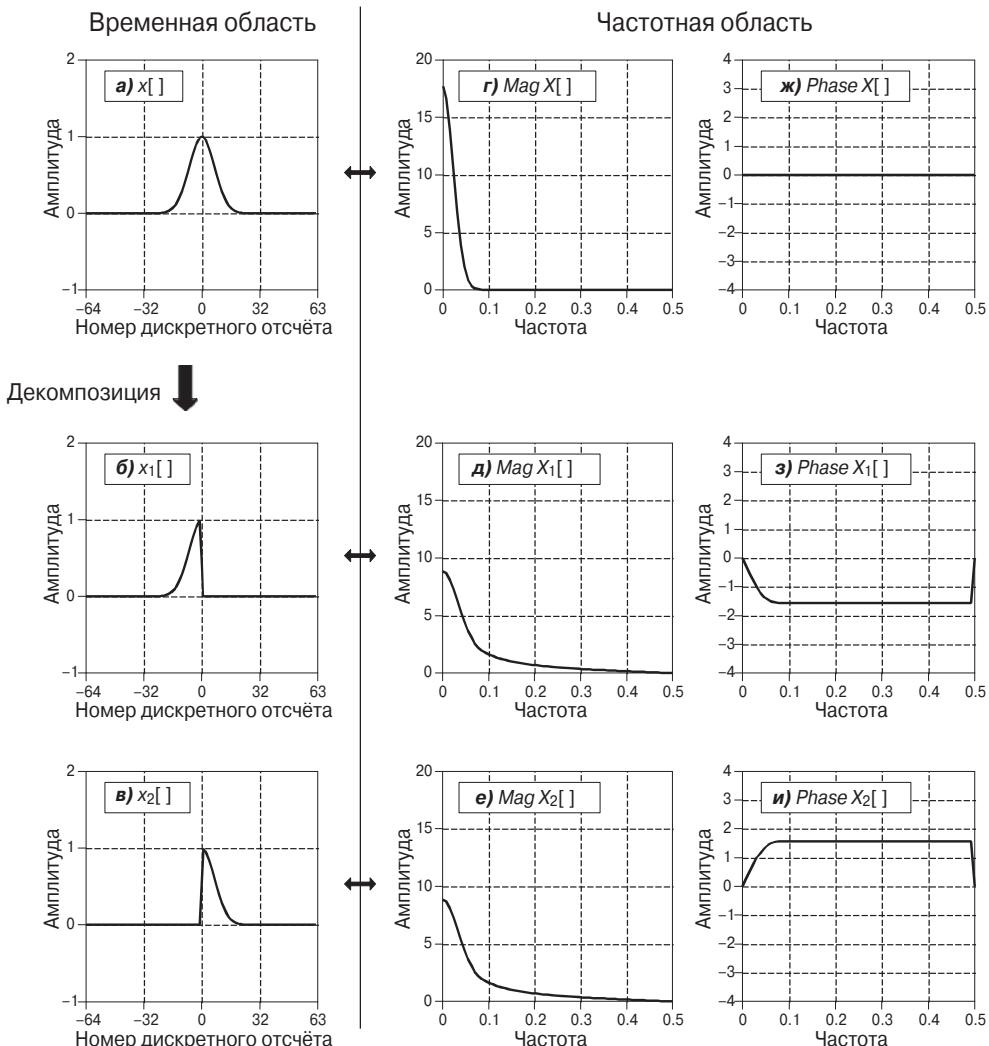


**Рис. 10.6.** Информация, заложенная в фазе. На (**а**) показан исходный сигнал, имеющий вид импульса. Сигнал на (**б**) получен взятием ДПФ исходного сигнала, заменой его фазовой характеристики случайным набором чисел и выполнением обратного ДПФ. Аналогично сигнал на (**в**) получен взятием ДПФ исходного сигнала, заменой модуля его частотной характеристики случайным набором чисел и выполнением обратного ДПФ. Моменты скачкообразного изменения формы сигнала сохранились на (**в**) и не сохранились на (**б**). Это говорит о том, что фаза сигнала несёт информацию о моментах событий, происходящих с сигналом во временной области.

Что же мы наблюдаем в результате? Точки скачкообразного изменения формы исходного сигнала полностью известны на (**в**), а на (**б**) эта информация отсутствует. Это объясняется тем, что скачок сигнала сопровождается одновременным повышением уровня большого числа гармоник, образующих сигнал, что возможно только при хорошей согласованности их фаз. Можно сделать вывод о том, что информация о форме сигнала во временной области переходит в частотной области в основном в фазу сигнала, а не в его модуль. Обратная ситуация будет наблюдаться с сигналами, несущими информацию в частотной области (*кодирование в частотной области*), например аудиосигналами. Основное значение для таких сигналов имеет модуль частотного представления, а фаза играет второстепенную роль. В будущих главах мы увидим, что такой подход к сигналам даёт возмож-

ность сформировать эффективные способы проектирования фильтров и обработки сигналов в целом. Понимание того, каким образом информация представлена в сигнале, — это всегда первый шаг на пути к успеху использования методов ЦОС.

Почему сигналы, обладающие свойством симметрии относительно вертикальной оси, имеют нулевую (или линейную) фазовую характеристику? Ответ поясняет **Рис. 10.7**. Сигнал такого типа можно подвергнуть декомпозиции — разложению на два компонента: левую и правую стороны, как показано на **(а...в)**. Центральный отсчёт (в данном случае — нулевой) должен так делиться между ле-



**Рис. 10.7.** Свойства фазы сигнала с левоправосторонней симметрией. Сигнал с левоправосторонней симметрией, изображённый на **(а)**, может быть представлен в виде суммы двух сигналов: левой **(б)** и правой **(в)** его сторон соответственно. Модули этих двух сигналов оказываются идентичны **(д, е)**, а фазы являются противоположными по знаку копиями друг друга **(ж, и)**.

вой и правой сторон, чтобы они были точным зеркальным отражением друг друга. Тогда модули частотного представления обоих компонентов разложения будут идентичны ( $\delta$ ,  $e$ ), а фазы — противоположны по знаку ( $z$ ,  $u$ ). Отсюда следуют два очень важных вывода. Во-первых, любой сигнал, имеющий симметричные правую и левую стороны, будет сигналом с линейной фазой, так как любая нелинейность фазы правостороннего сигнала всегда будет компенсироваться нелинейностью фазы левостороннего.

Во-вторых, представим, что сигнал на ( $\epsilon$ ) получен из сигнала на ( $\delta$ ) «разворотом» вокруг вертикальной оси. Получается, что подобный «разворот», выполняемый во временной области, не меняет модуль частотного представления сигнала, но меняет знак каждого отсчёта фазы. Верно и обратное: изменение знака каждого отсчёта фазы на противоположный приводит к «развороту» сигнала вокруг вертикальной оси во временной области. При этом, если сигналы являются *непрерывными*, «разворот» происходит только относительно нулевого отсчёта. Если же сигналы *дискретные*, «разворачивание» осуществляется одновременно относительно нулевого отсчёта и относительно отсчёта  $N/2$ .

Смена знака фазы — настолько часто используемая процедура, что она даже получила собственное наименование и обозначение. Её называют *комплексным сопряжением* и используют символ «звёздочка» в качестве верхнего индекса в обозначении сигнала. Например, если спектр  $X[f]$  представляется в виде модуля  $Mag X[f]$  и фазы  $Phase X[f]$ , то спектр  $X^*[f]$  является комплексно-сопряжённым ему и описывается модулем  $Mag X^*[f]$  и фазой  $Phase X^*[f]$ . В прямоугольной системе координат процедура комплексного сопряжения осуществляется путём смены знака у мнимой части на противоположный; действительная часть при этом не меняется. То есть если сигнал в частотной области  $X[f]$  представляется в виде совокупности действительной  $Re X[f]$  и мнимой  $Im X[f]$  частей, то комплексно-сопряжённый сигнал  $X^*[f]$  будет включать действительную  $Re X^*[f]$  и мнимую  $Im X^*[f]$  части.

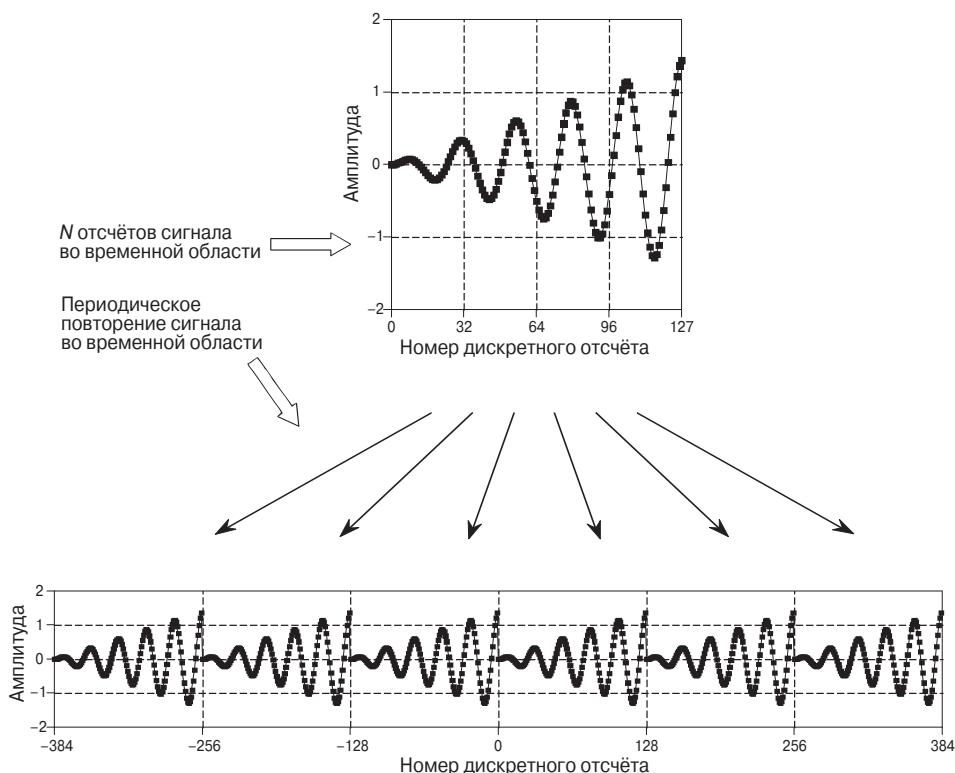
Приведём несколько примеров того, как комплексное сопряжение используется в ЦОС. Если сигналу  $x[n]$  соответствует преобразование Фурье  $X[f]$ , то сигналу  $x[-n]$  будет соответствовать преобразование Фурье  $X^*[f]$ . Иными словами, «разворот» сигнала вокруг вертикальной оси во временной области соответствует смене знака фазы в частотной области. В качестве другого примера можно привести случай, уже рассмотренный в Главе 7, — реализация *корреляции* с помощью *свёртки*. Эта операция как раз выполняется с помощью зеркального «разворота» одного из сигналов. В виде математических выражений это означает: если  $a[n] * b[n]$  — свёртка, то  $a[n] * b[-n]$  — корреляция. В частотной области те же выражения превращаются в  $A[f] \times B[f]$  и  $A[f] \times B^*[f]$  соответственно. И последний пример. Рассмотрим некоторый произвольный сигнал  $x[n]$  с соответствующим частотным представлением  $X[f]$ . Спектр такого сигнала можно преобразовать так, что он будет иметь *нулевую фазу*. Для этого достаточно умножить его на комплексно-сопряжённое:  $X[f] \times X^*[f]$ . Другими словами, какую бы фазу ни имел сигнал с частотным представлением  $X[f]$ , она может быть сведена к нулю простым суммированием с фазой, противоположной по знаку (вспомним, что при умножении спектров их фазовые составляющие суммируются). Во временной области это означает следующее: свёртка произвольного сигнала с собственной «перевёрнутой» вокруг вертикальной оси копией ( $x[n] * x[-n]$ ) всегда будет являться сигналом, симметричным относительно вертикальной оси, проходящей через ноль, независимо от характера симметричности исходного сигнала.

Для многих инженеров и учёных подобные подходы к работе с сигналами составляют суть ЦОС. Если вы хотите войти в их круг или хотя бы иметь возможность общения с этой группой специалистов, вы должны постараться освоить их язык.

## 10.3. Периодичность сигналов ДПФ

В отличие от трёх других членов семейства Фурье-преобразований *дискретное преобразование Фурье (ДПФ)* рассматривает сигнал как во *временной*, так и в *частотной области* периодическим. Это может казаться странным и не очень удобным, поскольку на практике сигналы, с которыми приходится работать, в действительности периодическими не являются. И тем не менее, если вы хотите «пользоваться услугами» ДПФ, вы должны уметь смотреть на мир его глазами.

На Рис. 10.8 показаны две различные интерпретации одного и того же сигнала. На верхнем рисунке сигнал представлен  $N$  дискретными временными отсчётаами. Именно в этом виде сигналы обычно поступают в систему обработки в реальных научных или инженерных экспериментах. Показанные 128 отсчётов могли бы быть получены, например, путём фиксации значений некоторого параметра в



**Рис. 10.8.** Периодичность дискретного представления сигнала во временной области. Сигнал во временной области может рассматриваться как сигнал конечной длины, состоящий из  $N$  дискретных отсчётов (верхний рисунок), или как бесконечный сигнал, у которого  $N$  отсчётов — это только один из бесконечного числа периодов повторения (нижний рисунок).

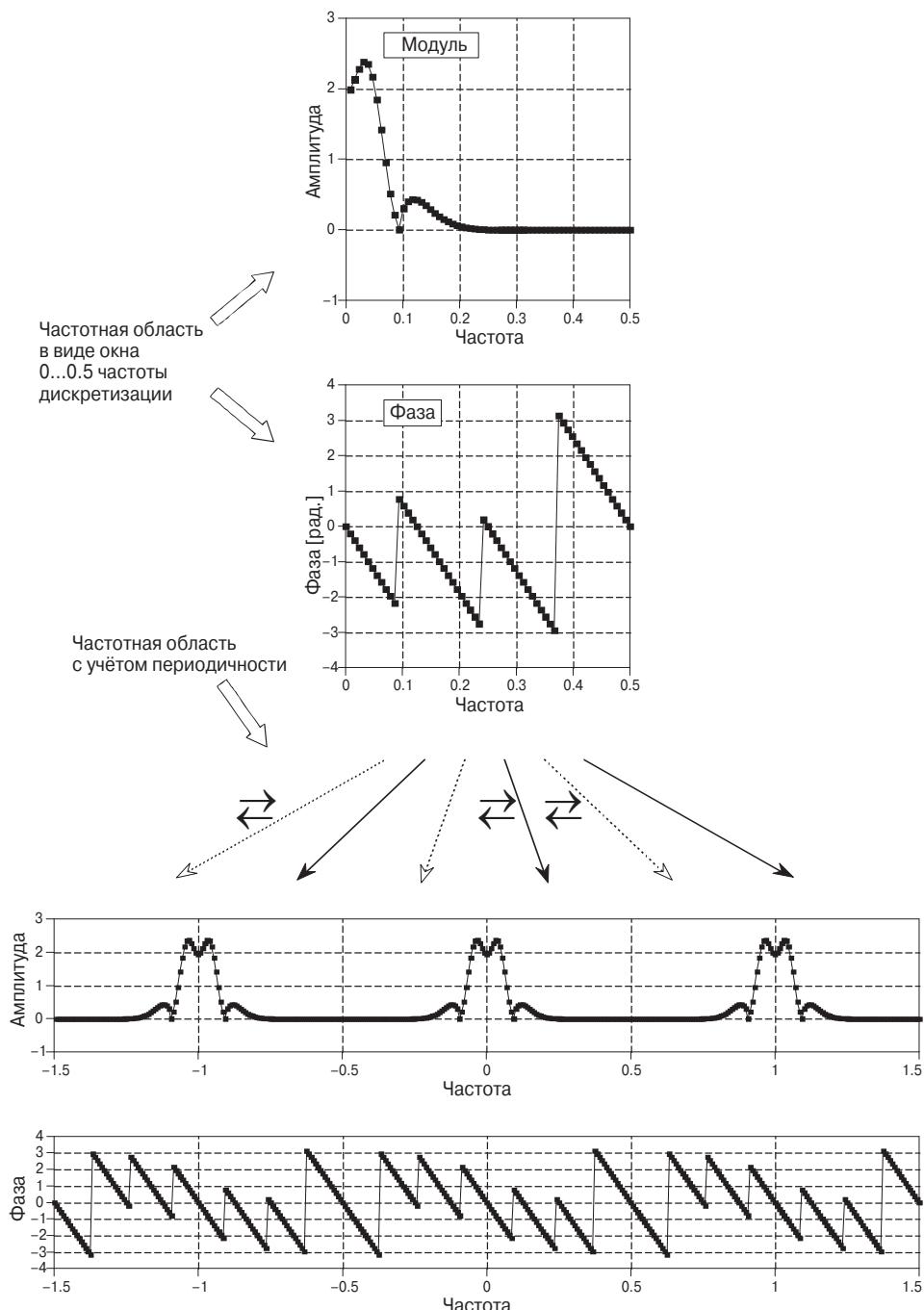
моменты времени, идущие с одинаковым шагом. Нулевой отсчёт отличается от 127-го и отделён от него, поскольку они взяты в разные моменты времени. В том виде, в котором данный сигнал был зарегистрирован, нет никаких оснований полагать, что крайний левый его отсчёт каким-либо образом связан с крайним правым.

К сожалению, ДПФ думает иначе. Оно рассматривает сигнал так, как показано на нижнем рисунке: исходные 128 отсчётов — это лишь один период сигнала бесконечной длины. А это означает, что левый край исходного сигнала является непосредственным продолжением его правого края, скопированного в предшествующий период повторения. Также и правый край связан с левым краем, перенесённым в следующий период повторения. В результате сигнал можно рассматривать «скрученным в кольцо», в котором правая граница соединяется с левой. Таким образом, 0-й отсчёт следует за 127-м точно так же, как 44-й следует за 43-м. Сигналы при таком рассмотрении являются *циклическими*, что абсолютно идентично рассмотрению таких сигналов как *периодических*.

Особенно серьёзным последствием представления сигналов как циклических является возможность *наложения во временной области*. Поясним это. Предположим, что есть сигнал во временной области, выполнив ДПФ которого мы получили его *частотный спектр*. Мы можем сразу же выполнить *обратное ДПФ* и восстановить исходный сигнал, однако это будет не очень интересно. Сначала, перед обратным ДПФ, изменим некоторым образом спектр сигнала. Предположим, мы удалим некоторые из частотных компонент, или изменим их амплитуду и фазу, или выполним какие-то иные преобразования. В ЦОС подобные манипуляции используются очень часто. В частотной области эти модификации вполне могут привести к увеличению длины сигнала во временной области сверх исходного периода повторения. Сигнал, таким образом, «перетечёт» в соседний период. А если вспомнить, что сигналы рассматриваются как циклические, то станет ясно, что распространение сигнала в следующий справа период означает его появление с левого края в этом же периоде. Таким образом, часть сигнала, оказавшаяся вне исходного периода повторения, будет во временной области накладываться на отсчёты сигнала в этом же периоде, искажая их. Возникает возможность потери информации. Такая ситуация характерна, например, для широко используемой в ЦОС циклической свёртки, выполняемой с помощью умножения в частотной области (см. Главу 9).

Периодичность сигналов в частотной области подобна рассмотренным особенностям, характерным для временной области, но оказывается сложнее их. Рассмотрим пример, приведённый на Рис. 10.9. На верхних рисунках показаны *модуль* и *фаза* частотного спектра сигнала, состоящие из  $N/2 + 1$  отсчётов, расположенных между нулём и частотой, равной половине *частоты дискретизации*. Это наиболее простое понимание структуры спектра, которое, однако, часто оказывается недостаточным для объяснения поведения сигнала при использовании ДПФ.

Нижние рисунки изображают, как ДПФ-преобразование рассматривает сигналы с учётом периодичности в частотной области. Ключевым моментом при этом является то, что спектр, лежащий в диапазоне 0...0.5 частоты дискретизации, имеет зеркальную копию в области отрицательных частот: 0...-0.5 частоты дискретизации. Перенос модуля и фазы спектра в область отрицательных частот осуществляется немного по-разному. Модуль только «разворачивается» относи-



**Рис. 10.9.** Периодичность дискретного представления сигналов в частотной области. Дискретное представление сигнала в частотной области может рассматриваться на интервале 0...1/2 частоты дискретизации (верхние два рисунка) или как бесконечный периодический сигнал, у которого каждый следующий фрагмент шириной 0...0.5 частоты дискретизации является перевёрнутой копией предшествующего фрагмента.

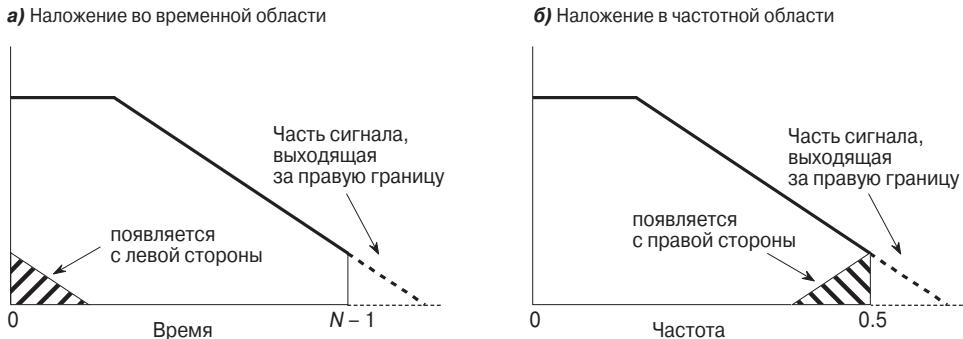
тельно вертикальной оси, проходящей через ноль, а фаза и «разворачивается», и меняет свой знак на противоположный. Выше мы уже упоминали, что есть *сигналы с чётной симметрией* (модуль частотного спектра) и *сигналы с нечётной симметрией* (фаза частотного спектра). Если спектр сигнала представлен в виде действительной и мнимой частей, то действительная часть всегда будет чётной функцией, а мнимая — нечётной.

С учётом наличия спектра в области отрицательных частот ДПФ-преобразование рассматривает сигналы в частотной области как периодически повторяющиеся. Период повторения при этом равен частоте дискретизации. В качестве одного периода сигнала в частотной области может выступать, например, диапазон  $-0.5\dots0.5$  или  $0\dots1.0$  частоты дискретизации. Иначе говоря, период повторения в частотной области составляет  $N$  дискретных отсчётов, т. е. равен периоду сигнала во временной области.

Периодичность спектра дискретного сигнала может приводить к *наложению в частотной области* аналогично наложению во временной области, рассмотренному выше. Допустим, имеется некоторый сигнал во временной области и его частотный спектр. Если сигнал каким-либо образом модифицируется во временной области, естественно, модифицируется и его спектр. При этом может оказаться, что изменение сигнала во временной области привело к расширению спектра за пределы выделенного диапазона и часть сигнала оказалась в соседнем периоде повторения. Как и ранее, это может привести к двум проблемам: частотные компоненты оказываются не там, где они должны были располагаться, а перекрытие спектров приводит к суммированию независимых компонент и искажает сигнал.

Наложение в частотной области сложнее проанализировать, чем наложение во временной области, поскольку форма частотного спектра более сложна. Рассмотрим единственную частотную составляющую, которая в результате некоторых преобразований смещается с 0.01 на 0.49 частоты дискретизации. При этом в области отрицательных частот произойдёт смещение соответствующей копии с  $-0.01$  на  $-0.49$  частоты дискретизации. Если в области положительных частот в результате смещения спектра какие-либо его составляющие оказались правее 0.5 частоты дискретизации, то в области отрицательных частот их копии будут располагаться левее  $-0.5$  частоты дискретизации. Из-за периодичности спектра то же самое произойдёт во всех остальных периодах, например в диапазоне  $0.5\dots1.5$  частоты дискретизации: правая составляющая спектра пересечёт границу 1.5, а левая составляющая — 0.5 частоты дискретизации. А теперь представьте, каким вы увидите спектр сигнала, рассматривая только окно  $0\dots0.5$  частоты дискретизации. Получается, что составляющая спектра, уходящая за правую границу периода, возвращается обратно, перемещаясь справа налево.

**Рис 10.10** иллюстрирует процесс наложения в частотной и временной области при рассмотрении только одного периода повторения сигнала. Если во временной области длина сигнала оказалась больше периода повторения и часть его отсчётов зашла за правую границу периода (**а**), то эта часть отсчётов «вырезается» и вставляется в левую часть сигнала в рассматриваемом окне. Если же в частотной области спектр сигнала оказывается шире периода повторения (**б**), то «выступающая» правая часть его спектра «заворачивается» внутрь рассматриваемого окна. Независимо от того, куда переносится вышедший за пределы периода фрагмент сигнала, он накладывается на уже имеющиеся там отсчёты и искажает информацию, заложенную в них.



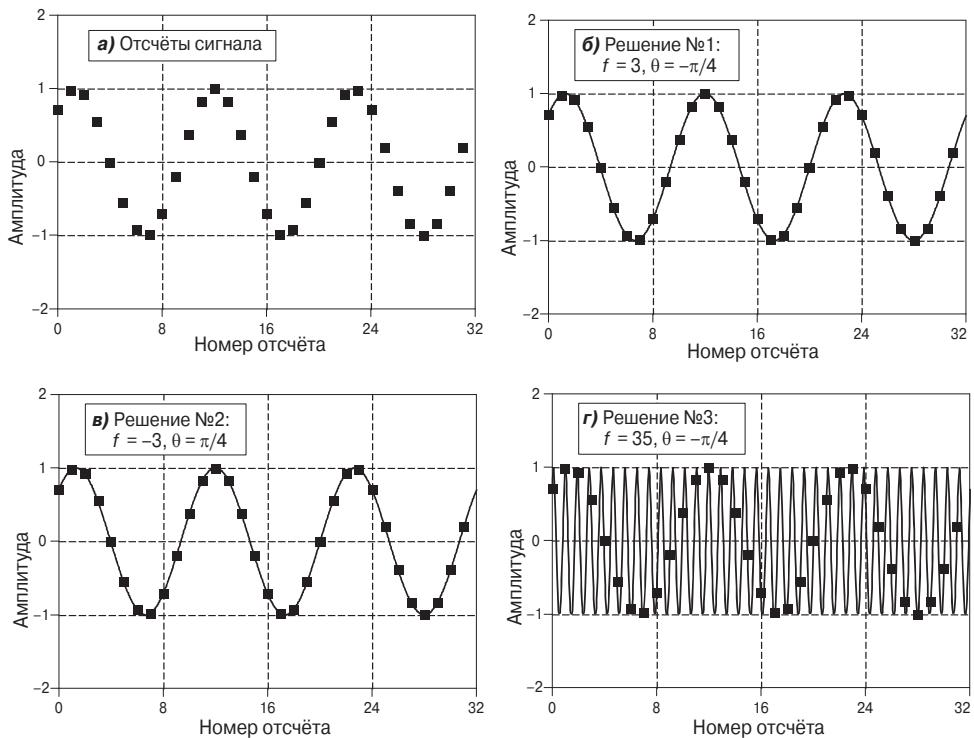
**Рис. 10.10.** Иллюстрация наложения во временной и частотной областях при рассмотрении только одного периода повторения сигнала. Во временной области (а) часть сигнала, выходящая за правую границу, появляется с левой стороны. В частотной области (б) часть сигнала, выходящая за правую границу, появляется с правой стороны так, как будто она заворачивается внутрь периода.

Обсудим более детально такое не очень понятное на первый взгляд понятие, как «отрицательные частоты». Является оно только искусственным математическим понятием или несёт какой-то физический смысл? Ответ на этот вопрос иллюстрирует Рис. 10.11. На (а) показан дискретный сигнал, состоящий из 32 отсчётов. Предположим, вы хотите построить спектр сигнала, соответствующий этим 32 точкам. Чтобы облегчить свою работу, вы скажете, что этот набор точек представляет дискретную синусоиду. Тогда вам следует найти такие значения частоты и фазы ( $f$  и  $\theta$ ), чтобы сигнал  $x[n] = \cos(2\pi n f/N + \theta)$  удовлетворял заданному набору точек. Мы уже рассматривали эту задачу и знаем, что искомый спектр имеет вид, показанный на (б). При этом  $f = 3$ , а  $\theta = -\pi/4$ .

Если вы остановите свой анализ на этом результате, знайте, что вы рассмотрели только треть проблемы. Дело в том, что есть ещё два решения, которые вы упустили. Одно из них:  $f = -3$ ,  $\theta = \pi/4$  (в). Даже если использованное здесь отрицательное значение частоты вас раздражает, вы должны принять тот факт, что с математической точки зрения это решение полностью справедливо для поставленной задачи. Любой гармонический сигнал с положительной частотой может также быть представлен гармоническим сигналом с отрицательной частотой. Это справедливо как для непрерывных, так и для дискретных сигналов.

Ещё одно решение представляет в действительности целое семейство бесконечного числа сигналов. На (г) показано, что синусоида с частотой  $f = 35$  и фазовым сдвигом  $\theta = -\pi/4$  проходит через все заданные дискретные отсчёты. То, что этот сигнал включает ещё ряд колебаний между соседними дискретными отсчётами, может казаться странным, но тем не менее это решение является правильным. Таким же образом и синусоиды с частотами  $f = \pm 29$ ,  $f = \pm 35$ ,  $f = \pm 61$  и  $f = \pm 67$  также будут являться решениями и включать колебания между соседними отсчётами исходного дискретного сигнала.

Все три решения относятся к разным частотным диапазонам. Для дискретных сигналов первое решение всегда лежит в диапазоне 0...0.5 частоты дискретизации. Второе решение располагается в области отрицательных частот на интервале 0...-0.5 частоты дискретизации. Наконец, третье решение представляет беско-



**Рис. 10.11.** Смысл понятия «отрицательные частоты». Ставится задача: построить спектр дискретного сигнала, показанного на (а). Это означает, что мы должны найти частоту и начальную фазу гармонического сигнала, который проходил бы через все заданные точки. Одно решение использует положительную частоту (б), в то время как другое — отрицательную (в). Сигнал на (г) представляет целое семейство решений поставленной задачи.

нечный ряд компонент, лежащих вне интервала  $-0.5...0.5$  частоты дискретизации. В случае непрерывных сигналов существуют только два решения. Первое располагается в области положительных частот, второе — в области отрицательных.

Многие алгоритмы ЦОС не требуют использования отрицательных частот, а также знаний о периодичности сигналов ДПФ. Это, например, два рассмотренных нами в прошлой главе примера спектрального анализа сигналов и использования частотной характеристики систем. Для этих приложений вполне достаточно рассматривать сигнал во временной области состоящим из  $N$  дискретных отсчётов  $0...(N - 1)$ , а в частотной области расположенным на интервале  $0...0.5$  частоты дискретизации. Такое упрощённое понимание ситуации применимо в данных случаях, поскольку в них исключается возможность выхода части сигнала за пределы периода повторения. Рассмотрение сигнала в пределах одного периода становится полностью идентичным его рассмотрению на бесконечно длинном окне.

В то же время многие процедуры требуют учёта возможного перекрытия периодов сигнала. Два примера таких ситуаций также уже были нами рассмотрены. Это *циклическая свёртка* и *аналого-цифровое преобразование*. В процессе цикли-

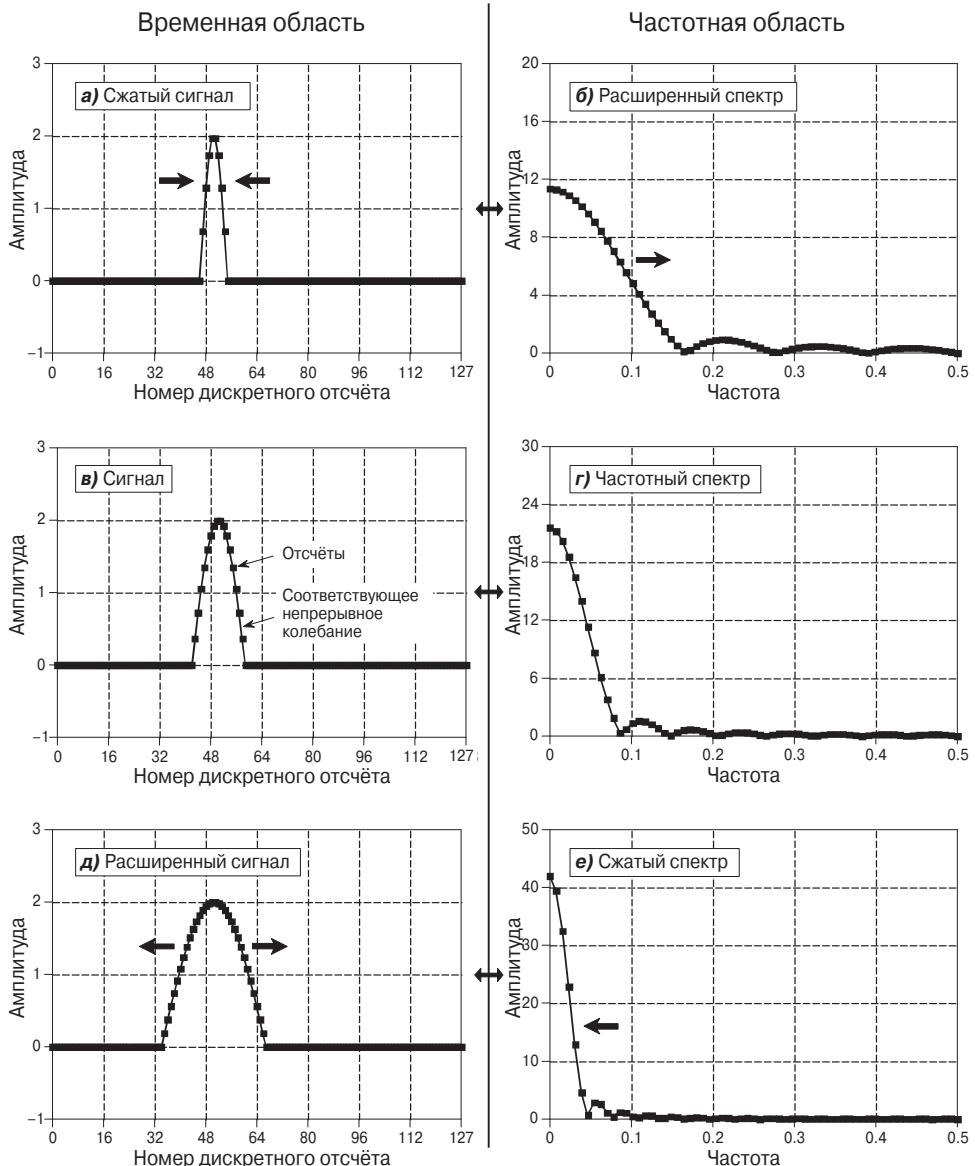
ческой свёртки два сигнала, спектры которых перемножаются в частотной области, сворачиваются во временной области. Если получаемый в результате временной сигнал оказывается длиннее исходного периода повторения, часть сигнала проникает в соседний период, вызывая наложение во временной области. Аналого-цифровое преобразование, напротив, является примером наложения в частотной области. Во временной области реализуется нелинейная операция — перевод сигнала из непрерывной формы в дискретную. Спектр исходного аналогового сигнала может оказаться шире спектра дискретного сигнала. Часть спектра, выходящая за границы периода, оказывается в соседнем периоде повторения. Рассмотрим ещё два примера, в которых понимание периодичности ДПФ играет существенную роль: сжатие/расширение сигналов и амплитудная модуляция.

## 10.4. Сжатие и расширение сигналов. Многоскоростная обработка

Как показано на Рис. 10.12, *сжатие* в одной области сопровождается *расширением* в другой, и наоборот. В случае *непрерывных сигналов*, если  $X(f)$  — это преобразование Фурье сигнала  $x(t)$ , то  $1/k \times X(f/k)$  — это преобразование Фурье сигнала  $x(kt)$ , где параметр  $k$  определяет степень расширения или сжатия. Если событие происходит быстрее (сигнал сжимается во временной области), то в его состав должны входить более высокие частоты. Если событие медленное (расширение во временной области), его представляют более низкие частоты. Это правило верно и для двух предельных случаев. Если во временной области сигнал сжат настолько, что он превращается в дельта-импульс, то его спектр расширяется так, что представляется постоянной величиной. Наоборот, если сигнал во временной области оказывается постоянной величиной (максимальное расширение во времени), в частотной области его спектр представляется дельта-функцией.

*Дискретные сигналы* ведут себя аналогично, но с некоторыми особенностями. Во-первых, им присущи *наложения*. Представьте, что сигнал на (a) сжат в несколько раз сильнее, чем это показано. Тогда *частотный спектр* будет в такое же число раз шире, и часть *боковых лепестков*, показанных на (b), окажется за пределами диапазона 0...0.5 *частоты дискретизации*. В результате возникнет *наложение спектров*, которое разрушает обычное введённое нами правило сжатия/расширения сигнала. Такое же наложение может появиться и во временной области. Допустим спектр, показанный на (e), оказывается намного *уже*. Тогда сигнал во временной области, изображённый на (d), расширится так, что часть сигнала попадёт в соседние периоды повторения.

Вторая особенность, касающаяся дискретных сигналов, связана с точным определением понятий *сжатия* и *расширения*. Сжатие дискретного сигнала подразумевает сжатие непрерывного сигнала, проходящего через заданные дискретные отсчёты, с последующей дискретизацией нового сжатого сигнала, в результате которой и получается искомый сигнал (a). Процесс расширения дискретного сигнала выполняется аналогично (d). При сжатии дискретного сигнала на длительность некоторого события (например, на длительность импульса) будет приходиться меньшее число *дискретных отсчётов*. Напротив, при расширении сигнала события занимают большее число отсчётов.

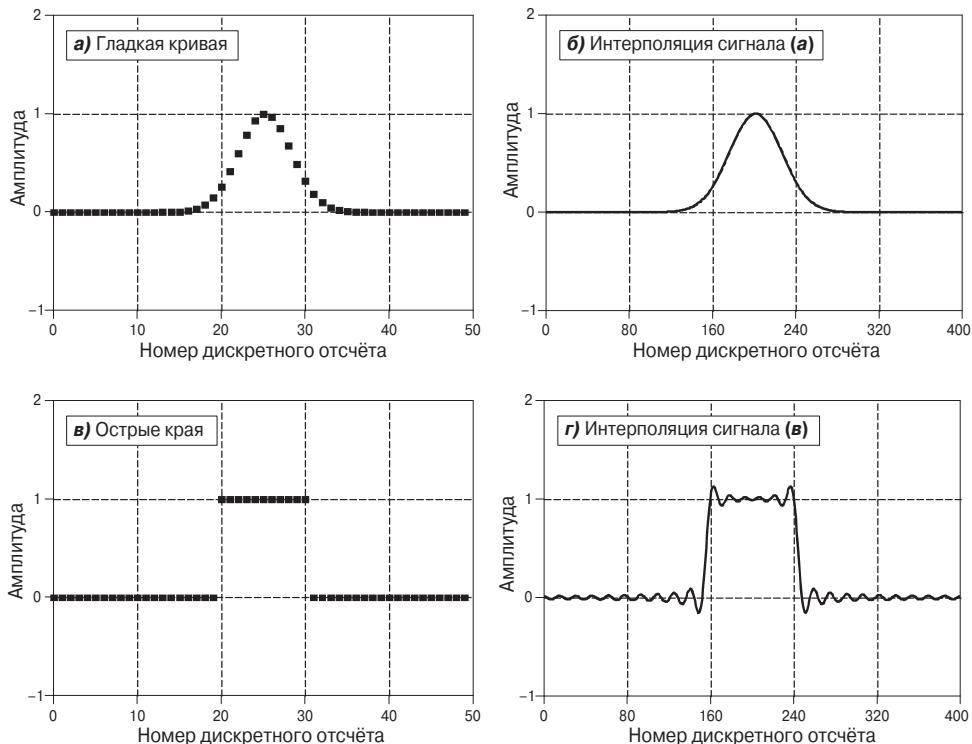


**Рис. 10.12.** Сжатие и расширение представлений сигналов. Сжатие сигнала в одной области приводит к расширению его представления в другой, и наоборот. На (в и г) показан дискретный сигнал и его спектр соответственно. Сжатие сигнала во временной области, сопровождающее расширением его спектра в частотной области, проиллюстрировано на (а и б). Обратный процесс показан на (д и е). На рисунках видно, что сжатие или расширение дискретного представления сигнала подразумевает соответствующее сжатие или расширение непрерывного сигнала, проходящего через заданный набор отсчётов, с последующей дискретизацией этого нового непрерывного сигнала.

Эквивалентный взгляд на эту процедуру заключается в том, что сжатие или расширение дискретного сигнала можно понимать как дискретизацию соответствующего непрерывного сигнала с другой частотой выборки без сжатия или расширения самого непрерывного сигнала. Посмотрим, например, на Рис. 10.13. На (а) показана дискретная кривая Гаусса, состоящая из 50 отсчётов. На (б) та же самая кривая представлена 400-и дискретными отсчётыами. Отличие (а) от (б) можно понимать двояко: либо мы считаем, что *частота дискретизации* осталась прежней, а сигнал расширился и стал в 8 раз длиннее, либо сам непрерывный сигнал считается неизменным, а частота дискретизации увеличилась в 8 раз. Такие методы изменения частоты дискретизации сигнала называются *многоскоростной обработкой*. Если дискретных отсчётов, представляющих сигнал, становится больше, это называется *интерполяцией*. Если отсчётов становится меньше — это *декомпрессия*. В Главе 3 мы рассматривали, как методы многоскоростной обработки сигналов используются в процессе аналого-цифрового и цифро-аналогового преобразования.

Возникает вопрос: если нам задан некоторый дискретный сигнал набором отсчётов, откуда мы можем знать, какой исходный непрерывный сигнал он представляет? Ответ зависит от того, какая область является более информативной: временная или частотная. В случае *кодирования во временной области* интересующий нас сигнал может принимать вид относительно гладкой кривой, проходящей через все заданные дискретные отсчёты. В простейшем случае нам достаточно соединить отсчёты последовательно прямыми линиями и закруглить (сгладить) углы. Более сложный подход связан с использованием какого-либо алгоритма *аппроксимации*, выполняемой, например, с помощью сплайнов или полиномов. Решение здесь не может быть единственным. Этот подход связан с минимизацией ошибок представления колебания во временной области, а частотное представление сигнала полностью игнорируется.

Если информация о сигнале *закодирована в частотной области*, то «игнорируется» временное представление сигнала и анализируется *частотный спектр*. В предыдущей главе мы говорили, что лучшее разрешение по частоте в спектре сигнала (большее число отсчётов в диапазоне частот 0...0.5 частоты дискретизации) может быть получено дополнением сигнала во временной области перед взятием ДПФ нулевыми отсчётами. *Дуальность* (о которой шла речь ранее) делает справедливым и обратное утверждение. Если нас интересует более высокое разрешение во временной области (интерполяция), следует дополнить спектр сигнала нулевыми отсчётами перед взятием обратного ДПФ. Допустим, мы хотим интерполировать дискретный сигнал, состоящий из 50 отсчётов так, чтобы он включал 400 отсчётов. Это может быть выполнено следующим образом. Берём сигнал из 50 отсчётов и дополняем его нулями до 64 отсчётов. Выполняем 64-точечное ДПФ и получаем частотный спектр, состоящий из 33 отсчётов действительной части и 33 отсчётов мнимой части. Дополняем и действительную, и мнимую части спектра справа 224 нулями до длины 257 отсчётов. Выполняем 512-точечное обратное ДПФ для перехода обратно во временную область. В результате имеем 512-точечный сигнал, полученный из 64-точечного сигнала и отличающийся более высоким разрешением по времени. Первые 400 отсчётов этого сигнала являются результатом интерполяции 50 отсчётов исходного сигнала.



**Рис. 10.13.** Интерполяция путём дополнения сигнала нулями в частотной области. Сигналы на (а и в) состоят из 50 дискретных отсчётов. Осуществляется интерполяция этих сигналов до длины 400 дискретных отсчётов с помощью добавления нулевых элементов в частотной области. Результаты показаны на (б и г) соответственно (несмотря на то, что кривые изображены непрерывными, сигналы являются дискретными, но включают большое число отсчётов на заданном интервале).

Ключевой особенностью описанного подхода является то, что спектральный состав сигнала при интерполяции не изменяется. Это приводит к тому, что форма сигнала иногда может, а иногда не может быть точно сохранена. Посмотрим, например, на (а и б). Сигнал, состоящий из 50 отсчётов, интерполируется до длины 400 отсчётов указанным методом. В результате получается гладкая кривая, такая же, как получилась бы при использовании аппроксимации. По-другому выглядит ситуация для сигналов на (в и г). Интерполяция привела к изменению формы сигнала во временной области. На фронтах импульса и в других точках разрыва в сигнале (к ним относится и разрыв при переходе от ( $N - 1$ )-го отсчёта к 0-му, так как сигнал рассматривается как циклический) появились характерные колебания (г). Такие колебания, возникающие в точках разрыва, получили название *колебаний Гиббса*. Они будут рассматриваться в Главе 11. Есть и другой подход к интерполяции сигнала, предполагающий вставку дополнительных нулевых значений между соседними отсчётами сигнала во временной области с последующей низкочастотной фильтрацией (см. Главу 3).

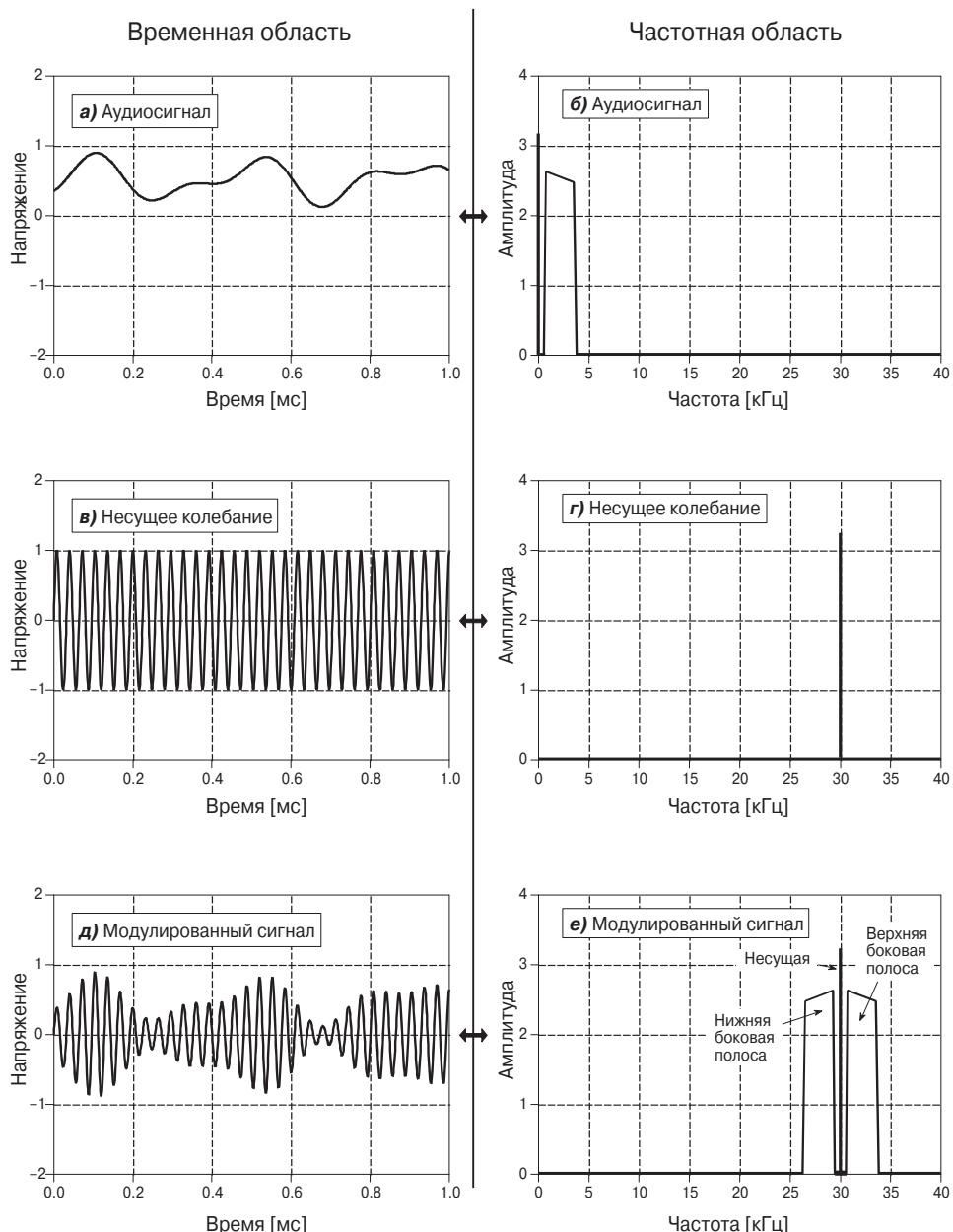
## 10.5. Умножение сигналов (амплитудная модуляция)

Замечательным свойством преобразования Фурье является то, что оно позволяет свети *свёртку* во временной области к умножению в частотной области. Мы рассмотрели использование этого свойства с одной стороны: свёртка сигналов во временной области сводится к перемножению их спектров. Теперь рассмотрим с другой: от свёртки в частотной области можно перейти к умножению во временной области. Примером такого подхода служит *амплитудная модуляция*. К тому же этот пример очень наглядно иллюстрирует практическую значимость, казалось бы, искусственно выдуманного понятия отрицательных частот.

Для передачи информации на близкие расстояния великолепно подходят аудиосигналы (звуки): вы говорите, и тот, кто находится в комнате, слышит вас. На большие расстояния хорошо распространяются сигналы радиочастотного диапазона волн. Например, если простой синусоидальный сигнал с амплитудой 100 В и частотой 1 МГц с помощью антенны излучается в пространство, то такая радиоволна может быть зафиксирована не только в этой же комнате, но и в следующей комнате, и в соседней стране, и даже на другой планете. *Модуляция* — это процесс объединения двух сигналов, в результате которого образуется третий сигнал, обладающий полезными свойствами двух исходных. Модуляция всегда подразумевает наличие нелинейной операции, так как включает умножение на переменную. Простое суммирование двух сигналов не даст результата. В радиосвязи с помощью модуляции удаётся использовать радиоволны для переноса аудио или другой информации на большие расстояния.

Радиосвязь достаточно развита в настоящее время, известно множество различных методов модуляции. Одним из простейших её видов является *амплитудная модуляция*. Её иллюстрация во временной и частотной областях представлена на Рис. 10.14. В этом примере использованы *непрерывные сигналы*, поскольку обычно модуляция выполняется с помощью аналоговой электроники. Тем не менее эта процедура может быть выполнена и в дискретном виде.

На (а) показан аудиосигнал, включающий постоянную составляющую, которая заставляет его всегда принимать положительные значения. На (б) изображён соответствующий спектр сигнала, включающий полосу звуковых частот от 300 Гц до 3 кГц, а также всплеск на нулевой частоте. Все другие частоты мы считаем поглощёнными с помощью аналогового фильтра. На (в и г) показано *несущее колебание*: синусоида с частотой, намного превышающей частоту аудиосигнала. Во временной области амплитудная модуляция состоит в перемножении аудиосигнала и несущего колебания. На (д) изображён результат модуляции: высокочастотное колебание, мгновенные значения амплитуды которого пропорциональны значениям исходного аудиосигнала. На языке радиотехники говорят, что *огибающая* несущего колебания пропорциональна *модулирующему сигналу*. Полученный сигнал подаётся на антенну и распространяется в пространстве в форме радиоволны, а впоследствии фиксируется принимающей антенной. На приёмной стороне регистрируется сигнал, совпадающий с переданным сигналом (д). Далее в этой цепочке используется детектор, или демодулятор, который из сигнала (д) восстанавливает исходный информационный сигнал (а).



**Рис. 10.14.** Амплитудная модуляция. Во временной области амплитудная модуляция сводится к перемножению аудиосигнала (**а**) и несущего колебания (**в**), в результате чего получается модулированный сигнал (**д**). Поскольку перемножению во временной области соответствует свёртка в частотной области, спектр модулированного сигнала представляет собой спектр исходного аудиосигнала, сдвинутый на величину частоты несущего колебания.

Поскольку сигналы во временной области в процессе амплитудной модуляции умножаются, то в частотной области их спектры сворачиваются. То есть спектр сигнала, представленный на (e), является результатом свёртки спектров сигналов (б и г). Поскольку спектр несущего колебания представляет собой сдвинутый вдоль частотной оси единичный импульс, то спектр модулированного сигнала будет равен спектру исходного аудиосигнала, сдвинутому на величину несущей частоты. В результате спектр модулированного сигнала включает три компоненты: *несущую частоту, верхнюю боковую полосу и нижнюю боковую полосу*.

Эти компоненты полностью соответствуют трем составляющим спектра исходного аудиосигнала: постоянной составляющей, полосе сигнала в области положительных частот (0.3...3 кГц) и полосе сигнала в области отрицательных частот (-0.3...-3 кГц). Несмотря на то что полоса отрицательных частот в исходном аудиосигнале является абстрактной, в процессе модуляции она переходит во вполне реальные частотные компоненты — нижнюю боковую полосу. Призрак приобрёл человеческий облик!

В области связи подобные манипуляции с частотным спектром распространены повсеместно. Примером может являться телевизионное радиовещание. Стандартный телевизионный сигнал занимает полосу частот от 0 до 6 МГц. Описанный метод смещения спектра позволяет вести передачу одновременно с нескольких телевизионных каналов шириной 6 МГц каждый, следующих в частотном диапазоне друг за другом. Например, канал 3 занимает полосу 60...66 МГц, канал 4 — 66...72 МГц, канал 83 — полосу 884...890 МГц и т. д. Телевизионный приёмник смещает спектр сигнала в выбранном канале на исходное положение от 0 до 6 МГц, позволяя воспроизводить телеизображение. Такая схема называется *мультиплексированием в частотной области*.

## 10.6. Преобразование Фурье дискретного времени

*Преобразование Фурье дискретного времени* — это один из видов Фурье-преобразований, который работает с *апериодическими дискретными сигналами*. Чтобы понять, что этот тип преобразования означает, лучше всего сравнить его с ДПФ. Для начала представьте, что у вас имеется  $N$  отсчётов некоторого дискретного сигнала и вы хотите построить его *спектр*. При использовании ДПФ сигнал представляется набором ( $N/2 + 1$ ) базисных синусо-косинусных функций с частотами, равномерно распределёнными на интервале от 0 до половины частоты дискретизации. Как говорилось в предыдущей главе, если сигнал во временной области дополнить нулями, то увеличится его период во времени, а расстояние между соседними отсчётами в частотной области станет меньше. При  $N$ , стремящемся к бесконечности, сигнал во временной области становится апериодическим, а в частотной — непрерывным. Это и приводит нас к преобразованию Фурье дискретного времени, ставящему в соответствие апериодическому дискретному сигналу периодический непрерывный спектр.

Математическое выражение для преобразования Фурье дискретного времени может быть получено из *уравнений анализа-синтеза*, характерных для ДПФ (выражение (8.2), (8.3) и (8.4)), с учётом увеличения  $N$  в пределе до бесконечности:

$$\begin{aligned} \operatorname{Re} X(\omega) &= \sum_{n=-\infty}^{+\infty} x[n] \cos(\omega n), \\ \operatorname{Im} X(\omega) &= - \sum_{n=-\infty}^{+\infty} x[n] \sin(\omega n). \end{aligned} \quad (10.1)$$

Уравнение анализа Фурье дискретного времени. Здесь  $x[n]$  — это сигнал во временной области, представленный  $N$  отсчётаами ( $n$  пробегает целые значения от 0 до  $(N-1)$ ). В частотной области сигнал представлен действительной  $\operatorname{Re} X(\omega)$  и мнимой  $\operatorname{Im} X(\omega)$  частями ( $\omega$  принимает значения от 0 до  $\pi$ ).

$$x[n] = \frac{1}{\pi} \int_0^\pi \operatorname{Re} X(\omega) \cos(\omega n) - \operatorname{Im} X(\omega) \sin(\omega n) d\omega. \quad (10.2)$$

Уравнение синтеза Фурье дискретного времени.

В приведенных формулах есть много тонких моментов. Во-первых, сигнал во временной области  $x[n]$  остаётся дискретным, и поэтому в его обозначении используются квадратные скобки. В то же время действительная и мнимая части частотного представления сигнала являются теперь непрерывными, и в их записи использованы круглые скобки:  $\operatorname{Re} X(\omega)$ ,  $\operatorname{Im} X(\omega)$ . Так как частотный спектр сигнала — это непрерывная функция частоты, то в уравнении синтеза использовано интегрирование, а не суммирование.

В Главе 8 мы говорили, что для обозначения номера частоты при использовании ДПФ-преобразования применяют один из трёх символов:  $k$  — индекс, принимающий целые значения от 0 до  $N/2$ ;  $f$  — доля частоты дискретизации, изменяющаяся в диапазоне от 0 до 0.5 частоты дискретизации;  $\omega$  — доля частоты дискретизации, выраженная в единицах круговой частоты и изменяющаяся в диапазоне от 0 до  $\pi$ . В случае преобразования Фурье дискретного времени спектр сигнала является непрерывным, и поэтому в качестве аргумента могут использоваться только величины  $f$  или  $\omega$ . Обычно используется  $\omega$ , так как математические выражения при этом становятся короче, — отпадает необходимость записи величины  $2\pi$ , которая присутствует почти всегда. Не забывайте, что использование приведенной круговой частоты  $\omega$ , меняющейся в диапазоне от 0 до  $\pi$ , соответствует изменению частоты от 0 до половины частоты дискретизации.

При вычислении обратного ДПФ перед использованием уравнения синтеза (8.2) необходимо значения дискретных отсчётов сигнала с номерами 0 и  $N/2$  разделить на 2 (8.3). В случае преобразования Фурье дискретного времени эта процедура оказывается не нужна. Вспомним, что деление на 2 в случае ДПФ было связано с определением частотного спектра как спектральной плотности, т. е. мощности сигнала, приходящейся на единицу полосы частот. В случае преобразования Фурье дискретного времени спектр становится непрерывным и необходимость в выделении крайних точек сигнала исчезает. Использование же нормирующего коэффициента остаётся необходимым. Вместо значения  $2/N$ , характерного для ДПФ (8.3), он становится равен  $1/\pi$  (10.2). Одни авторы используют нормирующий коэффициент в уравнении синтеза, другие — в уравнении анализа. Предположим, что вы имеете некоторый сигнал во временной области. Выполняя преобразование Фурье и затем обратное преобразование Фурье, вы ожидаете получить исходный сигнал. Поэтому коэффициент  $1/\pi$  (или  $2/N$  в слу-

чае ДПФ) должен быть включён в вычисления либо на этапе прямого, либо на этапе обратного преобразования. У некоторых авторов можно даже найти разделение нормирующего коэффициента между уравнениями синтеза и анализа и использование значения  $1/\sqrt{\pi}$  в обоих из них.

Уравнения преобразования Фурье дискретного времени включают операции суммирования и интегрирования, производимые в бесконечных пределах. Такие процедуры не могут быть реализованы с помощью вычислительной техники. Преобразование Фурье дискретного времени используется в теоретических рассуждениях как альтернатива ДПФ. Пусть, например, вы хотите найти *частотную характеристику* системы по её *импульсной характеристике*. Если импульсная характеристика задана в виде массива чисел, что характерно для экспериментальной регистрации сигнала или расчёта с помощью вычислительных средств, то вы смело можете использовать ДПФ. В частотной области вы получите спектр сигнала также в форме конечного массива чисел, представляющих значения частотных компонент сигнала, равномерно распределённых в диапазоне 0...0.5 частоты дискретизации.

Однако импульсная характеристика может быть задана и в форме уравнения. Это может быть, например, *sinc-функция*, о которой пойдёт речь в следующей главе, или *экспоненциально затухающий гармонический сигнал* и т. д. В этом случае для расчёта спектра сигнала используется *преобразование Фурье дискретного времени*. Оно позволяет математически получить частотный спектр в форме уравнения, непрерывный на интервале 0...0.5 частоты дискретизации. Несмотря на то что ДПФ также может быть использовано в этом случае, оно способно позволить нам рассчитать только отдельные отсчёты частотного спектра, но не всю непрерывную кривую.

## 10.7. Уравнение Парсеваля

Представления сигнала во *временной* и в *частотной области* являются альтернативными формами описания одного и того же сигнала. Следовательно, *энергия сигнала* в том и другом случае должна быть одинаковой. Эта тождественность и составляет суть *уравнения Парсеваля*. Оно справедливо для всего семейства *Фурье-преобразований*. Запишем его для случая ДПФ:

$$\sum_{i=0}^{N-1} x^2[i] = \frac{2}{N} \sum_{k=0}^{N/2} \text{Mag } X^2[k]. \quad (10.3)$$

Уравнение Парсеваля. В этом уравнении  $x[i]$  — это сигнал во временной области. Индекс  $i$  принимает значения 0... $N - 1$ .  $X[k]$  — модифицированный частотный спектр сигнала. Индекс  $k$  принимает значения 0... $N/2$ . Модифицированный частотный спектр равен ДПФ исходного сигнала с последующим делением крайних частот на корень квадратный из 2.

В левой части уравнения записана полная энергия, содержащаяся в сигнале, представленном во временной области. Она рассчитывается как сумма энергий  $N$  дискретных импульсных сигналов. В правой части уравнения записана энергия сигнала при его частотном представлении. Она рассчитывается аналогично —

суммированием энергий ( $N/2 + 1$ ) гармонических сигналов. Вспомним из физики, что энергия сигнала пропорциональна квадрату его амплитуды. Например, энергия звучания струны пропорциональна квадрату размаха её колебаний, а энергия, накапливаемая конденсатором, пропорциональна квадрату напряжения на нём. В выражении (10.3) величина  $X[k]$  — это частотный спектр сигнала  $x[n]$  с учётом небольшой модификации: крайние частотные компоненты  $X[0]$  и  $X[N/2]$  разделены на  $\sqrt{2}$ . Такая модификация, а также использование коэффициента  $2/N$  в правой части уравнения необходимы, чтобы учесть некоторые тонкости расчёта и суммирования энергий.

Разберёмся с этими вопросами. Найдём частотный спектр сигнала с использованием ДПФ. Затем сформируем на его основе набор амплитуд синусно-косинусных сигналов, позволяющих восстановить исходный сигнал в соответствии с выражением (8.3). При этом потребуется разделить крайние элементы (отсчёты с номерами 0 и  $N/2$ ) на 2, а затем все элементы — на  $N/2$ . Таким образом мы получаем значения амплитуд. Однако это их пиковые значения, а не среднеквадратические, необходимые для расчёта энергии. Чтобы перейти к среднеквадратическим значениям амплитуд, достаточно выполнить деление на  $\sqrt{2}$ . Это деление должно быть выполнено для всех значений в частотной области, за исключением 0-го и  $N/2$ -го. Крайние две частоты следует рассматривать индивидуально. Одна из них представлена постоянной величиной, а другая скачкообразно изменяет своё значение с одной постоянной величины на другую. Для этих частот пиковое значение и так оказывается равным среднеквадратическому. Все полученные таким образом значения в частотной области возводятся в квадрат и затем суммируются. Последним шагом является деление полученной суммы на величину  $N$ . В результате мы получаем выражение (10.3). Уравнение Парсеваля весьма интересно с позиции физики (оно описывает принцип сохранения энергии), однако в цифровой обработке сигналов широкого применения оно не нашло.

## ПАРЫ ФУРЬЕ-ПРЕОБРАЗОВАНИЙ

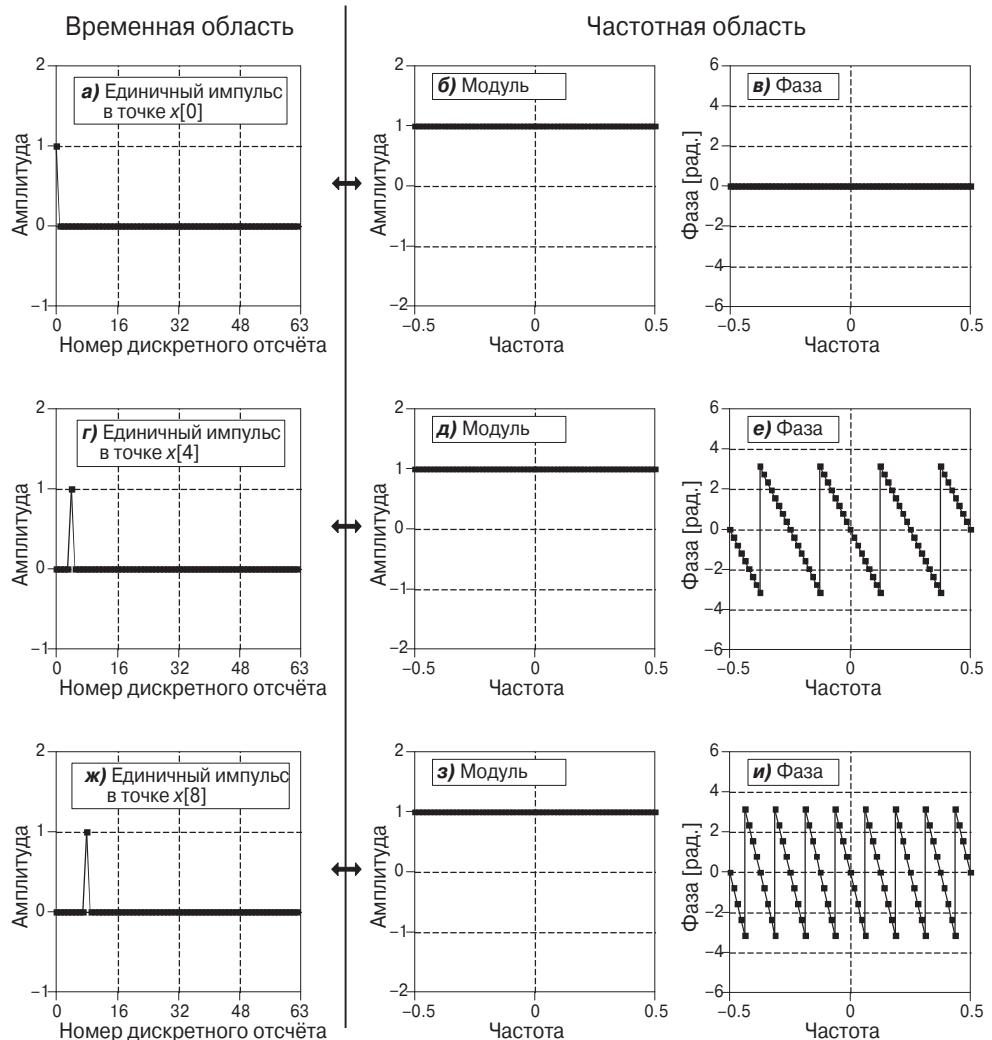
Любому сигналу, представленному во временной области, можно поставить в соответствие описание в частотной области, и наоборот. Например, прямоугольный импульс во временной области описывается в частотном представлении sinc-функцией (функцией  $\sin(x)/x$ ). При этом справедливо свойство дуальности преобразования Фурье: прямоугольному импульсу в частотной области соответствует sinc-функция во временной области. Сигналы и спектры, которые соотносятся таким образом друг с другом, называются *парами Фурье-преобразований*. В этой главе рассказывается о некоторых наиболее распространённых примерах пар преобразований Фурье.

### 11.1. Единичный импульс

Одним из простейших и одновременно наиболее широко используемых цифровых сигналов является *единичный импульс*. Соответствующая ему *пара преобразований Фурье* также имеет достаточно простую форму. На Рис. 11.1 $a$  представлен единичный импульс во *временной области*. Его *частотный спектр* показан на (б и в). *Амплитудная характеристика* оказывается постоянной величиной, а *фазовая характеристика* равна нулю на всех частотах. Такое соотношение формы сигнала во *временной области* и его представления в *частотной области* соответствует закономерностям *сжатия/расширения* сигналов, о которых шла речь в предыдущей главе. Если во временной области произошло сжатие сигнала до единичного импульса, то в частотной области спектр этого сигнала должен расширяться и превратиться в постоянный сигнал.

Если произвести сдвиг исходного единичного импульса вправо вдоль временной оси на 4 и 8 отсчётов ((г и ж) соответственно), то *модуль* его спектрального представления останется прежним, а *фаза* приобретёт вид линейно спадающей составляющей. На представленных рисунках фазовые характеристики не «развернуты», и их значения лежат в пределах интервала  $-\pi \dots \pi$ . Обратите также внимание на то, что на оси частот отсчёты откладываются в диапазоне  $-0.5 \dots 0.5$ . То есть показаны не только положительные частоты, но и отрицательные. Область отрицательных частот несёт избыточную информацию, но в ЦОС принято отображать её на графиках, и к этому следует привыкнуть.

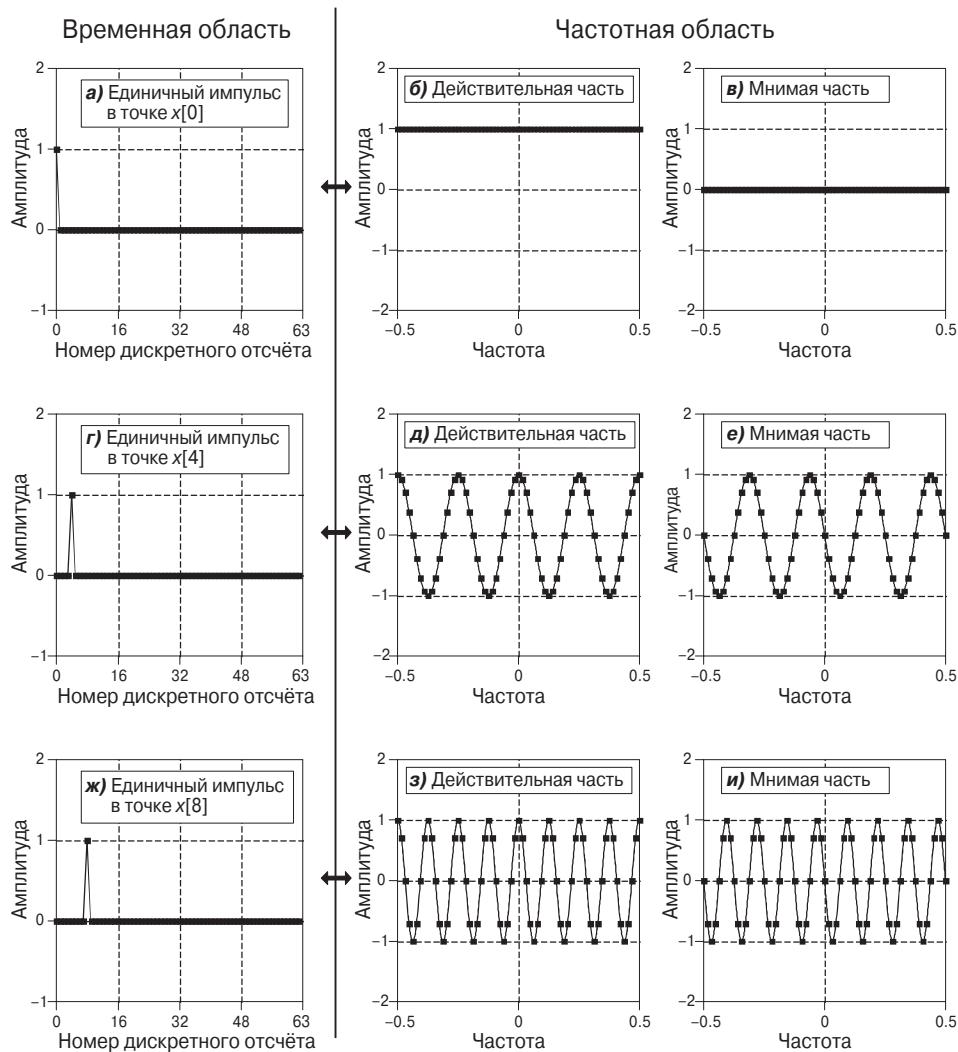
На Рис. 11.2 представлены те же сигналы, что и на Рис. 11.1, но для представления спектра сигнала в частотной области использована прямоугольная система координат. Здесь следует усвоить два урока. Во-первых, сравните формы графического отображения сигнала в частотной области в полярных и в прямоугольных координатах. Полярная форма является гораздо более наглядной (это справедливо в большинстве случаев): модуль представлен постоянной величиной, а фаза —



**Рис. 11.1.** Пара преобразований Фурье для единичного импульса в полярных координатах. Единичному импульсу во временной области соответствуют модуль в виде постоянной составляющей и линейная фаза в частотной области.

прямой линией. В прямоугольной же системе координат действительная и мнимая части являются гармоническими колебаниями, форма которых практически ничего нам не говорит.

Второе, что иллюстрирует **Рис. 11.2**, — это свойство *дуальности* преобразования Фурье. Обычно каждый отсчёт в частотной области на выходе ДПФ рассматривают как гармонический сигнал во временной области. Однако верно и обратное: каждому временному отсчёту соответствует гармоническое колебание в частотной области. То, что на графике учтены отрицательные частоты, позволяет проиллюстрировать свойство дуальности более наглядно. На (г..е) показано, что, например, единичный импульс, сдвинутый в четвёртый дискретный отсчёт во временной области, соответствует в частотной области четырём периодам повто-



**Рис. 11.2.** Пара преобразований Фурье для единичного импульса в прямоугольных координатах. Каждому отсчёту во временной области соответствует в частотной области косинусный сигнал в действительной части и синусный сигнал с обратным знаком в мнимой части.

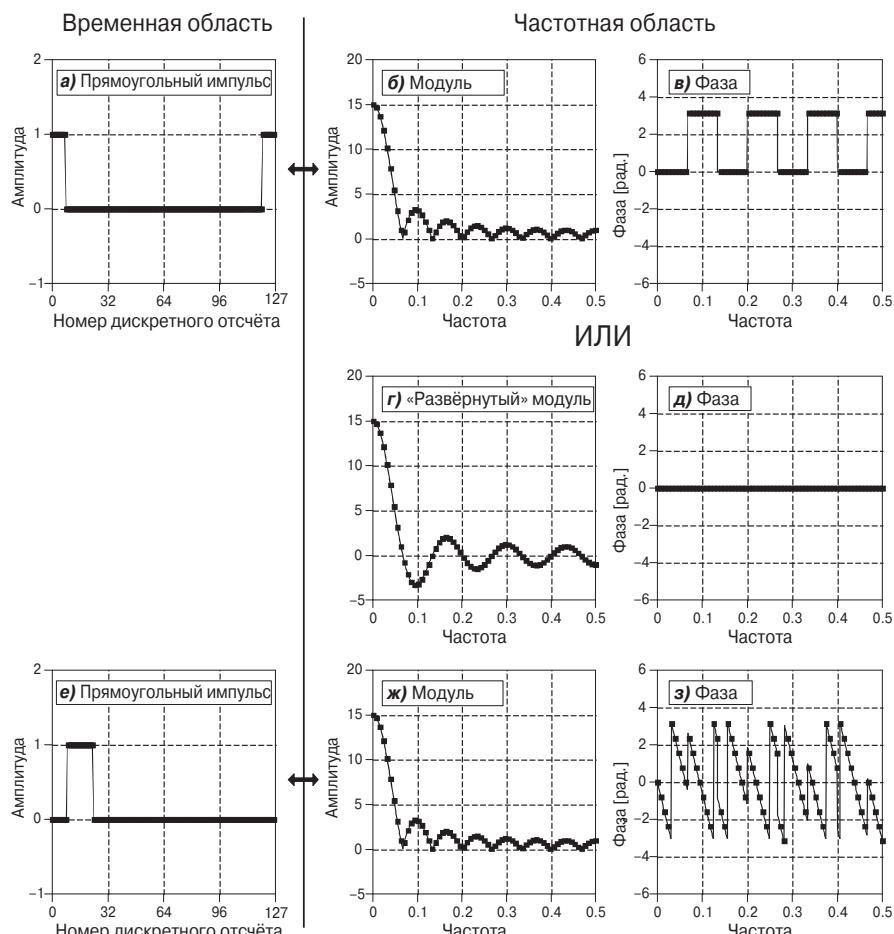
рения косинусоиды в действительной части и четырём периодам синусоиды с обратным знаком в мнимой части. Вспомним, что единичный импульс, сдвинутый на четвёртый отсчёт в действительной части спектра сигнала, соответствует во временной области четырём периодам косинусоидального сигнала. Аналогично единичный импульс, сдвинутый на четвёртый отсчёт в мнимой части спектра, соответствует во временной области четырём периодам синусоидального сигнала с обратным знаком.

На этом может быть основан ещё один подход к расчёту ДПФ, который упоминался в Главе 8 наряду с корреляцией сигнала с набором гармонических колебаний во временной области. Каждому дискретному отсчёту во временной области соответствует косинусный сигнал, добавляемый к действительной части

спектра сигнала, и синусный сигнал с обратным знаком, добавляемый к мнимой части. Амплитуда каждой гармонической составляющей в частотной области определяется амплитудой временного отсчёта, а частота — его номером. Получается следующий алгоритм: 1) «пробегаем» по всем отсчётам сигнала во временной области; 2) рассчитываем косинусную и синусную составляющие спектра сигнала, соответствующие каждому временному отсчёту; 3) проводим суммирование всех гармонических «сигналов» в частотной области. Программа, работающая по такому алгоритму, будет практически идентична программе, реализующей корреляционный метод (**Программа 8.2**), с той лишь разницей, что внутренний и внешний циклы поменяются местами.

## 11.2. Функция $\sin(x)/x$

На Рис. 11.3 проиллюстрирована получившая широкое распространение пара преобразований Фурье: *прямоугольный импульс* и функция  $\sin(x)/x$  (*sinc-функция*).



**Рис. 11.3.** ДПФ прямоугольного импульса. Прямоугольному импульсу во временной области соответствует функция  $\sin(x)/x$  в частотной области и наоборот

Функция  $\sin(x)/x$  представляет собой функцию синуса, амплитуда которой убывает по закону  $1/x$ . На **(а)** изображён прямоугольный импульс с центром, расположенным в нулевом отсчёте, так что правая половина импульса оказывается с одной, а левая — с другой стороны от вертикальной оси, проходящей через нулевой отсчёт. Благодаря свойству периодичности преобразование ДПФ рассматривает такой сигнал как один импульс. Спектр сигнала изображён на **(б и в)** или на **(г и д)**. Последние отличаются «развернутым» представлением спектра, т. е. искусственным снятием ограничения на значения модуля и фазы.

Рассмотрим более детально **(г и д)**. «Развернутый» модуль представляет собой колебательный процесс, характеризуемый уменьшением амплитуды с ростом частоты. Фазовая составляющая спектра равна нулю, как и следовало ожидать для сигнала, симметричного относительно вертикальной оси, проходящей через нулевой отсчёт. По отношению к модулю частотного представления «развернутый» означает, что модуль может принимать как положительные, так и отрицательные значения. Однако по определению модуль должен всегда быть положительным. Это показано на **(б и в)**. Все значения модуля оказываются больше нуля за счёт сдвига фазы на  $\pi$  там, где «развернутый» модуль **(г)** становится отрицательным.

На **(е)** показан исходный прямоугольный импульс, сдвинутый так, что он теперь действительно выглядит как один импульс, но при этом теряется симметричность относительно нуля. Это не влияет на модуль частотного представления сигнала, однако добавляет линейную составляющую к фазе. Вы спросите, а как выглядит спектр в терминах действительной и мнимой частей? Он оказывается настолько запутанным, что лучше к нему не обращаться.

$N$ -точечный сигнал во временной области, представляющий собой прямоугольный импульс единичной амплитуды и длительности  $M$  дискретных отсчётов, имеет частотный спектр, рассчитываемый по формуле

$$\text{Mag } X[k] = \left| \frac{\sin(\pi k M/N)}{\sin(\pi k/N)} \right|. \quad (11.1)$$

Частотный спектр прямоугольного импульса. В этом выражении  $N$  — число дискретных отсчётов сигнала во временной области, только  $M$  из которых принимают отличное от нуля значение.  $X[k]$  — частотный спектр. Аргумент  $k$  принимает целые значения в диапазоне  $0 \dots N/2$ . Чтобы избежать деления на ноль, значение спектра в точке  $X[0]$  следует принять равным  $M$ . Функция синуса измеряет аргумент в радианах, а не в градусах. Данное уравнение учитывает возможность наложения.

Как альтернатива может быть использовано преобразование Фурье дискретного времени, рассчитывающее спектр как функцию частоты  $f$ , измеряемую в долях частоты дискретизации:

$$\text{Mag } X(f) = \left| \frac{\sin(\pi f M)}{\sin(\pi f)} \right|. \quad (11.2)$$

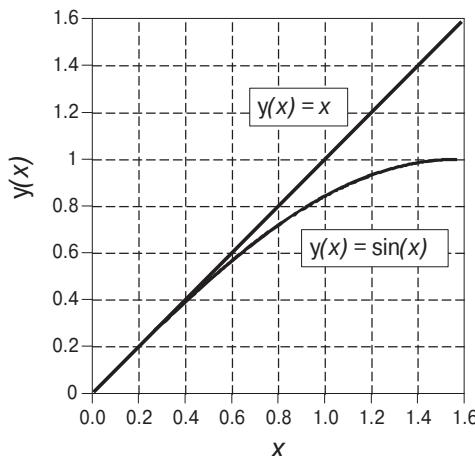
Уравнение, аналогичное выражению **(11.1)**, но использующее в качестве аргумента величину, выраженную в долях частоты дискретизации. Параметр  $f$  принимает непрерывный ряд значений в диапазоне  $0 \dots 0.5$ . Чтобы избежать деления на ноль, следует принять  $\text{Mag } X(0) = M$ .

Выражение (11.1) рассчитывает  $(N/2 + 1)$  значений частотного спектра сигнала, в то время как выражение (11.2) вычисляет спектр как непрерывную кривую, включающую и эти дискретные отсчёты. Приведённые выражения позволяют рассчитать только модуль частотного спектра. Вид фазы определяется исключительно положением импульса на временной оси, о чём говорилось в предыдущей главе.

Обратите внимание, что на Рис. 11.3б амплитуда колебания на границе окна ( $0.5$  частоты дискретизации) снижается не до нуля. Соответственно следует ожидать продолжения этого колебания в следующем периоде и, как следствие, возникновения **наложений**. Это меняет форму сигнала в частотной области, что находит отражение в выражениях (11.1) и (11.2).

Часто желательно знать, каким является частотный спектр сигнала, если наложения отсутствуют. Это важно, поскольку *дискретные сигналы* часто используются для моделирования преобразований *непрерывных сигналов*, а в случае непрерывных сигналов наложений нет. Чтобы исключить наложения, достаточно в выражениях (11.1) и (11.2) изменить знаменатель следующим образом:  $\sin(\pi k/N)$  заменить на  $(\pi k/N)$ , а  $\sin(\pi f)$  заменить на  $(\pi f)$ . Рис. 11.4 поясняет смысл этой замены. Поскольку частота  $f$  лежит в диапазоне  $0\dots0.5$ , величина  $\pi f$  будет изменяться  $0\dots1.5708$ . В этом диапазоне малых значений аргумента разница между  $\pi f$  и  $\sin(\pi f)$  невелика. В нулевой точке разность таких сигналов равна нулю, а в точке  $0.5$  она составляет всего около  $36\%$ . В случае отсутствия наложений кривая на Рис. 11.3б будет иметь несколько меньшую амплитуду колебаний у правого края и останется без изменений в левой части.

Когда в спектре прямоугольного импульса отсутствуют наложения (в случае, если рассматривается непрерывный сигнал или вы просто пренебрегаете наличием наложений), он приобретает типовую форму функции  $\sin(x)/x$ . Таким образом, в случае непрерывных сигналов прямоугольный импульс и sinc-функция представляют собой пару Фурье-преобразований. Для дискретных сигналов это справедливо лишь отчасти, с ошибкой, обусловленной наложениями.



**Рис. 11.4.** Сопоставление функций  $x$  и  $\sin(x)$ . Функции  $y(x) = x$  и  $y(x) = \sin(x)$  практически совпадают при значениях  $x$ , близких к нулю, и отличаются всего на  $36\%$  в точке  $x = 1.57$  ( $\pi/2$ ). Это соотношение позволяет оценить степень отличия спектра прямоугольного импульса от чистой функции  $\sin(x)/x$ , возникающей вследствие наложений.

Функция  $\sin(x)/x$  имеет одну неприятную особенность. В точке  $x = 0$  мы получаем деление на ноль. В математике это не является проблемой. При малых значениях  $x$  значение  $\sin(x)$  становится приблизительно равным  $x$  (Рис. 11.4) и выражение  $x/x$  оказывается равным единице. То есть с математической точки зрения при уменьшении аргумента  $x$  значение sinc-функции стремится к единице и в точке 0 оказывается равным единице ( $\text{sinc}(0) = 1$ ). Однако попробуйте объяснить это компьютеру! Всё, что он увидит, — это деление на ноль. Он выдаст ошибку и остановит выполнение программы. Поэтому важно запомнить, что программа, вычисляющая функцию  $\sin(x)/x$ , должна включать специальный код обработки ситуации  $x = 0$ .

Ключевой характеристикой функции  $\sin(x)/x$  является то, в каких точках происходит пересечение нуля (оси абсцисс). Это происходит на тех частотах, для которых соответствующие гармонические колебания имеют целое число периодов повторения на длительности прямоугольного импульса. Например, если длительность прямоугольного импульса составляет 20 дискретных отсчётов, то первая точка пересечения нуля функцией  $\sin(x)/x$  в частотной области будет равна частоте, на которой гармонический сигнал делает одно полное колебание за 20 периодов дискретизации. Также и для следующих точек пересечения нуля. Здесь можно провести параллель с вычислением ДПФ с помощью корреляции. Каждый отсчёт в частотной области получается умножением отсчётов временного сигнала на отсчёты гармонической базисной функции и суммированием произведений. Если сигнал во времени представлен прямоугольным импульсом единичной амплитуды, то это равносильно суммированию отсчётов гармонических сигналов, складывающихся на длине импульса. Очевидно, если суммирование производится на целом числе периодов, то результат окажется равным нулю.

Функция  $\sin(x)/x$  очень широко распространена в ЦОС благодаря тому, что она представляет пару Фурье-преобразований совместно с таким простым и общераспространённым сигналом, как прямоугольный импульс. Например, sinc-функция используется в спектральном анализе, о чём шла речь в Главе 9. Как производится анализ дискретного сигнала бесконечной длины? Поскольку ДПФ может быть применено только к сигналам конечной длины, приходится брать только  $N$  отсчётов в действительности более длительного сигнала. Такое ограничение длины сигнала равносильно умножению его на прямоугольное окно (прямоугольный импульс). Единичные отсчёты прямоугольного импульса «оставляют» отсчёты исходного сигнала, а нулевые отсчёты «убирают» соответствующие отсчёты бесконечного сигнала из рассмотрения. Какое влияние оказывает эта процедура на спектр исходного бесконечно длинного сигнала? Умножение во временной области на прямоугольный импульс равносильно свёртке в частотной области с функцией  $\sin(x)/x$ . Это уменьшает частотное разрешение спектра сигнала, как было показано на Рис. 9.5а.

## 11.3. Другие пары преобразований Фурье

На Рис. 11.5а и б продемонстрирована справедливость обратного соответствия выше рассмотренных сигналов: прямоугольному импульсу в частотной области соответствует функция  $\sin(x)/x$  (плюс наложения) во временной области. Временной сигнал с учётом наложений рассчитывается в соответствии с выражением

$$x[i] = \frac{1}{N} \frac{\sin(2\pi i (M - 1/2) / N)}{\sin(\pi i / N)}. \quad (11.3)$$

Обратное ДПФ прямоугольного импульса. В частотной области прямоугольный импульс имеет единичную амплитуду и длину  $0 \dots (M - 1)$  отсчёт. Параметр  $N$  — это раз мерность ДПФ, а  $x[i]$  — сигнал во временной области с индексом, «пробегающим» от 0 до  $(N - 1)$ . Чтобы избежать деления на ноль, следует принять  $x[0] = (2M - 1)/N$ .

Чтобы избежать эффектов, связанных с наложением, представим себе, что дискретные отсчёты в частотной области следуют очень часто, делая кривую непрерывной. В результате во временной области получится сигнал бесконечной длины, не имеющий периодических повторений. Здесь применимо преобразование Фурье дискретного времени. Получаемый во временной области сигнал рассчитывается в соответствии с выражением

$$x[i] = \frac{\sin(2\pi f_c i)}{i\pi}. \quad (11.4)$$

Обратное преобразование Фурье дискретного времени прямоугольного импульса. В частотной области прямоугольный импульс имеет единичную амплитуду и лежит в диапазоне от 0 до некоторой граничной частоты  $f_c$  на интервале  $0 \dots 0.5$ . Сигнал  $x[i]$  представлен во временной области. Индекс  $i$  «пробегает» целые значения от 0 до  $(N - 1)$ . Чтобы избежать деления на ноль, следует принять  $x[0] = 2f_c$ .

Это уравнение играет важную роль в ЦОС, поскольку спектр сигнала на выходе фильтра в виде прямоугольного импульса отвечает *идеальному низкочастотному фильтру*. Соответственно функция  $\sin(x)/x$ , описываемая этим выражением, является *импульсной характеристикой* идеального фильтра низких частот. Оно лежит в основе проектирования широко применяемого класса *оконных цифровых фильтров*, о чём говорится в Главе 15.

На Рис. 11.5в и г показано, что треугольному импульсу во временной области соответствует квадрат sinc-функции (плюс эффекты наложения) в частотной области. Данная пара Фурье-преобразований не так важна сама по себе, как причина, по которой она существует. Треугольный импульс длительностью  $(2M - 1)$  отсчётов может быть получен во временной области с помощью свёртки прямоугольного импульса с самим собой. От свёртки во временной области переходим к умножению в частотной области и получаем, что свёртка сигнала с самим собой равносильна возведению спектра в квадрат.

Существует ли такой сигнал, который по своей форме совпадает с собственным спектром? Существует, притом только один — *кривая Гаусса*. Вид этой кривой показан на (д), а её спектр, также являющийся кривой Гаусса, — на (е). Такое совпадение, однако, справедливо, только если не учитывать наложения. Связь *среднеквадратического отклонения* гауссовой кривой во временной и частотной областях определяется выражением  $2\pi\sigma_f = 1/\sigma_t$ . Несмотря на то что на (е) показана только правая часть кривой, в области отрицательных частот она дополняется до полной кривой Гаусса с центром на вертикальной оси, проходящей через ноль.

На (ж) показан ещё один сигнал, который мы назовём гауссовым всплеском. Он формируется умножением гармонического сигнала на кривую Гаусса. Например, сигнал на (ж) получен умножением синусоидального сигнала на кривую (д). Спектр такого сигнала будет иметь форму кривой Гаусса, сдвинутой вдоль оси

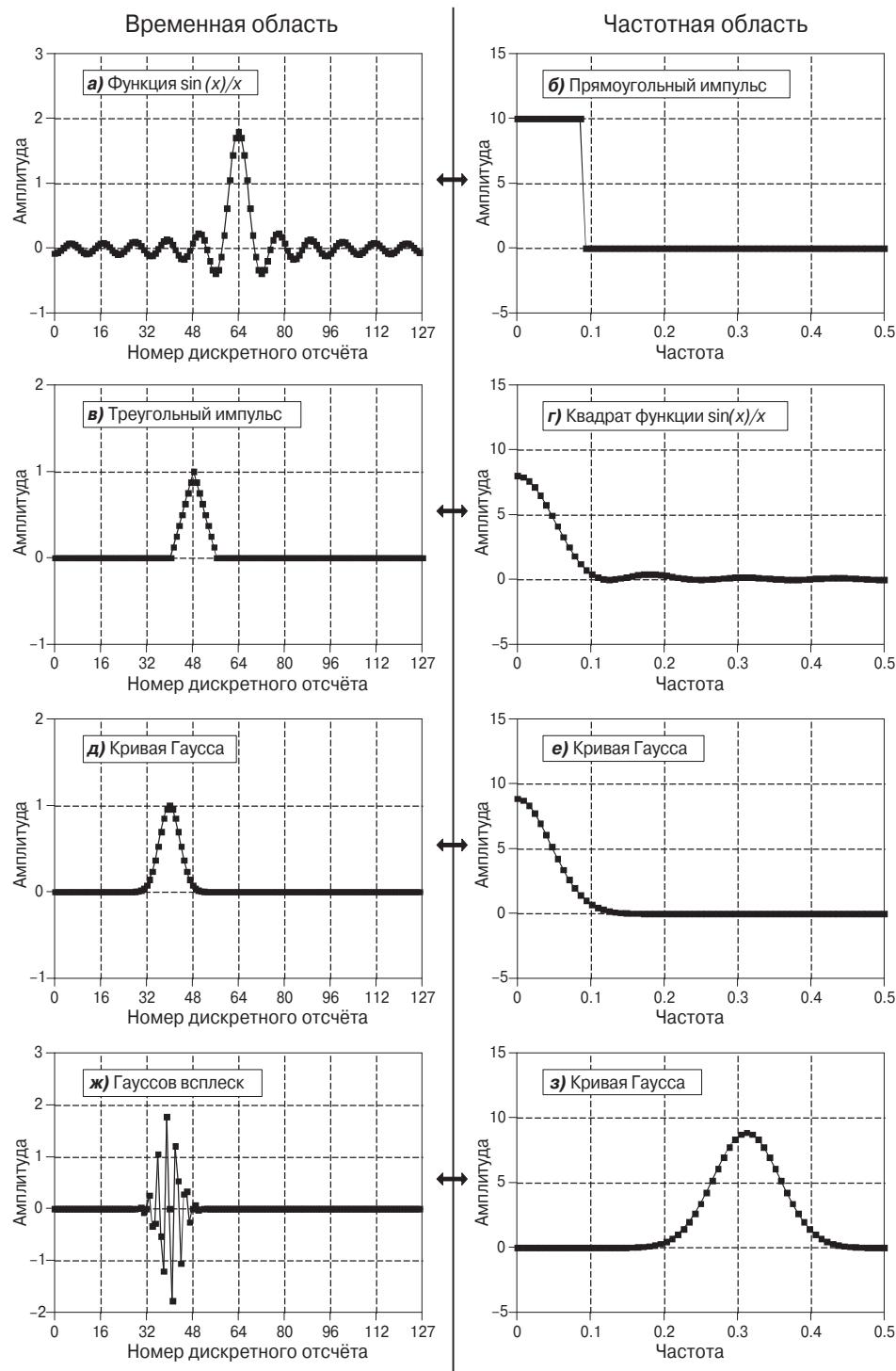


Рис. 11.5. Общераспространённые пары Фурье-преобразований.

частот. Вновь эта пара Фурье-преобразований интересна не сама по себе, а теми свойствами, которые в ней заключены. Мы сказали, что гауссов всплеск получается умножением сигналов во временной области, чему соответствует свёртка спектров двух сигналов в частотной области. Спектр гармонического сигнала — это делта-функция (или единичный импульс), сдвинутая на требуемую частоту. Спектр кривой Гаусса также гауссова кривая с центром на оси ординат. Свёртка таких двух спектров даёт нам гауссову кривую, сдвинутую по оси частот на частоту гармонического сигнала. Это должно показаться вам знакомым: процедура аналогична *амплитудной модуляции*, о которой шла речь в предыдущей главе.

## 11.4. Колебания Гиббса

На Рис. 11.6 показан сигнал во временной области как результат *синтеза* на основе ряда гармонических сигналов. Синтезируемый сигнал приведён на последнем рисунке — (з). Этот сигнал имеет длительность 1024 дискретных отсчёта, и, следовательно, в процедуре синтеза должно участвовать 513 гармонических сигналов. На (а...ж) показан вид синтезируемого сигнала, когда в уравнении синтеза учтено неполное число гармоник. Например, сигнал на (е) получен с использованием гармоник 0...100. Этот сигнал получен, как ДПФ сигнала на (з) с обнулением частот 101...512 и последующим обратным ДПФ-преобразованием.

Чем больше гармоник участвует в синтезе, тем больше результат синтезированного похож на исходный сигнал. Нас будет интересовать, как происходит приближение формы сигнала в местах резкого скачка уровня сигнала. Сигнал на (з) имеет три такие точки: две являются фронтами прямоугольного импульса; третья расположена между 1023-м и 0-м отсчётами (ДПФ рассматривает сигнал как периодический). Когда в процедуре синтеза участвуют не все необходимые частоты, в каждой из указанных точек возникают дополнительные постепенно затухающие колебания уровня сигнала в форме ревербераций. Этот эффект известен как *колебания Гиббса*, по имени физика-математика Джозайя Гиббса, давшего объяснения этим явлениям в 1899 году.

Рассмотрим колебания Гиббса, представленные на (д...ж), более детально. При увеличении числа гармоник уменьшается длительность колебательного процесса, при этом амплитуда колебаний остаётся практически неизменной, равной приблизительно 9%. В случае *дискретных сигналов* колебания Гиббса исчезают с включением в процедуру синтеза последнего гармонического сигнала. Проблема снимается. Однако в случае *непрерывных сигналов* ситуация усложняется. Чтобы компенсировать эффект Гиббса, требуется суммировать бесконечное число гармоник. А пока все они не учтены, амплитуда колебаний остаётся на уровне 9%. Принимая это во внимание (а также ряд других аргументов), возникает естественный вопрос: может быть суммирование бесконечного числа непрерывных гармонических сигналов поможет восстановить форму в точках скачкообразного изменения? Помните спор между Лагранжем и Фурье?

В этой загадке ключевой момент состоит в том, что при увеличении числа гармоник уменьшается длительность колебательного процесса. Если их число бесконечно, она становится нулевой, несмотря на то, что амплитуда колебаний по-прежнему неизменна. Значение синтезируемого сигнала будет сходиться к среднему

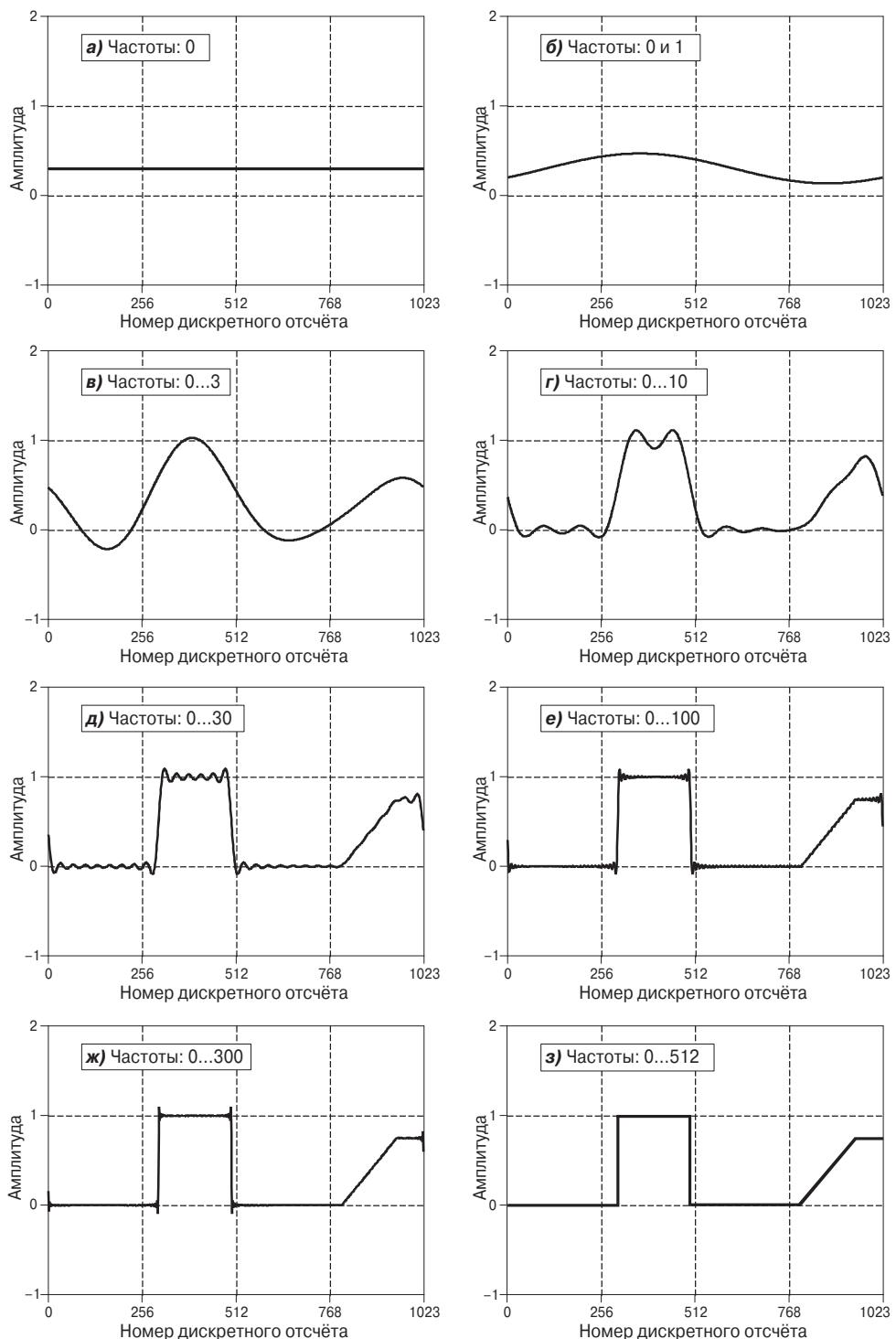


Рис. 11.6. Колебания Гиббса.

значению в точках прерывистого изменения уровня между двумя соседними уровнями. Как показал Гиббс, сумма гармоник сходится к значению восстанавливаемого сигнала с точки зрения равенства нулю энергии сигнала ошибки.

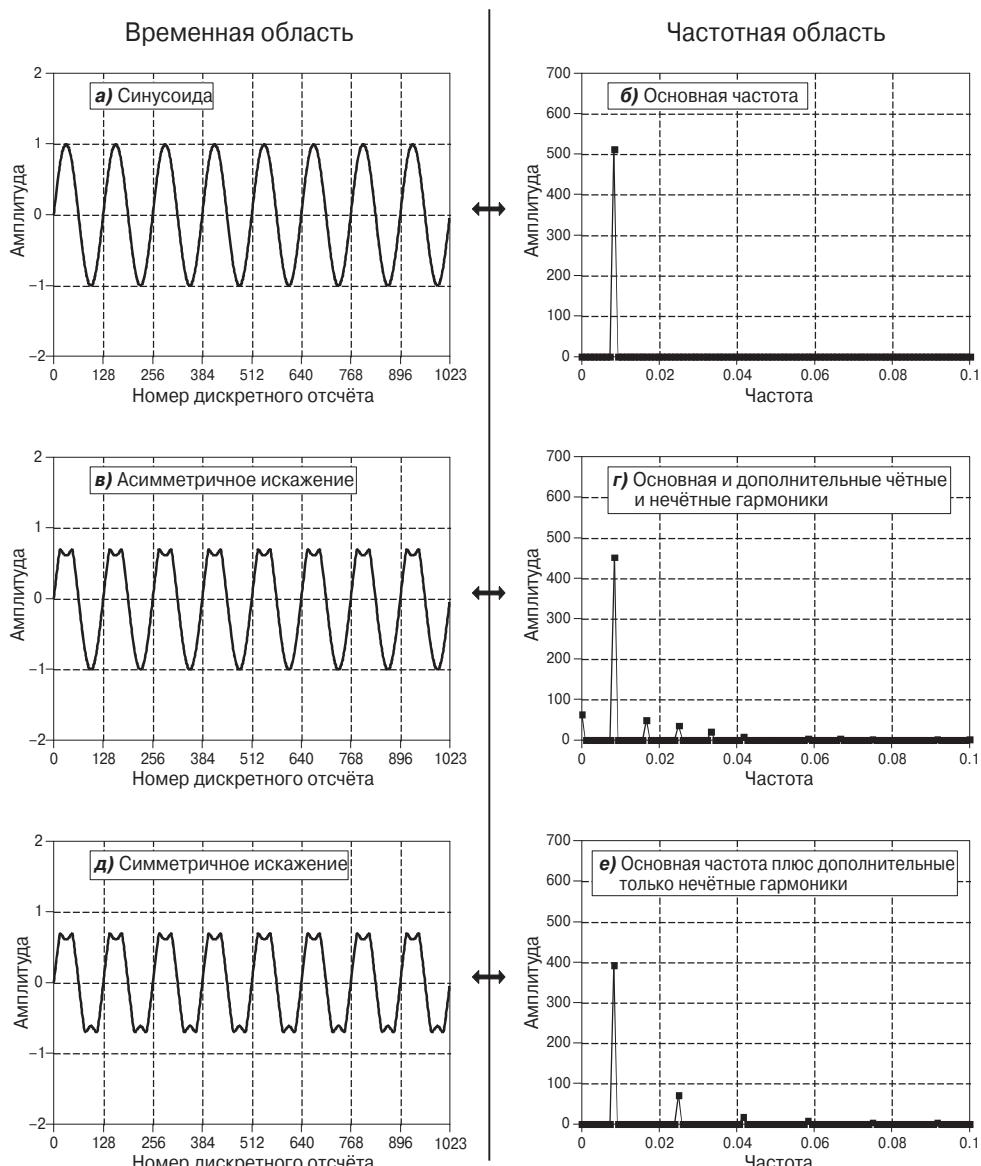
В цифровой обработке сигналов с проблемами, возникающими из-за эффекта Гиббса, приходится сталкиваться достаточно часто. Например, низкочастотный фильтр формируется путём подавления высоких частот. В результате во временной области в точках скачкообразного изменения сигнала появляются дополнительные колебания. Другой распространённый пример — сглаживание сигнала на краях путём умножения на весовое окно во временной области. Благодаря *дуальности* это приводит к искажению формы спектра сигнала в частотной области. К этой проблеме мы ещё вернёмся в последующих главах, посвящённых проектированию цифровых фильтров.

## 11.5. Гармоники

Если сигнал является периодическим с частотой  $f$ , то все частоты, входящие в его состав, будут кратны  $f$ . Эти частотные составляющие называются *гармониками*. *Первая гармоника* имеет частоту  $f$ , *вторая гармоника* —  $2f$ , *третья гармоника* —  $3f$  и т. д. Первая гармоника имеет также другое особое название — *основная частота*. Рассмотрим пример, представленный на **Рис. 11.7**. На (а) показана чистая синусоида, а на (б) — её спектр (единственный пик). На (в) синусоидальный сигнал искажён: имеет вогнутые верхушки. Спектр такого искажённого сигнала представлен на (г). Поскольку искажённый сигнал остался периодическим и период его повторения не изменился, в спектре сигнала наблюдается тот же пик и ещё ряд дополнительных гармоник. Значения гармоник могут быть различны, однако, как правило, их уровень падает с ростом частоты. Быстрые скачки уровня сигнала во временной области обуславливают наличие более высоких частот. Рассмотрим, например, обычный логический элемент ТТЛ, выполняющий функцию генерации последовательности прямоугольных импульсов с частотой 1 кГц. Уровень сигнала нарастает и падает за несколько наносекунд. Это обуславливает наличие в спектре сигнала гармоник до 100 МГц — до десятисычной гармоники!

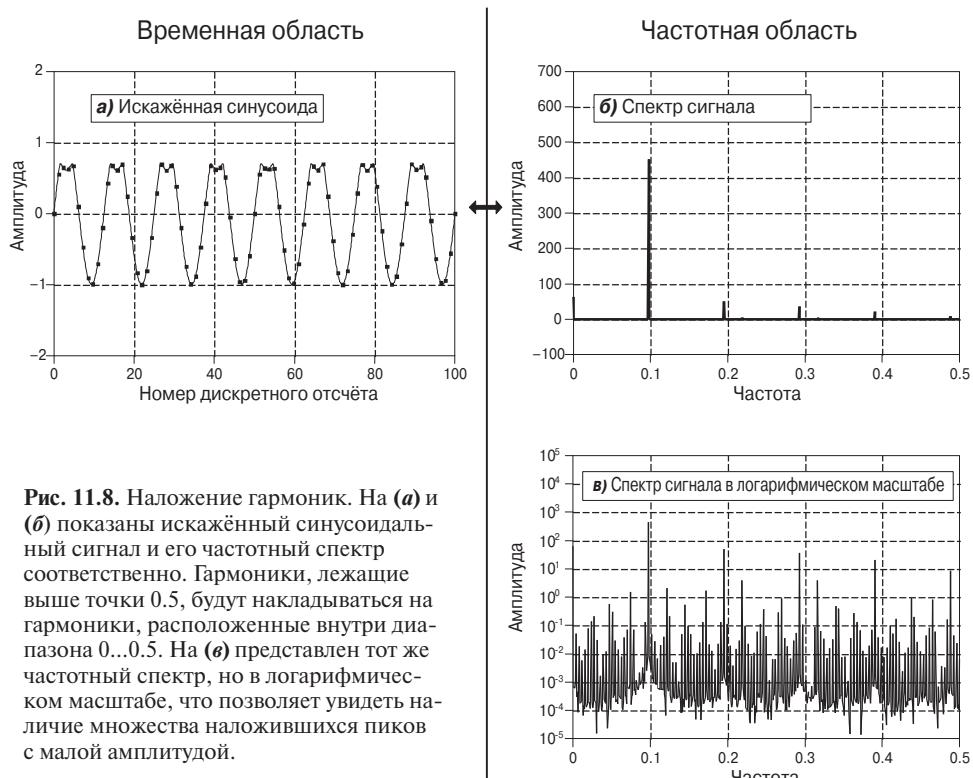
На (е) показан один интересный момент, характерный для гармонического анализа. Если сигнал симметричен относительно горизонтальной оси, т. е. его верхняя часть является зеркальным отражением нижней части, то все чётные гармоники в его спектре окажутся равными нулю. На (е) частотный спектр включает основную частоту, третью гармонику, пятую гармонику и т. д.

Любой непрерывный периодический сигнал может быть представлен суммой гармоник описанным образом. Для дискретных периодических сигналов возникает проблема, которая может сделать такой простой подход неприменимым. Как вы уже, возможно, догадались, это проблема наложения (**Рис. 11.8**). На (а) показана синусоида, искажённая так же, как на предыдущем рисунке, т. е. имеет вогнутую форму амплитуды. Однако эта вогнутость в окрестности точек перегиба выглядит не так гладко и естественно, потому что на интервал вогнутости приходится малое число дискретных отсчётов сигнала. На (б) показан спектр этого сигнала. Как и должно быть, в спектре присутствуют основная частота и дополнительные гармоники. На этом примере показано, что гармоники могут выйти за



**Рис. 11.7.** Гармоники. В результате изменения формы синусоиды так, что она становится несимметричной (**в**), кроме основной гармоники в спектре сигнала появляются дополнительные — чётные и нечётные (**г**). Если искажение оставляет сигнал симметричным (**д**), спектр сигнала дополняется только нечётными гармониками (**е**).

диапазон 0...0.5 частоты дискретизации и вызвать наложение спектров. Этот эффект не виден на (**б**), поскольку амплитуда таких гармоник очень мала. Чтобы увидеть наложившиеся в частотной области пики, необходимо построить тот же график в логарифмическом масштабе (**в**). На первый взгляд спектр выглядит как белый шум. Но это не шум. Это результат сложения большого числа гармоник, произошедшего в результате наложения.



**Рис. 11.8.** Наложение гармоник. На (а) и (б) показаны искажённый синусоидальный сигнал и его частотный спектр соответственно. Гармоники, лежащие выше точки 0.5, будут накладываться на гармоники, расположенные внутри диапазона 0...0.5. На (в) представлен тот же частотный спектр, но в логарифмическом масштабе, что позволяет увидеть наличие множества наложившихся пиков с малой амплитудой.

Важно иметь в виду, что в данном примере искажение сигнала происходит после дискретизации. Если бы оно вносилось в аналоговый сигнал, то «мешающие» гармоники можно было бы удалить с помощью антиэлайзингового фильтра. Такие проблемы при гармоническом анализе возникают только тогда, когда в цепи обработки уже дискретного сигнала присутствует нелинейный элемент. Но даже и в этом случае уровень гармоник наложения обычно настолько мал, что ими можно пренебречь.

Использование гармоник важно ещё по одной причине: они позволяют объяснить, почему дискретное преобразование Фурье рассматривает сигналы во временной и в частотной области как периодические. В частотной области сигнал на выходе  $N$ -точечного ДПФ представляет собой совокупность ( $N/2 + 1$ ) отсчётов частоты, равномерно распределённых на оси частот. Частоты в промежутках между этими отсчётами можно рассматривать либо равными нулю, либо считать их несуществующими. В любом случае в синтезе временного сигнала они не участвуют. То есть дискретный спектр состоит из отдельных гармоник, а не является непрерывным рядом частот. Это заставляет сигнал во временной области быть периодическим с периодом, соответствующим наименьшей частоте, входящей в состав спектра, т. е. основной частоте. Если не рассматривать постоянную составляющую, то наименьшая частота, представляемая в частотной области, совершаёт одно полное колебание за  $N$  дискретных отсчётов сигнала во временной области. То есть период повторения временного сигнала составляет  $N$  дискретных

отсчётов. Таким образом, если сигнал в одной области является дискретным, то в другой области он будет периодическим, и наоборот. Это справедливо для всего семейства Фурье-преобразований. ДПФ рассматривает сигнал в обеих областях (временной и частотной) как дискретный, а следовательно, и как периодический. При этом отсчёты представления сигнала в одной области определяют уровень соответствующих гармоник сигнала в другой области.

## 11.6. ЛЧМ-сигналы

Использование *линейно-частотно-модулированных (ЛЧМ) сигналов и ЛЧМ-систем*, импульсная характеристика которых относится к классу *ЛЧМ-сигналов*, является достаточно интересным методом решения задач обработки эхо-сигналов в таких приложениях, как радиолокация и гидролокация. Частотные характеристики ЛЧМ-системы показаны на Рис. 11.9. Амплитудно-частотная характеристика представляет собой постоянную величину единичного значения, а фазо-частотная является параболой:

$$\text{Phase } X[k] = \alpha k + \beta k^2. \quad (11.2)$$

Фазо-частотная характеристика ЛЧМ-системы

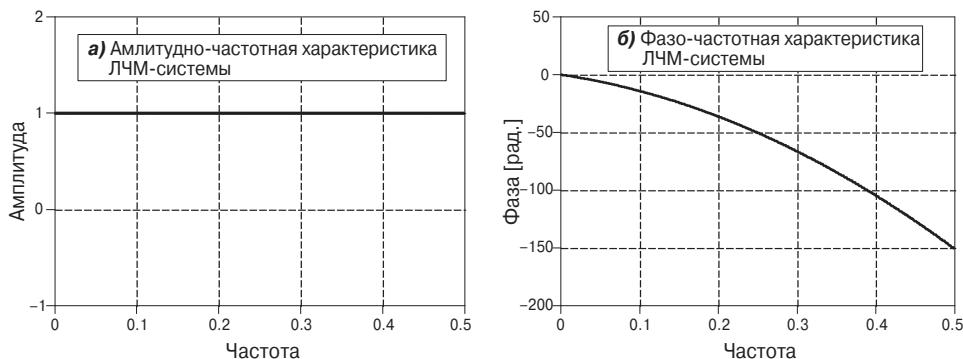


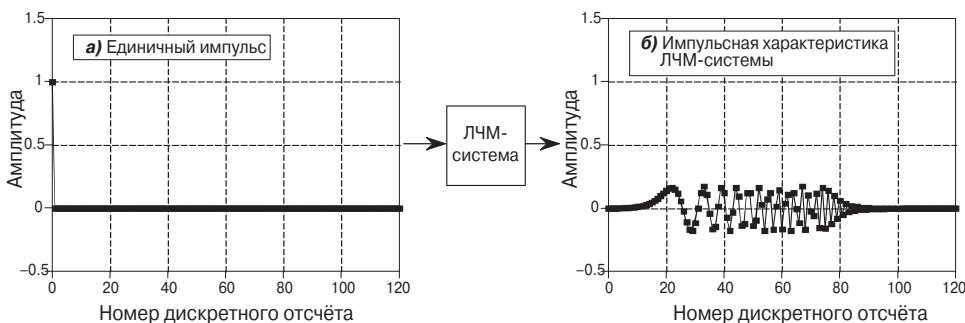
Рис. 11.9. Частотные характеристики системы с ЛЧМ.  
Модуль является постоянной величиной, а фаза — параболой.

Параметр  $\alpha$  задает линейный наклон фазо-частотной характеристики, определяющий смещение импульсной характеристики влево/вправо вдоль временной оси на требуемую величину. Параметр  $\beta$  задает кривизну фазовой характеристики. Эти два параметра должны выбираться так, чтобы значение фазы на частоте 0.5 (или  $k = N/2$ ) являлось множителем  $2\pi$ . Помните, что всякий раз, когда фаза задаётся вручную, её значения в точках 0 и 0.5 должны быть равны нулю (или быть множителем  $2\pi$ , что то же самое).

На Рис. 11.10 показан единичный импульс, поступающий на вход ЛЧМ-системы, и соответствующая реакция на её выходе (импульсная характеристика). Импульсная характеристика представляет собой колебательный процесс, который начинается на низких частотах и постепенно повышает частоту колебаний. В анг-

лийском языке ЛЧМ-сигнал называется «chirp-signal» (чириканье, щебетание), потому что если его подать на динамик, то он будет звучать как щебетание птицы.

Ключевой особенностью ЛЧМ-систем является *полная обратимость*: если подать ЛЧМ-сигнал на вход *обратной ЛЧМ-системы*, на выходе будет восстановлен исходный единичный импульс. Обратная ЛЧМ-система должна иметь АЧХ с таким же постоянным единичным значением и противоположную по отношению к прямой ЛЧМ-системе фазовую характеристику. А это, как говорилось в предыдущей главе, означает, что импульсная характеристика обратной ЛЧМ-системы «переворачивается» слева направо по сравнению с импульсной характеристикой прямой ЛЧМ-системы. Вы скажете: это интересно, но для чего всё это нужно?



**Рис. 11.10.** ЛЧМ-система. Импульсная характеристика ЛЧМ-системы представляет собой ЛЧМ-сигнал.

Рассмотрим, как функционирует радиолокационная система. Направленная антenna радиолокатора излучает короткий импульс радиоволны. Самолёт или какой-то иной объект отражает часть энергии этого радиоимпульса обратно в сторону радиолокационного приёмника, расположенного вблизи или совмещённо с излучающей антенной. Поскольку скорость распространения радиоволн является величиной постоянной, интервал времени между излучением импульса и моментом приёма его отражения позволяет рассчитать расстояние до объекта. Здесь возникает первое требование к так называемому зондирующему импульсу: он должен быть как можно короче. Например, если длительность импульса составляет 1 мкс, то протяжённость его отражения будет соответствовать 300 м. А это означает, что измерять расстояние до объектов мы сможем приблизительно с таким же разрешением по дальности. Если нам нужно более высокое разрешение, необходимо использовать более короткий импульс.

Второе требование является очевидным: если мы хотим увеличить дальность обнаружения, мы должны обеспечить большую энергию зондирующего импульса. К сожалению, между первым и вторым требованием (большая энергия/малая длительность) возникает противоречие. Электрическая мощность, необходимая для формирования импульса, определяется отношением энергии импульса к его длительности. Поэтому требование одновременного увеличения энергии и уменьшения длительности импульса ограничивается электрической мощностью, рассеиваемой в системе. Следует ограничивать мощность, рассеиваемую выходными цепями передатчика, чтобы они не вышли из строя.

ЛЧМ-сигналы позволяют преодолеть это ограничение. Перед тем как зондирующий импульс поступает на выходные цепи передатчика, он проходит через ЛЧМ-систему. Простой короткий импульс преобразуется в ЛЧМ-сигнал. Такой сигнал излучается в пространство, отражается от воздушных целей и поступает на радиолокационный приёмник. В приёмнике он поступает на обратную ЛЧМ-систему, восстанавливающую исходный короткий импульс. Таким образом, часть системы, непосредственно измеряющая время распространения сигнала и дальность до целей, работает с короткими импульсами, а цепи излучения/приёма, для которых важна электрическая мощность, работают с длительными сигналами. Такой подход к формированию зондирующего импульса в современных радиолокационных станциях является основополагающим.

## БЫСТРОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ

Существует несколько способов вычисления *дискретного преобразования Фурье (ДПФ)*. Два из них — решение системы линейных уравнений и корреляционный метод — обсуждались в Главе 8. В данной главе рассматривается ещё один метод — *быстрое преобразование Фурье (БПФ)*. Несмотря на то что алгоритм БПФ приводит к тем же самым результатам, что и остальные методы, но он оказывается невероятно эффективным с позиции минимизации вычислительной сложности и позволяет сократить время обработки сигнала в сотни раз. Разница в скорости обработки примерно того же порядка, как между пешей прогулкой и полётом на реактивном самолёте. Если бы не было БПФ, многое из того, что описано в этой книге, нельзя было бы реализовать на практике. Несмотря на то что алгоритм БПФ может быть представлен всего несколькими десятками строк программного кода, он является одним из сложнейших алгоритмов ЦОС. Но отчаяваться не стоит: вы можете использовать готовые процедуры, не вдаваясь в суть выполняемых в них преобразований.

### 12.1. Комплексное ДПФ для действительных сигналов

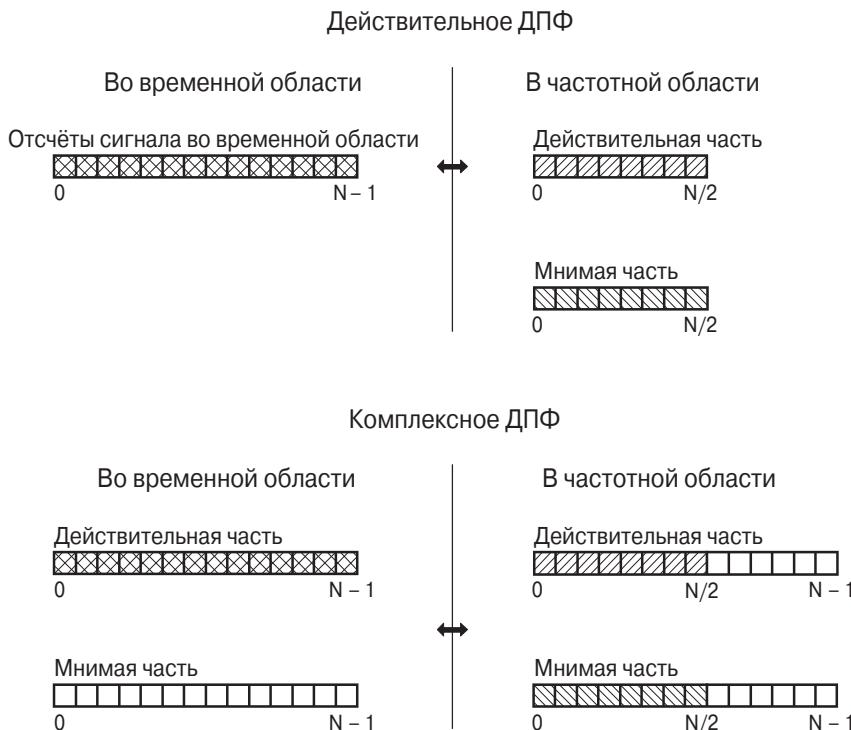
Следует отдать должное Дж. Кули и Дж. Тьюки, явившим свету алгоритм БПФ. Их легендарная статья «*An algorithm for the machine calculation of complex Fourier Series*» была опубликована в 1965 году в журнале «*Mathematics Computation*» (№ 19, стр. 297...301). Однако сама идея возникла ещё раньше. Более чем за сто лет до этого такой же метод использовал великий немецкий математик Карл Фридрих Гаусс (1777 – 1855). Но его работы по данной теме оказались забытыми из-за отсутствия технических средств, в которых этот подход можно было бы применить на практике, т. е. цифровых компьютеров. Кули и Тьюки удостоены чести считаться первооткрывателями только по той простой причине, что они разработали алгоритм БПФ именно в тот момент, когда он потребовался больше всего, — на заре компьютерной революции.

В основу алгоритма БПФ положен метод *комплексного ДПФ*, который является более сложным, чем *действительное ДПФ*, и обсуждается в последних четырёх главах книги. Действительное и комплексное ДПФ — это разные преобразования. Они названы в соответствии с типами данных, которые используются при обработке сигналов: комплексное ДПФ работает с комплексными числами, действительное ДПФ — с действительными числами. Подробнее о комплексных числах и комплексном ДПФ говорится в Главе 29; там же рассматриваются особенности алгоритма БПФ. В этой главе мы расскажем только о том, как применить

алгоритм БПФ для вычисления действительного ДПФ, не углубляясь в сложные математические преобразования.

Поскольку алгоритм БПФ предназначен для вычисления комплексного ДПФ, необходимо понять, как происходит преобразование формата данных из действительного в комплексный и обратно. На Рис. 12.1 сопоставляются наборы данных действительного и комплексного ДПФ. В случае действительного ДПФ  $N$  отсчётов *дискретного сигнала*, представленного во *временной области*, преобразуются в два массива, по  $(N/2 + 1)$  отсчётов в каждом, частотной формы представления этого сигнала. Во временной области имеется одна последовательность дискретных отсчётов сигнала, а в частотной области сигнал состоит из двух частей — действительной и мнимой. Отсчёты действительной части соответствуют амплитудам косинусных, а отсчёты мнимой части — амплитудам синусных компонентов разложения. После изучения предыдущих глав всё это должно быть вполне понятно.

Комплексное ДПФ преобразует два массива длиной  $N$  отсчётов каждый, описывающих сигнал во временной области, в два новых массива той же длины, описывающих спектр сигнала в частотной области. В обеих парах один массив соответствует действительной части, а другой — мнимой части сигнала. При этом во всех четырёх массивах хранятся обычные действительные числа. (Если вы знакомы



**Рис. 12.1.** Сравнение действительного и комплексного ДПФ. При выполнении действительного ДПФ массив, содержащий  $N$  отсчётов сигнала, представленного во временной области, преобразуется в два массива, по  $(N/2 + 1)$  отсчётов в каждом, описывающих сигнал в частотной области. При выполнении комплексного ДПФ используются два массива отсчётов во временной области и два массива отсчётов в частотной области; все четыре массива содержат равное число отсчётов —  $N$ . С помощью штриховки показаны те фрагменты сигналов, которые оказываются одинаковыми для обоих типов преобразования.

мы с комплексными числами, вам должен быть понятен смысл сказанного. Минимальная единица  $j$  в записи элементов массивов отсутствует, так как она нужна только в математических выкладках. Напомним, что результатом применения оператора  $\text{Im}(\ )$  является действительное число.)

Пусть имеется  $N$  отсчётов дискретного сигнала и требуется выполнить действительное ДПФ, используя для этой цели комплексное ДПФ, для вычисления которого можно воспользоваться алгоритмом БПФ. В этом случае прежде всего следует представить действительный входной сигнал как комплексный, разместить известные  $N$  отсчётов в массиве действительной части, а во все ячейки массива мнимой части записать нули. Затем, выполнив комплексное ДПФ, мы получим комплексный частотный спектр в виде массивов действительной и мнимой частей длиной  $N$  отсчётов каждый. Первые  $(N/2 + 1)$  отсчётов этих массивов соответствуют результату выполнения действительного ДПФ.

В Главе 10 говорилось о периодичности спектра дискретных сигналов при учёте области отрицательных частот (**Рис. 10.9**). Границами частотного интервала, соответствующего одному периоду спектра, могут быть произвольно выбраны:  $-1.0$  и  $0$ ,  $-0.5$  и  $0.5$ ,  $0$  и  $1.0$  или любые пары чисел, разность которых равна единице, т. е. частоте дискретизации (имеется в виду частота, измеряемая в долях частоты дискретизации). В результате выполнения ДПФ получается один период спектра, в котором частоты располагаются по возрастанию на интервале  $0\dots1.0$ . То есть отсчёты массивов с номерами  $0\dots(N - 1)$  соответствуют значениям частот  $0\dots1.0$ . При этом положительные частоты основного диапазона  $0\dots0.5$  представлены элементами массивов с номерами  $0\dotsN/2$ . Оставшиеся элементы  $(N/2 + 1)\dots(N - 1)$  соответствуют области отрицательных частот и обычно после выполнения ДПФ исключаются из дальнейшего рассмотрения.

Вычисление действительного *обратного ДПФ* с помощью комплексного обратного ДПФ оказывается несколько более сложной операцией, потому что в этом случае требуется для каждой положительной частоты спектра сигнала правильно образом сформировать соответствующую ей отрицательную частоту. Отметим, что выполнение действительного и комплексного ДПФ должно давать одинаковые результаты для элементов частотного спектра с порядковыми номерами  $0\dotsN/2$ . Кроме того, для действительной части выполняется равенство элементов с номерами  $(N/2 - 1)$  и  $(N/2 + 1)$ ,  $(N/2 - 2)$  и  $(N/2 + 2)$  и т. д. Последняя такая пара — это элементы с номерами  $(N - 1)$  и  $N$ . Для массива, хранящего отсчёты мнимой части спектра, справедливо похожее правило: пары элементов с такими же порядковыми номерами равны друг другу по модулю, но противоположны по знаку. Заметим, что у элементов  $0$  и  $N/2$  пары отсутствуют. Указанные свойства иллюстрирует **Рис. 10.9**. Таким образом, чтобы сформировать массивы, описывающие сигнал в частотной области, необходимо сначала записать в них отсчёты действительной и мнимой частей спектра с номерами  $0\dotsN/2$ , а затем выполнить некоторую дополнительную подпрограмму, которая сформирует остальные отсчёты:  $(N/2 + 1)\dots(N - 1)$ . **Программа 12.1** выполняет эти действия. Чтобы убедиться в правильности её работы, выполните обратное БПФ и проверьте содержимое массива мнимой части временного представления сигнала. Если все верно, то во всех ячейках указанного массива должны получиться нули или очень маленькие числа (миллионные доли) из-за ошибок округления.

### Программа 12.1

```

6000 'ФОРМИРОВАНИЕ ОТРИЦАТЕЛЬНЫХ ЧАСТОТ
6010 'Подпрограмма дополняет массив отсчётов спектра отрицательными частотами.
6020 'Перед входом в подпрограмму необходимо указать длину массива - N%
6030 'заполнить элементы 0...N%/2 в массивах отсчётов REX[ ] и IMX[ ].
6040 'Подпрограмма записывает в REX[ ] и IMX[ ] элементы с номерами 0...N%-1.
6050 '
6060 FOR K% = (N%/2+1) TO (N%-1)
6070 REX[K%] = REX[N%-K%]
6080 IMX[K%] = -IMX[N%-K%]
6090 NEXT K%
6100 '
6110 RETURN

```

## 12.2. Как работает алгоритм БПФ

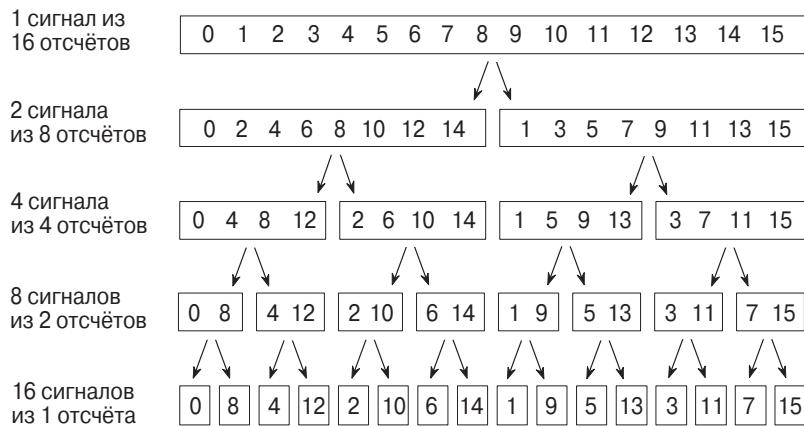
Алгоритм БПФ достаточно сложен, и в данной главе мы рассмотрим только наиболее важные моменты, связанные с его практическим использованием. Всё, что касается комплексной математики, пока обойдём стороной. Если вам знаком аппарат комплексных чисел, вы сможете понять суть алгоритма, излагаемую «между строк». Не огорчайтесь, если какие-либо тонкости алгоритма не будут вам совершенно ясны: даже среди инженеров и ученых, постоянно использующих БПФ, лишь немногие смогут сесть и с лёгкостью написать реализующую его программу «с нуля».

Когда говорят о комплексном сигнале, то подразумевают последовательность  $N$  комплексных отсчётов, описывающую сигнал в одной из двух возможных областей: временной или частотной. Каждый отсчёт комплексного сигнала состоит в свою очередь из двух чисел, называемых действительной и мнимой частью комплексного числа. Например, комплексный отсчёт  $X[42]$  состоит из пары чисел —  $Re\{X[42]\}$  и  $Im\{X[42]\}$ . Другими словами, любое комплексное число может быть представлено в виде пары обычных действительных чисел. Операция умножения двух комплексных чисел включает четыре операции умножения, однако полученные произведения комбинируются таким образом, чтобы снова получить два компонента нового комплексного отсчёта (9.1). При обсуждении того, как работает алгоритм БПФ, используют терминологию комплексной математики. Все используемые понятия, такие как «сигнал», «отсчёт», «величина», «элемент массива» и др., подразумевают наличие двух частей: действительной и мнимой.

Выполнение алгоритма БПФ разбивается на три этапа. Сначала исходная последовательность, описывающая сигнал во временной области и состоящая из  $N$  отсчётов, разбивается на  $N$  отдельных сигналов, по одному отсчёту в каждом (декомпозиция). Следующий этап связан с переходом от  $N$  временных сигналов к  $N$  частотным спектрам. Завершается работа алгоритма синтезом единого спектра на основе полученных  $N$  спектров.

Процедура декомпозиции сигнала во временной области поясняется на Рис. 12.2. Дискретная последовательность из 16 отсчётов разбивается за 4 шага. На первом шаге получаются два сигнала, по 8 отсчётов в каждом; на втором — четыре сигнала, по 4 отсчёта в каждом и т. д. Разложение ведётся по одному и тому

же принципу до тех пор, пока не получится  $N$  сигналов по одному отсчёту. При каждом разбиении отсчёты с чётными номерами отделяются от отсчётов с нечётными номерами, образуя два новых сигнала в 2 раза меньшей длины. Посмотрите на Рис. 12.2, и вам всё сразу станет понятно. Число шагов разбиения сигнала выражается формулой  $\log_2 N$ : для разбиения сигнала из 16 ( $= 2^4$ ) отсчётов нужно 4 шага, для сигнала из 512 ( $= 2^7$ ) отсчётов — 7, для сигнала из 4096 ( $= 2^{12}$ ) отсчётов — 12 шагов и т. д. Запомните эту формулу, потому что она ещё не раз встретится вам в данной главе.



**Рис. 12.2.** Разбиение (декомпозиция) сигнала при выполнении алгоритма БПФ. Сигнал из  $N$  отсчётов разбивается на  $N$  сигналов, по одному отсчёту в каждом. На каждом этапе происходит перегруппировка отсчётов, при которой в одну группу выбираются отсчёты с чётными номерами, а в другую — с нечётными.

Теперь, когда вы разобрались с принципом декомпозиции сигнала, расскажем о том, как упростить данную процедуру. На самом деле никаких перестановок отсчётов внутри сигналов не требуется. На Рис. 12.3 показана схема, которая описывает принцип упорядочивания дискретных отсчётов. Слева записаны номера отсчётов исходной последовательности, представленные в двоичном коде, справа — порядок размещения этих же отсчётов в результирующей последовательности также в двоичном коде. Переход от левого столбца к правому сводится к перестановке битов каждого числа в обратном порядке. Например, числу 3 левого столбца, имеющему двоичный код 0011, соответствует число 12 в правом столбце, которое имеет код 1100, числу 14 (1110) — число 7 (0111) и т. д. Таким образом, декомпозиция сигнала осуществляется с помощью алгоритма сортировки его отсчётов на основе *бит-реверсной адресации*, которая заключается в перегруппировке  $N$  отсчётов сигнала в новую последовательность, образующуюся по схеме с изменением порядка следования битов в индексах элементов массива (см. Рис. 12.3).

Вторым шагом выполнения БПФ, следующим за процедурой разбиения на  $N$  сигналов по одному отсчёту, является переход к спектрам этих сигналов. Ничего не может быть проще, чем спектр сигнала, состоящего из одного-единственного отсчёта, поскольку спектр такого сигнала равен соответствующему базисному сигналу ДПФ. На этом шаге не нужно выполнять вообще никаких действий. Хотя второй шаг БПФ не требует проведения никаких программных операций, следует

Порядок отсчётов в исходной последовательности		Порядок отсчётов после бит-реверсной сортировки	
Номер в десятичном коде	Номер в двоичном коде	Номер в десятичном коде	Номер в двоичном коде
0	0000	0	0000
1	0001	8	1000
2	0010	4	0100
3	0011	12	1100
4	0100	2	0010
5	0101	10	1010
6	0110	6	0100
7	0111	14	1110
8	1000	1	0001
9	1001	9	1001
10	1010	5	0101
11	1011	13	1101
12	1100	3	0011
13	1101	11	1011
14	1110	7	0111
15	1111	15	1111



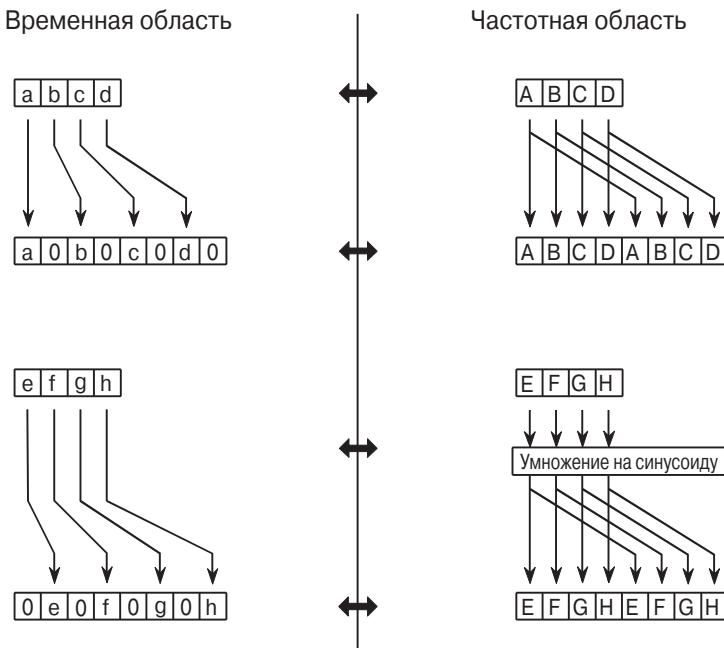
**Рис. 12.3.** Бит-реверсная адресация при реализации БПФ. Декомпозиция сигнала может быть выполнена на основе изменения порядка следования его отсчётов.

помнить, что каждый имеющийся отсчёт теперь становится отсчётом спектра сигнала и относится к частотной, а не к временной области.

Последний шаг БПФ связан с объединением  $N$  спектров в порядке, в точности обратном тому, который использовался на первом шаге при разбиении исходного сигнала, представленного во временной области. Именно этот шаг алгоритма БПФ является самым сложным. К сожалению, применённую на первом шаге и упростившую все действия процедуру бит-реверсной сортировки теперь применить не удастся и придётся последовательно пройти все этапы объединения отсчётов. На первом этапе 16 частотных спектров (по 1 отсчёту) группируются в 8 спектров (по 2 отсчёта); на втором этапе полученные 8 спектров объединяются в 4 спектра (по 4 отсчёта) и т. д. На последнем этапе должен получиться единственный спектр, состоящий из 16 отсчётов, который и является искомым представлением исходного сигнала в частотной области, т. е. окончательным результатом работы алгоритма БПФ.

На Рис. 12.4 показано, каким образом два спектра, каждый из которых состоит из 4 отсчётов, объединяются в один спектр из 8 отсчётов. Цель такого синтеза заключается в том, чтобы выполнить обратное преобразование по отношению к разбиению сигнала с перестановкой отсчётов, выполненному ранее во временной области. Другими словами, преобразование, выполняемое в частотной области, должно соответствовать процедуре объединения двух сигналов, по 4 отсчёта в каждом, выполняемой во временной области. Рассмотрим данную операцию на примере объединения во временной области сигналов  $abcd$  и  $efgh$  (каждый из которых содержит 4 отсчёта). Сигнал из 8 отсчётов формируется в два действия.

Сначала производится дополнение каждого из двух сигналов нулями. При этом нули вводятся между первоначально имеющимися отсчётами, а число отсчётов увеличивается до 8. Затем соответствующие элементы сигналов складываются. Таким образом, сначала сигнал  $abcd$  преобразуется в  $a0b0c0d0$ , а сигнал  $efgh$  — в сигнал  $0e0f0g0h$ ; последующее поэлементное сложение двух сигналов даёт  $aebfcgdh$ . Операция дополнения нулями во временной области соответствует повторению спектра в частотной области. Следовательно, в алгоритме БПФ происходит объединение частотных спектров путём повторения и последующего поэлементного сложения.

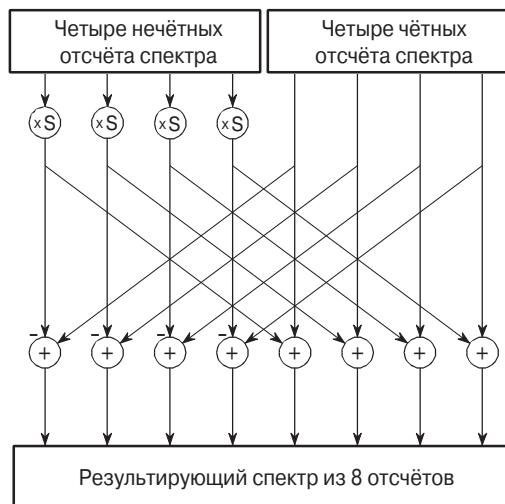


**Рис. 12.4.** Операция синтеза в алгоритме БПФ. Дополнение сигналов во временной области нулями приводит к повторению спектров в частотной области. Сдвиг на один отсчёт во временной области соответствует умножению на синусоиду в частотной области.

Чтобы при сложении сигналов во временной области их отсчёты совместились, нули должны вводиться в эти сигналы по-разному: в первом — в нечётные элементы, а во втором — в чётные. Поэтому один из сигналов (на Рис. 12.4 сигнал  $0e0f0g0h$ ) оказывается смещённым по времени на один отсчёт вправо. Такой сдвиг во временной области соответствует умножению на комплексную синусоиду в частотной области. Чтобы доказать данный факт, напомним, что сдвиг сигнала во временной области эквивалентен свёртке этого сигнала со смещённой дискретной дельта-функцией (единичным импульсом). Операция свёртки двух сигналов во временной области соответствует в частотной области умножению спектров этих сигналов, а спектр смещённой дельта-функции является синусоидой (Рис. 11.2).

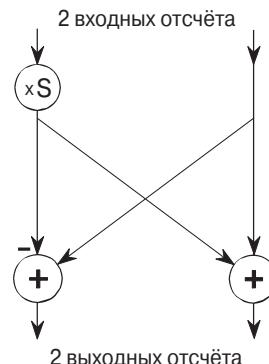
Структурная схема алгоритма объединения двух спектров, состоящих из 4 отсчётов каждый, в единый спектр из 8 отсчётов показана на Рис. 12.5. Можно упростить эту схему, если заметить, что данная структура собрана из типовых блоков

одного вида, изображённого на **Рис. 12.6**. Такой типовой структурный элемент называется «бабочкой» из-за внешнего сходства с бабочкой, распахнувшей свои крылья. Операция «бабочка» является основной операцией алгоритма БПФ и сводится к преобразованию одной пары комплексных чисел в другую.



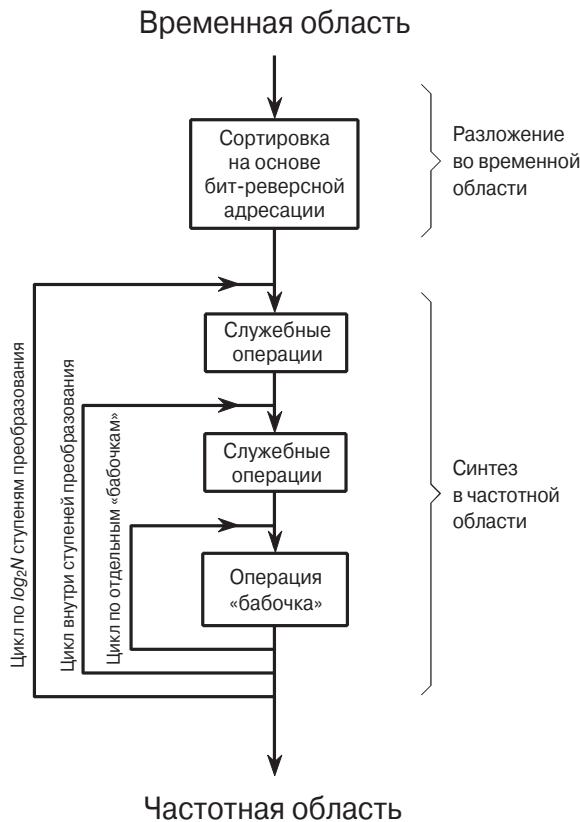
**Рис. 12.5.** Структурная схема операции синтеза в алгоритме БПФ. Данная структура отображает процесс объединения двух частотных спектров из 4 точек в один спектр из 8 точек. Символом « $\times S$ » обозначено умножение на синусоиду соответствующей частоты.

**Рис. 12.6.** «Бабочка» — основной элемент в структурной схеме алгоритма БПФ, который преобразует одну пару комплексных чисел в другую.



В общем виде алгоритм БПФ рассмотрен на **Рис. 12.7**. Разложение во временной области производится с помощью процедуры сортировки на основе бит-реверсной адресации. Переход от временной области к частотной на схеме не показан, так как для него не требуется выполнения никаких практических операций. Синтез в частотной области выполняется с помощью трёх вложенных циклов. Обработка внешнего цикла производится за  $\log_2 N$  итераций, что соответствует количеству уровней разложения (**Рис. 12.2**). Но теперь, на этапе синтеза, движение происходит в обратном направлении (снизу вверх). В среднем цикле осуществляется поочерёдный перебор отдельных спектров, участвующих в синтезе на дан-

ном уровне. Третий по уровню вложенности цикл предназначен для выполнения поэлементной обработки данных на основе операции «бабочка». В блоках служебных операций учитывается формирование индексов, используемых для организации циклов, а также вычисление отсчётов синусоиды, необходимых для выполнения «бабочек». Теперь мы можем перейти к основной части данной главы — к программной реализации алгоритма БПФ.



**Рис. 12.7.** Блок-схема алгоритма БПФ. Выполнение алгоритма БПФ происходит в три этапа: разложение во временной области сигнала из  $N$  отсчётов на  $N$  сигналов, по одному отсчёту в каждом; нахождение спектра каждого из полученных  $N$  сигналов (никаких операций не требуется); синтез единого спектра по  $N$  отдельным спектрам.

## 12.3. Примеры программной реализации БПФ

Как говорилось в Главе 8, *действительное ДПФ* может быть выполнено с помощью вычисления *корреляции* сигнала, представленного во временной области, с косинусными и синусными сигналами набора частот (**Программа 8.2**). Для вычисления *комплексного ДПФ* можно использовать тот же корреляционный подход. **Программа 12.2** выполняет такое преобразование.

### Программа 12.2

```

5000 'КОМПЛЕКСНОЕ ДПФ НА ОСНОВЕ КОРРЕЛЯЦИОННОГО МЕТОДА
5010 'Порядок ДПФ N% и массивы действительных и мнимых компонент XR[ ] и XI[ ]
5020 'заполняются до входа в подпрограмму.
5030 'Найденные отсчёты спектра сигнала размещаются в массивах REX[ ] и IMX[ ]
5040 'Все сигналы содержат элементы с номерами 0...(N%-1).
5050 '
5060 PI = 3.14159265      'Константа
5070 '
5080 FOR K% = 0 TO N%-1    'Обнуление всех элементов массивов REX[ ] и IMX[ ],
5090 REX[K%] = 0            'которые используются для накопления суммы
5100 IMX[K%] = 0
5110 NEXT K%
5120 '
5130 FOR K% = 0 TO N%-1    'Цикл по всем частотам
5140 FOR I% = 0 TO N%-1    'Цикл по отсчётам комплексной синусоиды: SR и SI
5150 '
5160 SR = COS(2*PI*K%*I%/N%) 'Вычисление отсчёта комплексной синусоиды
5170 SI = -SIN(2*PI*K%*I%/N%)
5180 REX[K%] = REX[K%] + XR[I%]*SR - XI[I%]*SI
5190 IMX[K%] = IMX[K%] + XR[I%]*SI + XI[I%]*SR
5200 '
5210 NEXT I%
5220 NEXT K%
5230 '
5240 RETURN

```

Алгоритм БПФ позволяет улучшить (с точки зрения минимизации вычислительных затрат) работу данной программы, никак не изменяя результата её выполнения. Реализация БПФ производится с помощью **Программы 12.3** и **Программы 12.4**, которые написаны на языках Бейсик и Фортран соответственно.

Рассмотрим сначала подпрограмму БПФ на языке Бейсик (**Программа 12.3**). Как уже было сказано, результат её выполнения полностью совпадает с результатом работы программы, использующей корреляционный метод (**Программа 12.2**). Однако время выполнения в данном случае оказывается намного меньше. В основу программы положен алгоритм, блок-схема которого показана на **Рис. 12.7**. Исходные данные поступают в массивы REX[ ] и IMX[ ], включающие элементы 0...(N – 1). После выполнения БПФ вычисленные отсчёты спектра сигнала перед возвратом из подпрограммы также помещаются в массивы REX[ ] и IMX[ ]. В этом состоит ещё одно достоинство алгоритма БПФ: для входных, выходных и промежуточных данных используются одни и те же массивы памяти. Это позволяет сделать вычисление БПФ более быстрым на аппаратном уровне. Такой подход, при котором на протяжении всей работы подпрограммы ведётся обращение к одной и той же области памяти данных, получил название *локальных, или местных, вычислений*.

Существует множество возможных реализаций программ БПФ, каждая из которых приводит к одному и тому же результату, но отличается своими уникальными особенностями, которые вам следует уметь обнаруживать. Рассмотрим, например, программу БПФ на языке Фортран (**Программа 12.4**). В ней использован так называемый алгоритм БПФ с прореживанием по частоте. В отличие от него в

предыдущей программе, написанной на языке Бейсик, использовался *алгоритм БПФ с прореживанием по времени*. В случае прореживания по частоте перестановка отсчётов с использованием *бит-реверсной адресации* выполняется после вложенных циклов. Известны также алгоритмы БПФ, позволяющие вообще обойтись без бит-реверсной адресации. Несмотря на большое количество различных вариантов реализации БПФ, все они обладают примерно равными показателями эффективности. Поэтому очень углубляться в проблему выбора конкретного решения не стоит.

### **Программа 12.3. Быстрое преобразование Фурье: язык Бейсик**

```

1000 'БЫСТРОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ
1010 'Порядок ДПФ N% и массивы действительных и мнимых компонент REX[ ] и IMX[ ]
1020 'заполняются до входа в подпрограмму. Выходные данные размещаются в
1030 'тех же массивах REX[ ] и IMX[ ]. Номера элементов массивов: 0...(N%-1).
1040 '
1050 PI = 3.14159265      'Константа
1060 NM1% = N%-1
1070 ND2% = N%/2
1080 M% = CINT(LOG(N%)/LOG(2))
1090 J% = ND2%
1100 '
1110 FOR I% = 1 TO N%-2    'Бит-реверсная сортировка
1120 IF I% >= J% THEN GOTO 1190
1130 TR = REX[J%]
1140 TI = IMX[J%]
1150 REX[J%] = REX[I%]
1160 IMX[J%] = IMX[I%]
1170 REX[I%] = TR
1180 IMX[I%] = TI
1190 K% = ND2%
1200 IF K% > J% THEN GOTO 1240
1210 J% = J%-K%
1220 K% = K%/2
1230 GOTO 1200
1240 J% = J%+K%
1250 NEXT I%
1260 '
1270 FOR L% = 1 TO M%      'Цикл по уровням разложения
1280 LE% = CINT(2^L%)
1290 LE2% = LE%/2
1300 UR = 1
1310 UI = 0
1320 SR = COS(PI/LE2%)    'Вычисление отсчётов синуса и косинуса
1330 SI = -SIN(PI/LE2%)
1340 FOR J% = 1 TO LE2%    'Цикл по спектрам внутри уровня
1350 JM1% = J%-1
1360 FOR I% = JM1% TO NM1% STEP LE%  'Цикл по отдельным «бабочкам»
1370 IP% = I%+LE2%
1380 TR = REX[IP%]*UR - IMX[IP%]*UI  'Операция «бабочка»
1390 TI = REX[IP%]*UI + IMX[IP%]*UR
1400 REX[IP%] = REX[I%]-TR

```

```

1410 IMX[IP%] = IMX[I%]-TI
1420 REX[I%] = REX[I%]+TR
1430 IMX[I%] = IMX[I%]+TI
1440 NEXT I%
1450 TR = UR
1460 UR = TR*SR - UI*SI
1470 UI = TR*SI + UI*SR
1480 NEXT J%
1490 NEXT L%
1500 '
1510 RETURN

```

#### **Программа 12.4. Быстрое преобразование Фурье: язык Фортран**

```

SUBROUTINE FFT(X,M)
COMPLEX X(4096),U,S,T
PI=3.14159265
N=2**M
DO 20 L=1,M
LE=2** (M+1-L)
LE2=LE/2
U=(1.0,0.0)
S=CMPLX(COS(PI/FLOAT(LE2)), -SIN(PI/FLOAT(LE2)))
DO 20 J=1,LE2
DO 10 I=J,N,LE
IP=I+LE2
T=X(I)+X(IP)
X(IP)=(X(I)-X(IP))*U
10   X(I)=T
20   U=U*S
ND2=N/2
NM1=N-1
J=1
DO 50 I=1,NM1
IF(I.GE.J) GO TO 30
T=X(J)
X(J)=X(I)
X(I)=T
30   K=ND2
40   IF(K.GE.J) GO TO 50
J=J-K
K=K/2
GO TO 40
50   J=J+K
RETURN
END

```

**Примечание.** Входные данные для программы: M — число степеней разложения, равное логарифму по основанию 2 от числа отсчётов сигнала (для сигнала из 256 отсчётов M = 8, для сигнала из 4096 отсчётов M = 12 и т. д.); X( ) — массив комплексных отсчётов сигнала, который требуется обработать. К моменту завершения работы подпрограммы в массиве X( ) размещаются отсчёты спектра сигнала — результат выполнения ДПФ. Обращаем внимание на то, что алгоритм использует элементы X(1)...X(N), начиная нумерацию с единицы.

Заметное отличие между двумя реализациями БПФ наблюдается также в том, как данные передаются в подпрограмму, и в том, как они выводятся из нее. В подпрограмме на Бейсике данные вводятся и выводятся через массивы REX[ ] и IMX[ ], индексация в которых ведётся на интервале  $0 \dots (N - 1)$ . При программировании на языке Фортран создается массив комплексных чисел X( ) и, кроме того, изменяется нумерация ячеек массива:  $1 \dots N$ . Так как X( ) — массив комплексных чисел, то, по сути, каждый его элемент содержит пару чисел — реальную и мнимую части комплексного числа. Кроме отсчётов сигнала в подпрограмму должен быть передан порядок ДПФ. В варианте, написанном на Бейсике, порядок ДПФ указывается в переменной N%. На языке Фортран для этой цели используется переменная M, определяемая как  $\log_2 N$ : для ДПФ 256-го порядка  $N = 8$ ; если порядок ДПФ равен 4096, то  $N = 12$ . Дело в том, что каждый программист по-своему организует интерфейс между подпрограммой БПФ и вызывающей её основной программой. Особенно часто возникает путаница при использовании чисел  $1 \dots N$  при адресации массивов на Фортране. При описании алгоритмов и программ, опубликованных в большинстве изданий, посвящённых ЦОС, предполагается, что адресация массивов ведётся от нуля. Когда для адресации используются индексы  $1 \dots N$ , что свойственно Фортрану, то симметрия спектра в частотной области наблюдается уже относительно отсчётов с порядковыми номерами 1 и  $(N/2 + 1)$ , а не 0 и  $N/2$ , как говорилось выше.

Использование комплексного ДПФ для вычисления действительного ДПФ имеет и другие замечательные преимущества. У комплексного ДПФ, по сравнению с действительным, наблюдается более чёткая симметрия между временной и частотной областями — частотно-временной дуализм принимает более строгую форму. Одним из следствий этого является то, что обратное ДПФ практически не отличается от прямого. В результате простейший способ выполнения обратного БПФ сводится к выполнению прямого БПФ с последующим масштабированием полученных отсчётов. Обратный алгоритм БПФ выполняет **Программа 12.5**.

### **Программа 12.5**

```

2000 'ПОДПРОГРАММА ОБРАТНОГО БЫСТРОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ
2010 'Порядок обратного ДПФ — N% и массивы действительных и мнимых компонент
2020 'REX[ ] и IMX[ ] заполняются до входа в подпрограмму. Выходные данные
2030 'размещаются в тех же массивах REX[ ] и IMX[ ], что и исходные данные.
2040 'Номера элементов всех массивов: 0 ... (N%-1).
2050 '
2060 FOR K% = 0 TO N%-1      'Изменение арифметического знака элементов IMX[ ]
2070 IMX[K%] = -IMX[K%]
2080 NEXT K%
2090 '
2100 GOSUB 1000              'Вычисление прямого БПФ (Прогр. 12.3)
2110 '
2120 FOR I% = 0 TO N%-1      'Деление на N% во временной области
2130 REX[I%] = REX[I%]/N%    'со сменой арифметического знака IMX[ ]
2140 IMX[I%] = -IMX[I%]/N%
2150 NEXT I%
2160 '
2170 RETURN

```

Допустим, вы скопировали код нужной вам подпрограммы БПФ в свою собственную программу. Как проверить, правильно ли она будет работать? Можно порекомендовать два приёма отладки. Заполните массив исходных данных произвольным набором элементов, полученным, например, от генератора случайных чисел. Обработайте их подпрограммой БПФ. Полученный спектр сигнала отправьте на обработку подпрограммой обратного БПФ и сравните результат с исходным сигналом. Они должны в точности совпасть друг с другом; отличие может наблюдаться только на уровне шумов округления (миллионные доли при одинарной точности вычислений). Второй способ проверки работоспособности основан на симметрии получаемых зависимостей. Если сигнал характеризуется нулевой мнимой частью во временной области (что справедливо в большинстве случаев), то в частотной области наблюдается симметрия относительно отсчётов 0 и  $N/2$ , о чём говорилось выше. Более того, из наличия симметрии в частотной области следует равенство нулю мнимой части сигнала во временной области после выполнения обратного БПФ (небольшие отклонения от нуля могут быть вызваны шумом округления).

## 12.4. Сравнение по точности и быстродействию

Подпрограмма вычисления ДПФ на основе корреляционного метода (в том виде, в котором она представлена в [Программе 12.2](#)) включает в себя два цикла, вложенных один в другой. При этом в каждом цикле выполняется перебор  $N$  отсчётов сигнала. Это означает, что время обработки пропорционально  $N$  раз по  $N$ , т. е.  $N^2$ . Поэтому время выполнения такой подпрограммы можно выразить следующей формулой:

$$\text{Время обработки} = k_{\text{ДПФ}} N^2. \quad (12.1)$$

**Время выполнения ДПФ.** Время, затрачиваемое на вычисление ДПФ корреляционным методом, пропорционально квадрату размерности ДПФ.

В этой формуле  $N$  — размерность ДПФ (число обрабатываемых отсчётов), а  $k_{\text{ДПФ}}$  — коэффициент пропорциональности. Если отсчёты синусов и косинусов вычисляются внутри вложенных циклов,  $k_{\text{ДПФ}}$  приблизительно равен 25 мкс при обработке на процессоре Pentium с частотой 100 МГц. Если отсчёты синусов и косинусов вычисляются заранее и хранятся в *справочной таблице*, величина  $k_{\text{ДПФ}}$  снижается примерно до 7 мкс. Например, если порядок ДПФ равен 1024, на выполнение подпрограммы потребуется около 25 с, т. е. примерно 25 мс на каждый отсчёт. Вот как медленно работает ДПФ!

Теперь оценим время выполнения БПФ. *Бит-реверсная адресация* требует ничтожно малых временных затрат. В каждой из  $\log_2 N$  ступеней БПФ необходимо выполнить  $N/2$  операций «бабочка». Это значит, что время выполнения подпрограммы можно оценить следующей приближённой зависимостью:

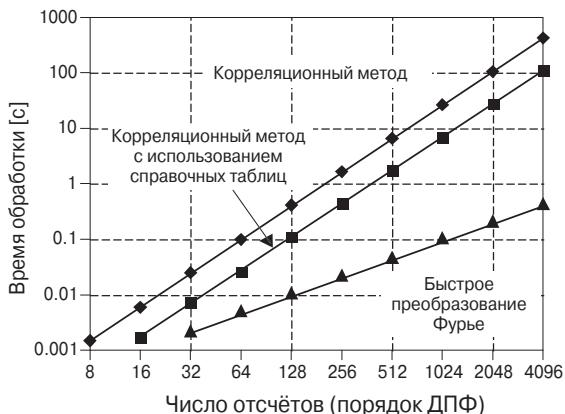
$$\text{Время обработки} = k_{\text{БПФ}} N \log_2 N. \quad (12.2)$$

**Время выполнения БПФ.** Время, затрачиваемое на вычисление ДПФ методом БПФ, пропорционально  $N$ , умноженному на логарифм от  $N$  по основанию 2.

Здесь коэффициент пропорциональности  $k_{БПФ}$  составляет около 10 мкс для процессора Pentium 100. Если порядок БПФ равен 1024, то на обработку требуется около 70 мс, т. е. 70 мкс на каждый отсчёт. Этот алгоритм более чем в 300 раз быстрее алгоритма, основанного на корреляционном методе<sup>1)</sup>.

Функция  $N \log_2 N$  не только меньше  $N^2$ , но и медленнее растет с ростом переменной  $N$ . При  $N = 32$  алгоритм БПФ примерно в 10 раз быстрее алгоритма, основанного на корреляционном методе, при  $N = 4096$  — уже в 1000 раз. Поэтому для малых  $N$  (в диапазоне 32...128) алгоритм БПФ очень важен, а для больших  $N$  (начиная с 1024-го и далее) он просто необходим. Эффективность вычисления ДПФ различными методами позволяет сравнить график, изображённый на Рис. 12.8.

**Рис. 12.8.** Эффективность вычисления ДПФ различными методами. Программа 12.2 вычисляет ДПФ корреляционным методом. Ускорить выполнение данного алгоритма позволяет предварительное вычисление отсчётов синусов и косинусов, которые затем сводятся в справочную таблицу. Алгоритм БПФ (Программа 12.3) превосходит эти алгоритмы по быстродействию, когда порядок ДПФ превышает 16. Время оценивалось для процессора Pentium с частотой 100 МГц.



Кроме преимущества в скорости вычислений, БПФ превосходит другие алгоритмы и по точности, так как уменьшение количества вычислительных операций обуславливает и уменьшение связанных с ними ошибок округления. Данное утверждение легко проверить, обработав произвольный сигнал сначала процедурой прямого, а затем обратного БПФ. Полученный сигнал будет отличаться от исходного на величину шума округления. Чтобы оценить этот шум, достаточно вычесть один сигнал из другого. Численно уровень шума выражается *среднеквадратическим отклонением*. Аналогичным образом следует обработать сигнал подпрограммой ДПФ на основе корреляционного метода, а затем соответствующей подпрограммой обратного ДПФ. Насколько велико различие в шуме округления, ясно из Рис. 12.9.

<sup>1)</sup> Для современного специализированного сигнального процессора ADSP-TS201 коэффициент пропорциональности можно оценить равным 1.7 нс. При этом время выполнения 1024-точечного БПФ оказывается немногим более 17 мкс. — Примеч. пер.

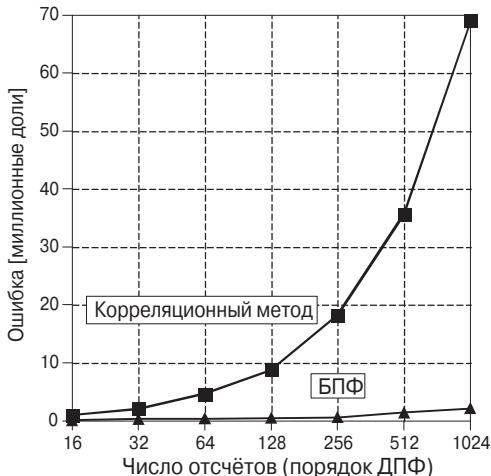


Рис. 12.9. Точность вычисления ДПФ. Кроме преимущества в быстродействии, алгоритм БПФ по сравнению с корреляционным методом обеспечивает повышение точности вычислений.

## 12.5. Дополнительное повышение быстродействия

Есть методы, позволяющие сделать БПФ ещё более быстрым, ориентировочно на 20...40%. В одном из таких методов разложение сигнала во временной области прекращается на два шага раньше, чем это было в предложенных выше примерах, т. е. после того, как образовались сигналы, состоящие из четырёх отсчётов каждый. Переход в частотную область осуществляется, минуя две последние ступени. При этом используется хорошо оптимизированный код, полученный с учётом уникальных свойств синусов и косинусов, содержащих по четыре отсчёта.

Другой метод повышения быстродействия алгоритма БПФ заключается в устранении избыточных вычислительных операций, которые можно исключить из алгоритма, пользуясь тем, что мнимая часть сигналов, представленных во временной области, равна нулю, а спектры таких сигналов в частотной области оказываются симметричными. То есть такое упрощение связано с переходом от комплексного ДПФ к действительному. Полученные алгоритмы называются *действительным прямым и действительным обратным БПФ*. Обработка ускоряется примерно на 30% по сравнению с обычными алгоритмами БПФ. Реализации действительных алгоритмов БПФ использованы в Программе 12.6 и 12.7.

### Программа 12.6

```

3000 'БПФ для действительных сигналов
3010 'Порядок ДПФ - N% и массив действительных компонент REX[ ] заполняются до
3020 'входа в подпрограмму. Содержимое массива и IMX[ ] игнорируется. Выходные
3030 'данные размещаются в REX[ ] и IMX[ ]. Индексация в массивах: 0...(N%-1).
3040 '
3050 NH% = N%/2-1           'Разделение чётных и нечётных отсчётов
3060 FOR I% = 0 TO NH%

```

```
3070 REX(I%) = REX(2*I%)
3080 IMX(I%) = REX(2*I%+1)
3090 NEXT I%
3100 '
3110 N% = N%/2           'Вычисление БПФ порядка N%/2
3120 GOSUB 1000          '(GOSUB 1000 - переход к Программе 12.3)
3130 N% = N%*2
3140 '
3150 NM1% = N%-1         'Разложение на чётные и нечётные элементы
3160 ND2% = N%/2          'в частотной области
3170 N4% = N%/4-1
3180 FOR I% = 1 TO N4%
3190 IM% = ND2%-I%
3200 IP2% = I%+ND2%
3210 IPM% = IM%+ND2%
3220 REX(IP2%) = (IMX(I%) + IMX(IM%))/2
3230 REX(IPM%) = REX(IP2%)
3240 IMX(IP2%) = -(REX(I%) - REX(IM%))/2
3250 IMX(IPM%) = -IMX(IP2%)
3260 REX(I%) = (REX(I%) + REX(IM%))/2
3270 REX(IM%) = REX(I%)
3280 IMX(I%) = (IMX(I%) - IMX(IM%))/2
3290 IMX(IM%) = -IMX(I%)
3300 NEXT I%
3310 REX(N%*3/4) = IMX(N%/4)
3320 REX(ND2%) = IMX(0)
3330 IMX(N%*3/4) = 0
3340 IMX(ND2%) = 0
3350 IMX(N%/4) = 0
3360 IMX(0) = 0
3370 '
3380 PI = 3.14159265      'Завершение последнего этапа БПФ
3390 L% = CINT(LOG(N%)/LOG(2))
3400 LE% = CINT(2^L%)
3410 LE2% = LE%/2
3420 UR = 1
3430 UI = 0
3440 SR = COS(PI/LE2%)
3450 SI = -SIN(PI/LE2%)
3460 FOR J% = 1 TO LE2%
3470 JM1% = J%-1
3480 FOR I% = JM1% TO NM1% STEP LE%
3490 IP% = I%+LE2%
3500 TR = REX[IP%]*UR - IMX[IP%]*UI
3510 TI = REX[IP%]*UI + IMX[IP%]*UR
3520 REX[IP%] = REX[I%]-TR
3530 IMX[IP%] = IMX[I%]-TI
3540 REX[I%] = REX[I%]+TR
3550 IMX[I%] = IMX[I%]+TI
3560 NEXT I%
3570 TR = UR
3580 UR = TR*SR - UI*SI
```

```
3590 UI = TR*SI + UI*SR
3600 NEXT J%
3610 RETURN
```

### Программа 12.7

```
4000 'ОБРАТНОЕ БПФ ДЛЯ ДЕЙСТВИТЕЛЬНЫХ СИГНАЛОВ
4010 'Перед входом в подпрограмму требуется указать порядок обратного ДПФ - N%
4020 'и заполнить элементы 0..N%/2 в массивах действительных и мнимых компонент
4030 'REX[ ] и IMX[ ], остальные элементы массивов игнорируются. Выходные
4040 'данные размещаются в массиве REX[ ], а в IMX[ ] все элементы нулевые.
4050 '
4060 '
4070 FOR K% = (N%/2+1) TO (N%-1)           'Введение симметрии в частотной области
4080 REX[K%] = REX[N%-K%]                   '(аналогично Программе 12.1)
4090 IMX[K%] = -IMX[N%-K%]
4100 NEXT K%
4110 '
4120 FOR K% = 0 TO N%-1                     'Сложение действительной и мнимой частей
4130 REX[K%] = REX[K%]+IMX[K%]
4140 NEXT K%
4150 '
4160 GOSUB 3000                            'Вычисление прямого ДПФ (Программа 12.6)
4170 '
4180 FOR I% = 0 TO N%-1                     'Сложение действительной и мнимой частей
4190 REX[I%] = (REX[I%]+IMX[I%])/N%      'и деление на N% во временной области
4200 IMX[I%] = 0
4210 NEXT I%
4220 '
4230 RETURN
```

Алгоритм действительного БПФ имеет два небольших недостатка. Во-первых, размер кода программы увеличивается примерно вдвое по сравнению с комплексным БПФ. Во-вторых, усложняется процедура отладки кода, так как при вычислении действительного БПФ отсутствует симметрия, о которой мы говорили выше. Дело в том, что равенство нулю всех отсчётов мнимой части сигнала, представленного во временной области, и симметрия в частотной области достигаются принудительно. Проверка алгоритмов действительного БПФ проводится путём сравнения полученных результатов с теми, которые достигаются при выполнении обычного комплексного БПФ.

Работа действительного БПФ иллюстрируется на Рис. 12.10 и 12.11. На Рис. 12.10 показана временная форма сигнала, действительная часть которого является прямоугольным импульсом, а мнимая равна нулю (**а** и **б**). Частотный спектр такого сигнала показан на (**в** и **г**). Как уже говорилось, действительная часть обладает чётной симметрией относительно отсчётов 0 и  $N/2$ , а мнимая — нечётной относительно тех же отсчётов.

Теперь обратимся к Рис. 12.11, на котором временная форма сигнала состоит из нулевой действительной и мнимой части, имеющей форму прямоугольного импульса. Виды симметрии в частотной области меняются местами: действительная часть приобретает нечётную симметрию, а мнимая — чётную. Данное обстоятельство будет подробнее рассмотрено в Главе 29, а пока примите на веру это свойство ДПФ.

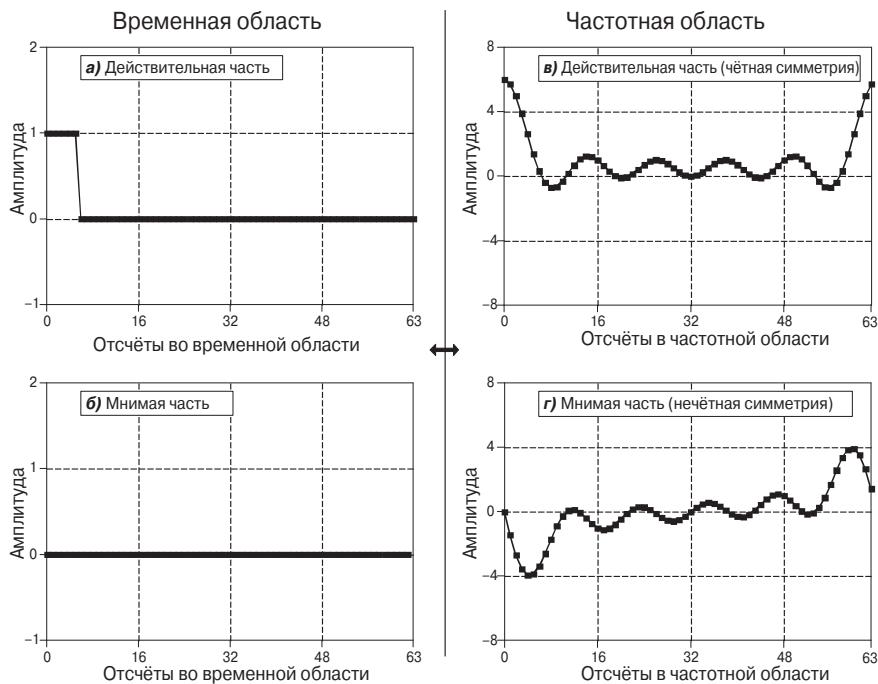


Рис. 12.10. Симметрия действительной части ДПФ.

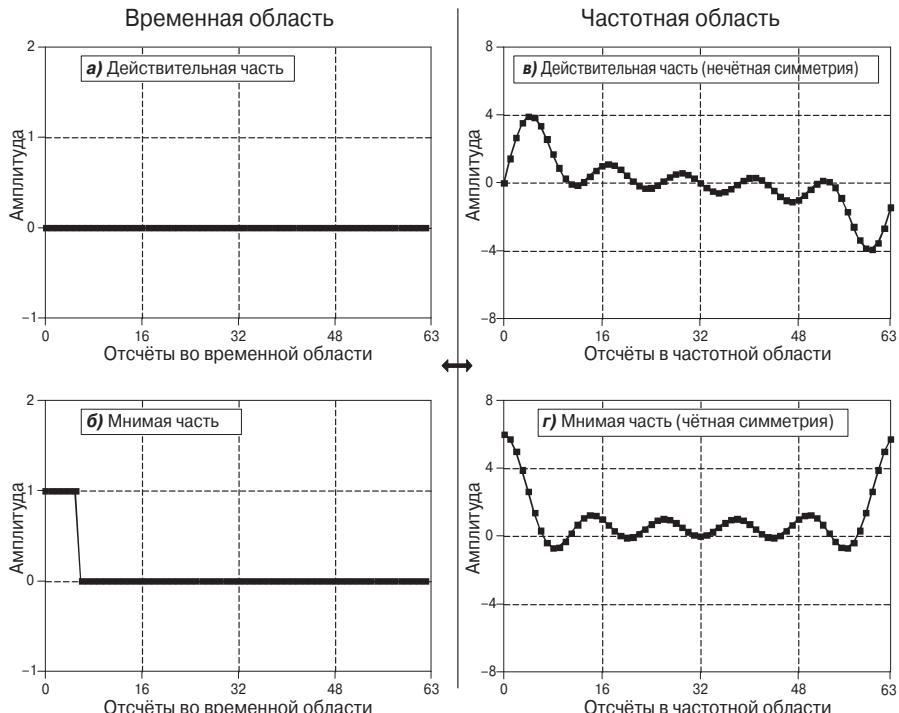


Рис. 12.11. Симметрия мнимой части ДПФ.

Что будет, если сигнал во временной области содержит обе составляющие — действительную и мнимую? Благодаря *свойству аддитивности* ДПФ, спектры этих составляющих складываются. А теперь вспомним ещё одно важное свойство: если спектр сигнала получен объединением сигналов с двумя видами симметрии, то его легко можно представить в виде двух соответствующих компонент. В Главе 5 для такого разделения применялись сигналы с чётной и нечётной симметрией. Следовательно, два действительных ДПФ можно выполнить с помощью одной процедуры БПФ, если один сигнал отождествить с действительной, а другой — с мнимой частью комплексного сигнала. После вычисления комплексного ДПФ (с помощью алгоритма БПФ) спектры разделяются в соответствии с разложением на основе сигналов с чётной и нечётной симметрией. Когда с помощью алгоритма БПФ требуется обработать много действительных сигналов, данный метод позволяет ускорить обработку примерно на 40%. Время обработки сокращается почти в 2 раза, потому что кроме БПФ приходится ещё выполнять декомпозицию. Описанный подход достаточно прост, но и он скрывает в себе несколько ловушек.

Следующий шаг связан с модификацией, позволяющей ускорить выполнение ДПФ для отдельного действительного сигнала. Это достаточно сложный вопрос, но мы всё же расскажем о том, как это делается. Отсчёты входного сигнала разбиваются на две группы.  $N/2$  чётных отсчётов образуют действительную часть временного сигнала, а  $N/2$  нечётных отсчётов — его мнимую часть. Затем выполняется БПФ размерности  $N/2$ , на которое затрачивается примерно в 2 раза меньше времени, чем на БПФ  $N$ -го порядка. Полученный результат в частотной области разделяется при помощи разложения на основе сигналов с чётной и нечётной симметрией. При этом образуются два спектра, каждый из которых соответствует одному из компонентов разложения во временной области. Далее рассчитанные спектры объединяются точно так же, как это делается на последнем уровне синтеза в обычном алгоритме БПФ, и формируют результирующий частотный спектр.

Завершая главу, заметим, что БПФ для цифровой обработки сигналов — это примерно то же самое, что транзистор для электроники. Этот базовый элемент предназначен для построения сложных схем, и поэтому каждый, кто его использует, должен хорошо знать его характеристики и принцип работы. При этом лишь немногие специалисты разбираются во всех особенностях его внутреннего функционирования.

# Глава 13

## АНАЛОГОВАЯ ОБРАБОТКА СИГНАЛОВ

*Аналоговая обработка сигналов* составляет альтернативу цифровой обработке. Развитие этих двух научно-технических направлений происходит параллельно, что объясняет родство используемых ими методов и сходство терминологии. Например, такие понятия, как «линейность», «свёртка», «спектральный анализ», обрели широкое распространение как в аналоговой обработке сигналов, так и в ЦОС.

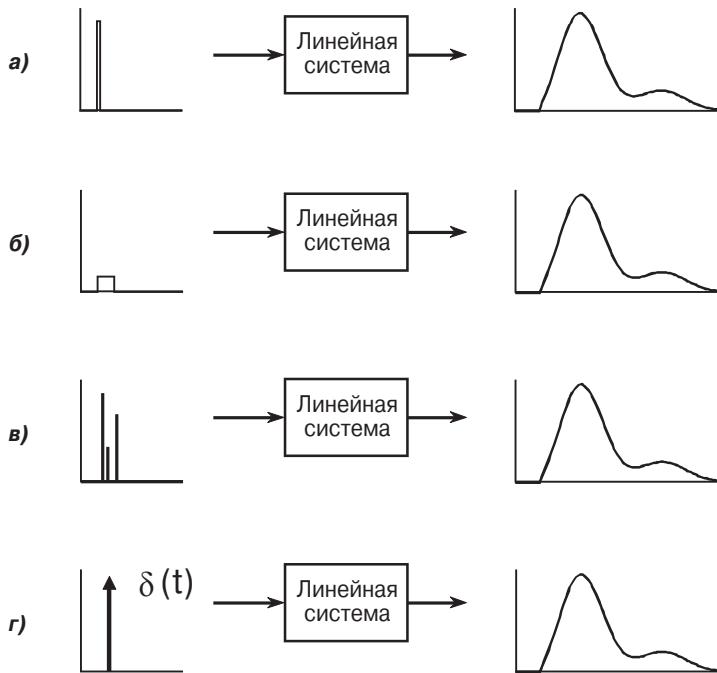
К сожалению, аналоговая форма представления сигналов не пригодна для их непосредственной обработки с помощью цифровых ЭВМ. Аналоговые сигналы являются непрерывными функциями, а модели аналоговых систем отражают правила преобразования одних непрерывных функций в другие. Подобно тому, как вся цифровая обработка строится вокруг написания компьютерных программ, аналоговая обработка сигналов основана на разнообразных математических преобразованиях, многие из которых были известны ещё за сотни лет до возникновения компьютеров.

### 13.1. Дельта-функция

*Аналоговые сигналы*, так же как и цифровые, могут быть представлены в виде суммы дельта-функций, сдвинутых вдоль оси времени и взятых с определёнными весовыми коэффициентами. Но в отличие от цифровой дельта-функции, которая представляет собой единичный импульс, аналоговая дельта-функция является физически не реализуемой абстракцией. Попробуем определить аналоговую дельта-функцию через её свойства.

Предположим, что имеется электрическая цепь, состоящая из пассивных линейных элементов: резисторов, конденсаторов и катушек индуктивности. Вход цепи подключим к генератору импульсов произвольной формы, а выход цепи — к осциллографу, который позволит наблюдать реакцию на каждый входной импульс. Постараемся выяснить, как форма выходного сигнала связана с формой входного импульса. Для упрощения задачи будем считать, что импульс на входе цепи значительно короче, чем реакция на её выходе. Пусть, для определённости, длительность реакции на выходе системы составляет миллисекунды, а длительность порождающего её входного импульса не превышает нескольких микросекунд.

После проведения большого числа экспериментов приходим к следующим трём выводам. Во-первых, форма входного импульса не влияет на форму выходного сигнала. Это отражено на Рис. 13.1, где различные по форме короткие входные импульсы производят совершенно одинаковые выходные реакции. Во-вторых, форма выходного сигнала целиком определяется характеристиками системы, т. е. номиналами резисторов, конденсаторов и индуктивностей, а также порядком их соединения в цепи. В-третьих, амплитуда выходного импульса пря-



**Рис. 13.1.** Аналоговая дельта-функция. Если сигнал на входе линейной системы является коротким импульсом по сравнению с сигналом на её выходе, то форма выходного сигнала зависит только от характеристики системы и не зависит от формы входного импульса. На (а...в) показаны примеры входных сигналов, представляющих собой короткие импульсы разной формы. Дельта-функция — это нормированный по мощности бесконечно короткий импульс, который отличен от нуля только при  $t = 0$  и имеет площадь, равную единице. Для обозначения дельта-функции используется математический символ  $\delta(t)$ , а для её графического изображения — вертикальная стрелка (г).

мо пропорциональна площади входного. То есть сигнал на выходе будет иметь одинаковую амплитуду для следующих прямоугольных импульсных входных воздействий: 1 В — на интервале 1 мкс, 10 В — на интервале 0.1 мкс, 1000 В — на интервале 1 нс и т. д. Эти соотношения справедливы и в случае отрицательных амплитуд. Например, для комбинированного сигнала, состоящего из близко расположенных двух положительных импульсов с амплитудой 2 В длительностью 2 мкс и одного отрицательного импульса с амплитудой 1 В и длительностью 4 мкс, суммарная площадь оказывается равной нулю. Реакция системы на такой входной сигнал нулевая, т. е. подключённый к выходу осциллограф никакого импульса не зарегистрирует.

Эти три свойства выполняются всегда, когда длительность входного сигнала достаточно мала по сравнению с протяжённостью реакции на выходе. Можно ли считать импульс коротким, зависит от конкретного приложения. Например, для микроволнового передатчика с временем реакции в несколько наносекунд коротким следует считать импульс, длительность которого измеряется пикосекундами. В то же время извержение вулкана, которое тянется годами, — это самый настоящий короткий импульс по отношению к геологическим процессам, происходящим после него в течение тысячелетий.

Для математики важно иметь универсальную функцию, чтобы не ограничиваться частными приближёнными решениями. Такая функция должна представлять собой бесконечно короткий импульс конечной мощности. По определению, аналоговая дельта-функция (или просто дельта-функция) представляет собой импульс, удовлетворяющий следующим трём свойствам: 1) длительность импульса бесконечно мала; 2) функция отлична от нуля только в нулевой момент времени; 3) площадь, ограниченная графиком функции, равна единице.

Так как дельта-функция отлична от нуля на бесконечно малом интервале и вместе с тем имеет конечную площадь, то амплитуда её бесконечно велика. Этот факт не должен смущать читателя, потому что дельта-функция — это всего лишь математическая абстракция. В реальности сигналы всегда имеют конечную длительность и амплитуду.

Непрерывную дельта-функцию принято обозначать символом  $\delta(t)$ . Выходная реакция системы на входное воздействие в виде дельта-функции называется *импульсной характеристикой* и, как правило, обозначается символом  $h(t)$ . Заметим, что круглые скобки используются для того, чтобы отличить аналоговые сигналы от цифровых. Для последних применяются квадратные скобки. Такая система обозначений используется как в этой, так и во многих других книгах по ЦОС, но она не универсальна. На графике дельта-функция изображается вертикальной стрелкой (**Рис. 13.1 $\sigma$** ), причём длина стрелки соответствует площади импульса.

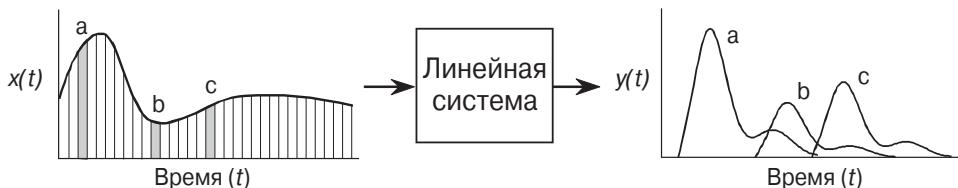
Для лучшего понимания смысла дельта-функции приведём ещё один пример. Найдите на ночном небе планету и звезду, например, Марс и Сириус. Их яркость и размер кажутся почти одинаковыми. Однако такое видимое сходство обманчиво. Марс имеет диаметр, равный приблизительно 6000 км и расположенный в 60 млн км от Земли. Сириус примерно в 300 раз больше Марса и более чем в 1 млн раз дальше его. Судя по этим показателям, видимый размер Марса должен быть в 3000 раз больше, чем размер Сириуса. Почему же они выглядят одинаковыми?

Это связано с тем, что видимые размеры обоих небесных тел настолько малы, что по отношению к зрительной системе человека свет, приходящий от них, играет роль дельта-импульса. Следовательно, мы воспринимаем импульсную характеристику нашего глаза, а не действительное изображение звезды или планеты. Наблюдение этих двух объектов в обычный любительский телескоп подтверждает такое предположение: Марс сразу превращается в расплывчатый диск, а Сириус по-прежнему остаётся яркой точкой. Именно по этой причине звёзды мерцают, а планеты нет. Изображение звезды очень мало и может на короткое время заслоняться частицами или преломляться из-за турбулентностей атмосферы, в то время как более крупное изображение планеты слабо зависит от подобного рода искажений.

## 13.2. Операция свёртки

Так же как и в случае с дискретными сигналами, суть операции *свёртки* для аналоговых сигналов можно объяснить, рассматривая систему как относительно её входа, так и относительно её выхода. Описание процесса свёртки со стороны входа системы позволяет лучше понять, как работает свёртка. В свою очередь описание со стороны выхода позволяет получить более удобные для практических целей математические выражения. Эти выражения оказываются похожими на те, которые приводились в Главе 6 для цифровых сигналов.

Описание свёртки со стороны входа системы поясняется на Рис. 13.2. Сигнал  $x(t)$  поступает на вход системы с импульсной характеристикикой  $h(t)$ , порождая на выходе системы сигнал  $y(t)$ . Такому процессу соответствует уже известная вам математическая форма записи:  $y(t) = x(t) * h(t)$ . Разобьём входной сигнал на короткие фрагменты, каждый из которых представляет собой дельта-импульс. В результате входной сигнал оказывается представленным суммой бесконечно большого числа дельта-функций, взятых с некоторой амплитудой и смешённых по оси времени на необходимый интервал. Каждый дельта-импульс создаёт на выходе реакцию, совпадающую по форме с импульсной характеристикикой линейной системы, но промасштабированную по амплитуде и задержанную по времени. Результирующая реакция складывается из совокупности всех элементарных реакций.



**Рис. 13.2.** Описание свёртки со стороны входа. Входной сигнал  $x(t)$  разбивается на короткие фрагменты, каждый из которых играет роль дельта-импульса. Выходной сигнал  $y(t)$  представляет собой сумму умноженных на весовые коэффициенты и смешённых вдоль временной оси импульсных характеристикик. На рисунке показано, как отдельные фрагменты входного сигнала влияют на формирование общего результирующего сигнала системы.

Данные рассуждения справедливы лишь в том случае, когда длительность фрагментов, на которые разбит входной сигнал, значительно меньше времени реакции системы. В математике принято рассматривать предельную ситуацию, при которой фрагменты считаются бесконечно короткими, и задача сводится к аналитическим преобразованиям. Описание свёртки со стороны входа позволяет понять, каким образом каждое мгновенное значение (бесконечно малый фрагмент) входного сигнала участвует в формировании сигнала на выходе системы.

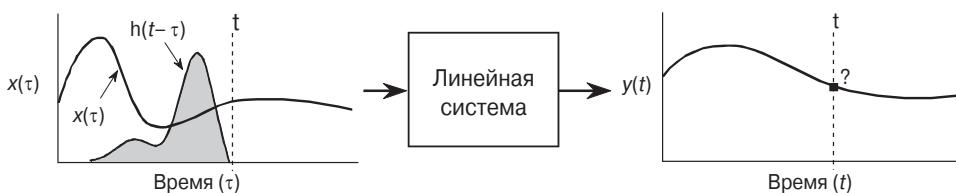
Описание свёртки со стороны выхода, напротив, связывает каждое мгновенное значение выходного сигнала со множеством значений входного. То есть, как и в случае с цифровыми сигналами, каждое мгновенное значение аналогового выходного сигнала определяется множеством мгновенных значений входного воздействия, сформированных импульсной характеристикикой линейной системы. Но если в случае цифровых сигналов отсчёты перемножаются и затем суммируются, то для аналоговых сигналов выполняются операции умножения и интегрирования:

$$y(t) = \int_{-\infty}^{+\infty} x(\tau) h(t - \tau) d\tau. \quad (13.1)$$

Интеграл свёртки. Это выражение раскрывает смысл записи  $y(t) = x(t) * h(t)$ .

Данное уравнение называется *интегралом свёртки* и представляет собой аналог выражения (6.1), применяемого для дискретных сигналов. Рис. 13.3 поясняет смысл выражения (13.1). При описании операции свёртки со стороны выхода

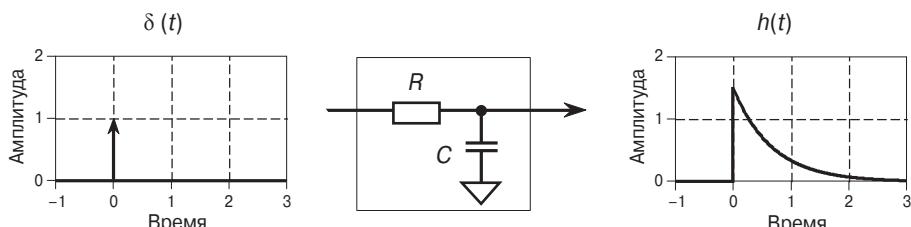
возникает следующая задача: необходимо найти выражение, связывающее мгновенное значение выходного сигнала линейной аналоговой системы, взятое в произвольный момент времени  $t$ , с входным сигналом системы. Прежде всего введём независимую переменную времени  $\tau$ , поскольку  $t$  обозначает теперь фиксированный момент времени и связана с выходным сигналом. Теперь перевернём импульсную характеристику слева направо. Это достигается простым изменением знака независимой переменной:  $h(\tau)$  превращается в  $h(-\tau)$ . Далее смещаем перевёрнутую импульсную характеристику на  $t$  и умножаем на неё входной сигнал:  $x(\tau)h(t - \tau)$ . Мгновенное значение выходного сигнала находится путём интегрирования по  $\tau$  на интервале  $-\infty \dots +\infty$  (13.1).



**Рис. 13.3.** Описание свёртки со стороны выхода системы. Каждое значение выходного сигнала формируется под воздействием целого фрагмента входного сигнала. На этом рисунке иллюстрируется вычисление значения выходного сигнала в момент времени  $t$ . Входной сигнал  $x(\tau)$  умножается на перевёрнутую слева направо и задержанную на  $t$  импульсную характеристику системы —  $h(t - \tau)$ . Выходной сигнал  $y(t)$  получается в результате интегрирования такого произведения.

Если вам трудно понять рассмотренный здесь принцип, то прежде ещё раз повторите аналогичные рассуждения, представленные в Главе 6 для случая цифровых сигналов. **Рис. 13.3** и **6.8** соответствуют двум разным подходам к описанию одного и того же механизма свёртки. Единственное различие между ними состоит в том, что сумма заменяется интегралом. Замена суммы на интеграл приводит всего лишь к более общей форме выражения механизма свёртки.

Как пользоваться интегралом свёртки, поясним на конкретных практических примерах. На **Рис. 13.4** представлена простейшая линейная аналоговая система: пассивный низкочастотный фильтр, состоящий из одного резистора и одного



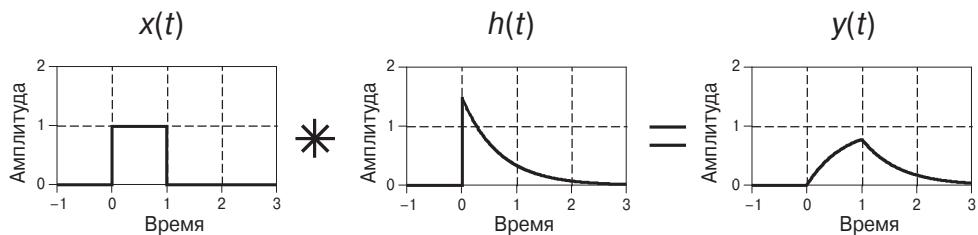
**Рис. 13.4.** Пример линейной аналоговой системы. Данная электрическая цепь представляет собой простейший низкочастотный фильтр, состоящий из одного резистора и одного конденсатора. Импульсной характеристикой этой системы является импульс, затухающий по экспоненте.

конденсатора. Как показано на рисунке, дельта-импульс, поступающий на вход системы, создает на её выходе сигнал, изменяющийся скачком до некоторого значения, а затем экспоненциально спадающий до нуля. То есть импульсная характеристика электрической цепи представляет собой затухающий по экспоненте импульс. Математически такая функция может быть задана двумя простыми выражениями на двух интервалах:

$$\begin{aligned} h(t) &= 0 && \text{при } t < 0, \\ h(t) &= \alpha \exp(-\alpha t) && \text{при } t \geq 0, \end{aligned}$$

где  $\alpha = 1/RC$ ,  $R$  измеряется в омах,  $C$  — в фарадах,  $t$  — в секундах. Импульсная характеристика аналоговой цепи (как и цифровой цепи) несёт в себе исчерпывающую информацию о свойствах линейной системы, позволяя определить реакцию на входное воздействие любой формы. **Рис. 13.5** показывает, что именно происходит при поступлении на вход  $RC$ -цепи прямоугольного импульса вида

$$\begin{aligned} x(t) &= 1 && \text{при } 0 \leq t \leq 1, \\ x(t) &= 0 && \text{при } t < 0 \text{ и при } t > 1. \end{aligned}$$



**Рис. 13.5.** Пример свёртки двух аналоговых сигналов. Этот рисунок иллюстрирует процесс обработки входного импульса низкочастотным фильтром, образованным  $RC$ -цепью (**Рис. 13.4**). Выходной сигнал является результатом свёртки входного импульса и импульсной характеристики системы.

Так как и входной сигнал, и импульсная характеристика заданы аналитически (в виде математических выражений), то выходной сигнал  $y(t)$  может быть вычислен с помощью интеграла свёртки (13.1). Но расчёт усложняется тем, что оба сигнала заданы на нескольких интервалах разными выражениями. Такой случай в аналоговой обработке сигналов широко распространён. Рассмотрим несколько характерных вариантов взаимного расположения сигналов. Из **Рис. 13.6а** ясно, что для всех  $t < 0$  сигналы не перекрываются. Это означает, что произведение сигналов равно нулю при любых  $t$ , т. е. результатирующий выходной сигнал:

$$y(t) = 0 \quad \text{при } t < 0.$$

Вариант, показанный на **Рис. 13.6б**, соответствует значениям  $t$  из диапазона 0...1. В этом случае сигналы перекрываются лишь частично, и произведение их отлично от нуля только на отрезке значений  $t$  от 0 до  $t$ . Поэтому при вычислении интеграла достаточно ограничиться этим отрезком.

$$y(t) = \int_{-\infty}^{\infty} x(\tau) h(t - \tau) d\tau,$$

$$y(t) = \int_0^t 1 \cdot \alpha e^{-\alpha(t-\tau)} d\tau \quad (\text{подстановка сигналов}),$$

$$y(t) = e^{-\alpha t} [e^{\alpha t}] \Big|_0^t \quad (\text{переход к первообразной}),$$

$$y(t) = e^{-\alpha t} [e^{\alpha t} - 1] \quad (\text{упрощение}),$$

$$y(t) = 1 - e^{-\alpha t} \quad \text{при } 0 \leq t \leq 1.$$

**Рис. 13.6в** соответствует третьему возможному варианту в расположении сигналов, когда  $t > 1$ . Функции перекрываются на всём диапазоне  $\tau$  от 0 до 1. Следовательно, подынтегральное выражение остаётся прежним, а пределы и интегрирования — 0 и 1:

$$y(t) = \int_0^1 1 \cdot \alpha e^{-\alpha(t-\tau)} d\tau,$$

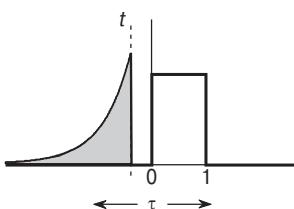
$$y(t) = e^{-\alpha t} [e^{\alpha t}] \Big|_0^1,$$

$$y(t) = e^{-\alpha t} [e^{\alpha t} - 1] \quad \text{при } t > 1.$$

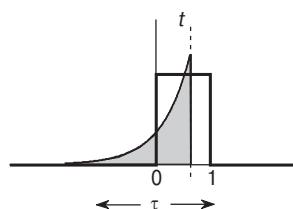
Форма полученного в процессе вычислений выходного сигнала хорошо согласуется с практическими представлениями, известными из электротехники:

1) выходной сигнал остаётся нулевым, пока сигнал на входе цепи равен нулю:  $y(t) = 0$  при  $t < 0$ ; 2) как только на вход системы поступает прямоугольный импульс, сигнал на выходе  $RC$ -цепи начинает нарастать по закону  $y(t) = 1 - e^{-\alpha t}$ ; 3) сразу после окончания действия входного импульса выходной сигнал начинает спадать до нуля по экспоненте:  $y(t) = ke^{-\alpha t}$  (где  $k = e^{\alpha} - 1$  — максимальное напряжение, до которого зарядился конденсатор).

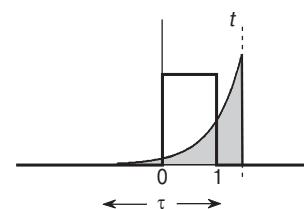
**a)** Нет перекрытия  
( $t < 0$ )



**б)** Частичное перекрытие  
( $0 \leq t \leq 1$ )



**в)** Полное перекрытие  
( $t > 1$ )



**Рис. 13.6.** Вычисление свёртки по частям. Так как аналоговые сигналы на разных интервалах могут быть заданы различными функциями, то вычисление свёртки для этих интервалов выполняется раздельно. В этом примере можно выделить три частных случая перекрытия входного сигнала и перевёрнутой импульсной характеристики: **а)** нет перекрытия; **б)** частичное перекрытие; **в)** полное перекрытие.

Рассмотренный выше подход применим для входных сигналов произвольной формы. Только преобразования для них могут получаться намного более сложными, чем в нашем примере. Попытка вычисления интеграла свёртки «в лоб» очень часто оказывается обречённой на неудачу. Во многих случаях помогает метод предварительного разбиения сложных сигналов на более простые компоненты. Свёртка для них выполняется независимо, а результат решения поставленной задачи получается путём соединения результатов, найденных для отдельных компонент. Здесь играет важную роль свойство линейности систем.

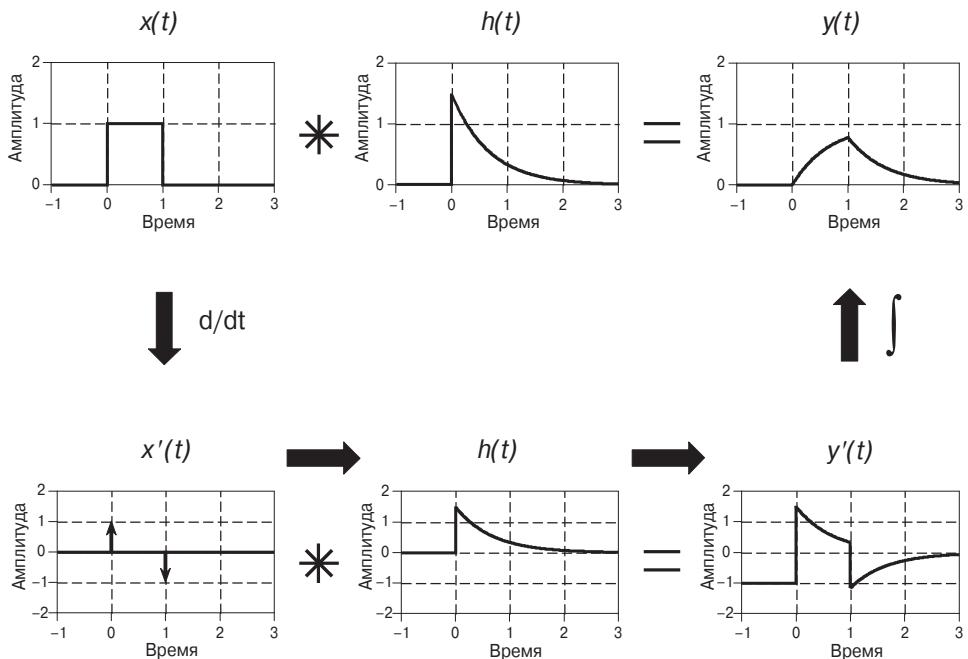
Возможен также принципиально иной подход, суть которого поясняется на Рис. 13.7. Исходный сигнал некоторым образом преобразуется (преобразование должно быть линейным), затем вычисляется свёртка, а в конце выполняется обратное преобразование. В приведённом примере в качестве прямого преобразования выбрано дифференцирование, а в качестве обратного — интегрирование. Производной от прямоугольного импульса единичной амплитуды является функция, состоящая из двух дельта-импульсов: площадь первого равна единице, а второго — минус единице. Чтобы доказать это, рассмотрим обратную операцию — вычисление интеграла от двух таких дельта-импульсов. Интегрирование первого импульса даст сигнал, мгновенно нарастающий от нуля до единицы, т. е. единичную ступенчатую функцию. Интегрирование второго дельта-импульса, напротив, обеспечит спад сигнала до нуля. В результате будет сформирован задний фронт прямоугольного импульса.

Взятие производной упрощает задачу благодаря тому, что свёртка для дельта-импульсов выполняется значительно проще. Каждый из дельта-импульсов в составе  $x'(t)$  приводит к формированию в сигнале  $y'(t)$  импульсной характеристики, умноженной на соответствующий весовой коэффициент и задержанной по времени:  $y'(t) = h(t) - h(t - 1)$ . Выходной сигнал  $y(t)$  зависит от конкретного вида импульсной характеристики  $h(t)$  и находится путём интегрирования этого выражения.

Маленькая хитрость в таком методе кроется в том, что в момент взятия производной теряется информация о постоянной составляющей входного сигнала. В результате выходной сигнал может быть вычислен с ошибкой по постоянной составляющей. Не существует универсального подхода, позволяющего устраниТЬ такую ошибку, но она все же может быть выявлена при дополнительном анализе решаемой задачи. Например, на Рис. 13.7 ошибка по постоянной составляющей отсутствует, так как выходной сигнал стремится к верноустановившемуся значению при  $t \rightarrow \infty$ . Если в какой-нибудь задаче такая ошибка всё-таки обнаружена, то её легко можно компенсировать, добавив постоянную величину к полученному выходному сигналу.

Описанный метод можно применять и к таким сигналам, которые приводятся к последовательности дельта-функций лишь после многократного дифференцирования. Такие сигналы называются *кусочно-полиномиальными*. Многократному выполнению операции дифференцирования соответствует обратная операция — многократное интегрирование. При многократном интегрировании корректировка постоянной составляющей должна выполняться на каждом очередном этапе интегрирования.

Прежде чем вычислить свёртку аналитически и преодолеть все возникающие вследствие этого трудности, подумайте, нужно ли вам точное математическое выражение или достаточно графически изобразить приблизительную форму выходного сигнала. Если достаточно одного графического изображения, то лучше обратиться к численным методам, т. е. представить сигнал последовательностью



**Рис. 13.7.** Метод вычисления свёртки, основанный на дифференцировании сигналов. Задачу вычисления свёртки часто удаётся упростить, воспользовавшись свойством линейности системы. В приведённом примере упрощение достигается за счёт дифференцирования входного сигнала. После вычисления свёртки производной от входного сигнала и импульсной характеристики системы выполняется обратное преобразование — интегрирование.

дискретных отсчётов и рассчитать реакцию системы при помощи компьютерной программы. Хотя, с точки зрения математики, численные методы всегда вносят некоторую ошибку в преобразование, зато их использование намного ускоряет поиск результата.

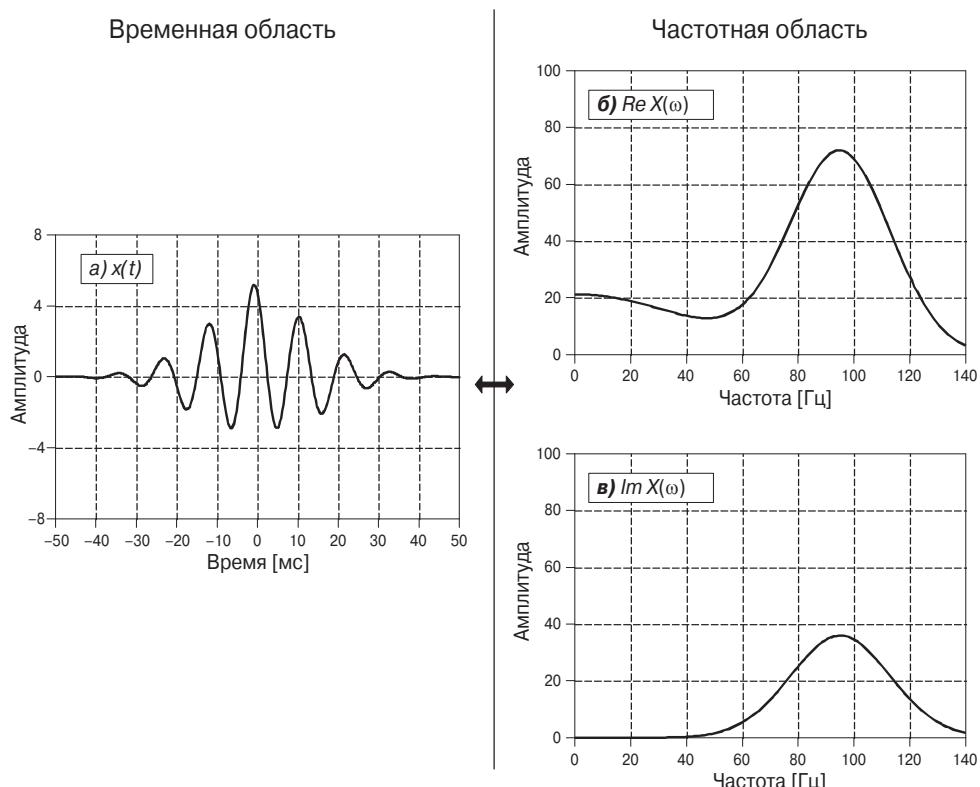
### 13.3. Интеграл Фурье

Преобразование Фурье определяется по-разному для периодических и непериодических функций. В случае периодических функций применяются так называемые ряды Фурье. Ряды Фурье мы рассмотрим в следующем разделе этой главы, а сейчас разберёмся с преобразованием Фурье для непериодических функций — *интегралом Фурье*. Пока ограничимся тригонометрической интерпретацией интегрального преобразования Фурье, как мы это сделали ранее для цифровых сигналов. Более мощное средство — комплексное преобразование Фурье — будет рассмотрено в Главе 31.

На **Рис. 13.8** приведён пример непрерывного апериодического сигнала и его частотного спектра. Во временной области сигнал определён на интервале  $-\infty \dots +\infty$ . Частотный спектр сигнала построен в прямоугольной системе координат (действительная и мнимая части), однако возможен переход к полярным координатам (модулю и фазе). Так же как в случае цифровых сигналов, *уравнение синтеза* (обратное преобразование Фурье) описывает переход от частотной формы представления сигнала к его временной форме:

$$x(t) = \frac{1}{\pi} \int_0^{+\infty} Re X(\omega) \cos(\omega t) - Im X(\omega) \sin(\omega t) d\omega. \quad (13.2)$$

Уравнение синтеза. В этом выражении  $x(t)$  — сигнал во временной области,  $Re X(\omega)$  и  $Im X(\omega)$  — действительная и мнимая части спектра.



**Рис. 13.8.** Преобразование Фурье. Сигнал во временной области определён на интервале  $-\infty \dots +\infty$ . В частотной области сигнал представлен действительной  $Re X(\omega)$  и мнимой  $Im X(\omega)$  частями, каждая из которых определена в диапазоне частот  $0 \dots +\infty$ . Частота на графиках указана в герцах. Для перехода к круговой частоте нужно линейную частоту умножить на  $2\pi$ .

Выражение (13.2) означает, что сигнал во временной области образуется в результате суммирования бесконечно большого числа (т. е. интегрирования) косинусных и синусных колебаний, умноженных на постоянные коэффициенты. В частотной области действительная часть представляет собой бесконечное множество коэффициентов, на которые умножаются косинусоиды, а мнимая часть — бесконечное множество коэффициентов, на которые умножаются синусоиды. Как и в случае цифровых сигналов, синусные колебания входят в уравнение синтеза с отрицательным знаком. Пока выбор отрицательного знака не имеет принципиального значения. Отрицательный знак выбран только для того, чтобы сделать запись более похожей на комплексное преобразование Фурье. Комплексную математику мы начнём использовать лишь в Главе 29. В других книгах в записи

обратного преобразования Фурье вам может встретиться как отрицательный знак, так и положительный. Кроме того, заметим, что частота в представленном выражении обозначена символом  $\omega$  (строчная буква греческого алфавита «омега»). Этот символ является общепринятым для *круговой частоты*, которая изменяется в радианах в секунду:  $\omega = 2\pi f$ , где  $f$  — частота в герцах. Круговая частота широко используется в математических выражениях, описывающих преобразования сигналов, поскольку позволяет упростить запись и уменьшить количество символов, присутствующих в ней.

*Уравнение анализа* (прямое преобразование Фурье) для аналоговых сигналов очень похоже на такое же уравнение для цифровых сигналов. Основное отличие состоит в том, что для вычисления корреляции (взаимосвязи) входного сигнала с набором базисных функций (синусов и косинусов) в нём используется интеграл:

$$\begin{aligned} \operatorname{Re} X(\omega) &= \int_{-\infty}^{+\infty} x(t) \cos(\omega t) dt, \\ \operatorname{Im} X(\omega) &= - \int_{-\infty}^{+\infty} x(t) \sin(\omega t) dt. \end{aligned} \quad (13.3)$$

Уравнение анализа.  $\operatorname{Re} X(\omega)$  и  $\operatorname{Im} X(\omega)$  — действительная и мнимая части спектра,  $x(t)$  — анализируемый сигнал как функция времени.

Примером практического использования прямого преобразования Фурье является вычисление частотной характеристики низкочастотного *RC*-фильтра по его импульсной характеристике. Пусть выбран фильтр с той же импульсной характеристикой, что и прежде (**Рис. 13.4**):

$$\begin{aligned} h(t) &= 0 && \text{для } t < 0, \\ h(t) &= \alpha e^{-\alpha t} && \text{для } t \geq 0. \end{aligned}$$

Подставляя импульсную характеристику в уравнение прямого преобразования Фурье, находим частотную характеристику системы. Сначала найдём действительную часть.

Используя этот же подход, вычислим мнимую часть частотной характеристики:

$$\operatorname{Re} H(\omega) = \int_{-\infty}^{+\infty} h(t) \cos(\omega t) dt \quad (\text{начинаем с выражения общего вида}),$$

$$\operatorname{Re} H(\omega) = \int_0^{+\infty} \alpha e^{-\alpha t} \cos(\omega t) dt \quad (\text{подставляем импульсную характеристику}),$$

$$\operatorname{Re} H(\omega) = \frac{\alpha e^{-\alpha t}}{\alpha^2 + \omega^2} [-\alpha \cos(\omega t) + \omega \sin(\omega t)] \Big|_0^{+\infty} \quad (\text{подставляем пределы интегрирования}),$$

$$\operatorname{Re} H(\omega) = \frac{\alpha^2}{\alpha^2 + \omega^2} .$$

Теперь тем же способом находим мнимую часть частотной характеристики:

$$\operatorname{Im} H(\omega) = \frac{-\omega\alpha}{\alpha^2 + \omega^2}.$$

Как и в случае цифровых сигналов, представление результата преобразования Фурье в прямоугольных координатах удобно для математических операций, но не обеспечивает той наглядности восприятия человеком, как представление частотной характеристики в полярных координатах:

$$\operatorname{Mag} H(\omega) = [\operatorname{Re} H(\omega)^2 + \operatorname{Im} H(\omega)^2]^{\frac{1}{2}},$$

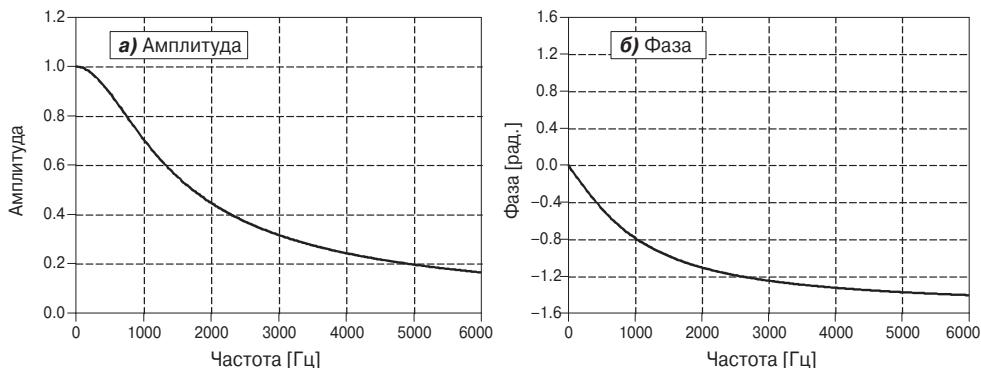
$$\operatorname{Phase} H(\omega) = \operatorname{arctg} [\operatorname{Im} H(\omega) / \operatorname{Re} H(\omega)].$$

Выполнив алгебраические преобразования, получим частотные характеристики низкочастотного *RC*-фильтра в полярной форме записи:

$$\operatorname{Mag} H(\omega) = \alpha / [\alpha^2 + \omega^2]^{\frac{1}{2}},$$

$$\operatorname{Phase} H(\omega) = \operatorname{arctg} [-\omega / \alpha].$$

Амплитудно-частотная и фазо-частотная характеристики для фильтра с частотой среза 1000 Гц ( $\omega = 2\pi \times 1000$  рад/с) изображены на **Рис. 13.9**.



**Рис. 13.9.** Амплитудно-частотная и фазо-частотная характеристики низкочастотного *RC*-фильтра. Частотные характеристики получены в результате применения преобразования Фурье к импульсной характеристике фильтра и последующего перехода к полярным координатам.

## 13.4. Ряд Фурье

Кроме интеграла Фурье в гармоническом анализе аналоговых сигналов используется ряд Фурье. Ряд Фурье применяют только к периодическим сигналам. Несколько примеров таких сигналов рассмотрено на **Рис. 13.10**. В Главе 11 уже упоминалось, что периодический во времени сигнал характеризуется дискретным спектром, состоящим из ряда гармоник. Например, если частота повторения сигнала равна 1000 Гц (период 1 мс), то первая гармоника спектра располагается на частоте 1000 Гц, вторая — на частоте 2000 Гц, третья — на частоте 3000 Гц и т. д. Первая гармоника, т. е. частота, с которой сигнал повторяется во времени, назы-

вается *основной частотой*. В результате понятие спектра периодического сигнала может быть введено двумя разными способами: 1) спектр определён на всём множестве действительных чисел, но отличен от нуля только на отдельных частотах — частотах его гармоник; 2) спектр — дискретная функция частоты, область определения которой образована множеством точек, совпадающих с частотами гармоник сигнала. В любом случае на временную форму сигнала оказывают влияние только дискретные отсчёты спектра.

Уравнение синтеза, в основе которого лежит ряд Фурье, позволяет представить периодический сигнал с частотой  $f$  в виде суммы взятых с определёнными весовыми коэффициентами синусоид и косинусоид с частотами  $f, 2f, 3f, 4f$  и т. д. Амплитуды косинусоид обозначаются символами  $a_1, a_2, a_3, a_4$  и т. д., тогда как амплитуды синусоид — символами  $b_1, b_2, b_3, b_4$  и т. д. Таким образом, коэффициенты  $a$  и  $b$  — это действительные и мнимые компоненты спектра. Коэффициент  $a_0$  позволяет учесть постоянную составляющую, которую можно рассматривать как амплитуду косинусоиды нулевой частоты. Иногда коэффициент  $a_0$  объединяют с остальными коэффициентами  $a$ , но чаще всего выносят за знак суммы и записывают как отдельное слагаемое. Коэффициент  $b_0$  отсутствует в уравнении синтеза, так как синус нулевой частоты тождественно равен нулю. Уравнение синтеза имеет следующий вид:

$$x(t) = a_0 + \sum_{n=1}^{\infty} a_n \cos(2\pi ftn) - \sum_{n=1}^{\infty} b_n \sin(2\pi ftn). \quad (13.4)$$

Уравнение синтеза, в основе которого лежит ряд Фурье. Любой периодический сигнал  $x(t)$  может быть представлен совокупностью синусоидальных и косинусоидальных колебаний с частотами, кратными основной частоте сигнала  $f$ . Коэффициенты ряда Фурье  $a_n$  и  $b_n$  — это амплитуды гармоник спектра.

При записи уравнений *анализа* обычно используют период сигнала  $T$ , а не частоту  $f$  ( $f = 1/T$ ). Так как сигнал является периодической функцией, то интеграл корреляции можно определить на интервале, равном одному периоду:  $-T/2 \dots T/2$ , или  $0 \dots T$ , или  $-T \dots 0$  и т. д. Несмотря на различие в пределах интегрирования, конечный результат получается всегда одинаковым. Уравнения анализа позволяют найти коэффициенты ряда Фурье:

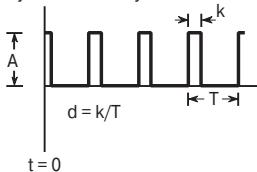
$$\begin{aligned} a_0 &= \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) dt, \quad a_n = \frac{2}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) \cos\left(\frac{2\pi tn}{T}\right) dt, \\ b_n &= \frac{2}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) \sin\left(\frac{2\pi tn}{T}\right) dt. \end{aligned} \quad (13.5)$$

Уравнения анализа позволяют найти коэффициенты ряда Фурье. В этих уравнениях  $x(t)$  — сигнал во временной области,  $a_0$  — постоянная составляющая,  $a_n$  — амплитуды косинусоидальных гармоник,  $b_n$  — амплитуды синусоидальных гармоник,  $T$  — период сигнала (величина, обратная основной частоте).

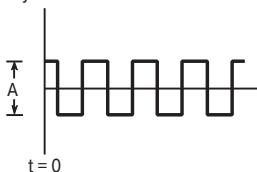
## Временная область

## а) Последовательность

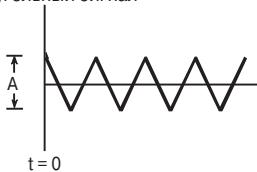
прямоугольных импульсов



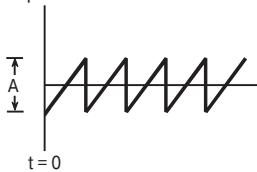
## б) Прямоугольный сигнал



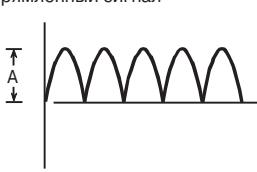
## в) Треугольный сигнал



## г) Пилообразный сигнал



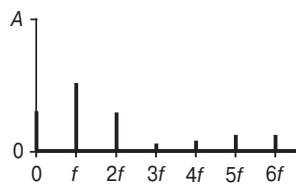
## д) Выпрямленный сигнал



## е) Косинусоидальное колебание



## Частотная область

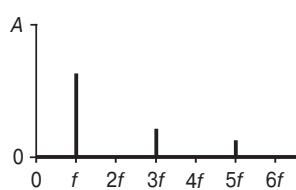


$$a_0 = A d$$

$$a_n = \frac{2A}{n\pi} \sin(n\pi d)$$

$$b_n = 0$$

( $d = 0.27$  в этом примере)

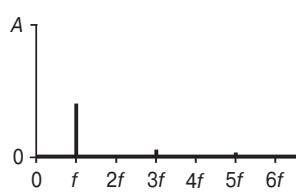


$$a_0 = 0$$

$$a_n = \frac{2A}{n\pi} \sin\left(\frac{n\pi}{2}\right)$$

$$b_n = 0$$

(все чётные гармоники равны нулю)

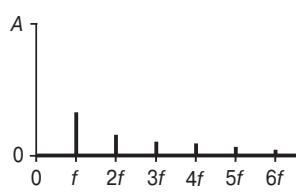


$$a_0 = 0$$

$$a_n = \frac{4A}{(n\pi)^2}$$

$$b_n = 0$$

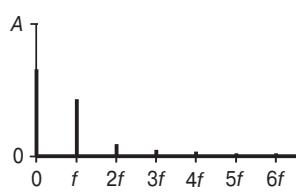
(все чётные гармоники равны нулю)



$$a_0 = 0$$

$$a_n = 0$$

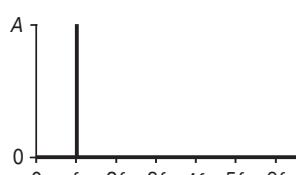
$$b_n = \frac{A}{n\pi}$$



$$a_0 = 2A/\pi$$

$$a_n = \frac{-4A}{\pi(4n^2-1)}$$

$$b_n = 0$$



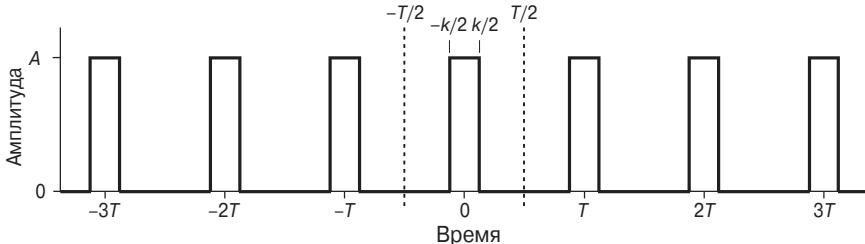
$$a_1 = A$$

(все другие коэффициенты равны нулю)

**Рис. 13.10.** Примеры разложения сигналов в ряд Фурье. Рассмотрены шесть наиболее распространённых в технике сигналов, их спектры и уравнения для расчёта коэффициентов ряда Фурье.

Разложим в ряд Фурье последовательность прямоугольных импульсов, изображённых на Рис. 13.11. Ограничимся интервалом  $-T/2 \dots T/2$ , соответствующим одному периоду сигнала:

$$\begin{aligned} x(t) &= A \text{ при } -k/2 \leq t \leq k/2, \\ x(t) &= 0 \text{ вне этого интервала.} \end{aligned}$$



**Рис. 13.11.** Пример разложения функции в ряд Фурье. Рассмотрена последовательность импульсов со скважностью  $d = k/T$ . Для нахождения коэффициентов ряда Фурье достаточно вычислять корреляцию с синусами и косинусами разной частоты на интервале одного периода сигнала. В этом примере выбран интервал  $-T/2 \dots T/2$ .

Скважностью последовательности прямоугольных импульсов называется отношение ширины импульсов к периоду их повторения:  $d = k/T$ . Коэффициенты ряда Фурье могут быть найдены с помощью (13.5). Сначала рассчитаем постоянную составляющую  $a_0$ :

$$a_0 = \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) dt \quad (\text{запишем выражение в общем виде}),$$

$$a_0 = \frac{1}{T} \int_{-\frac{k}{2}}^{\frac{k}{2}} A dt \quad (\text{перейдём к частному случаю}),$$

$$a_0 = \frac{Ak}{T} \quad (\text{вычислим интеграл}),$$

$$a_0 = Ad \quad (\text{упростим выражение}).$$

Следовательно, постоянная составляющая — это среднее значение сигнала, что понятно уже при первом взгляде на Рис. 13.11. Коэффициенты  $a$  получим аналогичным образом:

$$a_n = \frac{2}{T} \int_{-T/2}^{T/2} x(t) \cos\left(\frac{2\pi tn}{T}\right) dt \quad (\text{запишем выражение в общем виде}),$$

$$a_n = \frac{2}{T} \int_{-k/2}^{k/2} A \cos\left(\frac{2\pi tn}{T}\right) dt \quad (\text{перейдём к частному случаю}),$$

$$a_n = \frac{2A}{T} \left[ \frac{T}{2\pi n} \sin\left(\frac{2\pi tn}{T}\right) \right] \Big|_{-k/2}^{k/2} \quad (\text{вычислим интеграл}),$$

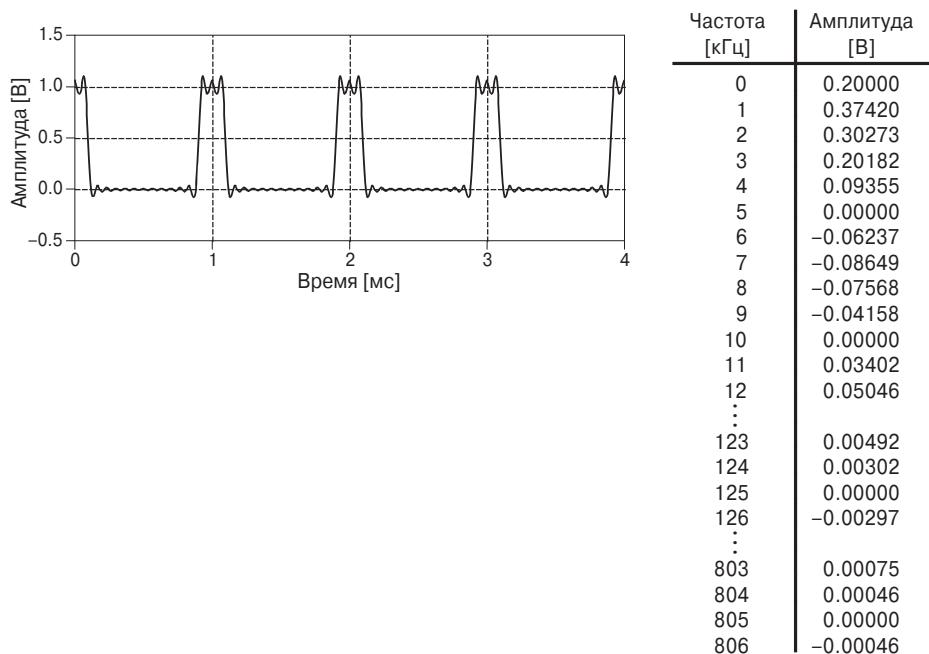
$$a_n = \frac{2A}{n\pi} \sin(\pi nd) \quad (\text{упростим выражение}).$$

Коэффициенты  $b$  вычисляются аналогично. В данном случае они равны нулю. То есть сигнал можно представить совокупностью только косинусоидальных колебаний, а синусоидальные компоненты в нём отсутствуют.

Значения коэффициентов  $a$  и  $b$  изменятся, если сдвинуть сигнал по временной оси вправо или влево. В рассмотренном примере коэффициенты  $b$  получились равными нулю только потому, что последовательность импульсов симметрична относительно оси ординат (т. е. относительно момента времени  $t = 0$ ). Так как сигнал обладает чётной симметрией, то он складывается только из тех колебаний, которые тоже обладают чётной симметрией, т. е. только из косинусоид. Поэтому все коэффициенты  $b$  равны нулю. Если сигнал характеризуется нечётной симметрией (это значит, что левая и правая части симметричны относительно  $t = 0$  после изменения арифметического знака одной из частей на противоположный), то он может состоять только из колебаний с нечётной симметрией, т. е. из синусоид. В этом случае все коэффициенты  $a$  равны нулю. При анализе спектра в полярных координатах,  $M_n$  и  $\theta_n$ , сдвиг во временной области не оказывает влияния на модуль  $M_n$ , при этом фаза  $\theta_n$  меняется прямо пропорционально задержке сигнала.

Вернёмся ещё раз к примеру с последовательностью прямоугольных импульсов. Пусть частота следования импульсов равна 1 кГц, амплитуда — 1 В, скважность — 0.2. Амplitуды гармоник такого сигнала сведены в таблицу (Рис. 13.12), а рядом с ними показан сигнал, восстановленный по первым 14 гармоникам ряда. Даже при столь большом числе гармоник результат синтеза получается очень далёким от идеала. Выражаясь языком математиков, следует сказать: «скорость сходимости ряда Фурье весьма мала». Это значит, что всякое резкое изменение уровня сигнала во временной области приводит к возникновению гармоник на очень высоких частотах. Здесь полезно вспомнить и обратный эффект — явление Гибса, подробно рассмотренное в Главе 11.

На разложении сигналов в ряд Фурье основана операция умножения частоты. Предположим, что вам необходимо создать генератор синусоидальных сигналов частотой 100 МГц с очень высокой стабильностью частоты. Такой генератор может потребоваться, например, для построения радиостанции. Требование высокой стабильности частоты колебаний приводит к необходимости использования кварцевого резонатора. Проблема в том, что кварцевые резонаторы рассчитаны на работу в диапазоне частот до 10 МГц. Оказывается, что кварцевого резонатора с рабочей частотой 1...10 МГц при наличии схемы умножения частоты вполне достаточно. С целью умножения частоты синусоидальный сигнал кварца намеренно искажается, например путём усечения максимумов синусоиды, в результате работы в режиме насыщения. Нужные гармоники, входящие в состав получившегося искажённого сигнала, выделяются полосовыми фильтрами. Таким способом удается удвоить, утроить частоту и в принципе повысить частоту в ещё большее число раз. Но наиболее широко распространён метод многократного использова-



**Рис. 13.12.** Пример синтеза сигнала по гармоникам ряда Фурье. Исходный сигнал — последовательность импульсов с частотой 1 кГц, амплитудой 1 В и скважностью 0.2 (см. Рис. 13.11). В таблице приводятся амплитуды гармоник такого сигнала, а на графике изображён сигнал, восстановленный (синтезированный) по 14 первым гармоникам.

ния операции умножения частоты при коэффициенте умножения на каждой ступени, равном 2 или 3. Ряд Фурье играет важную роль при описании такого типа устройств, потому что с его помощью можно рассчитать параметры результирующего сигнала полученного после нелинейного преобразования и выделения необходимых гармоник.

## ВВЕДЕНИЕ В ЦИФРОВУЮ ФИЛЬТРАЦИЮ

*Цифровые фильтры* предназначены для решения двух основных задач: 1) разделения двух и более распространяющихся совместно сигналов и 2) восстановления сигналов, которые были каким-либо образом искажены. Для решения данных задач можно воспользоваться аналоговыми фильтрами, однако цифровые фильтры позволяют достичь намного более высокой точности. Цифровые фильтры, получившие наиболее широкое распространение, подробно рассмотрены в следующих семи главах. Данная глава является вводной и описывает общие понятия и характеристики, которые будут использованы при изложении материала последующих глав.

### 14.1. Основные понятия

*Цифровые фильтры* занимают в ЦОС одно из наиболее важных мест. Уникальные свойства цифровых фильтров позволили значительно расширить область применения ЦОС. Как уже было сказано, цифровые фильтры применяются для решения двух основных задач: разделения и восстановления сигналов. Операция разделения применяется в тех случаях, когда полезный сигнал распространяется совместно с другими сигналами, играющими роль помех. Рассмотрим, например, сигнал электрокардиограммы (ЭКГ). Представьте себе прибор, измеряющий электрические колебания, производимые сердцем ребёнка, находящегося в утробе матери. Очевидно, что в регистрируемом сигнале будут присутствовать колебания, обусловленные дыханием и биением сердца матери. От фильтра требуется обеспечить такое разделение полезного сигнала и помехи, после которого их можно анализировать независимо друг от друга.

Задача восстановления решается в тех случаях, когда сигнал подвергается различного рода искажениям. Например, цифровую фильтрацию применяют, чтобы улучшить звучание музыкальных произведений, записанных на оборудовании низкого качества. Другой пример — устранение размытости изображения, полученной из-за плохой фокусировки объектива или в результате дрожания камеры.

Для решения описанных выше задач можно использовать и аналоговые, и цифровые фильтры. Вопрос в том, какие из них лучше. Реализация аналоговых фильтров связана, как правило, с меньшими денежными затратами. Кроме того, эти фильтры обладают более высоким быстродействием и широким динамическим диапазоном по амплитуде и по частоте. В свою очередь цифровые фильтры превосходят аналоговые по точности воспроизведения желаемых частотных характеристик. Например, фильтр, рассматриваемый в Главе 16, имеет коэффициент усиления  $1 \pm 0.0002$  в полосе частот от нуля до 1000 Гц и не более 0.0002 на частотах более 1001 Гц. Вся переходная область умещается в диапазоне шириной 1

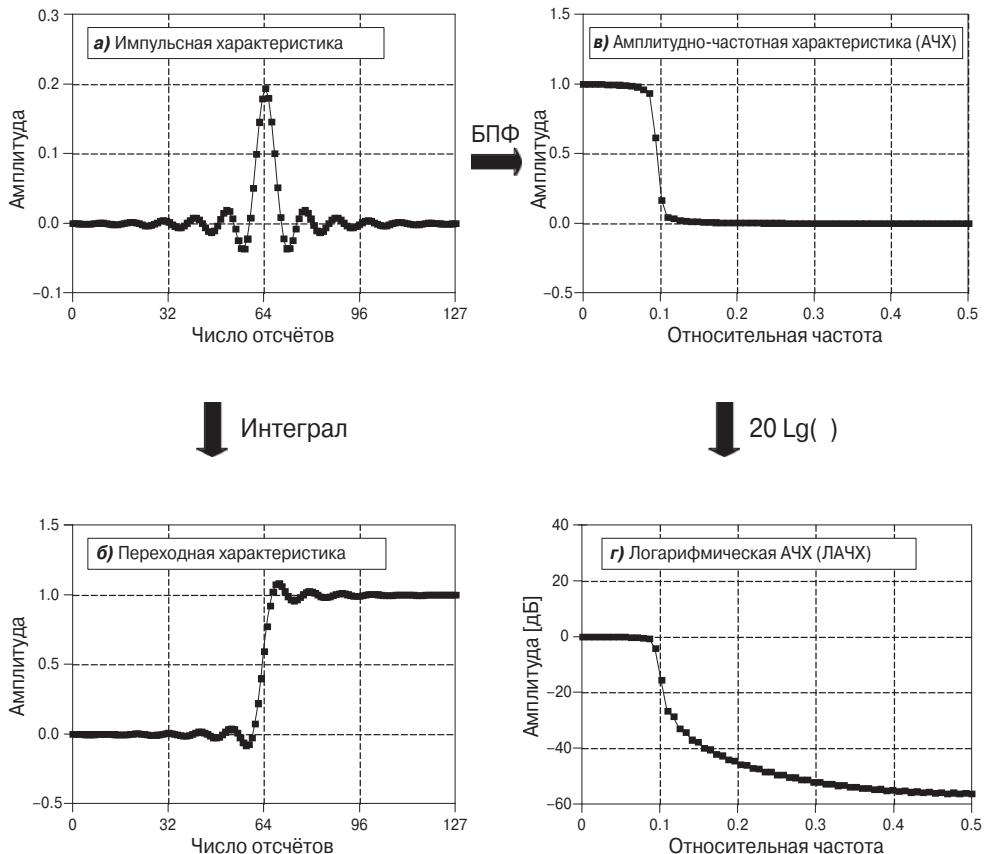
Гц. От операционных усилителей, применяемых в аналоговых схемах, такого качества ожидать не приходится, поэтому цифровые фильтры по точности превосходят аналоговые в тысячи раз. Следствием этого являются значительные различия в тех подходах, которые используются при проектировании аналоговых и цифровых фильтров. Для аналоговых фильтров важнее всего учесть ограничения, обусловленные элементной базой, такие как точность и стабильность параметров резисторов и конденсаторов. Цифровые фильтры чаще всего позволяют достигать заданной точности. Поэтому здесь на первый план выходят проблемы, связанные с ограниченностью динамического диапазона.

Наиболее широко в ЦОС используются сигналы, дискретизированные во временной области, — отсчёты сигнала выбираются с постоянным шагом по времени. Однако могут встречаться и другие сигналы. Например, выборка отсчётов может идти с равномерным интервалом в пространстве, примером чего служит считывание информации с массива тензодатчиков, размещённых по крылу самолёта с шагом в один сантиметр. Кроме временной и пространственной можно перечислить и ряд других областей описания сигналов, но на практике широкого применения они не получили. Поэтому следует помнить, что если в ЦОС встречается термин *дискретизация*, то он может относиться как к дискретизации непосредственно во временной области, так и к дискретному представлению сигнала в любых других областях.

На Рис. 14.1 рассматриваются три основные характеристики фильтров: *импульсная, переходная и частотная*. Каждая из них полностью определяет свойства линейного фильтра, т. е. содержит исчерпывающую информацию о его фильтрующих свойствах. Отличаются они лишь формой представления информации. Если задана одна из характеристик, то всегда можно рассчитать две другие. Все три характеристики играют важную роль, так как они позволяют определить реакцию фильтра при различных исходных данных.

Самый простой способ реализации цифрового фильтра — *свёртка* входного сигнала с импульсной характеристикой. Данный способ позволяет построить любой практически реализуемый линейный фильтр. (Это должно быть очевидно для вас. Если это не так, вы можете прочитать ещё раз предыдущие разделы, где дано общее представление о ЦОС.) Когда отсчёты импульсной характеристики используются в качестве весовых коэффициентов цифрового фильтра, то их называют *ядром фильтра*.

Другая форма построения цифровых фильтров — *рекурсивные фильтры*. В фильтрах, основанных на операции свёртки (*нерекурсивных*), каждый отсчёт выходного сигнала получается в результате умножения задержанных отсчётов входного сигнала на соответствующие весовые коэффициенты и последующего суммирования полученных произведений. Рекурсивные фильтры представляют собой более широкий класс фильтров и, кроме отсчётов входного сигнала, учитывают также отсчёты выходного сигнала, полученные на предыдущих итерациях. Набор весовых коэффициентов рекурсивного фильтра отличается от весовых коэффициентов (ядра) нерекурсивного фильтра. Подробнее о рекурсивных фильтрах рассказывается в Главе 19. А пока вам достаточно усвоить, что любой линейный фильтр полностью определяется импульсной характеристикой, которая представляет собой реакцию фильтра на *единичный импульс*. Импульсная характеристика рекурсивного фильтра может быть представлена в виде суммы синусоид



**Рис. 14.1.** Характеристики фильтров. Свойства линейного фильтра полностью определяются любой из трёх основных характеристик: импульсной, переходной и частотной. **а)** Импульсная характеристика. **б)** Переходная характеристика. Связана с импульсной характеристикой интегральной зависимостью. **в)** Амплитудно-частотная характеристика (АЧХ). Рассчитывается как быстрое преобразование Фурье (БПФ) импульсной характеристики с последующим вычислением модуля преобразования. **г)** Логарифмическая АЧХ (ЛАЧХ). Получается при переходе к логарифмическому масштабу по оси ординат.

дальних сигналов с убывающей по экспоненциальному закону амплитудой. Это означает, что рекурсивный фильтр имеет импульсную характеристику бесконечной длины. Однако, начиная с некоторого момента времени, амплитуда синусоидальных колебаний затухает настолько, что становится ниже уровня *шумов округления*, поэтому последующими отсчётами импульсной характеристики можно пренебречь. Рекурсивные фильтры называют *фильтрами с бесконечной импульсной характеристикой* или *БИХ-фильтрами*. Нерекурсивные фильтры, основанные на использовании операции свёртки, называют *фильтрами с конечной импульсной характеристикой* или *КИХ-фильтрами*.

Как вам уже известно, импульсная характеристика является реакцией системы на единичный импульс. Второй важной временной характеристикой является *переходная характеристика*, которая определяется как реакция на «единичный скачок» (единичную ступенчатую функцию). Так как «единичный скачок» пред-

ставляет собой интеграл от единичного импульса, то переходная характеристика связана с импульсной интегральной зависимостью. Это означает, что переходную характеристику можно получить двумя способами: 1) подать на вход фильтра сигнал в виде «единичного скачка» и зафиксировать его выходную реакцию или 2) проинтегрировать импульсную характеристику (с точки зрения математики вместе термина «интегрирование» более правильным считается использование термина «дискретное интегрирование», означающее суммирование с нарастающим итогом).

Частотная характеристика цифрового фильтра связана с его импульсной характеристикой *дискретным преобразованием Фурье (ДПФ)*, для вычисления которого используют алгоритм *БПФ*, но об этом чуть позже. Для графического отображения зависимости амплитуды от частоты может быть использован как линейный масштаб (**Рис. 14.1в**) — в этом случае имеем дело с *амплитудно-частотной характеристикой (АЧХ)*, так и логарифмический масштаб (**Рис. 14.2г**) — такая характеристика называется *логарифмической АЧХ* или *ЛАЧХ*. Линейный масштаб удобен для отображения амплитудной характеристики в полосе пропускания и в переходной зоне. Логарифмический масштаб (в децибелях) необходим для исследования частотной характеристики в зоне подавления фильтра.

Теперь стоит остановиться на децибелах. Бел — единица измерения логарифмической шкалы (названа по имени Александра Грехема Белла), которая соответствует изменению некоторой величины в 10 раз. Например, сигнал на выходе электронной схемы с коэффициентом усиления 3 бела превышает по мощности сигнал на её входе в  $10 \times 10 \times 10 = 1000$  раз. Децибел (dB) равен одной десятой бела. То есть значения -20 dB, -10 dB, 0 dB, 10 dB и 20 dB означают увеличение мощности в 0.01, 0.1, 1, 10 и 100 раз. Другими словами, каждые десять децибел означают десятикратное увеличение мощности.

Здесь очень легко допустить ошибку, потому что чаще всего измеряют не мощность сигнала, а его амплитуду. Допустим, коэффициент передачи усилителя по мощности равен 20 dB. Тогда, по определению, сигнал усиливается по мощности в 100 раз. Поскольку амплитуда пропорциональна квадратному корню из мощности, то она увеличивается только в 10 раз. То есть 20 dB соответствуют стократному увеличению мощности сигнала и лишь десятикратному увеличению амплитуды. Каждые последующие двадцать децибел соответствуют дополнительному десятикратному увеличению амплитуды. Справедливы следующие уравнения ( $P$  — мощность,  $A$  — амплитуда):

$$\begin{aligned} dB &= 10 \lg \frac{P_2}{P_1}, \\ dB &= 20 \lg \frac{A_2}{A_1}. \end{aligned} \tag{14.1}$$

**Определение понятия «децибел».** Децибел — единица измерения логарифмической шкалы, отношение уровня одного (выходного) сигнала к уровню другого (входного) сигнала. С отношением двух сигналов по мощности и их отношением по амплитуде децибели связаны разными уравнениями.

В представленных выше уравнениях используется десятичный логарифм, тогда как во многих компьютерных языках программирования в распоряжении раз-

работчика имеется только подпрограмма вычисления натурального логарифма. Чтобы воспользоваться натуральным логарифмом вместо десятичного, уравнения могут быть преобразованы к следующему виду:  $dB = 4.342945 \ln(P_2/P_1)$  и  $dB = 8.685890 \ln(A_2/A_1)$ .

Децибелы отражают относительное различие двух величин. Поэтому ими удобно пользоваться при измерении коэффициента усиления системы, который как раз и показывает отношение выходного сигнала к входному. Однако в радиотехнике децибелы используют также для измерения амплитуды (или мощности) отдельного сигнала, подразумевая при этом отношение к некоторой эталонной величине. Общепринятыми являются следующие обозначения: **dBV** — означает, что в качестве эталона для сравнения выбран сигнал со среднеквадратической амплитудой 1 В; **dBm** — указывает на то, что за эталон принят сигнал мощностью 1 мВт при измерении на нагрузке 600 Ом (действующее значение напряжения равно 0.78 В).

Если вы плохо разбираетесь в децибелях, запомните два важных правила. Во-первых, величина  $-3$  дБ означает уменьшение по амплитуде до уровня 0.707 от исходного значения (т. е. мощность сигнала уменьшается в 2 раза). Во-вторых, можно запомнить несколько простых соответствий между децибелами и отношениями сигналов по амплитуде:

60 дБ	=	1000
40 дБ	=	100
20 дБ	=	10
0 дБ	=	1
-20 дБ	=	0.1
-40 дБ	=	0.01
-60 дБ	=	0.001

## 14.2. Формы представления информации в сигнале

Очень важно при решении задач ЦОС понимать, в какой форме информация представлена в обрабатываемом сигнале. Применительно к искусственно созданным сигналам можно говорить о наличии множества форм представления информации, таких, как разнообразные виды модуляции: АМ, ЧМ, однополосная модуляция, импульсно-кодовая модуляция, широтно-импульсная модуляция и т. д. Этот список можно продолжать очень долго. К счастью, для сигналов естественного (природного) происхождения характерно наличие всего двух форм представления информации. Назовём их *кодированием во временной области* и *кодированием в частотной области*.

О кодировании во временной области говорят в том случае, когда в анализируемом сигнале нас интересует время наступления некоторого события и значение амплитуды сигнала в этот момент. Например, мы исследуем солнечное излучение, сохраняя результаты измерений с интервалом в одну секунду. Каждый отсчёт сигнала хранит информацию о мощности излучения на момент измерения. Если произойдёт солнечная вспышка, то информацию о моменте её возникновения, о длительности вспышки и характере её протекания можно получить

непосредственно по временным отсчётам сигнала. Информация содержится в каждом из отсчётов полученного сигнала, что позволяет пользоваться ими отдельно друг от друга. Даже при наличии всего лишь одного отсчёта можно говорить, что имеется некоторая полезная для нас информация. Данная форма представления информации в сигнале является самой простой.

Информация, заложенная в частотной области, напротив, «не лежит на поверхности». Многие вещи в нашем мире совершают периодические колебания: если по бокалу ударить ногтем, он начнёт генерировать звуковые волны и звенеть; маятник старых дедушкиных часов раскачивается из стороны в сторону; звезды и планеты врачаются друг относительно друга и вокруг собственной оси. Для получения информации о таких колебательных процессах требуется измерить частоту, фазу и амплитуду колебаний. Если нам удастся узнать частоту звука, издаваемого бокалом, а также частоты содержащихся в этом звуке гармоник, то мы сможем судить о массе и упругости материала, из которого он сделан. Каждый отдельно взятый отсчёт не несёт для нас информации. Она может быть извлечена только в результате анализа большого количества отсчётов поступившего сигнала.

Из приведенных рассуждений становится понятной разница в назначении переходной и частотной характеристик. Переходная характеристика описывает влияние системы на информацию, заложенную во временной области, тогда как частотная характеристика описывает преобразование информации, заложенной в частотной области. Об этом необходимо помнить при расчёте фильтров, поскольку улучшение временных характеристик фильтра приводит к ухудшению его частотных свойств. Если задача состоит в устраниении шумов на ЭКГ (измеряемый сигнал представлен во временной области), то более важным является точное воспроизведение переходной характеристики фильтра, а частотная характеристика играет второстепенную роль. Если же фильтр планируется использовать в слуховом аппарате (информация содержится в частотной области), то важна только частотная характеристика, а переходная характеристика не представляет никакого интереса. Далее рассмотрим временные и частотные параметры фильтров.

### 14.3. Временные характеристики

Наверное, то решающее значение, которое отводится *переходной характеристике* при проектировании фильтров во временной области, является не вполне очевидным. Действительно, почему же импульсная характеристика такой важности не представляет? Ответ на этот вопрос кроется в способе работы человеческого мозга по осознанию и обработке информации. Не забывайте, что переходная, импульсная и частотная характеристики содержат в себе одну и ту же информацию, которая отличается только формой её представления. Однако при анализе во временной области переходная характеристика оказывается самой важной, потому что она наилучшим образом соответствует человеческому восприятию информации, содержащейся в сигнале.

Предположим, что имеется запись сигнала какого-либо неизвестного происхождения и от вас требуется проанализировать этот сигнал. Первым вашим действием станет попытка разделить сигнал на отдельные фрагменты, на протяжении которых его параметры слабо изменяются. Вы так поступите, даже не осознавая

этого: так устроен наш мозг. Одни области могут оказаться более гладкими, другие будут характеризоваться значительными перепадами амплитуды, в ряде областей будут присутствовать сильные шумовые помехи. В результате такой сегментации определяются граничные точки областей. И вот тут как раз появляется необходимость введения ступенчатой функции. Ступенчатая функция позволяет наиболее просто описать переход между двумя различающимися по своим параметрам областями. Она соответствует началу или окончанию какого-либо события. Ступенчатая функция всегда указывает на существование некоторого различия между тем, что находится слева, и тем, что находится справа. Таким образом, в процессе своего восприятия информации, представленной во временной области, человеческий мозг неявно использует набор ступенчатых функций, чтобы разделить имеющуюся информацию на однородные области. В свою очередь переходная характеристика оказывается необходимой, так как она описывает характер изменения сигнала в точках перехода при обработке фильтром.

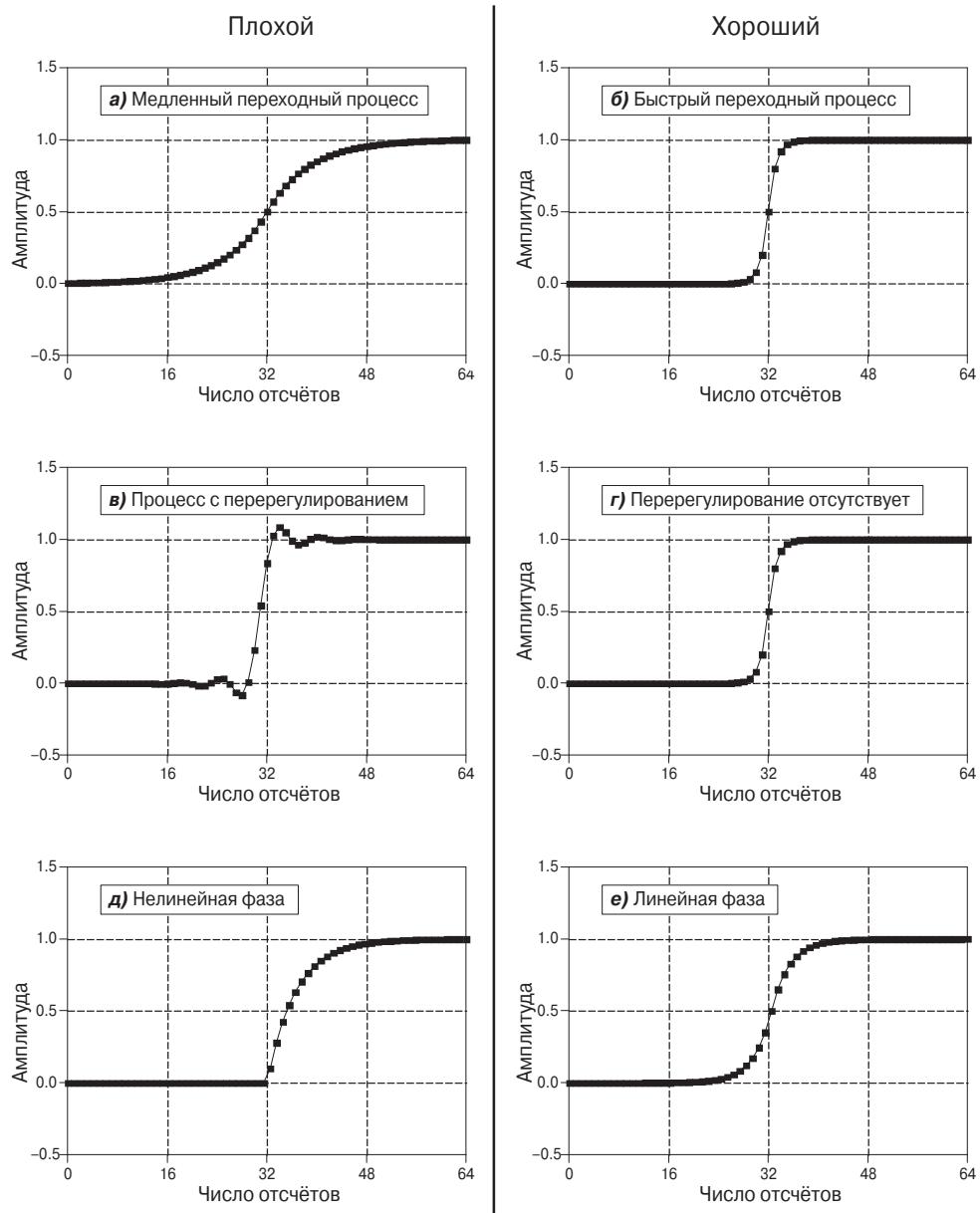
Основные параметры переходной характеристики, используемые при расчёте фильтров, показаны на Рис. 14.2. Для разделения присутствующих в сигнале событий требуется, чтобы длительность переходной характеристики не превышала интервала между событиями. Следовательно, как показано на (*а* и *б*), переходный процесс должен быть настолько быстрым, насколько это возможно. Как правило, время установления определяется количеством отсчётов дискретного сигнала, соответствующих изменению амплитуды сигнала от уровня 10% до 90% от её уставновившегося значения. Почему не всегда возможно использовать фильтры с высоким быстродействием? Причины могут быть самые разные: необходимость подавления шума, наличие собственных неустранимых ограничений в системах сбора данных, устранение наложения спектров и т. д.

На (*в* и *г*) демонстрируется очень важный параметр фильтра — *перерегулирование*. Предпочтительнее использовать фильтры с малым перерегулированием, чтобы уменьшить вносимые в сигнал амплитудные искажения, которые являются наиболее значимыми искажениями во временной области. При практическом измерении сигналов часто возникает следующий вопрос: связан ли наблюдаемый всплеск амплитуды с самим измеряемым сигналом или он отражает переходный процесс?

Очень часто требуется обеспечить симметрию верхней и нижней частей переходной характеристики, как показано на (*д* и *е*). Такая симметрия необходима для того, чтобы передний и задний фронты импульсов получали искажения одинаковой формы. Фильтры с подобной симметрией называются *линейно-фазовыми*, поскольку их фазо-частотная характеристика (ФЧХ) имеет вид прямой линии (Глава 19). Постарайтесь как следует разобраться с тремя вышеперечисленными показателями, так как они играют важную роль при анализе фильтров во временной области.

## 14.4. Частотные характеристики

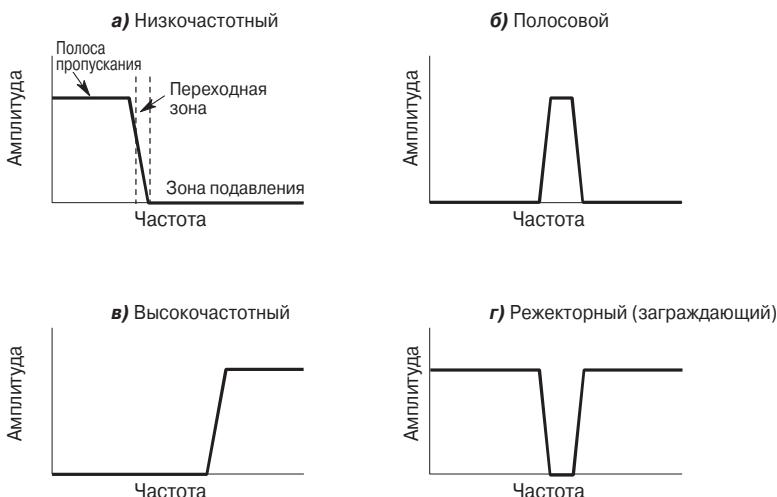
На Рис. 14.3 рассмотрены частотные характеристики четырёх основных типов фильтров. Эти фильтры предназначены для того, чтобы пропускать без изменения одни частоты входного сигнала и задерживать другие. Диапазон пропускае-



**Рис. 14.2.** Показатели, характеризующие качество фильтра во временной области. Качество работы фильтра во временной области позволяет оценить переходная характеристика. Наиболее важными являются три показателя: **(а и б)** — быстродействие (время установления); **(в и г)** — перерегулирование; **(д и е)** — линейность фазы (симметрия верхней и нижней частей переходной характеристики).

мых фильтром частот называется *полосой пропускания*, а диапазон задерживаемых частот — *зоной подавления*. Между ними располагается *переходная зона* фильтра. Если фильтр имеет узкую переходную зону, то говорят, что он обладает высокой

частотной избирательностью. Границная частота, разделяющая полосу пропускания и переходную зону, называется *частотой среза*. Для аналоговых фильтров частота среза определяется точкой, в которой АЧХ спадает до уровня 0.707 ( $-3\text{ dB}$ ). Для цифровых фильтров нет таких жестких стандартов, и граничная частота часто определяется точкой, в которой АЧХ спадает до одного из следующих уровней: 99, 90, 70.7 и 50%.

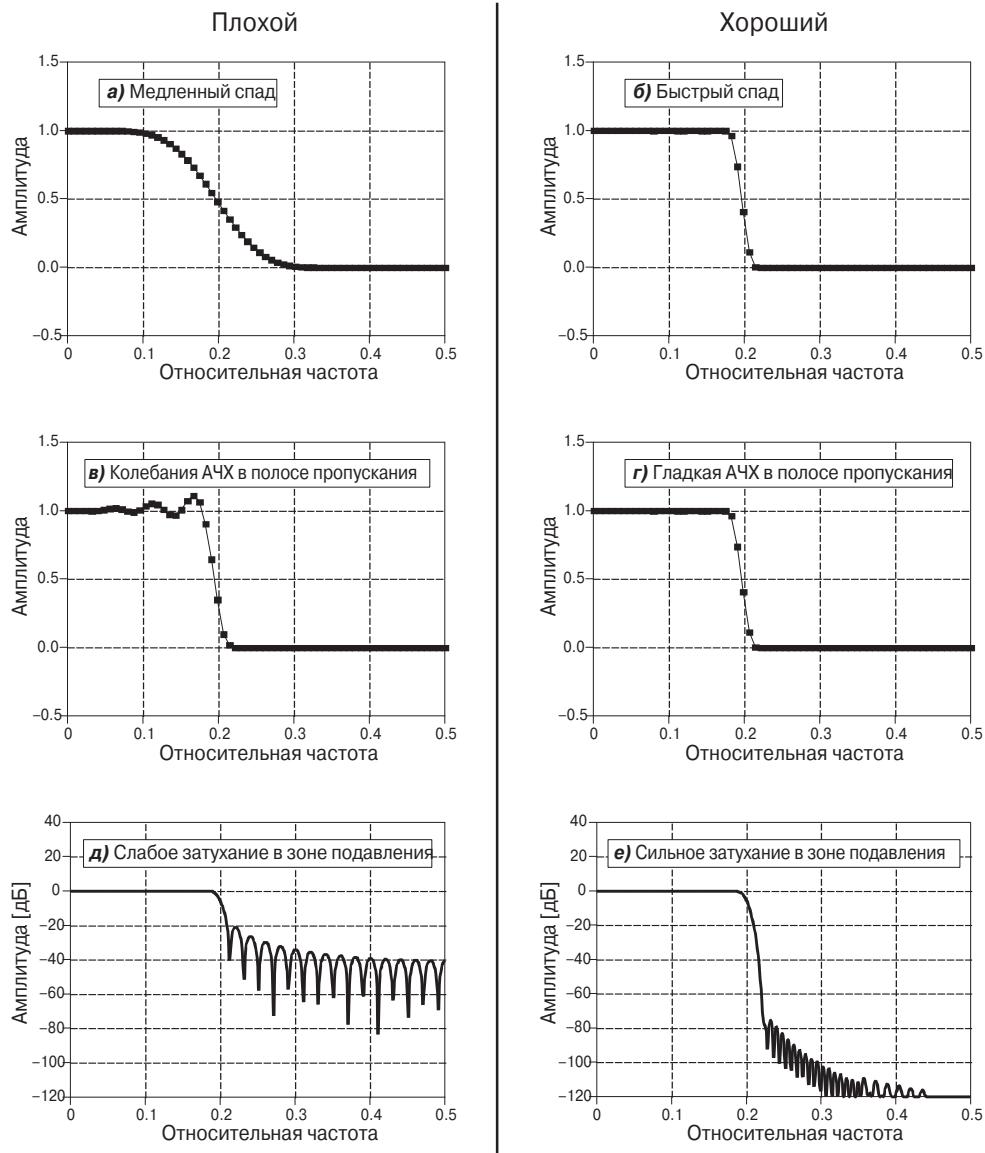


**Рис. 14.3.** Четыре основных типа АЧХ цифровых фильтров. Главная задача цифровой частотной селекции заключается в построении фильтра, способного почти без искажений пропускать сигналы в заданном частотном диапазоне (в полосе пропускания) и подавлять частоты за пределами этого диапазона (в зоне подавления).

На **Рис. 14.4** показаны три показателя, характеризующие качество работы фильтра в частотной области. Способность фильтра разделять близкие частоты называется *частотной избирательностью* и определяется крутизной спада АЧХ (**а** и **б**). Для устранения искажений, вносимых фильтром в пропускаемый им сигнал, необходимо, чтобы *неравномерность АЧХ* в полосе пропускания стремилась к нулю (**в** и **г**). Для эффективного подавления частот в зоне непрозрачности необходимо высокий уровень затухания (**д** и **е**).

Почему среди этих параметров нет ни одного, характеризующего фазу? Во-первых, в большинстве практических приложений, связанных с обработкой сигналов в частотной области, фаза не играет роли. Например, в звуковоспроизводящей аппаратуре фаза может принимать совершенно случайные значения и практически не несёт в себе никакой информации. Во-вторых, при расчёте цифровых фильтров очень легко добиться того, чтобы фазо-частотная характеристика получилась идеальной, т. е. чтобы все частоты, лежащие в полосе пропускания, проходили на выход фильтра с нулевым фазовым сдвигом (глава 19). Заметим, что для аналоговых фильтров данная проблема является «ужасно» сложной.

В предыдущих главах уже говорилось о том, как с помощью ДПФ найти по импульсной характеристике системы её частотную характеристику. Кратко повторим здесь этот материал. Самый быстрый способ вычисления ДПФ заключается в использовании алгоритма *БПФ*, описанного в Главе 12. Алгоритм БПФ позволяет для



**Рис. 14.4.** Показатели, характеризующие качество фильтра в частотной области (на примере низкочастотного фильтра). Наиболее важными являются следующие три показателя: крутизна спада АЧХ в переходной зоне (**а** и **б**), неравномерность в полосе пропускания (**в** и **г**), уровень затухания в зоне непрозрачности (**д** и **е**).

заданной импульсной характеристики фильтра длиной  $N$  отсчётов найти его частотную характеристику, вещественная и мнимая части которой состоят из  $N$  дискретных отсчётов каждая. Но лишь отсчёты с порядковыми номерами  $0 \dots N/2$  несут полезную информацию, а остальные являются их зеркальным отражением и, следовательно, могут быть исключены из дальнейшего рассмотрения (эти отсчёты частотной характеристики соответствуют отрицательным значениям частоты).

Разделение на вещественную и мнимую компоненты неудобно для человеческого восприятия, поэтому предпочтительнее оказывается полярная форма представления комплексных характеристик (Глава 8). Комплексная частотная характеристика при этом оказывается разбитой на амплитудно-частотную и фазо-частотную характеристики, каждая из которых представлена дискретными отсчётами с порядковыми номерами  $0...N/2$  (всего  $N/2 + 1$  отсчёт). Например, если исходная импульсная характеристика представлена 256 дискретными отсчётами, то в полученной частотной характеристике следует оставить отсчёты с номерами  $0...128$ . При этом отсчёт номер 0 соответствует постоянной составляющей (нулевой частоте), а отсчёт номер 128 — половине *частоты дискретизации*. Запомните, что спектры дискретных сигналов рассматриваются на частотах, не превышающих половины частоты дискретизации: спектр дискретного сигнала на более высоких частотах периодически повторяет спектр основного (низкочастотного) диапазона.

Число отсчётов импульсной характеристики не может быть выбрано произвольным. Например, требуется найти частотную характеристику КИХ-фильтра, имеющего 80 весовых коэффициентов. Поскольку БПФ можно применить только к массиву дискретных отсчётов, размер которого равен степени числа 2, то необходимо дополнить имеющийся изначально массив из 80 весовых коэффициентов 48 нулями, чтобы получить массив длиной 128 отсчётов. Дополнение нулями не изменяет импульсную характеристику фильтра. Чтобы лучше понять это, представьте себе весь процесс выполнения дискретной свёртки. Добавленные к импульсной характеристике нулевые отсчёты не оказывают влияния на формирование результирующей последовательности.

Продолжая применять эту операцию, можно и далее добавлять нули к импульсной характеристике, доведя её длину, скажем, до 256, 512 или 1024 отсчётов. Увеличение длины импульсной характеристики обеспечивает более плотное размещение отсчётов в полученной в результате применения алгоритма БПФ частотной характеристике. То есть отсчёты полученной частотной характеристики будут расположены чаще на интервале частот от постоянной составляющей до половины частоты дискретизации. В пределе при дополнении импульсной характеристики бесконечным числом нулей расстояние между соседними отсчётами получаемой частотной характеристики стремится к нулю, а сами отсчёты сливаются в одну непрерывную кривую. Можно сказать, что частотная характеристика фильтра является непрерывной функцией, заданной на интервале от постоянной составляющей до половины частоты дискретизации, а применение БПФ соответствует её дискретизации с некоторым постоянным шагом. До какой длины нужно дополнить импульсную характеристику фильтра, для того чтобы найти его частотную характеристику? Для начала можно попробовать дополнить импульсную характеристику нулями до 1024 отсчётов, а затем уже изменить это значение в большую или меньшую сторону, если выяснится, что разрешение по частоте слишком мало или что время вычисления недопустимо велико.

Учтите, что понятия «хороший» и «плохой», которыми мы часто пользовались в этой главе, носят идеализированный характер и в ряде случаев необходим дополнительный более глубокий анализ. На снятой ЭКГ в рассмотренном выше примере присутствуют наводки от сети питания. Хотя информация представлена во временной области, от помех проще всего избавиться, «вырезав» соответствующую полосу частот, т. е. рассматривая сигнал в частотной области. Наилучшее

решение данной проблемы заключается в достижении некоторого компромисса между повышением качества во временной и частотной областях. Такой комбинированный критерий несколько отличается от введённых выше показателей качества фильтра. Запомните первое правило обучения: в любой книжной фразе следует не раз усомниться и осмыслить её суть, а не принимать утверждения как неоспоримую истину.

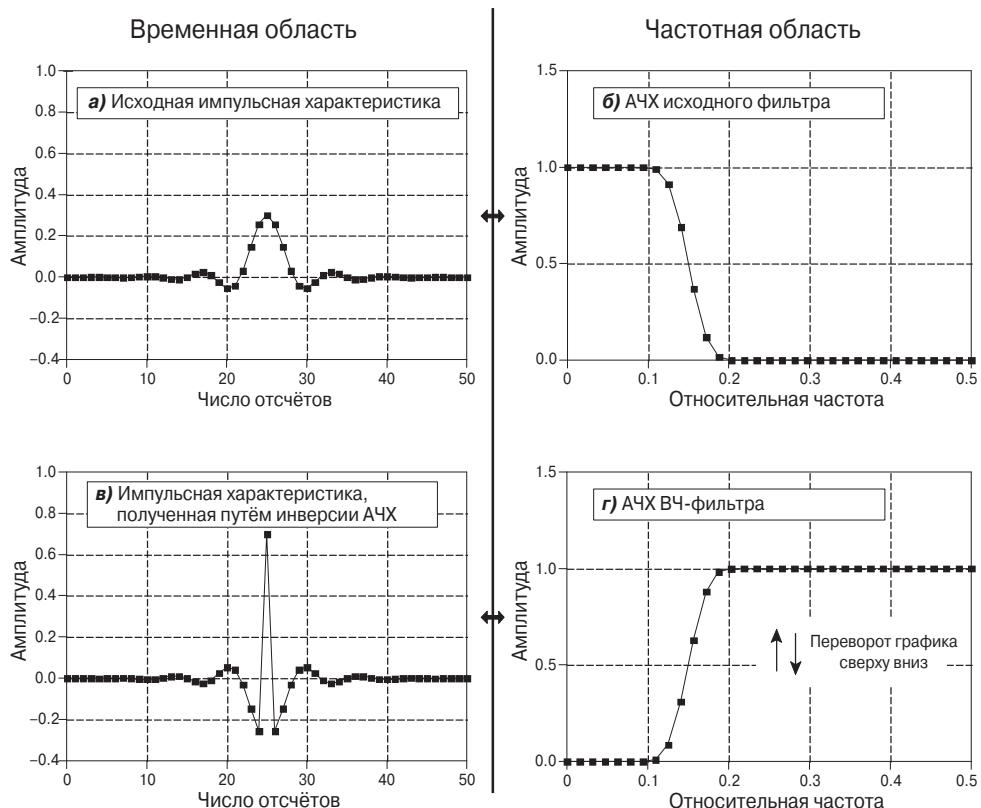
## 14.5. Высокочастотные, полосовые и режекторные фильтры

При проектировании *высокочастотных, полосовых и режекторных* (заграждающих) фильтров предварительно рассчитывают *низкочастотный фильтр (НЧ-фильтр)*, а затем преобразуют его к желаемому виду. Именно по этой причине, говоря о расчёте фильтров, чаще всего описывают только НЧ-фильтры. Существует два метода преобразования НЧ-фильтра в высокочастотный (ВЧ-фильтр) — *инверсия АЧХ* и *обращение АЧХ*. Оба метода нашли широкое распространение.

Пример инверсии показан на Рис. 14.5. На (а) изображена импульсная характеристика *оконного НЧ-фильтра* (Глава 16). Такой КИХ-фильтр имеет 51 весовой коэффициент, причём многие из коэффициентов настолько малы, что на графике неотличимы от нуля. АЧХ фильтра (б) можно получить, дополнив импульсную характеристику 13 нулями и применив БПФ 64-го порядка. Преобразование НЧ-фильтра в ВЧ-фильтр выполняется в два действия: 1) меняем арифметические знаки всех коэффициентов фильтра; 2) добавляем единицу к среднему отсчёту, относительно которого импульсная характеристика симметрична. В результате получаем ВЧ-фильтр, импульсная характеристика и АЧХ которого показаны на (в и г). Инверсия АЧХ выражается в перевороте графика АЧХ сверху вниз. При этом полоса пропускания превращается в зону подавления, а зона подавления — в полосу пропускания. Это значит, что НЧ-фильтр превращается в ВЧ-фильтр, ВЧ-фильтр — в НЧ-фильтр, полосовой — в режекторный, а режекторный — в полосовой.

Рис. 14.6 помогает разобраться в том, почему рассмотренное нами преобразование импульсной характеристики фильтра приводит к инверсии его частотной характеристики. Как показано на (а), входной сигнал  $x[n]$  поступает одновременно на два звена, одним из которых является НЧ-фильтр с импульсной характеристикой  $h[n]$ , а другим — звено задержки (всепропускающий фильтр), импульсная характеристика которого описывается смешённой дискретной дельта-функцией  $\delta[n]$ . Результирующий выходной сигнал  $y[n]$  равен разности сигналов, полученных на выходе звена задержки и на выходе НЧ-фильтра. Так как низкочастотные компоненты спектра вычитаются из исходного сигнала, то в выходном сигнале остаются только верхние частоты, а это значит, что мы получили ВЧ-фильтр.

Программная реализация данного метода предполагает два этапа: сначала сигнал обрабатывается НЧ-фильтром, а затем результат фильтрации вычитается из исходного сигнала. Две эти операции могут, однако, быть заменены одной процедурой, если объединить весовые коэффициенты двух соответствующих фильтров. Как указывалось в Главе 7, при параллельном соединении двух фильт-

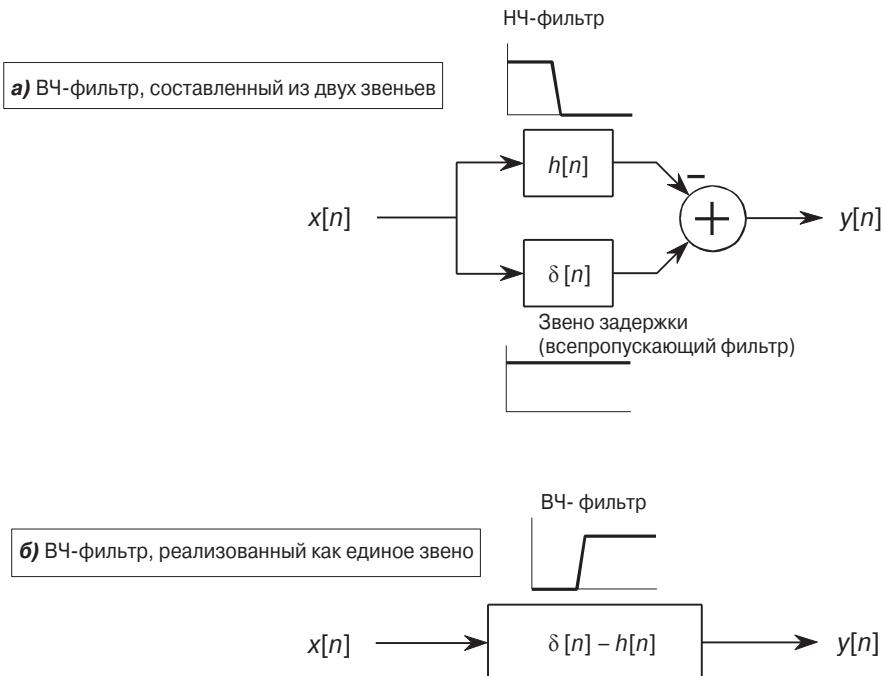


**Рис. 14.5.** Пример инверсии АЧХ: *а)* импульсная характеристика НЧ-фильтра; *б)* АЧХ НЧ-фильтра; *в)* импульсная характеристика ВЧ-фильтра, полученная из импульсной характеристики НЧ-фильтра после изменения арифметических знаков при весовых коэффициентах на противоположные и добавления единицы к среднему коэффициенту, представляющему собой центр симметрии; *г)* АЧХ полученного ВЧ-фильтра является отражением АЧХ исходного НЧ-фильтра «сверху вниз».

тров их импульсные характеристики складываются. Как показано на *(б)*, весовые коэффициенты ВЧ-фильтра могут быть найдены в результате вычитания коэффициентов НЧ-фильтра из дискретной дельта-функции:  $\delta[n] - h[n]$ . То есть необходимо выполнить знаковую инверсию всех элементов массива весовых коэффициентов исходного ВЧ-фильтра, после чего добавить единицу к коэффициенту, расположенному в середине массива.

В таком методе очень важно, чтобы низкочастотные компоненты спектра входного сигнала появлялись на выходе НЧ-фильтра и на выходе линии задержки в одной фазе. Обеспечить идеальное вычитание сигналов невозможно, поэтому появляются два требования: 1) импульсная характеристика исходного НЧ-фильтра должна обладать чётной симметрией (т. е. отвечать требованиям линейности фазы); 2) единицу необходимо добавлять к отсчёту, совпадающему с центром симметрии.

Другой метод получения ВЧ-фильтра — метод обращения АЧХ — проиллюстрирован на **Рис. 14.7**. Импульсная характеристика НЧ-фильтра *(а)* и его АЧХ *(б)* остались теми же, что и в приведённом ранее примере. Импульсная характеристи-

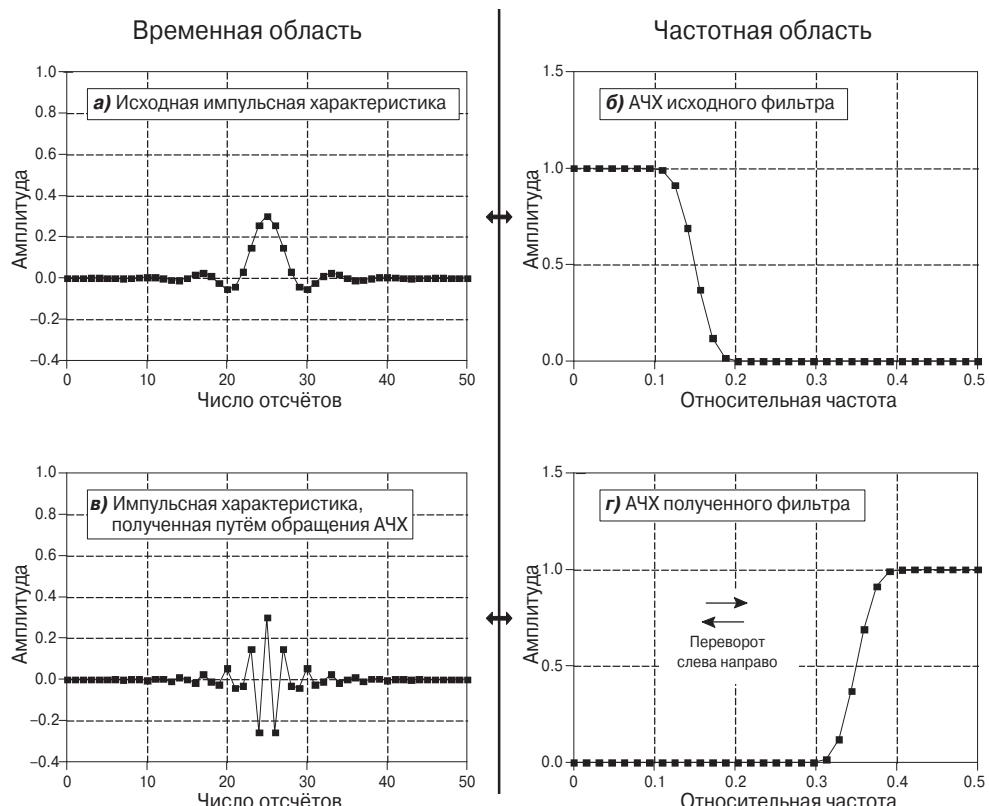


**Рис. 14.6.** Инверсия АЧХ: **а)** входной сигнал  $x[n]$  поступает одновременно на вход двух звеньев —  $h[n]$  и  $\delta[n]$ ; **б)** эти два звена могут быть объединены в одно с импульсной характеристикией  $\delta[n] - h[n]$ . АЧХ такого звена получается в результате инверсии АЧХ исходного НЧ-фильтра  $h[n]$ .

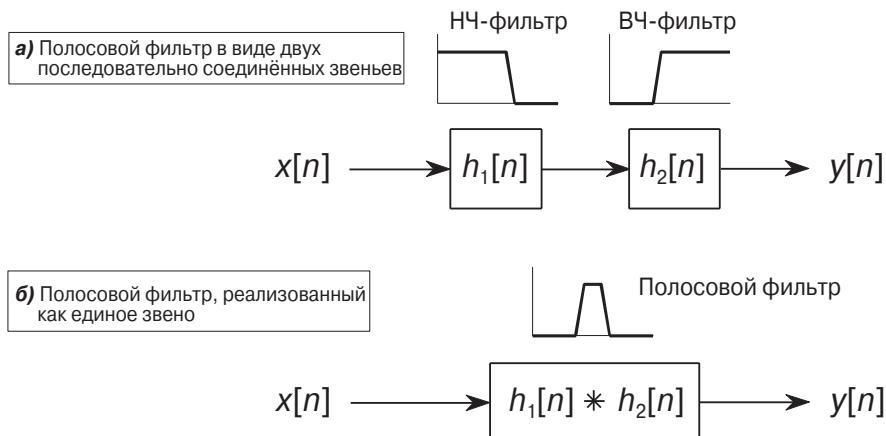
тика ВЧ-фильтра (**в**) образуется путём изменения арифметического знака у каждого второго отсчёта исходной импульсной характеристики НЧ-фильтра (**а**). Из (**г**) следует, что АЧХ ВЧ-фильтра получается поворотом АЧХ НЧ-фильтра слева направо, т. е. точки с абсциссами 0 и 0.5 меняются местами. Так как граничная частота НЧ-фильтра равна 0.15, то граничная частота ВЧ-фильтра равна 0.35.

Изменение арифметического знака каждого второго коэффициента эквивалентно умножению отсчётов импульсной характеристики на отсчёты синусоиды, имеющей относительную частоту 0.5. В Главе 10 было показано, что такая операция соответствует сдвигу в частотной области на 0.5. Представим себе, что на (**б**) изображены не только положительные, но и отрицательные частоты. Диапазон частот  $-0.5...0$  является зеркальным отражением диапазона  $0...0.5$ . Именно эти отрицательные частоты попадают в результат операции обращения АЧХ в тот диапазон, который мы видим на (**г**).

На Рис. 14.8 и 14.9 поясняется процесс получения импульсных характеристик полосовых и режекторных фильтров с помощью комбинации НЧ и ВЧ-фильтров. Основное отличие в способе получения характеристик полосовых и режекторных фильтров состоит в том, что для первых выполняется свёртка импульсных характеристик НЧ и ВЧ-фильтров, тогда как для вторых — простое поэлементное сложение. Это значит, что в первом случае используется последовательное включение фильтров, а во втором — параллельное (Глава 7). Один и тот же фильтр можно получить, используя самые разные комбинации описанных приемов. Например,

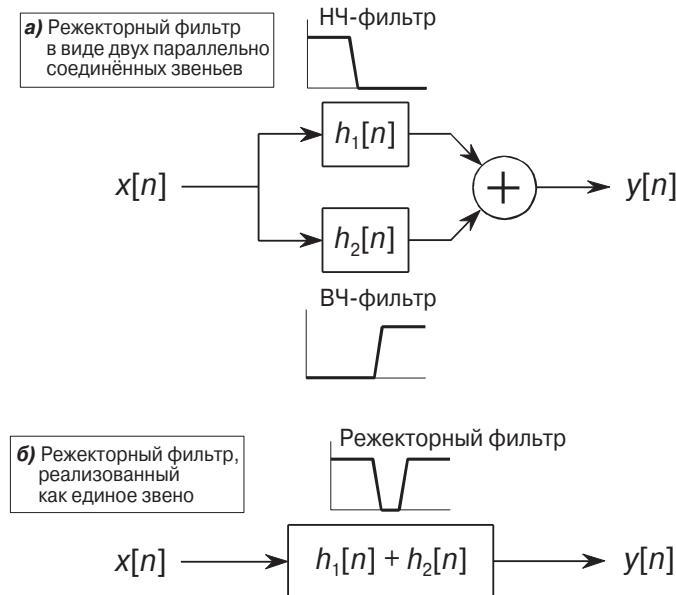


**Рис. 14.7.** Пример обращения АЧХ: **а)** импульсная характеристика НЧ-фильтра; **б)** АЧХ НЧ-фильтра; **в)** импульсная характеристика ВЧ-фильтра, полученная из импульсной характеристики НЧ-фильтра после изменения арифметического знака каждого второго отсчёта импульсной характеристики на противоположный; **г)** АЧХ полученного ВЧ-фильтра является зеркальным отражением АЧХ исходного НЧ-фильтра (повёрнута слева направо).



**Рис. 14.8.** Построение полосовых фильтров: **а)** полосовой фильтр может быть получен последовательным соединением НЧ- и ВЧ-фильтров; **б)** эти звенья можно объединить в одно звено, импульсная характеристика которого образуется путём вычисления свёртки.

можно получить полосовой фильтр, выполнив сначала сложение двух импульсных характеристик, а затем воспользовавшись методом инверсии или обращения АЧХ. Все описанные нами методы работают замечательно, но и для них существуют свои особенности.



**Рис. 14.9.** Построение режекторных фильтров: **а)** режекторный фильтр может быть получен параллельным соединением НЧ и ВЧ-фильтров; **б)** эти два звена можно объединить в одно, импульсная характеристика которого образуется сложением импульсных характеристик.

## 14.6. Классификация фильтров

Цифровые фильтры можно классифицировать по области применения и по внутренней структуре (Табл. 14.1). По области применения все цифровые фильтры можно разделить на три большие группы: *фильтры временной обработки*, *фильтры частотной обработки* и *специальные фильтры*. Как говорилось выше, обработка во временной области используется тогда, когда вся информация содержится непосредственно в форме сигнала (*кодирование во временной области*). Фильтрация во временной области используется при решении таких задач, как сглаживание, устранение постоянной составляющей, формирование огибающей и т. п. Обработка в частотной области применяется, когда вся переносимая сигналом информация содержится в амплитуде, частоте и фазе составляющих сигнала гармонических компонент (*кодирование в частотной области*). Основная задача таких фильтров — разделение частотных диапазонов (частотная селекция сигналов). В ряде приложений возникает необходимость в применении цифровых фильтров с импульсной характеристикой специальной формы. В Главе 17 рассмотрены два примера задач, требующих использования специальных фильтров: коррекция (выравнивание) АЧХ канала связи (или другой искажающей системы) и оптимальная фильтрация.

По внутренней структуре построения различают фильтры, основанные на операции *свёртки* (КИХ-фильтры), и рекурсивные фильтры (БИХ-фильтры). *КИХ-фильтры* характеризуются намного более высокой точностью работы по сравнению с *БИХ-фильтрами*, но значительно уступают последним по быстродействию.

В следующих шести главах цифровые фильтры будут рассмотрены более подробно с учётом предложенной классификации (Табл. 14.1). Сначала речь пойдет о КИХ-фильтрах. *Однородные фильтры* (Глава 15) используются для обработки во временной области; *оконные фильтры* (Глава 16) — для обработки в частотной области; *специальные КИХ-фильтры* (Глава 17) применяются в приложениях, требующих особого подхода. В заключение темы о КИХ-фильтрах, в Главе 18 рассказывается о быстрой свёртке, основанной на алгоритме БПФ. Применение БПФ позволяет значительно сократить вычислительные затраты на обработку сигнала.

Затем мы перейдем к рассмотрению БИХ-фильтров. Для обработки во временной области используются *однополосные рекурсивные фильтры* (Глава 19); в частотной области применяются *фильтры Чебышева* (Глава 20). *Специальные БИХ-фильтры* имеют итеративную структуру, поэтому разумнее оказалось рассмотреть их вместе с другой разновидностью итеративных процедур — *нейронными сетями* (Глава 26).

**Таблица 14.1. Классификация фильтров по рабочей области и структуре построения**

По области применения	По внутренней структуре	
	На основе свёртки (КИХ-фильтры)	С рекурсивной структурой (БИХ-фильтры)
Фильтры временной области (сглаживание, устранение постоянной составляющей)	Однородные фильтры (Глава 15)	Однополосные рекурсивные фильтры (Глава 19)
Фильтры частотной области (частотная селекция)	Оконные фильтры (Глава 16)	Фильтры Чебышева (Глава 20)
Специальные фильтры (коррекция АЧХ, оптимальная фильтрация)	Специальные КИХ-фильтры (Глава 17)	Фильтры с итеративной структурой (Глава 26)

Как видно из Табл. 14.1, КИХ-фильтры (фильтры на основе свёртки) и БИХ-фильтры (рекурсивные фильтры) — два соперничающих между собой класса. В каждом конкретном случае приходится отдавать предпочтение тому или другому. Как сделать правильный выбор? Чтобы ответить на этот вопрос, в Главе 21 мы устроим «соревнование», в результате которого определятся победители среди фильтров, работающих как в частотной, так и во временной области.

# Глава 15

## ОДНОРОДНЫЕ ФИЛЬТРЫ

*Однородные фильтры*, или *фильтры скользящего среднего*, получили наиболее широкое распространение в ЦОС, благодаря простоте их реализации. Кроме того, они являются оптимальными при решении такой классической задачи ЦОС, как подавление аддитивного шума при одновременном сохранении скорости нарастания переходной характеристики. Другими словами, однородные фильтры превосходят все остальные виды цифровых фильтров при обработке информации, представленной во временной области. В то же время эти фильтры оказываются малоэффективны при обработке информации в частотной области, так как не способны разделять между собой сигналы, лежащие в разных диапазонах частот. Имеется несколько модификаций однородных фильтров, в которых частотная избирательность повышается за счёт увеличения времени обработки: фильтр Гаусса, фильтр Блэкмана и фильтр с многократной обработкой.

### 15.1. Однородные нерекурсивные фильтры

*Однородные фильтры* часто называют *фильтрами скользящего среднего*, потому что они основаны на усреднении некоторого множества отсчётов входного сигнала, что выражается следующим разностным уравнением:

$$y[i] = \frac{1}{M} \sum_{j=0}^{M-1} x[i+j]. \quad (15.1)$$

Уравнение однородного нерекурсивного фильтра. Здесь  $x[ ]$  — входной сигнал,  $y[ ]$  — выходной сигнал,  $M$  — число усредняемых отсчётов. В данном уравнении все используемые отсчёты расположены по одну сторону от результирующего выходного отсчёта.

Например, для однородного фильтра 5-го порядка 80-й отсчёт выходного сигнала определяется следующим уравнением:

$$y[80] = \frac{x[80] + x[81] + x[82] + x[83] + x[84]}{5}.$$

Возможен другой вариант построения однородного нерекурсивного фильтра, при котором выбирается равное число отсчётов входного сигнала по обе стороны от текущего выходного отсчёта:

$$y[80] = \frac{x[78] + x[79] + x[80] + x[81] + x[82]}{5}.$$

Такой вариант усреднения связан с изменением пределов вычисления суммы в выражении (15.1). Вместо  $j = 0 \dots (M - 1)$  используем новые номера отсчётов входного сигнала:  $j = -(M - 1)/2 \dots (M - 1)/2$ . В результате при вычислении скользящего среднего по 11 отсчётам входного сигнала индекс  $j$  может меняться либо от 0 до 10 (одностороннее усреднение), либо от  $-5$  до  $5$  (симметричное усреднение). При симметричном усреднении необходимо, чтобы  $M$  было нечётным числом. Несмотря на то что программа для алгоритма одностороннего усреднения несомненно проще, данный алгоритм приводит к возникновению относительно-го сдвига между входным и выходным сигналами.

Запомните, что однородные фильтры — это фильтры с самой простой *импульсной характеристикой*. Например, однородный нерекурсивный фильтр 5-го порядка имеет импульсную характеристику вида  $\{..., 0, 0, 1/5, 1/5, 1/5, 1/5, 1/5, 0, 0, ...\}$ . То есть в основе однородных нерекурсивных фильтров лежит алгоритм свёртки входного сигнала с прямоугольным импульсом единичной площади. Однородный нерекурсивный фильтр реализуется в **Программе 15.1**.

### **Программа 15.1**

```

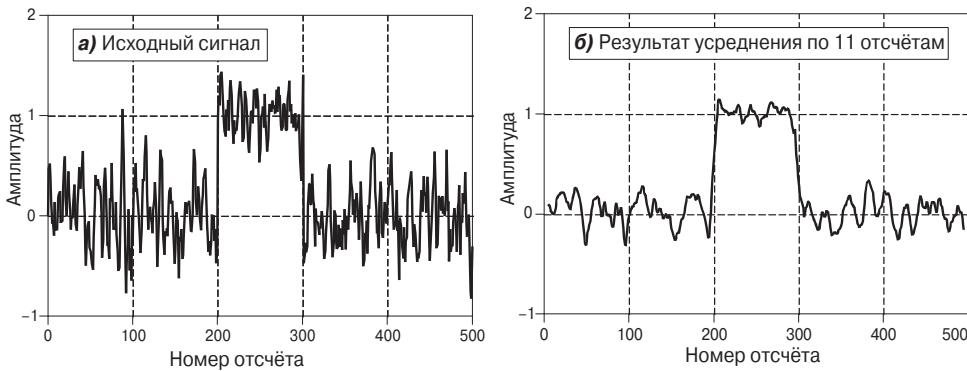
100 'ОДНОРОДНЫЙ НЕРЕКУРСИВНЫЙ ФИЛЬТР (ФИЛЬТР СКОЛЬЗЯЩЕГО СРЕДНЕГО)
110 'Данная программа предназначена для обработки 5000 отсчётов
120 'входного сигнала однородным фильтром 101-го порядка.
130 '
140 DIM X[5000]           'X[ ] - массив отсчётов входного сигнала
150 DIM Y[5000]           'Y[ ] - массив отсчётов выходного сигнала
160 '
170 GOSUB XXXX            'Некоторая подпрограмма загрузки данных в X[ ]
180 '
190 FOR I% = 50 TO 4949   'Цикл по числу отсчётов выходного сигнала
200 Y[I%] = 0              'Предварительное обнуление
210 FOR J% = -50 TO 50    'Цикл для вычисления очередного отсчёта
220 Y[I%] = Y[I%] + X[I%+J%]
230 NEXT J%
240 Y[I%] = Y[I%]/101    'Завершение усреднения - операция деления
250 NEXT I%
260 '
270 END

```

## **15.2. Переходная характеристика и подавление шума**

Очень часто учёные и инженеры, используя однородные фильтры, чувствуют некоторую неуверенность. Дело в том, что однородные фильтры, как самые простые, всегда стараются применить в первую очередь. И даже когда поставленная задача уже полностью решена, все равно мучает вопрос: а может стоит перейти к более сложным типам фильтров? Ситуация довольно забавна. Ведь однородные фильтры не только находят очень удачное применение во множестве практических приложений, но и являются оптимальными при решении такой распространённой задачи, как подавление *белого шума* при одновременном сохранении *занятой скорости нарастания переходной характеристики*.

Функционирование однородного фильтра иллюстрирует Рис. 15.1. На вход фильтра поступает сигнал, представляющий собой прямоугольный импульс, скрытый в шумах (**а**). В результате фильтрации (**б** и **в**) мощность шума заметно уменьшается (положительный эффект), но вместе с тем снижается крутизна фронтов импульса (отрицательный эффект). Среди всех линейных фильтров однородные фильтры характеризуются наиболееенным подавлением шума при заданной крутизне фронтов импульса. Коэффициент подавления шума равен квадратному корню из числа отсчётов, участвующих в усреднении. Так, фильтр 100-го порядка позволяет уменьшить шум в 10 раз.



**Рис. 15.1.** Использование однородного фильтра: **а)** прямоугольный импульс, скрытый в шумах; **б** и **в)** сигналы, полученные на выходе однородных фильтров 11-го и 51-го порядков соответственно. Чем больше отсчетов входного сигнала участвуют в операции усреднения, тем меньше оказывается мощность шума в выходном сигнале, но при этом более пологими становятся фронты импульса. Однородный фильтр является оптимальным при поиске компромисса в данной ситуации. Он обеспечивает максимальное подавление шума при заданной крутизне фронтов импульса.



Чтобы понять, почему однородный фильтр обеспечивает наилучшее подавление шума, присутствующего во входном сигнале, попытаемся рассчитать фильтр, задав ограничение по скорости нарастания переходного процесса. Пусть протяжённость переходного процесса составляет 11 отсчётов. Значит, фильтр имеет 11 весовых коэффициентов. Задача оптимизации заключается в ответе на вопрос: как правильно выбрать значения 11 весовых коэффициентов, чтобы мощность шума на выходе фильтра оказалась минимальной? Так как шумовая помеха — это случайный процесс, то ни один из отсчётов не обладает какими-либо уникальными свойствами по отношению к остальным, т. е. каждый отсчёт подвержен влиянию шума в той же степени, что и окружающие его отсчёты. Поэтому не имеет

смысла отдавать предпочтение какому-либо конкретному отсчёту входного сигнала, увеличивая с этой целью один из коэффициентов фильтра. Наименьший шум получается при выравнивании значений всех весовых коэффициентов, а значит, при использовании однородного фильтра. Ниже в этой главе мы рассмотрим фильтры, которые практически не уступают по своим характеристикам однородному. Однако следует помнить, что фильтра, лучшего, чем однородный, с точки зрения подавления шума не существует.

## 15.3. Частотная характеристика

На Рис. 15.2 показана АЧХ однородного фильтра. Математически она выражается преобразованием Фурье от прямоугольного импульса (Глава 11):

$$H(f) = \frac{\sin(\pi f M)}{M \sin(\pi f)}. \quad (15.2)$$

АЧХ однородного фильтра  $M$ -го порядка.

Нормированная частота  $f$  меняется от 0 до 0.5. При  $f = 0$ ,  $H(f) = 1$ .

Судя по Рис. 15.2, спад АЧХ очень медленный, а затухание в зоне непрозрачности слишком далеко от идеала. Совершенно очевидно, что однородный фильтр не может быть использован для отделения одного частотного диапазона от другого. Не забывайте, что хорошее качество работы во временной области обычно сопровождается слабым разрешением в частотной области и наоборот. Проще говоря, однородный фильтр является самым лучшим *сглаживающим фильтром* (во временной области), но вместе с тем и самым худшим *НЧ-фильтром* (в частотной области).

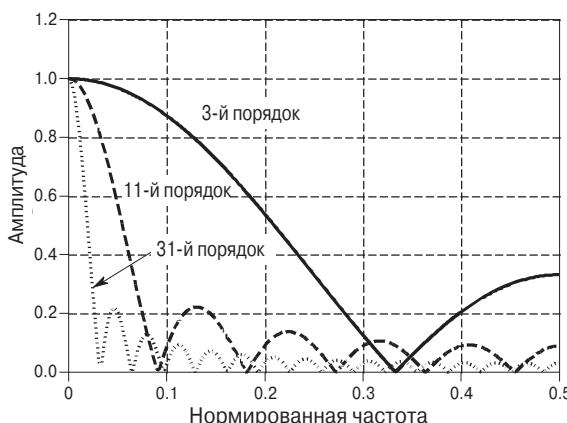


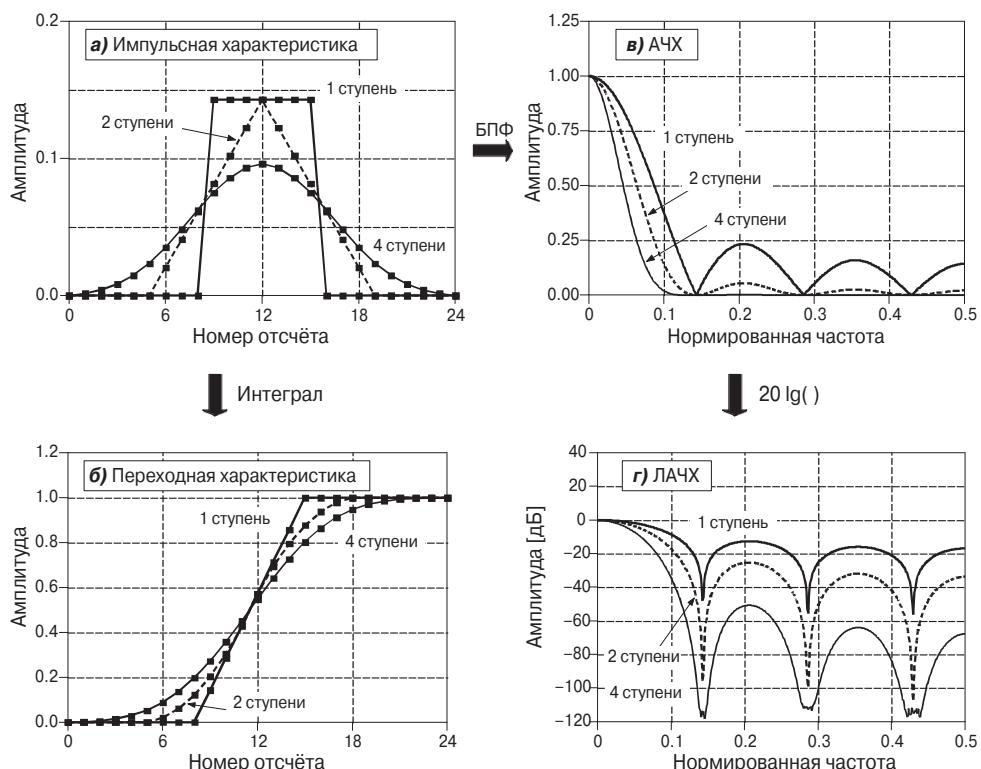
Рис. 15.2. АЧХ однородного фильтра. Однородный фильтр является очень слабым НЧ-фильтром: АЧХ спадает медленно, затухание в зоне непрозрачности очень мало. При построении графиков использовалось выражение (15.2).

## 15.4. Модифицированные однородные фильтры

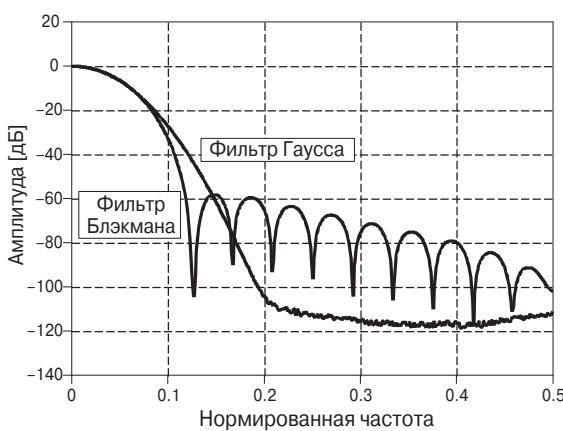
В своей профессиональной деятельности разработчикам цифровых фильтров приходится иметь дело с информацией, заложенной либо во временной, либо в частотной области (*кодирование во временной и частотной областях*), и никогда информация не размещается в этих двух областях одновременно. Однако встречаются практические приложения, в которых сигнал все же необходимо рассматривать сразу в двух областях. Одним из таких «неприятных» примеров служит телевизионный сигнал. Видеоинформация кодируется во временной области, так как распределение яркости по элементам изображения закладывается в форму передаваемого сигнала. Однако в процессе передачи видеосигнала огромное значение имеет его частотная структура: диапазон занимаемых частот, разделение частотных диапазонов для изображений и звука, подавление и восстановление постоянной составляющей и т. п. Другим примером является учёт электромагнитных помех, которые удобнее анализировать в частотной, а не во временной области. Типовыми помехами, например, для обычных научных и инженерных измерений, являются: на частоте 60 Гц — помеха от сетей электропитания; на частоте 30 кГц — от импульсных источников питания; на частоте 1320 кГц — со стороны радиостанций FM-диапазона. *Модифицированные однородные фильтры* отличаются от обычных более чёткой частотной локализацией и поэтому могут более эффективно использоваться в подобных приложениях.

*Фильтры с многократной обработкой* представляют собой последовательное соединение нескольких однородных фильтров, т. е. одна и та же процедура однородной фильтрации повторяется в них последовательно несколько раз. На Рис. 15.3а показаны *импульсные характеристики*, построенные для фильтров с одной, двумя и четырьмя ступенями обработки (каждая ступень — однородный фильтр 7-го порядка). При наличии двух ступеней фильтр имеет треугольную импульсную характеристику, поскольку треугольная характеристика может быть представлена как результат свёртки двух прямоугольных характеристик. При увеличении количества ступеней до четырёх и более форма импульсной характеристики приближается к форме *кривой Гаусса* (согласно *центральной предельной теореме*). По *переходным характеристикам* (б) видно, что для фильтров с многократной обработкой характерен *s*-образный переходный процесс в отличие от однородного фильтра, которому соответствует линейно нарастающая реакция. Частотные характеристики, приведённые на (в и г), получены в результате многократного умножения АЧХ выражения (15.2) на саму себя. Умножение производится столько раз, сколько последовательных звеньев содержится в фильтре. Такой подход оказывается возможным в силу того, что каждой операции свёртки во временной области соответствует операция умножения в частотной области.

На Рис. 15.4 изображены ЛАЧХ для ещё двух модификаций однородных фильтров. Как уже говорилось выше (Глава 11), если импульсная характеристика фильтра имеет вид кривой Гаусса, то и его АЧХ является также гауссовой. Импульсные характеристики систем как естественного, так и искусственного происхождения очень часто описываются кривой Гаусса. К примеру, если на вход длинного оптического кабеля подать короткий импульс света, то на выходе кабеля будет принят колоколообразный импульс, форма которого описывается функцией Гаусса. Та-



**Рис. 15.3.** Характеристики однородных фильтров с многократной обработкой: **а)** импульсные характеристики фильтров с многократной обработкой, полученные в результате последовательного соединения однородных фильтров седьмого порядка; **б)** переходные характеристики; **в)** АЧХ; **г)** ЛАЧХ.



**Рис. 15.4.** ЛАЧХ фильтров Блэкмана и Гаусса. Оба фильтра имеют более сильное затухание в зоне непрозрачности, чем однородный фильтр. Хотя это не даёт им никакого преимущества при решении задачи подавления аддитивного шума, решаемой во временной области, появляется возможность применять эти фильтры при частотно-временной обработке. Недостатком фильтров является то, что они основаны на алгоритме свёртки, требующем большого объёма вычислительных затрат.

кой эффект связан с различием длин траекторий распространения фотонов. Уникальные свойства гауссовой импульсной характеристики широко используются в системах *обработки изображений* при реализации алгоритма вычисления *двумерной свёртки* (Глава 24). На Рис. 15.4 рассматривается фильтр с *окном Блэкмана* (термин «окно» подразумевает принадлежность фильтра Блэкмана к классу оконных фильтров.) Подробнее об оконной функции Блэкмана будет сказано в Главе 16 ((16.2) и Рис. 16.2). По своему виду эта функция близка к гауссовой.

Какие же преимущества характерны для рассмотренных выше модификаций однородных фильтров? Основных преимуществ — три. Во-первых, модифицированные однородные фильтры позволяют достичь большего подавления в зоне непрозрачности, чем обычные однородные фильтры. Во-вторых, их импульсная характеристика на краях плавно сходится к нулю. Как вы помните, каждый отсчёт выходного сигнала определяется взвешенной суммой группы входных отсчётов. Поэтому сходимость к нулю импульсной характеристики означает, что отсчёты, близкие к центру такой группы, оказывают более сильное влияние на формирование выходного отсчёта. В-третьих, переходные процессы модифицированных фильтров оказываются более плавными и не имеют резких изгибов. Последние два преимущества менее важны, хотя, надо заметить, существует ряд приложений, в которых они оказываются очень полезными.

Модифицированные однородные фильтры и обычные однородные фильтры проявляют приблизительно равные возможности при решении задачи подавления шума при одновременном сохранении скорости нарастания переходной характеристики. Точное решение вопроса, какой из фильтров лучше, связано с тем, каким образом мы измеряем длительность *переходного процесса*. Если считать, что переходный процесс — это время нарастания переходной характеристики от 0 до 100% её установившегося значения, то преимущество оказывается на стороне обычного однородного фильтра, как было показано выше. Если длительность переходного процесса измерять интервалом времени, в течение которого переходная характеристика нарастает от 10 до 90% своей установившейся величины, то фильтр Блэкмана несомненно превосходит однородный фильтр. Так что спор по этому вопросу носит исключительно теоретический характер.

Наиболее заметным является различие в объёмах *вычислительных затрат*. Рекурсивные однородные фильтры, речь о которых пойдет далее, могут с лёгкостью быть реализованы на обычном персональном компьютере. Это самые экономные по вычислительным затратам фильтры. Затраты на реализацию фильтров с многократной обработкой несколько больше, но всё же невелики. А вот фильтры Гаусса и Блэкмана требуют очень большого объёма вычислений, так как основаны на операции свёртки. Если считать, что умножение выполняется приблизительно в 10 раз дольше, чем сложение, то *фильтр Гаусса* 100-го порядка требует примерно в 1000 раз больше вычислений, чем рекурсивный однородный фильтр.

## 15.5. Однородные рекурсивные фильтры

Однородные фильтры имеют одно важное преимущество перед всеми другими: для них существует самый простой алгоритм реализации, какой только можно придумать. Чтобы понять этот алгоритм, рассмотрим следующий пример.

Пусть сигнал  $x[ ]$  поступает на вход однородного нерекурсивного фильтра седьмого порядка, а  $y[ ]$  — реакция этого фильтра. Тогда отсчёты  $y[50]$  и  $y[51]$  выражаются соотношениями

$$y[50] = x[47] + x[48] + x[49] + x[50] + x[51] + x[52] + x[53]$$

и

$$y[51] = x[48] + x[49] + x[50] + x[51] + x[52] + x[53] + x[54].$$

Эти выражения во многом повторяют друг друга, поскольку отсчёты 48...53 используются для вычисления и  $y[50]$ , и  $y[51]$ . Если  $y[50]$  уже найден, то существует более эффективный способ вычислить  $y[51]$ :

$$y[51] = y[50] + x[54] - x[47].$$

Если найдено значение  $y[50]$ , то  $y[51]$  проще всего выразить через него. То же самое относится и к последующим отсчётам выходной последовательности. Если найден один отсчёт выходного сигнала  $y[ ]$ , каждый последующий может быть найден в результате одной операции сложения и одной операции вычитания:

$$y[i] = y[i-1] + x[i+p] - x[i-q],$$

где  $p = (M-1)/2$ ,

$$q = p + 1.$$

Однородный рекурсивный фильтр. Здесь  $x[ ]$  — входной сигнал,  $y[ ]$  — выходной сигнал,  $M$  — число (нечётное) усредняемых отсчётов. Перед началом использования этой формулы необходимо вычислить первый отсчёт выходной последовательности простым суммированием.

Как видим, этот алгоритм использует два источника данных: отсчёты входной последовательности и найденные ранее отсчёты выходной последовательности. Алгоритм, в котором найденные на предшествующих итерациях отсчёты выходной последовательности также используются для вычисления последующих отсчётов выходного сигнала, называется *рекурсивным*. В Главе 19 подробно рассмотрено множество рекурсивных фильтров. Однако однородные рекурсивные фильтры существенно отличаются от остальных рекурсивных фильтров. В частности, *импульсная характеристика* однородных фильтров имеет вид прямоугольного импульса, т. е. это *фильтры с конечной импульсной характеристикой*, или *КИХ-фильтры*, в то время как *рекурсивные фильтры* образуют класс *БИХ-фильтров*, а их импульсная характеристика имеет бесконечную длину и представляет собой набором синусоид и экспонент.

Превосходство по вычислительным затратам однородных фильтров перед другими объясняется несколькими причинами. Первая: требуется всего две операции на обработку каждого входного отсчёта. Вторая: используются только сложение и вычитание, тогда как большинство фильтров использует трудоёмкое умножение. Третья: процедура вычисления индексов в выражении (15.3) очень проста: достаточно выполнить сложение и вычитание предварительно найденных констант ( $p$  и  $q$ ). Четвёртая: все операции можно выполнять полностью в целочисленной арифметике. Выигрыш по вычислительным затратам зависит от выбранной аппаратной платформы, и в большинстве случаев эффективность вычислений в *формате с фиксированной точкой* оказывается на порядок выше, чем при выполнении тех же вычислений в *формате с плавающей точкой*.

Удивительно, но целочисленная арифметика позволяет не только сократить объём вычислительных затрат, но и повышает точность работы алгоритма фильтрации. При вычислениях в формате с плавающей точкой *ошибка округления*, если её не учитывать, может привести к весьма нежелательным последствиям. Предположим, что фильтр обработал 10 000 отсчётов входного сигнала. Значит, следующий отсчёт будет вычислен с ошибкой, накопленной от предшествующих 10 000 операций сложения и 10 000 операций вычитания. Эта ошибка проявляется в форме «дрейфа» выходного сигнала фильтра. Вычисления в рамках целочисленной арифметики, как правило, протекают без округлений и связанных с этим ошибок и не создают подобных трудностей. Но иногда избежать вычислений в формате с плавающей точкой не удается, тогда для решения проблемы «дрейфа» выходного сигнала приходится использовать двойную точность вычислений, что проиллюстрировано в **Программе 15.2**.

### **Программа 15.2**

```

100 ' ОДНОРОДНЫЙ РЕКУРСИВНЫЙ ФИЛЬТР (ФИЛЬТР СКОЛЬЗЯЩЕГО СРЕДНЕГО)
110 'Данная программа предназначена для обработки 5000 отсчётов
120 'входного сигнала однородным фильтром 101-го порядка.
130 'Для устранения дрейфа накопление происходит с двойной точностью.
140 '
150 DIM X[5000]           'X[ ] - массив отсчётов входного сигнала
160 DIM Y[5000]           'Y[ ] - массив отсчётов выходного сигнала
170 DEFDBL ACC            'Объявление переменной ACC с двойной точностью
180 '
190 GOSUB XXXX             'Некоторая подпрограмма загрузки данных в X[ ]
200 '
210 ACC = 0                 'Вычисление Y[50] усреднением отсчётов X[0]...X[100]
220 FOR I% = 0 TO 100
230 ACC = ACC + X[I%]
240 NEXT I%
250 Y[50] = ACC/101
260 '                         'Однородный рекурсивный фильтр (выражение 15.3)
270 FOR I% = 51 TO 4949
280 ACC = ACC + X[I%+50] - X[I%-51]
290 Y[I%] = ACC/101
300 NEXT I%
310 '
320 END

```

# Глава 16

## ОКОННЫЕ ФИЛЬТРЫ

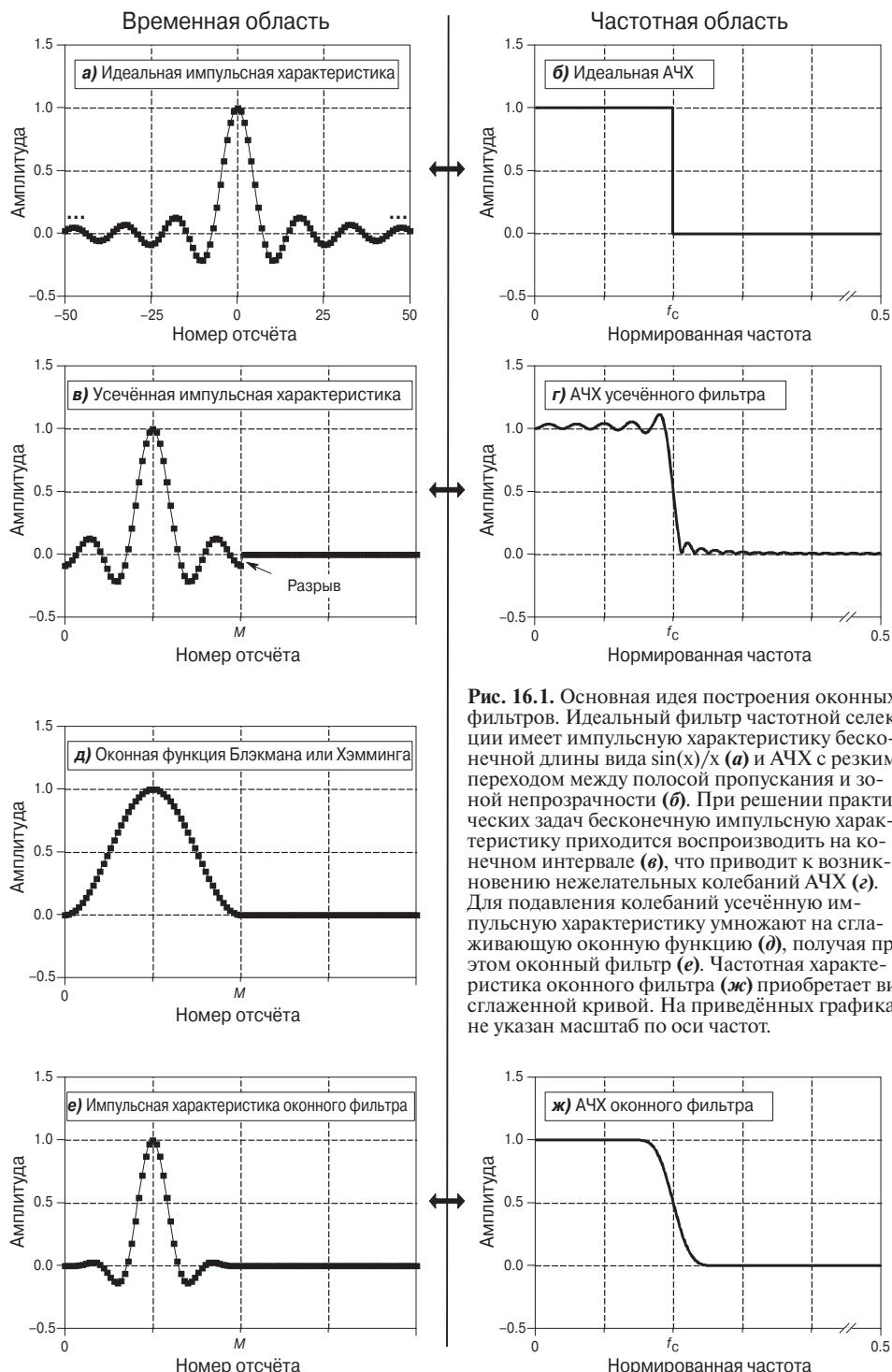
*Оконные фильтры* используются для решения задач частотной селекции сигналов. Эти фильтры обладают высокой устойчивостью, стабильностью характеристик и оказываются чрезвычайно эффективны для указанного класса задач. Высокая эффективность фильтров в частотной области сопровождается ухудшением временных характеристик — увеличиваются перерегулирование и колебательность переходного процесса. В основе работы оконных фильтров лежит операция *свёртки*. Поэтому их программная реализация оказывается довольно проста, но они требовательны к *вычислительным затратам*. В Главе 18 будет показано, каким образом можно значительно уменьшить вычислительную сложность оконных фильтров, воспользовавшись алгоритмом БПФ.

### 16.1. Принципы построения оконных фильтров

Основная идея построения *оконных фильтров* поясняется на Рис. 16.1. Идеальный НЧ-фильтр должен иметь единичный коэффициент передачи во всём диапазоне частот ниже *частоты среза*  $f_C$  и полностью подавлять все остальные частотные компоненты. Изображённая на (б) АЧХ сохраняет постоянное значение на протяжении всей полосы пропускания и обладает бесконечно большим затуханием в зоне непрозрачности фильтра. Разделяющая эти две области переходная зона бесконечно мала. Идеальная импульсная характеристика, полученная в результате выполнения обратного преобразования Фурье идеальной частотной характеристики, описывается функцией вида  $\sin(x)/x$  (а). Согласно выражению (11.4), идеальная импульсная характеристика может быть представлена в общем виде как

$$h[i] = \frac{\sin(2\pi f_C i)}{\pi i}.$$

Для того чтобы создать идеальный НЧ-фильтр, необходимо выполнить алгоритм свёртки входного сигнала с соответствующей ему импульсной характеристикой. Однако на практике данный алгоритм не может быть реализован, поскольку колебания функции  $\sin(x)/x$  не затухают до нулевого значения и продолжаются как в сторону положительных, так и в сторону отрицательных частот бесконечно долго. С точки зрения математики, построение идеального фильтра не представляет сложности, но на практике построение фильтра с бесконечным числом весовых коэффициентов оказывается неразрешимой задачей.



**Рис. 16.1.** Основная идея построения оконных фильтров. Идеальный фильтр частотной селекции имеет импульсную характеристику бесконечной длины вида  $\sin(x)/x$  (а) и АЧХ с резким переходом между полосой пропускания и зоной непрозрачности (б). При решении практических задач бесконечную импульсную характеристику приходится воспроизводить на конечном интервале (в), что приводит к возникновению нежелательных колебаний АЧХ (г). Для подавления колебаний усечённую импульсную характеристику умножают на сглаживающую оконную функцию (д), получая при этом оконный фильтр (е). Частотная характеристика оконного фильтра (ж) приобретает вид слаженной кривой. На приведённых графиках не указан масштаб по оси частот.

Обойти указанные трудности позволяют следующие два преобразования функции  $\sin(x)/x$  (**а**). На первом шаге выполняется усечение исходной функции до  $(M+1)$  отсчётов по оси абсцисс, расположенных симметрично относительно главного (основного) лепестка ( $M$  — чётное число). Всем остальным отсчётам присваиваются нулевые значения, т. е. они просто отбрасываются. На втором шаге полученная последовательность сдвигается вправо так, чтобы номера отсчётов изменялись в пределах  $0\dots M$ . В результате индексы всех весовых коэффициентов становятся неотрицательными. Цель такого сдвига импульсной характеристики фильтра на  $M/2$  отсчётов состоит в том, чтобы задержать выходной сигнал фильтра на то же число отсчётов. В результате преобразований получаем импульсную характеристику, показанную на (**в**).

Выполненные преобразования привели к тому, что фильтр уже не является идеальным и имеет другую АЧХ. Можно найти АЧХ полученного фильтра (**г**), выполнив преобразование Фурье его импульсной характеристики. Как же сильно она изменилась! Появились большие колебания в полосе пропускания фильтра, а подавление в зоне непрозрачности стало чрезвычайно малым. Эти неприятные последствия вызваны наличием разрывов на краях усечённой импульсной характеристики (*эффект Гиббса*, описанный в Главе 11). Увеличение длины импульсной характеристики не решает проблемы, так как не устраняет разрывов.

К счастью, существует простой метод компенсации возникших искажений. На (**д**) показана плавно сходящаяся к нулю кривая, названная *окном Блэкмана*. Импульсная характеристика оконного фильтра (**е**) представляет собой результат поэлементного умножения усечённой идеальной импульсной характеристики (**в**) на оконную функцию (**д**). Главной целью использования оконной функции является сглаживание усечённой импульсной характеристики на краях. Одновременно улучшаются и частотные свойства фильтра: АЧХ в полосе пропускания снова становится плоской, а затухание в зоне непрозрачности значительно увеличивается (**ж**).

Широкое применение получили несколько оконных функций, появившихся в 50-х годах прошлого века и названных по именам их создателей. Наиболее распространены две из них — *окно Хэмминга* (**16.1**) и *окно Блэкмана* (**16.2**).

$$w[i] = 0.54 - 0.46 \cos(2\pi i / M). \quad (16.1)$$

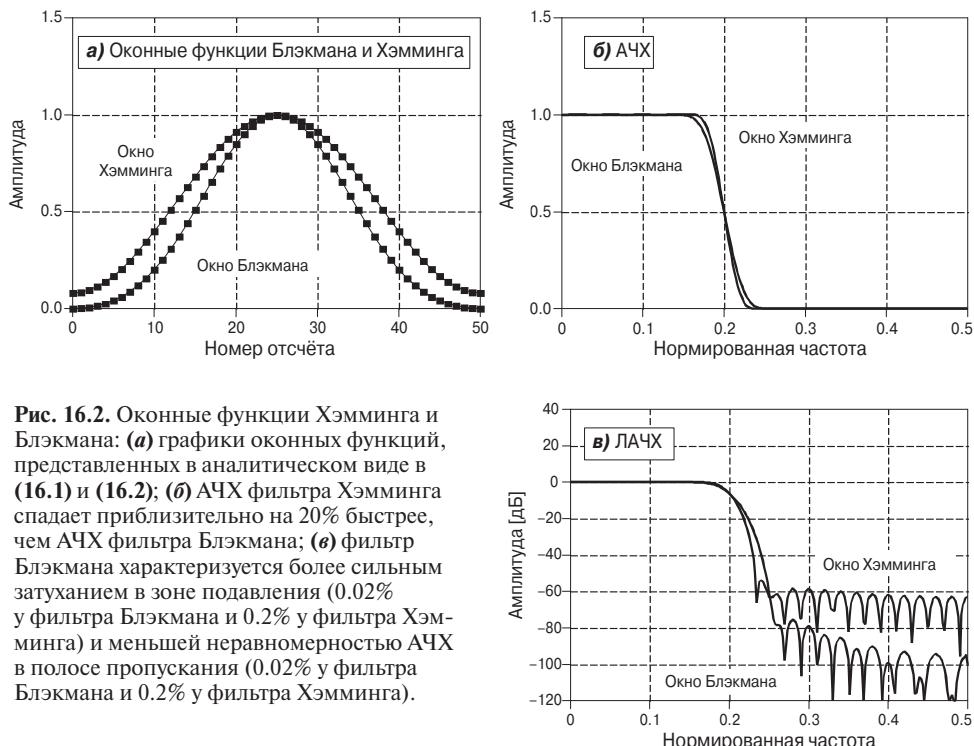
Оконная функция Хэмминга. Данное соотношение справедливо в диапазоне  $0\dots M$ ; вне этого диапазона значения нулевые.

$$w[i] = 0.42 - 0.5 \cos(2\pi i / M) + 0.08 \cos(4\pi i / M). \quad (16.2)$$

Оконная функция Блэкмана. Данное соотношение справедливо в диапазоне  $0\dots M$ ; вне этого диапазона значения нулевые.

На Рис. 16.2а изображены обе оконные функции для  $M = 50$  (т. е. для фильтра 51-го порядка). Ответ на вопрос о том, какая из двух функций лучше, зависит от конкретно решаемой задачи. Как видно на (**б**), АЧХ оконного фильтра Хэмминга спадает в переходной зоне приблизительно на 20% быстрее, чем АЧХ фильтра Блэкмана. Но, судя по (**в**), фильтр Блэкмана имеет более сильное затухание в зоне непрозрачности. В данном случае коэффициент передачи фильтра Блэкмана в зоне непрозрачности составляет  $-74$  дБ ( $-0.02\%$ ), а фильтра Хэмминга — всего

лишь  $-53$  дБ ( $-0.2\%$ ). Неразличимый на приведенных графиках уровень неравномерности АЧХ в полосе пропускания равен приблизительно  $0.02\%$  у фильтра Блэкмана и  $0.2\%$  у фильтра Хэмминга. Чаще всего предпочтение отдают фильтру Блэкмана, так как медленный спад его АЧХ представляется менее серьёзным недостатком, чем слабое затухание в диапазоне подавляемых фильтром частот.



**Рис. 16.2.** Окно Хэмминга и Блэкмана: **(а)** графики оконных функций, представленных в аналитическом виде в (16.1) и (16.2); **(б)** АЧХ фильтра Хэмминга спадает приблизительно на  $20\%$  быстрее, чем АЧХ фильтра Блэкмана; **(в)** фильтр Блэкмана характеризуется более сильным затуханием в зоне подавления ( $0.02\%$  у фильтра Блэкмана и  $0.2\%$  у фильтра Хэмминга) и меньшей неравномерностью АЧХ в полосе пропускания ( $0.02\%$  у фильтра Блэкмана и  $0.2\%$  у фильтра Хэмминга).

Имеются и другие оконные функции, о существовании которых необходимо знать. Приведем ряд функций, уступающих окнам Блэкмана и Хэмминга, но всё же иногда используемых на практике. Треугольная оконная функция получила название *окна Бартлетта*. *Окно Хеннинга*, которое также называют *функцией приподнятого косинуса*, описывается выражением  $w[i] = 0.5 - 0.5 \cos(2\pi i/M)$ . Соответствующие этим двум оконным функциям фильтры практически не отличаются по скорости спада АЧХ в переходной зоне от фильтра Хэмминга, но заметно уступают ему по затуханию (коэффициент передачи в зоне подавления равен  $-25$  дБ, или  $5.6\%$ , у фильтра Бартлетта и  $-44$  дБ, или  $0.63\%$ , у фильтра Хеннинга). В научной литературе часто встречается понятие *прямоугольного окна*. Прямоугольная оконная функция не оказывает никакого формирующего действия, оставляя усеченную импульсную характеристику **(в)** без изменений. Скорость спада АЧХ прямоугольного оконного фильтра в  $2.5$  раза выше, чем у фильтра Блэкмана, однако коэффициент передачи в зоне непрозрачности равен всего лишь  $-21$  дБ ( $8.9\%$ ).

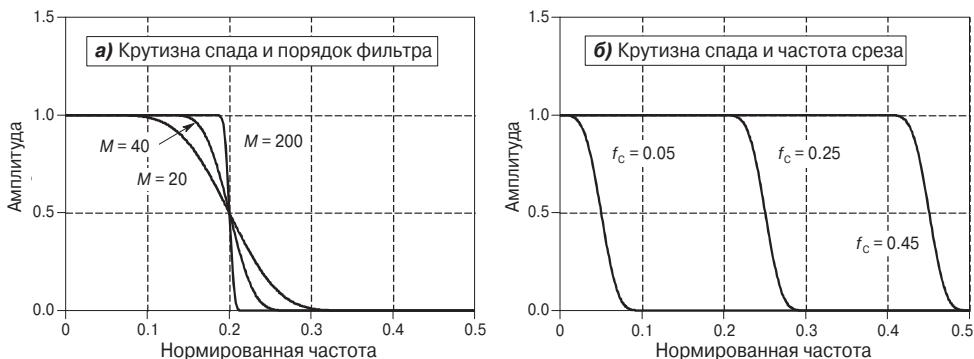
## 16.2. Расчёт оконного фильтра

Для расчёта оконного фильтра необходимо предварительно задать два параметра — частоту среза  $f_c$  и порядок фильтра  $M$ . Частота среза может быть задана отношением к частоте дискретизации. В этом случае она выбирается в диапазоне 0...0.5. Величину  $M$  связывает с крутизной АЧХ следующая приближённая формула:

$$M \approx \frac{4}{BW}. \quad (16.3)$$

Связь между порядком фильтра и крутизной АЧХ. Величина  $M$  определяет ширину переходной зоны фильтра  $BW$ . Равенство является приближённым, так как крутизна спада зависит от выбора оконной функции.

Здесь  $BW$  (bandwidth) — ширина переходной зоны фильтра, которая определяется как диапазон, разделяющий область с близким к единице коэффициентом передачи, и область, в которой коэффициент передачи близок к нулю (границы областей соответствуют уровням 99 и 1% от максимального коэффициента усиления). Ширину переходной зоны можно выразить через приведенную частоту (т.е. через отношение к частоте дискретизации). В этом случае допустимые значения ширины переходной зоны лежат в интервале 0...0.5. На Рис. 16.3а поясняется смысл выражения (16.3). Три кривые на графике построены для  $M = 20, 40$  и  $200$ . Из выражения (16.3) следует, что для этих значений  $BW = 0.2, 0.1$  и  $0.02$  соответственно. На Рис. 16.3б показано, что крутизна АЧХ не зависит от выбора частоты среза.



**Рис. 16.3.** Зависимость крутизны спада АЧХ от порядка оконного фильтра: (а) оконные фильтры порядков  $M = 20, 40$  и  $200$  имеют ширину переходной зоны  $BW = 0.2, 0.1$  и  $0.02$  соответственно; (б) крутизна спада не зависит от выбора граничной частоты (здесь  $M = 60$ ).

Так как объём вычислительных затрат, необходимый для выполнения операции свёртки, пропорционален порядку фильтра  $M$ , то в выражении (16.3) отражается противоречие между стремлением понизить вычислительную сложность алгоритма и желанием уменьшить ширину переходной зоны фильтра  $BW$ . Например, чтобы повысить крутизну спада АЧХ фильтра Блэкмана на те самые 20%, которые он проигрывает фильтру Хэмминга, достаточно увеличить количество его весовых коэффициентов на 20%, что приведёт в свою очередь к такому же увеличению вычислительных затрат. Но поскольку реализация оконных фильтров

тров, основанных на выполнении свёртки, и без того требует большого объёма вычислений, то это очень существенный недостаток.

На Рис. 16.3б верхняя граничная частота определяется по уровню половинной амплитуды. Почему вместо принятого для аналоговых фильтров уровня 0.707 ( $-3$  дБ) в данном случае выбран уровень 0.5? Это объясняется наличием у оконных фильтров строгой взаимосвязи между полосой пропускания и зоной непрозрачности. Например, для рассмотренного ранее фильтра Хэмминга уровень неравномерности в полосе пропускания равен 0.2% и совпадает по величине с уровнем подавления в зоне непрозрачности, который тоже равен 0.2%. Другие цифровые фильтры не имеют такой закономерности, и поэтому для них нет никакого преимущества в определении верхней частоты среза фильтра по уровню половинной амплитуды. Как будет показано далее, описанная взаимосвязь поведения АЧХ в полосе пропускания и в зоне непрозрачности обуславливает отличную возможность применения метода инверсии АЧХ при расчёте оконных фильтров.

После того как заданы параметры  $f_C$  и  $M$ , весовые коэффициенты фильтра могут быть найдены по формуле

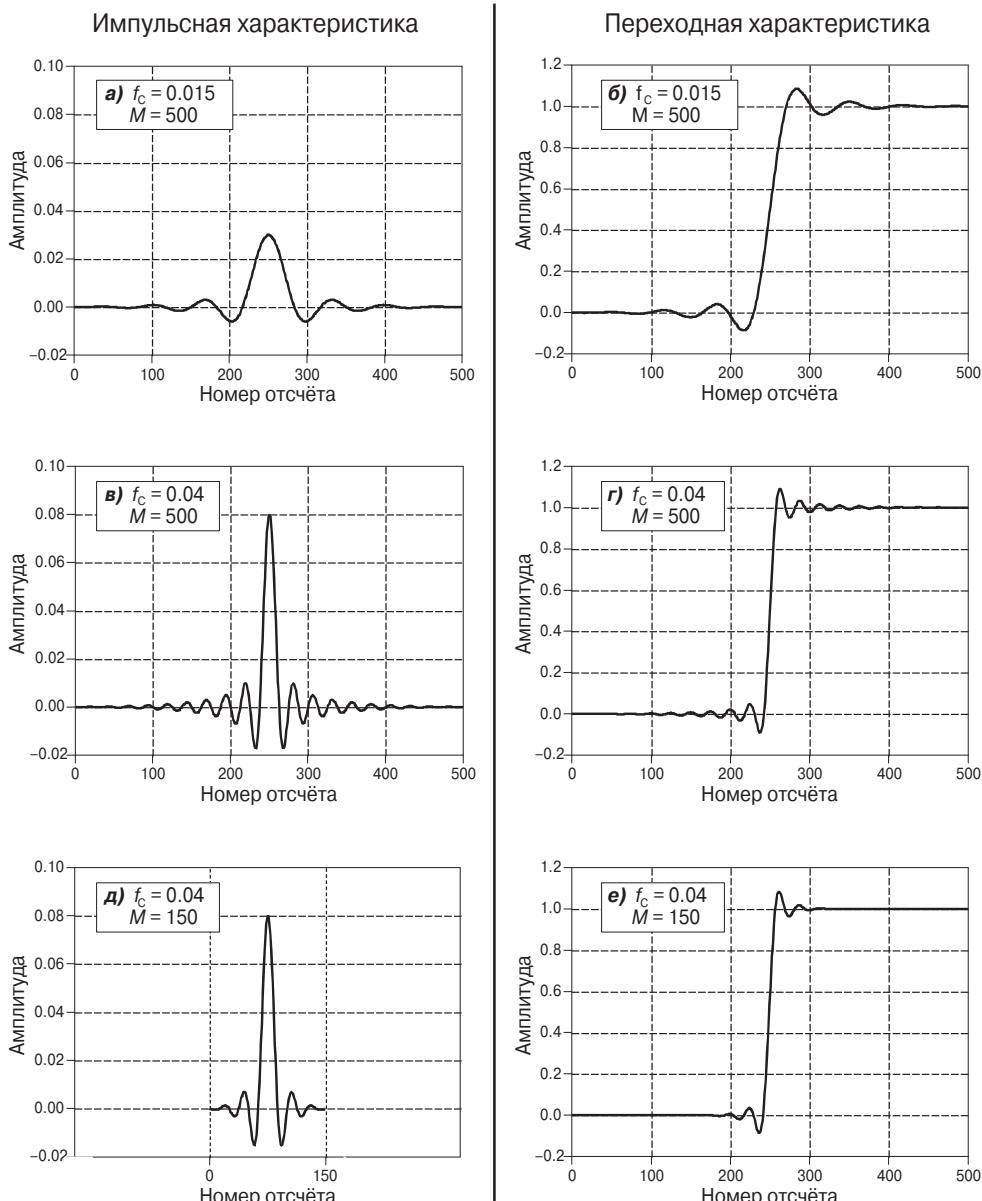
$$h[i] = K \frac{\sin(2\pi f_C (i - M/2))}{i - M/2} \left[ 0.42 - 0.5 \cos\left(\frac{2\pi i}{M}\right) + 0.08 \cos\left(\frac{4\pi i}{M}\right) \right]. \quad (16.4)$$

Импульсная характеристика оконного фильтра. Здесь  $f_C = 0\dots 0.5$  — приведённая частота среза (отношение верхней частоты среза в герцах к частоте дискретизации). Номера весовых коэффициентов фильтра меняются в диапазоне  $0\dots M$ , т. е. порядок фильтра равен  $(M+1)$ . Параметр  $K$  выбирается таким, чтобы обеспечить единичный коэффициент усиления на нулевой частоте. Чтобы устранить неопределенность при  $i = M/2$ , следует выполнить преобразование и получить  $h[i] = 2\pi f_C K$ .

Стоящее в правой части сложное выражение не должно вас пугать. Теперь вы легко можете выделить в нём три важные составляющие: функцию вида  $\sin(x)/x$ , сдвиг на  $M/2$  и оконную функцию Блэкмана. Для получения фильтра с единичным коэффициентом усиления на нулевой частоте нужно так выбрать параметр  $K$ , чтобы сумма всех весовых коэффициентов фильтра получилась равной единице. Обычно при программной реализации фильтра избегают использования дополнительной операции умножения на нормирующий коэффициент  $K$ , предварительно умножая на  $K$  все весовые коэффициенты фильтра, как это сделано в Программе 16.1. Кроме того, следует обратить внимание на вычисление центрального коэффициента с индексом  $i = M/2$ , так как в выражении (16.4) в этом случае возникает неопределенность вида  $0/0$ .

Выражение (16.4) достаточно громоздко, но использовать его очень просто: достаточно ввести указанное выражение в код программы и предоставить все вычисления своему компьютеру. Если вы захотите выполнять все вычисления вручную, то допустите много ошибок.

Как уже говорилось, весовые коэффициенты КИХ-фильтра совпадают с отсчётами его импульсной характеристики. Теперь рассмотрим подробнее принцип размещения отсчётов, найденных в соответствии с выражением (16.4), в памяти компьютера. Пусть  $M$  равно 100. Тогда первый отсчёт импульсной характеристики размещается в нулевой ячейке выделенного массива, а последний — в сотой ячейке. То есть для хранения всех весовых коэффициентов достаточно массива, состоящего из 101 элемента. Центральный коэффициент хранится в ячейке с по-



**Рис. 16.4.** Примеры импульсных характеристик оконных фильтров и соответствующих им переходных характеристик. Частота колебаний, наблюдающихся в переходном процессе, приблизительно равна верхней граничной частоте фильтра  $f_B$ .  $M$  — номер последнего отсчёта импульсной характеристики, т. е.  $(M + 1)$  — порядок фильтра.

рядковым номером  $M/2 = 50$ . Левые и правые 50 отсчётов симметричны относительно центрального. Содержимое ячейки 0 равно содержимому ячейки 100, а 49-й отсчёт равен отсчёту с номером 51. Во многих случаях важно, чтобы количество весовых коэффициентов фильтра принадлежало некоторому ряду допустимых значений. Такая проблема решается введением дополнительных нулевых

значений. Например, для алгоритма БПФ необходимо, чтобы число отсчётов было равно степени двойки. При  $M = 100$  достаточно просто увеличить длину массива весовых коэффициентов фильтра до 128, записав нули в ячейки с номерами 101...127.

На Рис. 16.4 показано несколько вариантов импульсных характеристик оконных фильтров и соответствующие им переходные характеристики. Отсчёты, близкие к краям импульсных характеристик, настолько малы, что на графиках их значения практически неотличимы от нулевого уровня. Но это вовсе не значит, что ими можно пренебречь! Будучи малыми по абсолютной величине, взятые в совокупности крайние отсчёты оказывают большое влияние на частотные характеристики оконного фильтра. Именно по этой причине для оконных фильтров больше подходит арифметика с плавающей точкой, в то время как целочисленная арифметика оказывается неспособной охватить широкий динамический диапазон значений весовых коэффициентов. Попытаемся теперь ответить на вопрос: насколько хороши оконные фильтры для обработки информации во временной области? Ответим сразу: они просто непригодны! На графиках, изображённых справа, заметны ярко выраженный колебательный процесс и большое перерегулирование. Такие фильтры не подходят для обработки сигналов, в которых информация заложена во временной области.

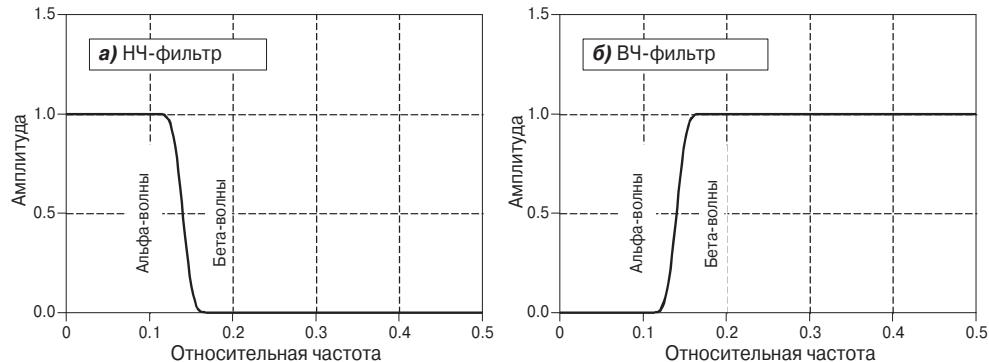
## 16.3. Примеры использования оконных фильтров

Электроэнцефалограмма (ЭЭГ) — это запись биотоков мозга. Получение ЭЭГ связано с измерением напряжений порядка милливольта, возникающих на при соединённых к голове человека электродах. От каждой нервной клетки исходят электрические импульсы. ЭЭГ отражает результат совместного действия огромного числа электрических импульсов. Хотя взаимосвязь между мыслительным процессом и электрическими колебаниями ещё недостаточно изучена, известно, что разные частоты в ЭЭГ связаны с разными состояниями подсознания. Если вы закроете глаза и расслабитесь, в ЭЭГ будут преобладать колебания низкой частоты: приблизительно 7...12 Гц. Такие колебания получили название *альфа-ритма* и соответствуют состоянию удовлетворённости и пониженного уровня внимания. Как только вы откроете глаза и посмотрите на окружающие предметы, возникнет *бета-ритм*. Частота бета-волн лежит в диапазоне 17...20 Гц. Можно обнаружить и колебания на других частотах, возникающие у детей, а также в различных фазах сна и при нарушении работы мозга, например при эпилепсии.

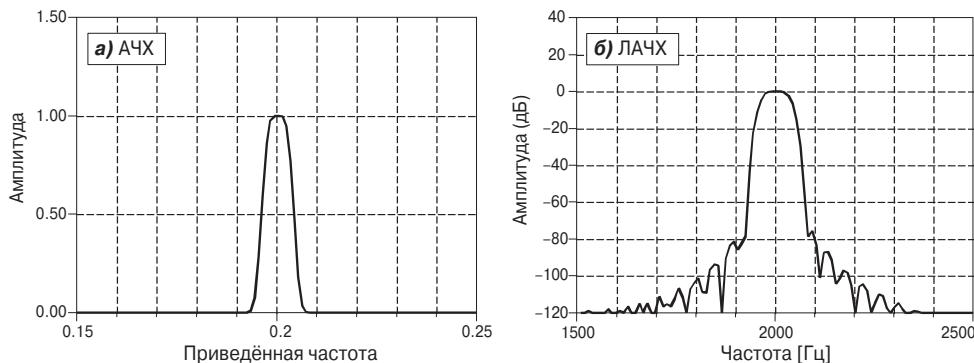
Предположим, что сигнал ЭЭГ усиливается аналоговой частью аппаратуры, а затем переводится в цифровую форму с частотой дискретизации 100 отсчётов в секунду. Значит, за 50 секунд будет получено 5000 отсчётов сигнала. Задача заключается в том, чтобы отделить альфа-ритм от бета-ритма. Для этого достаточно применить цифровой НЧ-фильтр с верхней граничной частотой 14 Гц (приведённая частота равна 0.14) и с шириной переходной зоны, не превышающей 4 Гц (в единицах приведенных частот — 0.04). Пользуясь выражением (16.3), находим, что фильтр должен иметь около 101 весового коэффициента. Для повышения крутизны спада АЧХ в переходной зоне лучше выбрать фильтр Хэмминга.

**Программа 16.1** реализует такой фильтр. АЧХ фильтра (Рис. 16.5) получена по импульсной характеристике с помощью преобразования Фурье.

Рассмотрим ещё один пример. Требуется рассчитать *полосовой фильтр* для выделения *тонального сигнала*, который возникает в процессе набора номера при нажатии кнопок на телефонном аппарате. Предположим, что дискретизация



**Рис. 16.5.** Пример использования оконных фильтров. При снятии ЭЭГ альфа-ритм и бета-ритм разделяются НЧ- и ВЧ-фильтрами, для которых  $M = 100$ . **Программа 16.1** реализует НЧ-фильтр. Для ВЧ-фильтра можно написать аналогичную программу, в которой будут использоваться весовые коэффициенты, полученные в результате инверсии АЧХ.



**Рис. 16.6.** Пример полосового оконного фильтра. Данный фильтр рассчитан для работы на частоте дискретизации 10 кГц. Центральная частота равна 2 кГц, ширина полосы пропускания — 80 Гц, ширина переходных зон — 50 Гц. Для достижения заданной частотной избирательности необходим 801 весовой коэффициент. Чтобы увеличить затухание в зоне непрозрачности, предпочтение следует отдать фильтру Блэкмана. На (а) и (б) изображены соответственно АЧХ и ЛАЧХ рассчитанного оконного фильтра. На графике АЧХ по горизонтальной оси отложена приведённая частота дискретного сигнала, а на графике ЛАЧХ — частота в герцах, получаемая в предположении, что обрабатывается аналоговый сигнал.

производится с частотой 10 кГц и что требуется выделить полосу шириной 80 Гц с центром на частоте 2 кГц. Измеряя частоту в долях частоты дискретизации, можно сказать, что необходим цифровой фильтр, подавляющий все частоты ниже 0.196 и все частоты выше 0.204 (эти значения соответствуют 1960 и 2040 Гц). Что бы ширина переходных зон на границах полосы пропускания не превышала 50 Гц

(относительная ширина 0.005), порядок фильтра должен быть равен 801. В данном случае предпочтение следует отдать фильтру Блэкмана. Расчёт весовых коэффициентов выполняется в **Программе 16.2**, а частотная характеристика полученного фильтра рассмотрена на **Рис. 16.6**. Процедура расчёта фильтра включает несколько этапов. Сначала рассчитываются два НЧ-фильтра с частотами среза 0.196 и 0.204 соответственно. Для второго фильтра выполняется инверсия АЧХ, в результате чего он превращается в ВЧ-фильтр (см. **Рис. 14.6**). Затем весовые коэффициенты двух фильтров попарно складываются, образуя режекторный фильтр (**Рис. 14.8**). Теперь остаётся лишь выполнить *инверсию АЧХ*, чтобы фильтр стал полосовым.

### Программа 16.1

```

100 'НИЗКОЧАСТОТНЫЙ ОКНОВЫЙ ФИЛЬТР
110 'Данная программа предназначена для обработки 5000 отсчётов
120 'и получения 4900 отсчётов выходного сигнала.
130 '
140 DIM X[5000]           'X[ ] - массив отсчётов входного сигнала
150 DIM Y[5000]           'Y[ ] - массив отсчётов выходного сигнала
160 DIM H[100]            'H[ ] - массив весовых коэффициентов НЧ-фильтра
170 '
180 PI = 3.14159265
190 FC = .14              'Определение частоты среза (в диапазоне 0...0.5)
200 M% = 100              'Определение порядка фильтра (101-й порядок)
210 '
220 GOSUB XXXX           'Некоторая подпрограмма загрузки данных в X[ ]
230 '
240 '                      'Вычисление весовых коэффициентов НЧ-фильтра (16.4)
250 FOR I% = 0 TO 100
260 IF (I%-M%/2) = 0 THEN H[I%] = 2*PI*FC
270 IF (I%-M%/2) <> 0 THEN H[I%] = SIN(2*PI*FC * (I%-M%/2)) / (I%-M%/2)
280 H[I%] = H[I%] * (0.54 - 0.46*COS(2*PI*I%/M%))
290 NEXT I%
300 '
310 SUM = 0                'Нормирование коэффициентов НЧ-фильтра для получения
320 FOR I% = 0 TO 100      'единичного коэффициента усиления на нулевой частоте
330 SUM = SUM + H[I%]
340 NEXT I%
350 '
360 FOR I% = 0 TO 100
370 H[I%] = H[I%] / SUM
380 NEXT I%
390 '
400 FOR J% = 100 TO 4999 'Операция свёртки с входным сигналом
410 Y[J%] = 0
420 FOR I% = 0 TO 100
430 Y[J%] = Y[J%] + X[J%-I%] * H[I%]
440 NEXT I%
450 NEXT J%
460 '
470 END

```

**Программа 16.2**

```

100 'ПОЛОСОВОЙ ОКНОННЫЙ ФИЛЬТР
110 'Данная программа выполняет расчёт фильтра 801-го порядка
120 '
130 DIM A[800]           'A[ ] область памяти для НЧ-фильтра
140 DIM B[800]           'B[ ] область памяти для ВЧ-фильтра
150 DIM H[800]           'H[ ] область памяти для полосового фильтра
160 '
170 PI = 3.1415926
180 M% = 800             'Определение порядка фильтра (801-й порядок)
190 '
200 '                   'Вычисление весовых коэффициентов НЧ-фильтра с частотой
210 FC = 0.196            'среза 0.196 (16.4) и размещение их в A[ ]
220 FOR I% = 0 TO 800
230 IF (I%-M%/2) = 0 THEN A[I%] = 2*PI*FC
240 IF (I%-M%/2) <> 0 THEN A[I%] = SIN(2*PI*FC * (I%-M%/2)) / (I%-M%/2)
250 A[I%] = A[I%] * (0.42 - 0.5*COS(2*PI*I%/M%) + 0.08*COS(4*PI*I%/M%))
260 NEXT I%
270 '
280 SUM = 0               'Нормирование коэффициентов НЧ-фильтра для получения
290 FOR I% = 0 TO 800     'единичного коэффициента усиления на нулевой частоте
300 SUM = SUM + A[I%]
310 NEXT I%
320 '
330 FOR I% = 0 TO 800
340 A[I%] = A[I%] / SUM
350 NEXT I%
360 '                   'Вычисление весовых коэффициентов НЧ-фильтра с частотой
370 FC = 0.204            'среза 0.204 (16.4) и размещение их в B[ ]
380 FOR I% = 0 TO 800
390 IF (I%-M%/2) = 0 THEN B[I%] = 2*PI*FC
400 IF (I%-M%/2) <> 0 THEN B[I%] = SIN(2*PI*FC * (I%-M%/2)) / (I%-M%/2)
410 B[I%] = B[I%] * (0.42 - 0.5*COS(2*PI*I%/M%) + 0.08*COS(4*PI*I%/M%))
420 NEXT I%
430 '
440 SUM = 0               'Нормирование коэффициентов НЧ-фильтра для получения
450 FOR I% = 0 TO 800     'единичного коэффициента усиления на нулевой частоте
460 SUM = SUM + B[I%]
470 NEXT I%
480 '
490 FOR I% = 0 TO 800
500 B[I%] = B[I%] / SUM
510 NEXT I%
520 '
530 FOR I% = 0 TO 800     'Преобразование НЧ-фильтра (массив B[ ])
540 B[I%] = - B[I%]       'в ВЧ-фильтр путём инверсии АЧХ (см. Рис. 14.5)
550 NEXT I%
560 B[400] = B[400] + 1
570 '
580 '
590 FOR I% = 0 TO 800     'Попарное сложение коэффициентов НЧ-фильтра A[ ]

```

```

600 H[I%] = A[I%] + B[I%] 'и ВЧ-фильтра B[ ] и сохранение получившегося
610 NEXT I%           'режекторного фильтра в H[ ] (см. Рис. 14.8)
620 '
630 FOR I% = 0 TO 800      'Преобразование режекторного фильтра в полосовой
640 H[I%] = -H[I%]          'с помощью инверсии АЧХ
650 NEXT I%
660 H[400] = H[400] + 1      'Коэффициенты полосового фильтра находятся в H[ ]
670 '
680 END

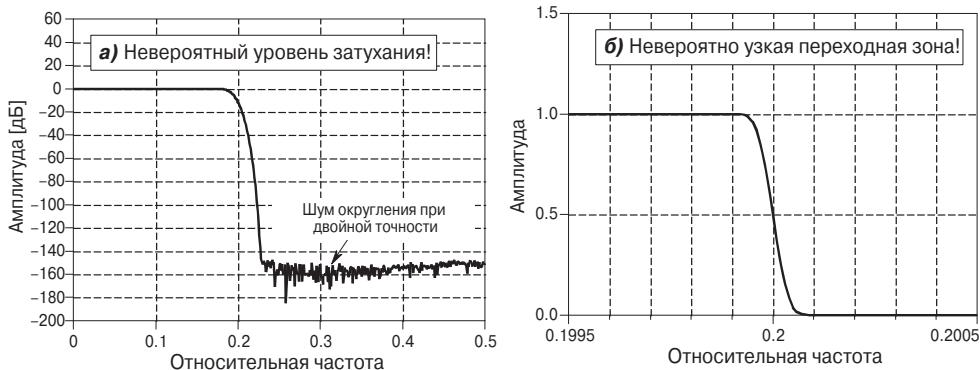
```

## 16.4. Достижение сверхвысокой точности

Оконные фильтры позволяют достигать невероятной *точности*. Допустим, вы хотите выделить сигнал напряжением 1 мВ, принятый по сети переменного тока 120 В. Значит, необходим фильтр с уровнем затухания в зоне подавления по меньшей мере  $-120$  дБ (одна миллионная в линейном масштабе). Уже было показано выше, что оконная функция Блэкмана позволяет достичь лишь  $-74$  дБ (приблизительно одна пятитысячная). Но оказывается, что затухание в зоне подавления легко увеличить, выполнив процедуру фильтрации ещё раз. При этом коэффициент подавления станет равным  $-148$  дБ (это одна тридцатимиллионная!). Можно объединить два использованных фильтра в один, выполнив *свёртку* их импульсных характеристик. То есть свёртка импульсной характеристики с самой собой позволяет значительно увеличить подавление сигнала в зоне непрозрачности. Но за увеличение подавления придётся заплатить возрастанием числа весовых коэффициентов (удлинением импульсной характеристики) и уменьшением крутизны спада АЧХ в переходной зоне. На Рис. 16.7а показана ЛАЧХ НЧ-фильтра 201-го порядка, полученного в результате выполнения свёртки импульсных характеристик двух оконных фильтров Блэкмана 101-го порядка. Качество полученного фильтра просто удивительное! Чтобы достичь очень большого затухания (более сильного, чем  $-100$  дБ), придётся проводить вычисления с двойной точностью. При обычной точности представления данных обработки сигналов, лежащих в *полосе пропускания*, приводит к возникновению шума округления, который во многих случаях «проникает» в зону подавления фильтра, достигая уровня  $-100...-120$  дБ и более.

Увеличение числа весовых коэффициентов до 32001 позволяет получить фильтр с невероятно узкой переходной зоной (Рис. 16.7б). Её ширина составляет всего лишь 0.000125 часть от частоты дискретизации. Насколько хорош данный фильтр? Попытайтесь построить аналоговый фильтр, пропускающий все частоты в диапазоне 0...1000 Гц с максимальной ошибкой по амплитуде 0.02% и подавляющий все частоты, превышающие 1001 Гц, до уровня 0.02% от их первоначальной амплитуды. С помощью аналоговых фильтров достичь таких характеристик просто невозможно! Но ещё больше поражает то, что оба фильтра на Рис. 16.7 используют вычисления с обычной точностью, а переход к двойной точности позволяет повысить качество фильтрации в миллион раз.

Наиболее серьёзный недостаток оконных фильтров связан с высокой *вычислительной сложностью* их реализации. На обработку каждого очередного отсчёта входного сигнала может потребоваться недопустимо большой объём вычислений,



**Рис. 16.7.** Невероятная точность частотных характеристик оконных фильтров: **(а)** повышение уровня затухания в зоне подавления достигается свёрткой импульсной характеристики оконного фильтра с самой собой; **(б)** оконный фильтр 32001-го порядка имеет очень узкую переходную зону.

т. к. в основе оконных фильтров лежит трудоёмкая операция свёртки. Значитель- но сократить вычислительные затраты позволяет использование алгоритма *быст- рой свёртки* (Глава 18). Альтернативу оконным фильтрам представляют *рекурсив- ные фильтры* (Глава 19), также обеспечивающие высокое качество частотной селекции сигналов.

Являются ли оконные фильтры оптимальными фильтрами частотной селек- ции? Нет, потому что многие более сложные алгоритмы позволяют получать фильтры более высокого качества. Но прежде чем перейти к изучению таких ал- горитмов, основанных на сложных математических преобразованиях, постарай- тесь понять, какого именно качества вам требуется достичь. С помощью оконных фильтров можно добиться практически любого, сколь угодно высокого, качества фильтрации. Все сложные алгоритмы расчёта и реализации фильтров частотной селекции созданы лишь для того, чтобы хоть немного уменьшить количество ве- совых коэффициентов при заданном качестве фильтрации. Такое уменьшение порядка фильтра приводит к уменьшению вычислительной сложности. Однако потребуется разобраться в сложных математических выражениях, чтобы таким образом хоть немного повысить эффективность вашей системы.

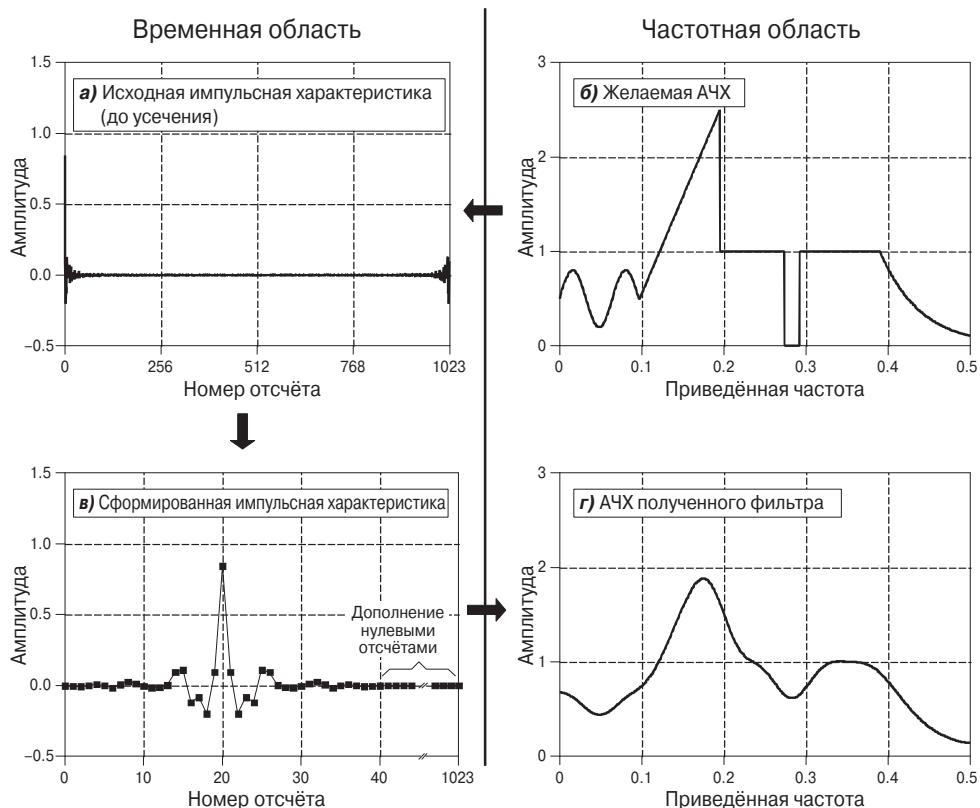
## СПЕЦИАЛЬНЫЕ ФИЛЬТРЫ

Большинство *цифровых фильтров* воспроизводят одну из четырёх форм АЧХ и соответствуют одному из четырёх основных типов: *НЧ-фильтры*, *ВЧ-фильтры*, *полосовые* и *режекторные*. Но в ряде приложений возникает необходимость применения фильтров с частотной характеристикой специального вида. Методика расчёта таких фильтров изложена в первой части данной главы. Использование ЦОС позволяет обеспечить гораздо более высокую точность воспроизведения частотной характеристики произвольно заданной формы по сравнению с аналоговыми системами. Во второй части главы рассмотрены два варианта практических приложений, в которых специальные фильтры играют важную роль: коррекция частотной характеристики — восстановление формы сигнала на выходе системы с неидеальной частотной характеристикой и оптимальная фильтрация — разделение сигналов с перекрывающимися частотными спектрами. Методами ЦОС качество обработки сигналов в таких приложениях удается значительно повысить.

### 17.1. АЧХ произвольной формы

Описанный в предыдущей главе подход к построению оконных фильтров на самом деле может быть использован для расчёта *цифровых фильтров с частотной характеристикой* произвольного вида. Единственное отличие проявляется в способе перехода от описания свойств фильтра в частотной области к их описанию во временной области. Если в случае оконных фильтров импульсная и частотная характеристики описывались некоторыми уравнениями и были связаны аналитически посредством преобразования Фурье, то в рассмотренном ниже методе расчёта эти характеристики представляются в форме числовых массивов, которые связывают друг с другом преобразование, основанные на алгоритме БПФ.

Последовательность выполняемых операций показана на Рис. 17.1. Изначально задаём вид желаемой АЧХ цифрового фильтра. АЧХ, показанная на (б), содержит много резких переходов и поэтому не может быть воспроизведена на базе аналоговых цепей. Желаемая АЧХ задана не с помощью математических уравнений, а посредством массива отсчётов, взятых с постоянным шагом по частоте: в рассмотренном примере на интервале от нулевой частоты до половины частоты дискретизации расположено 513 точек. Чем чаще размещаются точки, в которых задана АЧХ фильтра, тем точнее она будет воспроизведена, но при этом возрастут и вычислительные затраты на реализацию фильтра. Для большинства практических приложений 513 точек вполне достаточно. Кроме АЧХ, показанной на (б), необходимо задать в тех же точках соответствующую ФЧХ. В нашем примере все отсчёты ФЧХ принимают нулевые значения (данный массив, состоящий целиком из нулей, на рисунке не показан). В общем случае отсчёты ФЧХ, как и отсчёты АЧХ,



**Рис. 17.1.** Пример расчёта КИХ-фильтра. **а)** Первоначальный вид импульсной характеристики. Данная импульсная характеристика состоит из 1024 отсчётов, найденных в результате обратного ДПФ желаемой АЧХ. **б)** Желаемая АЧХ фильтра, заданная в 513 точках, размещенных равномерно в полосе нормированных частот 0...0.5. **в)** Импульсная характеристика, полученная после применения операций усечения до  $(M + 1)$  отсчётов, сдвига на  $M/2$  отсчётов и умножения на оконную функцию Хэмминга или Блэкмана. В рассмотренном примере  $M = 40$ . Все перечисленные выше операции по формированию импульсной характеристики выполняет **Программа 17.1**. **г)** АЧХ полученного фильтра, найденная путём применения ДПФ к дополненной нулями импульсной характеристике фильтра.

могут принимать произвольные значения, за исключением первого и последнего отсчётов, которым всегда соответствует нулевая или кратная  $2\pi$  фаза. Если отсчёты комплексной частотной характеристики представить не в полярной, а в прямоугольной системе координат, то частотную характеристику фильтра можно разложить на действительную и мнимую составляющие.

Следующий шаг расчёта связан с применением обратного ДПФ с целью перехода от частотной области во временнную. Самый простой способ перехода во времennую область заключается в использовании прямоугольных координат для представления комплексной частотной характеристики и применении алгоритма БПФ. В результате получаем некоторую функцию времени (**а**) протяжённостью 1024 отсчёта: 0...1023. Полученная функция является импульсной характеристикой фильтра, имеющего комплексную частотную характеристику заданного вида. Тем не менее импульсная характеристика в такой форме не подходит для практи-

ческого использования. Как и в рассмотренной ранее процедуре расчёта оконных фильтров, требуется выполнить три дополнительных операции: сдвиг, усечение и взвешивание *оконной функцией*. Примером программы, позволяющей перейти от исходной импульсной характеристики (*а*) к сформированной (*в*), является **Программа 17.1**. В этом примере  $M = 40$ , т.е. длина импульсной характеристики ограничена после операции усечения до 41 отсчёта: 0...40. Как и при расчёте оконных фильтров, отсёты, близкие к краям полученной импульсной характеристики (*в*), на графике практически неотличимы от нуля. И, как и раньше, ни в коем случае нельзя пренебрегать этими отсчётами.

### Программа 17.1

```

100 'ПРОГРАММА РАСЧЁТА СПЕЦИАЛЬНОГО ФИЛЬТРА
110 'Данная программа позволяет преобразовать 1024 отсчёта исходной
120 'импульсной характеристики (ИХ) (Рис. 17.1а) в ( $M + 1$ ) отсчёт
130 'результатирующей ИХ (Рис. 17.1в), т.е. получить весовые коэффициенты.
140 '
150 DIM REX[1023]           'REX[ ] - массив отсчётов исходной ИХ
160 DIM T[1023]             'T[ ] - буфер временного хранения данных
170 '
180 PI = 3.14159265
190 M% = 40                 'Параметр, определяющий порядок фильтра (41 порядок)
200 '
210 GOSUB XXXX              'Некоторая подпрограмма загрузки отсчётов ИХ в REX[ ]
220 '
230 FOR I% = 0 TO 1023      'Сдвиг (циклический) ИХ на  $M/2$  отсчётов вправо
240 INDEX% = I% + M%/2
250 IF INDEX% > 1023 THEN INDEX% = INDEX%-1024
260 T[INDEX%] = REX[I%]
270 NEXT I%
280 '
290 FOR I% = 0 TO 1023
300 REX[I%] = T[I%]
310 NEXT I%
320 '                         'Усечение ИХ и формирование оконной функцией
330 FOR I% = 0 TO 1023
340 IF I% <= M% THEN REX[I%] = REX[I%] * (0.54 - 0.46 * COS(2*PI*I%/M%))
350 IF I% > M% THEN REX[I%] = 0
360 NEXT I%
370 '                         'Результатирующая ИХ размещается в REX[0]...REX[40]
380 END

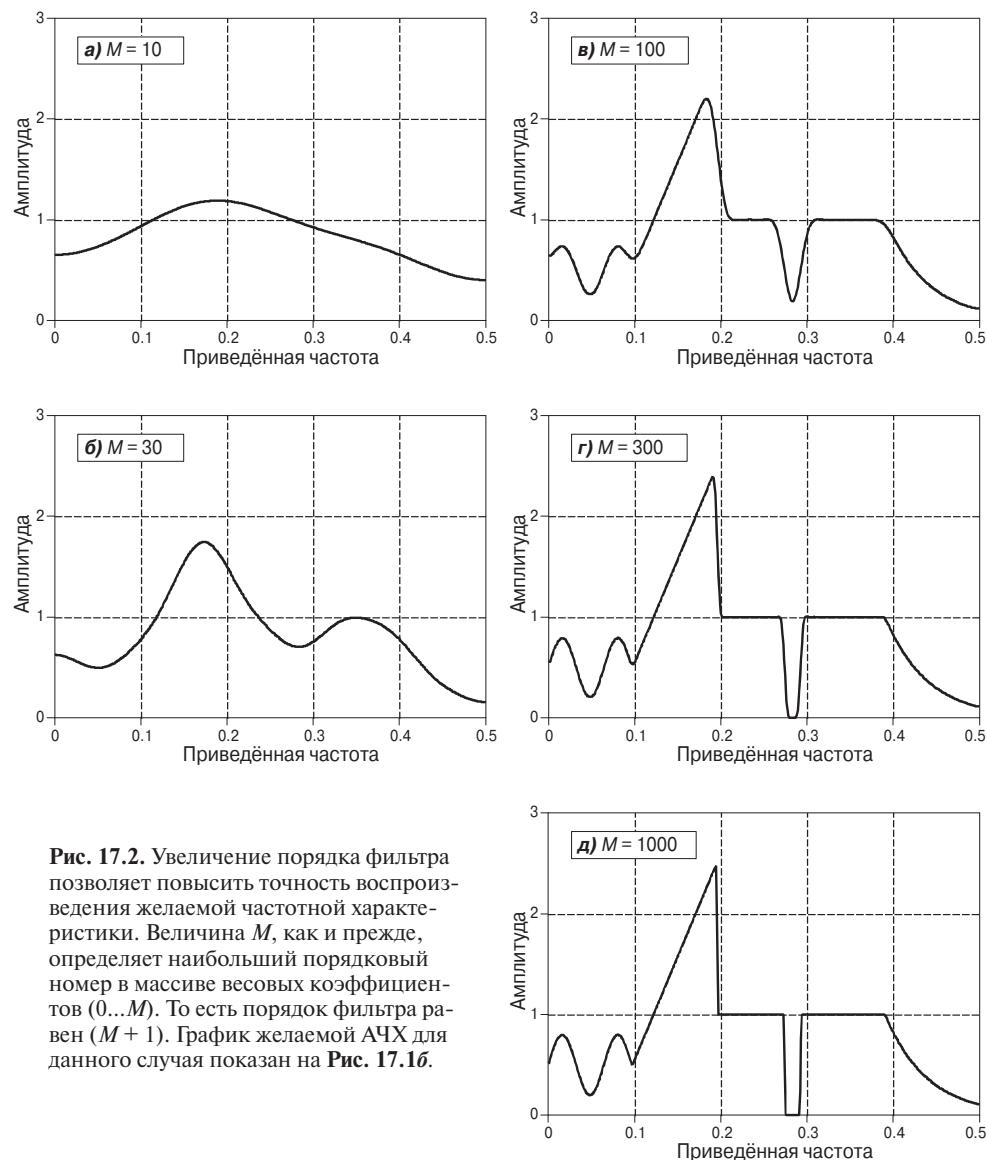
```

Последним шагом при расчёте фильтра является оценка правильности выполненных преобразований. С помощью ДПФ (используется алгоритм БПФ) находим АЧХ полученного фильтра (*г*). Чтобы повысить разрешающую способность при построении частотной характеристики, необходимо предварительно дополнить импульсную характеристику полученного фильтра нулями и только затем воспользоваться алгоритмом БПФ. В нашем примере импульсная характеристика дополняется нулями до 1024 отсчётов (41 отсчёт рассчитанной ранее импульсной характеристики и 983 нулевых отсчёта), что позволяет получить АЧХ, заданную в

513 точках, перекрывающих диапазон частот 0...0.5 с постоянным шагом по частоте.

Чем выше порядок фильтра, тем ближе его частотная характеристика к желаемой (**Рис. 17.2**). Теоретически, повышая порядок фильтра, можно достичь сколь угодно точного воспроизведения практически любой желаемой частотной характеристики.

Мы описали всю процедуру расчёта фильтров с произвольной частотной характеристикой. Однако чтобы глубже разобраться с особенностями данного метода расчёта, необходимо выявить основные «подводные камни». В первую очередь возникает вопрос: почему нельзя воспользоваться непосредственно исходной им-



**Рис. 17.2.** Увеличение порядка фильтра позволяет повысить точность воспроизведения желаемой частотной характеристики. Величина  $M$ , как и прежде, определяет наибольший порядковый номер в массиве весовых коэффициентов (0... $M$ ). То есть порядок фильтра равен ( $M + 1$ ). График желаемой АЧХ для данного случая показан на **Рис. 17.1б**.

пульсной характеристикой (**Рис. 17.1а**), а обязательно надо её дополнительно формировать? Ведь если фильтру с импульсной характеристикой, показанной на **Рис. 17.1а**, соответствует желаемая АЧХ (**Рис. 17.1б**), то, наверное, следует использовать именно эту импульсную характеристику? И всё же такое рассуждение ошибочно. Объясним почему.

При расчёте специальных фильтров желаемая АЧХ задана массивом отсчётов. Поведение АЧХ на интервалах между отсчётами может быть произвольным. Для упрощения рассуждений предположим, что имеются два варианта поведения: «хорошее» и «плохое». «Хорошее» подразумевает, что частотная характеристика имеет вид гладкой кривой, а «плохое» — что АЧХ содержит много скачков. Импульсной характеристике, показанной на **Рис. 17.1а**, соответствует АЧХ как раз с таким «плохим» поведением. Чтобы убедиться в этом, достаточно дополнить импульсную характеристику большим числом нулевых отсчётов и выполнить ДПФ. Полученная АЧХ характеризуется динамичным изменением на интервалах между изначально заданными отсчётами. Поэтому такой фильтр совершенно не пригоден для практического использования.

Для того чтобы понять, почему так происходит, предположим, что мы определили желаемую «хорошую» частотную характеристику в каждой из бесконечного множества точек, расположенных на интервале приведённых частот  $0\dots0.5$ . Это означает, что желаемая АЧХ представляет собой непрерывную функцию частоты. Импульсная характеристика, получающаяся при выполнении непрерывного обратного преобразования Фурье, имеет бесконечную длину во временной области. Такая импульсная характеристика не может быть воспроизведена на компьютере, потому что она содержит бесконечное число отсчётов. Если мы ограничим частотную характеристику  $(N/2 + 1)$  отсчётами, то во временной области получим  $N$  отсчётов импульсной характеристики, которые не могут содержать всю информацию об исходном сигнале. В итоге бесконечная импульсная характеристика сокращается до  $N$  дискретных отсчётов, превращаясь при этом из «хорошей» в «плохую». К счастью, взвешивание оконной функцией позволяет достаточно хорошо сгладить возникающие искажения.

И всё же рассчитать цифровой фильтр по заданной частотной характеристике не так трудно, как определить вид этой желаемой частотной характеристики для каждого конкретного приложения. Рассмотрим далее две практические задачи, при решении которых требуется рассчитывать специальные фильтры.

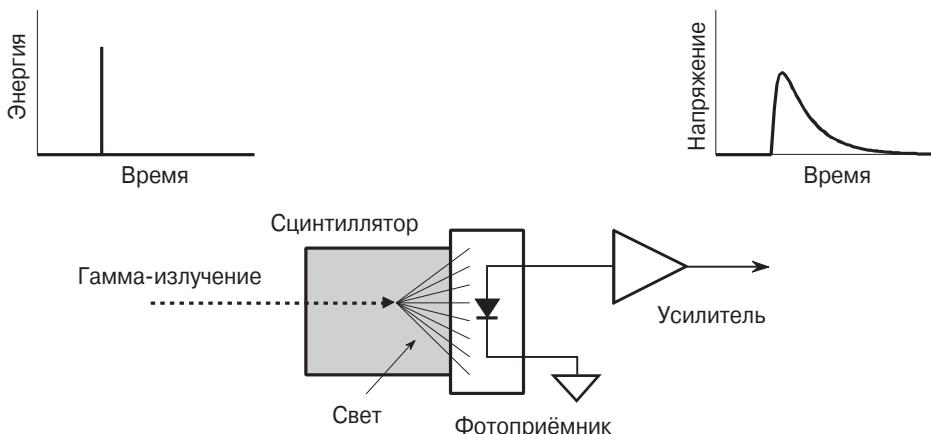
## 17.2. Коррекция (выравнивание) частотной характеристики

Передача информации в аналоговой форме обычно связана с искажениями, которые могут быть описаны при помощи интеграла свёртки: размытием изображения, вызванным дрожанием видеокамеры; возникновением эха на протяжённых телефонных линиях; ограничением полосы частот сигнала аналоговыми датчиками и другой аналоговой аппаратурой и т. д. *Коррекция (выравнивание) частотной характеристики* динамической системы (канала связи) — это компенсация нежелательных искажений, вызванных самой динамической системой в процессе передачи информации. Для восстановления первоначальной формы

сигнала методом коррекции частотной характеристики необходимо обладать информацией о характеристиках искажающей системы. Такая информация, в частности, может быть получена в результате так называемого *слепого выравнивания*, т. е. без предварительной идентификации системы. Слепое выравнивание является, в свою очередь, ещё более сложной проблемой, для которой не существует общего решения, и в каждом конкретном приложении приходится подбирать собственный подход.

Суть метода обычной коррекции частотной характеристики проще всего понять, рассматривая сигнал в частотной области. Действие искажающей системы, которое можно описать во временной области при помощи свёртки, в частотной области выражается в различии коэффициентов передачи (как по амплитуде, так и по фазе) на разных частотах. Поэтому, чтобы получить исходный сигнал, достаточно воспользоваться таким фильтром, который способен компенсировать искажения, полученные в процессе свёртки, используя соответствующие преобразования в частотной области. Например, если искажающая система на некоторой частоте имеет коэффициент усиления по амплитуде, равный 0.5, и вносит фазовый сдвиг на 30 градусов, то корректирующий фильтр должен усиливать сигнал на данной частоте в 2 раза и обеспечивать фазовый сдвиг на  $-30$  градусов.

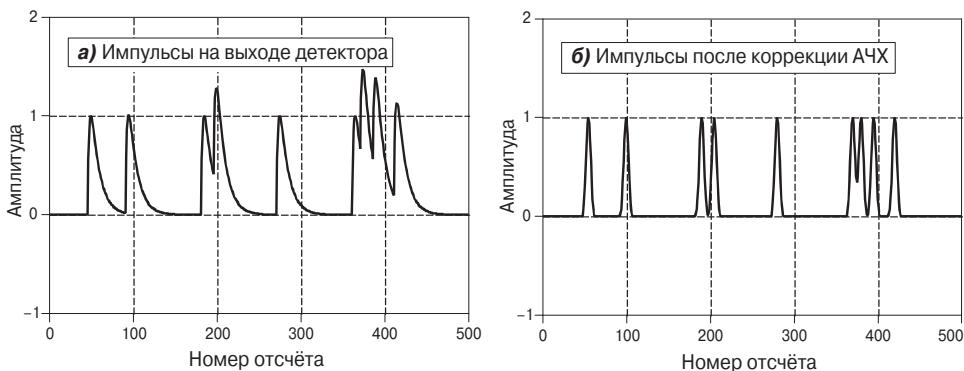
Применение метода *обращения свёртки* рассмотрим на примере детектора гамма-излучения (Рис. 17.3). Детектор состоит из *сцинтиллятора* и *фотоприёмника*. Для построения сцинтиллятора используется особое прозрачное вещество: йодид натрия или германат висмута. Эти химические соединения способны преобразовывать энергию гамма-квантов в короткие вспышки видимого света. Под влиянием света в фотоприёмнике, конструктивно представляющем собой фотодиод или фотоумножитель, создаются электрические импульсы, имеющие вид затухающей экспоненты. Передний фронт импульсов оказывается немного сглаженным. Такая форма электрических импульсов связана с физическими



**Рис. 17.3.** Пример системы, которая вносит искажения, описываемые операцией свёртки. Детектор гамма-излучения состоит из сцинтиллятора и фотоприёмника. Энергия гамма-кванта преобразуется сцинтиллятором в световую энергию. Полученный световой импульс преобразуется фотоприёмником в электрический сигнал. Гамма-квант представляет собой очень короткий импульс, в то время как реакция на выходе детектора (импульсная характеристика) имеет форму импульса, медленно затухающего по экспоненциальному закону.

процессами, протекающими в сцинтилляторе. Энергия гамма-кванта передаётся группе близко расположенных атомов сцинтиллятора, в результате чего эти атомы переходят на более высокий энергетический уровень. Затем возбуждённые атомы в случайные моменты времени возвращаются в своё прежнее состояние, излучая фотоны света. В результате возникает световой импульс, амплитуда которого убывает в течение нескольких сот наносекунд (для йодида натрия). Так как каждый гамма-квант представляет собой короткий импульс, то реакция детектора на него (импульс с экспоненциальным спадом) есть нечто иное, как импульсная характеристика регистрирующей системы.

Гамма-кванты воздействуют на сцинтиллятор в случайные моменты времени, приводя к возникновению на выходе детектора импульсов напряжения с экспоненциальным спадом (**Рис. 17.4а**). Полезная информация содержится в амплитудах полученных импульсов, пропорциональных энергиям породивших их гамма-квантов. Величина энергии гамма-квантов может рассказать много интересного о тех местах, в которых побывали эти кванты. В ней, к примеру, могут содержаться медицинская информация о состоянии больного, или сведения о возрасте далёких галактик, или предупреждение о наличии бомбы в багаже авиалайнера.



**Рис. 17.4.** Пример применения метода коррекции АЧХ. **(а)** Сигнал, снимаемый с выхода детектора гамма-излучения. Гамма-кванты регистрируются в случайные моменты времени. **(б)** Последовательность коротких импульсов, полученных благодаря коррекции АЧХ. Сокращение длительности импульсов позволяет повысить точность оценки значений их амплитуды.

Если бы все импульсы, возникающие при регистрации гамма-квантов, были чётко разделены во времени, то измерить их амплитуду было бы нетрудно. Однако близкие по времени возникновения импульсы очень часто перекрываются (**а**), и измерение амплитуды выполняется с большой ошибкой. Одним из возможных вариантов решения проблемы является применение метода коррекции АЧХ к снимаемому с выхода детектора сигналу. В результате импульсы становятся значительно короче и накладываются друг на друга намного меньше. Конечно, в идеале длительность регистрируемых импульсов должна сократиться до длительности импульсов породивших их гамма-квантов, но, к сожалению, это невозможно. Цель обращения свёртки — максимально уменьшить длину импульсов (**б**).

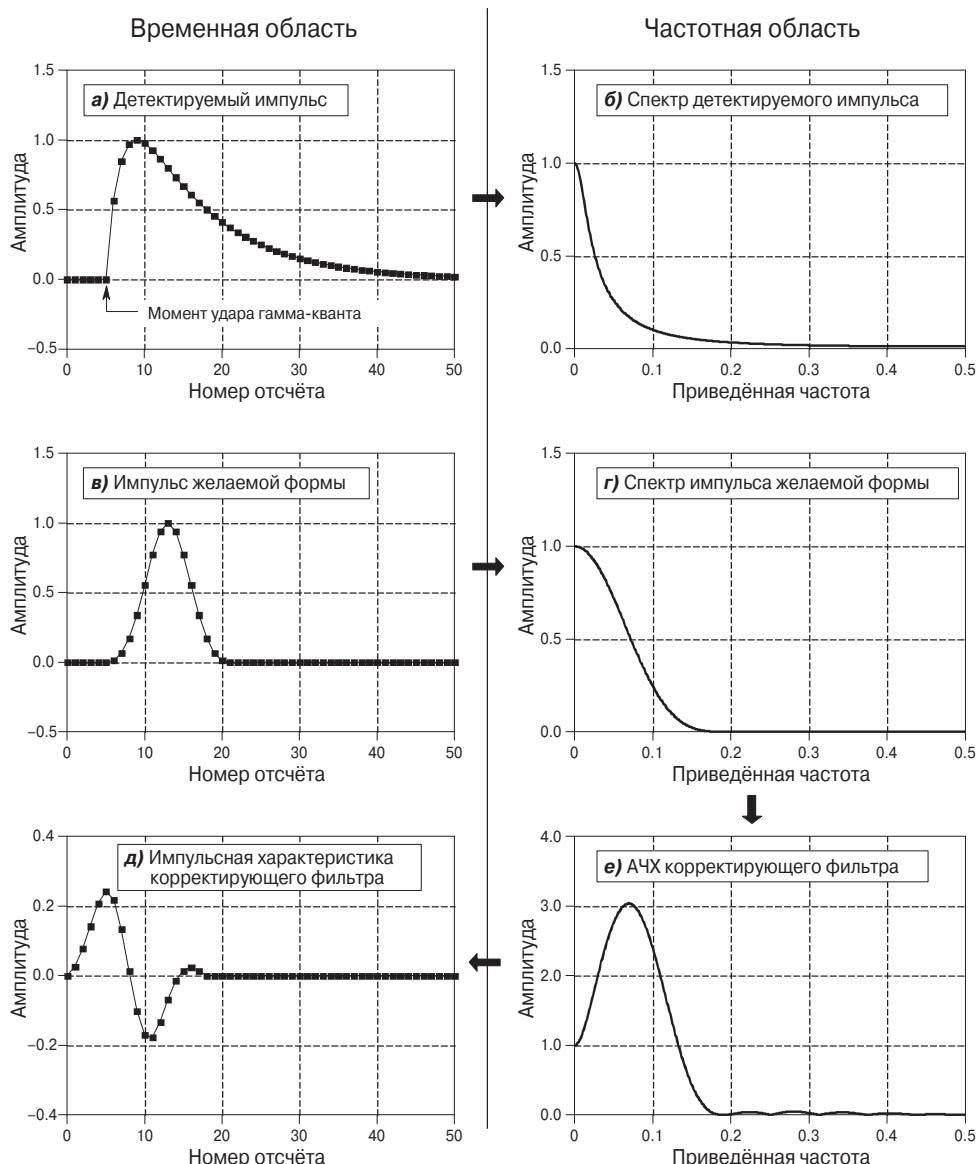
Несмотря на то, что получаемая детектором информация представлена во временной области, наши рассуждения большей частью относятся к частотной области, так как это позволяет лучше понять выполняемые действия. На Рис. 17.5а показана форма импульса, полученного на выходе детектора (она нам известна); на (в) — желаемая форма импульса (она также известна). В качестве примера желаемой формы импульса выбрана оконная функция Блэкмана, временная протяжённость которой равна приблизительно одной третьей протяжённости исходного импульса. Цель решаемой задачи состоит в том, чтобы найти такую импульсную характеристику (*д*), свёртка которой с поступившим на вход фильтра импульсом (*а*) позволит получить импульс желаемой формы (*в*). Поставленную задачу можно выразить в аналитической форме:  $a * d = c$ ; *a* и *v* известны; требуется найти *d*.

Если бы сигналы объединялись на основе сложения или умножения, то решение подобной задачи не вызывало бы труда. Достаточно было бы выполнить обратные операции: вычитание и деление. Но свёртка — это совсем другое дело. Для неё не существует простой обратной операции. Обращение свёртки во временной области оказывается чрезвычайно сложной задачей.

К счастью, решение значительно упрощается при переходе к частотной области. Вспомним, что свёртка в одной области соответствует умножению в другой. В отношении Рис. 17.5 сказанное означает, что задача коррекции АЧХ может быть сформулирована следующим образом:  $b \times e = g$ ; *b* и *g* известны; требуется найти *e*. Решить такую задачу просто: чтобы найти частотную характеристику исходного фильтра (*е*), достаточно частотный спектр желаемого импульса (*г*) разделить на частотный спектр полученного на выходе детектора импульса (*б*). Поскольку обнаруженные детектором импульсы имеют несимметричную форму, то их фазовый спектр отличен от нуля. Это значит, что требуется выполнять комплексное деление (делимое и делитель характеризуются амплитудой и фазой). О том, как выполняется деление комплексных спектров, рассказывается в Главе 9. Искомую импульсную характеристику (*д*) можно найти по частотной характеристике, используя процедуру расчёта специальных фильтров (ОБПФ, сдвиг, усечение и умножение на оконную функцию).

Операция обращения свёртки позволяет достичь лишь некоторого ограниченного улучшения формы импульса. Поэтому все принимаемые усилия сводятся к получению приемлемо коротких импульсов. Почему так получается? Чем короче желаемый импульс, тем шире его частотный спектр. Но, поскольку амплитуды высокочастотных компонент в принимаемом с выхода датчика импульсе очень малы, то восстанавливающий фильтр должен обладать очень высоким коэффициентом усиления на данных частотах. Например, для получения желаемого импульса, показанного на (в), необходимо трёхкратное усиление восстанавливающего фильтра на отдельных частотах (е). Если потребуется достичь ещё большего сокращения длительности импульсов, то потребуется фильтр с ещё более высоким коэффициентом усиления.

Серьёзную проблему представляет сильное влияние небольших ошибок. Например, если на некоторой частоте коэффициент усиления оказался равным 30 вместо 28, то такое изменение существенно исказит форму сигнала, полученного в результате коррекции АЧХ. Следовательно, для повышения точности восстановления сигнала требуется как можно более точно знать характеристики вноси-



**Рис. 17.5.** Метод коррекции АЧХ при рассмотрении во временной и в частотной областях: (а) реакция на удар гамма-кванта, полученная на выходе детектора гамма-излучения; (в) желаемая форма импульса; (б и г) амплитудные спектры импульсов; (д) импульсная характеристика искомого восстановливающего фильтра, для построения которой был использован алгоритм расчёта специальных фильтров (ОБПФ, сдвиг, усечение и умножение на оконную функцию); (е) АЧХ фильтра, позволяющего преобразовать сигнал с исходным спектром в сигнал с желаемым спектром. На графиках показаны только амплитудные характеристики, но следует помнить, что фазовые характеристики отличны от нуля и потому тоже участвуют в преобразованиях.

мых искажений. Но в большинстве практических приложений всегда присутствует некоторая неопределённость в измерении характеристик, связанная с влиянием таких воздействий, как электронные помехи, температурный дрейф,

различные отклонения характеристик от номинала и т. д. Все эти неопределённости устанавливают предельное качество восстановления сигнала, которого позволяет достичь метод коррекции АЧХ.

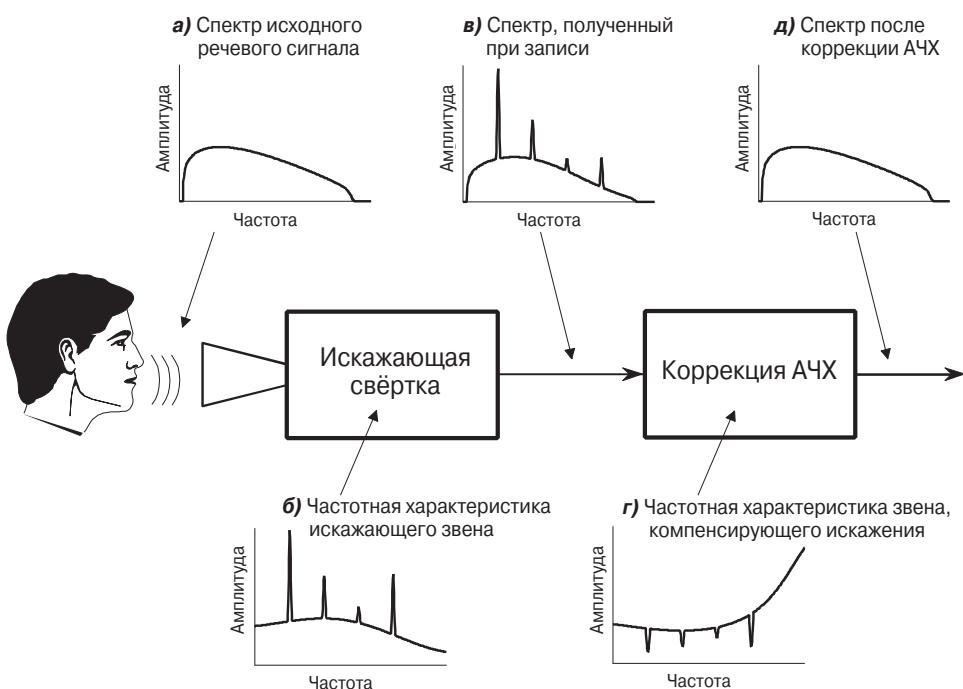
Даже если мы обладаем полной информацией об искажениях, описываемых свёрткой, то коррекция АЧХ все равно выполняется с ограниченной точностью, и причиной тому является наличие шума. Как правило, модель искажений, описываемых свёрткой, соответствует НЧ-фильтру, подавляющему высокочастотные компоненты сигнала. В ходе обращения свёртки требуется восстановить подавленные высокочастотные составляющие. Но если амплитуда этих составляющих оказалась ниже уровня собственного шума системы, информация на высоких частотах будет потеряна. Никакие алгоритмы обработки уже не помогут её восстановить. Она потеряна безвозвратно! Попытка восстановления данных приведёт только к увеличению мощности шума. В самом худшем случае амплитуда отдельных частотных компонент может упасть до нуля. В результате не только полностью исключается возможность передачи информации на этих частотах, но и может возникнуть попытка построения фильтра с бесконечным коэффициентом усиления на данных частотах. Решение проблемы заключается в снижении требований к точности, достигаемой при коррекции АЧХ, или во введении ограничений на максимальное значение коэффициента усиления фильтра.

К какому пределу нужно стремиться при восстановлении сигнала? И начиная с какого момента повышение качества восстановления сигнала становится «чрезмерным»? Все это зависит от решаемой задачи. Если сигнал отличается достаточно стабильным поведением и низок уровень шума, то можно достичь высокого качества восстановления (уменьшение ошибки в 5...10 раз). Если характер вносимых в сигнал искажений постоянно меняется и точно не известен или если сигнал сопровождается сильным шумом, то качество восстановления намного ниже (уменьшение ошибки только в 1...2 раза). Для того чтобы достичь успеха в восстановлении сигнала, необходимо провести экспериментальные исследования: если некоторый уровень качества уже достигнут, то можно попытаться достичь более высокого уровня и продолжать до тех пор, пока не будет достигнут предел. Нет теоретических рекомендаций, позволяющих обойтись без такой итеративной процедуры.

Метод коррекции АЧХ может применяться при восстановлении сигналов, информация в которых представлена в частотной области. Классическим примером является восстановление старых записей знаменитого оперного певца Энрико Карузо (1873 – 1921). По современным меркам эти записи были сделаны на ужасно примитивном оборудовании. В те далёкие годы основной проблемой при записи являлись резонансные частоты, возникающие в длинной граммофонной трубе, которая была необходима для усиления звука. Как только голос певца совпадал с какой-нибудь резонансной частотой, громкость звука на этой частоте резко усиливалась. С помощью цифровой коррекции АЧХ звукозаписывающего тракта можно улучшить субъективное качество звучания записи, уменьшив громкость звука на участках возникновения резонанса. Опишем далее ключевые моменты такого метода, предоставив читателю при желании самому познакомиться с этим вопросом более подробно<sup>1)</sup>.

<sup>1)</sup> Статья T. Stockham, T. Cannon, and R. Ingebretsen «Blind Deconvolution Through Digital Signal Processing», *Proc. IEEE*, vol. 63, Apr. 1975, pp. 678–692.

Общая идея метода проиллюстрирована на Рис. 17.6. На (а) показан спектр исходного речевого сигнала. АЧХ звукозаписывающей системы (б) имеет вид гладкой кривой во всём диапазоне частот, за исключением нескольких точек, которым соответствуют резонансные пики. Спектр записанного сигнала (в) получается путём умножения АЧХ звукозаписывающей системы на спектр входного сигнала. Цель коррекции АЧХ состоит в устранении нежелательного влияния искажений звукозаписывающей аппаратуры. Другими словами, частотная характеристика корректора АЧХ должна быть обратной по отношению к частотной характеристике, показанной на (б). Поэтому каждому пику на (б) соответствует провал на (в). Если восстановливающий фильтр является в точности обратным по отношению к искажающему фильтру, то получаемый на его выходе сигнал (д) совпадает с исходным сигналом — голосом певца. Тут обнаруживается одно принципиальное препятствие: устройство, при помощи которого проводилась звукозапись, давно уже перестало существовать, и поэтому его частотная характеристика неизвестна. Фактически в данном случае приходится решать задачу слепого выравнивания: известен только спектр записанного сигнала (в), необходимо получить восстановленный спектр (д).

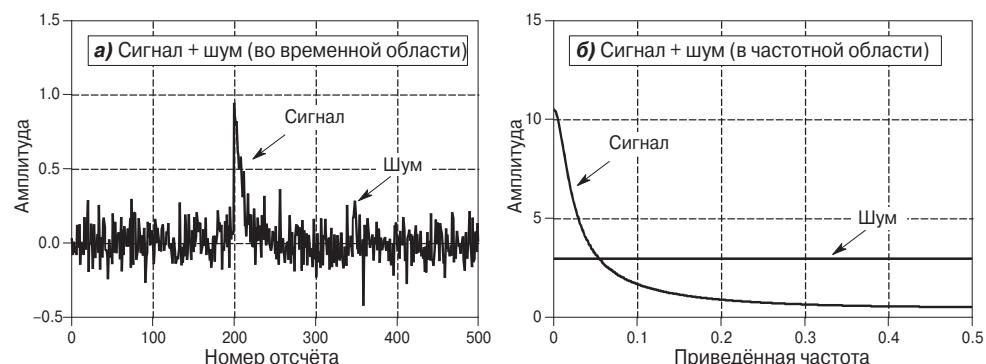


**Рис. 17.6.** Восстановление старинных граммофонных записей: (а) голос певца до записи; (б) резонансные искажения, внесённые примитивным устаревшим оборудованием; (в) искажённый в грамзаписи голос; (г) АЧХ фильтра, компенсирующего внесённые ранее искажения; (д) спектр восстановленного сигнала. Графики приводятся только с целью иллюстрации описанного в этом разделе алгоритма, но не соответствуют реальным сигналам.

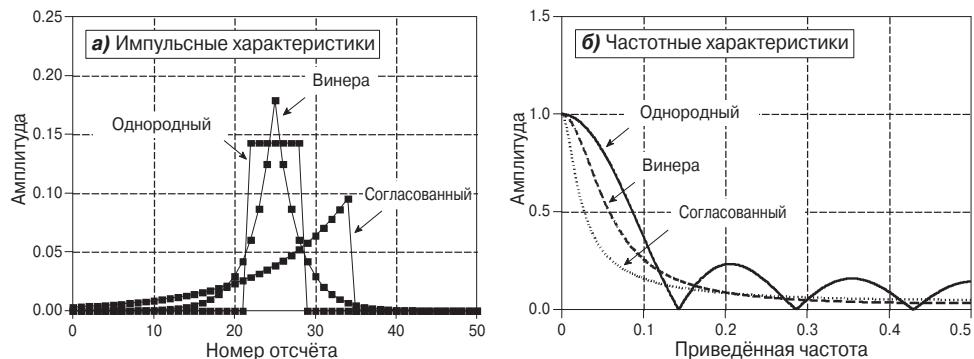
При решении задачи слепого выравнивания всегда требуется наличие предварительной оценки. В нашем случае вводится предположение, что усреднённый спектр той музыки, которая была записана много лет назад, не отличается от среднего спектра этой же музыкальной композиции, исполненной современным певцом. Методика получения усреднённого спектра изложена в Главе 9. Согласно этой методике сигнал разбивается на большое число сегментов, затем выполняется ДПФ каждого сегмента и вычисляется среднее значение амплитуд полученных компонентов спектра. Простейший способ нахождения неизвестной частотной характеристики — это деление соответствующих отсчётов усреднённого спектра старой искажённой записи на отсчёты усреднённого спектра качественной современной записи. (Методика, предложенная в упомянутой выше работе Стокхама (T. Stockham) и других учёных, основана на более сложном алгоритме, получившем название *гомоморфной обработки* и позволяющем достичь более точного измерения характеристик искажающей системы.)

## 17.3. ОПТИМАЛЬНАЯ ФИЛЬТРАЦИЯ

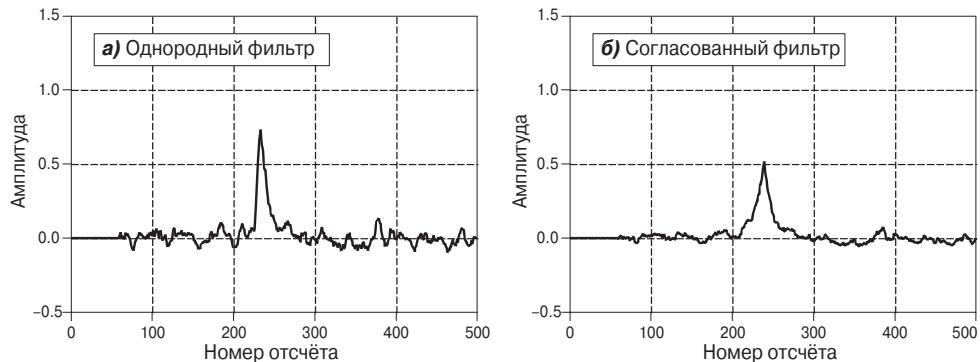
Одной из важнейших задач фильтрации является обнаружение импульса известной формы (в нашем примере — импульса с экспоненциальным спадом) на фоне мощной шумовой помехи (**Рис. 17.7а**). Судя по **Рис. 17.7б**, решение данной задачи при рассмотрении сигналов в частотной области оказывается не намного проще. Заметим, что спектр полезного сигнала сосредоточен преимущественно в низкочастотной области, в то время как спектральная плотность шума остаётся постоянной на протяжении всего частотного диапазона (*белый шум*). Возникает вопрос: как наилучшим образом разделить перекрывающиеся спектры полезного сигнала и помехи? Ответ на поставленный вопрос зависит от выбора критерия качества. Рассмотрим три фильтра, каждый из которых является по-своему «наилучшим», а, точнее, оптимальным. На **Рис. 17.8** изображены импульсные и частотные характеристики этих фильтров. Сравнить качество работы фильтров можно по графикам, приведенным на **Рис. 17.9** (входной сигнал показан на **Рис. 17.7а**).



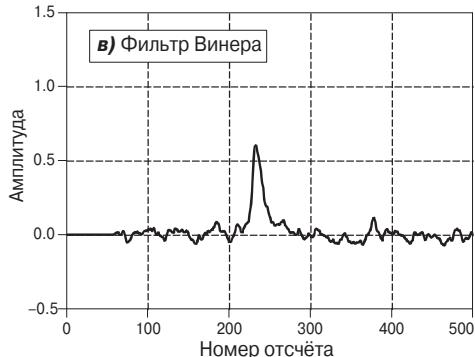
**Рис. 17.7.** К пояснению постановки задачи оптимальной фильтрации. **а)** Импульс с экспоненциальным спадом, сопровождаемый шумовой помехой. **б)** Спектры полезного сигнала и шумовой помехи. Поскольку сигнал и помеха перекрываются как во временной, так и в частотной областях, то выбор метода их разделения не очевиден.



**Рис. 17.8.** Примеры оптимальных фильтров. **а)** Импульсные характеристики трёх фильтров, каждый из которых является оптимальным по некоторому критерию. **б)** Частотные характеристики тех же фильтров. Импульсная характеристика однородного фильтра имеет вид прямоугольного импульса. Импульсная характеристика согласованного фильтра повторяет форму обнаруживаемого импульса. При построении АЧХ фильтра Винера учитывается только отношение сигнал/шум на соответствующей частоте.



**Рис. 17.9.** Реакции на сигнал, показанный на Рис. 17.7, которые получены на выходе фильтров, изображенных на Рис. 17.8. Каждый из трёх фильтров оптимален по своему критерию. **а)** Однородный фильтр обеспечивает при заданном уровне подавления шума самое быстрое нарастание переднего фронта импульса, полученного после выполнения фильтрации. **б)** Согласованный фильтр позволяет получить самое большое отношение пикового значения амплитуды импульса к уровню шума. **в)** С помощью фильтра Винера удается достичь наибольшего отношения сигнал/шум.



*Однородный фильтр* подробно рассмотрен в Главе 15. Напомним, что каждый очередной отсчёт выходного сигнала однородного фильтра получается в результате усреднения некоторого числа последовательных отсчётов входного сигнала. То есть импульсная характеристика такого фильтра имеет вид прямоугольного импульса,

амплитуда которого обратно пропорциональна числу усредняемых отсчётов. Однородный фильтр оптимален в том смысле, что позволяет достичь минимальной длительности переходного процесса при заданном уровне подавления шума.

*Согласованный фильтр* был рассмотрен в Главе 7. Импульсная характеристика согласованного фильтра в точности повторяет форму обнаруживаемого импульса, являясь его зеркальной копией (развернута слева направо). Суть согласованной фильтрации заключается в корреляционном анализе, а чтобы найти корреляционную функцию путём выполнения свёртки, требуется «развернуть» импульс слева направо. Каждый отсчёт полученного выходного сигнала показывает, насколько велика степень сходства импульсной характеристики фильтра с некоторым участком входного сигнала. Напомним, что полученный с помощью согласованного фильтра сигнал отличается по форме от обнаруживаемого импульса. Но это вовсе не является недостатком, т. к. при обнаружении импульса согласованным фильтром предполагается, что форма этого импульса заранее известна. Согласованный фильтр (**Рис. 17.9б**) оптимален в том смысле, что позволяет достичь наибольшего пикового значения амплитуды выходного импульса при фиксированной мощности шума по сравнению с любым другим фильтром.

*Фильтр Винера* (назван в честь Норберта Винера, разработавшего теорию оптимального оценивания) разделяет сигналы в соответствии с их спектральными характеристиками. Как показано на **Рис. 17.7б**, на некоторых частотах «преобладает» сигнал, а на других — шум. С этой позиции вполне логичным оказывается такой подход к построению фильтра, когда фильтр пропускает на выход те частоты, на которых преобладает сигнал, и подавляет те, на которых преобладает шум. В методике построения фильтра Винера используется дальнейшее развитие данной идеи. Коэффициент усиления фильтра определяется на каждой частоте как отношение мощности полезного сигнала к суммарной мощности сигнала и шума на данной частоте:

$$H[f] = \frac{S[f]^2}{S[f]^2 + N[f]^2}. \quad (17.1)$$

Фильтр Винера. АЧХ фильтра Винера  $H[f]$  выражается через спектральные плотности мощности помехи  $N[f]$  и полезного сигнала  $S[f]$ . При этом учитываются только амплитуды, а ФЧХ фильтра Винера равна нулю.

С помощью выражения (17.1) можно определить АЧХ фильтра Винера (**Рис. 17.8б**), если известны спектральные плотности мощности полезного сигнала и помехи (**Рис. 17.7б**). Фильтр Винера оптимален с позиции получения максимального отношения мощности сигнала к мощности шума (мощность оценивается не для каждой частоты в отдельности, а во всём диапазоне частот). Чтобы найти весовые коэффициенты (отсчёты импульсной характеристики) фильтра Винера, применяется процедура расчёта специальных фильтров.

Хотя сама идея построения оптимальных фильтров на первый взгляд кажется привлекательной, на практике от неё часто приходится отказываться. Это вовсе не означает, что описанные выше фильтры никогда не используются. Просто всегда следует помнить, что встретившееся в книге слово «оптимальный» ещё не говорит о том, что процесс решения задачи полностью завершён и можно поставить точку.

Во-первых, отсутствуют ярко выраженные различия между сигналами, показанными на **Рис. 17.9**. То есть если вам неизвестно заранее, по какому критерию

проводилась оптимизация, то вам вряд ли удастся определить это по предлагающим изображениям. Отсутствие характерных различий связано с *наложением спектров*. Небольшое увеличение качества оптимизации достигается ценой усложнения программы фильтрации, дополнительных затрат в процессе расчёта коэффициентов фильтра, а также возрастанием времени обработки сигнала.

Во-вторых, параметры согласованного фильтра и фильтра Винера жёстко определяются условиями решаемой задачи, в то время как параметры других фильтров, таких как оконные или однородные, мы можем корректировать в зависимости от собственных предпочтений. Оптимальность расчёта предполагает, что любые отклонения могут только понизить эффективность фильтра. Это очень важный момент: каждый оптимальный фильтр является оптимальным только по определённому критерию. Возможность утверждать, что найдено «абсолютно оптимальное» решение задачи, в том смысле, что принятый критерий является наилучшим, появляется очень редко. Каждый человек оценивает оптимальность полученного решения в соответствии с личными предпочтениями. Например, при разработке медицинской аппаратуры технический специалист может отдать предпочтение фильтру Винера, позволяющему достичь наибольшего отношения сигнал/шум при снятии электрокардиограммы. Однако выбор такого фильтра вовсе не говорит о том, что полученный с его помощью сигнал будет оптимальен для врача и предоставляет ему наилучшую возможность обнаружения неполадок в работе сердца.

В-третьих, согласованный фильтр и фильтр Винера основаны на выполнении очень трудоёмкой в вычислительном плане операции свёртки. Даже при использовании быстрых алгоритмов (быстрая свёртка), речь о которых пойдёт в следующей главе, вычислительные затраты остаются большими. Использование рекурсивных фильтров (как однородных рекурсивных фильтров, так и других, о которых говорится в Главе 19) позволяет значительно снизить необходимые вычислительные затраты при достижении приемлемого качества фильтрации.

# Глава 18

---

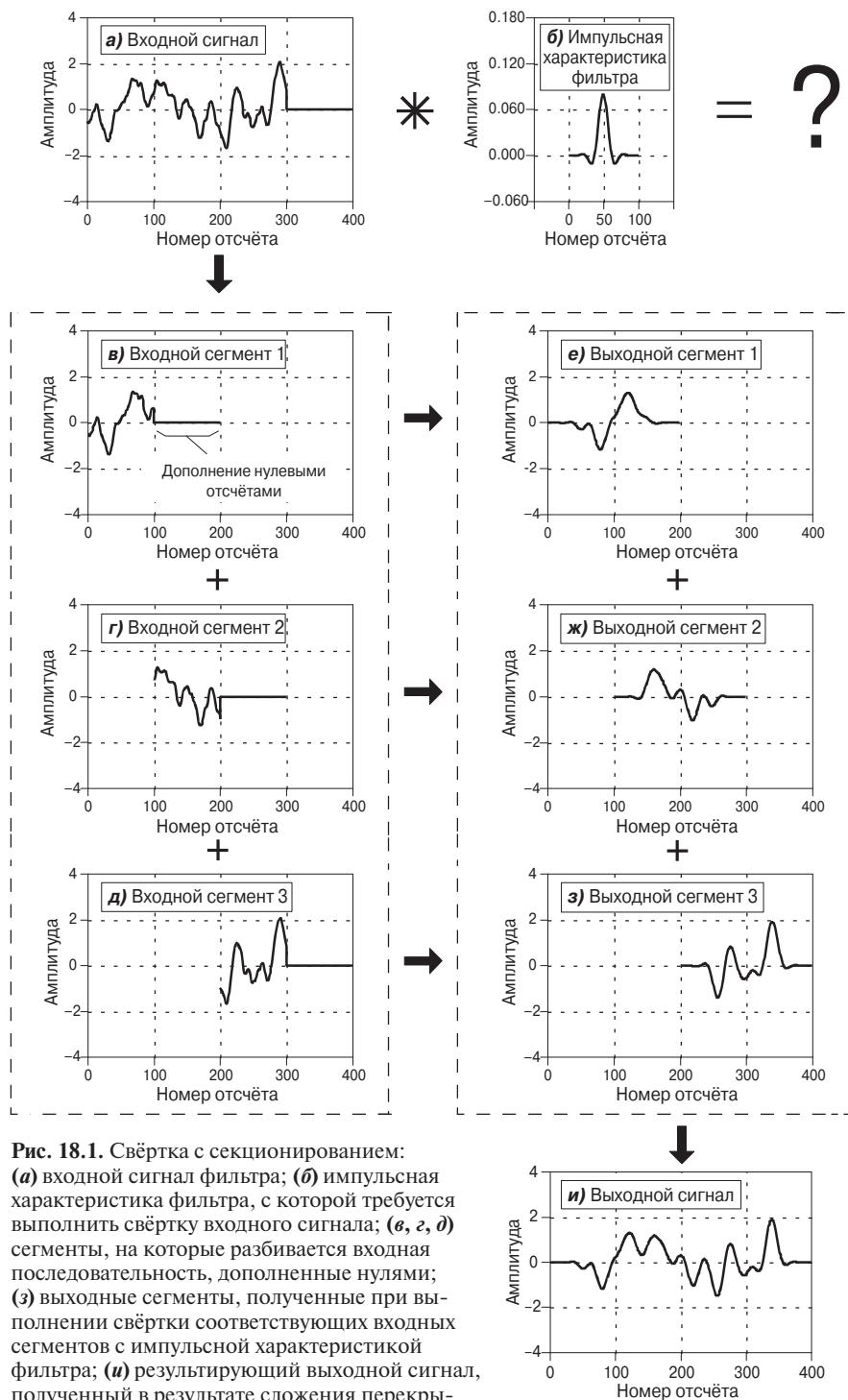
## БЫСТРАЯ СВЁРТКА

В этой главе рассказывается о двух процедурах, играющих важнейшую роль в ЦОС: *свёртке с секционированием* и *быстрой свёртке*. Секционирование используется для разбиения длинных последовательностей отсчётов на сегменты (секции) меньшей длины с целью упрощения дальнейшей обработки. Алгоритм быстрой свёртки, основанный на *быстром преобразовании Фурье* и использующий секционирование, позволяет заменить операцию свёртки умножением частотных спектров соответствующих сигналов. Для фильтров, порядок которых превышает 64, переход к быстрой свёртке значительно сокращает объём необходимых вычислительных затрат, при этом все характеристики фильтров остаются неизменными.

### 18.1. Свёртка с секционированием

Во многих практических приложениях возникает необходимость обработки дискретных сигналов, имеющих значительную протяжённость во времени. Например, в современных мультимедийных системах каждую минуту приходится обрабатывать около 5 МБ аудиоданных и около 500 МБ видеоданных. При таких скоростях обычным вычислительным устройствам не хватает памяти для хранения всех отсчётов сигнала, подлежащих обработке. Существует и другая проблема: во многих системах обработка сигнала производится в режиме *реального времени*. Например, в телефонии допускается задержка сигнала в системе обработки всего на несколько сотых долей секунды. Это также ограничивает максимальную длину обрабатываемого сегмента данных. Кроме того, имеются разнообразные алгоритмы, в которых обработку сигналов просто необходимо осуществлять отдельными фрагментами. К таким алгоритмам, в частности, относится алгоритм быстрой свёртки, о котором рассказывается в этой главе.

В самом общем виде *свёртку с секционированием* можно разделить на три этапа: 1) разбиение сигнала на отдельные секции, выполняемое с перекрытием; 2) обработка каждого из полученных фрагментов в отдельности; 3) объединение секций в единый выходной сигнал. Механизм секционированной свёртки проиллюстрирован на Рис. 18.1. Необходимо выполнить обработку некоторого сигнала (**a**) оконным НЧ-фильтром (**b**). Сигнал, полученный на выходе фильтра, изображён на самом нижнем графике (**u**) и представляет собой слаженный входной сигнал (**a**). Один из вопросов, возникающих при реализации свёртки с секционированием, связан с учётом длин отдельных сегментов при объединении результатов обработки в одну выходную последовательность. Свёртка  $N$  отсчётов входной последовательности с *импульсной характеристикой* фильтра  $M$ -го порядка даёт  $(N + M - 1)$  отсчёт выходной последовательности. Если входной сигнал (**a**) пред-



**Рис. 18.1.** Свёртка с секционированием:  
**(а)** входной сигнал фильтра; **(б)** импульсная характеристика фильтра, с которой требуется выполнить свёртку входного сигнала; **(в, г, д)** сегменты, на которые разбивается входная последовательность, дополненные нулями; **(з)** выходные сегменты, полученные при выполнении свёртки соответствующими входным сегментам с импульсной характеристикой фильтра; **(и)** результирующий выходной сигнал, полученный в результате сложения перекрывающихся выходных сегментов.

ставлен 300 отсчётаами (0...299), а импульсная характеристика (**б**) задана в 101 точке (0...100), то выходной сигнал (**и**) содержит 400 отсчётов (0...399).

Таким образом, фильтрация входного сигнала, состоящего из  $N$  отсчётов, приводит к его «удлинению» на выходе фильтра на  $(M + 1)$  отсчёт вправо. Предполагается, что порядок фильтра равен  $(M + 1)$ , где  $M$  — максимальный индекс весового коэффициента, когда нумерация начинается с нуля. Если имеются коэффициенты с отрицательными индексами, то выходная последовательность «удлиняется» в левую сторону. На (**а**) входной сигнал дополнен нулевыми отсчётыами с индексами 300...399, чтобы подчеркнуть эффект «удлинения» выходной последовательности. Постепенное уменьшение амплитуды на краях выходной последовательности (**и**) связано с формой импульсной характеристики оконного НЧ-фильтра. Все 400 отсчётов выходной последовательности (**и**) отличны от нуля, хотя на графике это отличие может быть незаметным.

На (**в**, **г** и **д**) показаны отдельные секции, на которые разбивается входной сигнал. Секционирование производится с перекрытием. Каждый из полученных сегментов содержит 100 отсчётов исходной последовательности, к которым добавляются справа 100 нулевых отсчётов. Все сегменты обрабатываются отдельно от других путём свёртки с импульсной характеристикой фильтра, при этом получаются отдельные компоненты выходной последовательности (**е**, **ж** и **з**). Так как длина входных сегментов составляет 100 отсчётов, а порядок фильтра равен 101, то длина выходных сегментов равна 200 отсчётом. Увеличение каждого сегмента на 100 отсчётов происходит в результате выполнения операции свёртки.

Следует заметить, что удлинение сегментов в дальнейшем приведет к их перекрытию друг с другом. Чтобы получить отсчёты выходной последовательности (**и**), требуется сложить соответствующие отсчёты полученных сегментов. Например, отсчёты с индексами 200...299 получаются после сложения соответствующих отсчётов сегментов, показанных на (**ж** и **з**). Секционированная свёртка с перекрытием позволяет получить точно такой же результат, как и при непосредственном выполнении свёртки (без разбиения на сегменты). Недостатком описанного метода является увеличение вычислительных затрат, связанное с обработкой перекрывающихся (избыточных) фрагментов секций.

## 18.2. Быстрая свёртка

В основе алгоритма *быстрой свёртки* лежит принцип соответствия свёртки во временной области умножению в частотной области. С помощью ДПФ входной сигнал переводится в частотную область, где он умножается на частотную характеристику фильтра, после чего вновь переносится во временную область с помощью обратного ДПФ. Этот подход был известен со времён Фурье, но никто им не пользовался, т. к. считалось, что вычисление ДПФ требует большего числа арифметических операций, чем обычная свёртка. Ситуация изменилась после разработки в 1965 году алгоритма *быстрого преобразования Фурье (БПФ)*. Алгоритм БПФ позволил значительно снизить вычислительную сложность операций прямого и обратного ДПФ, а следовательно, и свёртки в частотной области. Проведённые в дальнейшем многочисленные исследования показали, что основным достоинством

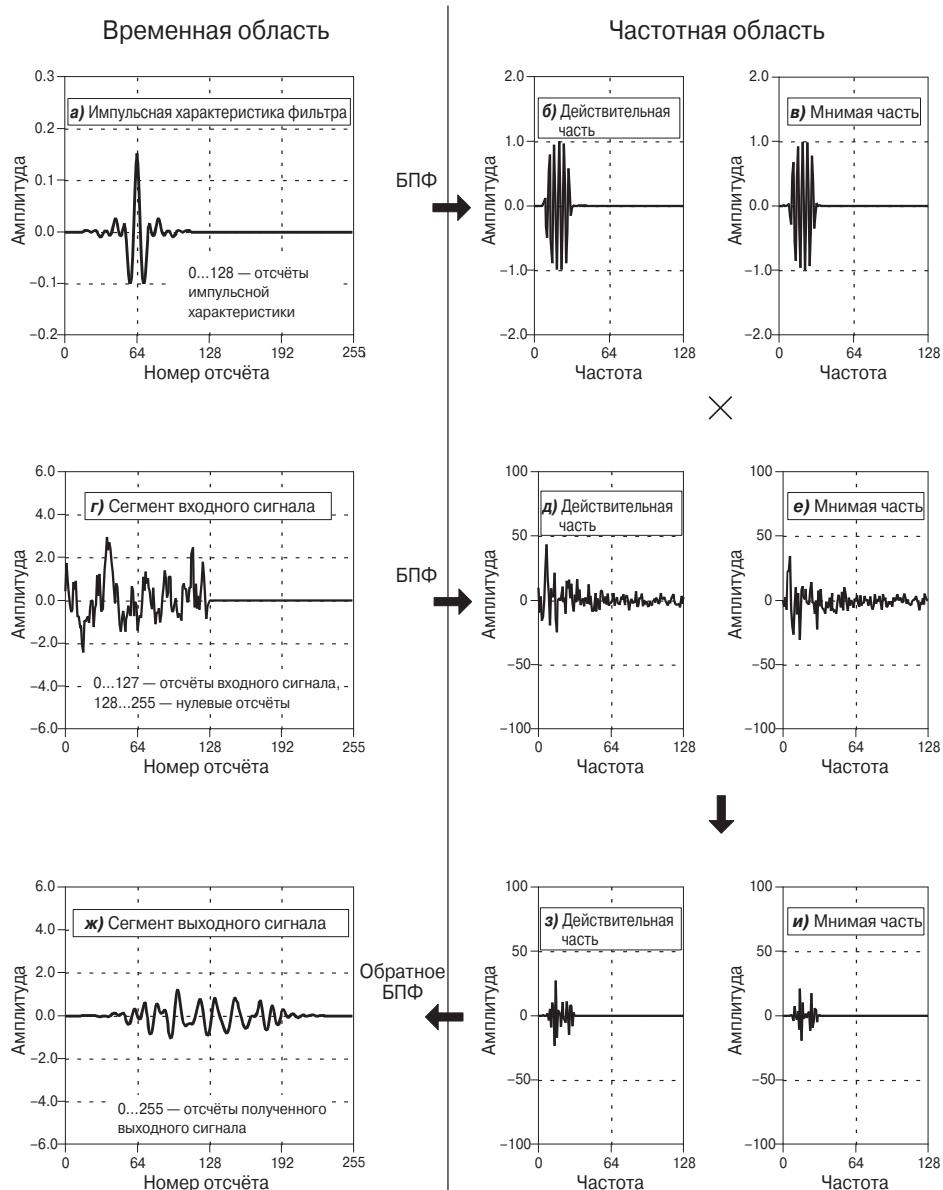
использования БПФ является повышение вычислительной эффективности. Поэтому свёртка на основе БПФ получила название *быстрой свёртки*.

Алгоритм быстрой свёртки использует *секционирование* с перекрытием (Рис. 18.1). Процесс преобразования входных секций сигнала в выходные сегменты проиллюстрирован на Рис. 18.2. Прежде всего находят частотную характеристику фильтра, которая, как известно, связана с импульсной характеристикой преобразованием Фурье. Для этого применяют алгоритм БПФ.

Пусть задана импульсная характеристика оконного полосового фильтра (**а**). С помощью БПФ найдём частотную характеристику, которая может быть представлена как совокупность действительной и мнимой частей (**б** и **в**). Частотная характеристика представлена в виде, сильно отличающемся от привычной нам формы АЧХ полосового фильтра, потому что она представлена в прямоугольной, а не полярной системе координат (напомним, что для человеческого восприятия более удобна полярная форма представления характеристик, а для проведения математических операций обычно больше подходит прямоугольная форма). Найденные действительная и мнимая части частотной характеристики хранятся в памяти компьютера постоянно и используются при обработке каждого очередного сегмента поступающих данных.

Порядок БПФ должен быть выбран достаточно большим, чтобы исключить возможное влияние *циклической свёртки* (он должен быть равен длине выходного сегмента, см. Главу 9). В примере, рассмотренном на Рис. 18.2, порядок фильтра равен 129, а входной сигнал разбивается на сегменты по 128 отсчётов, поэтому длина выходного сегмента равна 256 отсчётом. Значит, необходимо использовать БПФ 256-го порядка. То есть импульсная характеристика должна дополняться 127 нулевыми отсчётыами (**а**), а каждый сегмент входного сигнала — 128 нулевыми отсчётыами (**г**). Приведём другой пример. Пусть длина импульсной характеристики фильтра равна 600 отсчётом. В таком случае входной сигнал можно разбивать на сегменты длиной по 425 отсчётов, а порядок БПФ выбрать равным 1024. Но существуют и альтернативные варианты: например, увеличив порядок БПФ до 2048, можно увеличить длину сегментов до 1449 отсчётов.

**Программа 18.1** представляет собой пример программы, выполняющей быструю свёртку. В этом примере выполняется обработка входного сигнала длиной 10 миллионов отсчётов фильтром 400-го порядка. Входной сигнал разбивается на 16 000 сегментов, по 625 отсчётов в каждом. В результате свёртки каждого из сегментов с импульсной характеристикой фильтра получается  $625 + 400 - 1 = 1024$  отсчёта выходного сигнала, т. е. порядок БПФ равен 1024. После определения и инициализации всех необходимых массивов (строки 130...230) на первом шаге алгоритма вычисляется частотная характеристика фильтра (строки 250...310) и сохраняется в памяти для последующего использования. В строке 260 вызывается некоторая подпрограмма, сохраняющая отсчёты импульсной характеристики в ячейках памяти XX[0]...XX[399] и заполняющая ячейки памяти XX[400]...XX[1023] нулями. В строке 270 вызывается ещё одна подпрограмма, выполняющая БПФ с массивом XX[ ] и сохраняющая действительные и мнимые части полученных 513 отсчётов комплексной частотной характеристики в массивах REX[ ] и IMX[ ] соответственно. Затем эти отсчёты переписываются в массивы REFR[ ] и IMFIR[ ] (от англ. «REal and IMaginary Frequency Response»), где они будут храниться постоянно.



**Рис. 18.2.** Алгоритм быстрой свёртки. Переход в частотную область позволяет заменить свёртку умножением. **а)** Импульсная характеристика фильтра. **б** и **в)** Действительная и мнимая части соответствующей частотной характеристики фильтра. **г)** Сегмент входной последовательности. **д** и **е)** Действительная и мнимая части спектра сегмента входного сигнала. **з** и **и)** Действительная и мнимая части спектра сегмента выходного сигнала, найденные после умножения в частотной области. **ж)** Сегмент выходного сигнала, полученный в результате выполнения алгоритма быстрой свёртки после заключительной операции обратного БПФ.

Цикл FOR–NEXT в строках 340...580 служит для обработки 16 000 сегментов входного сигнала. Подпрограмма, вызываемая в строке 360, записывает в ячейки XX[0]...XX[624] отсёты каждого очередного сегмента, а в ячейки XX[625]...X[1023] — нули. Затем из строки 370 снова вызывается подпрограмма, выполняющая БПФ массива XX[ ] и сохраняющая действительные и мнимые части полученных 513 отсётов комплексной частотной характеристики в массивах REX[ ] и IMX[ ]. В строках 390...430 выполняется процедура умножения частотного спектра сегмента входного сигнала (REX[ ] и IMX[ ]) на частотную характеристику фильтра (REFR[ ] и IMFR[ ]). Сразу после умножения результат размещается в REX[ ] и IMX[ ], заменяя данные, записанные ранее в эти массивы. Результатом умножения является спектр выходного сегмента, поэтому для перехода во временную область требуется выполнить обратное БПФ. Подпрограмма, выполняющая обратное БПФ, вызывается в строке 450. Входными данными этой подпрограммы являются массивы REX[ ] и IMX[ ], в которых к этому моменту размещаются действительные и мнимые части спектра выходного сегмента. 1024 отсёта выходного сегмента сохраняются в массиве XX[ ].

В строках 470...550 осуществляется совмещение перекрывающихся выходных сегментов в единый массив выходного сигнала. Каждый сегмент разбивается на две части. Первые 625 отсётов (0...624) складываются с отсётом предыдущего выходного сегмента, а потом записываются в массив выходного сигнала. Оставшиеся 399 отсётов (625...1023) временно сохраняются таким образом, чтобы на следующей итерации можно было сложить их с отсётом следующего сегмента, с которыми они перекрываются.

Поясним суть сказанного на примере, проиллюстрированном на **Рис. 18.1**. Отсёты с номерами 100...199 на (**ж**) складываются с соответствующими отсётом предыдущего выходного сегмента (**е**), а отсёты с номерами 200...299 сохраняются в буфере до тех пор, пока они не будут прибавлены к отсётом следующего сегмента (**з**).

Теперь вновь вернёмся к программе. Массив OLAP[ ] используется для временного хранения 399 отсётов сегмента, которые необходимо сложить с отсётом следующего выходного сегмента. В строках 470...490 описано сложение элементов массива OLAP[ ], содержащего отсёты предыдущего сегмента, с элементами массива XX[ ], в котором хранятся отсёты текущего сегмента. Некоторая подпрограмма, вызываемая в строке 550, выводит 625 отсётов, хранящихся в ячейках XX[0]...XX[624], в файл отсётов выходного сигнала. Последние 399 отсётов текущего сегмента сохраняются до следующей итерации цикла обработки в массиве OLAP[ ] (строки 510...530).

После окончания обработки сегментов с номерами 0...15999 в массиве OLAP[ ] остаются отсёты выходного сигнала, которые следует сложить с отсётом сегмента номер 16000. Но сегмента с таким порядковым номером в обрабатываемом сигнале нет (можно считать, что все его отсёты равны нулю), поэтому в строке 600 эти отсёты переписываются в массив выходного сигнала без изменений. В результате проведенной обработки длина выходного сигнала равна  $16\ 000 \times 625 + 399 = 10\ 000\ 399$  отсётом, что соответствует длине входного сигнала, сложенной с порядком фильтра и уменьшенной на единицу.

**Программа 18.1**

```

100 'БЫСТРАЯ СВЁРТКА
105 'Данная программа позволяет выполнить свёртку 10 000 000 отсчётов входного
110 'сигнала с 400 отсчётами импульсной характеристики. Входной сигнал
115 'разбивается на 16 000 сегментов по 625 отсчётов. Порядок БПФ равен 1024.
120 '
130 ' ИНИЦИАЛИЗАЦИЯ МАССИВОВ
140 DIM XX[1023]      'Массив отсчётов сигнала (для БПФ)
150 DIM REX[512]        'Массив действительных частей отсчётов (для БПФ)
160 DIM IMX[512]        'Массив мнимых частей отсчётов (для БПФ)
170 DIM REFR[512]       'Массив действительных компонент частотной характеристики
180 DIM IMFR[512]       'Массив мнимых компонент частотной характеристики
190 DIM OLAP[398]       'Массив временного хранения перекрывающихся отсчётов
200 '
210 FOR I% = 0 TO 398   'Обнуление массива перекрывающихся отсчётов
220 OLAP[I%] = 0
230 NEXT I%
240 '
250 ' ВЫЧИСЛЕНИЕ И СОХРАНЕНИЕ ЧАСТОТНОЙ ХАРАКТЕРИСТИКИ
260 GOSUB XXXX          'Подпрограмма, сохраняющая импульсную характеристику в XX[]
270 GOSUB XXXX          'Подпрограмма вычисления БПФ: XX[] --> REX[] и IMX[]
280 FOR F% = 0 TO 512   'Сохранение частотной характеристики в REFR[] и IMFR[]
290 REFR[F%] = REX[F%]
300 IMFR[F%] = IMX[F%]
310 NEXT F%
320 '
330 ' ПООЧЕРЁДНАЯ ОБРАБОТКА 16 000 СЕГМЕНТОВ
340 FOR SEGMENT% = 0 TO 15999
350 '
360 GOSUB XXXX          'Подпрограмма, загружающая очередной сегмент в XX[]
370 GOSUB XXXX          'Подпрограмма вычисления БПФ: XX[] --> REX[] и IMX[]
380 '
390 FOR F% = 0 TO 512   'Умножение спектра сигнала на частотную характеристику
400 TEMP = REX[F%]*REFR[F%] - IMX[F%]*IMFR[F%]
410 IMX[F%] = REX[F%]*IMFR[F%] + IMX[F%]*REFR[F%]
420 REX[F%] = TEMP
430 NEXT F%
440 '
450 GOSUB XXXX          'Подпрограмма вычисления ОБПФ: REX[] и IMX[] --> XX[]
460 '
470 FOR I% = 0 TO 624   'Сложение с отсчётами предыдущего сегмента
480 XX[I%] = XX[I%] + OLAP[I%]
490 NEXT I%
500 '
510 FOR I% = 625 TO 1023 'Сохранение отсчётов, образующих перекрытие
520 OLAP[I%-625] = XX[I%]
530 NEXT I%
540 '
550 GOSUB XXXX          'Подпрограмма вывода 625 отсчётов выходного сигнала,
560 ' хранящихся в ячейках XX[0]...XX[624]
570 '

```

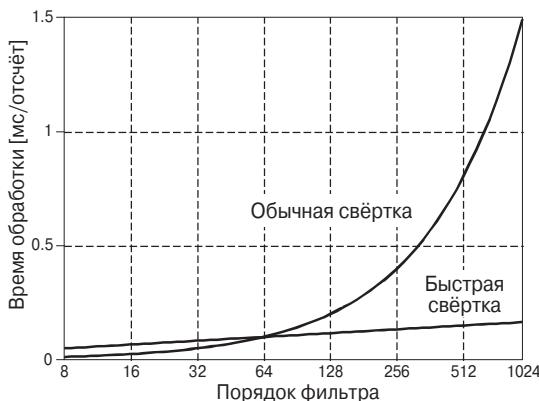
```

580 NEXT SEGMENT%
590 '
600 GOSUB XXXX      'Подпрограмма вывода 399 отсчётов массива OLAP[]
610 END

```

## 18.3. Сокращение вычислительных затрат

В каких случаях алгоритм БПФ ускоряет вычисление *свёртки*? Всё зависит от порядка фильтра (**Рис. 18.3**). Вычислительные затраты на выполнение свёртки обычным методом возрастают пропорционально порядку фильтра, в то время как затраты на выполнение *быстрой свёртки* увеличиваются намного медленнее и связаны с порядком фильтра логарифмической зависимостью. Пересечение графиков происходит при порядке фильтра, равном примерно 40...80 (более точное значение определяется используемой элементной базой).



**Рис. 18.3.** Вычислительные затраты на реализацию алгоритма быстрой свёртки. Благодаря использованию БПФ, быстрая свёртка позволяет существенно сократить объём вычислительных затрат по сравнению с обычным методом вычисления свёртки во временной области. Преимущество быстрой свёртки начинает проявляться, как только порядок фильтра превышает 60. Время обработки оценивалось для вычислений в формате с плавающей точкой, выполняемых на процессоре Pentium с тактовой частотой 100 МГц.

По **Рис. 18.3** можно сделать следующие важные на практике выводы. При реализации фильтров, порядок которых не превышает 60, предпочтение следует отдать обычному методу реализации свёртки, при котором объём вычислительных затрат на каждый отсчёт сигнала пропорционален порядку фильтра. Для фильтров более высокого порядка выгоднее воспользоваться алгоритмом быстрой свёртки, при этом зависимость становится логарифмической и затраты растут гораздо медленнее. Например, увеличение порядка фильтра с 64-го до 16 000-го требует всего лишь в 2 раза большего объёма вычислений.

Другой положительный эффект, наблюдающийся вследствие перехода к быстрой свёртке, выражается в повышении точности обработки (о чём говорилось в Главе 12). Дело в том, что общая ошибка, вызванная округлением, зависит от количества выполняемых арифметических операций, а следовательно, пропорциональна времени, затраченному на обработку. Таким образом, чем быстрее произ-

ведены вычисления, тем точнее полученный результат. Предположим, что порядок фильтра равен 1000, а вычисления проводятся в формате с плавающей точкой с одинарной точностью. Тогда в случае обычной свёртки *ошибка округления* составит приблизительно 1/20000 (см. Главу 4), а в случае быстрой свёртки она будет на порядок меньше, т. е. приблизительно равна 1/200000.

Алгоритм быстрой свёртки нужно всегда «держать под рукой» на тот случай, если потребуется фильтр очень высокого порядка. Этот алгоритм позволяет легко оперировать с массивами входного сигнала, содержащими миллионы отсчётов, и с фильтрами, порядки которых выражаются тысячами, позволяя избежать неоправданно больших затрат. Если вам не хочется писать программу быстрой свёртки самостоятельно, можно воспользоваться готовыми библиотеками функций.

## РЕКУРСИВНЫЕ ФИЛЬТРЫ

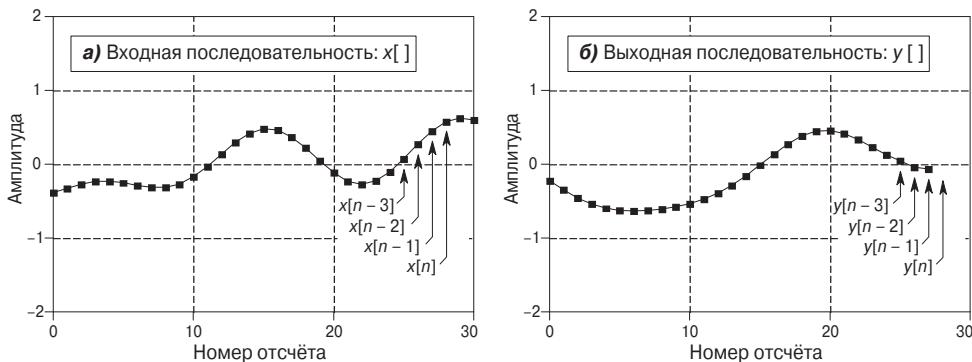
*Рекурсивные фильтры*, в отличие от нерекурсивных, не используют операцию *свёртки* временных последовательностей, требующую значительных вычислительных затрат. Поэтому их применение повышает вычислительную эффективность алгоритмов фильтрации, но при этом могут возникать значительные ограничения по реально достижимой точности воспроизведения желаемых частотных характеристик. Рекурсивные фильтры относятся к классу цепей с *бесконечной импульсной характеристикой (БИХ)*. Импульсная характеристика БИХ-фильтров складывается из множества синусоид, амплитуда которых убывает по экспоненциальному закону. Форма импульсной характеристики является основным отличием рекурсивных фильтров от фильтров, построенных на основе свёртки, которые образуют, в свою очередь, класс цепей с *конечной импульсной характеристикой (КИХ)*. Данная глава представляет собой введение в теорию рекурсивных фильтров. В ней также описана методика расчёта некоторых простых фильтров этого семейства. Более сложным методам посвящены Главы 20, 26 и 33.

### 19.1. Рекурсивный метод

Предположим, вам необходимо выделить информационную составляющую, содержащуюся в сигнале  $x[n]$ . Для вас это очень важно, и вы обращаетесь к очень крупному специалисту, профессору, доктору математики, чтобы он помог вам в обработке имеющихся данных. Задача профессора состоит в том, чтобы в результате фильтрации получить из сигнала  $x[n]$  сигнал  $y[n]$ , в котором информация содержится в форме, доступной для восприятия. Профессор приступает к последовательному вычислению отсчётов сигнала  $y[n]$ , руководствуясь каким-то одному ему известным алгоритмом, который он нашёл где-то в глубинах своих чрезвычайно обширных научных знаний. И вдруг на очередном этапе вычислений происходит неприятное событие. Профессор начинает бессвязно бормотать что-то, упоминая частные преобразования, аналитические сингулярности и других «чудовищ» из математического «кошмара». Совершенно ясно, что профессор «тронулся умом». С тревогой вы наблюдаете за тем, как его, а вместе с ним и этот алгоритм увозят люди в белых халатах.

В отчаянии просматривая записи профессора, пытаясь найти там используемый им алгоритм, вы обнаруживаете, что он уже вычислил отсчёты  $y[0] \dots y[27]$  и собрался рассчитать  $y[28]$ . Обозначим через  $n$  номер отсчёта, вычисляемого на текущей итерации (**Рис. 19.1**). Тогда если  $y[n]$  — 28-й отсчёт выходной последовательности, то  $y[n-1]$  — 27-й отсчёт,  $y[n-2]$  — 26-й отсчёт и т. д. Аналогично  $x[n]$  — 28-й отсчёт входной последовательности,  $x[n-1]$  — 27-й и т. д. Чтобы ра-

зобраться в алгоритме, попытаемся ответить на вопрос: какая информация была доступна профессору при вычислении отсчёта  $y[n]$  на последней выполняемой им итерации?



**Рис. 19.1.** Иллюстрация работы рекурсивного фильтра. Для вычисления текущего отсчёта выходного сигнала используются как отсчёты входной последовательности:  $x[n]$ ,  $x[n - 1]$ ,  $x[n - 2]$  и т. д., так и отсчёты выходной последовательности:  $y[n - 1]$ ,  $y[n - 2]$ ,  $y[n - 3]$  и т. д. Номер вычисляемого на текущей итерации отсчёта  $n = 28$ .

Самым очевидным источником данных являются отсчёты входной последовательности:  $x[n]$ ,  $x[n - 1]$ ,  $x[n - 2]$ , ... Профессор мог бы умножить каждый из них на соответствующий весовой коэффициент и сложить:

$$y[n] = a_0x[n] + a_1x[n - 1] + a_2x[n - 2] + a_3x[n - 3] + \dots$$

Вы конечно же сразу узнали это выражение. Это не что иное, как операция *свёртки*, в которой весовые коэффициенты  $a_0$ ,  $a_1$ ,  $a_2$ , ... играют роль одной из сворачиваемых последовательностей. Если бы только эту операцию использовал профессор, то нам бы не пришлось придумывать такую историю, и вся эта глава тоже не появилась бы в книге. Но профессор пользовался и другим источником данных — предыдущими отсчётами выходного сигнала:  $y[n - 1]$ ,  $y[n - 2]$ ,  $y[n - 3]$ , ... . В результате учёта дополнительной информации алгоритм приобрёл новый вид:

$$\begin{aligned} y[n] = & a_0x[n] + a_1x[n - 1] + a_2x[n - 2] + a_3x[n - 3] + \dots, \\ & + b_1y[n - 1] + b_2y[n - 2] + b_3y[n - 3] + \dots. \end{aligned} \tag{19.1}$$

Уравнение рекурсивного фильтра. Здесь  $x[ ]$  — входной сигнал,  $y[ ]$  — выходной сигнал,  $a$  и  $b$  — весовые коэффициенты прямой и обратной связей.

Теперь значение каждого очередного отсчёта выходной последовательности вычисляется путём умножения отсчётов входного сигнала на коэффициенты  $a_i$  и умножения полученных на предыдущих итерациях отсчётов выходного сигнала на коэффициенты  $b_i$  с последующим суммированием всех произведений. Заметьте, что коэффициент  $b_0$  не входит в выражение (19.1), так как соответствует текущему отсчёту выходного сигнала. Фильтры, описываемые выражением (19.1), называют *рекурсивными фильтрами*. Коэффициенты  $a_i$  и  $b_i$  являются его весовыми коэффициентами. Порядок рекурсивных фильтров, как правило, не

превышает 12, что связано с проблемой *устойчивости* (на выходе фильтра могут возникнуть неконтролируемый рост сигнала или свободные колебания). Примером программной реализации рекурсивного фильтра является **Программа 19.1**.

Главное достоинство рекурсивных фильтров заключается в том, что при их реализации удаётся избежать использования операции свёртки, требующей большого количества арифметических операций. Для примера подадим на вход рекурсивного фильтра *дискретную дельта-функцию* (*единичный импульс*). Реакцией на такое воздействие будет импульсная характеристика фильтра, которая представляет собой синусоидальные колебания с убывающей по экспоненциальному закону амплитудой. Поскольку такая импульсная характеристика бесконечна по времени, рекурсивные фильтры называют *фильтрами с бесконечной импульсной характеристикой (БИХ)*. В действительности происходит свёртка входного сигнала с бесконечной импульсной характеристикой при конечном числе весовых коэффициентов.

Взаимосвязь между *импульсной характеристикой* фильтра и его весовыми коэффициентами определяется при помощи *Z-преобразования*, о котором пойдёт речь в Главе 33. *Z-преобразование* используется для решения следующих задач: определение частотной характеристики по весовым коэффициентам рекурсивного фильтра, объединение последовательных и параллельных звеньев в единый фильтр, проектирование рекурсивных систем, повторяющих по своим свойствам аналоговые фильтры, и т. д. К сожалению, *Z-преобразование* требует специальных знаний из области высшей математики и оказывается для многих слишком сложным. Данное преобразование создано для специалистов в области ЦОС и цифровых систем.

Есть три способа, позволяющих найти коэффициенты рекурсивного фильтра для тех, кто не владеет *Z-преобразованием*. Первый способ: в данной главе содержатся уравнения для расчёта нескольких типов простейших рекурсивных фильтров. Второй способ: в Главе 20 приводится компьютерная программа для расчёта более сложных НЧ- и ВЧ-фильтров Чебышева. Третий способ: в Главе 26 рассмотрен итеративный метод построения рекурсивных фильтров с произвольной формой частотной характеристики.

### **Программа 19.1**

```

100 'РЕКУРСИВНЫЙ ФИЛЬТР
110 '
120 DIM X[499]           'X[ ] - массив отсчётов входного сигнала
130 DIM Y[499]           'Y[ ] - массив отсчётов выходного сигнала
140 '
150 GOSUB XXXX          'Некоторая подпрограмма вычисления весовых
160 '                    'коэффициентов: A0, A1, A2, B1, B2
170 '
180 GOSUB XXXX          'Некоторая подпрограмма загрузки входных данных в X[ ]
190 '
200 FOR I% = 2 TO 499
210 Y[I%] = A0*X[I%] + A1*X[I%-1] + A2*X[I%-2] + B1*Y[I%-1] + B2*Y[I%-2]
220 NEXT I%
230 '
240 END

```

## 19.2. Однополосный рекурсивный фильтр

Пример однополосного НЧ-фильтра показан на Рис. 19.2. Этот рекурсивный фильтр имеет всего два весовых коэффициента, принимающих значения  $a_0 = 0.15$  и  $b_1 = 0.85$ . Подадим ступенчатое входное воздействие. Как и следовало ожидать, выходная реакция НЧ-фильтра плавно нарастает до некоторого установившегося значения. Этот график, возможно, напомнил вам известный пример из электротехники. Рекурсивный НЧ-фильтр оказывает на цифровой сигнал точно такое же действие, какое оказывает обычный  $RC$ -фильтр на аналоговый сигнал.

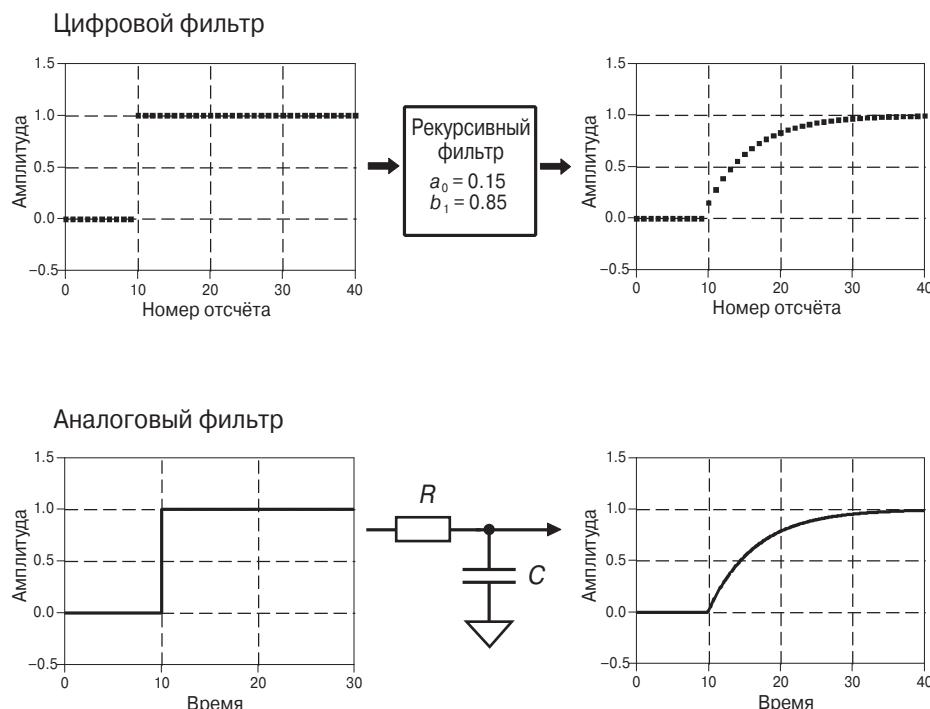
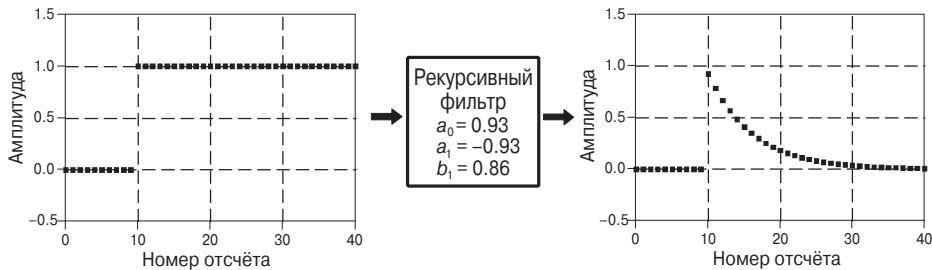


Рис. 19.2. Однополосный НЧ-фильтр. Цифровые рекурсивные фильтры в точности повторяют аналоговые фильтры, построенные на резисторах и конденсаторах. Данный пример показывает, что однополосный рекурсивный фильтр точно так же сглаживает ступенчатое входное воздействие, как и аналоговый  $RC$ -фильтр.

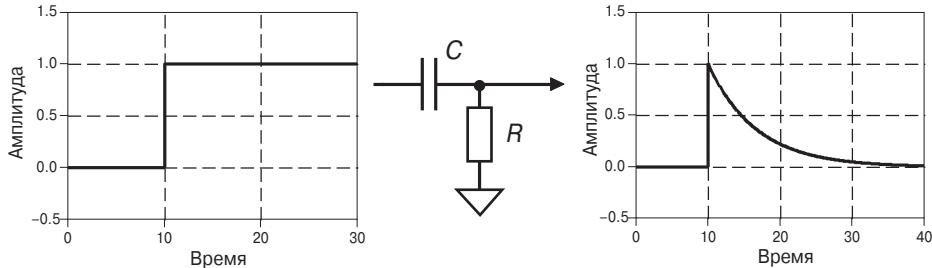
Достоинство рекурсивного метода состоит в том, что, меняя всего несколько параметров, можно создавать самые разные импульсные характеристики. Фильтр, характеристики которого приведены на Рис. 19.3, имеет три коэффициента:  $a_0 = 0.93$ ,  $a_1 = -0.93$  и  $b_1 = 0.86$ . Подавая разнообразные тестовые воздействия, можно показать, что он полностью повторяет свойства аналогового ВЧ-фильтра, построенного на основе  $RC$ -цепи.

Разумеется, такие фильтры вам очень хотелось бы иметь в наборе доступных инструментов ЦОС. Их можно использовать при обработке цифровых сигналов так же, как  $RC$ -цепи используются в аналоговой обработке. Решаемые задачи точно те же: устранение постоянной составляющей, подавление высокочастотного

## Цифровой фильтр



## Аналоговый фильтр



**Рис. 19.3.** Однополюсный ВЧ-фильтр. Выбрав подходящие весовые коэффициенты, можно получить цифровой рекурсивный фильтр, который полностью повторяет свойства аналогового ВЧ-фильтра, построенного на основе  $RC$ -цепи. Рекурсивные фильтры играют ту же роль в ЦОС, что и  $RC$ -фильтры в аналоговой технике.

шума, формирование огибающей, сглаживание и т. д. Для таких фильтров легко написать программу, они не требуют больших вычислительных затрат и в процессе использования практически не преподносят неприятных сюрпризов. Для расчёта весовых коэффициентов применяются следующие простые уравнения:

$$a_0 = 1 - x, \quad (19.2)$$

$$b_1 = x.$$

Однополюсный НЧ-фильтр. Форма импульсной характеристики определяется параметром  $x = 0 \dots 1$ .

$$a_0 = (1 + x)/2,$$

$$a_1 = -(1 + x)/2, \quad (19.3)$$

$$b_1 = x.$$

Однополюсный ВЧ-фильтр.

Характеристики таких фильтров определяются параметром  $x = 0 \dots 1$ . Физический смысл параметра  $x$  — относительное затухание на интервале между соседними отсчёты. Например, у фильтра на Рис. 19.3  $x = 0.86$ , вследствие чего каждый очередной отсчёт импульсной характеристики равен предыдущему, умноженному на 0.86. Чем больше  $x$ , тем медленнее спад. Если  $x$  больше единицы, фильтр оказывается неустойчивым. В этом случае появление любого ненулевого отсчёта на входе фильтра приведёт к переполнению (появлению слишком больших значений) на выходе.

Параметр  $x$  может быть задан непосредственным образом, а также может быть выражен через постоянную времени фильтра. Величина  $R \times C$  имеет размерность времени, и в простейших  $RC$ -цепях она соответствует интервалу времени, через который отклонение амплитуды выходного сигнала от установившегося значения сокращается в ходе переходного процесса до 36.8% от первоначальной величины. Если  $d$  — постоянная времени, измеряемая числом отсчётов дискретного сигнала, то имеет место следующее равенство:

$$x = e^{-1/d}. \quad (19.4)$$

Постоянная времени однополюсного фильтра. Взаимосвязь между относительным затуханием  $x$  (за период дискретизации) и постоянной времени фильтра  $d$ .

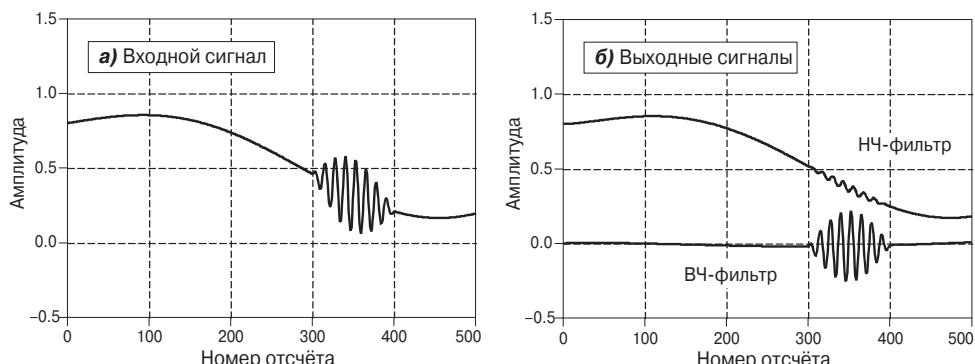
Относительному затуханию  $x = 0.86$  соответствует постоянная времени  $d = 6.63$  дискретных отсчётов (Рис. 19.3). Кроме того, между  $x$  и частотой среза, определяемой на уровне  $-3$  дБ, существует взаимосвязь:

$$x = e^{-2\pi f_C}. \quad (19.5)$$

Частота среза однополюсного фильтра. Взаимосвязь между относительным затуханием  $x$  и частотой среза фильтра  $f_C = 0 \dots 0.5$ .

Из сказанного следует, что коэффициенты  $a_1$  и  $b_1$  могут быть выражены через следующие три параметра: постоянную времени, частоту среза и непосредственно через относительное затухание  $x$ .

Пример работы рекурсивных однополюсных фильтров иллюстрируется на Рис. 19.4. Входной сигнал (**а**) представляет собой гладкую кривую, на которую на

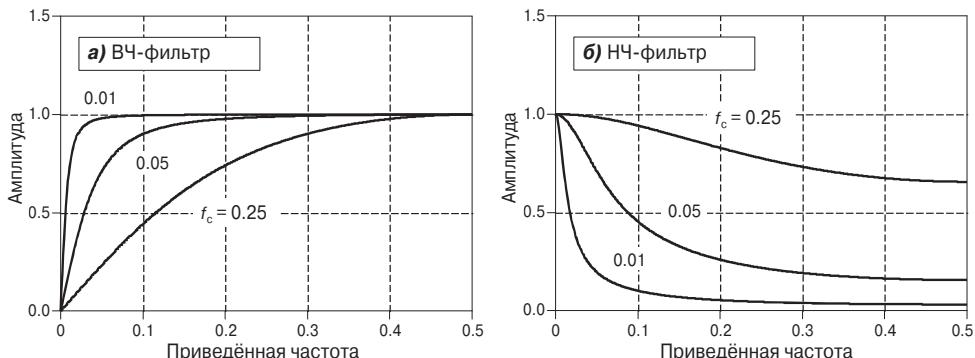


**Рис. 19.4.** Пример работы однополюсных рекурсивных фильтров. **а)** Входной сигнал представляет собой низкочастотное колебание, на который наложено кратковременное высокочастотное воздействие. **б)** Низкочастотная составляющая входного сигнала выделяется НЧ-фильтром с параметром  $x = 0.95$ , а высокочастотная — ВЧ-фильтром с параметром  $x = 0.86$ .

протяжении короткого интервала времени накладывается всплеск более высокой частоты. С помощью НЧ- и ВЧ-фильтров можно разделить компоненты входного сигнала (**б**), но всё же качество фильтрации оказывается низким, как и при использовании аналоговых  $RC$ -цепей.

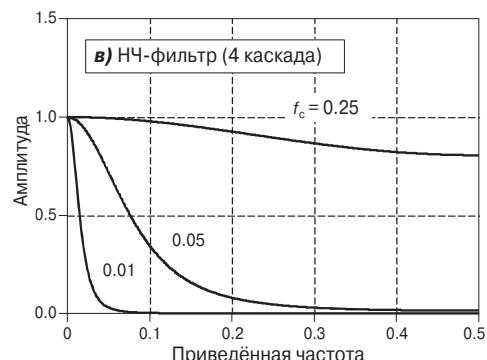
На Рис. 19.5 показаны АЧХ однополюсных рекурсивных фильтров. Чтобы получить частотную характеристику рекурсивного фильтра, сначала находят его импульсную характеристику как реакцию на единичное импульсное воздействие. Затем применяют БПФ, чтобы перейти от импульсной характеристики к частотной. Теоретически импульсная характеристика имеет бесконечную длину, но в реальности через интервал, в 15...20 раз превышающий постоянную времени, она затухает ниже уровня *шумов округления*. Например, для  $d = 6.63$  отсчёта в импульсной характеристике значащими остаются только около 128 отсчётов.

На Рис. 19.5 хорошо заметна важная особенность однополюсных рекурсивных фильтров, заключающаяся в слабых возможностях разделения частотных диапазонов. Эти фильтры отлично проявляют себя при обработке сигналов во временной области, но оказываются недостаточно мощными в задачах частотной селекции. Некоторого улучшения частотных характеристик позволяет достичь использование последовательного (каскадного) соединения нескольких однополюсных фильтров. Здесь возможны два способа. Во-первых, сигнал можно пропускать через один и тот же фильтр несколько раз. Во-вторых, воспользовавшись  $Z$ -преобразованием, можно найти весовые коэффициенты цифрового рекурсив-



**Рис. 19.5.** АЧХ однополюсных фильтров.

**а)** АЧХ однополюсных НЧ-фильтров. **б)** АЧХ однополюсных ВЧ-фильтров. **в)** АЧХ каскадного соединения четырёх однополюсных НЧ-фильтров. Не всегда при расчёте рекурсивных фильтров частотные характеристики оказываются такими, какими мы желали бы их видеть. Особенно часто это проявляется при попытке достичь некоторых предельных параметров. Например, на Рис. 14.5в кривая, полученная при выборе частоты среза  $f_c = 0.25$ , совершенно не соответствует заданным требованиям. В этом эффекте проявляется действие многих факторов: наложение спектров, шумы округления и нелинейность фазовой характеристики.



ного фильтра, эквивалентного по своим свойствам последовательному соединению нескольких звеньев. Оба метода хорошо работают и находят широкое применение. На Рис. 19.5в показаны АЧХ 4-каскадных рекурсивных НЧ-фильтров. Хотя затухание в зоне подавления при переходе к каскадной форме построения значительно увеличивается, переходная зона по-прежнему оставляет желать лучшего. Если вам необходимо повысить качество в частотной области, ориентируйтесь на использование фильтров Чебышева, речь о которых пойдет в следующей главе.

Многокаскадный НЧ-фильтр сопоставим с фильтрами Блэкмана и Гаусса, являющимися модификациями однородных КИХ-фильтров (Глава 15), но при этом он требует существенно меньших вычислительных затрат. Для расчёта 4-каскадного НЧ-фильтра можно воспользоваться следующими уравнениями:

$$\begin{aligned} a_0 &= (1-x)^4, \\ b_1 &= 4x, \\ b_2 &= -6x^2, \\ b_3 &= 4x^3, \\ b_4 &= -x^4. \end{aligned} \tag{19.6}$$

4-каскадный НЧ-фильтр. Данные уравнения позволяют рассчитать коэффициенты  $a_i$  и  $b_i$  для 4-каскадного соединения однополюсных НЧ-фильтров. Взаимосвязь  $x$  и  $f_C$  определяется выражением вида (19.5), в котором  $2\pi$  заменено на 14.445.

## 19.3. Узкополосный рекурсивный фильтр

При построении цифровых устройств очень часто возникает задача выделения очень узкой полосы частот в исходном широкополосном сигнале. Например, может потребоваться устраниТЬ наводку от сети питания в измерительной аппаратуре или отделить тональные сигналы в телефонной сети. Для этой цели используются фильтры двух видов: *полосовые* и *режекторные*. Характеристики узкополосных рекурсивных фильтров показаны на Рис. 19.6. Весовые коэффициенты фильтров найдены при помощи следующих систем уравнений:

$$\begin{aligned} a_0 &= 1 - K, \\ a_1 &= 2(K - R) \cos(2\pi f), \\ a_2 &= R^2 - K, \\ b_1 &= 2R \cos(2\pi f), \\ b_2 &= -R^2, \end{aligned} \tag{19.7}$$

Полосовой узкополосный фильтр. Примеры АЧХ показаны на Рис. 19.6а. Исходными параметрами являются центральная частота  $f$  и ширина полосы пропускания  $BW$  (BandWidth), которые должны быть заданы относительно частоты дискретизации, т. е. в диапазоне 0...0.5. Затем вычисляются параметры  $R$  и  $K$ , и только после этого рассчитываются весовые коэффициенты:

$$a_0 = K,$$

$$a_1 = 2 \cdot K \cdot \cos(2\pi f),$$

$$a_2 = K,$$

$$b_1 = 2R\cos(2\pi f), \quad (19.8)$$

$$b_2 = -R^2,$$

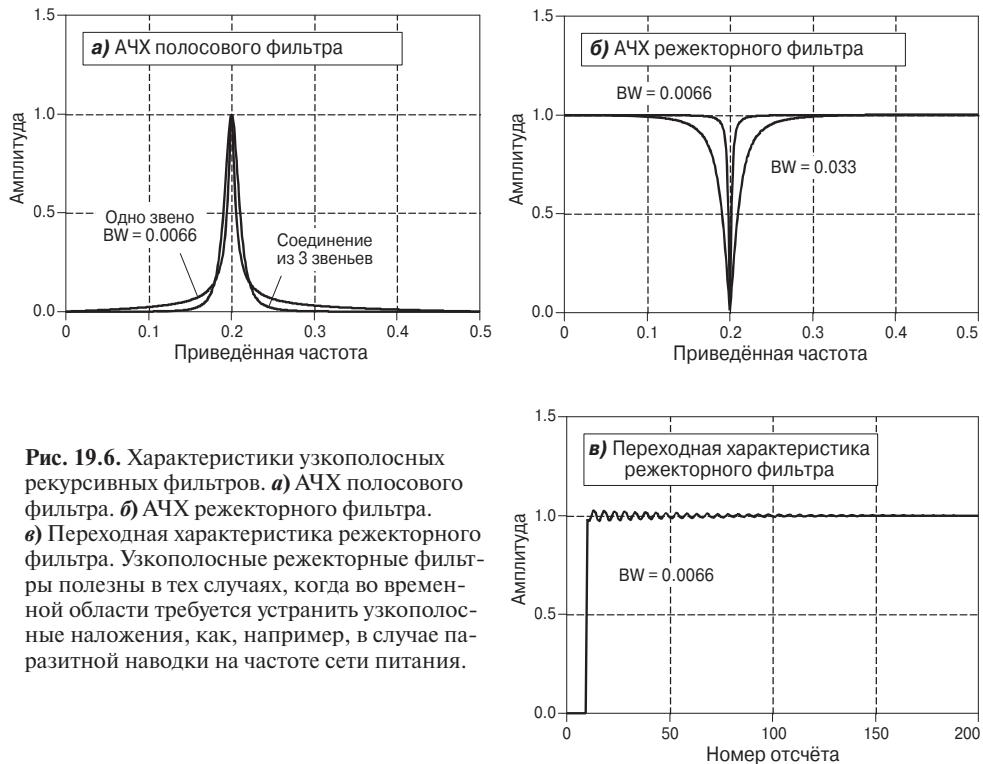
где

$$K = \frac{1 - 2R\cos(2\pi f) + R^2}{2 - 2\cos(2\pi f)},$$

$$R = 1 - 3BW.$$

Режекторный узкополосный фильтр.

Возможные частотные характеристики фильтра показаны на Рис. 19.6б.



**Рис. 19.6.** Характеристики узкополосных рекурсивных фильтров. **а)** АЧХ полосового фильтра. **б)** АЧХ режекторного фильтра. **в)** Переходная характеристика режекторного фильтра. Узкополосные режекторные фильтры полезны в тех случаях, когда во временной области требуется устраниить узкополосные наложения, как, например, в случае паразитной наводки на частоте сети питания.

Чтобы воспользоваться этими выражениями, необходимо предварительно задать два параметра: центральную частоту  $f$  и ширину полосы пропускания  $BW$  (на уровне 0.707), измеряемые относительно частоты дискретизации и, следовательно, принимающие значения в диапазоне 0...0.5. По заданным параметрам сначала

вычисляют промежуточные переменные  $R$  и  $K$ , а затем переходят к вычислению самих весовых коэффициентов.

Относительно медленный спад АЧХ полосового фильтра (**Рис. 19.6а**) можно сделать более крутым, если включить последовательно несколько фильтров. Так как уравнения получаются довольно длинными, проще несколько раз пропустить сигнал через один и тот же фильтр, чем рассчитывать весовые коэффициенты результирующего фильтра, полученного последовательным соединением нескольких каскадов.

На (**б**) показаны примеры АЧХ узкополосных режекторных фильтров. Наименьшая достижимая ширина полосы пропускания равна примерно 0.0003. После достижения этого предела уровень затухания на центральной частоте начинает падать. Переходная характеристика режекторного фильтра изображена на (**в**). Характеристика имеет ярко выраженный колебательный процесс, но амплитуда колебаний всё же невелика. В результате фильтр позволяет устраниć паразитную помеху на частоте сети питания, внося очень незначительные искажения во временной области.

## 19.4. ФЧХ рекурсивных фильтров

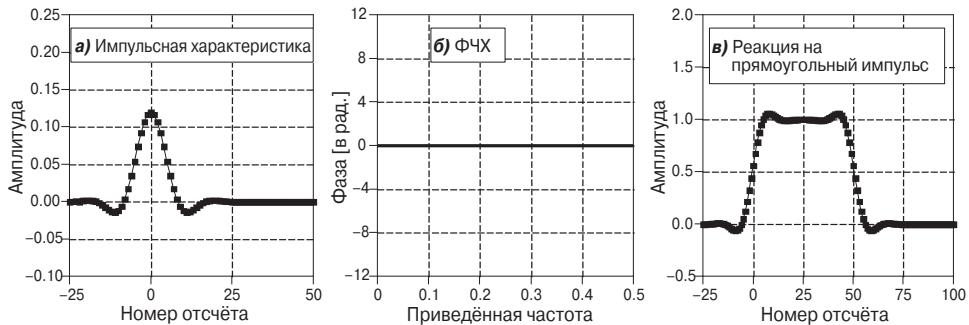
С позиций воспроизводимой ФЧХ существует три типа фильтров: *с нулевой фазой*, *с линейной ФЧХ* и *с нелинейной ФЧХ* (**Рис. 19.7**). Фильтр с нулевой фазой имеет симметричную относительно нуля импульсную характеристику (**а**). Конкретная форма импульсной характеристики не играет роли, главное, чтобы отсчёты слева от оси ординат были зеркальным отражением отсчётов, расположенных справа. При такой характеристике преобразование Фурье приводит к получению отсчётов со строго нулевой фазой (**б**).

Недостатком фильтров с нулевой фазой является наличие отрицательных индексов, которое означает физическую нереализуемость цепи. Устранить указанный недостаток позволяет переход к фильтрам с линейной ФЧХ. Импульсная характеристика *линейно-фазового фильтра* (**г**) имеет ту же форму, что и характеристика фильтра с нулевой фазой (**а**), за исключением такого сдвига по оси абсцисс, который делает все индексы положительными. Сохраняется и симметрия импульсной характеристики, но теперь уже ось симметрии смешена относительно оси ординат. Этот сдвиг во временной области приводит к изменению наклона ФЧХ (**д**), которая имеет форму прямой линии, что отражено в названии фильтра: фильтр с линейной ФЧХ. Тангенс угла наклона ФЧХ прямо пропорционален сдвигу импульсной характеристики. Так как единственным следствием сдвига импульсной характеристики является задержка выходной реакции фильтра, то фильтр с линейной фазой во многих приложениях можно считать эквивалентным фильтру с нулевой фазой.

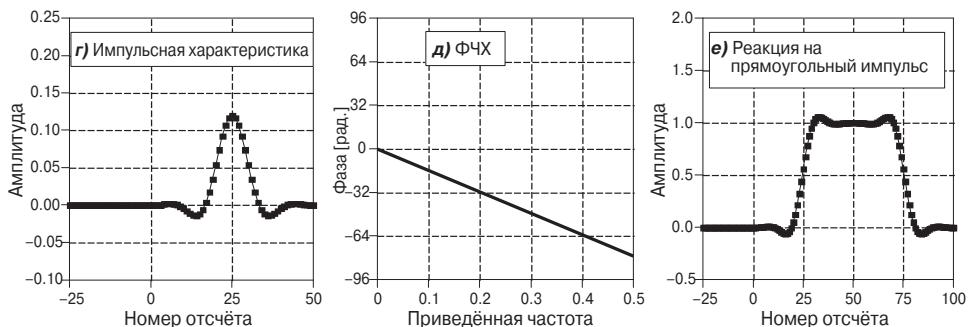
Импульсная характеристика на (**ж**) не симметрична, поэтому ФЧХ такого фильтра (**з**) отличается от линейной. Не путайте линейность и нелинейность ФЧХ с понятием «линейная система», введённым в Главе 5. Хотя в обоих случаях говорится о линейности, это совершенно разные понятия.

Какая разница, линейная фаза или нет? Ответ очевиден из (**в, е и и**), на которых показаны реакции фильтров на прямоугольный импульс. Реакция на прямо-

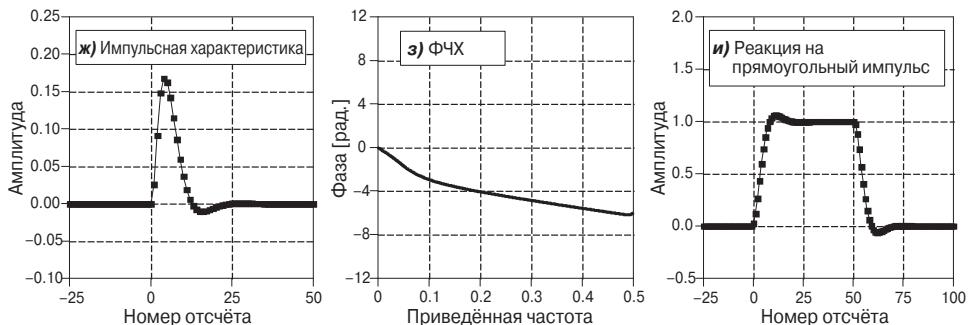
### Фильтр с нулевой фазой



### Фильтр с линейной ФЧХ



### Фильтр с нелинейной ФЧХ



**Рис. 19.7.** Три вида ФЧХ фильтров. **а)** Фильтр с нулевой фазой имеет импульсную характеристику, симметричную относительно оси ординат. **б)** ФЧХ фильтра равна нулю во всем частотном диапазоне. **в)** При фильтрации прямоугольного импульса передний и задний его фронты получают одинаковые (симметричные) искажения, что связано с симметрией верхней и нижней частей переходной характеристики фильтра с нулевой фазой. **г)** Импульсная характеристика фильтров с линейной ФЧХ тоже симметрична, но ось симметрии в этом случае смещена относительно оси ординат. **д)** Результатом смещения оси симметрии является наклон ФЧХ. **е)** Фильтр с линейной ФЧХ сохраняет все достоинства фильтра с нулевой фазой. **ж)** Фильтры с нелинейной ФЧХ имеют несимметричную импульсную характеристику. **з)** ФЧХ таких фильтров отличается от линейной. **и)** Самым худшим свойством фильтров с нелинейной ФЧХ является отсутствие симметрии при искажении переднего и заднего фронтов прямоугольного импульса.

угольный импульс представляет собой комбинацию двух переходных характеристик, из которых первая формирует передний фронт импульса, а вторая — перевёрнутая переходная характеристика — формирует его задний фронт. Как это видно по графикам, при обработке фильтрами с нулевой фазой и с линейной ФЧХ оба фронта получают одинаковые искажения, в то время как при нелинейной ФЧХ фронты импульса различаются. Во многих практических приложениях подобная асимметрия недопустима. Например, при снятии осциллограммы подобное различие может быть ошибочно приписано свойствам измеряемого сигнала. Приведем еще более интересный пример из области обработки изображений. Представьте себе, что, включив телевизор, вы обнаруживаете, что у вашего любимого актёра или актрисы левое ухо не похоже на правое.

Получить КИХ-фильтр с линейной ФЧХ не представляет труда. Это объясняется тем, что отсчёты импульсной характеристики *КИХ-фильтра* (его весовые коэффициенты) определяются непосредственно в процессе его расчёта, и для линейности ФЧХ достаточно ввести требование симметрии импульсной характеристики. Весовые коэффициенты БИХ-фильтров не совпадают с отсчётами импульсной характеристики, которая оказывается несимметричной, что выражается в свою очередь в нелинейности ФЧХ.

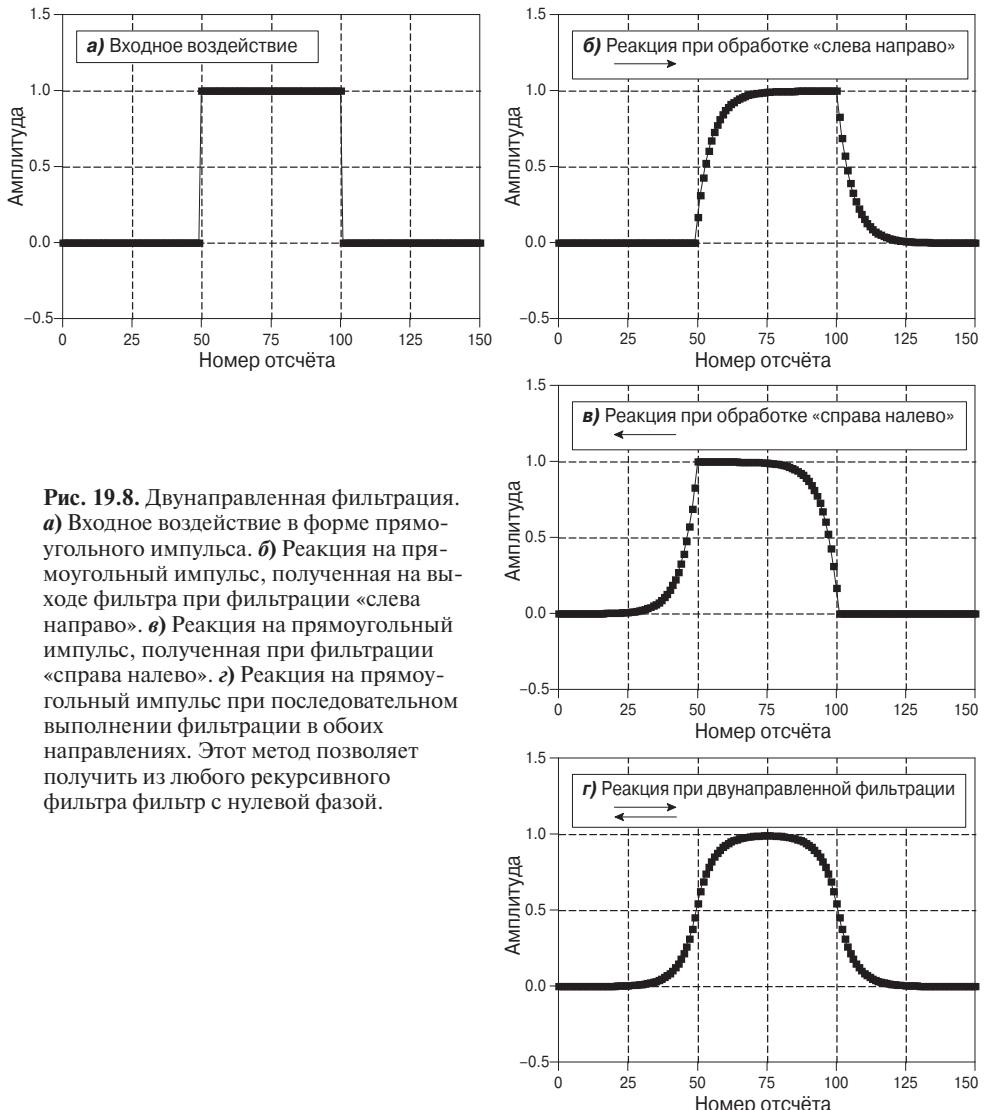
В аналоговой схемотехнике тоже приходится решать проблему нелинейности ФЧХ. Представьте на своём рабочем столе электрическую схему с резисторами и конденсаторами. Пока сигнал на входе равен нулю, выходной сигнал тоже сохраняет нулевое значение. Если на вход подать короткий импульс, конденсаторы мгновенно заряжаются и далее начнут разряжаться по экспоненциальному закону через резисторы. Импульсная характеристика (наблюдаемый выходной сигнал) представляет собой комбинацию этих спадающих с разными скоростями экспонент. Импульсная характеристика никак не может оказаться симметричной, поскольку до поступления импульса на вход схемы на её выходе сигнал равнялся нулю, а экспоненты убывают до бесконечности, так и не достигая нуля. Разработчики аналоговых фильтров решают эту проблему при помощи *фильтра Бесселя* (Глава 3). При построении фильтра Бесселя ставится задача сделать ФЧХ как можно более линейной. Но такие фильтры всё равно сильно уступают по качеству *цифровым фильтрам*, для которых обеспечение линейной ФЧХ — элементарная задача.

К счастью, есть очень простой способ построения рекурсивного фильтра с нулевой фазой (**Рис. 19.8**). Пусть, для примера, на вход однополюсного НЧ-фильтра поступает прямоугольный импульс (**а**). Так как ФЧХ фильтра является нелинейной, передний и задний фронты импульса оказываются несимметричными после выполнения фильтрации: точнее говоря, задний фронт по форме совпадает с перевёрнутым передним фронтом импульса (**б**). Данный фильтр вычисляет отсчёты выходного сигнала, начиная с нулевого и заканчивая 150-м, аналогично тому, как это происходило и в других рассмотренных ранее примерах.

Теперь предположим, что вместо продвижения от нулевого отсчёта к 150-му, фильтр производит обработку сигнала в обратном направлении: от 150-го отсчёта к нулевому. Другими словами, каждый отсчёт выходного сигнала выражается через отсчёты входного и выходного сигналов, расположенные на временной оси справа от него. Тогда уравнение рекурсивного фильтра (**19.1**) приобретает новую форму:

$$\begin{aligned} y[n] = & a_0x[n] + a_1x[n+1] + a_2x[n+2] + a_3x[n+3] + \dots, \\ & + b_1y[n+1] + b_2y[n+2] + b_3y[n+3] + \dots. \end{aligned} \quad (19.9)$$

Уравнение рекурсивного фильтра с обратным направлением фильтрации. Данное уравнение аналогично выражению (9.1), за исключением того, что фильтрация выполняется «справа налево», а не «слева направо».



**Рис. 19.8.** Двунаправленная фильтрация.  
**а)** Входное воздействие в форме прямоугольного импульса. **б)** Реакция на прямоугольный импульс, полученная на выходе фильтра при фильтрации «слева направо». **в)** Реакция на прямоугольный импульс, полученная при фильтрации «справа налево». **г)** Реакция на прямоугольный импульс при последовательном выполнении фильтрации в обоих направлениях. Этот метод позволяет получить из любого рекурсивного фильтра фильтр с нулевой фазой.

На **(в)** показан импульс, который должен получиться при *обратном направлении фильтрации*. Наблюдаемый эффект полностью аналогичен тому случаю, который наблюдался бы при прохождении прямоугольного импульса через  $RC$ -цепь, если бы время пошло назад:

*!йоннелес В дерв итсенан тежжом инемерв яинелварпан еиненемзи :онжоротсО*

Само по себе изменение направления фильтрации не приносит никакой пользы: передний и задний фронты импульса по-прежнему отличаются друг от друга. Но когда фильтрация производится в обоих направлениях, происходит чудо. Импульс, изображённый на (2), получен при выполнении фильтрации сначала в прямом, а затем в обратном направлении. Ну, как вам такой фокус?! Получается рекурсивный фильтр с нулевой фазой. Рассмотренный метод *дву направленной фильтрации* позволяет получить фильтр с нулевой фазой практически из любого рекурсивного фильтра. Единственной «платой» за улучшение ФЧХ является усложнение кода программы и увеличение вдвое времени её выполнения.

Как определить импульсную и частотную характеристики полученного фильтра с двунаправленной фильтрацией? При изменении направления фильтрации АЧХ фильтра остается прежней, а все отсчёты ФЧХ меняют свой знак на противоположный. В результате объединения двух таких фильтров в один все отсчёты АЧХ возводятся в квадрат, а отсчёты ФЧХ обращаются в ноль. Во временной области объединение фильтров выражается в свёртке исходной импульсной характеристики с её отражением в направлении «слева направо». Для примера возьмем однополосный НЧ-фильтр с экспоненциально спадающей ФЧХ. Чтобы определить импульсную характеристику соответствующего двунаправленного фильтра, нужно выполнить свёртку экспоненты, которая спадает «слева направо», с экспонентой, имеющей такую же скорость спада, но в направлении «справа налево». После выполнения нескольких математических преобразований выясняется, что импульсная характеристика результирующего фильтра имеет вид импульса, образованного двумя экспонентами: сначала возрастающей, а затем убывающей. При этом по скорости спада обе экспоненты повторяют исходную импульсную характеристику.

Во многих практических приложениях данные обрабатываются отдельными блоками. В таких случаях двунаправленная фильтрация может быть использована совместно с методом *секционирования*, о котором говорилось в предыдущей главе. Если вас спросят, какова в этом случае длина импульсной характеристики, не говорите, что она бесконечна, потому что это означало бы, что каждый сегмент дополняется бесконечным числом нулевых отсчётов. Помните, что импульсную характеристику можно ограничить интервалом, за пределами которого она становится ниже уровня шумов округления, т. е. периодом, в 15...20 раз превышающим постоянную времени. Для согласования длины сегмента и длины импульсной характеристики двунаправленного фильтра справа и слева добавляется необходимое количество нулевых отсчётов.

## 19.5. Применение целочисленной арифметики

Для простых *рекурсивных фильтров* хорошо подходят числа с одинарной точностью в *формате с плавающей точкой*. Использование целых чисел хотя и возможно, но создаёт ряд трудностей. Наиболее сложными являются следующие две проблемы. Во-первых, *ошибка округления* может серьёзно исказить характеристики фильтра или даже сделать его неустойчивым. Во-вторых, дробные значения весовых коэффициентов должны быть представлены при помощи целых чисел.

Один вариант решения этой проблемы — замена весовых коэффициентов отношениями целых чисел. Например, 0.15 можно заменить примерно на 19/128. То есть вместо умножения на 0.15 сначала придётся выполнить умножение на 19, а затем деление на 128. Другой вариант — использование *справочной таблицы (таблицы соответствия)*. Например, выходными значениями 12-битного АЦП могут быть целые числа в диапазоне 0...4095. Следовательно, умножение на 0.15 можно заменить процедурой поиска соответствия в таблице из 4096 элементов. Значение, выбираемое из таблицы, составляет 0.15 от исходного значения. Последний рассмотренный метод работает намного быстрее, однако требует большого объёма памяти. Но прежде чем перейти к выбору одного из рассмотренных здесь методов, следует убедиться, что вам точно не подходит однородный нерекурсивный фильтр. Ведь он просто «обожает» целые числа!

# Глава 20

## ФИЛЬТРЫ ЧЕБЫШЕВА

*Фильтры Чебышева* предназначены для частотной селекции сигналов. Несмотря на то что они уступают по некоторым показателям фильтрам, полученным оконными методами, для многих практических приложений такие фильтры оказываются предпочтительнее. Самым главным достоинством чебышевских фильтров является значительное уменьшение количества вычислений. По этому параметру они более чем на порядок превосходят *оконные фильтры*. Это связано с *рекурсивной структурой* фильтров Чебышева, обеспечивающей более высокое быстродействие по сравнению со *свёрткой*. В основе расчёта фильтров Чебышева лежит математический аппарат, получивший название *Z-преобразования*, о котором пойдет речь в Главе 33. В этой главе мы постараемся рассказать о практике использования фильтров Чебышева, обходя по мере возможности термин высшей математики.

### 20.1. Частотные характеристики фильтров Чебышева и Баттерворта

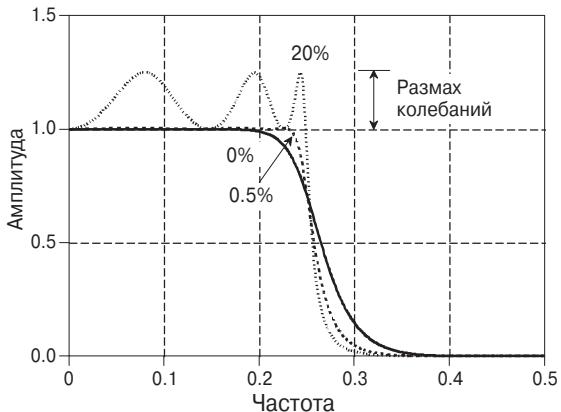
При расчёте фильтров Чебышева за счёт допущения *неравномерности амплитудно-частотной характеристики (АЧХ)* удается увеличить крутизну спада на границе полосы пропускания. Такой подход характерен как для аналоговых, так и для цифровых фильтров Чебышева. Аналоговые чебышевские фильтры применяются, например, в *аналого-цифровых и цифро-аналоговых преобразователях*, речь о которых шла в Главе 3. Название этих фильтров происходит от *полиномов Чебышева*, положенных в основу их математического описания. Данные полиномы были получены русским математиком Пафнутием Чебышевым (1821 – 1894)<sup>1)</sup>.

На Рис. 20.1 показаны характеристики низкочастотных фильтров Чебышева с неравномерностью в полосе пропускания: 0, 0.5 и 20%. Увеличение неравномерности АЧХ в полосе пропускания позволяет повысить крутизну её спада в переходной зоне. Расчёт фильтра Чебышева связан с нахождением некоторого компромисса между этими двумя параметрами. Фильтр с нулевым уровнем неравномерности, т. е. с *максимально гладкой АЧХ*, называют *фильтром Баттервортса* (по имени английского инженера С. Баттерворта, впервые описавшего его характеристики в 1930 году). Как правило, используют цифровые фильтры с уровнем неравномерности АЧХ 0.5%, которые по точности и качеству работы наилучшим образом соответствуют компонентам аналоговой электронной техники.

<sup>1)</sup> В англоязычной литературе можно встретить несколько вариантов написания фамилии Пафнутия Чебышева: Chebychev, Tschebyscheff, Tchebysheff и Tchebichef. — Примеч. пер.

**Рис. 20.1.** АЧХ фильтров Чебышева.

Применение фильтров Чебышева позволяет добиться увеличения крутизны спада АЧХ на границе полосы пропускания. Фильтр с нулевым уровнем неравномерности, т. е. с *максимально гладкой АЧХ*, называют *фильтром Баттервортса*. Рекомендуем задавать уровень колебаний АЧХ 0.5% для практических приложений, так как при этом колебания настолько малы, что практически незаметны на графике, а по крутизне спада фильтр намного превосходит фильтр Баттервортса.



В этой главе рассматриваются чебышевские *фильтры первого типа*, для которых допускается наличие колебаний АЧХ только в *полосе пропускания*. Заметим, что *фильтры второго типа* имеют колебания только в зоне *подавления*. Мы не будем подробнее останавливаться на фильтрах второго типа, так как они редко используются на практике. Существует ещё один тип фильтров — *эллиптические*. Для них характерно наличие колебаний и в *полосе пропускания*, и в зоне *подавления*. Описание этого типа фильтров также остаётся за рамками книги, но следует помнить, что очень часто профессиональные разработчики выбирают именно эти фильтры (как аналоговые, так и цифровые). Если вам требуются эллиптические фильтры, вы можете найти необходимое для их расчёта программное обеспечение.

## 20.2. Расчёт фильтра

Расчёт фильтра Чебышева предполагает выбор четырёх параметров: 1) тип фильтра (высокочастотный или низкочастотный); 2) частота среза; 3) уровень неравномерности АЧХ в полосе пропускания; 4) количество полюсов. Что такое *полюсы*? Приведём далее два варианта ответа на этот вопрос. Если будет непонятен первый — обратитесь ко второму.

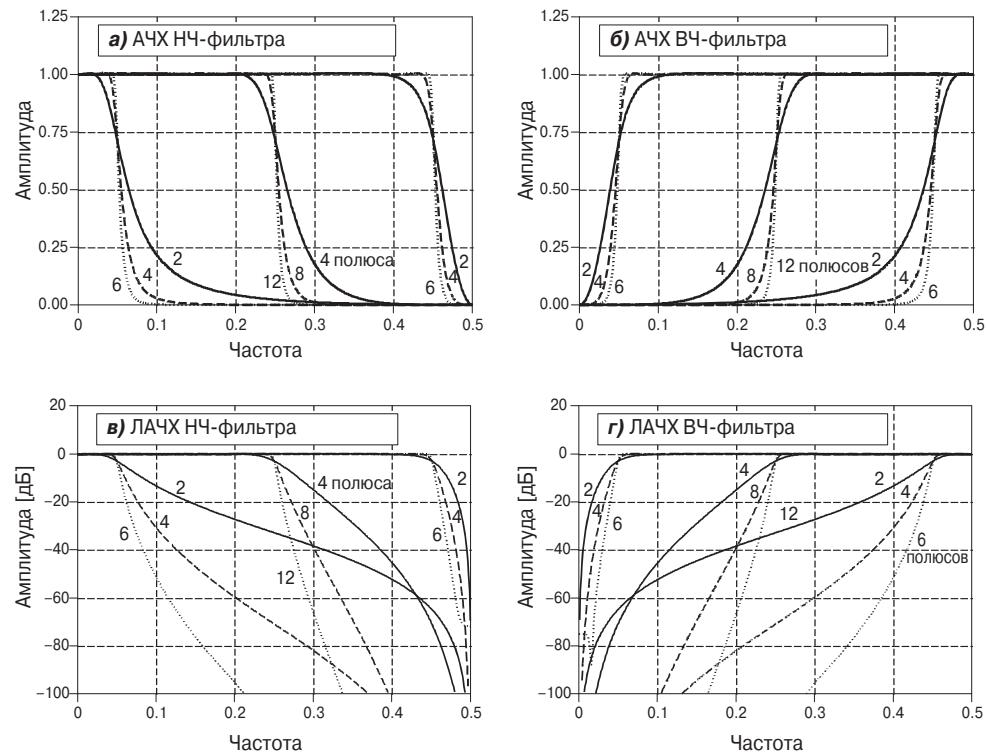
Ответ 1. *Преобразование Лапласа* и *Z-преобразование* позволяют описать *импульсную характеристику* фильтра с помощью набора синусоид и убывающих экспонент. Для этого требуется предварительно представить частотную характеристику в форме отношения двух многочленов. Корни числителя передаточной функции называют *нулями*, а корни знаменателя — *полюсами*. Полюсы и нули могут выражаться комплексными числами, поэтому говорят, что они «располагаются» на комплексной плоскости. Чем больше нулей и полюсов, тем выше качество системы. Расчёт рекурсивных фильтров начинают с размещения полюсов и нулей, а уже затем находят подходящие весовые коэффициенты. Например, полюсы фильтра Баттервортса лежат в комплексной плоскости на *окружности*, тогда как у фильтра Чебышева они расположены по *эллипсу*. Подробнее об этом говорится в Главах 32 и 33.

Ответ 2. Каждый полюс — это такой «сундучок» с магической силой. Чем больше полюсов у фильтра, тем лучше он работает.

Но если серьёзно, то вам просто следует усвоить, что можно научиться эффективно использовать фильтры, не углубляясь в «занудную» математику. Расчёт фильтров — это отдельная задача, требующая специальных знаний. Большинство же инженеров и программистов представляют себе фильтры в повседневной своей работе скорее на уровне ответа 2, чем ответа 1.

На Рис. 20.2 показаны частотные характеристики фильтров Чебышева с уровнем неравномерности АЧХ — 0.5%. Использованный метод расчёта требует чётного числа полюсов. Граница полосы пропускания определяется точкой пересечения амплитудно-частотной характеристикой уровня 0.707 (-3 дБ). Крутизна спада АЧХ фильтров, частота среза которых близка к 0 или 0.5, оказывается выше, чем у фильтров с частотой среза ближе к середине частотного диапазона. Например, двухполюсный фильтр, у которого  $f_c = 0.05$ , оказывается близким по крутизне спада к четырёхполюсному фильтру с частотой среза  $f_c = 0.25$ . Отсюда появляется замечательная возможность уменьшения числа полюсов у фильтров с очень узкой или, наоборот, с очень широкой (близкой к 0.5) полосой пропускания. Но об этом чуть позже.

Существуют два способа нахождения коэффициентов обратной связи без использования Z-преобразования. Первый способ для «чайников» — можно воспользоваться *справочной таблицей*. В Табл. 20.1 и 20.2 приводятся коэффициенты



**Рис. 20.2.** АЧХ фильтров Чебышева в линейном и логарифмическом масштабе: **а и б** — НЧ-фильтр, **в и г** — ВЧ-фильтр. Уровень неравномерности АЧХ в полосе пропускания равен 0.5%. Цифры, указанные рядом с кривыми на графиках, — количество полюсов для каждого фильтра.

Таблица 20.1. Низкочастотные фильтры Чебышева с уровнем неравномерности АЧХ 0,5%

$f_c$	2 полоса	4 полоса	6 полосов
0.01	$a_0 = 8.663387E-04$ $a_1 = 1.732678E-03$ $a_2 = 8.663387E-04$ $b_1 = 1.919129E+00$ $b_2 = -9.225943E-01$	$a_0 = 4.149425E-07$ $(!! \text{ нестабильно !!})$ $a_1 = 1.659770E-06$ $a_2 = 2.489655E-06$ $a_3 = 1.659770E-06$ $a_4 = 4.149425E-07$ $b_1 = 3.8945453E+00$ $b_2 = -5.688233E+00$ $b_3 = 3.695783E+00$ $b_4 = -9.010106E-01$	$a_0 = 1.391351E-10$ $(!! \text{ нестабильно !!})$ $a_1 = 8.348109E-10$ $a_2 = 2.087027E-09$ $a_3 = 2.782703E-09$ $a_4 = 2.087027E-09$ $a_5 = 8.348109E-10$ $a_6 = 1.391351E-10$ $b_1 = 5.883343E+00$ $b_2 = -1.442798E+01$ $b_3 = 1.887786E+01$ $b_4 = -1.389914E+01$ $b_5 = 5.459909E+00$ $b_6 = -8.939932E-01$
0.025	$a_0 = 5.112374E-03$ $a_1 = 1.022475E-02$ $a_2 = 5.112374E-03$ $b_1 = 1.797154E+00$ $b_2 = -8.176033E-01$	$a_0 = 1.504626E-05$ $a_1 = 6.018503E-05$ $a_2 = 9.027754E-05$ $a_3 = 6.018503E-05$ $a_4 = 1.504626E-05$ $b_1 = 3.722385E+00$ $b_2 = -5.226004E+00$ $b_3 = 3.270902E+00$ $b_4 = -7.705239E-01$	$a_0 = 3.136210E-08$ $(!! \text{ нестабильно !!})$ $a_1 = 1.881726E-07$ $a_2 = 4.704314E-07$ $a_3 = 6.272419E-07$ $a_4 = 4.704314E-07$ $a_5 = 1.881726E-07$ $a_6 = 3.136210E-08$ $b_1 = 5.691653E+00$ $b_2 = -1.353172E+01$ $b_3 = 1.719986E+01$ $b_4 = -1.232689E+01$ $b_5 = 4.722721E+00$ $b_6 = -7.556340E-01$
0.05	$a_0 = 1.868823E-02$ $a_1 = 3.737647E-02$ $a_2 = 1.868823E-02$ $b_1 = 1.593937E+00$ $b_2 = -6.686903E-01$	$a_0 = 2.141509E-04$ $a_1 = 8.566037E-04$ $a_2 = 1.284906E-03$ $a_3 = 8.566037E-04$ $a_4 = 2.141509E-04$ $b_1 = 3.4245455E+00$ $b_2 = -4.479272E+00$ $b_3 = 2.643718E+00$ $b_4 = -5.933269E-01$	$a_0 = 1.771089E-06$ $a_1 = 1.0626534E-05$ $a_2 = 2.656634E-05$ $a_3 = 3.542179E-05$ $a_4 = 2.656634E-05$ $a_5 = 1.0626534E-05$ $a_6 = 1.771089E-06$ $b_1 = 5.330512E+00$ $b_2 = -1.196611E+01$ $b_3 = 1.447067E+01$ $b_4 = -9.937710E+00$ $b_5 = 3.673283E+00$ $b_6 = -5.707561E-01$
0.075	$a_0 = 3.869430E-02$ $a_1 = 7.738860E-02$ $a_2 = 3.869430E-02$ $b_1 = 1.392667E+00$ $b_2 = -5.474446E-01$	$a_0 = 9.726342E-04$ $a_1 = 3.890537E-03$ $a_2 = 5.835806E-03$ $a_3 = 3.890537E-03$ $a_4 = 9.726342E-04$ $b_1 = 3.103944E+00$ $b_2 = -3.774453E+00$ $b_3 = 2.111238E+00$ $b_4 = -4.562908E-01$	$a_0 = 1.797538E-05$ $a_1 = 1.078523E-04$ $a_2 = 2.696307E-04$ $a_3 = 3.595076E-04$ $a_4 = 2.696307E-04$ $a_5 = 1.078523E-04$ $a_6 = 1.797538E-05$ $b_1 = 4.921746E+00$ $b_2 = -1.935734E+01$ $b_3 = 1.889764E+01$ $b_4 = -7.854533E+00$ $b_5 = 2.822109E+00$ $b_6 = -4.307710E-01$

(продолжение)

$f_c$	2 полоса		4 полосы		6 полосов	
	a	b	a	b	a	b
0.1	a0 = 6.372802E-02 a1 = 1.274560E-01 a2 = 6.372802E-02	b1 = 1.194365E+00 b2 = -4.492774E-01	a0 = 2.780755E-03 a1 = 1.112302E-02 a2 = 1.668453E-02 a3 = 1.112302E-02 a4 = 2.780755E-03	b1 = 2.764031E+00 b2 = -3.122854E+00 b3 = 1.664554E+00 b4 = -3.502332E-01	a0 = 9.086148E-05 a1 = 5.451688E-04 a2 = 1.362922E-03 a3 = 1.817229E-03 a4 = 1.362922E-03 a5 = 5.451688E-04 a6 = 9.086148E-05	b1 = 4.470118E+00 b2 = -8.755594E+00 b3 = 9.543712E+00 b4 = -6.079376E+00 b5 = 2.140062E+00 b6 = -3.247363E-01
0.15	a0 = 1.254285E-01 a1 = 2.508570E-01 a2 = 1.254285E-01	b1 = 8.0070778E-01 b2 = -3.087918E-01	a0 = 1.180009E-02 a1 = 4.720034E-02 a2 = 7.080051E-02 a3 = 4.720034E-02 a4 = 1.180009E-02	b1 = 2.039039E+00 b2 = -2.012961E+00 b3 = 9.897915E-01 b4 = -2.046700E-01	a0 = 8.618665E-04 a1 = 5.171199E-03 a2 = 1.292800E-02 a3 = 1.723733E-02 a4 = 1.292800E-02 a5 = 5.171199E-03 a6 = 8.618665E-04	b1 = 3.455239E+00 b2 = -5.754735E+00 b3 = 5.645387E+00 b4 = -3.394902E+00 b5 = 1.177469E+00 b6 = -1.836195E-01
0.2	a0 = 1.997396E-01 a1 = 3.994792E-01 a2 = 1.997396E-01	b1 = 4.291048E-01 b2 = -2.280633E-01	a0 = 3.224554E-02 a1 = 1.289821E-01 a2 = 1.934732E-01 a3 = 1.289821E-01 a4 = 3.224554E-02	b1 = 1.265912E+00 b2 = -1.203878E+00 b3 = 5.405908E-01 b4 = -1.185538E-01	a0 = 4.187408E-03 a1 = 2.512445E-02 a2 = 6.281112E-02 a3 = 8.374816E-02 a4 = 6.281112E-02 a5 = 2.512445E-02 a6 = 4.187408E-03	b1 = 2.315806E+00 b2 = -3.293776E+00 b3 = 2.904826E+00 b4 = -1.694128E+00 b5 = 6.021426E-01 b6 = -1.029147E-01
0.25	a0 = 2.858110E-01 a1 = 5.716221E-01 a2 = 2.858110E-01	b1 = 5.423258E-02 b2 = -1.974768E-01	a0 = 7.015301E-02 a1 = 2.806120E-01 a2 = 4.209180E-01 a3 = 2.806120E-01 a4 = 7.015301E-02	b1 = 4.541481E-01 b2 = -7.417536E-01 b3 = 2.361222E-01 b4 = -7.096476E-02	a0 = 1.434449E-02 a1 = 8.606697E-02 a2 = 2.151674E-01 a3 = 2.868899E-01 a4 = 2.151674E-01 a5 = 8.606697E-02 a6 = 1.434449E-02	b1 = 1.076052E+00 b2 = -1.662847E+00 b3 = 1.191063E+00 b4 = -7.403087E-01 b5 = 2.752158E-01 b6 = -5.722251E-02

(продолжение)

$f_c$	2 полюса	4 полюса	6 полюсов
0.3	$a_0 = 3.849163E-01$ $a_1 = 7.698326E-01$ $a_2 = 3.849163E-01$ $b_1 = -3.249116E-01$ $b_2 = -2.147536E-01$	$a_0 = 1.335566E-01$ $a_1 = 5.342263E-01$ $a_2 = 8.013394E-01$ $a_3 = 5.342263E-01$ $a_4 = 1.335566E-01$ $b_1 = -3.904486E-01$ $b_2 = -6.784138E-01$ $b_3 = -1.412021E-02$ $b_4 = -5.392238E-02$ $b_5 = -3.278741E-02$	$a_0 = 3.997487E-02$ $a_1 = 2.398492E-01$ $a_2 = 5.996231E-01$ $a_3 = 7.994975E-01$ $a_4 = 5.996231E-01$ $a_5 = 2.398492E-01$ $a_6 = 3.997487E-02$ $b_1 = -2.441152E-01$ $b_2 = -1.130306E+00$ $b_3 = 1.063167E-01$ $b_4 = -3.463299E-01$ $b_5 = 8.882992E-02$ $b_6 = -3.278741E-02$
0.35	$a_0 = 5.001024E-01$ $a_1 = 1.000205E+00$ $a_2 = 5.001024E-01$ $b_1 = -7.158993E-01$ $b_2 = -2.845103E-01$	$a_0 = 2.340973E-01$ $a_1 = 9.363892E-01$ $a_2 = 1.404584E+00$ $a_3 = 9.363892E-01$ $a_4 = 2.340973E-01$ $b_1 = -1.263672E+00$ $b_2 = -1.080487E+00$ $b_3 = -3.276296E-01$ $b_4 = -7.376791E-02$	$a_0 = 9.792321E-02$ $a_1 = 5.875393E-01$ $a_2 = 1.468848E+00$ $a_3 = 1.958464E+00$ $a_4 = 1.468848E+00$ $a_5 = 5.875393E-01$ $a_6 = 9.792321E-02$ $b_1 = -1.627573E+00$ $b_2 = -1.955020E+00$ $b_3 = -1.075051E+00$ $b_4 = -5.106501E-01$ $b_5 = -7.239843E-02$ $b_6 = -2.639193E-02$
0.40	$a_0 = 6.362308E-01$ $a_1 = 1.272462E+00$ $a_2 = 6.362308E-01$ $b_1 = -1.125379E+00$ $b_2 = -4.195441E-01$	$a_0 = 3.896966E-01$ $a_1 = 1.558787E+00$ $a_2 = 2.338180E+00$ $a_3 = 1.558787E+00$ $a_4 = 3.896966E-01$ $b_1 = -2.161179E+00$ $b_2 = -2.033992E+00$ $b_3 = -8.789098E-01$ $b_4 = -1.610655E-01$	$a_0 = 2.211834E-01$ $a_1 = 1.327100E+00$ $a_2 = 3.317751E+00$ $a_3 = 4.423668E+00$ $a_4 = 3.317751E+00$ $a_5 = 1.327100E+00$ $a_6 = 2.211834E-01$ $b_1 = -3.058672E+00$ $b_2 = -4.390465E+00$ $b_3 = -3.523254E+00$ $b_4 = -1.684185E+00$ $b_5 = -4.414881E-01$ $b_6 = -5.767513E-02$
0.45	$a_0 = 8.001101E-01$ $a_1 = 1.600220E+00$ $a_2 = 8.001101E-01$ $b_1 = -1.556269E+00$ $b_2 = -6.441713E-01$	$a_0 = 6.291693E-01$ $a_1 = 2.516677E+00$ $a_2 = 3.775016E+00$ $a_3 = 2.516677E+00$ $a_4 = 6.291693E-01$ $b_1 = -3.077062E+00$ $b_2 = -3.641323E+00$ $b_3 = -1.949229E+00$ $b_4 = -3.990945E-01$	$a_0 = 4.760635E-01$ $a_1 = 2.856381E+00$ $a_2 = 7.140952E+00$ $a_3 = 9.521270E+00$ $a_4 = 7.140952E+00$ $a_5 = 2.856381E+00$ $a_6 = 4.760635E-01$ $b_1 = -4.522403E+00$ $b_2 = -8.676844E+00$ $b_3 = -9.007512E+00$ $b_4 = -5.328429E+00$ $b_5 = -1.702543E+00$ $b_6 = -2.303303E-01$

Таблица 20.2. Высокочастотные фильтры Чебышева с уровнем неравномерности АЧХ 0.5%

$f_c$	2 полюса	4 полюса	6 полюсов
0.01	$a_0 = 9.567529E-01$ $a_1 = -1.913566E+00$ $a_2 = 9.567529E-01$  $b_1 = 1.911437E+00$ $b_2 = -9.155749E-01$	$a_0 = 9.121579E-01$ $a_1 = -3.648632E+00$ $a_2 = 5.472947E+00$ $a_3 = -3.648632E+00$ $a_4 = 9.121579E-01$  $b_1 = 3.815952E+00$ $b_2 = -5.465026E+00$ $b_3 = 3.481295E+00$ $b_4 = -8.32259E-01$  $b_5 = 4.689218E+00$ $b_6 = -7.451429E-01$	$a_0 = 8.630195E-01$ $a_1 = -5.178118E+00$ $a_2 = 1.294529E+01$ $a_3 = -1.726039E+01$ $a_4 = 1.294529E+01$ $a_5 = -5.178118E+00$ $a_6 = 8.630195E-01$  $b_1 = 5.705102E+00$ $b_2 = -1.356935E+01$ $b_3 = 1.722231E+01$ $b_4 = -1.230214E+01$ $b_5 = 4.689218E+00$ $b_6 = -7.451429E-01$
0.025	$a_0 = 8.950355E-01$ $a_1 = -1.790071E+00$ $a_2 = 8.950355E-01$  $b_1 = 1.777932E+00$ $b_2 = -8.022106E-01$	$a_0 = 7.941874E-01$ $a_1 = -3.176750E+00$ $a_2 = 4.765125E+00$ $a_3 = -3.176750E+00$ $a_4 = 7.941874E-01$  $b_1 = 3.5538919E+00$ $b_2 = -4.722213E+00$ $b_3 = 2.814036E+00$ $b_4 = -6.318300E-01$	$a_0 = 6.912863E-01$ $a_1 = -4.147718E+00$ $a_2 = 1.036929E+01$ $a_3 = -1.382573E+01$ $a_4 = 1.036929E+01$ $a_5 = -4.147718E+00$ $a_6 = 6.912863E-01$  $b_1 = 5.261399E+00$ $b_2 = -1.157800E+01$ $b_3 = 1.363599E+01$ $b_4 = -9.063840E+00$ $b_5 = 3.223738E+00$ $b_6 = -4.793541E-01$
0.05	$a_0 = 8.001102E-01$ $a_1 = -1.600220E+00$ $a_2 = 8.001102E-01$  $b_1 = 1.556269E+00$ $b_2 = -6.441715E-01$	$a_0 = 6.291694E-01$ $a_1 = -2.516678E+00$ $a_2 = 3.775016E+00$ $a_3 = -2.516678E+00$ $a_4 = 6.291694E-01$  $b_1 = 3.077062E+00$ $b_2 = -3.641324E+00$ $b_3 = 1.949230E+00$ $b_4 = -3.990947E-01$	$a_0 = 4.760636E-01$ $a_1 = -2.856382E+00$ $a_2 = 7.140954E+00$ $a_3 = -9.521272E+00$ $a_4 = 7.140954E+00$ $a_5 = -2.856382E+00$ $a_6 = 4.760636E-01$  $b_1 = 4.522403E+00$ $b_2 = -8.676846E+00$ $b_3 = 9.007515E+00$ $b_4 = -5.328431E+00$ $b_5 = 1.702544E+00$ $b_6 = -2.303304E-01$
0.075	$a_0 = 7.142028E-01$ $a_1 = -1.428406E+00$ $a_2 = 7.142028E-01$  $b_1 = 1.338264E+00$ $b_2 = -5.185469E-01$	$a_0 = 4.965350E-01$ $a_1 = -1.986140E+00$ $a_2 = 2.979210E+00$ $a_3 = -1.986140E+00$ $a_4 = 4.965350E-01$  $b_1 = 2.617304E+00$ $b_2 = -2.749252E+00$ $b_3 = 1.325548E+00$ $b_4 = -2.524566E-01$	$a_0 = 3.259100E-01$ $a_1 = -1.955460E+00$ $a_2 = 4.888651E+00$ $a_3 = -6.518201E+00$ $a_4 = 4.888651E+00$ $a_5 = -1.955460E+00$ $a_6 = 3.259100E-01$  $b_1 = 3.787397E+00$ $b_2 = -6.288362E+00$ $b_3 = 5.747801E+00$ $b_4 = -3.041570E+00$ $b_5 = 8.808669E-01$ $b_6 = -1.122464E-01$

(продолжение)

$f_c$	2 полюса	4 полюса	6 полюсов
0.1	$a_0 = 6.362307E-01$ $a_1 = -1.272461E+00$ $a_2 = 6.362307E-01$	$a_0 = 3.896966E-01$ $a_1 = -1.558786E+00$ $a_2 = 2.338179E+00$ $a_3 = -1.558786E+00$ $a_4 = 3.896966E-01$	$b_1 = 2.161179E+00$ $b_2 = -2.033991E+00$ $b_3 = 8.789094E-01$ $b_4 = -1.610655E-01$ $b_5 = 4.414878E-01$ $b_6 = -5.767508E-02$
0.15	$a_0 = 5.0001024E-01$ $a_1 = -1.000205E+00$ $a_2 = 5.001024E-01$	$a_0 = 2.340973E-01$ $a_1 = -9.363892E-01$ $a_2 = 1.404584E+00$ $a_3 = -9.363892E-01$ $a_4 = 2.340973E-01$	$b_1 = 1.263672E+00$ $b_2 = -1.080487E+00$ $b_3 = 3.276296E-01$ $b_4 = -7.376791E-02$ $b_5 = 5.875393E-01$ $b_6 = 9.792321E-02$
0.2	$a_0 = 3.849163E-01$ $a_1 = -7.698326E-01$ $a_2 = 3.849163E-01$	$a_0 = 1.335566E-01$ $a_1 = -5.342262E-01$ $a_2 = 8.013393E-01$ $a_3 = -5.342262E-01$ $a_4 = 1.335566E-01$	$b_1 = 3.904484E-01$ $b_2 = -6.784138E-01$ $b_3 = 1.412016E-02$ $b_4 = -5.392238E-02$ $b_5 = -2.398492E-01$ $b_6 = 3.997486E-02$
0.25	$a_0 = 2.858111E-01$ $a_1 = -5.423343E-01$ $a_2 = 2.858111E-01$	$a_0 = 7.015302E-02$ $a_1 = -2.806121E-01$ $a_2 = 4.209182E-01$ $a_3 = -2.806121E-01$ $a_4 = 7.015302E-02$	$b_1 = -4.541478E-01$ $b_2 = -7.417535E-01$ $b_3 = -2.361221E-01$ $b_4 = -7.096475E-02$ $b_5 = -8.606701E-02$ $b_6 = 1.434450E-02$

(продолжение)

$f_c$	2 полюса	4 полюса	6 полюсов
0.3	$a_0 = 1.997396E-01$ $a_1 = -3.994792E-01$ $a_2 = 1.997396E-01$  $b_1 = -4.291049E-01$ $b_2 = -2.280633E-01$	$a_0 = 3.224553E-02$ $a_1 = -1.289821E-01$ $a_2 = 1.934732E-01$ $a_3 = -1.289821E-01$ $a_4 = 3.224553E-02$  $b_1 = -1.265912E+00$ $b_2 = -1.203878E+00$ $b_3 = -5.405908E-01$ $b_4 = -1.185538E-01$  $a_0 = 4.187407E-03$ $a_1 = -2.512444E-02$ $a_2 = 6.281111E-02$ $a_3 = -8.374815E-02$ $a_4 = 6.281111E-02$ $a_5 = -2.512444E-02$ $a_6 = 4.187407E-03$	$a_0 = 4.187407E-03$ $a_1 = -2.512444E-02$ $a_2 = 6.281111E-02$ $a_3 = -8.374815E-02$ $a_4 = 6.281111E-02$ $a_5 = -2.512444E-02$ $a_6 = 4.187407E-03$
0.35	$a_0 = 1.254285E-01$ $a_1 = -2.508570E-01$ $a_2 = 1.254285E-01$  $b_1 = -8.070777E-01$ $b_2 = -3.087918E-01$	$a_0 = 1.180009E-02$ $a_1 = -4.720335E-02$ $a_2 = 7.080051E-02$ $a_3 = -4.720335E-02$ $a_4 = 1.180009E-02$  $b_1 = -2.039039E+00$ $b_2 = -2.012961E+00$ $b_3 = -9.897915E-01$ $b_4 = -2.046700E-01$  $a_0 = 8.618665E-04$ $a_1 = -5.171200E-03$ $a_2 = 1.292800E-02$ $a_3 = -1.723733E-02$ $a_4 = 1.292800E-02$ $a_5 = -5.171200E-03$ $a_6 = 8.618665E-04$	$a_0 = 8.618665E-04$ $a_1 = -5.171200E-03$ $a_2 = 1.292800E-02$ $a_3 = -1.723733E-02$ $a_4 = 1.292800E-02$ $a_5 = -5.171200E-03$ $a_6 = 8.618665E-04$
0.40	$a_0 = 6.372801E-02$ $a_1 = -1.274560E-01$ $a_2 = 6.372801E-02$  $b_1 = -1.194365E+00$ $b_2 = -4.492774E-01$	$a_0 = 2.780754E-03$ $a_1 = -1.112302E-02$ $a_2 = 1.668453E-02$ $a_3 = -1.112302E-02$ $a_4 = 2.780754E-03$  $b_1 = -2.764031E+00$ $b_2 = -3.122854E+00$ $b_3 = -1.664534E+00$ $b_4 = -3.502233E-01$  $a_0 = 9.086141E-05$ $a_1 = -5.451685E-04$ $a_2 = 1.362921E-03$ $a_3 = -1.817228E-03$ $a_4 = 1.362921E-03$ $a_5 = -5.451685E-04$ $a_6 = 9.086141E-05$	$a_0 = 9.086141E-05$ $a_1 = -5.451685E-04$ $a_2 = 1.362921E-03$ $a_3 = -1.817228E-03$ $a_4 = 1.362921E-03$ $a_5 = -5.451685E-04$ $a_6 = 9.086141E-05$
0.45	$a_0 = 1.868823E-02$ $a_1 = -3.737647E-02$ $a_2 = 1.868823E-02$  $b_1 = -1.593937E+00$ $b_2 = -6.686903E-01$	$a_0 = 2.141509E-04$ $a_1 = -8.566037E-04$ $a_2 = 1.284906E-03$ $a_3 = -8.566037E-04$ $a_4 = 2.141509E-04$  $b_1 = -3.425455E+00$ $b_2 = -4.479272E+00$ $b_3 = -2.643718E+00$ $b_4 = -5.933269E-01$  $a_0 = 1.771089E-06$ $a_1 = -1.062654E-05$ $a_2 = 2.656634E-05$ $a_3 = -3.542179E-05$ $a_4 = 2.656634E-05$ $a_5 = -1.062654E-05$ $a_6 = 1.771089E-06$	$a_0 = 1.771089E-06$ $a_1 = -1.062654E-05$ $a_2 = 2.656634E-05$ $a_3 = -3.542179E-05$ $a_4 = 2.656634E-05$ $a_5 = -1.062654E-05$ $a_6 = 1.771089E-06$

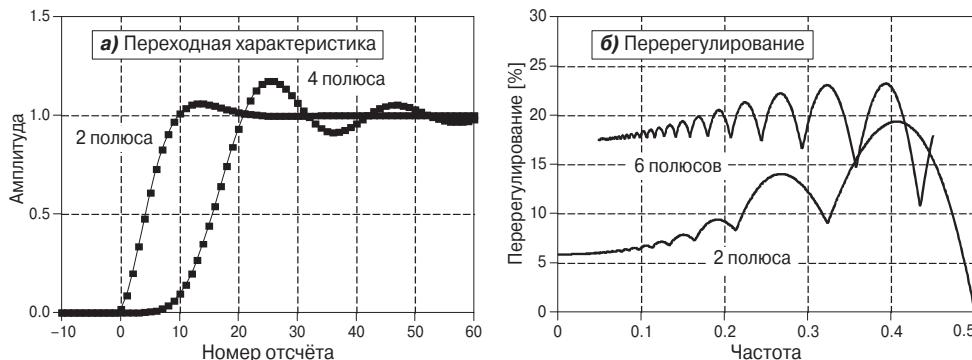
обратной связи для низкочастотных и высокочастотных фильтров с неравномерностью в полосе пропускания на уровне 0.5%. Если вам нужно быстро спроектировать более или менее нормальный фильтр, достаточно просто скопировать эти коэффициенты в свою программу.

Однако использование табличных значений приводит к двум серьёзным проблемам. Первая связана с ограничением в выборе параметров: в таблицах представлены лишь 12 значений граничной частоты полосы пропускания; фильтры имеют не более 6 полюсов; совершенно отсутствует возможность выбора уровня неравномерности АЧХ в полосе пропускания. Не имея возможности выбора этих параметров из непрерывного ряда значений, нельзя получить оптимальный фильтр, подходящий вам наилучшим образом. Вторая проблема состоит в том, что табличные коэффициенты придётся вводить в программу вручную, что оказывается весьма трудоёмким и не даёт возможности экспериментировать, выбирая то один, то другой фильтр.

Чтобы облегчить свой труд, попробуйте вместо ввода табличных данных воспользоваться программой расчёта коэффициентов. Примером такой программы может служить **Программа 20.1**. Достоинство этой программы состоит в относительной её простоте: необходимо только ввести упомянутые нами ранее четыре параметра фильтра, и программа сама запишет коэффициенты  $a_i$  и  $b_i$  в массивы  $A[ ]$  и  $B[ ]$ . Но есть и недостаток: программа должна обращаться к подпрограмме (**Программа 20.2**). После первого беглого просмотра эта подпрограмма может привести вас в ужас. Но не отчаивайтесь: она совсем не такая уж страшная, как это кажется на первый взгляд! В ней всего лишь один оператор условного перехода в строке 1120. Всё остальное — это последовательно выполняемые сложные математические операции. Подпрограмма использует шесть входных, пять выходных и пятнадцать временных переменных. В **Табл. 20.4** имеется два набора тестовых данных, необходимых для отладки подпрограммы. Подробнее работа программы рассмотрена в Главе 31.

## 20.3. Переходная характеристика и перерегулирование

Величина *перерегулирования* для фильтров *Баттерворт* и *Чебышева* лежит в диапазоне 5...30% и возрастает по мере увеличения количества полюсов. **Рис. 20.3а** позволяет сравнить переходные характеристики двух фильтров Чебышева. На (б) отражена характерная черта *цифровых фильтров*, которая не проявляется у аналоговых: величина перерегулирования переходной характеристики зависит (хоть и в небольшой степени) от частоты среза фильтра. Следовательно, оптимизация параметров чебышевских фильтров в частотной области без учёта их временных характеристик может привести к недопустимому увеличению перерегулирования и колебаний переходного процесса.



**Рис. 20.3.** Переходная характеристика фильтра Чебышева. Величина перерегулирования для чебышевских фильтров лежит в диапазоне 5...30% и зависит при этом от количества полюсов (**а**) и от частоты среза (**б**). На основе изображённых на (**а**) переходных характеристик, построенных для частоты среза 0.05, легко получить переходные характеристики для других значений частоты среза путём простого изменения масштаба.

## 20.4. Устойчивость

Область применения цифровых фильтров, основанных на использовании операции свёртки, ограничивается в большинстве случаев временем, требуемым на обработку сигнала. Можно достичь воспроизведения практически любых желаемых характеристик, но при условии, что вы можете достаточно долго ждать результата обработки. Для рекурсивных фильтров всё обстоит иначе. Обработка сигнала происходит почти молниеносно, но качество работы ограничено. В качестве примера рассмотрим низкочастотный фильтр с верхней граничной частотой, равной 0.01, характеризующийся наличием шести полюсов и уровнем неравномерности АЧХ 0.5%. Его коэффициенты можно найти по данным **Табл. 20.1**:

$a_0 = 1.391351E-10$	$b_1 = 5.883343E+00$
$a_1 = 8.348109E-10$	$b_2 = -1.442798E+01$
$a_2 = 2.087027E-09$	$b_3 = 1.887786E+01$
$a_3 = 2.782703E-09$	$b_4 = -1.389914E+01$
$a_4 = 2.087027E-09$	$b_5 = 5.459909E+00$
$a_5 = 8.348109E-10$	$b_6 = -8.939932E-01$
$a_6 = 1.391351E-10$	

Посмотрите внимательно на эти коэффициенты. Значения  $b_i$  по абсолютной величине близки к 10. С помощью простейших подсчётов можно найти, что вносимый каждым из них *шум округления* составляет около одной десятимиллионной доли от абсолютной величины, т. е.  $10^{-6}$ . А теперь обратим внимание на коэффициенты  $a_i$ , абсолютная величина которых близка к  $10^{-9}$ . Тут явно кроется какая-то ошибка. Полезный сигнал на выходе фильтра с учётом коэффициентов  $a_i$  оказывается в 1000 раз меньше *шума*, который вносят ранее полученные отсчёты выходного сигнала, умноженные на коэффициенты  $b_i$ . Такой фильтр просто не может работать! На практике максимальное число полюсов слабо зависит от уровня

неравномерности АЧХ и типа фильтра (низкочастотный или высокочастотный). В Табл. 20.3 даны приблизительные оценки максимального количества полюсов для случая вычислений с обычной точностью.

**Таблица 20.3. Максимальное число полюсов при работе с обычной точностью**

Частота среза	0.02	0.05	0.10	0.25	0.40	0.45	0.48
Максимальное число полюсов	4	6	10	20	10	6	4

Превышение этих максимальных значений приводит к тому, что качество фильтра начинает ухудшаться. Перерегулирование возрастает, затухание в зоне подавления снижается, а неравномерность АЧХ становится недопустимо большой. Если продолжать увеличивать количество полюсов фильтра при неточном представлении его коэффициентов, то на выходе могут возникнуть свободные колебания, сохраняющиеся до тех пор, пока в фильтре происходят переполнения.

Существует два способа увеличения максимального числа полюсов. Первый способ заключается в применении вычислений с двойной точностью. В этом случае потребуется также выбор двойной точности при вычислении весовых коэффициентов (и двойной точности для записи числа  $\pi$ ).

Второй способ, которым фактически пользуются специалисты, состоит в разбиении фильтра на несколько звеньев. Например, фильтр, имеющий шесть полюсов, можно представить в виде последовательного соединения трёх фильтров, каждый из которых имеет два полюса. **Программа 20.1** объединяет (для упрощения) весовые коэффициенты этих трёх фильтров в один набор, соответствующий результирующему фильтру. Следует помнить, что фильтр, представленный в виде совокупности отдельных простых звеньев, значительно более устойчив. Найти значения весовых коэффициентов  $a_i$  и  $b_i$  для каждого звена позволяет та же **Программа 20.1**. Отдельно для каждого звена вызывается **Программа 20.2**. Для фильтра с шестью полюсами подпрограмму (**Программа 20.2**) необходимо вызывать трижды. После своего завершения подпрограмма возвращает пять параметров:  $A_0, A_1, A_2, B_1$  и  $B_2$ . Это и есть весовые коэффициенты одного из двухполюсных фильтров, входящих в состав результирующего многоакаскадного фильтра.

### **Программа 20.1**

```

100 'ФИЛЬТР ЧЕБЫШЕВА - ВЫЧИСЛЕНИЕ ВЕСОВЫХ КОЭФФИЦИЕНТОВ
110 '
120      'НАЧАЛЬНЫЕ ЗНАЧЕНИЯ
130 DIM A[22] 'В массив А записываются коэффициенты  $a_i$ 
140 DIM B[22] 'В массив В записываются коэффициенты  $b_i$ 
150 DIM TA[22] 'Массив ТА используется для объединения звеньев
160 DIM TB[22] 'Массив ТВ используется для объединения звеньев
170 '
180 FOR I% = 0 TO 22
190 A[I%] = 0
200 B[I%] = 0
210 NEXT I%
220 '
230 A[2] = 1
240 B[2] = 1
250 PI = 3.14159265

```

```

260                                ' ВВОД ЧЕТЫРЁХ ПАРАМЕТРОВ ФИЛЬТРА
270 INPUT "Введите частоту среза(от 0 до .5):", FC
280 INPUT "Введите 0 для НЧ-фильтра, 1 для ВЧ-фильтра:", LH
290 INPUT "Введите уровень неравномерности АЧХ в процентах (0...29):", PR
300 INPUT "Введите количество полюсов (2,4,...20):", NP
310 '
320 FOR P% = 1 TO NP/2          ' ЦИКЛ ДЛЯ КАЖДОЙ ПАРЫ ПОЛЮСОВ
330 '
340 GOSUB 1000                ' к Программе 20.2
350 '
360 FOR I% = 0 TO 22           ' Добавление коэффициентов
370 TA[I%] = A[I%]
380 TB[I%] = B[I%]
390 NEXT I%
400 '
410 FOR I% = 2 TO 22
420 A[I%] = A0*TA[I%] + A1*TA[I%-1] + A2*TA[I%-2]
430 B[I%] = TB[I%] - B1*TB[I%-1] - B2*TB[I%-2]
440 NEXT I%
450 '
460 NEXT P%
470 '
480 B[2] = 0                  ' Окончание объединения коэффициентов
490 FOR I% = 0 TO 20
500 A[I%] = A[I%+2]
510 B[I%] = -B[I%+2]
520 NEXT I%
530 '
540 SA = 0                   ' НОРМИРОВАНИЕ КОЭФФИЦИЕНТА УСИЛЕНИЯ
550 SB = 0
560 FOR I% = 0 TO 20
570 IF LH = 0 THEN SA = SA + A[I%]
580 IF LH = 0 THEN SB = SB + B[I%]
590 IF LH = 1 THEN SA = SA + A[I%] * (-1)^I%
600 IF LH = 1 THEN SB = SB + B[I%] * (-1)^I%
610 NEXT I%
620 '
630 GAIN = SA / (1 - SB)
640 '
650 FOR I% = 0 TO 20
660 A[I%] = A[I%] / GAIN
670 NEXT I%
680 '                          ' Результирующие коэффициенты содержатся в A[ ] и B[ ]
690 END

```

### **Программа 20.2**

```

1000 ' ЭТА ПОДПРОГРАММА ВЫЗЫВАЕТСЯ ИЗ СТРОКИ 340 ПРОГРАММЫ 20.1
1010 '
1020 ' Входные переменные подпрограммы: PI, FC, LH, PR, HP, P%
1030 ' Выходные переменные подпрограммы: A0, A1, A2, B1, B2
1040 ' Внутренние переменные: RP, IP, ES, VX, KX, T, W, M, D, K,
1050 ' X0, X1, X2, Y1, Y2

```

```

1060 '
1070 ' Вычисление координат полюса на единичной окружности
1080 RP = -COS(PI/(NP*2) + (P%-1) * PI/NP)
1090 IP = SIN(PI/(NP*2) + (P%-1) * PI/NP)
1100 '
1110 ' Деформация окружности в эллипс
1120 IF PR = 0 THEN GOTO 1210
1130 ES = SQR( (100 / (100-PR))^2 - 1 )
1140 VX = (1/NP) * LOG( (1/ES) + SQR( (1/ES^2) + 1 ) )
1150 KX = (1/NP) * LOG( (1/ES) + SQR( (1/ES^2) - 1 ) )
1160 KX = (EXP(KX) + EXP(-KX)) / 2
1170 RP = RP * ( (EXP(VX) - EXP(-VX)) / 2 ) / KX
1180 IP = IP * ( (EXP(VX) + EXP(-VX)) / 2 ) / KX
1190 '
1200 ' Преобразование S-области в Z-область
1210 T = 2 * TAN(1/2)
1220 W = 2*PI*FC
1230 M = RP^2 + IP^2
1240 D = 4 - 4*RP*T + M*T^2
1250 X0 = T^2/D
1260 X1 = 2*T^2/D
1270 X2 = T^2/D
1280 Y1 = (8 - 2*M*T^2)/D
1290 Y2 = (-4 - 4*RP*T - M*T^2)/D
1300 '
1310 ' Преобразование НЧ в НЧ или НЧ в ВЧ
1320 IF LH = 1 THEN K = -COS(W/2 + 1/2) / COS(W/2 - 1/2)
1330 IF LH = 0 THEN K = SIN(1/2 - W/2) / SIN(1/2 + W/2)
1340 D = 1 + Y1*K - Y2*K^2
1350 A0 = (X0 - X1*K + X2*K^2) / D
1360 A1 = (-2*X0*K + X1 + X1*K^2 - 2*X2*K) / D
1370 A2 = (X0*K^2 - X1*K + X2) / D
1380 B1 = (2*K + Y1 + Y1*K^2 - 2*Y2*K) / D
1390 B2 = (-(K^2) - Y1*K + Y2) / D
1400 IF LH = 1 THEN A1 = -A1
1410 IF LH = 1 THEN B1 = -B1
1420 '
1430 RETURN

```

Приведенная программа вычисляет коэффициенты  $a_i$  и  $b_i$  для рекурсивных фильтров Чебышева. Четыре параметра вводятся в строках 270...300. Предполагается, что частота среза FC выражается в долях частоты дискретизации и принимает значения 0...0.5. Переменная LH принимает два значения: *единица* — для высокочастотного фильтра и *нуль* — для низкочастотного фильтра. Переменная PR выражает уровень неравномерности АЧХ фильтра в процентах и ограничена значениями 0...29. Количество полюсов задается переменной NP, которая может принимать чётные значения в диапазоне 2...20. По завершении выполнения программы значения коэффициентов  $a_i$  и  $b_i$  размещаются в массивах A[ ] и B[ ] ( $a_0 = A[0]$ ,  $a_1 = A[1]$  и т. д.). **Программа 20.2** должна быть вызвана из строки 340 главной программы. Подпрограмма (**Программа 20.2**) использует шесть входных и пять выходных переменных. В **Табл. 20.4** содержатся два набора данных для от-

ладки программы. Функции SIN и COS работают с радианами, а не с градусами. Функция LOG — натуральный логарифм (по основанию  $e$ ). Все переменные представлены в формате с плавающей точкой (включая константу  $\pi$ ) с двойной точностью, что позволяет увеличить число полюсов фильтра. Данные Табл. 20.1 и 20.2 получены с помощью приведенной программы и могут быть использованы для её тестирования. Математическое описание программы можно найти в Главе 33.

Таблица 20.4. Данные для тестирования Программы 20.2

НАБОР ДАННЫХ 1	НАБОР ДАННЫХ 2
<b>Входные переменные подпрограммы:</b>	
FC = 0.1	FC = 0.1
LH = 0	LH = 1
PR = 0	PR = 10
NP = 4	NP = 4
P% = 1	P% = 2
PI = 3.141592	PI = 3.141592
<b>Значения, которые должны получиться в строке 1200:</b>	
RP = -0.923879	RP = -0.136178
IP = 0.382683	IP = 0.933223
ES = не используется	ES = 0.484322
VX = не используется	VX = 0.368054
KX = не используется	KX = 1.057802
<b>Значения, которые должны получиться в строке 1310:</b>	
T = 1.092605	T = 1.092605
W = 0.628318	W = 0.628318
M = 1.000000	M = 0.889450
D = 9.231528	D = 5.656972
X0 = 0.129316	X0 = 0.211029
X1 = 0.258632	X1 = 0.422058
X2 = 0.129316	X2 = 0.211029
Y1 = 0.607963	Y1 = 1.038784
Y2 = -0.125227	Y2 = -0.789584
<b>Значения, которые подпрограмма возвращает в главную программу:</b>	
A0 = 0.061885	A0 = 0.922919
A1 = 0.123770	A1 = -1.845840
A2 = 0.061885	A2 = 0.922919
B1 = 1.048600	B1 = 1.446913
B2 = -0.296140	B2 = -0.836653

## СРАВНИТЕЛЬНЫЙ АНАЛИЗ ФИЛЬТРОВ

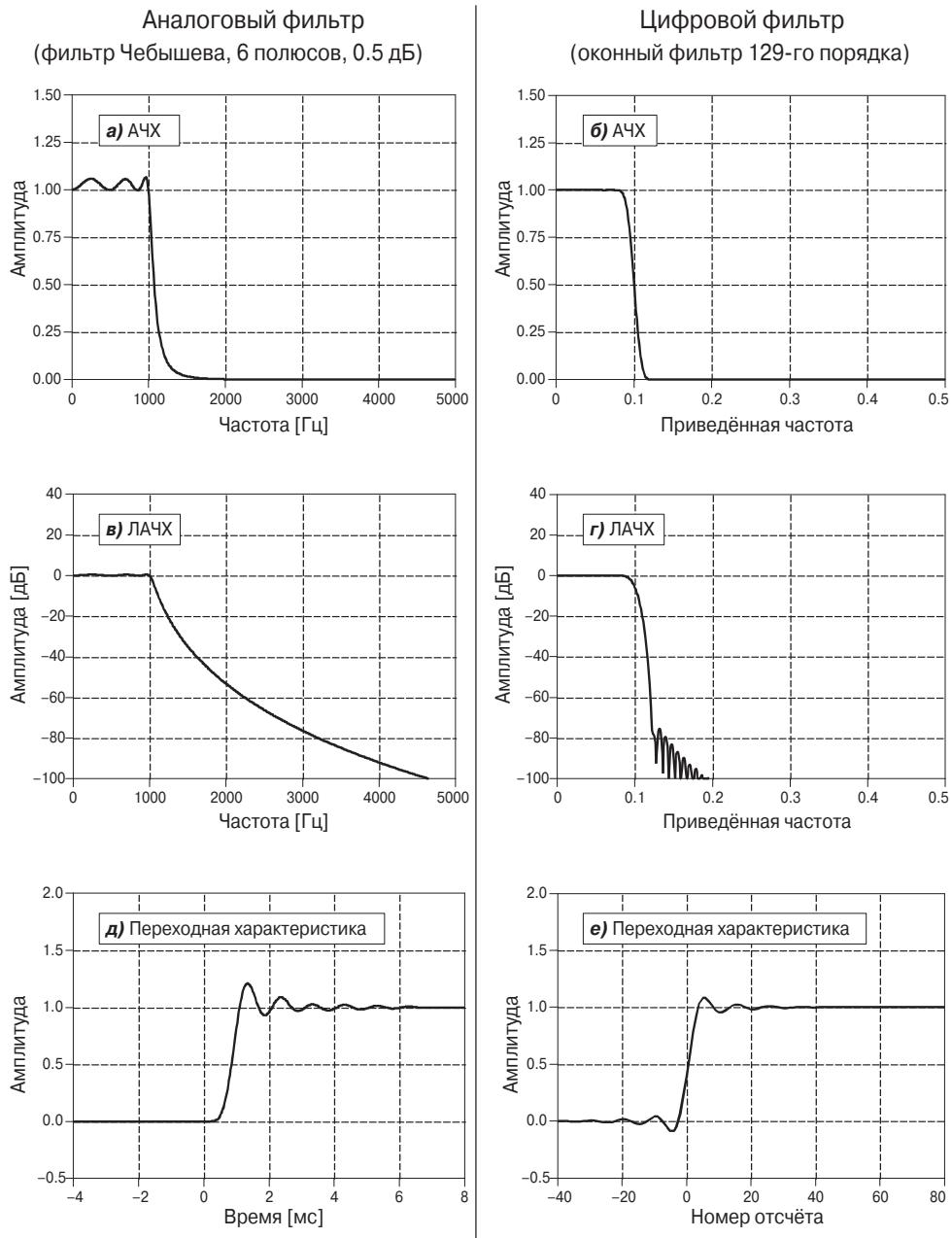
Как при столь широком разнообразии *фильтров* выбрать из них наиболее подходящий? Ответить на этот вопрос поможет данная глава, в которой мы устроим для фильтров «спортивное состязание». Для участия в турнире выберем лучших представителей в каждой группе. В первом раунде будут сражаться цифровой и аналоговый фильтры, чтобы выяснить, какая из двух технологий лучше. Во втором раунде за звание сильнейшего в частотной области будут бороться оконный фильтр и фильтр Чебышева. И наконец, в заключительной схватке столкнутся временные фильтры: однородный и однополюсный. Однако довольно слов! Начнём турнир!

### 21.1. Первый раунд: аналоговый фильтр против цифрового

В большинстве случаев сигналы, используемые в цифровой обработке, поступают с аналоговых схем. Что эффективнее: использовать *аналоговый фильтр* до дискретизации сигнала или *цифровой фильтр* после неё? Итак, выпускаем на ринг по одному лидеру из этих групп.

Пусть наша задача заключается в построении НЧ-фильтра с частотой среза 1 кГц. Со стороны аналоговых фильтров на «бой» выходит шестиполюсный *фильтр Чебышева* с неравномерностью в полосе пропускания 0.5 дБ (6%). Для такого фильтра требуется 3 операционных усилителя, 12 резисторов и 6 конденсаторов (см. Главу 3). В другом углу ринга разминается и вот уже готов к «бою» *оконный фильтр* с частотой дискретизации 10 кГц, т. е. частотой среза в приведенных единицах, равной 0.1. Порядок цифрового фильтра должен быть равен 129, что обеспечит одинаковую крутизну спада между уровнями 0.9 и 0.1 для аналогового и цифрового фильтров. Все должно быть справедливо: оба соперника поставлены в равные условия. АЧХ, ЛАЧХ и переходные характеристики фильтров показаны на Рис. 21.1.

Итак, «бой» начат. Неравномерность в полосе пропускания аналогового фильтра достигает 6% (**а**), в то время как у цифрового фильтра визуально она практически незаметна, а в действительности не превышает 0.02% (**б**). Сторонники аналоговых фильтров могут оспорить такое неравенство и потребовать сравнения фильтров с одинаковыми уровнями неравномерности. Подобное заявление является совершенно необоснованным. Неравномерность АЧХ аналоговых фильтров ограничена точностью параметров выбранных резисторов и конденсаторов. Даже при использовании фильтра Баттервортса, АЧХ которого теоретически не содержит колебаний, в реально действующей аналоговой схеме колебания



**Рис. 21.1.** Сравнительный анализ основных характеристик цифрового и аналогового фильтров. По многим характеристикам цифровой фильтр превосходит аналоговый: **а** и **б**) по уровню колебаний АЧХ в полосе пропускания; **в** и **г**) по крутизне спада и уровню затухания; **д** и **е**) по симметричности переходной характеристики. Верхняя граница полосы пропускания аналогового фильтра находится на частоте 1 кГц, а цифрового — на приведённой частоте 0.1, что позволяет говорить о равноправии фильтров при выборе частоты дискретизации 10 кГц.

могут достигать порядка 1%. Что касается цифровых фильтров, то для них гладкость АЧХ ограничена главным образом *ошибками округления*, поэтому они в сотни раз превосходят аналоговые фильтры. Итак, один-ноль, либо счёт 1:0 в пользу цифрового фильтра.

Далее рассмотрим частотную характеристику в логарифмическом масштабе (*в* и *г*). Цифровой фильтр снова одерживает убедительную победу: по спаду АЧХ и по затуханию в зоне подавления. Даже улучшая качество аналогового фильтра путём введения дополнительных каскадов, нам всё равно не достичь сравнимых с цифровым фильтром характеристик. Пусть, к примеру, требуется улучшить два вышеупомянутых параметра в 100 раз. В случае оконного фильтра это достигается с помощью простых преобразований, а для аналоговой схемы оказывается фактически невозможно. Ещё два балла в пользу цифрового фильтра.

*Переходные характеристики* показаны на (*д* и *е*). У цифрового звена наблюдается симметричный переходной процесс, т. е. фильтр имеет *линейную ФЧХ*. Переходная характеристика аналогового фильтра явно несимметрична, т. е. ФЧХ далека от линейной. *Перерегулирование* аналогового фильтра составляет 20%, но колебания возникают только после подъёма переходной характеристики. У цифрового фильтра колебания достигают примерно 10%, но наблюдаются как перед подъёмом, так и после него. На этот раз ни одному из фильтров не удалось превзойти соперника, и счёт остаётся прежним.

Несмотря на столь явную победу цифрового фильтра, во множестве приложений следует и даже необходимо использовать аналоговые фильтры. Это связано не с самой работой фильтра (т. е. с тем, что у него на входе и на выходе), а с преимуществами общего характера, которые аналоговые схемы имеют перед цифровыми. Первое преимущество — скорость: цифровые фильтры медленнее аналоговых. Например, персональный компьютер способен при использовании быстрой свёртки обрабатывать около 10 000 отсчётов в секунду, тогда как даже простой операционный усилитель может работать на частотах 100...1000 кГц, т. е. на два-три порядка быстрее цифровой системы!

Второе неотъемлемое преимущество, обусловленное природой аналоговых фильтров, — это *динамический диапазон*. Здесь различают два показателя: амплитудный и частотный динамические диапазоны. *Амплитудный динамический диапазон* — это отношение максимального уровня сигнала, который способна обработать система, к уровню *собственного шума*. Если разрядность АЦП составляет 12 бит, то уровень насыщения равен 4095, а среднеквадратическое отклонение шума квантования не превышает 0.29 цены младшего бита, следовательно, динамический диапазон приблизительно равен 14000. Для сравнения: напряжение насыщения стандартных операционных усилителей достигает 20 В, а уровень собственного шума — около 2 мкВ, что позволяет реализовать систему с динамическим диапазоном, равным 10 миллионам. Опять же простой операционный усилитель явно превосходит цифровую систему.

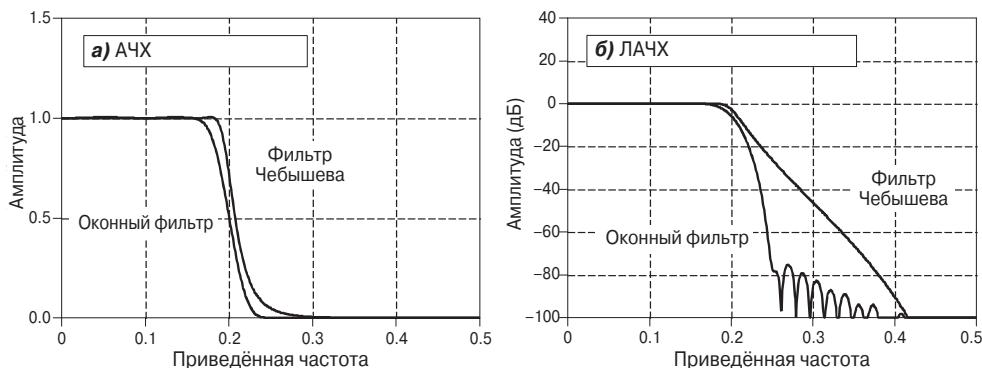
Теперь перейдем ко второму показателю, которым является *частотный динамический диапазон*. На основе операционного усилителя довольно просто создать схему, позволяющую обрабатывать одновременно частоты 0.01...100 кГц (семь декад). При попытке применить цифровую обработку система просто оказывается «зашлой» входными данными: при частоте дискретизации 200 кГц один период сигнала частотой 0.01 Гц простирается на 20 миллионов отсчётов. Возможно,

вы уже заметили, что частотная характеристика цифровых фильтров почти всегда строится на линейной частотной шкале, в то время как для аналоговых обычно применяют логарифмическую. Причина в том, что линейная шкала больше подходит для отображения замечательных фильтрующих свойств цифровых фильтров, а логарифмическая позволяет показать широкий динамический диапазон аналоговых схем.

## 21.2. Второй раунд: оконный фильтр против фильтра Чебышева

И оконные фильтры, и фильтры Чебышева предназначены для частотной селекции. Разница в том, что оконные фильтры относятся к классу **КИХ-фильтров** и используют операцию свёртки, а фильтры Чебышева относятся к классу **БИХ-фильтров** и имеют рекурсивную структуру. Какой же из фильтров окажется лучшим в частотной области? Начнём поединок!

Претендентом на победу со стороны рекурсивных фильтров будет 6-полюсный НЧ-фильтр Чебышева с неравномерностью 0.5%. Сопоставимость фильтров осложнена тем, что частотная характеристика фильтра Чебышева зависит от верхней граничной частоты. Пусть для определённости частота среза равна 0.2, а порядок оконного фильтра равен 51. При таких условиях оба фильтра имеют одинаковую скорость спада АЧХ между уровнями 0.9 и 0.1 (Рис. 21.2а).

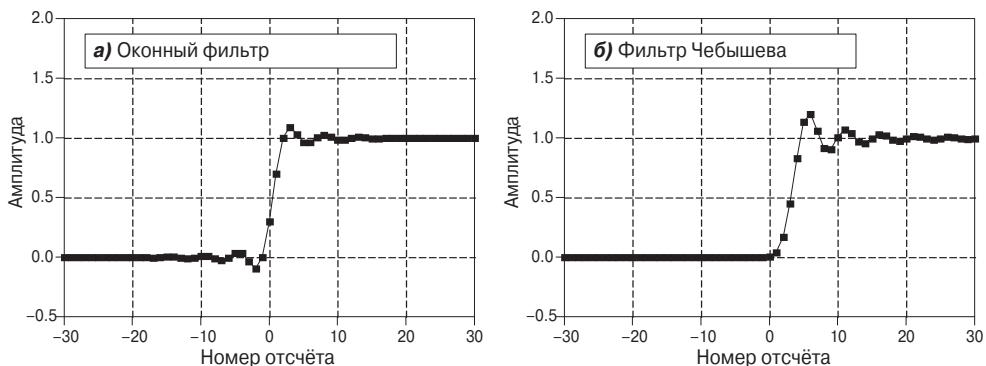


**Рис. 21.2.** Частотные характеристики оконного фильтра 51-го порядка и 6-полюсного фильтра Чебышева с уровнем неравномерности 0.5%. Окноный фильтр обеспечивает более сильное затухание в зоне подавления, но в приложениях со средними требованиями данное преимущество не играет роли. Верхняя граничная частота обоих фильтров равна 0.2, причём у оконного фильтра она измеряется по уровню 0.5, а у фильтра Чебышева — по уровню 0.707.

Посмотрим, какой из фильтров окажется сильнее в этой «схватке». Уровень колебаний АЧХ в полосе пропускания фильтра Чебышева составляет 0.5%; у оконного фильтра колебания отсутствуют. Однако при необходимости можно задать нулевой уровень неравномерности и для фильтра Чебышева, поэтому оба фильтра оказываются в равном положении. Зато по уровню затухания в зоне по-

давления побеждает оконный фильтр, что хорошо видно по Рис. 21.2б. Счёт «один—ноль» в пользу оконного фильтра.

На Рис. 21.3 сравниваются *переходные характеристики* фильтров. Как и следовало ожидать, для фильтров частотной селекции временные характеристики оказались очень невысокого качества. Рекурсивный фильтр имеет *нелинейную fazу*, но это может быть легко исправлено применением двунаправленной фильтрации. Объявляем ничью: на этот раз ни один из фильтров не имеет явных преимуществ.

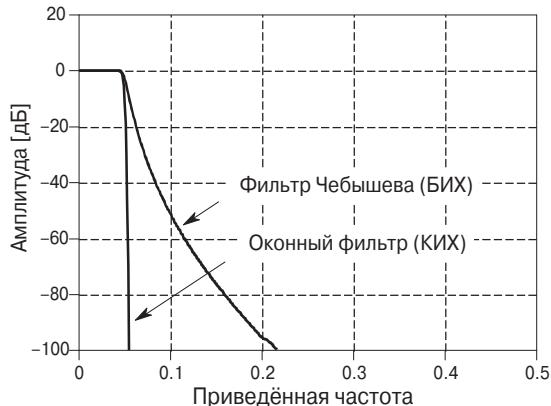


**Рис. 21.3.** Переходные характеристики оконного фильтра 51-го порядка и 6-полюсного фильтра Чебышева (уровень неравномерности — 0.5%). Приведенная частота среза обоих фильтров равна 0.2. Переходная характеристика оконного фильтра несколько лучше: перерегулирование меньше, фаза линейна (характеристика симметрична).

До настоящего момента мы не обнаружили никакого существенного различия в работе фильтров, т. е. в приложениях со средними требованиями оба фильтра работают одинаково хорошо. Заметное преимущество проявляется только в двух крайних случаях: при достижении максимальной разрешающей способности фильтра по частоте и вычислительной эффективности. Оконный фильтр обеспечивает наивысшее разрешение (можно сказать, что он более «мощный»), а фильтр Чебышева является в свою очередь более быстрым и «проворным». Представьте, что возникла по-настоящему серьёзная проблема: нужно выделить сигнал напряжением 100 мВ с частотой 61 Гц, распространяющийся по сети электропитания 120 В/60 Гц. На Рис. 21.4 сравниваются частотные характеристики двух рассмотренных фильтров, рассчитанных с целью достижения наивысшего разрешения по частоте. Фильтр Чебышева имеет 6 полюсов и уровень неравномерности 0.5%. Такое число полюсов является максимально возможным при частоте среза 0.05 и одинарной точности вычислений. Оконный фильтр 1001-го порядка получен в результате свёртки импульсной характеристики фильтра 501-го порядка с самой собой. В Главе 16 говорилось, что данное преобразование повышает уровень затухания в зоне подавления оконного фильтра.

Какой же из фильтров победит при необходимости получить максимальное разрешение по частоте? Оконный фильтр отправляет фильтр Чебышева в нокаут! Любые попытки улучшить рекурсивный фильтр (увеличение числа полюсов, введение дополнительных каскадов, переход к двойной точности и т. д.) не позволяют достичь характеристик, близких к характеристикам КИХ-фильтра. Это осо-

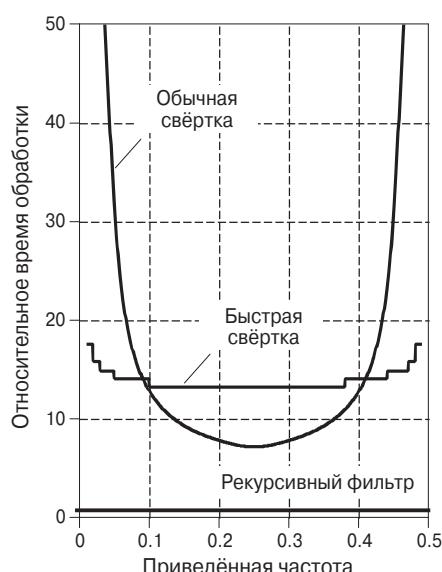
**Рис. 21.4.** Достижение максимальной разрешающей способности при использовании КИХ- и БИХ-фильтров. В классе КИХ-цепей (оконный фильтр) возможно воспроизведение частотной характеристики практически любой заданной формы, тогда как возможности БИХ-цепей (фильтр Чебышева) очень ограничены. ЛАЧХ построены для 6-полюсного фильтра Чебышева и оконного фильтра 1001-го порядка.



бенно впечатляет, если учесть, что оконный фильтр нанёс сейчас лишь свой первый удар. Рекурсивные фильтры отличаются ограниченной точностью воспроизведения желаемых частотных характеристик. Напротив, при использовании оконных фильтров она может достичь невероятной степени точности. Сказанное справедливо при одном условии: если у вас достаточно времени на вычисления. Теперь пора рассмотреть второй критический параметр — вычислительную эффективность.

Сравнивать быстроту обработки отсчётов входного сигнала 6-полюсным рекурсивным фильтром и оконным фильтром — всё равно что наблюдать состязание Феррари и карта (**Рис. 21.5**). Известно, что крутизна спада АЧХ рекурсивного фильтра повышается по мере приближения частоты среза либо к нулевой частоте, либо к половине частоты дискретизации. В силу сохранения равенства по разрешающей способности (чтобы соревнования между фильтрами проводились «честно») необходимо увеличивать порядок оконного фильтра. В результате вблизи

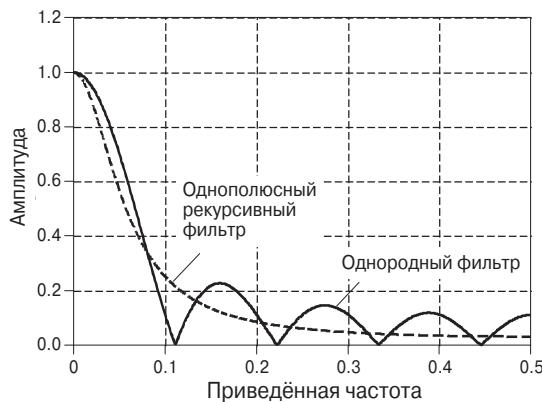
**Рис. 21.5.** Сравнение КИХ- и БИХ-фильтров по скорости обработки отсчётов входного сигнала. Показаны кривые, отражающие относительное время обработки сигналов оконным фильтром и сравнимым по качеству 6-полюсным фильтром Чебышева. Для оконного фильтра рассмотрены две структуры: на основе стандартной и быстрой свёртки. Время обработки для оконного фильтра возрастает на низких и высоких частотах, т.к. необходимо увеличивать его порядок, чтобы сохранять равенство по крутизне спада АЧХ, которая для фильтра Чебышева зависит от выбора частоты среза. Как правило, БИХ-фильтры оказываются на порядок быстрее сравнимых с ними по точностным параметрам КИХ-фильтров.



частот 0 и 0.5 время обработки оконным фильтром увеличивается. Таким образом, при сравнимых точностных характеристиках КИХ-фильтры работают на порядок медленнее, чем БИХ-фильтры (карт — 25 км/ч; Феррари — 250 км/ч).

## 21.3. Третий раунд: однородный фильтр против однополюсного

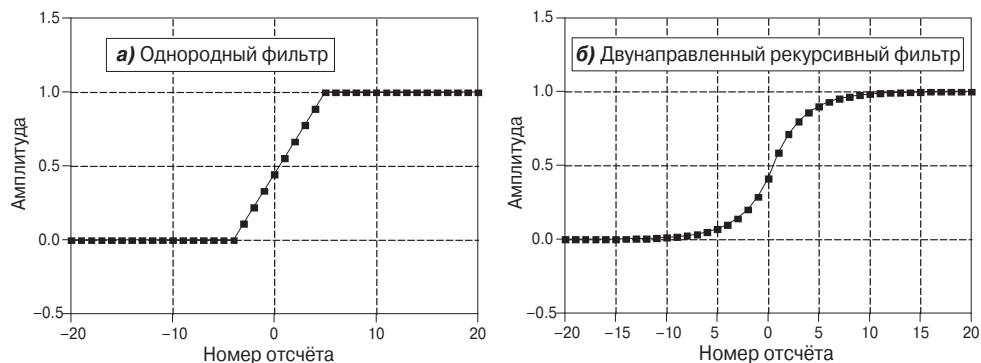
В третьем «бою» сразятся фильтры, работающие во временной области. Первым на ринг выходит *однородный фильтр* с усреднением по 9 отсчётам. Его соперник — *однополосный рекурсивный фильтр*, построенный по двунаправленной технологии. Для получения сопоставимой частотной характеристики коэффициент «забывания» однополосного фильтра  $x = 0.70$ . Состязание между фильтрами начинается с **Рис. 21.6**, где показаны АЧХ обоих фильтров. Характеристики явно не впечатляют, что и понятно, так как фильтры предназначены вовсе не для частотной селекции. Поэтому баллы не присуждаются ни одному из фильтров.



**Рис. 21.6.** АЧХ однородного и однополосного рекурсивного фильтров. Частотные характеристики не отличаются высоким качеством, что вполне естественно для фильтров, работающих во временной области.

*Переходные характеристики* фильтров изображены на **Рис. 21.7**. *Переходный процесс* однородного фильтра (**а**) имеет вид отрезка прямой, что соответствует наиболее быстрому переходу с одного уровня на другой. Переходная характеристика рекурсивного фильтра (**б**) имеет гладкий переходный процесс, что в некоторых случаях обеспечивает преимущество. По одному баллу получают оба фильтра.

Фильтры мало отличаются по качеству работы, поэтому выбор часто зависит только от предпочтений разработчика. Тем не менее некоторое преимущество обнаруживается при анализе двух характеристик — времени разработки фильтра и времени обработки отсчётов входного сигнала. В первом случае требуется сократить время разработки фильтра, пренебрегая увеличением вычислительных затрат. Например, необходимо всего один раз воспользоваться фильтром, порядок которого равен нескольким тысячам. Поскольку вся программа фильтрации вы-



**Рис. 21.7.** Переходные характеристики однородного фильтра и двунаправленного однополюсного фильтра. Преимуществом однородного фильтра является более короткий переходный процесс; преимуществом однополюсного фильтра — слаженный переходный процесс.

полняется всего за несколько секунд, бессмысленно тратить время на оптимизацию алгоритма. Ясно, что предпочтение следует отдать вычислениям в формате с плавающей точкой. Остаётся выбрать из двух вариантов: либо однородный фильтр, реализуемый на базе свёртки, либо однополюсный рекурсивный фильтр. Победителем здесь будет рекурсивный фильтр. Его несколько проще программировать и настраивать, а его вычислительные затраты намного меньше.

Во втором случае всё наоборот: необходимо минимизировать вычислительные затраты, а времени на разработку достаточно. Например, фильтр может входить в состав серийно выпускаемого изделия, где будет использован миллионы раз. Для ускорения обработки, вероятно, нужно будет использовать целые числа. Выбор здесь производится между однородным фильтром на основе рекурсивного алгоритма и однополюсным рекурсивным фильтром, использующим справочную таблицу (таблицу соответствия) или целочисленную математику. Победителем оказывается однородный фильтр. Он работает быстрее и не создаёт проблем, связанных с целочисленной математикой, на стадиях расчёта и обработки.

## ОБРАБОТКА ЗВУКА

Задачи обработки звуковых сигналов встречаются во многих приложениях, связанных с воспроизведением звука для прослушивания. Наиболее значимыми являются три направления: 1) высококачественное воспроизведение звука, такое как проигрывание лазерных компакт-дисков; 2) голосовые телекоммуникации, т. е. телефонные сети; 3) искусственная речь, когда компьютер генерирует и распознает образцы человеческой речи. Несмотря на то что эти приложения имеют различные цели и специфику задач, все они связаны общим критерием оценки качества результата: «судьей» выступает ухо человека. Цифровая обработка произвела революционные изменения в этих и других задачах обработки звуковых сигналов.

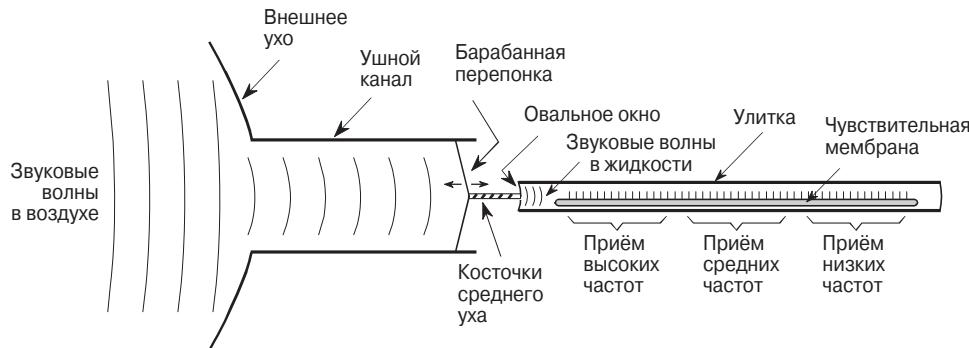
### 22.1. Слух человека

Ухо человека имеет чрезвычайно сложное строение. Но ещё большую сложность придаёт слуху человека то, что информация поступает от двух ушей и объединяется в сложной нервной сети и мозге. Заметим, что приводимое в этой книге описание механизма восприятия человеком звуков является лишь поверхностным обзором. В действительности в функционировании этой системы имеется много тонких моментов и слабо изученных явлений.

На Рис. 22.1 показаны основные структуры и проиллюстрированы процессы, лежащие в основе функционирования уха человека. Внешнее ухо состоит из двух частей — кожной складки и хряща, расположенных снаружи головы, и ушного канала, представляющего собой трубку диаметром около 0.5 см, расположенную внутри головы и имеющую длину около 3 см. Задача внешнего уха состоит в том, чтобы улавливать звуки из внешней среды и передавать их к чувствительным среднему и внутреннему уху, безопасно размещённым внутри черепной коробки. На конце ушного канала натянут тонкий лист ткани — барабанная перепонка. Звуковая волна падает на барабанную перепонку и заставляет её вибрировать. Среднее ухо — множество маленьких косточек — передаёт эти вибрации улитке — внутреннему уху, где они преобразуются в нервные импульсы. Улитка представляет собой заполненную жидкостью трубку диаметром примерно 2 мм и длиной 3 см. Хотя на Рис. 22.1 улитка показана прямой, в действительности она скручена и похожа на раковину маленькой улитки, откуда и происходит её название.

Когда звуковая волна проходит из воздушной среды в жидкость, только малая её часть передаётся через поверхность раздела, в то время как основная часть энергии отражается. Это происходит потому, что воздух имеет низкий механический импеданс (т. е. низкое акустическое давление и высокую скорость перемещения частиц, что является результатом низкой плотности и высокой сжимаемос-

ти), в то время как жидкость имеет высокий механический импеданс. Это различие в импедансах заставляет большую часть звука отражаться от поверхности раздела воздух/жидкость.



**Рис. 22.1.** Структура уха человека. Внешнее ухо принимает звуковые волны из окружающей среды и доставляет их к барабанной перепонке — тонкому листу натянутой ткани, которая вибрирует синхронно с колебанием воздуха. Косточки среднего уха (молоточек, наковалня и стремя) передают эти вибрации к овальному окну — гибкой мембрane, расположенной на конце заполненной жидкостью улитки. Внутри улитки располагается чувствительная мембрана, содержащая около 12 000 нервных клеток, формирующих слуховой нерв. Благодаря различию жёсткости этой мембранны на разных её участках каждая нервная клетка откликается только на узкий диапазон звуковых частот, делая ухо анализатором спектра частот.

Среднее ухо выполняет функцию согласования импедансов, увеличивая часть звуковой энергии, проникающей в жидкость внутреннего уха. Например, у рыб нет барабанной перепонки и среднего уха, потому что им нет необходимости слышать в воздухе. Согласование импедансов осуществляется в основном за счёт различия в площадях мембранны барабанной перепонки (принимающей звук из воздуха) и овального окна (передающего звук в жидкость, см. **Рис. 22.1**). Барабанная перепонка имеет площадь около  $60 \text{ mm}^2$ , в то время как овальное окно — около  $4 \text{ mm}^2$ . Так как давление определяется отношением силы к площади, это различие в площадях увеличивает давление звуковой волны почти в 15 раз.

Основной частью улитки является мембрана, содержащая около 12 000 чувствительных ячеек, формирующих слуховой нерв. Эта мембрана обладает высокой жёсткостью со стороны овального окна и постепенно становится более гибкой к противоположному краю. За счёт этого она приобретает способность функционировать как анализатор частотного спектра. Когда на мембрану действует сигнал с высокой частотой, она резонирует в том месте, где её жесткость повышена, возбуждая нервные клетки около овального окна. Если же звуковая волна характеризуется низкой частотой, то возбуждаются нервные клетки на противоположном конце мембранны. Таким образом, различные волокна, образующие слуховой нерв, оказываются настроенными на определённые частоты. Этот принцип передачи информации сохраняется на всей звуковой магистрали, ведущей в мозг.

Есть и другой принцип передачи информации, также используемый ухом человека. Нервные клетки кодируют звуковую информацию, генерируя короткие электрические импульсы — биоэлектрические потенциалы — в ответ на каждый период вибрации. Например, звуковая волна с частотой 200 Гц представляется нейроном последовательностью биопотенциалов с частотой 200 импульсов в се-

кунду. Однако такой механизм работает только на частотах не выше 500 Гц — максимальной частоты, с которой нейроны могут создавать биопотенциалы. Ухо человека решает эту проблему использованием нескольких нервных клеток для воспроизведения одной частоты. Например, тон 3000 Гц может быть представлен десятью нервными клетками, попеременно «запускающимися» 300 раз в секунду. Это расширяет диапазон передаваемых частот до 4 кГц, что превышает возможности первого описанного принципа передачи информации.

**Табл. 22.1** иллюстрирует соотношение между интенсивностью звука и воспринимаемой громкостью. Общепринято для измерения интенсивности звука использовать логарифмическую шкалу и единицы, называемые *декибелами уровня звукового давления* (dB УЗД). На этой шкале 0 dB УЗД — это мощность звуковой волны, равная  $10^{-16}$  Вт/см<sup>2</sup>, что соответствует самой низкой мощности звука, детектируемой ухом человека. Для обычной речи человека характерна мощность порядка 60 dB УЗД, а порог болевых ощущений лежит на уровне 140 dB УЗД.

Разница между самым громким и самым слабым звуками, которые способен слышать человек, около 120 dB, что соответствует диапазону изменения амплитуды в миллион раз. Слушатель может ощутить перемену громкости, когда сигнал изменяется на 1 dB (12%-е изменение амплитуды). Таким образом, существует только около 120 уровней громкости, которые может различить человек, — от шёпота до самого громкого раската грома. Чувствительность уха изумляет: при прослушивании самых слабых звуков барабанная перепонка колеблется на величину, меньшую диаметра молекулы!

Таблица 22.1. Единицы интенсивности звука

	Вт/см <sup>2</sup>	дБ УЗД	Пример звука
Громче ↑ ↓ Типе	$10^{-2}$	140	Болевой порог
	$10^{-3}$	130	
	$10^{-4}$	120	Дискомфорт
	$10^{-5}$	110	Отбойный молоток или рок-концерт
	$10^{-6}$	100	
	$10^{-7}$	90	Допустимый предел индустриального шума
	$10^{-8}$	80	
	$10^{-9}$	70	
	$10^{-10}$	60	Обычный разговор
	$10^{-11}$	50	
	$10^{-12}$	40	Самый слабый звук на 100 Гц
	$10^{-13}$	30	
	$10^{-14}$	20	Самый слабый звук на 10 кГц
	$10^{-15}$	10	
	$10^{-16}$	0	Самый слабый звук на 3 кГц
	$10^{-17}$	-10	
	$10^{-18}$	-20	

**Примечание.** Интенсивность звука выражается в мощности на единицу площади (Вт/см<sup>2</sup>) или в логарифмическом масштабе в децибелах УЗД. Как показывает эта таблица, слух человека имеет наибольшую чувствительность в диапазоне 1...4 кГц

Восприятие громкости соотносится с мощностью звука как степень  $1/3$ . Например, если вы увеличили мощность звука в 10 раз, слушатель скажет, что громкость увеличилась в 2 раза ( $10^{1/3} \approx 2$ ). Это является основной проблемой при необходимости уменьшить нежелательные звуки из окружающей среды, например оглушающее стерео из соседней квартиры. Предположим, вы прилежно покрыли 99% стены в своей комнате совершенно звуконепроницаемым материалом, оставив только 1% площади поверхности незащищенным (всегда есть двери, углы, отдушины и т. п.). Хотя мощность звука упадет при этом до 1% от её первоначального значения, воспринимаемая громкость уменьшится только в  $0.01^{1/3} \approx 0.2$  раза, т. е. до уровня 20%.

Обычно считается, что ухо человека улавливает звуки в диапазоне частот 20 Гц...20 кГц, проявляя наивысшую чувствительность к частотам 1...4 кГц. Например, слушатель способен слышать на 3 кГц слабые звуки на уровне 0 дБ УЗД, а на частоте 100 Гц ему потребуется 40 дБ УЗД (т. е. в 100 раз большая амплитуда сигнала). Кроме того, человек может сказать, что два тона различны, если их частоты отличаются на 0.3% на 3 кГц. На 100 Гц требуется различие частот на 3%. Для сравнения: соседние клавиши на пианино отличаются по частоте примерно на 6%.

Основное преимущество в восприятии звука двумя ушами — способность определять его направление. Слушатель может ощутить различие между двумя источниками звука, которые удалены всего на три градуса друг от друга, т. е. примерно на ширину человека на расстоянии 10 м. Информация о направлении детектируется двумя разными способами. Первый состоит в том, что частоты выше 1 кГц сильно «затеняются» головой. То есть ухо, расположенное ближе к источнику звука, принимает гораздо более сильный сигнал, чем другое. Второй способ определения направления основан на том, что ухо, расположенное дальше от источника звука, слышит его немного позже, чем ближнее, вследствие большего расстояния до источника. Если принять во внимание, что размер головы составляет в среднем 22 см, а скорость звука — 340 м/с, то угловое разрешение в три градуса означает разрешение по времени, равное 30 мкс. Этот способ определения направления в основном используется для звуков с частотой меньше 1 кГц.

Использовать оба этих способа получения информации о направлении помогает способность поворачивать голову и наблюдать изменения в сигналах. Интересное ощущение возникает, когда оба уха слышат звуки одинаково, как, например, при прослушивании монозаписей через наушники. Наш мозг при этом считает, что звук идёт прямо по центру головы!

Ухо человека способно достаточно хорошо идентифицировать направление на источник звука, но оно очень плохо определяет расстояние до источника. Это происходит потому, что звуковая волна содержит очень мало факторов, несущих такую информацию. Например, человеку свойственно считать, что высокочастотные звуки формируются на близком расстоянии, а низкочастотные — на дальнем. Дело в том, что звуковые волны рассеивают свои верхние частоты при распространении на большие расстояния. Другим фактором, усложняющим измерение расстояния, является эхо. На характер звука влияет объём помещения. Например, для звуков в большой аудитории будут характерны эхо-сигналы с интервалом около 100 мс, а для небольших офисных помещений — с интервалом 10 мс. Некоторые виды животных смогли использовать эту проблему с пользой.

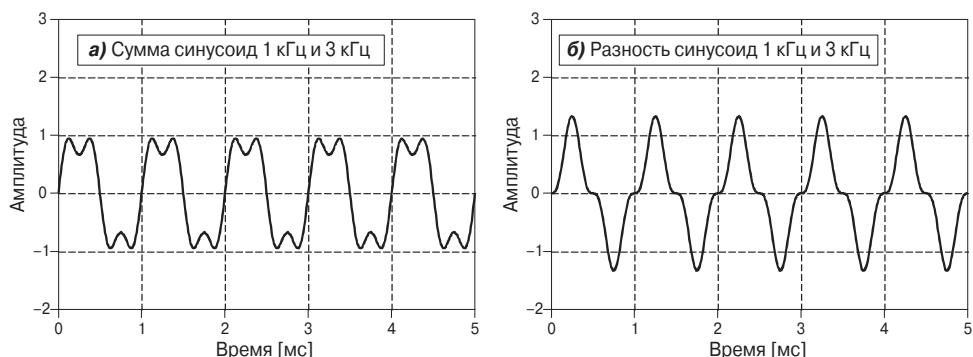
Они измеряют расстояние с помощью активной локации. Например, летучие мыши и дельфины издают щелчки, которые отражаются от близко расположенных объектов. Измеряя интервалы времени между щелчком и его эхом, эти животные определяют расстояние до объектов с точностью до 1 см. Эксперименты показали, что некоторые люди, в особенности слепые, также могут в некоторой степени использовать принцип активной локации.

## 22.2. Тембр

Восприятие непрерывного звука, такого как нота, взятая на музыкальном инструменте, в основном определяется тремя факторами: *громкостью*, *высотой* и *тембром*. Громкость — это мера интенсивности звуковой волны, о чём говорилось ранее. Высота — это частота основного компонента в звуке, т. е. частота, с которой волна повторяет сама себя. Хотя имеются тонкие особенности восприятия этих параметров звука, в целом они строго согласованы с известными физическими величинами.

Тембр более сложно охарактеризовать, так как он определяется гармоническим составом сигнала. **Рис. 22.2** иллюстрирует две волны, каждая из которых представляет собой сумму двух синусоидальных волн: первая — с частотой 1 кГц и амплитудой, равной единице, и вторая — с частотой 3 кГц и амплитудой 1/2. Разница между этими двумя сигналами состоит в том, что один из них (**б**) включает высокочастотный сигнал, который перед суммированием был проинвертирован. То есть гармоника с частотой 3 кГц имеет сдвиг по фазе на 180 градусов по сравнению с гармоникой 1 кГц. Несмотря на различие сигналов во временной области, оба они звучат идентично. Это происходит потому, что на слух воспринимаются амплитуды сигналов, а к их fazам ухо оказывается малочувствительным. Таким образом, форма сигнала во временной области плохо описывает слуховое восприятие, и поэтому она обычно не рассматривается в аудиосистемах.

Можно объяснить нечувствительность уха к фазе, если проанализировать, как звук распространяется в среде. Предположим, вы слушаете голос человека, стоя-



**Рис. 22.2** Чувствительность уха человека к фазе сигнала. Ухо человека нечувствительно к фазе гармонических компонентов сигнала. Например, показанные два сигнала слышатся идентично, потому что амплитуды их компонентов одинаковы, хотя фазы отличаются.

щего в противоположном конце небольшой комнаты. Большая часть звуковой волны, которую вы воспринимаете, является отражением от стен, потолка и пола. Из-за того что характер распространения звука различен для разных частот (отличаются затухание, переотражения и эффекты резонанса), разные частоты достигнут вашего уха различными путями. Это означает, что относительная фаза каждой частоты будет меняться, когда вы передвигаетесь по комнате. Благодаря тому что ухо игнорирует эти изменения фаз, вы воспринимаете голос как неменяющийся, несмотря на изменение вашего положения. С физической точки зрения фаза звукового сигнала становится случайной, когда он распространяется в сложной среде (со множеством преград и переотражений). Можно считать и по-другому: ухо нечувствительно к фазе, потому что она содержит мало полезной информации.

Однако нельзя сказать, что ухо полностью безразлично к фазе. Ведь изменение фаз может полностью перестроить временную последовательность звукового сигнала. Примером является система с линейной частотной модуляцией (см. главу 11), которая превращает (единичный) импульс в сигнал много большей длительности. Хотя эти сигналы отличаются только фазой, ухо способно различать два звука, потому что они имеют разную длительность. Это по большей части занимательный пример, не играющий на практике никакой роли.

Предположим, вы просите скрипача сыграть, скажем, ноту ЛЯ малой октавы. Если этот звук отобразить на осциллографе, он будет выглядеть, как пилообразное колебание, показанное на Рис. 22.3а. Сигнал имеет такой характер, потому что на волос смычка нанесена липкая канифоль. Когда смычок перемещается по струне, она прилипает к смычку, натягивается и рывком освобождается. Это повторяется вновь и вновь, формируя сигнал пилообразной формы.

Рис. 22.3б показывает, как такой звук воспринимается ухом: слышится частота 220 Гц и гармоники 440, 660, 880 Гц и т. д. Если эту же ноту взять на другом музыкальном инструменте, форма сигнала на осциллографе окажется иной, но ухом будут восприниматься все те же 220 Гц плюс гармоники. Так как оба инструмента издают звук с одной и той же основной частотой, они звучат похоже, и говорят, что взята одинаковая высота звука (униссон). Однако относительные амплитуды гармоник различны для двух инструментов, и поэтому они звучат по-разному. Говорят, что инструменты имеют разный тембр.

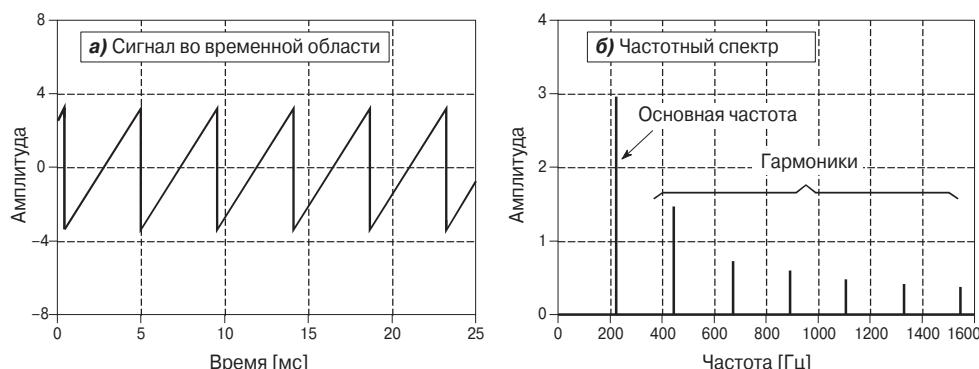
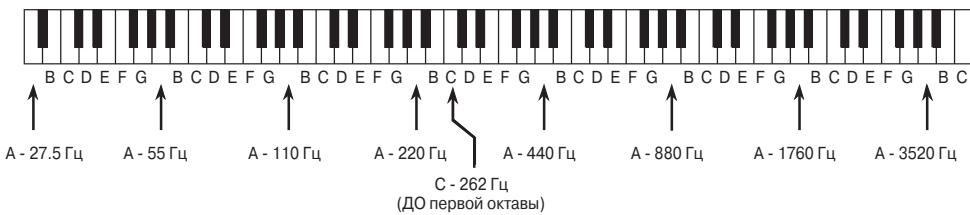


Рис. 22.3 Форма сигнала при звучании скрипки. Смычок скрипки создает пилообразный сигнал, показанный на (а). Звук, воспринимаемый ухом, проиллюстрирован на (б). В его состав входят основная частота и гармоники.

Часто говорят, что тембр определяется формой сигнала. Это верно, но иногда вводит в заблуждение. Восприятие тембра является результатом того, как ухо воспринимает гармоники. Так как состав гармоник определяется формой сигнала, то низкая чувствительность уха к фазе сигнала делает соотношение односторонним: сигнал конкретно заданной формы может иметь лишь один-единственный тембр; в то же время для одного заданного тембра можно подобрать бесконечное число возможных форм сигнала.

Ухо привыкло слышать только основную частоту плюс её гармоники. Звучание комбинации двух синусоидальных волн с частотами 1 и 3 кГц покажется естественным и приятным. В то же время сумма волн с частотами 1 и 3.1 кГц будет «резать слух».

Это является основой общепринятого музыкального звукоряда, проиллюстрированного на Рис. 22.4 в виде клавиатуры фортепиано. Нажатие самой левой клавиши пианино даёт основную частоту 27.5 Гц и гармоники 55, 110, 220, 440, 880 Гц и т. д. (между этими частотами также присутствуют гармоники, но для данных рассуждений они не важны). Эти гармоники могут совпадать с основными частотами других клавиш фортепиано. Говоря конкретно, каждая седьмая белая клавиша имеет основную частоту, соответствующую одной из гармоник самой левой клавиши. То есть 8-я клавиша слева имеет основную частоту 55 Гц, 15-я клавиша — 110 Гц и т. д. Являясь гармониками друг друга, все эти клавиши звучат похоже и при одновременном нажатии дают приятный, гармоничный звук. По этой причине все они названы нотой ЛЯ (А). Аналогично все следующие справа клавиши за клавишами ЛЯ соответствуют ноте СИ (В)<sup>1)</sup> и представляют звуки, являющиеся гармониками друг друга. Всё это повторяется для всех семи нот: ЛЯ (А), СИ (В), ДО (С), РЕ (Д), МИ (Е), ФА (Ф) и СОЛЬ (Г).



**Рис. 22.4.** Клавиатура фортепиано. Для клавиатуры фортепиано характерна логарифмическая частотная шкала с основными частотами, удваивающимися на каждой седьмой белой клавише.

Термин *октава* означает соотношение частот один к двум (различие в 2 раза). На фортепиано одна октава состоит из восьми белых клавиш, что объясняет название термина (по-латыни *octo* — восемь). Другими словами, частота фортепиано удваивается каждые семь белых клавиш, а вся клавиатура включает чуть больше семи октав. Считается, что слух человека охватывает частотный диапазон от 20 Гц до 20 кГц, что соответствует ещё половине октавы слева и двум октавам справа на клавиатуре фортепиано. Так как октавы представляют собой удвоение частоты на фиксированном числе клавиш, то таким образом получается логарифмическая

<sup>1)</sup> В ряде нотаций, особенно русскоязычных, нота СИ обозначается буквой Н, а буква В соответствует ноте СИ-бемоль. — Примеч. пер.

шкала частот. Это важно, потому что звуковая информация обычно распределяется именно таким способом. То есть, например, в частотном диапазоне 50...100 Гц (одна октава) переносится такое же количество информации, что и в диапазоне 10...20 кГц (также одна октава). Хотя фортепиано перекрывает только около 20% частот, которые способен слышать человек (4 кГц из 20), оно способно генерировать более 70% звуковой информации, которую люди могут воспринять (7 из 10 октав). Можно привести здесь другой пример: в течение жизни человека наивысшая частота, которую он способен слышать, уменьшается с 20 до 10 кГц. Однако при этом он теряет лишь около 10% способности воспринимать информацию на слух (одна октава из десяти). Как показано в дальнейшем, это логарифмическое распределение аудиоинформации непосредственно влияет на выбор требуемой частоты дискретизации звуковых сигналов.

## 22.3. Компромисс между качеством звука и частотой дискретизации

При проектировании цифровой аудиосистемы всегда возникают два вопроса: 1) какого качества звучания нужно добиться? и 2) какова может быть допустимая скорость передачи данных? Ответ на эти вопросы обычно можно разделить на три категории. Первая категория — это *аудиосистемы высокого качества (high fidelity music)*, где основное значение имеет высокое качество звука, а скорость данных можно принять любой. Вторая категория относится к *телефонным системам связи*, в которых требуется обеспечить качество звука, достаточное для нормального восприятия речи, и одновременно необходимо сохранить невысокую скорость передачи данных, чтобы сократить стоимость системы. Третья категория — это *системы сжатия речи*, основное требование к которым состоит в максимальном сокращении скорости передачи данных, а некоторое искажение речи является допустимым. Сюда относятся, например, системы военной связи и аппаратура цифровой записи и хранения речи для голосовой почты и мультимедиа-систем.

**Табл. 22.2** отражает компромисс между качеством звука и скоростью данных, характерный для этих трёх категорий систем. Для систем высокого качества характерна частота (скорость) дискретизации (44.1 кГц) и точность оцифровки данных (16 бит), достаточная для воспроизведения фактически всех звуков, которые способен слышать человек. Такое великолепное качество звучания обеспечивается ценой значительного увеличения требуемой скорости передачи и обработки данных:  $44 \text{ кГц} \times 16 \text{ бит} = 706 \text{ Кбит/с}$ . Это, так сказать, решение проблемы «в лоб».

В то время как качественный музыкальный звук требует выделения полосы частот до 20 кГц, для речевых сигналов с приемлемым качеством достаточно всего 3.2 кГц. Несмотря на то что частотный диапазон сократился до 16% (с 20 до 3.2 кГц), сигнал всё ещё содержит 80% первоначальной звуковой информации (8 из 10 октав). Системы связи обычно работают с частотой дискретизации 8 кГц, что обеспечивает хорошее качество речи, но не подходит для воспроизведения музыки. Вы, вероятно, уже знакомы с этой разницей в качестве звучания. Радиостанции FM-диапазона занимают полосу частот около 20 кГц, в то время как радиостанции со средней длиной волны ограничены полосой примерно 3.2 кГц. Голос звучит на этих радиостанциях нормально, а качество музыки оказывается низким.

Таблица 22.2. Компромисс между скоростью данных и качеством звука

Требуемое качество звука	Полоса частот [кГц]	Частота выборки [кГц]	Число бит	Скорость данных [Кбит/с]	Примечания
Аудио высокого качества (компакт-диски)	0.05...20	44.1	16	706	Удовлетворяет самую взыскательную аудиторию. Превышает возможности человеческого слуха
Телефонное качество речи (с компандированием)	0.2...3.2	8	12	96	Хорошее качество речи, но низкое качество музыки.
	0.2...3.2	8	8	64	Нелинейное аналого-цифровое преобразование уменьшает скорость данных на 50%. Это широко известный метод
Декодирование речи с использованием линейного предсказания	0.2...3.2	8	12	4	Цифровой метод сжатия речи. Очень низкая скорость данных. Плохое качество речи

**Примечание.** Качество звучания оцифрованного аудиосигнала зависит от скорости данных, т. е. произведения скорости выборки (частоты дискретизации) на число бит, которым представляется каждое число. Можно выделить три категории систем: аудиосистемы высокого качества (706 Кбит/с), речь с качеством телефонного разговора (64 Кбит/с) и сжатая речь (4 Кбит/с).

Кроме того, для систем, работающих только с речевыми сигналами, допускается уменьшение точности представления данных с 16 до 12 бит на выборку. Если шаг квантования по уровню выбирать неравномерным, то разрядность данных может даже быть уменьшена до 8 бит на выборку. Эта широко распространённая процедура называется *компандированием* и будет рассматриваться далее в этой главе. При скорости выборки 8 кГц и точности аналого-цифрового преобразования 8 бит на выборку скорость данных оказывается равна 64 Кбит/с. Обратите внимание, что речь требует менее 10% скорости данных по сравнению с высококачественными аудиосистемами.

Выбор скорости данных 64 Кбит/с представляет собой непосредственное применение теории дискретизации и квантования к звуковым сигналам. Методы дальнейшего снижения скорости данных связаны со сжатием потока данных за счёт удаления избыточной информации, содержащейся в речевом сигнале. Сжатие данных — это тема Главы 27. Одним из наиболее эффективных путей сжатия звукового сигнала является *линейное предсказание*, встречающееся в нескольких разновидностях. В зависимости от требуемого качества речи, линейное предсказание может обеспечить уменьшение скорости данных до 2...6 Кбит/с. Мы ещё вернемся к линейному предсказанию в этой главе, когда будем рассматривать синтез речи.

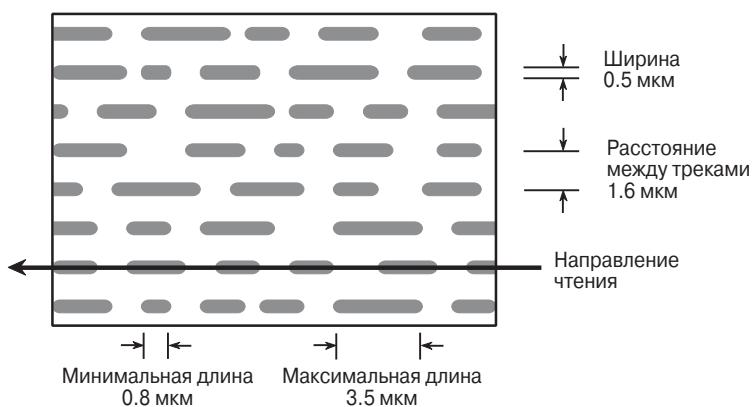
## 22.4. АУДИО ВЫСОКОГО КАЧЕСТВА

От записей музыкальных произведений всегда требуется высочайшее качество звучания, и все другие факторы рассматриваются как второстепенные. Здесь лучше перестараться, чем получить неудовлетворительный результат. Системы аудио высокого качества конструируются не так, чтобы точно удовлетворять возможностям слуха человека, а скорее так, чтобы превысить эти возможности (с запасом). Это единственный способ гарантировать, что звук точно будет воспроиз-

водится без искажений. Цифровое аудио вошло в наш мир с появлением лазерных компакт-дисков, или CD. Это была революция в музыке: качество звука CD-систем намного превысило возможности старых систем, использовавших пластинки или магнитные ленты. Цифровая обработка сигналов была «на переднем фронте» этой технологии.

**Рис. 22.5** иллюстрирует поверхность лазерного компакт-диска такой, как она выглядит под мощным микроскопом. Рабочая поверхность блестящая (отражает свет), со множеством тёмных пятнышек, выжженных лазерным лучом и представляющих записанную цифровую информацию. Данные записываются дорожкой (треком), представляющей собой спираль, идущую от центра диска к наружной стороне (в отличие от грампластинок). Скорость вращения CD меняется от 210 до 480 оборотов в минуту при переходе от наружного витка спирали к внутреннему, что обеспечивает постоянную скорость считывания 1.2 м/с. Для сравнения: грампластинки вращаются с фиксированной скоростью 33, 45 или 78 оборотов в минуту. Во время проигрывания оптический сенсор определяет, отражается в данной точке свет или нет, и генерирует соответствующую двоичную информацию.

Как показано на **Рис. 22.5**, CD способен хранить около 1 бита информации на квадратный микрон поверхности, что соответствует 1 миллиону бит на миллиметр квадратный и 15 миллиардам бит на диск. Применяемый здесь минимальный размер элемента примерно совпадает с используемым в производстве интегральных схем. Одним из свойств света является невозможность его фокусировки до размера меньше, чем половина длины волны, т. е. 0.3 мкм. Так как и интегральные схемы, и лазерные диски создаются оптическими средствами, размытость света при уменьшении ширины луча ниже 0.3 мкм делает нежелательным использование меньшей площади одного элемента.

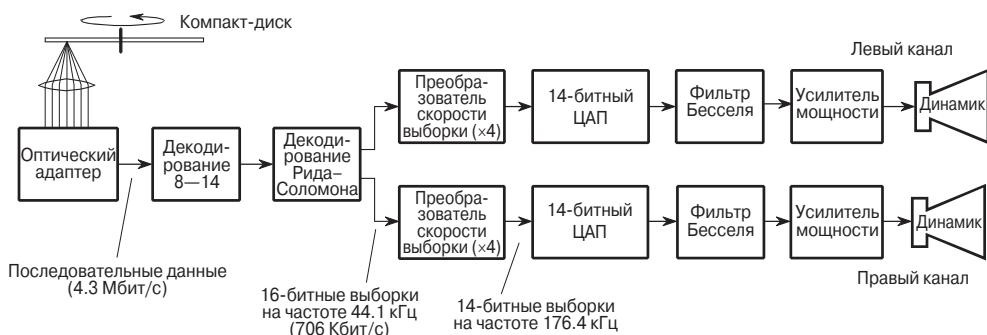


**Рис. 22.5** Поверхность компакт-диска. На поверхности CD выжигаются ямки размером в микрон, которые представляют единицы и нули. В результате плотность данных оказывается 1 бит на  $\text{мкм}^2$  или 1 миллион бит на  $\text{мм}^2$ . Глубина ямки составляет 0.16 мкм.

На **Рис. 22.6** показана структура типовой системы воспроизведения компакт-дисков. Скорость данных составляет 4.3 миллиона бит/с, что соответствует

одному биту на каждые 0.28 мкм длины дорожки. Однако это противоречит определённой установленной геометрии CD. Каждое выжженное пятно должно быть не короче 0.8 мкм и не длиннее 3.5 мкм. Другими словами, каждая двоичная единица должна быть представлена группой, включающей от 3 до 13 единиц. Достоинством такого подхода является уменьшение ошибок, вызванных несовершенством механизма считывания. Но как формировать двоичные данные при наличии таких особенностей нанесения информации на диск группами нулей и единиц?

Ответ даёт схема кодирования, называемая *модуляция восемь—четырнадцать* (EFM). Вместо того чтобы записывать на диск непосредственно один байт данных, каждые 8 бит проходят через таблицу соответствия (справочную таблицу), которая ставит им в соответствие комбинацию из 14 бит. Эти 14 бит удовлетворяют желаемым требованиям записи и сохраняются на лазерном диске. При воспроизведении двоичные значения, прочитанные с диска, проходят через обратную декодирующую таблицу соответствия, превращающую каждую группу из 14 бит в правильные 8 бит.



**Рис. 22.6.** Схема воспроизведения компакт-дисков. Цифровая информация читается с диска оптическим сенсором, корректируется декодером 8–14 и декодером Рида–Соломона, а затем преобразуется в аналоговый стереосигнал.

В дополнение к EFM данные кодируются в формат, называемый *двухуровневым кодированием Рида–Соломона*. Оно включает объединение левого и правого каналов вместе с данными для обнаружения ошибок и коррекции. Цифровые ошибки, обнаруженные во время проигрывания, корректируются с использованием избыточных данных в схеме кодирования, маскируются интерполяцией соседних выборок или приглушаются установкой значения выборки в нуль. В результате эти схемы кодирования утраивают скорость данных, т. е. 1.4 Мбит/с для стерео аудиосигнала становятся 4.3 Мбит/с, сохранёнными на диске.

После декодирования и коррекции аудиосигналы представляются 16-битными выборками при скорости выборки 44.1 кГц. В простейших системах эти сигналы могут пройти через 16-битный ЦАП, за которым следует узкополосный аналоговый фильтр. Однако это может потребовать применения высококачественной аналоговой электроники, для того чтобы пропустить частоты ниже 20 кГц, в то же время подавляя все частоты выше 22.05 кГц, т. е. 1/2 скорости выборки. Наиболее общим методом является использование *многоскоростного* способа, т. е. преобразование цифровых данных к более высокой скорости выборки перед ЦАП. Обычно используется коэффициент 4, преобразуя 44.1 кГц в 176.4 кГц. Такое преобра-

зование называется *интерполяцией* и может рассматриваться как двухшаговый процесс. Во-первых, три выборки со значением ноль помещаются между первона-чальными выборками, создавая более высокую скорость выборки. В частотной об-ласти это имеет эффект переноса спектра 0...22.05 кГц на участки 22.05...44.1 кГц, 44.1...66.15 кГц и 66.15...88.2 кГц. На втором шаге дополнительный цифровой фильтр используется для того, чтобы удалить вновь добавленные частоты.

Рост скорости выборки делает выборочный интервал меньше, в результате че-го ЦАП будет генерировать более гладкий сигнал. Сигнал ещё содержит частоты 20 Гц...20 кГц, однако частоту Найквиста нужно увеличить в 4 раза. Это означает, что аналоговый фильтр должен пропускать частоты ниже 20 кГц и подавлять час-тоты выше 88.2 кГц. Обычно это делается с помощью трёхполюсного фильтра Бесселя. Почему используют фильтр Бесселя, если ухо нечувствительно к фазе? Здесь лучше перестараться, помните?

Так как имеется в 4 раза больше выборок, число бит на выборку можно умень-шить с 16 до 15 без ухудшения качества звука. Для компенсации ступеней ЦАП нулевого порядка необходима коррекция вида  $\sin(x)/x$ , которая может быть час-тью аналогового или цифрового фильтра.

Систему с более чем одним каналом называют *стереосистемой (или трёхмер-ной)*. Несколько каналов посыпают звук слушателю с разных направлений, обес-печивая более точное воспроизведение музыки. Музыка, проигрываемая через монофоническую систему (с одним каналом), звучит неестественно и неточно. Для сравнения: при хорошей стереорепродукции создаётся впечатление, что му-зыканты находятся в нескольких шагах от слушателя. Так, в 1960-е годы высоко-качественная музыка использовала два канала (левый и правый), в то время как для кино использовались четыре (левый, правый, центральный и пространствен-ный). В ранних стереозаписях (скажем, «Битлз») каждого певца часто можно бы-ло слышать только в одном из каналов. Ситуация прогрессировала в направлении более качественного сведения фонограмм, когда звук из многих микрофонов в студии записи объединялся в двух каналах. Сведение фонограмм — это искусство создания для слушателя эффекта присутствия.

Четырёхканальный звук, используемый в кино, называется *Dolby Stereo (Долби Стерео)*, с домашней версией, называемой *Dolby Surround Pro Logic*. Четыре канала кодируют в стандартные левый и правый каналы, позволяя обычным двухка-нальным стереосистемам воспроизводить музыку. Декодер Dolby используется во время проигрывания для восстановления четырёхканального звука. Левый и пра-вый каналы из динамиков, расположенных на каждой стороне кино- и телеэкра-нов, подобны обычным двухканальным стереосистемам. Динамик для централь-ного канала обычно размещается выше или ниже экрана. Его цель — воспроизводить речь и другие звуки, связанные с изображением, центрируя их на экране, независимо от позиции зрителя/слушателя. Динамики *Surround* размеща-ются слева и справа от слушателя. В большой аудитории могут подключаться бо-льше 20 динамиков. Канал *Surround* содержит частоты среднего диапазона 100 Гц...7 кГц, и вносит задержки на 15...30 мс. Задержка выбирается так, чтобы у слушателя создавалось впечатление, что звук идёт с экрана, а не со стороны. Ког-да человек слышит звук, приходящий спереди, сопровождаемый задержанной копией этого же звука, идущей со стороны, его мозг интерпретирует задержан-ный сигнал как отражение от стен и игнорирует его.

## 22.5. Командирование

Скорость данных имеет огромное значение для систем связи, потому что она во многом определяет их стоимость. Экономия битов является экономией денег. *Командирование* — это общераспространённый метод сокращения скорости данных звукового сигнала за счёт неравномерного квантования сигнала по уровню. Как отмечалось ранее, самый громкий звук, воспринимаемый человеком, имеет интенсивность 120 дБ УЗД, в миллион раз превышающую интенсивность самого слабого звука — 0 дБ УЗД. Однако ухо не может различать звуки, отличающиеся по амплитуде менее чем на 1 дБ (12% по амплитуде). Другими словами, можно различить только около 120 различных уровней громкости, распределённых логарифмически по амплитудному диапазону в один миллион.

Это важно для оцифровки звуковых сигналов. Если уровни квантования распределены равномерно, то для получения качественной телефонной речи необходимо использовать 12 бит. Однако достаточным может быть только 8 бит, если квантование по уровню осуществить неравномерно, используя указанную особенность человеческого слуха. Интуитивно ясно: если сигнал мал, уровни должны быть расположены близко друг к другу; если сигнал большой, можно использовать большее разнесение.

Командирование можно выполнить тремя способами:

- пропустить аналоговый сигнал через нелинейную цепь до линейного 8-битного АЦП;
- использовать 8-битный АЦП, имеющий неравномерные ступени;
- использовать линейный 12-битный АЦП, за которым следует цифровая кодирующая таблица (12-битный вход — 8-битный выход).
- Каждый из этих трёх вариантов требует использования одинаковой нелинейной процедуры, но выполняемой на разных этапах преобразования: в аналоговой цепи, в АЦП или в цифровой схеме.

Для построения кривой командирования используют два очень близких стандарта: *закон μ255*, или *μ-закон*, используемый в Северной Америке, и *A-закон*, используемый в Европе. Оба используют нелинейную функцию логарифма, так как она хорошо переводит нелинейное распределение уровней интенсивности звуков, различаемых человеческим ухом, в линейное распределение. В математической форме кривые, используемые в *μ-законе* и *A-законе*, имеют вид

$$y = \ln(1 + \mu x) / \ln(1 + \mu), \text{ при } 0 \leq x \leq 1. \quad (22.1)$$

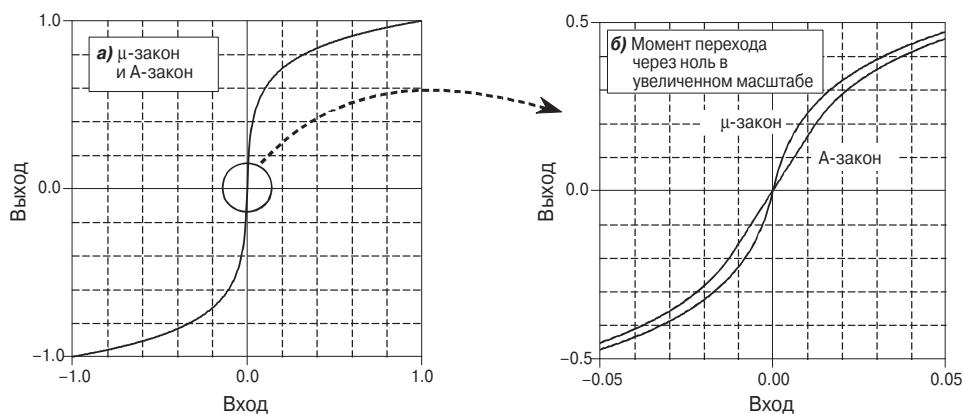
*μ-закон* командирования. Это уравнение описывает нелинейную зависимость, лежащую в основе компандирования по *μ-закону*. Константа  $\mu$  имеет значение 255, благодаря чему этот закон также называют законом *μ255*.

$$\begin{aligned} y &= (1 + \ln(Ax)) / (1 + \ln(A)), \text{ при } 1/A \leq x \leq 1; \\ y &= Ax / (1 + \ln(A)), \text{ при } 0 \leq x \leq 1/A. \end{aligned} \quad (22.2)$$

*A-закон* командирования. Константа  $A$  имеет значение 87.6.

**Рис. 22.7** показывает графики этих уравнений для входной переменной  $x$ , лежащей между  $-1$  и  $+1$ , предполагая, что результат на выходе также лежит в пределах  $[-1; +1]$ . Выражения (22.1) и (22.2) справедливы только для положительных входных значений; части кривых для отрицательных входных значений

найдены из соображений симметрии. Как показано на (а), кривые для обоих законов практически идентичны. Единственная существенная разница имеется вблизи нуля (б).



**Рис. 22.7.** Кривые компандирования. Кривые  $\mu$ -закона и А-закона компандирования почти идентичны. Компандирование увеличивает амплитуду малых и уменьшает амплитуду больших сигналов.

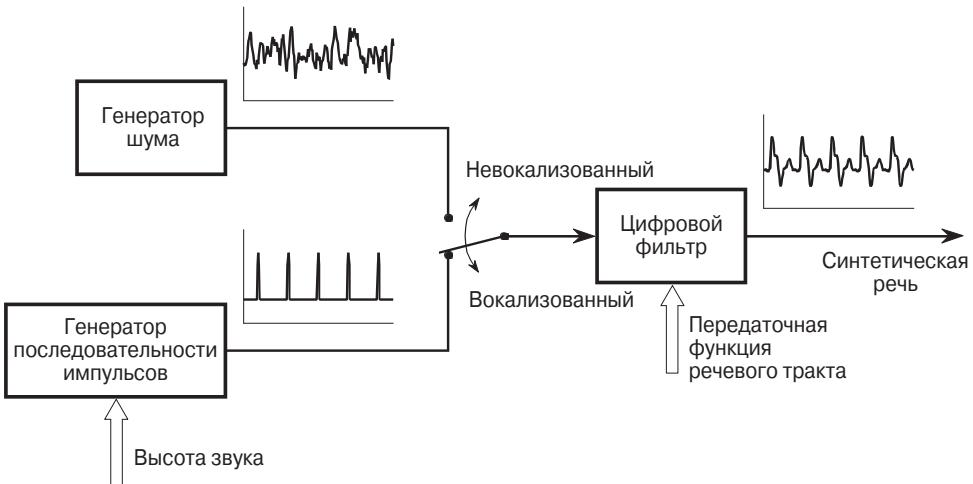
Обеспечение стабильности нелинейной операции — трудная задача для аналоговой электроники. Одним из методов является использование логарифмических соотношений между током и напряжением  $p-n$ -перехода диода, а затем добавление схемы коррекции влияния сильного температурного дрейфа. Большинство схем компандирования использует другую стратегию: аппроксимацию нелинейности с помощью нескольких прямых линий. Типичная схема — это аппроксимация логарифмической кривой с помощью 16 линейных сегментов, называемых *хордами*. Первый бит 8-битного выхода показывает знак сигнала (положительный или отрицательный). Следующие три бита идентифицируют, какие из 8 положительных или 8 отрицательных хорд используются. Последние 4 бита разбивают каждую хорду на 16 равных приращений. Как и большинство интегральных схем, устройства компандирования имеют сложное и оригинальное внутреннее устройство. Не стоит выяснять, как работает то или иное устройство внутри; достаточно изучить цоколёвку микросхемы и спецификацию на неё.

## 22.6. Синтез и распознавание речи

Компьютерное генерирование и распознавание речи — это очень сложная задача, для решения которой было сделано много попыток, но немногие из них оказались успешными. Это область активных исследований в области цифровой обработки сигналов, и она будет оставаться таковой ещё в течение многих лет. Вы будете разочарованы, если ожидаете найти в этой главе описание того, как построить схемы синтеза и распознавания речи. Здесь может быть представлено только краткое описание типовых подходов. Прежде чем начать, необходимо отметить, что большинство коммерческих продуктов, которые воссоздают звучание челове-

ческой речи, не синтезируют ее, а просто проигрывают записанный ранее цифровыми методами сегмент речи диктора. Этот подход имеет высокое качество звука, но он ограничивается набором предварительно записанных слов и выражений.

Почти все методы синтеза и распознавания речи основаны на модели формирования речи человеком, показанной на Рис. 22.8.



**Рис. 22.8.** Модель формирования речи человека. В течение короткого интервала времени (2...40 мс) речь можно моделировать тремя параметрами: 1) выбором периодического или шумового возбуждения; 2) высотой периодического возбуждения и 3) коэффициентами рекурсивного линейного фильтра, имитирующего передаточную функцию, соответствующую речевому тракту.

Большинство звуков человеческой речи можно классифицировать как **вокализованные** или **невокализованные**. Вокализованные звуки появляются тогда, когда воздух, выбрасываемый лёгкими, проходит через голосовые связки и рот и/или нос. Голосовые связки — это две тонкие полоски ткани, натянутые поперёк пути воздушного потока сразу позади «адамова яблока». В соответствии с усилием мышц голосовые связки могут вибрировать с разными частотами — от 50 до 1000 Гц, в результате чего образуются периодические колебания воздуха, вдуванного в область горла. Гласные являются примером вокализованных звуков. На Рис. 22.8 вокализованные звуки представлены генератором последовательности импульсов с регулируемой высотой (т. е. основной частотой сигнала).

Невокализованные (или фрикативные) звуки формируются как случайный шум, а не как периодические вибрации голосовых связок. Такие звуки появляются, когда воздушный поток почти полностью блокируется языком, губами и/или зубами, в результате чего возникает турбулентность в области сужения. Невокализованные звуки соответствуют, например, английским буквам и сочетаниям букв s, f, sh, z, v, th. В модели на Рис. 22.8 эти звуки представлены генератором шума.

Оба описанных источника звука дополняются акустическими полостями, сформированными языком, губами, ртом, горлом и носовым проходом. Так как распространение звука через эти структуры — это линейный процесс, оно может быть представлено как линейный фильтр с соответствующим образом выбранной

импульсной характеристики. В большинстве случаев в модели используется рекурсивный фильтр с коэффициентами рекурсии, определяющими характеристики речевого тракта. Из-за того что акустические полости имеют в размере несколько сантиметров, форма частотной характеристики в большой степени определяется резонансами в диапазоне нескольких килогерц. Эти резонансные частоты называются *формантными частотами*. Изменением относительного положения языка и губ можно изменять формантные частоты по частоте и амплитуде.

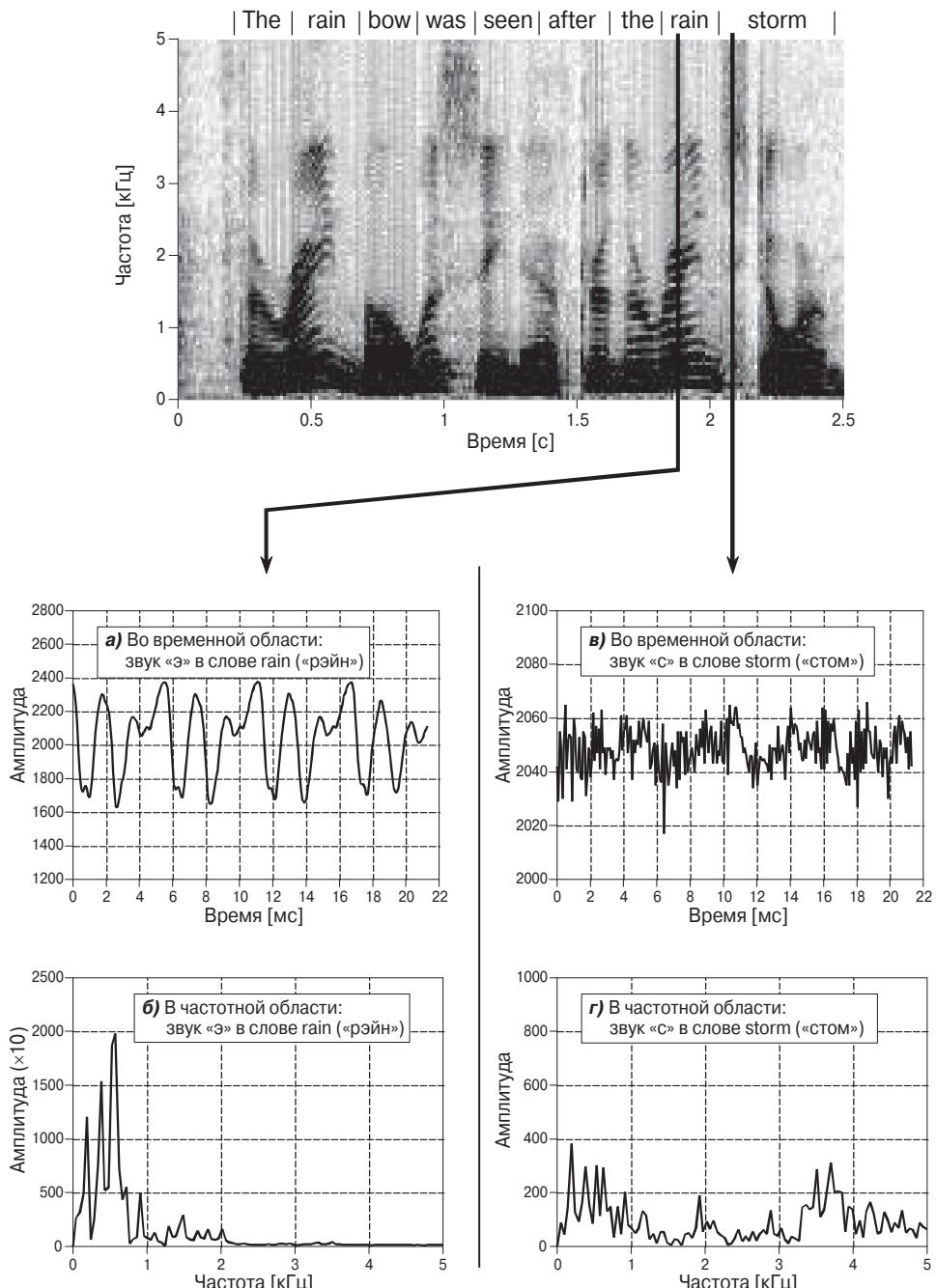
На Рис. 22.9 показан пример речевого сигнала так, как принято иллюстрировать такие сигналы в форме *спектрограммы голоса*. Звуковой сигнал разбивается на короткие сегменты от 2 до 40 мс и используется БПФ, чтобы найти частотный спектр каждого сегмента. Спектры сегментов размещаются друг за другом и отображаются в шкале серого, т. е. малые амплитуды показаны светлыми, а большие — тёмными. Это обеспечивает графический способ наблюдения за тем, как частота речи изменяется во времени. Длина сегмента выбирается в результате поиска компромисса между разрешением по частоте (предпочтительны длинные сегменты) и разрешением по времени (предпочтительны короткие сегменты).

Буква «а» (звук «э») в английском слове rain («рэйн») показывает, что вокализованные звуки имеют периодическую форму во временной области, показанную на (а), и частотный спектр, состоящий из ряда равноотстоящих друг от друга гармоник (б). Для сравнения: звук «с» («с») в английском слове storm («сторм») показывает, что фрикативные звуки имеют во временной области шумоподобную форму (в), и их спектр выглядит так же, как спектр шума (г). Представленные спектры иллюстрируют также наличие формантных частот в обоих сигналах. Заметим, что частотно-временное изображение дважды произносимого слова rain выглядит похоже в обоих случаях.

На коротком интервале длительностью 25 мс речевой сигнал можно аппроксимировать, задавшись тремя параметрами: 1) периодическим или случайным источником возбуждения; 2) частотой периодического сигнала (для вокализованных звуков); 3) коэффициентами цифрового фильтра, используемого для имитации обработки сигнала в речевом тракте. Непрерывная речь может быть синтезирована с помощью периодического изменения этих трёх параметров примерно 40 раз в секунду. Такой подход был применён для одного из первых коммерчески успешных продуктов этого направления — программы электронного обучения детей «Speak & Spell». Качество звука у этого типа синтезаторов речи низкое, речь получается механической и неестественной. Зато достоинством является очень низкая скорость данных, обычно лишь несколько Кбит/с.

Этот метод является также основой для сжатия речи с использованием кодера на основе *линейного предсказания* (ЛПК). Записанная в цифровой форме человеческая речь разбивается на короткие сегменты, и каждый из них характеризуется в соответствии с тремя параметрами модели. Обычно это требует около 12 Б на сегмент, или 2..6 КБ/с. Информация о сегментах передаётся или хранится в памяти, а при необходимости воспроизведения реконструируется синтезатором речи.

Алгоритмы распознавания речи также используют описанный подход, пытаясь узнать известные образы в принятых речевых сигналах по оцениваемым параметрам. Распознавание речи в этом случае включает сравнение сегментов инфор-



**Рис. 22.9** Спектрограмма голоса. Приводится спектрограмма фразы : «*The rainbow was seen after the rain storm*» («После дождя наблюдалась радуга»). На (а и б) показаны временной и частотный сигналы звука «Э» в слове *rain* («рэйн»); на (в и г) показаны временной и частотный сигналы для фрикативного звука «с» в слове *storm* («стом»).

мации с примерами ранее записанных звуков, для того чтобы попытаться идентифицировать сказанные слова. Проблема состоит в том, что этот метод не очень хорошо работает. Он полезен лишь для некоторых применений, но его возможности оказываются значительно ниже возможностей человека. Чтобы понять, почему распознать речь так трудно для компьютера, представьте, что кто-то вдруг произнёс следующую неожиданную фразу:

*Большой бежит медицинский покупать собак к счастью почти когда.*

Конечно, вы не поймете смысла этого предложения, потому что его нет. Важнее, однако, то, что вы не поймёте даже всех слов, которые были сказаны. Так устроено наше восприятие речи. Слова распознаются не только по их звучанию, но и по контексту в предложении и ожиданиям слушателя. Прослушайте, например, два предложения:

*Её свадебный наряд был восхитителен.*

*Здесь невольно обращаешь внимание на ряд высоких стройных кипарисов.*

Даже если звуки, соответствующие подчёркнутым буквам, были произнесены абсолютно одинаково, человек всё равно правильно поймёт сказанные слова, благодаря контексту. Ваш жизненный опыт точно подскажет вам, что у стройных кипарисов не может быть свадебного наряда. Это не осознанное действие, а неотъемлемая часть человеческого слуха.

Большинство алгоритмов распознавания речи работают только со звучанием отдельных слов и не учитывают контекста. Они пытаются распознать слова, но не понять речь. Это указывает на ужасные недостатки по сравнению с человеческим восприятием речи. Три требования являются общими для систем распознавания речи, ограничивающими их возможности.

1. Распознаваемая речь должна иметь отчётливые паузы между словами. Это уменьшает вероятность встречи с фразами, которые похоже звучат, но состоят из разных слов (т.е. «её наряд» и «на ряд»). Для человека это требование неудобно, т. к. заставляет его замедлять речь.
2. Допустимый лексикон часто ограничивается только несколькими сотнями слов из-за того, что алгоритм должен искать похожие слова в ограниченном множестве примеров, находя наилучшее согласование.
3. Алгоритм должен быть «знаком» с говорящим. Это означает, что каждый человек, использующий систему, должен произносить все слова, которые будут распознаваться, повторяя их от 5 до 10 раз. Эта персонализация базы данных увеличивает точность распознавания слов, но она неудобна и требует времени для подготовки.

Награда за разработку успешной технологии распознавания речи громадна. Ведь речь — это самый быстрый и самый эффективный способ человеческого общения. Распознавание речи должно в будущем заменить письмо, печатающие машинки, ввод с клавиатуры, а также электронное управление переключателями и кнопками. Осталось чуть-чуть улучшить современные технологии, чтобы они стали доступны для коммерческого рынка. Прорыв в распознавании речи должен произойти под влиянием разработок в областях искусственного интеллекта и нейронных сетей, а также самой ЦОС. Не стоит думать об этой технологии как о технической проблеме — лучше как о технической возможности.

## 22.7. Нелинейная обработка звуковых сигналов

Цифровая фильтрация может улучшить звуковой сигнал разными способами. Например, винеровскую фильтрацию можно использовать для того, чтобы отдельить частоты полезного сигнала от частот, соответствующих преимущественно шумам (см. Главу 17). Процедура, обратная свёртке (обращение свёртки), используется для компенсации нежелательной свёртки полезного сигнала с импульсной характеристикой искажающей системы, что используется, например, при восстановлении старых аудиозаписей (см. также Главу 17). Эти типы линейных методов составляют основу ЦОС. Однако некоторые нелинейные методы преобразования также полезны для обработки звука. Два из них будут кратко описаны здесь.

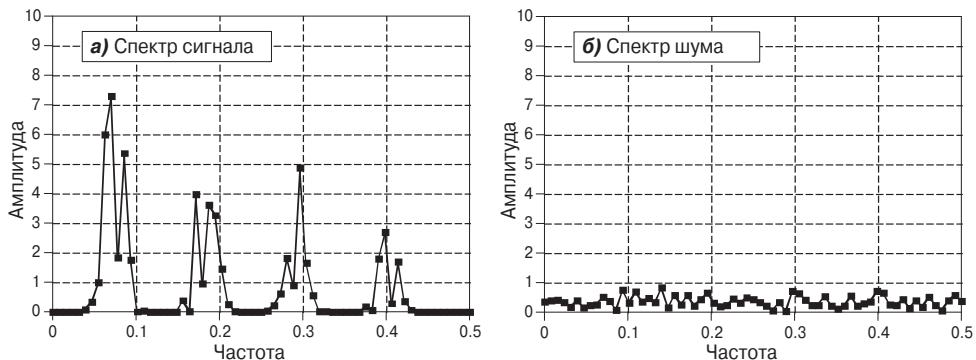
Первый из нелинейных методов используется для уменьшения широкополосного шума в речевом сигнале. Этот тип шума включает: шипение магнитной ленты, электронный шум в аналоговых схемах, ветер, обдувающий микрофоны, шум толпы и т. д. Линейную фильтрацию здесь используют мало, потому что частоты шума полностью перекрывают частоты голосового сигнала в диапазоне 200 Гц...3.2 кГц. Как можно разделить два сигнала, когда они перекрыты и во временной, и в частотной областях?

Покажем, как это сделать нелинейными методами. На коротких сегментах речи амплитуды разных частотных компонентов оказываются существенно различны. Например, на Рис. 22.10*a* показан частотный спектр 16-миллисекундного сегмента речи (т. е. 128 выборок со скоростью выборки 8 кГц). Основная часть сигнала содержится на нескольких частотах с большой амплитудой. Рис. 22.10*b*, напротив, иллюстрирует спектр, в котором представлен только шум. Его форма сильно изрезана, но зато он более равномерно распределён: все амплитуды имеют малое значение.

Теперь ключевая концепция. Если в одной полосе частот наблюдаются и сигнал, и шум, то они могут быть частично разделены путём анализа амплитуды на каждой частоте. Если амплитуда большая — это вероятнее всего сигнал. Он должен быть сохранён. Если амплитуда мала — это скорее всего шум, и его можно отбросить, т. е. приравнять нулю. Частотные компоненты средних амплитуд сводятся к одному из крайних случаев некоторым образом выполняемым сглаживанием.

Другой иллюстрацией использования указанного метода может служить фильтр Винера с переменными коэффициентами. Мы говорили, что частотная характеристика фильтра Винера пропускает частоты, которые преимущественно относятся к сигналу и подавляет частоты, относящиеся к шуму. Это требует знать заранее расположение спектров сигнала и шума, чтобы правильно подобрать частотную характеристику фильтра. Нелинейный метод винеровской фильтрации использует ту же идею, но частотная характеристика фильтра пересчитывается для каждого сегмента в зависимости от его спектра. Другими словами, частотная характеристика фильтра изменяется от сегмента к сегменту, и её форма определяется самим сигналом.

Одной из трудностей использования этого (и других) нелинейного метода является то, что при фильтрации сигналов большой длительности *перекрытие с суммированием* использовать непосредственно не удается. Так как частотная характе-



**Рис. 22.10.** Спектры речи и шума. Хотя спектры речи и шума перекрываются, если сегмент сигнала выбран достаточно коротким, в них можно отметить несколько различий, позволяющих разделить полезный сигнал и шум. На (а) показан спектр 16-миллисекундного сегмента речи, показывая, что многие из частот не несут полезной речевой информации (на этом коротком сегменте). На (б) проиллюстрирован спектр случайного шумового сигнала, все компоненты которого имеют малую амплитуду. Заметим, что приведённые графики не являются реальными, но они позволяют проиллюстрировать принцип разделения сигнала и шума.

ристика меняется, то следующие друг за другом сегменты во временной области на выходе фильтра не будут стыковаться. Можно решить эту проблему, если вспомнить, что звуковая информация закодирована в частотном составе сигнала, который меняется во времени, а не в его форме во временной области. Типичный подход в этом случае состоит в разделении исходного сигнала во временной области на перекрывающиеся сегменты. После обработки к каждому сегменту применяют сглаживание оконной функцией, а затем они комбинируются в выходной сигнал. Таким образом достигается гладкий переход от частотного спектра в одном сегменте к спектру следующего сегмента.

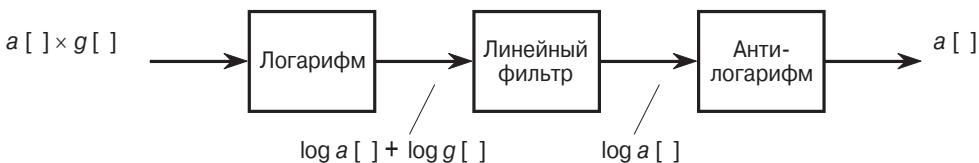
Второй нелинейный метод называется *гомоморфная* обработка сигнала. Этот термин буквально означает «однородная структура». Сложение — не единственный путь, которым шум и помехи могут накладываться на полезный сигнал. Умножение и свёртка также достаточно распространены. Если сигналы объединены нелинейным способом (т. е. не сложением, а любым другим), их нельзя разделить линейной фильтрацией. Гомоморфный метод направлен на попытку разделить сигналы, объединённые нелинейным способом, так, чтобы задача стала линейной. То есть задача преобразуется к однородной структуре — линейному случаю.

Например, рассмотрим радиосигнал, переданный с помощью радиоволны АМ-диапазона. Когда изменяются атмосферные условия, принимаемая амплитуда сигнала возрастает или убывает, из-за чего громкость принимаемого радиосигнала медленно изменяется во времени. Это можно смоделировать как произведение звукового сигнала  $a[ ]$  на медленно изменяющийся сигнал  $g[ ]$ , представляющий коэффициент усиления. Разделение таких сигналов обычно выполняется электронными схемами, называемыми схемами автоматической регулировки усиления (АРУ), но эта задача может быть решена также нелинейной ЦОС.

Как показано на Рис. 22.11, входной сигнал  $a[ ] \times g[ ]$  проходит через логарифмическую функцию. Так как  $\log(xy) = \log(x) + \log(y)$ , два сигнала становятся аддитивными, т. е.  $\log a[ ] + \log g[ ]$ . Другими словами, логарифм является гомоморф-

ным преобразованием, которое преобразует нелинейную задачу умножения в линейную задачу сложения.

Далее аддитивные сигналы разделяются с помощью обычного линейного фильтра. То есть одни частоты пропускаются, а другие подавляются. В случае АРУ сигнал усиления  $g[ ]$  представлен низкими частотами, расположеными значительно ниже полосы голосового сигнала 200 Гц...3.2 кГц. Логарифм этого сигнала будет иметь более сложный спектр, но идея та же: высокочастотный фильтр используется для подавления сигнала, представляющего переменный коэффициент усиления.



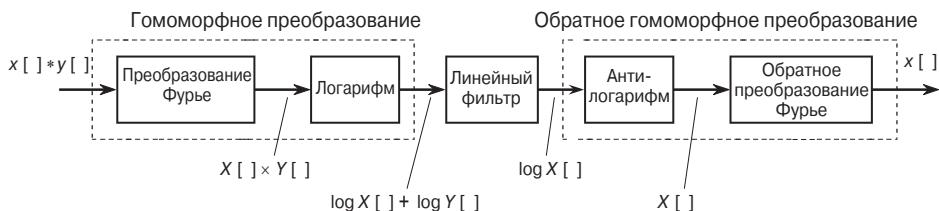
**Рис. 22.11.** Гомоморфное разделение мультиплексированных сигналов. Логарифмирование входного сигнала преобразует мультиплексированные компоненты в компоненты аддитивные. Эти компоненты можно разделить линейным фильтром, а влияние логарифма скомпенсировать.

Вследствие фильтрации  $\log a[ ] + \log g[ ]$  преобразуется в  $\log a[ ]$ . На последнем шаге производится обратное преобразование с помощью экспоненциальной функции  $\exp( )$  (антилогарифмирование), что даёт в результате желаемый выходной сигнал  $a[ ]$ .

На Рис. 22.12 показана гомоморфная система для разделения сигналов, которые объединены свёрткой. Примером задачи разделения таких сигналов может служить задача удаления эха из аудиосигналов. Аудиосигнал с эхом образуется свёрткой исходного сигнала с импульсной характеристикой, представленной суммой дельта-функции и дельта-функции, смещённой и умноженной на постоянный коэффициент. Гомоморфное преобразование для свёртки складывается из двух шагов: преобразования Фурье, превращающего свёртку в произведение, и последующего логарифмирования, превращающего произведение в сумму. Как и прежде, сигналы разделяются линейной фильтрацией и обратным гомоморфным преобразованием.

На Рис. 22.12 проиллюстрирован интересный факт: линейная фильтрация выполняется над сигналами в частотной области, точно так же, как обычно она работает с сигналами во временной области. Другими словами, временная и частотная области «меняются местами» по сравнению с их обычным использованием. Например, если бы для выполнения этапа линейной фильтрации использовалась быстрая свёртка (на основе БПФ), то во временной области оказались бы перемножаемые «спектры». Эта «смена ролей» порождает странную терминологию. Например, кепструм (перестановка слогов в слове спектр — *spectrum*) — это преобразование Фурье от логарифма преобразования Фурье. Подобным образом прижились понятия «долговременный» и «кратковременный» фильтры вместо низкочастотного и высокочастотного. Некоторые авторы используют даже такие термины, как «квиинфренси-анализ» и «лифтация».

Не забывайте, что здесь даётся лишь упрощённое описание сложных алгоритмов ЦОС; на самом деле гомоморфная обработка наполнена множеством тонких нюансов. Например, логарифмирование должно выполняться как над положи-



**Рис. 22.12.** Разделение свёрнутых сигналов с помощью гомоморфной обработки. Компоненты, которые объединены свёрткой, преобразуются в аддитивные компоненты преобразованием Фурье, за которым следует логарифмирование. После линейной фильтрации, которая служит для разделения компонентов, выполняются обратные первоначальные операции.

тельными, так и над отрицательными значениями входного сигнала, — таков аудиосигнал. Это требует использования более сложного описания логарифма, чем то, которое обычно применяется в науке и технике. Когда линейная фильтрация ограничена случаем использования фильтров с нулевой фазой, сложный логарифм рассчитывается как простой логарифм от абсолютного значения сигнала. После прохождения через фильтр с нулевой фазой знак исходного сигнала переприсваивается отфильтрованному сигналу.

Другой проблемой является наложение, которое происходит, когда берётся логарифм. Например, представим оцифровку непрерывной синусоидальной волны. В соответствии с теоремой отсчётов достаточно двух и более выборок на период. Теперь рассмотрим оцифровку логарифма этой непрерывной синусоидальной волны. Острые углы требуют намного больше выборок на период, чтобы правильно отследить форму сигнала, т. е. предотвратить наложения. Требуемая скорость выборки может запросто оказаться в 100 раз больше после логарифмирования, чем до него. Кроме того, не имеет значения, применяется логарифмирование к непрерывному сигналу или к его цифровому представлению, — результат одинаков. Наложения обязательно будут происходить, если скорость выборки (частота дискретизации) выбрана недостаточной для представления острых углов (скачкообразных изменений формы сигнала), появляющихся в результате нелинейного преобразования. Поэтому дискретизация аудиосигнала может потребовать скорости выборки 100 кГц и более вместо стандартных 8 кГц.

Даже если эти детали учтены, нет гарантии, что линеаризованный сигнал может быть разделён линейным фильтром. Это происходит вследствие того, что спектры линеаризованных сигналов могут перекрываться, даже если не перекрывались спектры исходных сигналов. Рассмотрим например, сумму двух синусоидальных волн: одну — с частотой 1 кГц, а вторую — с частотой 2 кГц. Так как эти сигналы не перекрываются в частотной области, они могут быть полностью разделены линейной фильтрацией. Теперь предположим, что эти две волны перемножаются. Используя гомоморфную обработку, возьмём логарифм объединённого сигнала и получим сумму логарифмов этих волн. Проблема заключается в том, что логарифм синусоидальной волны содержит много гармоник. Так как гармоники этих двух сигналов перекрываются, их полное разделение невозможно.

Несмотря на эти трудности, гомоморфная обработка преподает нам урок: сигналы должны обрабатываться в соответствии с тем, как они были сформированы, или, другими словами, первый шаг в любой задаче ЦОС — понять, каким образом информация представлена в обрабатываемом сигнале.

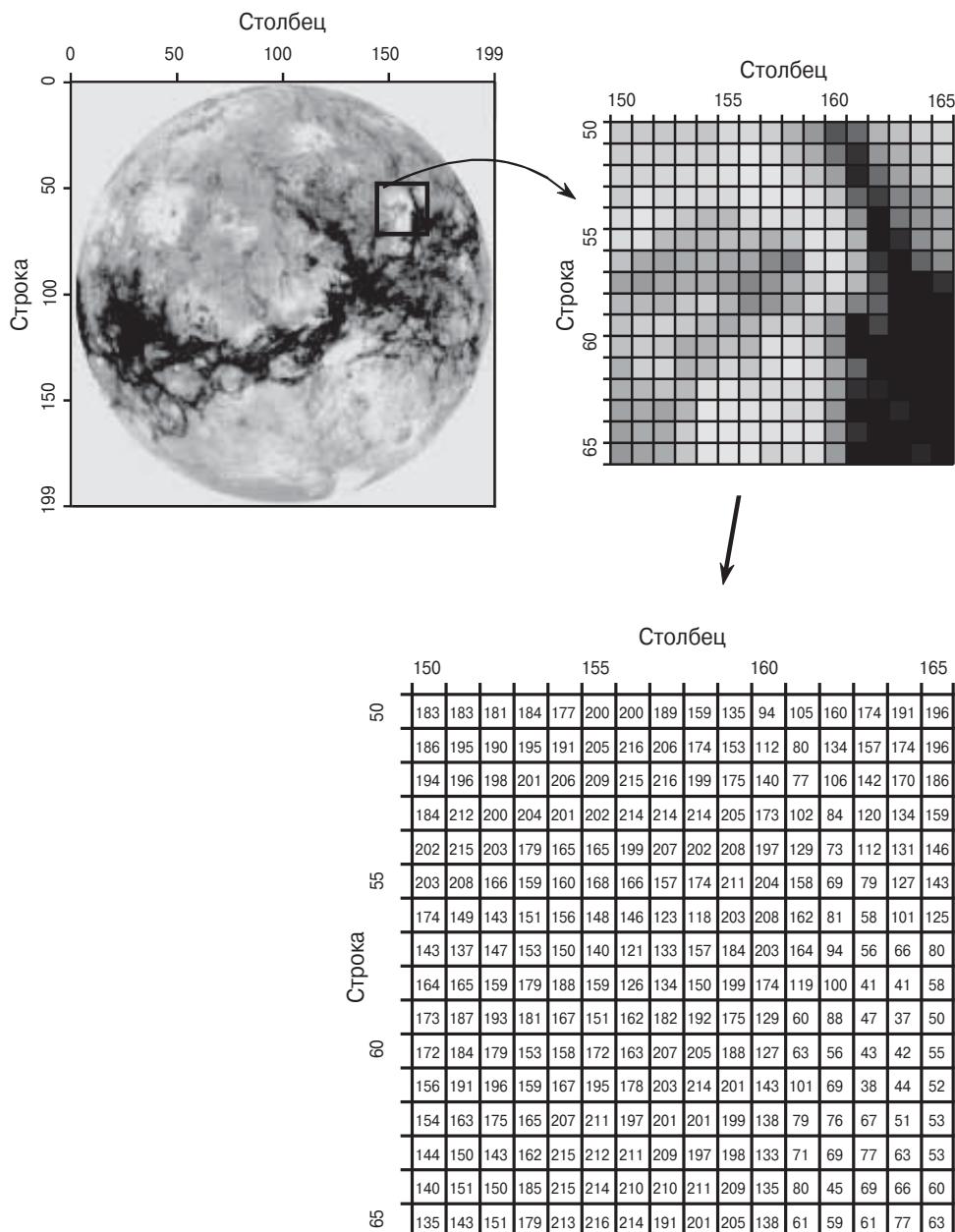
## ФОРМИРОВАНИЕ И ДЕМОНСТРАЦИЯ ИЗОБРАЖЕНИЙ

*Изображение* представляет собой поверхность наблюдения, на которой некоторый параметр меняется по определённому закону. Например, обычные изображения, с которыми мы привыкли иметь дело, описываются зависимостью интенсивности света от координат на двумерной плоскости. Однако интенсивность света — это не единственный параметр, который может использоваться для формирования изображений. Например, изображение может быть сформировано на основе температуры нагрева микросхемы, или скорости тока крови в артериях пациента, или уровня излучения рентгеновских лучей от отдалённой галактики, или амплитудой колебаний земной поверхности во время землетрясений и т. п. Эти «экзотические» изображения обычно преобразуют в удобную для восприятия картину, т. е. световой образ, привычный глазу человека. Эта глава является первой, посвящённой обработке изображений. В ней описывается, как цифровое изображение формируется и как оно преобразуется в подходящую для демонстрации форму.

### 23.1. Структура цифрового изображения

Рис. 23.1 иллюстрирует структуру цифрового изображения. Вы видите планету Венера, снимок которой получен микроволновым радаром с орбитального спутника. Плотные слои атмосферы «блокируют» видимый свет и делают обычную фотографию невозможной. Изображение, сформированное микроволновым радаром, представляет собой 40 000 точек, организованных в виде двумерного массива размером 200 строк на 200 столбцов. Как и в случае одномерного сигнала, элементы в строках и столбцах двумерного массива можно нумеровать с 0-го по 199-й или с 1-го по 200-й. На языке обработки изображений каждая точка полученного массива называется *пикселям* (*pixel* — сокращение от *picture element*). Каждый пиксель в рассматриваемом примере представлен числом в диапазоне 0...255, которые при формировании изображения соответствуют значениям энергии микроволновых зондирующих сигналов, отражённых от определённого участка поверхности планеты. Переход к такому набору чисел необходим для того, чтобы полученное изображение можно было продемонстрировать. Говорят, что значения пикселей преобразуются в *шкалу серого*. Число 0 — это чёрное, 255 — белое, а промежуточные значения — оттенки серого.

Информация закладывается в изображениях в *пространственной области*, являющейся аналогом временной области обычных одномерных сигналов. Различимость мелких деталей изображений определяется не частотами, а размерами. То есть число пикселей, визуально выделяемых в наблюдаемом фрагменте пространства, определяется тем, насколько мала деталь, которую нужно показать, а не формальными ограничениями теоремы отсчётов. Явления наложений



**Рис. 23.1.** Структура цифрового изображения. Это пример изображения планеты Венера, видимого в отражённых микроволнах. Цифровые изображения представляются двумерным массивом чисел, называемых пикселями. Для данного изображения массив состоит из 200 строк и 200 столбцов, а каждый пиксель — это число от 0 до 255. Каждый пиксель изображения соответствует уровню отражённой микроволновой энергии. Для изображений, представляемых в оттенках серого, значения пикселей лежат в диапазоне 0...255.

(*aliasing*) могут происходить и здесь, но они оказываются скорее небольшой не- приятностью, чем серьёзной проблемой. Например, костюм в светлую полоску выглядит в телевизоре ужасно, потому что частота повторения полосок оказывается больше, чем частота Найквиста. Наложение приводит к появлению светлых и тёмных полос, которые движутся по одежде, когда человек меняет положение.

Типичное цифровое изображение состоит примерно из 500 строк и 500 столбцов. Изображения такого качества характерны для телевизоров, мониторов персональных компьютеров и дисплеев исследовательского оборудования различного рода. Изображения с меньшим числом пикселей, скажем 250 на 250, считаются изображениями с плохим разрешением. Эти изображения выглядят совершенно ненатурально; на них часто можно увидеть отдельные пиксели. В свою очередь изображения с разрешением более чем 1000 на 1000 пикселей считаются исключительно хорошими. Такое качество характерно для лучших компьютерных графических систем, высококачественного телевидения и 35-миллиметровых кинолент. Имеются приложения, где нужно ещё более высокое разрешение. Это, например, цифровые рентгеновские снимки; космическая фотография; глянцевая реклама в журналах.

Главная причина использования изображений с низким разрешением состоит в том, что гораздо проще обработать меньшее число пикселей. Это очень важно, ведь одной из основных проблем обработки изображений является необходимость работы с огромным количеством данных. Например, запись одной секунды цифрового звукового сигнала требует около 8 КБ памяти. В то же время одна секунда телевизионного сигнала занимает около 8 МБ. Передача изображения 500 на 500 пикселей через modem со скоростью 33.6 КБ/с требует около минуты! Переход на изображение размером 1000 на 1000 пикселей усложняет задачу в 4 раза.

Общепринятым является использование 256 оттенков серого (уровней квантования) при обработке изображений, что соответствует выделению одного байта на пиксель. Для этого есть несколько причин. Во-первых, один байт удобен для управления данными, потому что именно так компьютер хранит данные. Во-вторых, большое число пикселей в изображении способно в определённой степени компенсировать ограниченное число уровней квантования. Представим, например, группу смежных пикселей со значениями яркости, лежащими между 145 и 146. Глаз человека воспринимает эту область со значением яркости, равным 145.5. Изображение получается сглаженным. В-третьих, что наиболее важно, шаг по яркости, равный 1/256 (0.39%), оказывается меньше минимального перепада яркости, который глаз способен воспринять. Поэтому увеличение числа уровней свыше 256 не приводит к повышению качества изображения.

Однако некоторые изображения всё же нужно представлять более чем восемью битами на пиксель. Ведь большинство изображений, встречающихся в ЦОС, несут информацию о невидимых параметрах. С помощью большего числа уровней квантования можно извлечь больше информации из полученного изображения, выявляя мелкие детали сигнала. В этих случаях не требуется, чтобы глаз человека увидел всю информацию, содержащуюся в изображении со сверхвысоким разрешением. Мы вернёмся к этой проблеме при обсуждении вопросов, касающихся яркости и контрастности.

Значение каждого пикселя в цифровом изображении представляет малую *область* исходного непрерывного изображения. Если, например, космический

зонд, перемещаясь над поверхностью Венеры, делает снимок планеты каждые 10 метров (*шаг выборки*), то каждый пиксель формируемого изображения будет представлять поверхность площадью  $10 \times 10 \text{ м}^2$ . С другой стороны, при каждом снимке происходит излучение и регистрация пачки радиоимпульсов, разбивающих пространство наблюдения на круглые области, скажем, по 15 м в диаметре. Следовательно, каждый пиксель содержит информацию об этой области независимо от шага выборки.

Та область непрерывного изображения, которая определяет значение одного пикселя, называется *апертурой выборки*. Размер апертуры выборки определяется внутренними возможностями системы. Например, для микроскопа ограничением является качество оптики и длина волны света; возможности электронных камер ограничиваются случайной диффузией электронов в датчике изображения и т. д. В большинстве случаев шаг выборки берут приблизительно равным апертуре выборки системы. Разрешение цифрового изображения ограничивается большим из двух факторов: шагом и апертурой выборки. Мы вернёмся к этому вопросу в Главе 25, когда будем обсуждать пространственное разрешение цифровых изображений.

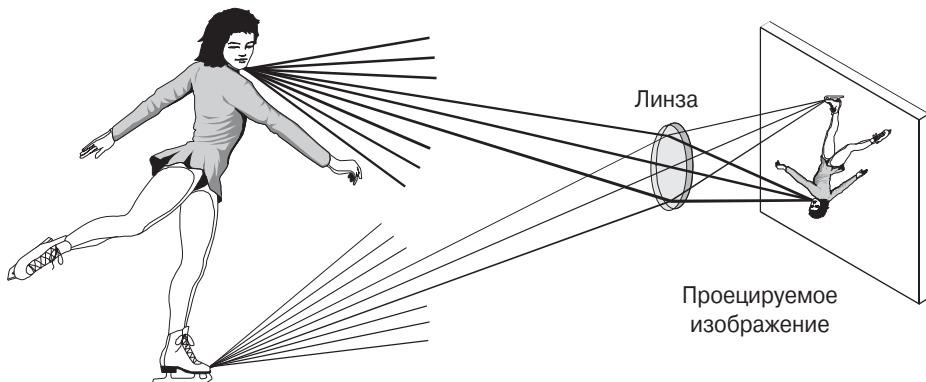
Информация о цвете закладывается в цифровое изображение с помощью трёх чисел на каждый пиксель. Числа несут информацию об интенсивности трёх первичных цветов: красного, зелёного и синего. Смесь этих трёх цветов даёт любые возможные цвета, воспринимаемые глазом человека. Для хранения интенсивности каждого цвета обычно используется один байт, позволяя охватить  $256 \times 256 \times 256 = 16.8$  миллиона различных цветов.

Цвет очень важен, когда цель состоит в том, чтобы представить наблюдателю истинную визуальную картину мира, как, например, на телевизионном экране или на фотографии. Однако часто в задачах науки и техники целью является анализ двумерного сигнала, использующий зрительную систему человека лишь в качестве инструмента. Здесь часто оказывается достаточно чёрно-белых изображений.

## 23.2. Фотоаппарат и глаз человека

Структура и работа глаза человека и электронного фотоаппарата очень похожи. Попробуем их сравнить. И та, и другая системы основаны на двух главных компонентах — наборе линз и датчике изображения. Набор линз «захватывает» порцию света, излучаемую объектом, и фокусирует её на датчике изображения. Датчик изображения преобразует световой образ в видеосигнал, электрический ток или нервный импульс.

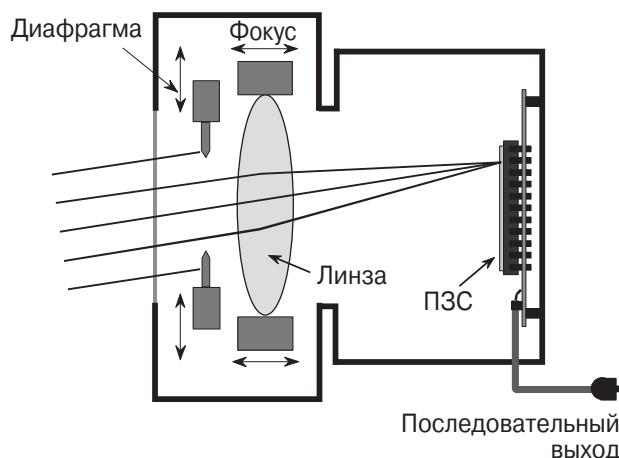
**Рис. 23.2** иллюстрирует работу линз. В этом примере на экране фокусируется изображение фигуристки. Фокусировка означает обеспечение того, что каждой точке наблюдаемого объекта (фигуристки) в точности соответствует точка на экране. Рассмотрим, например, область на объекте размером  $1 \times 1 \text{ мм}$ . При ярком свете 100 миллиардов фотонов ударяют в этот один квадратный миллиметр поверхности каждую секунду. В зависимости от характеристик поверхности от 1 до 99 процентов этих фотонов света отражаются в случайных направлениях. Только малая часть отражённых фотонов проходит через систему линз. Так, через линзу диаметром один сантиметр, расположенную на расстоянии 3 м от объекта, пройдёт только около одной миллионной доли всего отражённого света.



**Рис. 23.2.** Фокусировка с помощью линзы. Линза собирает свет от точечного источника и усиливает его, чтобы превратить в соответствующую точку на другом объекте. Это позволяет линзе проецировать изображения на различные поверхности.

Рефракция в линзе изменяет направление отдельных фотонов в зависимости от места и угла, под которым они падают на поверхность раздела стекло/воздух. Одна точка на исходном объекте переводится линзой в одну точку проекции на экране. Все фотоны, которые отражены от носка ботинка фигуристки, проходят через линзу и переводятся в «носок» на проецируемом изображении. То же характерно и для любой другой точки изображения.

**Рис. 23.3** и **23.4** показывают основные компоненты электронного фотоаппарата и глаза человека соответственно. Обе системы являются светонепроницаемыми и содержат линзу на одном конце и датчик изображения — на другом. Фотоаппарат заполнен воздухом, а глаз — прозрачной жидкостью. Каждая линза имеет два регулируемых параметра — *фокусное расстояние* и *размер диафрагмы*.



**Рис. 23.3.** Структура электронного фотоаппарата. Фокусировка достигается перемещением линзы ближе или дальше от датчика изображения. То количество света, которое достигает датчика, управляет диафрагмой — механическим устройством, которое изменяет эффективный диаметр линзы. Наиболее известный на сегодняшний день датчик изображения — это прибор с зарядовой связью (ПЗС) — двумерный массив светочувствительных элементов.



**Рис. 23.4.** Структура глаза человека. Глаз — это заполненная жидкостью сфера около 3 см в диаметре, заключенная в плотную внешнюю оболочку, называемую склерой. Фокусировка обеспечивается роговицей — фиксированной линзой в передней части глаза. Фокус регулируется сокращением мышц, прикрепленных к гибкой линзе внутри глаза. Свет, попадающий в глаз, регулируется диафрагмой, сформированной из непрозрачной мышечной ткани, закрывающей часть линзы. Задняя полусфера глаза содержит сетчатую оболочку — слой светочувствительных нервных клеток, которые преобразуют изображение в сигнал в зрительном нерве.

Если линза плохо сфокусирована, то каждая точка объекта будет проецироваться в круглую размытую область на датчике изображения, из-за чего изображение будет расплывчатым. В фотокамере фокусировка достигается механическим движением линзы в сторону датчика изображений или от него. Глаз содержит две линзы: выпуклую — впереди глазного яблока, называемую роговицей, и регулируемую линзу (хрусталик) — внутри глаза. Роговица выполняет наибольшую часть рефракции света, но она имеет постоянную форму и положение. Фокусировка выполняется внутренней линзой, гибкая структура которой может деформироваться под действием цилиарной мышцы.

В этих двух системах диафрагма используется для управления размером площади линзы, доступной для света, и, следовательно, яркостью изображения, проецируемого на датчик изображений. Диафрагма глаза — светонепроницаемая мышечная ткань, которая может сокращаться, чтобы сделать зрачок больше. Диафрагма фотокамеры — механическое устройство, выполняющее ту же функцию.

Параметры оптических систем достаточно сильно взаимосвязаны. Рассмотрим, например, как количество пропускаемого света и чувствительность датчика изображений влияют на резкость полученного изображения. Диаметр диафрагмы и время экспонирования регулируются для того, чтобы пропустить необходимое количество света от рассматриваемой сцены до датчика изображений. Если количество света превышает достаточную величину, диаметр диафрагмы может быть уменьшен, что увеличивает глубину резкости — диапазон расстояний от фотокамеры, на котором объект остаётся в фокусе. Кроме того, избыток света позволяет сократить время экспозиции, что уменьшает расплывчатость из-за дрожания фотокамеры и перемещения объекта. Оптические системы полны такого рода компромиссов.

Регулируемая диафрагма необходима как для фотоаппарата, так и для глаза человека, потому что интенсивность света в окружающей среде намного больше, чем может непосредственно воспринимать датчик изображения. Например, солнечный свет примерно в миллион раз интенсивнее лунного света. Учитывая, что объекты способны отражать от 1 до 99% этого света, диапазон интенсивностей, который хотелось бы перекрывать, составляет почти сто миллионов раз.

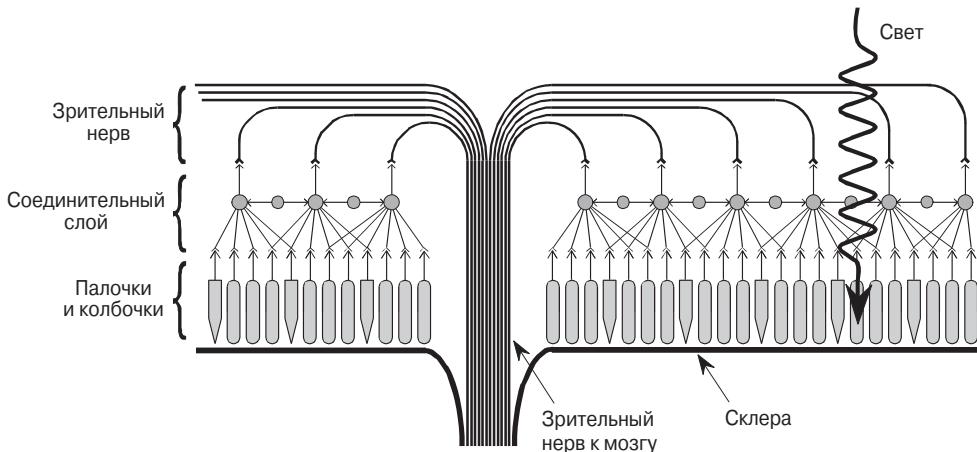
Динамический диапазон электронной фотокамеры обычно составляет от 300 до 1000 и определяется как отношение самого большого уровня сигнала, который можно измерить, к уровню внутренних шумов устройства. Пусть, например, максимальный регистрируемый сигнал имеет уровень 1 В, а среднеквадратическое значение внутреннего шума, измеряемого в полной темноте, составляет 1 мВ. Обычно фотокамеры имеют диафрагму, которая может изменять площадь пропускания света в 300 раз. Это означает, что типичная электронная фотокамера имеет динамический диапазон в несколько сот тысяч. Очевидно, что один и тот же фотоаппарат не может быть достаточно хорош одновременно и при ярком солнечном свете, и тёмной ночью.

А вот глаз человека имеет динамический диапазон, почти полностью перекрывающий диапазон изменения внешних условий. Интересно, что при этом регулировка диафрагмы не является основным инструментом достижения такого гигантского динамического диапазона. От темноты к свету площадь зрачка изменяется только примерно в 20 раз. А дело в том, что нервные клетки, воспринимающие свет, способны изменять свою чувствительность, дополнительно увеличивая динамический диапазон. Для наших глаз достаточно нескольких минут, чтобы приспособиться к плохому освещению, скажем, сцены в зале кинотеатра после прогулки по улице в яркий солнечный день.

Одним из способов улучшения качества изображения методами ЦОС является сужение динамического диапазона, требуемого для просмотра. Нет необходимости на одном и том же изображении показывать и очень яркие, и очень тёмные области. Изображение формируется на основе двух факторов: двумерного освещавшего сигнала и двумерной отражающей способности освещаемого объекта. Отражающая способность имеет динамический диапазон не более 100, потому что все обычные материалы переотражают от 1 до 99% падающего света. Именно в отражающей способности содержится наибольшее количество информации о том, где находятся объекты на наблюдаемой поверхности и каковы их характеристики. В свою очередь освещавший сигнал зависит только от источников света вокруг объекта, но не от самих интересующих нас объектов. Он может иметь динамический диапазон, измеряемый миллионами (хотя обычно для одного изображения характерны величины от 10 до 100). Освещавший сигнал несёт мало информации, но может существенно ухудшить изображение, завышая его динамический диапазон. ЦОС может улучшить ситуацию путём подавления освещавшего сигнала, позволяя параметрам отражения доминировать на изображении. Об использовании этой методики мы поговорим в следующей главе.

Светочувствительная поверхность, входящая в состав зрительной системы человека, называется сетчатой оболочкой или сетчаткой. Как показано на Рис. 23.5, сетчатка может быть разделена на три главных слоя специализированных нервных клеток: один — для преобразования света в нервный сигнал, второй — для обработки изображения, третий — для передачи информации к зрительному н-

рву, ведущему в мозг. Почти у всех животных эти слои как бы перевёрнуты: светочувствительные клетки находятся в последнем слое, чтобы свет прошёл через другие слои прежде, чем он будет обнаружен.

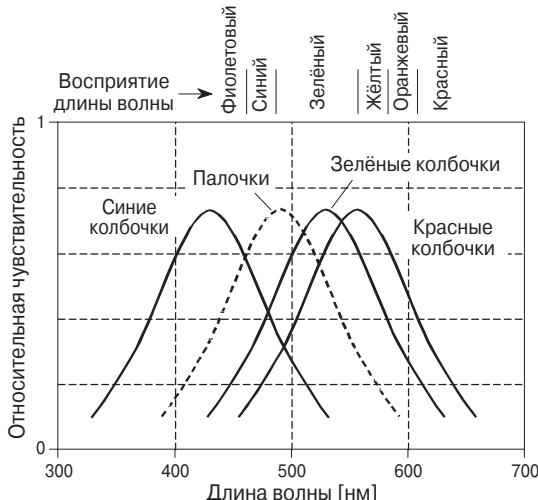


**Рис. 23.5.** Сетчатая оболочка глаза человека. Сетчатая оболочка содержит три слоя: 1) световые рецепторы — палочки и колбочки; 2) промежуточный слой для уменьшения количества данных и обработки изображения и 3) волокна зрительного нерва, которые ведут в мозг.

Существуют два типа клеток, которые ощущают свет, — *палочки и колбочки*, названные так по их форме, видимой под микроскопом. Палочки специализируются в работе с очень слабым светом, как, например, в ночное время на природе. Глаз в темноте воспринимает изображение в очень искажённом виде, оно как будто заполнено непрерывно изменяющимся зернистым узором. В результате сигнал изображения будет очень слабым, что не является недостатком зрения. В глаз проникает так мало света, что может быть видно случайное проявление отдельных фотонов. *Статистический шум* встречается на всех слабо освещённых изображениях, таких, как системы ночного видения военного назначения. В Главе 25 мы снова вернёмся к этому вопросу. Так как палочки не реагируют на цвет, то при плохом освещении зрение становится чёрно-белым.

Рецепторы-колбочки отвечают за различение цветов, но они работают только при достаточном количестве света. Имеется три типа колбочек: чувствительные к красному, зелёному и синему цвету. Выделяемый цвет зависит от содержащихся в колбочках фотопигментов — химических веществ, которые поглощают свет с разной длиной волны. **Рис. 23.6** показывает длины волн света, которые запускают каждый из трёх рецепторов. Описанный подход называется *RGB-кодированием* и соответствует принципу передачи цветовой информации из глаза человека через зрительный нерв в мозг. Человек воспринимает цвет, производя более сложную обработку на уровне нейронов мозга. Здесь RGB-кодирование преобразуется в другую схему кодирования, в которой цвета классифицируются как красный или зелёный, синий или жёлтый и светлый или тёмный.

RGB-кодирование является важным ограничивающим фактором зрительной системы человека. Все длины волн, которые существуют в окружающей среде, сводятся только к трём весьма широким категориям. Для сравнения: специаль-



**Рис. 23.6.** Спектральная характеристика глаза. Три типа колбочек в глазу человека соответствуют областям видимого спектра — красному, зелёному и синему. Комбинация этих трёх цветов формирует все цвета, которые человек может воспринять. Колбочки не имеют достаточной чувствительности, чтобы их можно было использовать в слабо освещённой среде, где для восприятия изображения используются палочки. Вот почему цвет плохо воспринимается ночью.

ные фотокамеры могут различать в оптическом спектре сотни и тысячи отдельных цветов. Это может использоваться, например, для определения клеток как раковых или здоровых; анализа физических процессов, происходящих на поверхности удалённых звезд; обнаружения замаскированных под покровом леса воинских подразделений и т. д. Почему глаз так ограничен в восприятии цвета? По-видимому, всем людям для выживания вполне достаточно находить *красное яблоко* среди *зелёных листьев* на фоне *синего неба*.

Палочки и колбочки имеют размер около 3 мкм и плотно упакованы внутри поверхности сетчатой оболочки размером 3×3 см. В результате сетчатая оболочка создаёт массив «данных» примерно  $10\,000 \times 10\,000 = 100$  миллионов точек-рецепторов. В то же время зрительный нерв имеет только около одного миллиона нервных волокон, которые подсоединены к сетчатке. В среднем каждое волокно зрительного нерва подсоединенено через соединительный слой примерно к 100 светорецепторам. Соединительный слой улучшает качество изображения, подчёркивая границы и подавляя компоненты освещдающего сигнала. Эта биологическая обработка изображения будет обсуждена в следующей главе.

Непосредственно в центре сетчатой оболочки находится область, называемая *фовеа* (на латыни «ямка» или «лунка»), которая используется для зрения с высоким разрешением (см. **Рис. 23.4**). Эта область отличается от остальной сетчатой оболочки. Во-первых, зрительный нерв и внутренний соединительный слой здесь сильнее прижаты к внешнему слою, позволяя рецепторам быть более подверженными действию падающего света. Вследствие этого фовеа оказывается немножко опущена в сетчатой оболочке. Во-вторых, в ней располагаются только колбочки, и они более тесно упакованы, чем в остальной части сетчатой оболочки. Отсутствием палочек, кстати, объясняется, почему зрение часто оказывается лучше, ес-

ли смотреть в сторону от объекта, а не прямо на него. В-третьих, каждое волокно зрительного нерва находится под влиянием нескольких колбочек, обеспечивая хорошую локализационную способность. Фовеа необычайно мала. На расстоянии, на котором принято читать книгу, она видит только область диаметром 1 мм, т. е. меньше одной буквы! Разрешение внутри этой области в цифровом эквиваленте составляет  $20 \times 20$  пикселей.

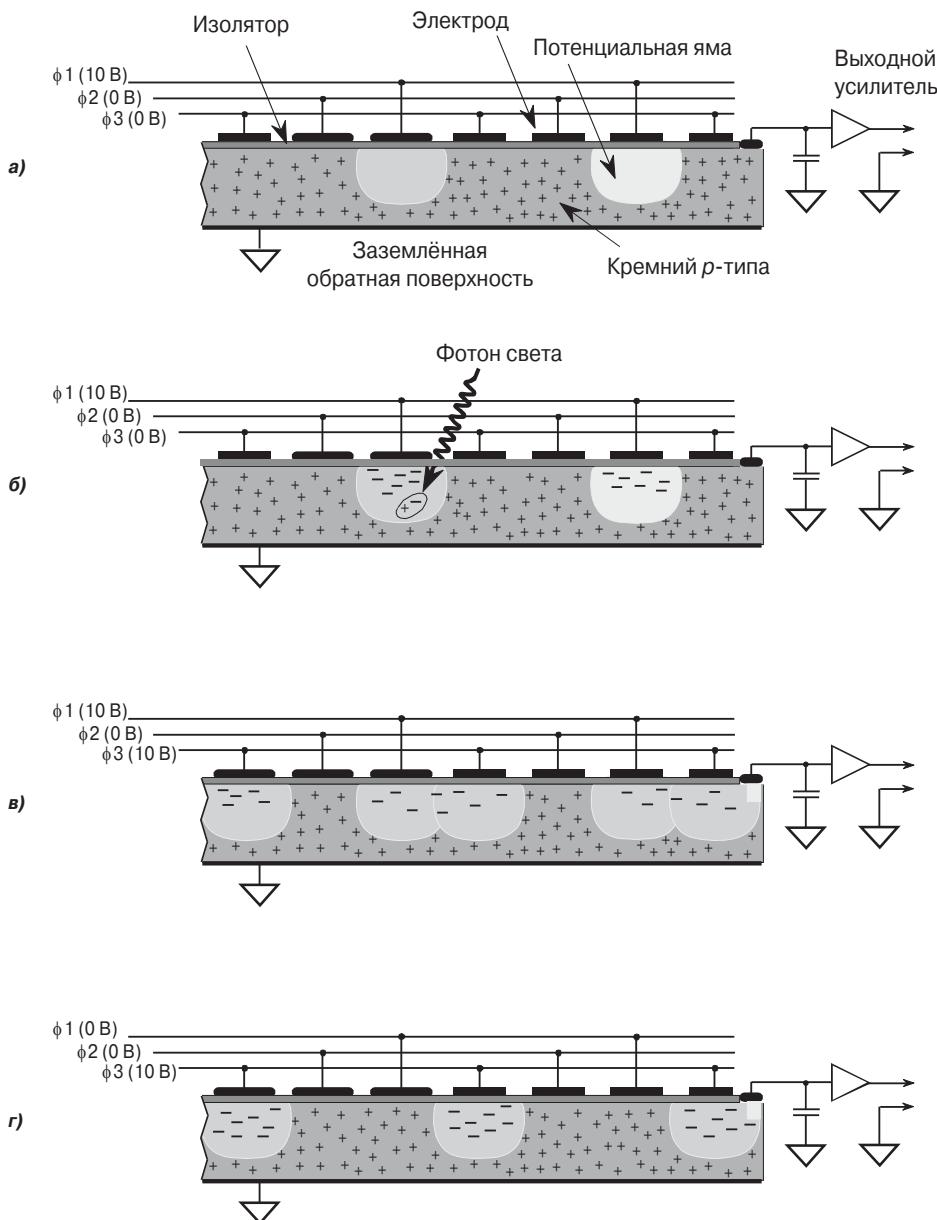
Зрение человека преодолевает проблему малого размера области фовеа за счёт скачкообразных перемещений глаз. Эти резкие движения позволяют макуле с высоким разрешением быстро сканировать всё поле зрения, где находится нужная информация. Кроме того, при этих движениях палочкам и колбочек передаётся непрерывно изменяющийся рисунок света. Здесь проявляется важная особенность нервов сетчатой оболочки — способность адаптироваться к изменению уровней интенсивности света. Действительно, если глаз заставить смотреть в одном направлении, через несколько секунд детали и цвета начнут расплываться.

Наиболее часто используемым в электронных фотокамерах датчиком изображений является прибор с зарядовой связью (ПЗС). ПЗС — это микросхемы, которые в 1980-е годы заменили ламповые камеры точно так же, как за 20 лет до этого транзисторы заменили ламповые усилители. Сердцем ПЗС является тонкая подложка из кремния площадью обычно около 1 см<sup>2</sup>. Как показано на Рис. 23.7, обратная её сторона покрыта тонким слоем металла, соединённого с потенциалом земли. На верхней стороне расположен тонкий слой изолятора и равномерный ряд электродов. Наиболее распространённый тип ПЗС — это прибор с трёхфазным считыванием, где каждый третий электрод соединяется в одну цепь. Используется кремний так называемого *p*-типа проводимости, означающий, что он имеет избыток положительно заряженных носителей, называемых дырками. В нашем случае дырка может рассматриваться как положительно заряженная частица, которая свободно перемещается в кремнии. Дырки обозначены на рисунке символом «+».

В примере, проиллюстрированном на Рис. 23.7а на одну из трёх фаз подано напряжение +10 В, а на две другие — 0 В. Вследствие этого дырки перемещаются от каждого третьего электрода, так как положительные заряды отталкиваются положительным напряжением. Так формируется область под каждым электродом, называемая потенциальной ямой.

Каждая потенциальная яма в ПЗС — это очень эффективный датчик света. Как показано на (б), один фотон света, попадая в кремний, генерирует две заряженные частицы: электрон и дырку. Дырка удаляется, а электрон остаётся в потенциальной яме и удерживается положительным зарядом на электроде. Электроны на этой иллюстрации обозначены символом «—». В течение *периода накопления* свет, падающий на ПЗС, преобразуется в заряды в потенциальных ямах ПЗС, т. е. формируется электронное изображение. Более слабый источник света требует большего периода накопления. Например, период накопления для стандартного телевизора равен 1/60 секунды, а для космической фотосъёмки может потребоваться накопление света в течение нескольких часов.

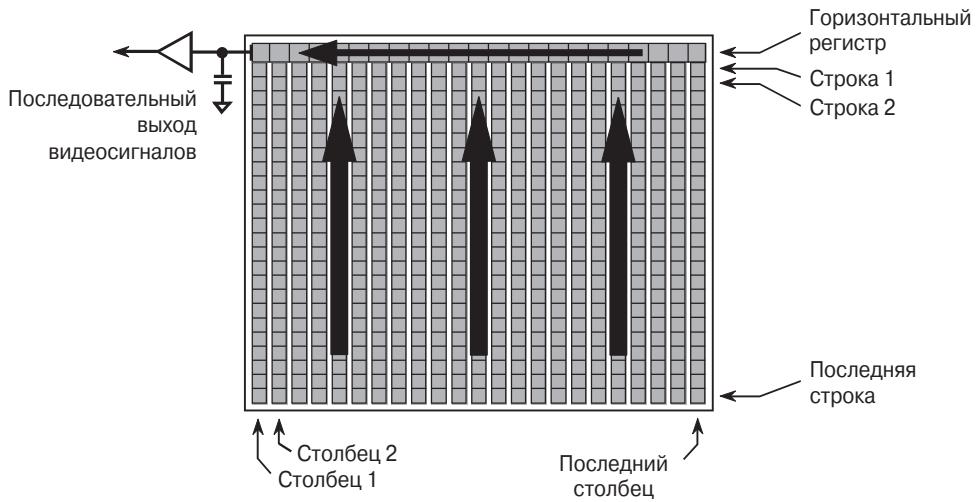
Способ считывания электронного изображения очевиден: накопленные в каждой потенциальной яме электроны передаются в выходной усилитель. Как показано на (в), положительное напряжение прикладывается к двум фазам. В результате каждая потенциальная яма расширяется «влево». Как показано на (г),



**Рис. 23.7.** Работа прибора с зарядовой связью. Как показано на рисунке, пластина кремния *p*-типа покрыта изолирующим слоем и равномерным рядом электродов. Электроды объединены в три группы, чтобы можно было подавать три разных напряжения:  $\phi 1$ ,  $\phi 2$  и  $\phi 3$ . Когда к электроду приложено положительное напряжение, дырки (т. е. положительно заряженные носители, отмеченные знаком «+») отталкиваются от него. В результате в области, называемой потенциальной ямой, количество дырок уменьшается. Приходящий свет генерирует дырки и электроны, в результате чего электроны накапливаются в каждой потенциальной яме (обозначены символом « $-$ »). Управляя напряжением на трёх электродах, электроны с каждой потенциальной ямы можно переместить к зарядо-чувствительному усилителю, преобразующему заряд в напряжение.

следующий шаг состоит в том, чтобы убрать напряжение с первой фазы, что вызывает исчезновение первоначальной потенциальной ямы. В результате накопленные электроны перемещаются на одну потенциальную яму правее того места, откуда они «стартовали». Повторение этой процедуры заставляет электроны перемещаться вправо до тех пор, пока они не достигнут *зарядо-чувствительного усилителя*, представляющего собой конденсатор и согласующий буферный усилитель. Когда электроны попадают на конденсатор с последней потенциальной ямой, они начинают его заряжать и создают на нём потенциал напряжения. Для достижения высокой чувствительности конденсаторы делают очень малой ёмкости, обычно меньше 1 пФ. Этот конденсатор и усилитель являются частью микросхемы ПЗС и располагаются на том же кристалле кремния. Выходной сигнал ПЗС представляет собой последовательность уровней напряжения, пропорциональных количеству света, попавшего на соответствующую потенциальную яму.

**Рис. 23.8** показывает, как двумерное изображение считывается из ПЗС. После периода накопления заряд, созданный в каждой потенциальной яме, движется по столбцам строка за строкой. Например, все потенциальные ямы в строке 15 сначала перемещаются в строку 14, затем в строку 13, затем в строку 12 и т. д. Каждый раз, когда строки передвигаются, все потенциальные ямы в строке номер 1 пересылаются в *горизонтальный регистр*, который быстро перемещает заряд в горизонтальном направлении к зарядо-чувствительному усилителю.



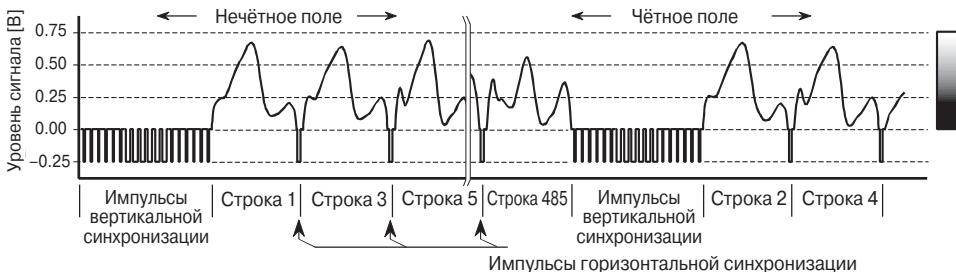
**Рис. 23.8.** Структура ПЗС. Потенциальные ямы изображения ПЗС размещены в столбцах. Во время чтения заряды из каждой потенциальной ямы перемещаются по столбцам в горизонтальный регистр, а затем считаются зарядо-чувствительным усилителем.

Заметим, что такая архитектура считывания преобразует двумерный массив в последовательный поток данных. Первый пиксель, который должен быть прочитан, находится в левом верхнем углу изображения. Вывод происходит слева направо по первой строке, а затем продолжается слева направо на следующих строках. Это называется *главным порядком строк*. Он почти всегда используется, когда двумерный массив преобразуется в последовательные данные.

## 23.3. ТЕЛЕВИЗИОННЫЙ ВИДЕОСИГНАЛ

Хотя телевидению уже больше 50 лет, стандартный *телеизионный сигнал* все ещё остаётся одним из самых распространённых способов передачи изображений. Рис. 23.9 показывает, как телевизионный сигнал выглядит на осциллографе. Такой сигнал называется *составным* и содержит, кроме действительной информации о картинке, импульсы горизонтальной и вертикальной синхронизации. Эти импульсы используются в телевизионном приёмнике, чтобы синхронизировать цепи горизонтального и вертикального отклонения электронного луча. Каждая секунда стандартного телесигнала содержит 30 полных изображений, называемых *кадрами*. Теленженер может сказать, что каждый кадр содержит 525 строк. А нас это число может ввести в заблуждение. Дело в том, что только 480...486 строк содержат видеинформацию, а оставшиеся 39...45 строк предназначены для синхроимпульсов, обеспечивающих правильную развёртку по горизонтали и вертикали.

Стандартный телевизионный сигнал использует чересстрочный формат, чтобы уменьшить *дрожание* воспроизведимого изображения. Это значит, что сначала передаются все нечётные строки каждого кадра, за которыми следуют чётные строки. Группа нечётных строк называется *нечётным полем*, а группа чётных строк — *чётным полем*. Так как каждый кадр состоит из двух полей, то телесигнал передаётся со скоростью 60 полей в секунду. Каждое поле начинается со сложной последовательности вертикальных синхроимпульсов длительностью 1.3 мс. За ними следуют либо нечётные, либо чётные строки видеосигнала. Каждая строка имеет длительность 63.5 мкс, включая 10.2 мкс горизонтального синхроимпульса, отделяющего одну строку от другой. В каждой строке аналоговое напряжение соответствует уровню серого в изображении со значением яркости, отсчитываемым относительно синхроимпульса. Синхроимпульсы размещаются ниже диапазона чёрного цвета. Можно сказать, что синхроимпульсы «*чёрнее чёрного*».



**Рис. 23.9.** Составной видеосигнал. Телевизионный видеосигнал в соответствии с требованиями Национального комитета по телевизионным сигналам состоит из 30 полных кадров (изображений) в секунду, каждый из которых содержит 480...486 строк. Каждый кадр разбит на два поля, одно содержит нечётные строки, а другое — чётные. Каждое поле начинается с серии вертикальных синхроимпульсов, за которыми следуют последовательно идущие строки видеинформации, разделённые горизонтальными синхроимпульсами (горизонтальная ось на этом рисунке нарисована не в масштабе).

Устройство, используемое для аналого-цифрового преобразования видеосигнала, называется *устройством ввода изображения* или *граббером*. Обычно это электронная плата, подключённая к компьютеру и соединённая с видеокамерой через коаксиальный кабель. По команде программного обеспечения граббер на-

чинаят ожидать следующий кадр, начало которого отмечено импульсом вертикальной синхронизации. В течение следующих двух полей каждая строка видеосигнала квантуется: обычно используется 512, 640 или 720 выборок на строку при 8 битах на выборку. Эти выборки сохраняются в памяти компьютера как одна строка цифрового изображения.

Такой способ ввода цифрового образа выявляет существенное различие между вертикальным и горизонтальным направлениями. Каждая строка в цифровом изображении соответствует одной строке видеосигнала и, следовательно, одной строке потенциальных ям ПЗС. К сожалению, соответствие столбцов не такое строгое. В ПЗС каждая строка содержит 400...800 потенциальных ям (столбцов) в зависимости от конкретно используемого устройства. Когда строки потенциальных ямчитываются из ПЗС, видеосигнал фильтруется, превращаясь в сглаженный аналоговый сигнал, такой как на **Рис. 23.9**. При этом результирующий видеосигнал не зависит от того, как много столбцов имеется в ПЗС. Разрешение по горизонтали ограничено допустимой скоростью изменения аналогового сигнала. Для цветного телевидения это обычно 3.2 МГц, так что время нарастания сигнала около 100 нс, т. е. около 1/500 от видеостроки 53.2 мкс.

Когда видеосигнал оцифровывается устройством ввода, он преобразуется в столбцы. Однако эти столбцы в цифровом изображении не имеют никакого отношения к столбцам в ПЗС. Число столбцов в цифровом изображении зависит только от того, как много раз устройство ввода квантует каждую строку. Например, ПЗС может иметь 800 потенциальных ям на строку, в то время как цифровое изображение только 512 пикселей (т. е. столбцов) на строку.

Число столбцов в цифровом изображении важно и по другой причине. Стандартное телевизионное изображение имеет *отношение сторон*  $4\times 3$ , т. е. ширина немного больше высоты. Киноформат имеет большее отношение:  $25\times 9$ . ПЗС, используемый для научных приложений, часто имеет отношение сторон  $1\times 1$ , т. е. идеальный квадрат. Во всех случаях отношение сторон в ПЗС определяется размещением электродов и не может меняться. В то же время отношение сторон цифрового изображения зависит от числа выборок на строку. Может возникнуть проблема, когда изображение показывают на видеомониторе или в виде распечатки. Если отношение поддерживается не точно, изображение может сжиматься по вертикали или по горизонтали.

Описанный здесь 525-строчный сигнал называется *NTSC (National Television Systems Committee)*. Этот стандарт был определён ещё в 1954 году и используется в США и Японии. В Европе существует два стандарта, называемые *PAL (Phase Alteration by Line)* и *SECAM (Sequential Chrominance And Memory)*. Базовые концепции этих стандартов одинаковы, но числа различны. И PAL, и SECAM работают с 25 чересстрочными кадрами в секунду и 625 строками на кадр. Так же как в NTSC, некоторые из этих строк относятся к вертикальной синхронизации, так что остается всего примерно 576 строк, несущих информацию о картинке. Другое более тонкое различие относится к тому, как к сигналу добавляются цвет и звук.

Наиболее простой путь обеспечения цвета телевизионных изображений состоит в отдельной передаче аналоговых сигналов, для каждого из трёх цветов, которые различает человеческий глаз: красного, зелёного и синего. К сожалению, историческое развитие телевидения не позволяло реализовать такую простую схему. Информация о цвете была добавлена в телевизионный сигнал так, чтобы

многочисленные уже используемые телевизионные приёмники чёрно-белого изображения могли использоваться без какой-либо модификации. Сохранив сигнал, несущий информацию о яркости, прежним, просто добавили отдельный сигнал для информации о цвете. В результате получили сигнал яркости и сигнал цветности. Сигнал цветности передаётся на несущей частоте 3.58 МГц, добавленной к «чёрно-белому сигналу». Звук добавляется таким же способом на несущей 4.5 МГц. Телевизионный приёмник разделяет эти три сигнала, обрабатывает их индивидуально и объединяет при воспроизведении.

## 23.4. Другой способ формирования и демонстрации изображений

Не все изображения воспринимаются во времени целым кадром. Другой очень часто встречающийся вариант — это *сканирование по строкам*. Оно подразумевает использование детектора, формирующего одномерный массив пикселей, скажем, 2048 пикселей в длину и 1 пиксель в ширину. Когда объект движется мимо детектора, изображение строится строки за строкой. Такой подход используется, например, в факсах и рентгеновских сканерах багажа в аэропорту. Объект может быть и стационарным, а детектор перемещаться относительно его. Например, детектор, расположенный на корпусе самолёта, «снимает» изображение земли под ним. Достоинством сканирования по строкам является возможность выбора компромисса между скоростью работы и простотой устройства детектора. Например, факсу требуется несколько секунд, чтобы отсканировать полную страницу текста, но получаемое в результате изображение содержит тысячи строк и столбцов.

Ещё более упрощённый подход — восприятие изображения точка за точкой. Например, микроволновое изображение Венеры в описанном нами примере формировалось по одному пикселию за одно зондирование. Другой пример — это сканирующий зондовый микроскоп, способный «видеть» отдельные атомы. Маленький зонд, часто размером с один атом, очень близко придвигается к образцу, который должен быть изображён. Между зондом и образцом могут быть обнаружены квантово-механические эффекты, позволяющие зонду останавливаться на точном расстоянии от поверхности образца. Зонд перемещается над поверхностью образца, поддерживая постоянное расстояние и отслеживая пики и впадины. В конечном изображении значение каждого пикселя представляет высоту соответствующего места на поверхности образца.

Печатные изображения делятся на две категории: *в шкале серого и полутонаовые*. Каждый пиксель изображения в шкале серого — это оттенок серого между чёрным и белым, как на фотографии. Для сравнения: каждый пиксель полутонаового изображения формируется из множества отдельных точек, каждая из которых совершенно чёрная или совершенно белая. Оттенки серого воспроизводятся различным числом этих чёрных и белых точек. Например, представьте лазерный принтер с разрешением 600 точек на дюйм. Чтобы воспроизвести 256 уровней яркости между чёрным и белым, каждому пикселию должен соответствовать массив  $16 \times 16$  печатаемых точек. Чёрный пиксель формируется из 256 точек чёрного цве-

та; белый пиксель — из 256 точек белого цвета. Средний (серый) пиксель состоит из половины точек белого и половины точек чёрного цвета. Так как каждая точка слишком мала, чтобы её можно было разглядеть на обычном расстоянии, глаз воспринимает как бы сформированную шкалу серого. Полутоновые изображения удобны для принтеров и фотокопировальных машин. Недостатком является то, что изображение часто бывает худшего качества по сравнению с картинкой, сформированной в шкале серого.

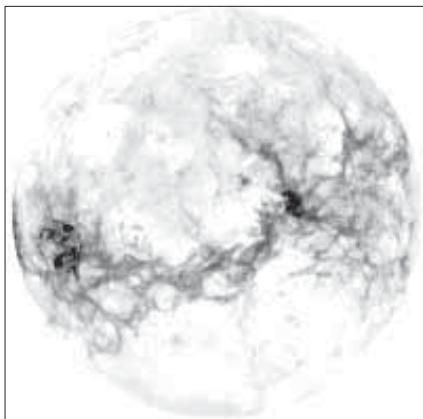
## 23.5. Регулировка яркости и контрастности

Изображение должно иметь подходящую для просмотра **яркость** и **контрастность**. Термин «яркость» означает степень освещённости всего изображения, а контрастность — это разница в яркости между отдельными объектами или областями изображения. Например, белый кролик, бегущий по снежному полю, имеет плохую контрастность, а чёрный пёс на таком же белом фоне имеет хорошую контрастность. **Рис. 23.10** показывает четыре возможных варианта плохой настройки яркости и контрастности. Когда яркость слишком высока, как на **(а)**, наиболее светлые пиксели оказываются в насыщении, увеличивая процент белых пикселей и скрывая детали в этих областях. Обратная картина показана на **(б)**: там, где яркость очень низка, преобладают чёрные пиксели. На **(в)** проиллюстрирован случай высокой контрастности, в результате чего чёрное становится слишком чёрным, а белое слишком белым. Наконец, изображение на **(г)** имеет слишком низкую контрастность, и все пиксели имеют средний оттенок по шкале серого, что делает объекты сливающимися.

**Рис. 23.11** и **23.12** иллюстрируют значение яркости и контрастности более детально. Тестовое изображение, показанное на **Рис. 23.12**, использует шесть различных уровней яркости и контрастности. **Рис. 23.11** показывает структуру тестового изображения: массив из  $80 \times 32$  пикселей, где каждый пиксель имеет значение 0...255. Фон изображения заполнен случайным шумом, равномерно распределённым между 0 и 255. Три квадрата формируются пикселями, значения которых слева направо равны 75, 150 и 225. Каждый квадрат содержит два треугольника со значениями пикселей, слабо отличающимися от их среднего значения. Другими словами, это тёмные, серые и светлые области с мало различающимися деталями.

**Рис. 23.12** демонстрирует, как регулировкой контрастности и яркости можно сделать видимыми разные детали одного и того же изображения. На **(а)** яркость и контрастность установлены на нормальном уровне, как показывают *полосы прокрутки В и С* слева от изображения. Теперь обратим внимание на график, приведённый рядом с каждым изображением, называемый *выходным преобразованием, выходной таблицей соответствия или гамма-кривой*. Гамма-кривая задаёт правило воспроизведения изображения. Значение каждого пикселя в исходном изображении — это число 0...255. Каждый пиксель пропускается через таблицу соответствия и преобразуется в другое число: 0...55. Гамма-кривая определяет закон, по которому числа (яркости изображения), записанные в память, преобразуются в яркости изображения, выводимого на экран.

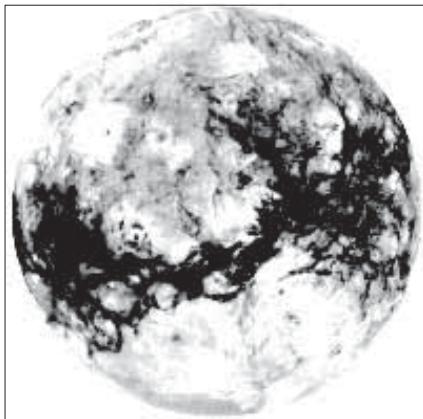
На **(а)** показан случай, когда гамма-кривая настроена так, чтобы ничего не менялось и цифровой выход был идентичен цифровому входу. Каждый пиксель в



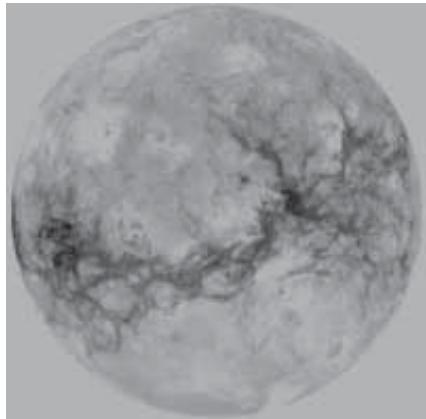
а) Яркость слишком высока



б) Яркость слишком низка



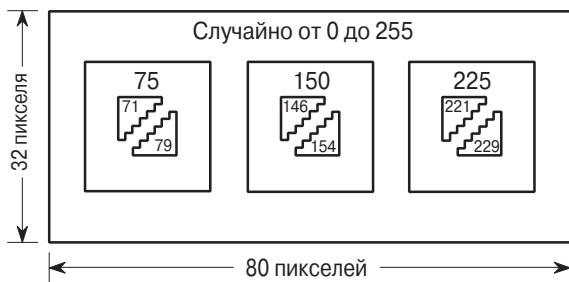
в) Контрастность слишком высока



г) Контрастность слишком низка

**Рис. 23.10.** Регулировка яркости и контрастности. Увеличение яркости приводит к тому, что каждый пиксель в изображении становится более светлым. Увеличение контрастности делает светлые области более светлыми, а тёмные области — более тёмными. Эти изображения показывают влияние плохой настройки яркости и контрастности.

**Рис. 23.11.** Тестовое изображение для настройки яркости и контрастности. Это структура цифрового изображения, используемого на **Рис. 23.12**. Три показанных квадрата — это тёмный, среднетёмный и светлый объекты; каждый из них содержит два низкоконтрастных треугольника. На рисунке приводятся числа, характеризующие яркость пикселей в соответствующих областях.



шумовом фоне является оттенком серого, равномерно распределённым между чёрным и белым. Три квадрата, показанные как тёмное, среднее и светлое, явно отличаются друг от друга. А вот треугольники внутри каждого квадрата рассмотреть нелегко, потому что контрастность слишком мала для глаза, чтобы отличить эти области от фона.

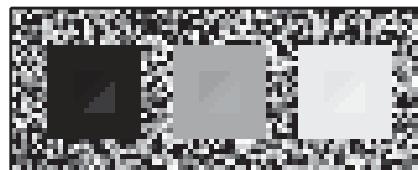
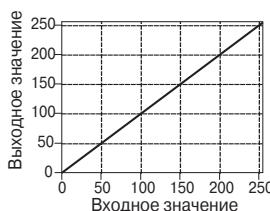
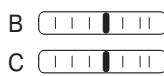
На **(б и в)** показано влияние регулировки яркости. Увеличение яркости «сдвигает» гамма-кривую влево, а уменьшение яркости — вправо. Увеличение яркости делает каждый пиксель в изображении более светлым. И наоборот, при уменьшении яркости каждый пиксель изображения становится более тёмным. Эти изменения могут улучшить визуальное восприятие чрезмерно чёрной или чрезмерно белой областей изображения. Но, если зайти слишком далеко, картинка будет *перенасыщена*. Например, все пиксели в крайнем правом квадрате на **(б)** изображены с максимальной интенсивностью, т. е. 255. Противоположный эффект показан на **(в)**, где все пиксели в крайнем левом квадрате показаны совершенно чёрными, т. е. численно равными 0. Так как все пиксели в этих областях имеют одинаковое значение, треугольники совершенно стираются. Отметим, что на **(б и в)** треугольники увидеть не легче, чем на **(а)**. Изменение яркости обеспечивает небольшое улучшение в различении объектов низкой контрастности от их окружения (фона).

На **(г)** проиллюстрирован случай оптимальных для просмотра значений пикселей, численно расположенных около 75. Такой результат был достигнут увеличением контрастности за счёт повышения крутизны гамма-кривой. Например, значения пикселей 71 и 75 исходного изображения переходят в значения 100 и 116 при отображении с повышением контрастности в 4 раза. Пиксели со значениями 49...109 из совершенно чёрных превращаются в совершенно белые. Плата за повышение контрастности состоит в том, что значения пикселей 0...45 рассматриваются теперь как чёрные, а значения пикселей 110...255 — как белые. Как показано на **(г)**, увеличение контрастности приводит к тому, что треугольники в левом квадрате становятся видны за счёт насыщения среднего и правого квадратов.

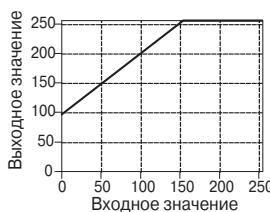
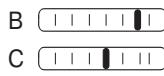
На **(д)** иллюстрируется дальнейшее увеличение контрастности, в результате которого получается, что только исходные уровни со значениями 16 и 256 выглядят ненасыщенными. Яркость падает так, что 16 пригодных уровней принимают центральное значение 150. Детали в центральном квадрате теперь очень хорошо видны, однако почти все другие детали изображения оказались слишком насыщены. Они стали выглядеть как шум вокруг рамки изображения. Имеется очень немного пикселей с промежуточным оттенком серого; почти каждый пиксель либо абсолютно белый, либо абсолютно чёрный. Этот метод использования высокой контрастности с целью выявления лишь нескольких уровней называется *растяжением шкалы серого*.

Регулировка контрастности — это способ сконцентрироваться на некотором малом диапазоне значений пикселей. Регулировка яркости позволяет «центрировать» интересующую нас область значений пикселей вокруг наиболее благоприятного значения. Большинство цифровых видеосистем позволяют управлять яркостью и контрастностью изображения и часто обеспечивают графический вывод гамма-кривой (как на **Рис. 23.12**). Для сравнения: управление и яркостью, и контрастностью в аналоговых телевизионных и видеосистемах осуществляют аналоговые цепи, и работать они могут отдельно. Например, управлять контрастнос-

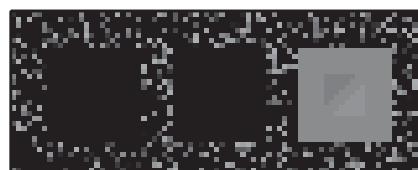
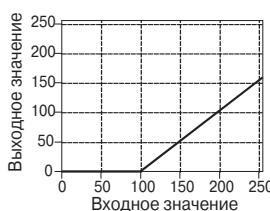
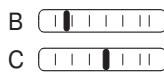
**a)** Нормальное



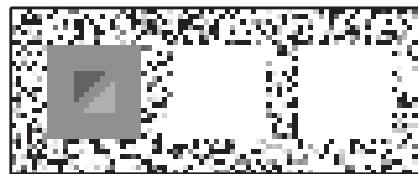
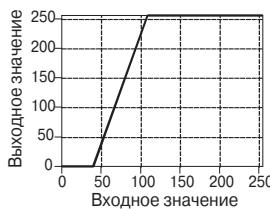
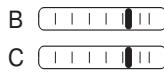
б) Увеличенная яркость



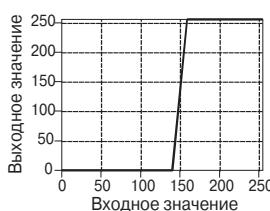
**в)** Уменьшенная яркость



г) Слабо увеличенная контрастность при значениях пикселей, равных 75



д) Сильно увеличенная контрастность при значениях пикселей, равных 150



**е)** Увеличенная контрастность при значениях пикселей, равных 75 и 225

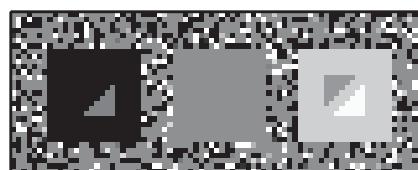
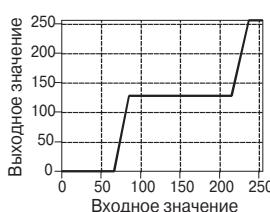
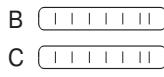


Рис. 23.12. Шесть уровней яркости и контрастности.

тью монитора можно регулировкой усиления аналогового сигнала, а яркость можно добавить или убавить смещением напряжения постоянного тока. При этом не удивляйтесь, если регулировка не даст удовлетворительного результата.

## 23.6. Преобразование шкалы серого

Изображение на Рис. 23.12e отличается от остальных. Гамма-кривая здесь идёт под углом не на *одном*, а на *двух* участках. Это позволяет показывать одновременно треугольники в левом и в правом квадратах. Конечно, результатом этого является то, что ряд пикселей, значения которых удалены от выбранных областей, оказываются в насыщении. Заметьте, что полосы прокрутки, управляющие контрастностью и яркостью, не показаны на рисунке. Здесь проиллюстрирован такой результат настройки изображения, который не может быть достигнут только регулировкой яркости и контрастности.

*Преобразование шкалы серого* даёт хорошие результаты в многочисленных методах повышения качества изображений. Идея состоит в том, чтобы повысить контрастность пикселей, представляющих интерес, за счёт пикселей, не представляющих интереса. Это делается определением относительной важности каждого возможного значения пикселей от 0 до 255. Чем более важно значение, тем большей контрастностью оно наделяется в выходном изображении. Рассмотрим использование этой процедуры на примере.

а) Исходное инфракрасное изображение



б) Преобразование шкалы серого

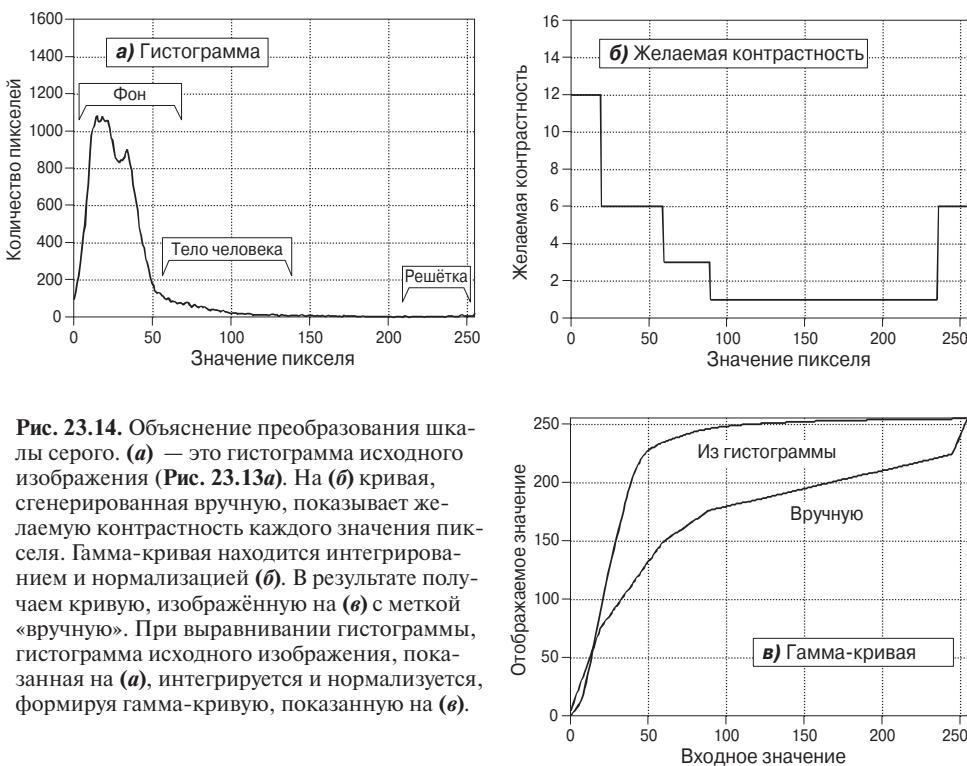


Рис. 23.13. Преобразование шкалы серого. Изображение на (а) получено инфракрасной камерой в полной темноте. Яркость изображения соответствует температуре, в результате чего вырисовывается силуэт человека и контуры горячей решётки радиатора автомобиля. Изображение на (б) получено с помощью преобразования шкалы серого, подобранного «вручную» и проиллюстрированного на Рис. 23.14в.

Изображение на Рис. 23.13а получено в полной темноте с использованием ПЗС-камеры, которая чувствительна к инфракрасному излучению. Отобража-

мый параметр — температура: чем горячее объект, тем больше инфракрасной энергии он излучает и более ярким выглядит на изображении. С учётом этого фон оказался очень чёрным (холодным), человек — серым (тёплым), а решётка радиатора автомобиля — белой (горячей). Системы такого рода широко используются военными и полицией. Они позволяют видеть другого человека, когда он не может видеть даже сам себя! Изображение на Рис. 23.13а трудно разглядеть из-за неравномерного распределения значений пикселей. Большая часть изображения так темна, что детали не могут быть видны на снимке. С другой стороны, изображение решётки сосредоточено вблизи белого.

Гистограмма этого изображения, представленная на Рис. 23.14а, показывает, что фон, человек и решётка имеют существенно различающиеся уровни яркости. Увеличим контрастность фона и решётки за счёт всего остального, включая тело человека (б). Мы установили, что самые малые значения пикселей, т. е. фон, будут иметь относительный контраст 12. Точно так же наибольшие значения пикселей, т. е. решётка, будут иметь относительный контраст 6. Человек будет иметь относительный контраст 1. Переход между этими областями оказывается ступенчатым. Все указанные значения выбраны экспериментально.



Преобразование шкалы серого, подобранное «вручную» с использованием описанной стратегии, проиллюстрировано на (в). Оно найдено вычислением текущей суммы (т. е. дискретного интеграла) кривой на (б) и нормализацией до требуемого максимального значения, равного 255. Почему применяется интегриро-

вание? Давайте подумаем. Значение контрастности конкретного пикселя равно крутизне гамма-кривой. Поэтому мы хотим, чтобы график на (б) был производной (крутизной наклона) кривой на (в). Это значит, что кривая на (в) должна быть интегралом кривой на (б).

Подвергая изображение на Рис. 23.13а преобразованию шкалы серого, получаем изображение, показанное на Рис. 23.13б. Фон стал более светлым; решётка — более тёмной; повысилась контрастность того и другого. Эти улучшения получены ценой снижения контрастности силуэта человека, который становится менее детальным (отметим, что он не может получиться хуже, чем в первоначальном изображении).

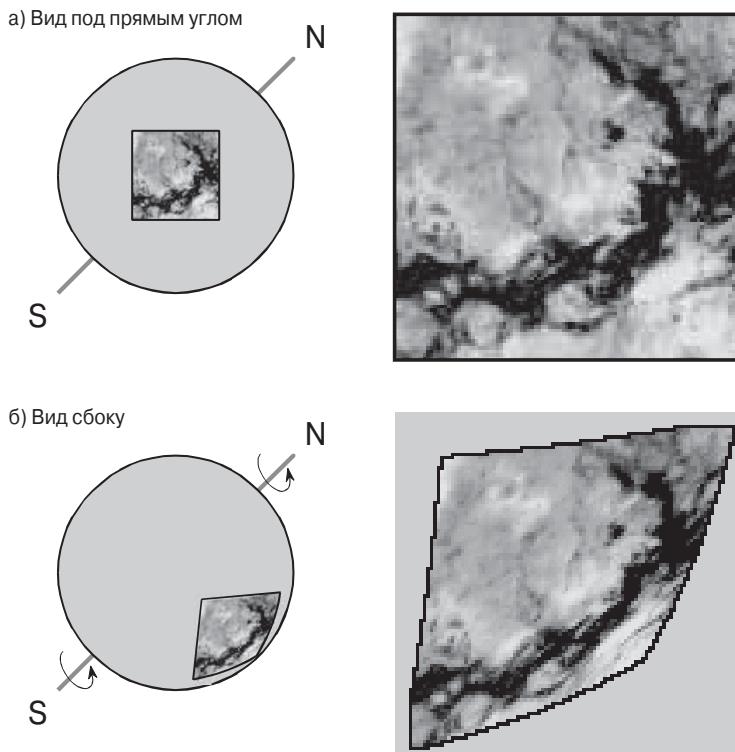
Преобразование шкалы серого может значительно улучшить качество изображения. Проблема состоит в том, что оно требует большого объёма работы методом проб и ошибок. Рациональной является автоматизация этой процедуры. Для этих целей служит так называемое *выравнивание гистограммы*. Заметим, что гистограмма на Рис. 23.14а и кривая весов контрастности на Рис. 23.14в имеют одинаковую общую форму. Выравнивание гистограммы «вслепую» использует её как кривую весов контрастности, уменьшая необходимость участия человека. Выходное преобразование (гамма-кривая) рассчитывается интегрированием и нормализацией самой гистограммы, а не подобранный «вручную». В результате наибольшей контрастностью наделяются те пиксели, количество которых наиболее велико.

Выравнивание гистограммы — достаточно интересная математическая процедура. Она максимизирует энтропию изображения — меру того, как много информации передаётся фиксированным числом битов. Недостаток процедуры состоит в том, что она «путает» количество пикселей с их важностью. Например, на Рис. 23.13 наиболее значимыми являются решётка автомобиля и человек. Несмотря на это, выравнивание гистограммы почти полностью игнорирует эти объекты, так как они представлены относительно малым числом пикселей. Выравнивание гистограммы является быстрым и лёгким, но необходимо помнить: если оно работает плохо, то нужно попробовать получить лучший результат с помощью кривой, сформированной «вручную».

## 23.7. Деформирование изображений

Одной из проблем при фотографировании поверхности планет является искажение, связанное со сферичностью их формы. Предположим, например, что вы используете телескоп, чтобы сфотографировать квадратную область вблизи центра планеты, как показано на Рис. 23.15а. Через несколько часов планета повернется вокруг своей оси и будет выглядеть, как на (б). Ранее сфотографированная область станет очень искажённой, искривляясь относительно горизонтальной оси планеты. Каждое из двух изображений содержит полную информацию об интересующей нас области, но с разных ракурсов. Часто получают фотографию, такую как на (а), желая в действительности получить изображение, как на (б), или наоборот. Например, спутник, фотографируя поверхность планеты, может снять тысячи изображений непосредственно под ним, как на (а). Чтобы получить естественно выглядящую картину всей планеты, такую как изображение Венеры на

**Рис. 23.1,** каждая фотография должна быть трансформирована и помещена в необходимое место. Другим примером является метеоспутник, ведущий наблюдение за ураганом, находящимся не прямо под ним, а в стороне. Изображение будет получено под углом, как на **(б)**. Потребуется дополнительное преобразование, чтобы изображение стало таким, как оно должно выглядеть сверху **(а)**.



**Рис. 23.15.** Деформирование изображения. Как показано на **(а)**, небольшой фрагмент планеты, наблюдаемый под прямым углом, выглядит относительно неискажённым. А вот вид сбоку **(б)** характеризуется значительными пространственными искажениями. Деформирование — это метод преобразования одного из этих изображений в другое.

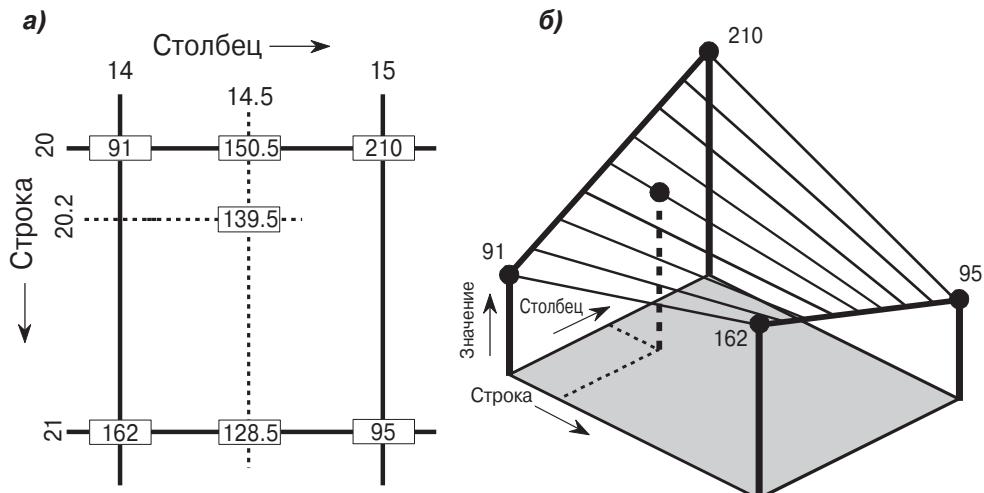
Это пространственное преобразование называется *деформированием изображения*. Космическая фотография наиболее часто использует деформирование, но имеются и другие примеры его использования. Например, многие ламповые детекторы изображений имеют разную величину пространственных искажений. Это приборы ночного видения, используемые военными, и приёмники рентгеновских лучей, применяемые в медицине. Цифровое деформирование изображения может использоваться, чтобы скорректировать нежелательные искажения, присущие этим устройствам. Другой пример — спецэффекты в кино. Здесь художники любят искажать изображения различными методами. Например, метод, называемый *морфингом*, постепенно трансформирует один объект в другой за се-

рию кадров. Это может создать иллюзию того, как ребенок превращается во взрослого или как человек превращается в оборотня.

В процедуре деформирования на «вход» подаётся исходное изображение (двумерный массив), а на выходе получается деформированное изображение (другой двумерный массив). Преобразование осуществляется циклическим опросом всех пикселей формируемого изображения, и необходимо ответить на вопрос: какое правильное значение пикселя следует здесь поместить? Выбирая конкретные строку и столбец, находим элемент деформированного изображения, которому в исходном изображении соответствует свой элемент, со своими номерами строки и столбца. Этот элемент исходного изображения преобразуется в соответствующий пиксель формируемого изображения, в чём и состоит процедура деформирования. Номера строки и столбца элемента исходного изображения, на основе которого формируется пиксель нового деформированного изображения, называют *исходным адресом*. Преобразование каждого пикселя исходного изображения в соответствующие пиксели деформированного изображения — это самая простая часть задачи. Трудная часть состоит в вычислении *исходных адресов*, относящихся к каждому рассчитываемому пикселю деформированного изображения. Обычно это чисто математическая задача, оказывающаяся достаточно сложной и запутанной. Наиболее просто выполнить лишь растягивание изображения в горизонтальном или вертикальном направлениях, для чего достаточно выполнять умножение номеров строк и/или столбцов, определяя исходные адреса.

Одной из методик, используемых при деформировании, является *субпиксельная интерполяция*. Предположим, например, что вы разработали систему уравнений, переводящую номера строк и столбцов деформированного изображения в соответствующие адреса в исходном изображении. Рассмотрим, что может случиться, когда вы попытаетесь найти значение пикселя в строке 10 и столбце 20 деформированного изображения. Вы подставляете *строку* 10, *столбец* 20 в ваши уравнения и получаете, что номер строки исходного изображения равен 20.2, а номер столбца равен 14.5. Вычисленные вами числа оказались дробными и не могут использоваться в качестве исходных адресов. В этом случае проще всего использовать *алгоритм ближайшего соседа*, просто округляя адреса до ближайшего целого числа. Это действительно просто, но то, что пиксели будут часто оказываться не на своих местах, может привести к появлению сильной волнистости на границах объектов.

Существенно лучших результатов можно добиться с помощью *билинейной интерполяции*, которая оказывается не намного сложнее. **Рис. 23.16** показывает, как она работает. Вам известны значения четырёх пикселей вокруг некоторого адреса, т. е., например, значения пикселей в строках 20 и 21 и столбцах 14 и 15. Будем полагать их равными 91, 210, 162 и 95. Задача состоит в вычислении искомого значения пикселей интерполяцией на основе этих четырёх значений. Это делается в два шага. Сначала проводится интерполяция в горизонтальном направлении между столбцами 14 и 15. Это даёт два промежуточных значения: 150.5 — в строке 20 и 128.5 — в строке 21. Затем проводится интерполяция между этими промежуточными значениями в вертикальном направлении. Это даёт билинейно интерполированное значение пикселя 139.5, которое затем переносится в деформированное изображение. Почему сначала интерполируют в горизонтальном направлении, а затем в вертикальном, а не наоборот? В действительности это дело вкуса; конечный результат одинаков в любом случае.



**Рис. 23.16.** Субпиксельная интерполяция. Субпиксельная интерполяция, используемая при деформировании изображений, обычно выполняется с помощью билинейной интерполяции. Как показано на (а), два промежуточных значения вычисляются с помощью линейной интерполяции в горизонтальном направлении. Искомое значение находится затем с использованием линейной интерполяции в вертикальном направлении между промежуточными значениями. Как показано на (б), эта процедура помогает определить значения любых точек между четырьмя известными пикселями, находящимися в углах.

## ЛИНЕЙНАЯ ОБРАБОТКА ИЗОБРАЖЕНИЙ

*Линейная обработка изображений* основана на тех же двух методах, которые повсеместно используются в ЦОС: *свёртке* и *анализе Фурье*. Свёртка — наиболее важный из них, так как изображение содержит информацию, закодированную скорее в пространственной области, чем в частотной. *Линейная фильтрация* даёт улучшение качества изображений в различных аспектах: увеличивается резкость границ объектов, уменьшается шум, корректируется неравномерность освещения, устраняется расплывчатость, обусловленная движением объектов. Все это достигается свёрткой исходного изображения с соответствующим ядром фильтра, в результате чего создается так называемое отфильтрованное изображение. Серьёзной проблемой при фильтрации изображений оказывается громадное количество вычислений, которое нужно выполнить. Время обработки часто бывает просто неприемлемым. В этой главе описаны методики проектирования ядер фильтров для разных задач обработки изображений. Представлены два важных способа сокращения времени обработки: *свёртка при выполнении условия сепаральности* и *быстрая свёртка*.

### 24.1. Свёртка

*Свёртка изображений* работает так же, как и одномерная. Изображение рассматривается как сумма единичных импульсов, смешённых относительно нуля и умноженных на коэффициент пропорциональности. Линейные системы полностью описываются тем, как они реагируют на единичный импульс, т. е. их *импульсной характеристикой*. Точно так же, как и в случае одномерных сигналов, изображение на выходе системы считается равным входному изображению, свёрнутому с импульсной характеристикой системы.

Единичный импульс применительно к обработке изображений интерпретируется следующим образом: это изображение, состоящее из всех нулей кроме единственного пикселя с координатами — строка 0, столбец 0, имеющего значение, равное единице. Если предположить, что индексы строки и столбца могут иметь положительные и отрицательные значения, то единица оказывается в центре всех нулей. Когда такой *двумерный единичный импульс* проходит через линейную систему, эта единственная ненулевая точка будет превращаться в некоторый двумерный образ. Что может при этом случиться с точкой? Она «рассеивается». Поэтому импульсную характеристику систем обработки изображений часто называют *функцией рассеяния точки* (*FPT*).

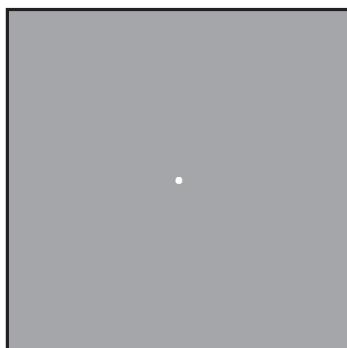
Глаз человека может служить прекрасным примером такого подхода к описанию линейных систем обработки изображений. Как было описано в предыдущей главе, первый слой сетчатки преобразует изображение, представленное световым

образом, в изображение в виде совокупности нейроимпульсов. Второй слой сетчатки обрабатывает это «нейроизображение» и пропускает его к третьему слою — волокнам, формирующими зрительный нерв. Представьте, что изображение, спроецированное на сетчатку, — это очень малое пятно света в центре тёмного фона. Это единичный импульс, поданный на вход зрительной системы человека. В предположении, что система линейна, алгоритм обработки изображения, реализуемый сетчаткой (т. е. поведение системы), может быть определён на основе выходного изображения — образа, появившегося в зрительном нерве. С его помощью мы можем найти *функцию рассеяния точки*, характерную для глаза человека. К вопросу о линейности оптической системы глаза мы вернёмся позднее.

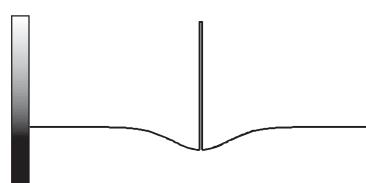
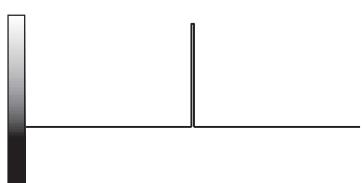
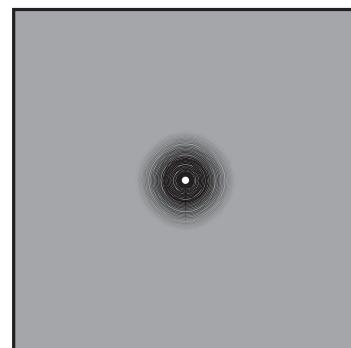
На **Рис. 24.1** проиллюстрирован описанный эксперимент. На **(а)** изображён единичный импульс, попадающий на сетчатку, а **(б)** показывает образ, появившийся в зрительном нерве. Средний слой глаза пропускает яркий пик, но создаёт вокруг него затемнённую область. Эта особенность работы глаза известна как *латеральное торможение*. Если активизирована нервная клетка в среднем слое, она уменьшает способность её ближайших соседей становиться активными. Действительные изображения, видимые глазом, можно рассматривать как совокупность точек (единичных импульсов), каждая из которых вносит свой вклад в изображение, поступающее на зрительный нерв, в виде сдвинутой и промасштабированной функции рассеяния точки. Другими словами, видимое изображение сворачивается с ФРТ глаза для того, чтобы создать нейроизображение, передаваемое в мозг. Возникает естественный вопрос: нельзя ли так выполнить свёртку наблюдаемого изображения с ФРТ глаза, чтобы улучшить функции зрительной системы?

Люди и животные используют зрение для идентификации расположенных вблизи объектов и их классификации, например: враг, пища, друзья и т. д. При

**а) Изображение на первом слое**



**б) Изображение на третьем слое**



**Рис. 24.1.** ФРТ глаза. Средний слой сетчатки трансформирует единичный импульс **(а)** в импульс, окружённый тёмной областью **(б)**. Функция рассеяния точки подчёркивает границы объекта.

этом необходимо осуществлять отделение одних областей на изображении от других, что основано на различиях в яркостях и цвете. Первым шагом в распознавании объектов является идентификация их границ, представляемых как неоднородности, отделяющие объект от фона. Средний слой сетчатки помогает решить эту задачу путём увеличения резкости границ рассматриваемого изображения. В качестве иллюстрации этого подхода на Рис. 24.2 показано изображение, которое медленно изменяется от тёмного к светлому, создавая размытую и нечёткую границу. На (а) показаны уровни яркости, характерные для изображения, поступающего на сетчатку, а на (б) — уровни яркости изображения, появляющегося в зрительном нерве и передаваемого в мозг. Обработка сетчаткой делает границу между светлой и тёмной областями более резкой, усиливая различие между ними.

Однако выбросы на краях граничной полосы создают интересную оптическую иллюзию. Следующая за границей тёмная область оказывается слишком тёмной, а светлая область становится слишком светлой. Получившиеся тёмные и светлые полоски называются *полосами Маха*, в честь австрийского физика Эрнста Маха (1838–1916), описавшего их первым.



**Рис. 24.2.** Полоски Маха. Плавное изменение яркости на границе объекта в исходном изображении (а) после обработки сетчаткой становится более резким (б). Это делает объекты легче различаемыми, но создаёт оптическую иллюзию, называемую *полосами Маха*: заметные выбросы на концах граничной полосы заставляют тёмную область стать ещё темнее, а светлую — ещё светлее. В результате образуются тёмные и светлые полосы, которые идут параллельно границе.

Как и в случае одномерных сигналов, свёртка изображений может рассматриваться со стороны входа и со стороны выхода системы. При рассмотрении со стороны входа считается, что каждый пиксель исходного изображения вносит в выходное изображение определённый вклад: масштабированную и сдвинутую версию функции рассеяния точки. При рассмотрении со стороны выхода определяют, как каждый пиксель выходного изображения формируется на основе группы пикселей входного изображения. Для одномерных сигналов группа входных отсчётов выделяется импульсной характеристикой, «повёрнутой» слева направо. В случае изображения используется ФРТ, «перевёрнутая» слева направо и сверху вниз. Так как большинство ФРТ, используемых в ЦОС, симметричны относительно вертикальной и горизонтальной осей, то переворотов можно не делать. Позднее в этой главе мы расскажем о несимметричных ФРТ, для которых указанные перевороты необходимо принимать в расчёте.

На Рис. 24.3 показано несколько наиболее распространённых ФРТ. На (a) изображена круглая ФРТ, которая имеет вершину округлой формы. Например, если линзы фотокамеры недостаточно хорошо сфокусированы, то каждая точка изображения будет проецироваться на датчик изображения в виде круглого пятна (посмотрите на Рис. 23.2 и представьте, что экран перемещается вперёд-назад по отношению к линзе). Можно сказать, что круглая ФРТ — это функция рассеяния точки несфокусированной линзы.

*Гауссиан (гауссова ФРТ)*, показанный на (б), — это ФРТ, характерная для систем обработки изображений, находящихся под влиянием случайных искажающих воздействий. Например, изображение, полученное с помощью телескопа, расплывается из-за турбулентностей в атмосфере, и каждая точка исходного изображения становится гауссианом в конечном изображении. Датчики изображений, такие как ПЗС или сетчатка, часто испытывают влияние рассеяния света и/или электронов. Центральная предельная теорема гласит, что эти случайные процессы описываются гауссовским законом.

Круглая и гауссова ФРТ используются в обработке изображений так же, как и фильтр скользящего среднего, применяемый к одномерным сигналам. Свёртка изображения с этими ФРТ приводит к расплывчатости и менее резким границам, что не так заметно при наличии шума. Такие фильтры называют *сглаживающими* благодаря характеру их действия во временной области, или *низкочастотными* из-за характера их функционирования в частотной области.

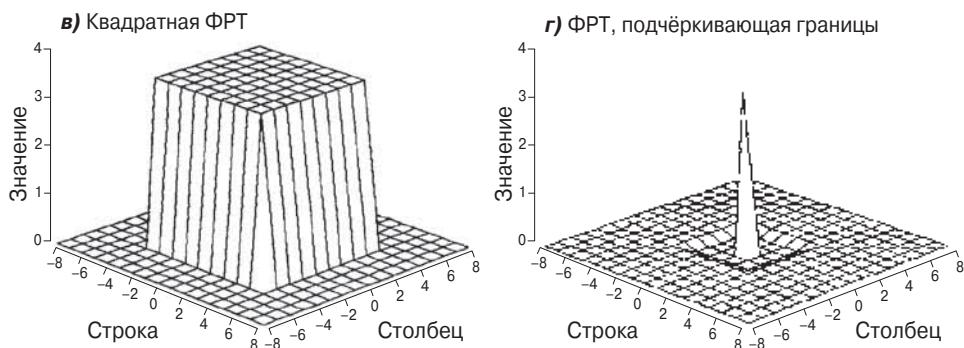
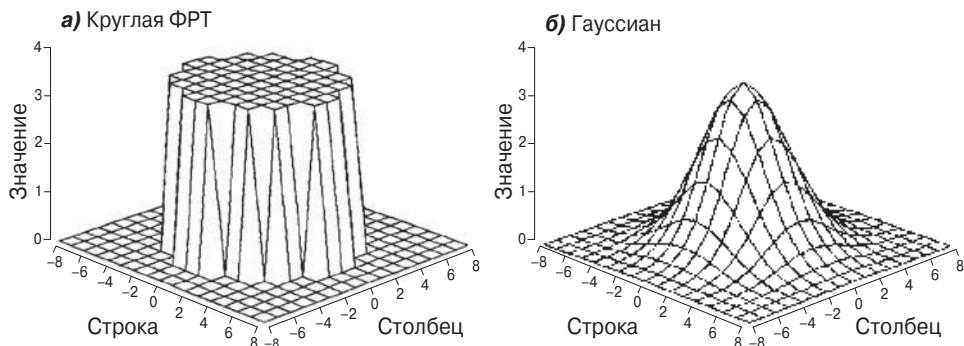
*Квадратная* ФРТ, показанная на (в), также может использоваться как сглаживающий фильтр, но она не является симметричной. Расплывчатость по диагонали отличается от горизонтальной и вертикальной. Это может быть важно или не важно, в зависимости от конкретной ситуации.

Противоположным сглаживающему является фильтр, *увеличивающий резкость границ* (фильтр высоких частот). Метод инверсии АЧХ, о котором шла речь в Главе 14, используется для перехода от одного к другому. Как показано на (г), ядро фильтра, увеличивающего резкость границ, формируется обращением сглаживающего фильтра и добавлением единичного импульса в центре. Обработка изображений, которая происходит в сетчатке глаза, является примером фильтра этого типа.

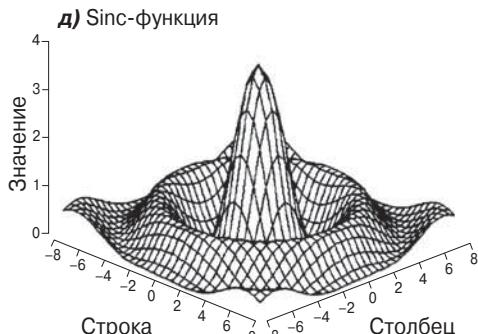
На (д) показана двумерная sinc-функция. Обработка одномерного сигнала использует sinc-окно для выделения полосы частот. Так как изображения не имеют

информации, закодированной в частотной области, то sinc-функция редко используется в качестве ядра фильтра для изображений, хотя она может найти применение в некоторых теоретических проблемах. Sinc-функция может быть трудной для применения, так как её «хвосты» уменьшаются очень медленно ( $1/x$ ), т. е. она может рассматриваться как бесконечно длинная. Для сравнения: хвост гауссиана спадает очень быстро ( $\exp(-x^2)$ ) и может в конечном счёте быть безболезненно усечён.

Все эти ядра фильтров используют отрицательные индексы строк и столбцов, позволяя ФРТ иметь центр в точке: строка 0, столбец 0. Отрицательные индексы часто исключаются при одномерной ЦОС сдвигом ядра фильтра вправо до тех пор,



**Рис. 24.3.** Наиболее распространенные виды функций рассеяния точки. Круглая ФРТ, Гауссиан и квадратная ФРТ, показанные на (а), (б) и (в), являются общераспространёнными ядрами сглаживающих (низкочастотных) фильтров. Фильтры, подчёркивающие границы (высокочастотные), формируются вычитанием низкочастотного ядра из единичного импульса, как показано на (г). Sinc-функция (д) используется очень редко в обработке изображений из-за того, что изображения несут информацию, закодированную в пространственной области, а не в частотной.



пока все ненулевые выборки не будут иметь положительные индексы. Этот сдвиг перемещает весь выходной сигнал на произвольную величину отсчётов, и каких-либо проблем при этом обычно не возникает. А вот относительный сдвиг входного и выходного изображений оказывается вообще недопустим. Поэтому отрицательные индексы являются нормой для ядер фильтров в обработке изображений.

Проблема со свёрткой изображений состоит в том, что требуется очень большое количество вычислений. Например, когда изображение размером  $512 \times 512$  пикселей сворачивается с ФРТ размером  $64 \times 64$  пикселя, то необходимо более миллиарда умножений и сложений ( $64 \times 64 \times 512 \times 512$ ). Большое время обработки делает метод непрактичным. Для ускорения вычислений используют три подхода.

Первая методика состоит в использовании малых ФРТ, часто только  $3 \times 3$  пикселя. Это выполняется циклической обработкой с получением каждого отсчёта в выходном изображении, на основе хорошо оптимизированной подпрограммы, реализующей умножение и сложение девяти пикселей входного изображения. Неожиданно большое число задач можно решить с использованием ФРТ малой размерности (всего лишь  $3 \times 3$  пикселя) благодаря тому, что такая размерность оказывается достаточной для выделения границ объектов.

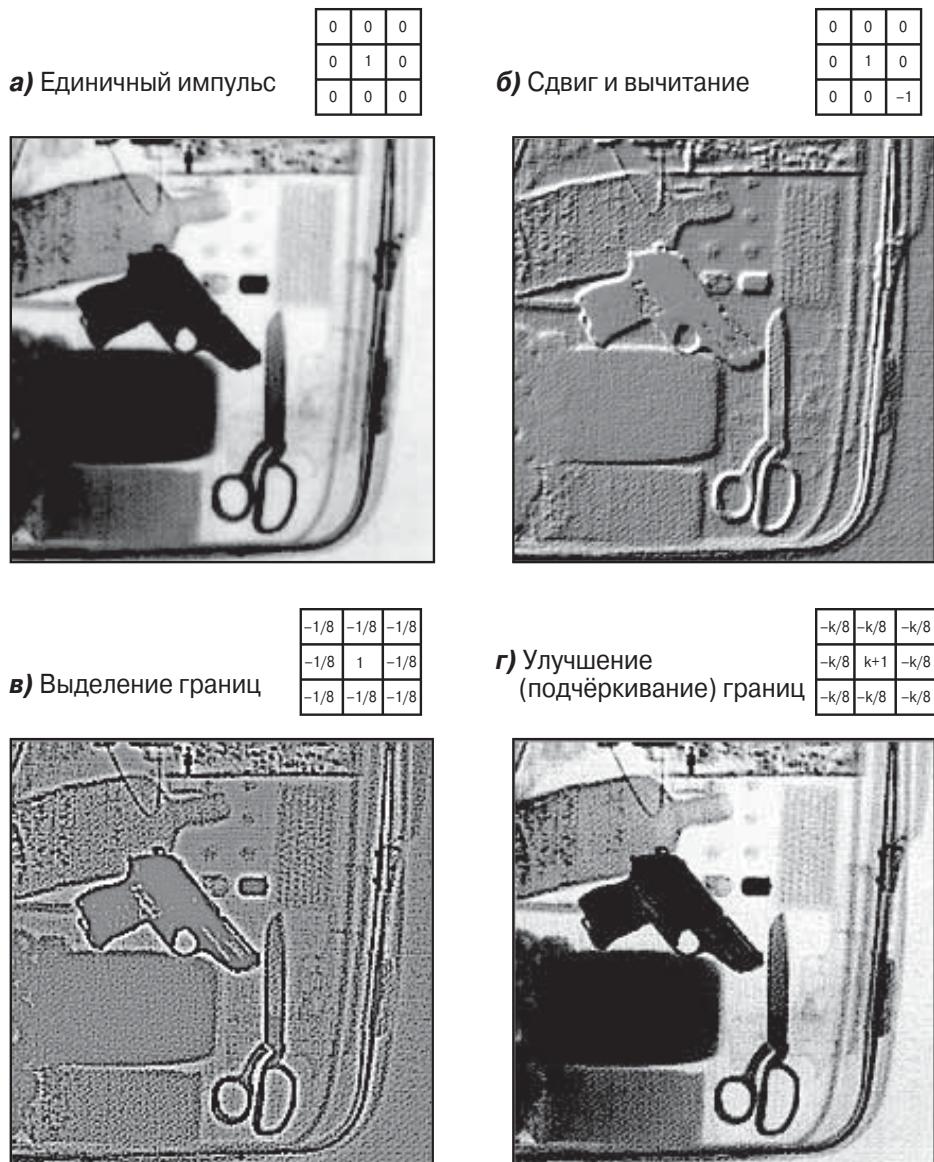
Вторая стратегия используется, когда необходима ФРТ большой размерности, но её форма не критична. В этом случае используют ядра фильтров, называемые *сепарельными*, которые позволяют выполнить свёртку изображений как последовательность одномерных операций. Этот приём может повысить скорость обработки в сотни раз.

Третья стратегия — это быстрая свёртка (свёртка на основе БПФ), используемая, когда ядро фильтра большое и имеет специфичную форму. Даже при значительном увеличении скорости, обеспечиваемой высокой эффективностью БПФ, время выполнения будет слишком большим. Рассмотрим в деталях эти три подхода и их применение в обработке изображений.

## 24.2. Модификация границ с помощью ФРТ размерностью $3 \times 3$ пикселя

Рис. 24.4 показывает некоторые операции с ФРТ размерностью  $3 \times 3$  пикселя. На (а) показано изображение, полученное рентгеновским сканером багажа в аэропорту. Когда это изображение сворачивается с единичным импульсом размером  $3 \times 3$  пикселя (единица, окружённая 8 нулями), изображение остаётся неизменным. Это само по себе не представляет пока интереса, но формирует основу для других ядер фильтров.

На (б) показано изображение, свёрнутое с ядром, состоящим из единицы, минус единицы и 7 нулей. Эта операция называется сдвиг и вычитание, потому что сдвинутая версия изображения (соответствующая  $-1$ ) вычитается из первоначального изображения (соответствует  $1$ ). Эта обработка создаёт оптическую иллюзию того, что некоторые объекты ближе или дальше, чем фон, создавая эффект трёхмерного или рельефного изображения. Мозг интерпретирует изображение так, как если бы свет падал сверху, что является естественным для окружающего



**Рис. 24.4.** Модификация границ с помощью ФРТ размером  $3 \times 3$  пикселя. Первоначальное изображение (**а**) было получено рентгеновским сканером в аэропорту. Операция «сдвиг и вычитание», показанная на (**б**), создаёт псевдотрёхмерный эффект. Оператор выделения границ (**в**) удаляет все контрасты, оставляя только информацию о границах. Фильтр улучшения границ (**г**) складывает изображения (**а** и **в**) с разными весами, определяемыми параметром  $k$ . Значение  $k$  в данном случае равно двум.

нас мира. Если границы объекта яркие на вершине и тёмные у основания, то объект выглядит приподнятым над фоном. Чтобы увидеть другой интересный эффект, повернём картину верхней стороной вниз, и объекты будут выглядеть «вдавленными» в фон.

На (в) изображена ФРТ, выделяющая границы, и результат обработки фильтром с такой ФРТ. Каждая граница в исходном изображении трансформируется в узкие тёмную и светлую полосы, которые идут параллельно исходной границе. Последующее применение пороговой обработки способно удалить или тёмную, или светлую полосу, обеспечивая простой алгоритм для выделения границ.

Общеизвестным методом обработки изображений, проиллюстрированным на (г), является *улучшение*, или *подчёркивание, границ*. На (а) объекты имеют хорошую контрастность (соответствующие уровни света и темноты), но очень расплывчатые границы. На (в) объекты имеют очень низкую контрастность, но хорошо выделенные границы. Стратегия состоит в том, чтобы умножить изображение с хорошими границами на константу  $k$  и суммировать результат с изображением с хорошей контрастностью. Это эквивалентно свёртке исходного изображения с ФРТ размерностью  $3 \times 3$  пикселя, показанной на (г). Если  $k$  установить равным нулю, ФРТ превратится в единичный импульс, а образ не изменится. Когда  $k$  становится больше, подчёркивание границ улучшается. Для изображения на (г) значение  $k = 2$ : две части изображения (в) к одной части изображения (а). Эта операция имитирует способность глаза подчёркивать границы, позволяя более легко отличить объект от фона.

Свёртка с любой из этих ФРТ может дать отрицательные значения пикселей в конечном изображении. Даже если программа способна работать с отрицательными значениями, показать такое изображение будет невозможно. Наиболее известный способ решения этой проблемы состоит в добавлении смещения к каждому вычисленному пикселию, как это сделано на показанных изображениях. Альтернативой может служить исключение значений, лежащих вне допустимого диапазона.

## 24.3. Свёртка при выполнении условия сепарабельности

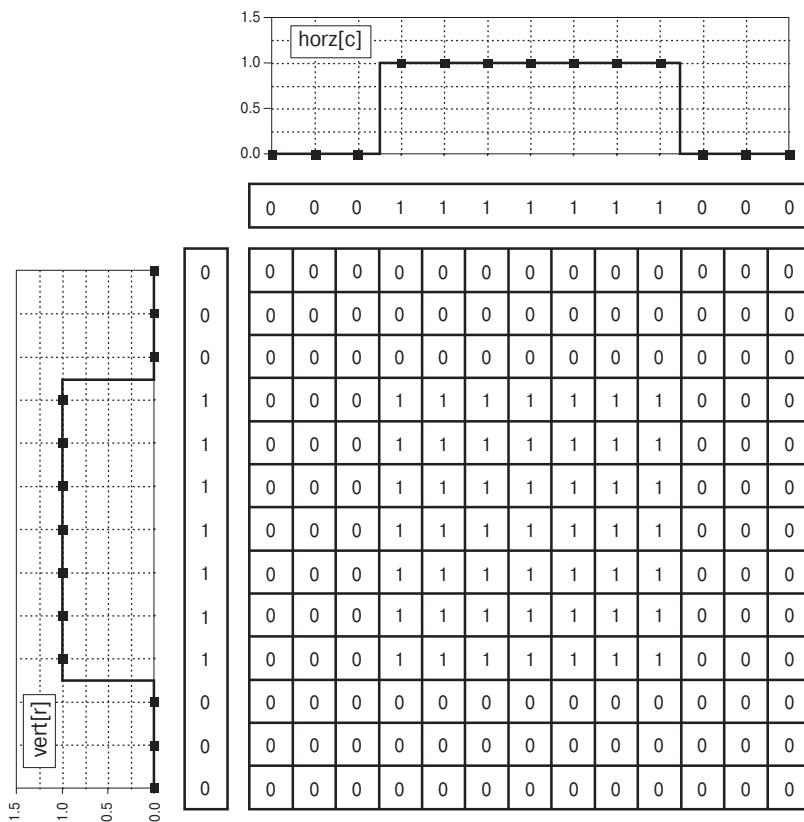
Свёртка при выполнении условия сепарабельности может повысить скорость вычисления в тех случаях, когда ФРТ системы является сепарабельной. Говорят, что ФРТ *сепарабельна*, если её можно разбить на два одномерных сигналов: вертикальную и горизонтальную проекции. Рис. 24.5 показывает пример сепарабельного изображения — квадратной ФРТ. Отметим, что значение каждого пикселя исходного изображения равно произведению точки горизонтальной проекции и соответствующей ей точки вертикальной проекции. В математической форме:

$$x[r, c] = \text{vert}[r] \times \text{horz}[c], \quad (24.1)$$

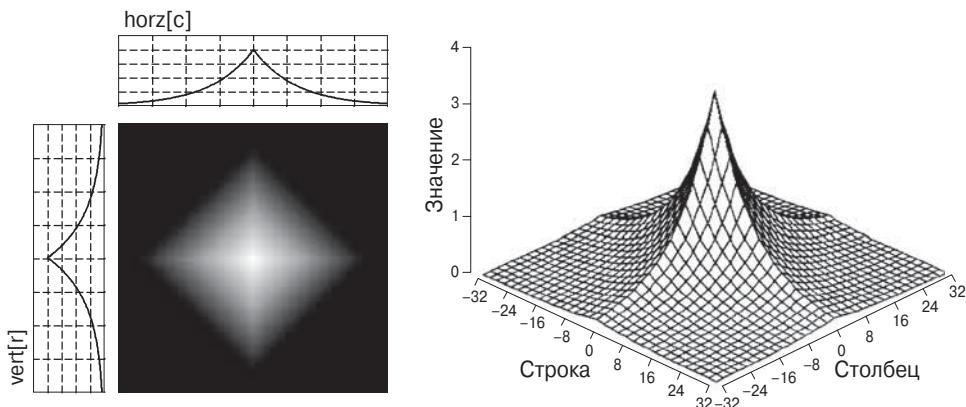
где  $x[r, c]$  — двумерное изображение, а  $\text{vert}[r]$  и  $\text{horz}[c]$  — одномерные проекции.

Разбиение изображения на две проекции. Изображение является сепарабельным, если оно может быть разложено на горизонтальную и вертикальную проекции.

Большинство изображений не удовлетворяют требованию сепарабельности. Например, круглая ФРТ не сепарабельна. В то же время сепарабельных изображений существует бесконечное множество. Такое множество можно, например, получить, генерируя произвольные горизонтальные и вертикальные проекции и находя соответствующие им изображения. Например, Рис. 24.6 иллюстрирует



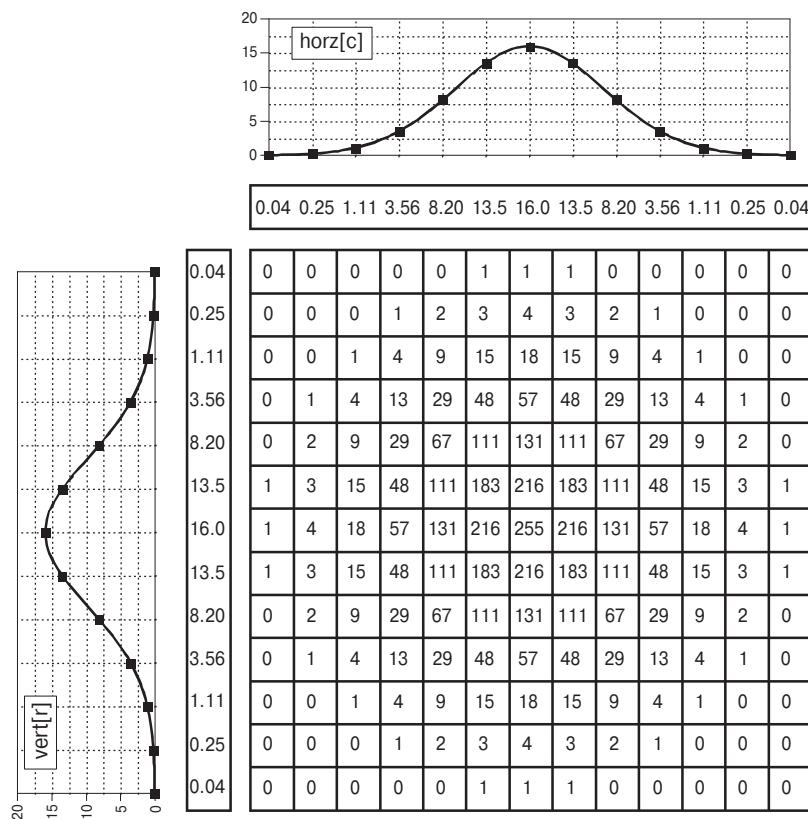
**Рис. 24.5.** Разбиение прямоугольной ФРТ на две проекции. Говорят, что ФРТ сепарабельна, если её можно разложить на горизонтальную и вертикальную проекции. Сепарабельные ФРТ важны, так как они позволяют повысить скорость вычисления свёртки.



**Рис. 24.6.** Формирование сепарабельной ФРТ. Бесконечное число сепарабельных ФРТ можно сгенерировать, произвольно задав их проекции и вычислив затем соответствующую им двумерную функцию. В этом примере в качестве проекций взяты двусторонние экспоненты, формирующие ФРТ в виде ромба.

этот подход при использовании в качестве проекций двусторонние экспоненты. Изображение, которое соответствует таким проекциям, найдено на основе выражения (4.1). При отображении подобное изображение выглядит как ромб, грани которого подчиняются экспоненциальному закону, убывая по модулю при удалении от начала координат.

В большинстве задач обработки изображений идеальной является *ФРТ, обладающая свойством круговой симметрии*, такая как круглая ФРТ. Хотя оцифрованное изображение обычно хранится и обрабатывается в прямоугольной системе строк и столбцов, желательно модифицировать изображение одинаково по всем направлениям. Встаёт вопрос: существует ли ФРТ, обладающая свойствами и круговой симметрии, и сепарабельности? Ответ — да, и только одна — гауссиан. Как показано на Рис. 24.7, двумерное гауссово изображение имеет своими проекциями также гауссовые функции. Отметим, что гауссовые изображения и их проекции имеют одинаковые СКО.



**Рис. 24.7.** Разбиение на проекции гауссиана. Гауссиан — это единственная ФРТ, которая обладает свойствами и круговой симметрии, и сепарабельности. Это делает его наиболее часто используемым ядром фильтра при обработке изображений.

Чтобы свернуть изображение с ядром сепарабельного фильтра, необходимо осуществить свёртку каждой строки изображения с горизонтальной проекцией, получая промежуточное изображение. Затем требуется свернуть каждый столбец этого промежуточного изображения с вертикальной проекцией ФРТ. Результиру-

ющее изображение будет идентично прямой свёртке исходного изображения и ядра фильтра. Если вам нравится сначала использовать столбцы, а затем строки — пожалуйста, результат от этого не изменится.

Свёртка изображения размером  $N \times N$  пикселей с ядром фильтра размером  $M \times M$  пикселей требует времени вычислений, пропорционального  $N^2 M^2$ . Другими словами, каждый пиксель выходного изображения зависит от всех пикселей ядра фильтра. Для сравнения свёртка при сепарабельности требует времени вычислений, пропорционального только  $N^2 M$ . Для ядер фильтров, которые имеют сотни пикселей в ширину, этот метод будет сокращать время вычислений в сотни раз.

При этом если вы будете использовать квадратную ФРТ (**Рис. 24.5**) или двустороннюю экспоненту (**Рис. 24.6**), то вычисления будут ещё более эффективны. Это происходит потому, что одномерные свёртки — это фактически *фильтр скользящего среднего* (см. Главу 15) и *двунаправленный однополосный фильтр* (см. Главу 19) соответственно. Оба этих одномерных фильтра могут быть быстро выполнены рекурсией. В результате время свёрток становится пропорционально только  $N^2$  и совершенно не зависит от размера ФРТ. Другими словами, свёртка изображения  $512 \times 512$  пикселей требует только нескольких сот миллисекунд на персональном компьютере. Это быстро! Вам не нравятся формы этих двух ядер фильтров? Сверните изображение с одним из них несколько раз, чтобы аппроксимировать ФРТ в форме гауссиана (это гарантируется центральной предельной теоремой, см. Главу 7). Описанные методы просто великолепны! Они стоят того, чтобы о них знать.

## 24.4. ФРТ большой размерности. Выравнивание освещённости

Наиболее часто ФРТ большой размерности требуется в задачах улучшения изображений с неравномерной освещённостью. Свёртка при выполнении условия сепарабельности — идеальный алгоритм для решения поставленной задачи. Как правило, изображения, видимые глазом, формируются отражённым светом. То есть видимое изображение условно можно считать равным произведению отражательной способности объекта на «сигнал» освещённости. **Рис. 24.8** показывает, как это работает. На **(а)** проиллюстрирована отражательная способность наблюдаемой сцены: в данном случае ряда светлых и тёмных полос. На **(б)** показан сигнал освещённости: рисунок света, падающего на сцену. Обычно освещённость слабо меняется в пределах рассматриваемой области. Изображение на **(в)** — это изображение, видимое глазом, равное отражательной способности объекта **(а)**, умноженной на освещённость сцены **(б)**. Области плохой освещённости трудно увидеть на **(в)**: они слишком тёмные, и их контрастность очень низка.

Проведём аналогию со зрением человека. Представьте, что вы рассматриваете двух одинаково одетых людей. Один из них стоит на ярком солнечном свете, а другой — в тени дерева. Процент отражения падающего света для обоих людей одинаков. Например, их лица могут отражать 80% падающего света, их серые рубашки — 40%, а тёмные брюки — 5%. При этом освещённость различается, скажем, в 10 раз. Это делает изображение человека в тени в 10 раз темнее, чем человека на солнце, и контрастность (различия между лицом, рубашкой и брюками) в 10 раз меньше.

Цель обработки изображений — выровнять сигналы освещённости для разных областей рассматриваемой сцены. Другими словами, мы хотим, чтобы обработанное изображение представляло отражательную способность объекта, а не условия его освещённости. Это задача нелинейной фильтрации, так как исходное изображение получено с помощью операции умножения. Добиться идеального результата невозможно, но качество изображения может быть существенно улучшено.

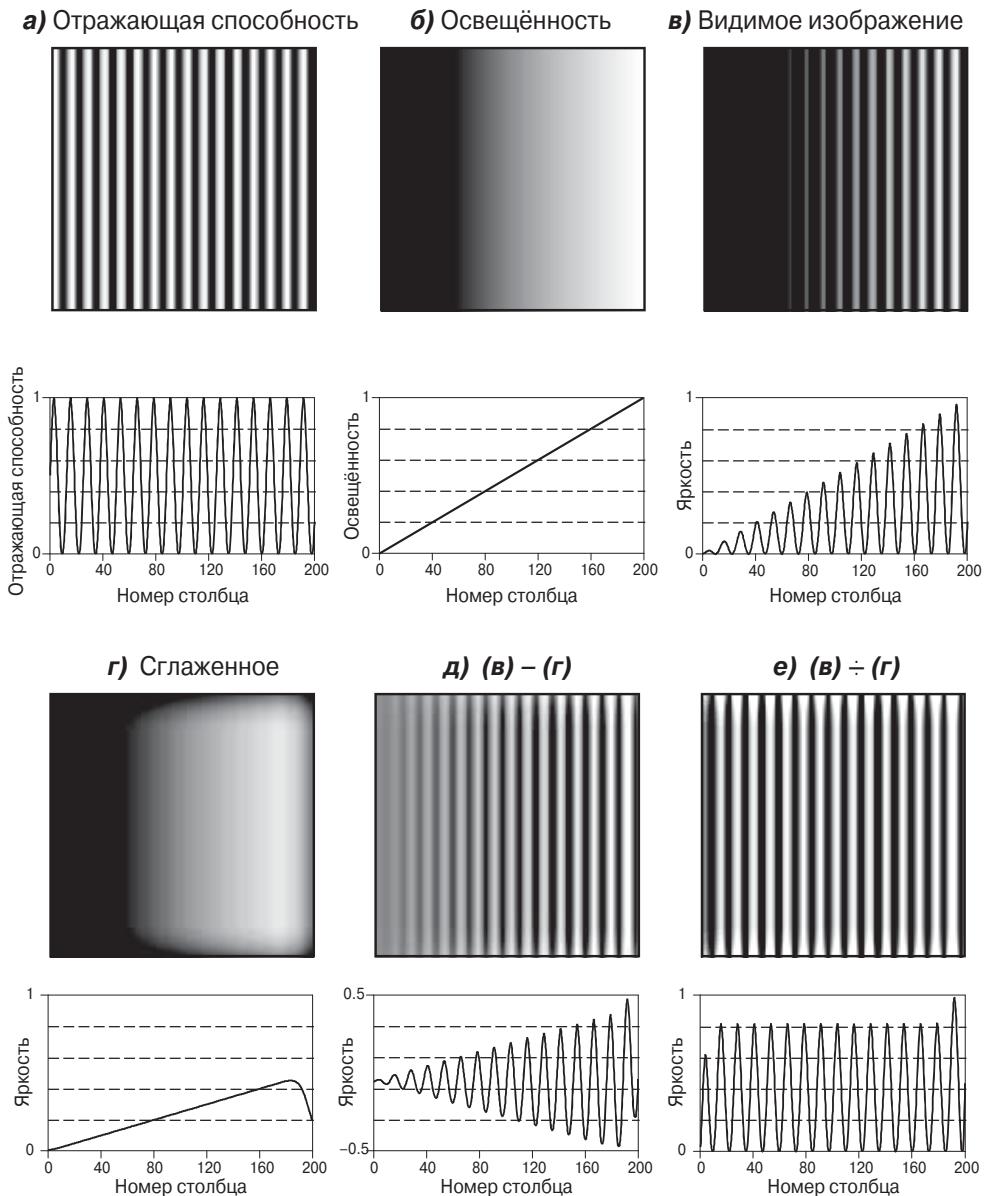
Для начала мы свернём изображение (e) с ФРТ большой размерности, равной одной пятой размера всего изображения. Цель — уменьшить острые детали на (в), чтобы приблизиться к первоначальному сигналу освещённости (б). На этом этапе можно использовать свёртку при условии сепарабельности. Точная форма ФРТ не важна. Важно только то, что она много шире, чем детали в отражательной способности. Результат использования гауссовского ядра фильтра проиллюстрирован на (г).

Так как сглаживающий фильтр даёт оценку освещённости, то, чтобы найти отражательную способность, мы будем использовать выходной улучшающий фильтр. То есть изображение (г) должно быть свёрнуто с ядром фильтра, равным разности единичного импульса и гауссиана. Для сокращения вычислительных затрат это может быть выполнено вычитанием сглаженного изображения (г) из первоначального изображения (в). Результат показан на (д). Подход оказался неработоспособен! Несмотря на то, что тёмные области стали «правильно» освещены, контрастность по-прежнему просто ужасна.

Линейная фильтрация в этом примере не дала результата, потому что сигналы отражательной способности и освещённости были объединены умножением, а не сложением. Линейная обработка не может разделить сигналы, объединённые нелинейной операцией. Необходимо использовать «антиумножение». Сигналы могут быть *разделены* сглаживанием изображения, как показано на (е). Откорректирована яркость и восстановлена до приемлемого уровня контрастность.

Такая процедура разделения тесно связана с *гомоморфной обработкой*, ранее описанной в Главе 22. Гомоморфная обработка — это способ обработки сигналов, объединённых нелинейной операцией. Основной принцип состоит в том, чтобы превратить нелинейную задачу в линейную выполнением соответствующей математической операции. Когда два сигнала объединены умножением, гомоморфная обработка начинается со взятия логарифма принятого сигнала. Учитывая выражение  $\log(a \times b) = \log(a) + \log(b)$ , задача разделения перемноженных сигналов превращается в задачу разделения сложенных сигналов. Например, после взятия логарифма изображения (в) можно использовать линейный высокочастотный фильтр, чтобы выделить логарифм отражательной способности. Как и прежде, самый быстрый способ выполнить высокочастотную фильтрацию — вычесть сглаженную версию изображения. Затем используется «антилогарифм» (экспонента), чтобы избавиться от логарифма, получая в результате искомую аппроксимацию отражательной способности.

Какой способ лучше: деление или гомоморфная обработка? Они почти одинаковы, так как взятие логарифма и вычитание фактически эквивалентны делению. Единственное различие — используемая аппроксимация отражательной способности. Один метод использует сглаженную версию полученного изображения, а другой — сглаженную версию логарифма полученного изображения.



**Рис. 24.8.** Модель построения изображения. Видимое изображение (**в**) является результатом перемножения «сигнала» освещённости (**б**) и отражательной способности (**а**). Цель обработки изображения — модифицировать (**в**), чтобы сделать его более похожим на (**а**). Это проиллюстрировано на (**г**, **д** и **е**). На (**г**) показана сглаженная версия (**в**), используемая как аппроксимация сигнала освещённости. На (**д**) проиллюстрирована аппроксимация отражательной способности, созданная вычитанием сглаженного изображения из видимого изображения. Наилучшая аппроксимация, показанная на (**е**), получена нелинейным процессом деления двух изображений.

Подобные методы выравнивания освещённости используются и в структуре глаза человека. Обработка в среднем слое сетчатки ранее описывалась как подчёркивание границ или высокочастотная фильтрация. Но это не все возможности глаза. Первый слой глаза выполняет нелинейную обработку и берёт логарифм видимого изображения. То есть обработку изображения глазом человека можно считать гомоморфным процессом. Как описано выше, за взятием логарифма следует линейный фильтр подчёркивания границ, выравнивающий освещённость, позволяя глазу видеть при слабом освещении. Другое представляющее интерес использование гомоморфной обработки имеет место в фотографии. Плотность (тёмнота) негатива равна логарифму яркости в получаемой фотографии. Это значит, что любая манипуляция с негативом на стадии проявления является видом гомоморфной обработки.

Расскажем о некоторых нюансах. Как говорилось в Главе 6, свёртка  $N$ -точечного сигнала с  $M$ -точечным ядром фильтра даёт сигнал длиной  $(N + M - 1)$  отсчётов. Точно так же, когда изображение  $M \times M$  пикселей сворачивается с ядром фильтра  $N \times N$  пикселей, результатом является изображение размером  $(N + M - 1) \times (N + M - 1)$  пикселей. Иногда такое резкое увеличение размера изображения оказывается проблемой. Должна измениться отведённая память; необходимо отрегулировать видеодисплей; изменить индексацию массива и т. п. Наиболее простой путь — игнорировать всё это. Если исходное изображение имело размер  $512 \times 512$  пикселей, то обычно и в конечном изображении нас интересуют  $512 \times 512$  пикселей. «Лишние» точки отбрасываются.

Так мы сохраняем размер изображения тем же, но всей проблемы это не решает. По-прежнему имеются *ограничивающие условия* для свёртки. Попытаемся, например, вычислить пиксель в верхнем правом углу (г). Это делается центрированием гауссовой ФРТ вокруг интересующей нас точки (в). Каждый пиксель изображения (в) далее умножается на соответствующий пиксель ФРТ, и их произведения складываются. Проблема в том, что три четверти ФРТ лежит при этом вне изображения. Самое простое — присвоить неопределённым пикселям нулевые значения. При этом на результирующем изображении (г) наблюдается по периметру тёмная полоса: значение яркости вне изображения плавно уменьшается до нуля.

К счастью, эту тёмную полосу можно скорректировать (хотя это не сделано в рассматриваемом примере). Это делается делением каждого пикселя изображения (г) на корректирующий коэффициент. Корректирующий коэффициент оказывается равен части ФРТ, которая перекрывалась с входным изображением, когда вычислялся пиксель. Чтобы скорректировать пиксели изображения (г), определим, в каком положении центрирована ФРТ на изображении (в). Например, верхний правый пиксель в (в) получился на основе только 25% размера ФРТ. Следовательно, корректировать этот пиксель на (г) можно делением на коэффициент 0.25. В результате такой коррекции пиксели в центре (г) не будут изменяться, а «проявятся» тёмные пиксели по периметру изображения. Поясним выбор коэффициентов коррекции следующим образом. Представим себе свёртку ядра фильтра с изображением, имеющим значения всех пикселей, равные единице. Тогда пиксели в результирующем изображении будут представлять собой коэффициенты коррекции, которые мы хотим определить.

## 24.5. Фурье-анализ изображений

*Анализ Фурье* используется в обработке изображений приблизительно так же, как и при обработке одномерных сигналов. Изображения, однако, не несут информации, закодированной в частотной области, что делает этот метод менее полезным. Например, когда преобразование Фурье выполняется для аудиосигнала, малопонятная во временной области форма сигнала преобразуется в удобный для восприятия и анализа частотный спектр. Для сравнения: выполнение преобразования Фурье изображений преобразует упорядоченную в пространственной области информацию в беспорядочную форму в частотной области. То есть не следует ожидать, что преобразование Фурье может помочь проанализировать информацию, заложенную в изображении.

Точно так же частотная область не годится для проектирования фильтров. Основная информация в изображениях заложена в границах — линиях, отделяющих одни объекты или области от других. Границы включают множество частотных компонентов, и попытка модифицировать изображение, манипулируя частотным спектром, обречена на неудачу. Фильтры изображений обычно проектируются в пространственной области, где информация представлена в непосредственном виде. Рассуждения при этом ведутся в терминах *сглаживания* или *подчёркивания границ* в пространственной области, а не в терминах *высокочастотной* или *низкочастотной* фильтрации, как в частотной области.

Тем не менее анализ Фурье обладает несколькими полезными свойствами при обработке изображений. Например, свёртка в пространственной области, так же как и свёртка во временной области, может быть сведена к умножению в частотной области. А умножение — гораздо более простая математическая операция по сравнению со свёрткой. Как и в случае одномерных сигналов, переход в частотную область позволяет быстро выполнить свёртку или процедуру, обратную свёртке (деконволюцию). Другое полезное свойство частотной области связано с *теоремой среза Фурье*, позволяющей связать изображение с его проекциями (тем, как изображение видится с разных сторон). Это в свою очередь положено в основу компьютерной томографии — метода построения изображения при облучении рентгеновскими лучами, широко используемого в медицинской технике и в промышленности.

Частотный спектр изображения может быть вычислен несколькими способами, но на практике обычно применяют БПФ. Анализируемое изображение должно состоять из  $N$  строк и  $N$  столбцов, где  $N$  — степень 2, т. е., например, 256, 512, 1024 и т. д. Если размер изображения не равен степени 2, то оно дополняется соответствующим числом пикселей с нулевыми значениями. Мы будем называть двумерный массив, который хранит значения пикселей изображения, *действительным массивом*. В дополнение нам понадобится другой массив такого же размера, который мы будем называть *мнимым массивом*.

Вычислить преобразование Фурье изображения достаточно просто. Нужно выполнить одномерное БПФ каждой строки, а затем одномерное БПФ каждого столбца. Обычно начинают с  $N$  значений в 0-й строке действительного массива. Действительная часть результата БПФ возвращается в 0-ю строку действительного массива, а мнимая часть записывается в 0-ю строку мнимого массива. Эта процедура повторяется для всех строк до  $(N - 1)$ -й включительно. В результате полу-

чаются действительный и мнимый массивы, которые содержат промежуточное изображение. Далее аналогичная процедура повторяется для всех столбцов промежуточных данных. Берутся  $N$  значений пикселей 0-го столбца действительного массива и мнимого массивов. Вычисляется БПФ. Действительная часть результа-та возвращается в 0-й столбец действительного массива, а мнимая часть записы-вается в 0-й столбец мнимого массива. После того как эта процедура повторится для всех столбцов по  $(N - 1)$ -й включительно, в действительном и мнимом массивах оказывается искомый частотный спектр изображения.

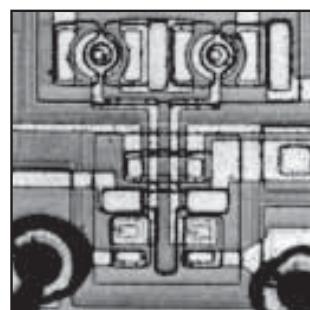
Вертикальное и горизонтальное направления по сути эквивалентны. Поэтому описанный алгоритм может выполняться сначала для столбцов, а потом для строк. Результат при этом не изменится. БПФ манипулирует данными таким об-разом, что в выходном массиве двумерного частотного спектра амплитуды низ-кочастотных компонентов будут располагаться по углам, а высокочастотных ком-понентов — в центре. Обратное преобразование Фурье изображения вычисляется взятием обратного БПФ всех строк и последующим вычислением обратного БПФ всех столбцов.

**Рис. 24.9** иллюстрирует пример преобразования Фурье изображения. Исход-ное изображение на **(а)** — это вид под микроскопом входного каскада интеграль-ной микросхемы операционного усилителя. На **(б)** показаны действительная и мнимая части частотного спектра этого изображения. Так как в частотной облас-ти могут присутствовать отрицательные значения пикселей, для отображения ин-формации все значения сдвигаются так, чтобы удовлетворять требованиям шка-лы серого. При этом отрицательные значения соответствуют тёмным пикселям, нулевые значения — серым, а положительные значения — светлым пикселям. Низкочастотные компоненты на изображении обычно много больше по амплиту-де, чем высокочастотные. Это приводит к расположению наиболее светлых и тёмных пикселей по углам изображения **(б)**. В отличие от частотных спектров, обычные изображения не имеют таких закономерностей и могут быть случайны-ми. Очевидно, однако, что изображение всегда можно придумать так, чтобы его спектр имел любой желаемый вид.

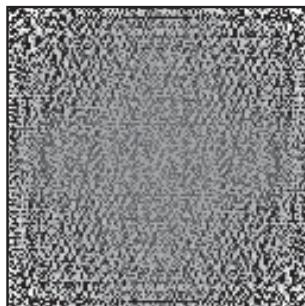
Как показано на **(в)**, полярная форма представления спектра изображения оказывается немного удобнее для восприятия. Низкие частоты имеют большие положительные значения амплитуды (белые углы), а высокие частоты — малые положительные значения (чёрный центр). Фазы выглядят одинаково на низких и высоких частотах и принимают случайные значения в диапазоне  $-\pi \dots \pi$  радиан.

На **(г)** показан альтернативный способ демонстрации спектра изображения. Так как мы рассматриваем дискретные в пространственной области изображе-ния, то в частотной области они являются периодическими. Другими словами, массивы, представляющие изображение в частотной области, периодически пов-торяются бесконечное число раз слева направо и сверху вниз. Представьте себе выложенную плиткой стену. Каждая плитка — это массив размером  $N \times N$  значе-ний, показанный на **(в)**. На **(г)** изображена секция воображаемой стены, включа-ющая четыре плитки. Центр изображения находится на стыке этих четырёх пли-ток. То есть **(г)** — это то же изображение, что и **(в)**, за исключением того, что оно сдвинуто на  $N/2$  пикселей по горизонтали (влево или вправо) и на  $N/2$  пикселей по вертикали (вверх или вниз) с периодическим повторением в частотной облас-ти. Яркие пиксели, находящиеся в четырёх углах на **(в)**, переносятся и оказыва-ются все вместе в центре **(г)**.

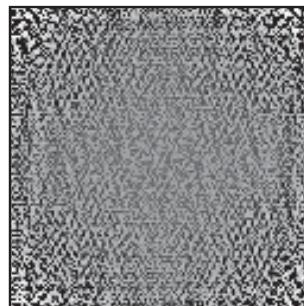
**Рис. 24.9.** Частотный спектр изображения. Изображение на (а) — это сделанная под микроскопом фотография поверхности интегральной микросхемы. Частотный спектр можно представить или действительной и мнимой частями, показанными на (б), или амплитудой и фазой, показанными на (в). На (б) и (в) низкие частоты отображены в углах, а высокие частоты — в центре. Благодаря периодичности в частотной области эта картина может быть заменена на прямо противоположную. Это показано на (г), где низкие частоты помещены в центр, а высокие — в углы.

**а)** Изображение**б)** Частотный спектр показан в прямоугольной форме

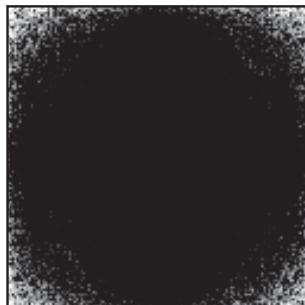
Действительная часть



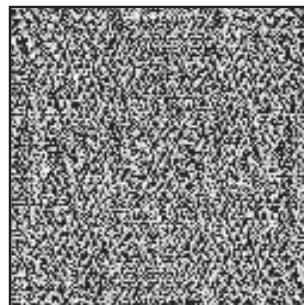
Мнимая часть

**в)** Частотный спектр показан в полярной форме

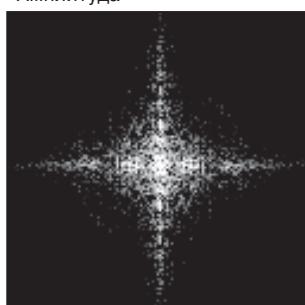
Амплитуда



Фаза

**г)** Частотный спектр показан в полярной форме и сдвинут так, что нулевая частота находится в центре

Амплитуда



Фаза



**Рис. 24.10** поясняет структуру двумерной частотной области (в случае размещения низких частот в углах). Стока  $N/2$  и столбец  $N/2$  разбивают частотный спектр на четыре квадрата. Если рассматривать действительную часть или амплитудную составляющую спектра, то верхний правый квадрат оказывается зеркальным отражением нижнего левого квадрата, а верхний левый квадрат — зеркальным отражением нижнего правого квадрата. Эта симметрия сохраняется и для мнимой части фазы спектра, за исключением того, что зеркально симметричные пиксели противоположны по знаку. Другими словами, каждая точка в частотном спектре имеет соответствующую точку, размещенную симметрично на другой стороне от центра (строка  $N/2$  и столбец  $N/2$ ). Одна из точек соответствует положительной частоте, а другая — отрицательной, — аналогично тому, что обсуждалось в Главе 10 для случая одномерных сигналов. В виде уравнения эта симметрия выражается следующим образом:

$$\begin{aligned} \operatorname{Re} X[r, c] &= \operatorname{Re} X[N - r, N - c], \\ \operatorname{Im} X[r, c] &= -\operatorname{Im} X[N - r, N - c]. \end{aligned} \quad (24.2)$$

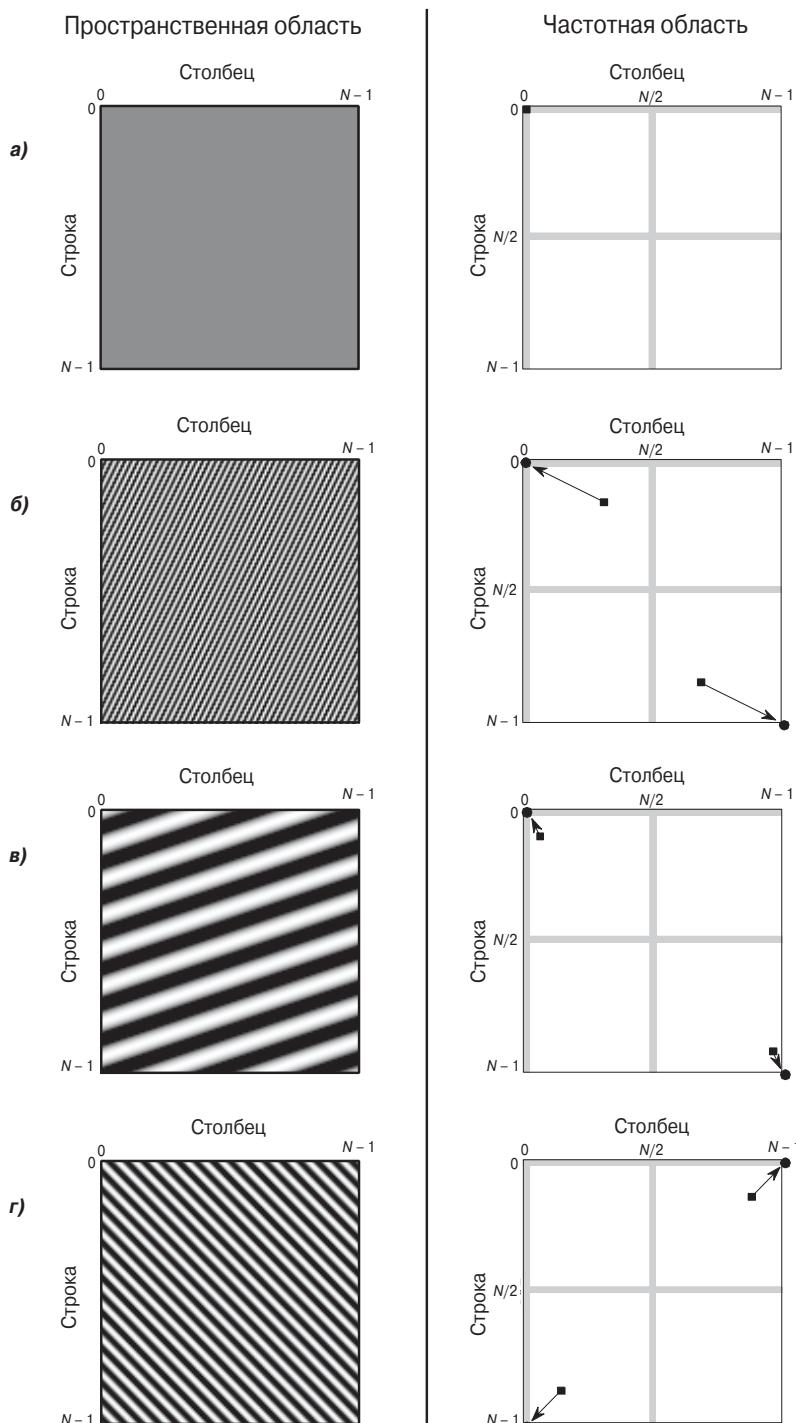
Симметрия изображения в частотной области. Эти уравнения остаются справедливыми и тогда, когда низкие частоты размещаются в углах, и тогда, когда их помещают в центр. В полярной форме представления амплитуда так же симметрична, как и действительная часть, а фаза — так же, как и мнимая часть.

Эти уравнения учитывают, что частотный спектр является периодическим с периодом  $N$  отсчетов с индексами  $0\dots(N-1)$ . Другими словами,  $X[r, N]$  должно интерпретироваться как  $X[r, 0]$ ;  $X[N, c]$  — как  $X[0, c]$ , а  $X[N, N]$  — как  $X[0, 0]$ . Эта симметрия делает четыре точки в спектре соответствующими самим себе:  $[0, 0]$ ,  $[0, N/2]$ ,  $[N/2, 0]$  и  $[N/2, N/2]$ .

Каждая пара точек в частотной области соответствует синусоиде в пространственной области. Как показано на **(а)**, значение в точке  $[0, 0]$  соответствует синусоиде с нулевой частотой в пространственной области, т. е. постоянной составляющей изображения. На рисунке присутствует только одна точка, потому что это один из тех случаев, когда точка соответствует самой себе. Как показано на **(б, в и г)**, другие пары точек соответствуют двумерным синусоидам, которые выглядят подобно волнам на поверхности воды. Одномерные синусоиды имеют частоту, фазу и амплитуду. Двумерные синусоиды имеют ещё и направление.

Частота и направление каждой синусоиды определяются местоположением пары точек в частотной области. Как показано на рисунке, нужно провести линию из каждой точки к местоположению точки нулевой частоты, находящейся в углу того квадрата, в котором она находится, т. е. к  $[0, 0]$ ,  $[0, N/2]$ ,  $[N/2, 0]$  или  $[N/2, N/2]$  (точки нулевых частот на этом рисунке отмечены кружками). Направление этой линии определяет направление пространственной синусоиды, а её длина пропорциональна частоте. В результате низкие частоты будут располагаться около углов, а высокие частоты — вблизи центра.

В случае, если спектр построен с нулевой частотой в центре (**Рис. 24.9г**), линии от каждой пары точек следует проводить к точке нулевой частоты в центре изображения, т. е. к  $[N/2, N/2]$ . Такая организация удобна и более проста для понимания, так как все линии проводятся к одной и той же точке. Другое достоинство размещения нуля в центре состоит в том, что это соответствует случаю спектра непрерывного изображения. Когда изображение в пространственной области непрерывно, в частотной области оно апериодично. Это помещает нулевую час-



**Рис. 24.10.** Двумерные синусоиды. Изображения гармонических колебаний имеют не только частоту, но и направление. На рисунке приведено четыре примера. Спектры показаны с низкими частотами по углам. Кружками отмечены точки нулевой частоты.

тоту в центр и заставляет всё более высокие частоты расходиться по всем направлениям до бесконечности. Как правило, в большинстве учебников, журнальных статей и различной документации частотный спектр дискретного изображения строится с нулевой частотой в центре. Однако в практических вычислениях, выполняемых с помощью компьютера, массивы данных хранятся в другом формате (низкие частоты в углах), потому что такой формат характерен для БПФ.

Даже при использовании БПФ при обработке изображений требуется достаточно много времени для вычисления преобразования Фурье. Например, преобразование Фурье изображения размером  $512 \times 512$  пикселей требует нескольких минут при вычислении на персональном компьютере. Это примерно в 10 000 раз медленнее, чем нужно для обработки изображения в реальном времени, т. е. со скоростью 30 кадров в секунду. Такое большое время вычислений является следствием огромного количества информации, содержащейся в изображении. В типовом случае одно изображение содержит такое же число пикселей, как количество слов в этой книге. Обработка изображений в частотной области может стать более популярной, когда компьютеры станут быстрее. Это технология двадцать первого века. Время бежит!

## 24.6. Быстрая свёртка

Несмотря на то, что преобразование Фурье требует большого времени выполнения, оно все же остаётся самым эффективным инструментом для повышения скорости свёртки изображения и ядра фильтра большой размерности. Например, свёртка изображения  $512 \times 512$  пикселей с ФРТ  $50 \times 50$  точек выполняется с использованием БПФ в 20 раз быстрее по сравнению с традиционной свёрткой. Глава 18 была посвящена обсуждению использования *быстрой свёртки* для одномерных сигналов. Здесь этот материал просто расширяется для случая двумерных сигналов.

Продемонстрируем, как работает быстрая свёртка на примере алгоритма поиска на изображении заданного фрагмента. Предположим, перед нами стоит задача построить систему проверки однодолларовых купюр, используемую для контроля качества печати, обнаружения подделок или проверки оплаты в торговых автоматах. Как показано на **Рис. 24.11**, изображение размером  $100 \times 100$  пикселей анализируемой купюры центрировано на портрете Джорджа Вашингтона. Целью является выявление наличия известного фрагмента — в данном примере изображения лица размером  $29 \times 29$  пикселей.

Задача формулируется следующим образом: имеется исследуемое изображение и известный эталонный фрагмент. Необходимо определить, присутствует ли на изображении искомый фрагмент и где он расположен. Какой метод является наиболее эффективным для решения такой задачи? Если вспомнить материал Главы 6, то можно легко ответить, что это корреляция, реализуемая с помощью согласованного фильтра на основе свёртки.

Прежде чем реализовать этот подход, нужно выполнить две модификации, превращающие искомый фрагмент в ФРТ. Это иллюстрируется на **Рис. 24.12**. На **(а)** показан эталонный фрагмент изображения, который мы пытаемся обнаружить. На **(б)** этот фрагмент повернут на  $180^\circ$ , т. е. развернут слева направо и сверху вниз. Как обсуждалось в Главе 7, когда корреляция реализуется с использованием свёр-



**Рис. 24.11.** Пример обнаружения заданного фрагмента на изображении. Задача состоит в исследовании портрета Джорджа Вашингтона размером  $100 \times 100$  пикселей (а) и поиска заданного фрагмента — изображения лица размером  $29 \times 29$  пикселей (б). Оптимальное решение — корреляция, которая может быть выполнена с помощью свёртки.

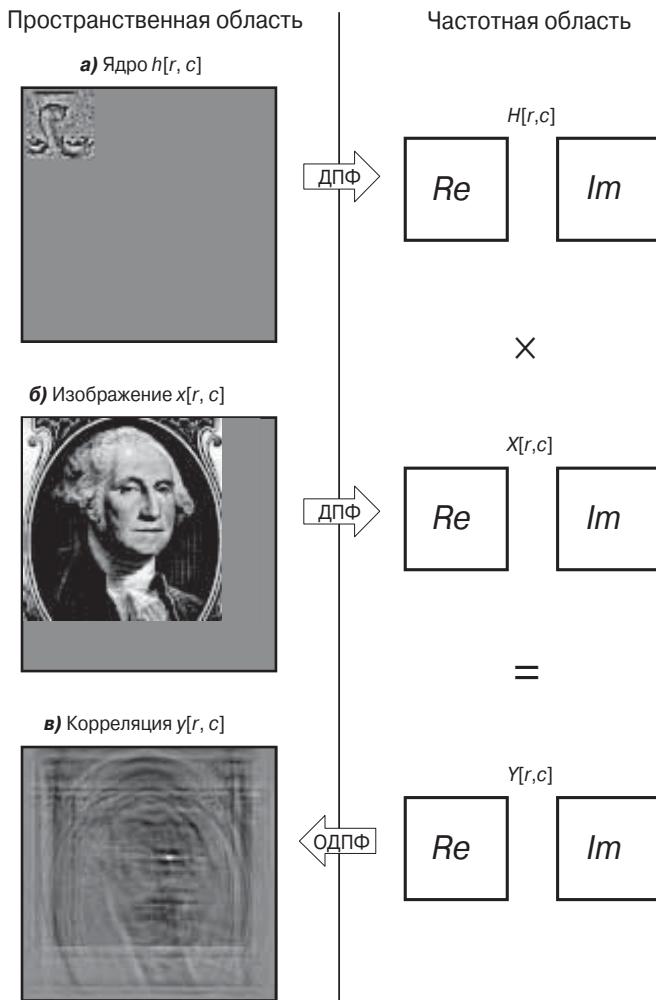


**Рис. 24.12.** Формирование ядра согласованного фильтра. Искомый фрагмент изображения показан на (а). На (б) он повернут на  $180^\circ$ , чтобы устранить поворот, присущий свёртке, позволяя с её помощью выполнить корреляцию. Применение фильтра подчёркивания границ даёт результат, показанный на (в). Это и есть ядро проектируемого фильтра.

тки, эталонный сигнал необходимо «перевернуть», чтобы скомпенсировать разворот, который происходит при свёртке. Мы ещё вернёмся к этому позже.

Вторая модификация служит для повышения эффективности алгоритма. Вместо того чтобы пытаться обнаружить в исходном изображении лицо Джорджа Вашингтона, лучше вести обнаружение границ лица среди границ объектов исходного изображения. Границы резче, чем детали, и корреляция в этом случае имеет более выраженный пик. Этот подход необязателен, но он делает результат значительно лучше. В простейшем случае перед корреляцией исходное изображение и эталонный фрагмент предварительно обрабатываются фильтром подчёркивания границ с ядром  $3 \times 3$  пикселя. Свойство ассоциативности, которым обладает свёртка, позволяет получить тот же результат, выполняя дважды фильтрацию подчёркивания границ для эталонного фрагмента и уйти от фильтрации самого изображения. Более того, на практике обычно оказывается достаточно однократной фильтрации. Таким образом, изображение (б) трансформируется в изображение (в), превращаясь в ФРТ, используемую при свёртке.

**Рис. 24.13** иллюстрирует детали быстрой свёртки. В этом примере изображение (а) сворачивается с изображением (б) и формируется изображение (в). Как



**Рис. 24.13.** Схема реализации быстрой свёртки изображений. Изображения на **(а** и **б**) переводятся в частотную область с помощью БПФ. Два полученных частотных спектра перемножаются, и результат с помощью обратного БПФ возвращается в пространственную область. В этом примере исходное изображение было предварительно обработано так, чтобы выполнить коррекцию через операцию свёртки.

были получены эти изображения и их предварительная обработка, здесь не рассматривается — нас интересует только схема выполнения свёртки. Первым шагом является дополнение обоих сворачиваемых изображений достаточным количеством нулей, делающим их размеры равными степени 2 и достаточными для формирования конечного изображения. Например, когда сворачиваются изображения размером  $100 \times 100$  и  $29 \times 29$  пикселей, результирующее изображение может быть  $128 \times 128$  пикселей. Следовательно, к изображениям (*а* и *б*) должно быть добавлено достаточно нулей, чтобы их размеры стали  $128 \times 128$  пикселей. Если этого не сделать, то произойдёт циклическая свёртка и конечное изображение окажется искажено. Если это непонятно, вернитесь к Главе 18, где более детально рассматривается одномерный случай.

Алгоритм БПФ используется для перевода изображений (*a* и *b*) в частотную область. В результате формируются четыре массива размером  $128 \times 128$  пикселей: действительные и мнимые части двух сворачиваемых изображений. Умножая действительную и мнимую части изображения (*a*) на действительную и мнимую части изображения (*b*), получаем действительную и мнимую части изображения (*c*) (если необходимо вспомнить, как это делается, посмотрите выражение (9.1)). Выполнение обратного БПФ завершает алгоритм быстрой свёртки.

Значение каждого пикселя в полученном изображении — это мера того, как сильно фрагмент исходного изображения, взятый в этой точке, согласуется с исключенным эталонным фрагментом. В рассматриваемом примере полученное изображение (*c*) состоит из шумоподобных данных и яркого пика, указывающего момент хорошего согласования фрагмента изображения и эталона. Положение этого пика (самого яркого пикселя) будет определять обнаруженные координаты лица. Если бы мы не использовали модификацию с подчёркиванием границ, то полученный пик был бы намного меньше выражен.

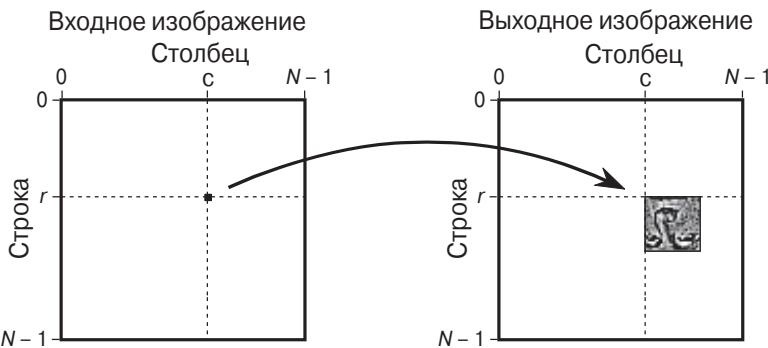
Хотя корреляция является мощным инструментом при обработке изображений, для неё характерен ряд существенных ограничений: эталонный фрагмент должен иметь точно такие же размеры и пространственную ориентацию, как и соответствующая область в исследуемом изображении. Шум и другие изменения амплитуды каждого пикселя не так важны, но очень критичным оказывается строгое пространственное согласование. Это, в частности, делает данный метод почти бесполезным для обнаружения военной техники по фотографиям, при диагностике опухолей по рентгеновским снимкам, поиска оружия при сканировании багажа в аэропорту. Одним из выходов является корреляция изображений несколько раз при различных формах и поворотах эталонного изображения. Это в принципе работает, но из-за существенного увеличения времени выполнения данный метод может оказаться неприемлемым.

## 24.7. Более внимательный взгляд на свёртку изображений

Давайте рассмотрим последний пример двумерной свёртки более детально. Как и в случае одномерных сигналов, *свёртку изображений* можно рассматривать как со стороны входа, так и со стороны выхода системы обработки. В Главе 6 говорилось, что описание свёртки со стороны входа удобно для анализа принципа работы свёртки, а описание со стороны выхода соответствует тому, как свёртка обычно записывается в математической и алгоритмической форме. Необходимо познакомиться с обеими этими формами представления свёртки.

**Рис. 24.14** иллюстрирует описание свёртки изображений со стороны входа. Каждый пиксель входного изображения формирует на выходе соответствующим образом смешённую и промасштабированную (умноженную на коэффициент пропорциональности) копию ФРТ. Выходное изображение вычисляется как сумма всех этих копий ФРТ для всех входных пикселей. На рисунке показано влияние на выходное изображение точки с координатами  $[r, c]$  входного изображения. ФРТ сдвинута так, что пиксель  $[0, 0]$  исходной ФРТ оказывается в положении  $[r, c]$  в выходном изображении. Если ФРТ определена только с положительными индексами,

как в этом примере, смещённая ФРТ может располагаться полностью ниже и правее точки  $[r, c]$ . Пусть вас не смущает, что лицо на этом рисунке перевёрнуто: оно и есть ФРТ, которую мы используем (**Рис. 24.13а**). При рассмотрении свёртки со стороны входа поворот ФРТ не производится; она только сдвигается.



**Рис. 24.14.** Рассмотрение свёртки изображений со стороны входа системы. Каждый пиксель входного изображения переносится в выходное изображение в виде промасштабированной и смещённой ФРТ. Выходное изображение — это сумма всех таких копий ФРТ. Изображение лица перевёрнуто на этой иллюстрации, потому что такой должна быть ФРТ, которую мы хотим использовать.

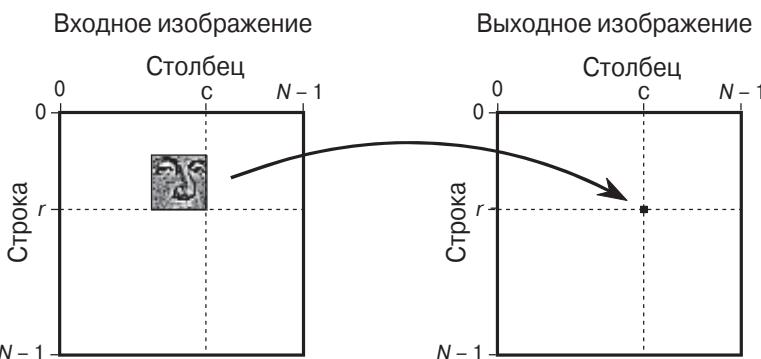
Рассмотрение свёртки изображений со стороны выхода иллюстрируется на **Рис. 24.15**. Каждый пиксель выходного изображения, как, например, показанная на рисунке точка  $[r, c]$ , формируется под воздействием многих пикселей входного изображения. Изображение ФРТ перевёрнуто на  $180^\circ$  вокруг пикселя  $[0, 0]$  и затем сдвинуто так, что пиксель  $[0, 0]$  выровнен с пикселям  $[r, c]$  входного изображения. Если ФРТ использует только положительные индексы, то она может оказаться вверху и слева от пикселя  $[r, c]$ . Значение пикселя в точке  $[r, c]$  в выходном изображении рассчитывается перемножением пикселей в повёрнутой ФРТ на соответствующие пиксели входного изображения и суммированием произведений. Эта процедура в виде уравнения записана в выражении (24.3). Соответствующие вычисления производят **Программа 24.1**.

$$y[r, c] = \sum_{k=0}^{M-1} \sum_{j=0}^{M-1} h[k, j] x[r - k, c - j] \quad (24.3)$$

Свёртка изображений. Изображения  $x[ , ]$  и  $h[ , ]$  сворачиваются, формируя изображение  $y[ , ]$ . Размер  $h[ , ]$  —  $M \times M$  пикселей с индексами  $0 \dots (M - 1)$ . В этом уравнении каждый пиксель выходного изображения  $y[ , ]$  вычисляется в соответствии с рассмотрением свёртки со стороны выхода. Индексы  $j$  и  $k$  используются в цикле по строкам и столбцам  $h[ , ]$  для вычисления суммы произведений.

Заметим, что в результате поворота ФРТ в процессе свёртки её предварительный поворот оказывается не нужен. Поэтому изображение лица в данном случае выглядит более привычно (**Рис. 24.15**), оно также сориентировано в пространстве, как и исследуемое входное изображение. Таким образом, мы успешно реализовали корреляцию с использованием свёртки. Сравните **Рис. 24.13** с **Рис. 24.15**. Яркая точка в выходном изображении означает, что цель обнаружена.

Быстрая свёртка обеспечивает точно такой же результат, как и обычная программа свёртки (**Программа 24.1**). Возникает вопрос: действительно ли сокраще-



**Рис. 24.15.** Рассмотрение свёртки изображений со стороны выхода системы. Каждый пиксель в выходном изображении равен сумме произведений пикселей входного изображения на соответствующие пиксели предварительно повёрнутой ФРТ.

ние времени выполнения за счёт использования БПФ окупает дополнительную сложность программы? Попробуем с этим разобраться. На **Рис. 24.16** проведено сравнение времени выполнения обычной свёртки, использующей числа в формате с плавающей точкой (FP), обычной свёртки, использующей целочисленную арифметику (INT), и быстрой свёртки, использующей формат с плавающей точкой (FFT). Данные представлены для изображений двух размерностей:  $512 \times 512$  и  $128 \times 128$  пикселей.

### Программа 24.1

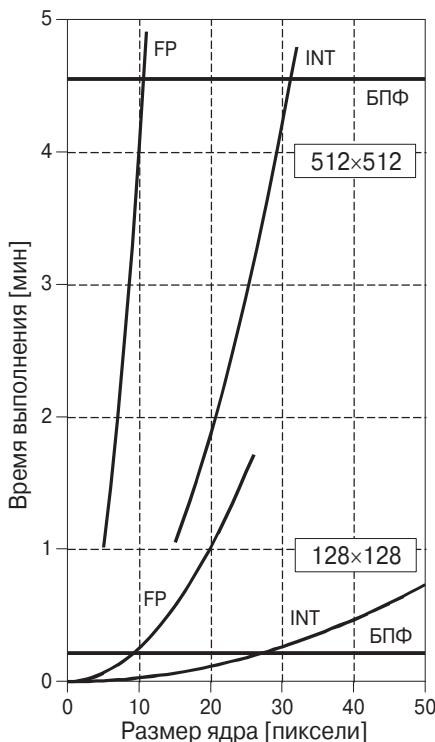
```

100 ОБЫЧНАЯ СВЁРТКА ИЗОБРАЖЕНИЙ
110 '
120 DIM X[99,99]      'хранит входное изображение, 100x100 пикселей
130 DIM H[28,28]       'хранит ядро фильтра, 29x29 пикселей
140 DIM Y[127,127]    'хранит выходное изображение, 128x128 пикселей
150 '
160 FOR R% = 0 TO 127  'цикл по каждой строке и столбцу в выходном
170 FOR C% = 0 TO 127  'изображении, вычисляющий значение пикселей по уравнению (24.3)
180
190 Y[R%,C%] = 0       'используется как аккумулятор
200 '
210 FOR J% = 0 TO 28   'умножает каждый пиксель ядра на соответствующий
220 FOR K% = 0 TO 28   'пиксель входного изображения и прибавляет к аккумулятору
230 Y[R%,C%] = Y[R%,C%] + H[J%,K%] * X[R%-J%,C%-K%]
240 NEXT K%
250 NEXT J%
260 '
270 NEXT C%
280 NEXT R%
290 '
300 END

```

Отметим, что время выполнения, требуемое для быстрой свёртки, не зависит от размера ядра, что показано на графике горизонтальной линией. На персональном компьютере Pentium 100 МГц изображение  $128 \times 128$  пикселей можно обрабо-

тать за 15 секунд, используя быструю свёртку, а изображение  $512 \times 512$  пикселей потребует более четырёх минут. Увеличение количества вычислений показывает, что время выполнения быстрой свёртки пропорционально  $N^2 \log_2(N)$  для изображения размером  $N \times N$  пикселей. То есть изображение  $512 \times 512$  требует примерно в 20 раз больше времени, чем изображение  $128 \times 128$  пикселей.



**Рис. 24.16.** Время выполнения свёртки изображений. На графике отражено время выполнения свёртки на процессоре Pentium 100 МГц с помощью трёх методов: обычной свёртки, выполняемой с числами в формате с плавающей точкой (FP), обычной свёртки, использующей целые числа (INT), и быстрой свёртки (БПФ), использующей плавающую точку. Для изображений  $512 \times 512$  и  $128 \times 128$  пикселей построены два набора кривых. Для свёртки, использующей БПФ, время зависит только от размера изображения и не зависит от размера ядра. Обычная же свёртка зависит и от размера изображения, и от размера ядра.

Обычная свёртка имеет время выполнения, пропорциональное  $N^2 M^2$  для изображения размером  $N \times N$  пикселей, сворачиваемого с ядром  $M \times M$ . Это можно понять, проанализировав текст **Программы 24.1**. Другими словами, время выполнения обычной свёртки очень сильно зависит от размера ядра. Как показано на графике, свёртка на основе БПФ гораздо быстрее обычной, использующей плавающую точку, если размер ядра превышает  $10 \times 10$  пикселей.

В большинстве случаев для обычной свёртки могут использоваться целые числа, при этом указанная граница возрастает до  $30 \times 30$  пикселей. График иллюстрирует, что этот предельный размер ядра слабо зависит от размера сворачиваемых изображений. Следует помнить, что быстрая свёртка полезна только при ядрах фильтра большой размерности.

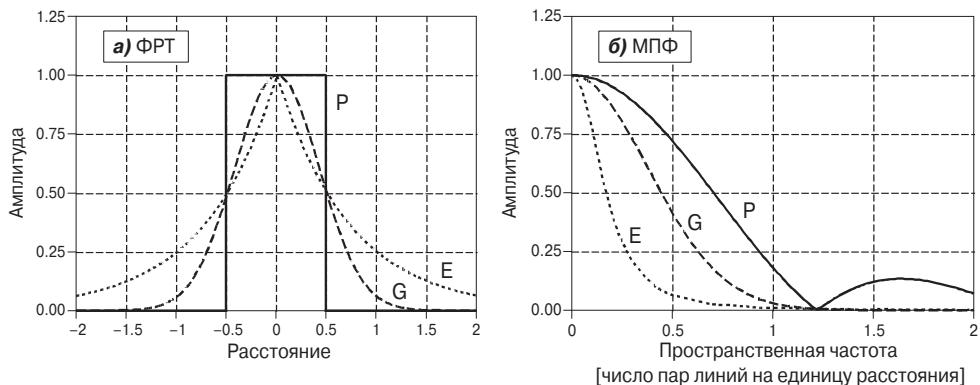
## ОСОБЫЕ МЕТОДЫ ОБРАБОТКИ ИЗОБРАЖЕНИЙ

В этой главе обсуждаются четыре специфических вопроса, связанных с обработкой изображений. Во-первых, рассматриваются способы определения пространственного разрешения, которые описывают минимальный размер объекта, видимый на изображении. Во-вторых, исследуется отношение сигнал/шум, определяющее, насколько нечётким или тусклым может быть объект, чтобы его ещё можно было различить. В-третьих, обсуждаются морфологические методы, использующие нелинейные операции для работы с бинарными изображениями (где каждый пиксель может быть либо белым, либо чёрным). В-четвёртых, описываются замечательные методы компьютерной томографии. Они произвели революцию в медицинской диагностике, обеспечив возможность получения детальных изображений внутренних органов человека.

### 25.1. Пространственное разрешение

Предположим, мы хотим сравнить две системы обработки изображений с целью определения, какая из них имеет наилучшее пространственное разрешение. Другими словами, мы хотим узнать, какая из систем способна обнаруживать более мелкие объекты. Для удобства мы хотим, чтобы в качестве характеристики системы по этому показателю выступало одно-единственное число. Это позволило бы очень легко сравнивать системы. К сожалению, один параметр не всегда достаточен для характеристики всех аспектов обработки изображений системой. Дело в том, что пространственное разрешение ограничивается двумя отдельными, но взаимосвязанными параметрами: *шагом выборки* и *размером апертуры выборки*. Далее мы рассмотрим два основных вопроса: как единственный параметр можно использовать для характеристики пространственного разрешения и каково соотношение между шагом выборки и размером апертуры выборки.

На Рис. 25.1 показаны три ФРТ с круговой симметрией: Круглая, гауссова и экспоненциальная. Эти три типа ФРТ являются показательными, так как они очень широко распространены в системах обработки изображений. Как было сказано в предыдущей главе, круглая ФРТ может быть результатом неточно сфокусированной системы линз; гауссиан формируется, когда объединяются случайные ошибки, например при наблюдении звёзд под влиянием турбулентности атмосферы; экспоненциальная ФРТ генерируется, когда электроны или рентгеновские лучи «бомбардируют» слой фосфора и превращаются в свет. Это явление используется в детекторах радиации, усилителях света в приборах ночного видения и электронно-лучевых трубках. Точная форма этих трёх ФРТ не важна для наших рассуждений; важно только то, что они представляют ФРТ, реально существующие в природе.



**Рис. 25.1.** Измерение пространственного разрешения на основе ФРТ и МПФ. На (а) показаны ФРТ трёх типов, общераспространённых в системах обработки изображений: кнопка (Р), гауссова (Г) и экспоненциальная (Е). Каждая из них на уровне половины от максимального значения имеет ширину, равную единице. Соответствующие МПФ показаны на (б). К сожалению, одинаковые значения ширины ФРТ на уровне 0.5 от максимума не означают, что и кривые МПФ окажутся одинаковыми.

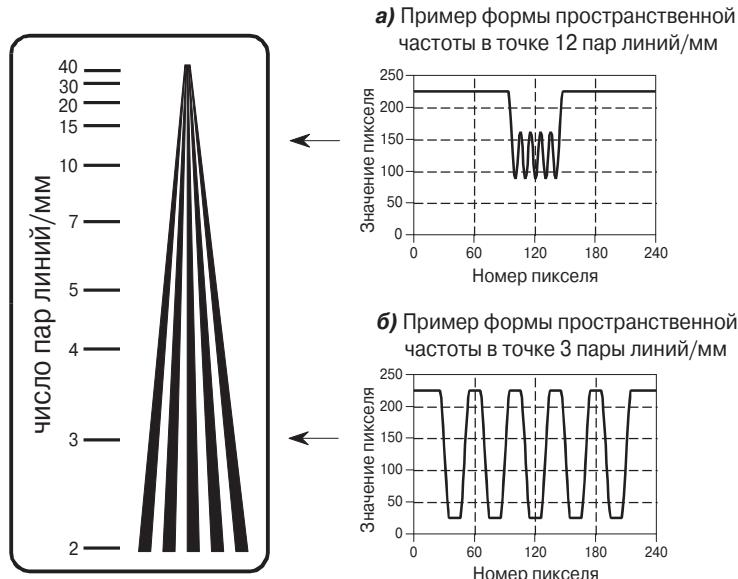
ФРТ содержит полную информацию о пространственном разрешении. Чтобы выразить пространственное разрешение единственным числом, мы можем игнорировать форму ФРТ и просто измерять её ширину. Наиболее распространённым является определение ширины ФРТ на уровне половины её максимального значения. Например, все ФРТ на Рис. 25.1а имеют единичную ширину.

К сожалению, этот метод имеет два существенных недостатка. Во-первых, он не вполне соответствует другим показателям пространственного разрешения, в том числе тому, как субъективно оценивает разрешение человек, рассматривающий некоторое изображение. Во-вторых, непосредственно измерить ФРТ обычно очень трудно. Представьте себе, что на вход системы обработки изображений подан дельта-импульс, т. е. изображение в виде одной очень маленькой белой точки на чёрном фоне. По определению, на выходе мы получим изображение, являющееся ФРТ системы. Проблема, однако, состоит в том, что полученная ФРТ будет содержать лишь несколько пикселей и её контрастность окажется очень низкой. Даже если вы будете очень аккуратны, случайные шумы сделают любые измерения ошибочными. Например, представим изображение-импульс как массив из  $512 \times 512$  точек, все из которых нулевые, кроме одной, имеющей значение 255. Теперь сравним его с обычным изображением, где все  $512 \times 512$  пикселей имеют среднее значение около 128. Получается, что сигнал изображения дельта-импульса примерно в 100 000 раз слабее, чем сигнал обычного изображения. Неудивительно, что отношение сигнал/шум будет очень низким и сигнала мы практически не увидим.

Один из основных постулатов этой книги гласит, что сигналы следует рассматривать в той области, в которой заложена переносимая ими информация. Например, звуковые сигналы должны рассматриваться в частотной области, а изображения — в пространственной. Несмотря на это, одним из методов измерения пространственного разрешения изображений является рассмотрение частотной характеристики. Это идёт вразрез с одной из базовых концепций книги, однако является общепринятым методом, с которым мы должны познакомиться.

Выполнение двумерного преобразования Фурье от ФРТ даёт двумерную частотную характеристику системы. Если ФРТ обладает свойством круговой симметрии, то и её частотная характеристика также будет обладать этим свойством. В этом случае полная информация о частотной характеристике содержится в её контуре. То есть после перехода в частотную область с помощью БПФ вся необходимая информация оказывается в нулевой строке столбца 0...( $N/2$ ). В обработке изображений эта частотная характеристика называется *модуляционной передаточной функцией* (МПФ). На (б) проиллюстрирован МПФ для трёх ФРТ, приведённых на (а). В случаях, когда ФРТ не обладает свойством круговой симметрии, информация содержится во всей двумерной частотной характеристике. При этом, однако, достаточно знать кривые МПФ в вертикальном и горизонтальном направлениях (т. е. столбцы 0...( $N/2$ ) в строке 0 и строки 0...( $N/2$ ) в столбце 0). Заметим, что эта процедура извлечения строки и столбца из двумерного частотного спектра не эквивалентна одномерному БПФ кривых ФРТ, показанных на (а). Мы ещё вернемся к этому вопросу. Как показано на Рис. 25.1, одинаковые значения ширины ФРТ на уровне половины максимума не означают, что и кривые МПФ окажутся одинаковыми.

На Рис. 25.2 показан так называемый *измеритель пространственного разрешения методом пар линий*. Принцип его действия основан на использовании МПФ. Такие измерители могут быть различных типов в зависимости от конкретного применения. Например, показанный на Рис. 25.2 измеритель с чередованием чёрных и белых линий можно непосредственно использовать для тестирования видеокамер. В случае применения измерителей для систем, использующих рент-



**Рис. 25.2.** Измеритель пространственного разрешения методом пар линий. Такой измеритель является инструментом, используемым для определения разрешающей способности систем обработки изображений. Последовательность чередующихся чёрных и белых линий создает континuum пространственных частот. Разрешение системы определяется как частота, на которой визуально перестают различаться отдельные линии. В приведённом примере измеритель увеличен в несколько раз, что видно по калибровочной шкале.

геновское излучение, чёрные линии могут быть выполнены в виде свинцовых рёбер, а в качестве белых линий может выступать некоторый пропускающий рентгеновские лучи материал. Ключевым моментом в структуре измерителя является то, что чёрные и белые линии к одному краю сближаются. При построении изображения чередующихся пар линий на одном краю линии будут сливаться вместе, а на другом краю они будут чётко разделяться. Где-то между двумя крайними случаями окажется точка, в которой линии будут минимально различимы. Найдя визуально эту точку, можно по калибровочной шкале определить соответствующее пространственное разрешение, обеспечивающее системой.

Необходимо представить себе, как происходит слияние отдельных линий в одно целое, чтобы понять ограниченность описанного метода измерения. На (**а** и **б**) показаны два примера формы кривых, соответствующих низкой и высокой пространственным частотам. Для низкой частоты (**б**) характерны плоские вершины, но колебания являются хорошо выраженным. На высокой пространственной частоте (**а**) амплитуда колебаний (глубина модуляции) существенно уменьшается. Это полностью отвечает формам кривых МПФ на Рис. 25.1б: более высокие пространственные частоты имеют меньшие амплитуды. Отдельные линии будут различаться на изображении до тех пор, пока амплитуда колебаний пространственной частоты принимает значения не менее 3...10% от начального уровня. Конкретное значение зависит от способности глаза различать максимумы и минимумы колебания с уменьшением их контрастности и в присутствии шумов.

Существенным достоинством использования измерителя пространственного разрешения методом пар линий является его простота и скорость. Основной недостаток — зависимость результата от возможностей глаза человека, т. е. субъективного фактора. Даже если удаётся полностью построить кривую МПФ, то в наиболее распространённом случае пространственное разрешение определяется по частоте, в которой уровень кривой спадает до 3, 5 или 10% от начального уровня. К сожалению, какое конкретно из этих значений использовано, указывается при описании характеристик системы далеко не всегда. В документации на свою продукцию изготовители часто пользуются расплывчатым термином «пределное разрешение». Им выгодно, чтобы в спецификациях их устройства выглядели максимально хорошо (вне зависимости от того, что эти устройства способны демонстрировать в действительности). Поэтому рекомендуем интерпретировать неопределённость в спецификациях как измерения разрешения по уровню 3% от максимума кривой МПФ.

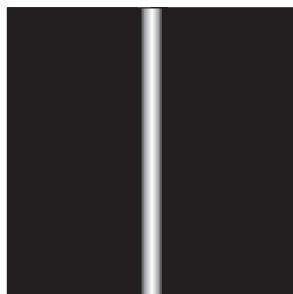
Здесь необходимо отметить один тонкий момент, связанный с тем, что МПФ определяется в терминах синусоидальных волн, а измерители на основе пар линий используют прямоугольные колебания. Линии измерителя, как светлые, так и тёмные, имеют равномерный тон по всей их ширине. Это сделано для удобства изготовления: было бы гораздо труднее сформировать линии, изменяющие свою яркость по синусоидальному закону. Каковы последствия замены синусоидальных волн прямоугольными? На высоких пространственных частотах все частоты, кроме основной, удаляются из прямоугольного колебания. В результате модулирующее колебание выглядит как синусоидальное, как показано на Рис. 25.2а. На низких частотах (Рис. 25.2б), колебание выглядит прямоугольным. Основная частота прямоугольного колебания имеет амплитуду, равную  $4/\pi = 1.27$  амплитуды самого

прямоугольного колебания (см. Рис. 13.10). Таким образом, измеритель на основе пар линий даёт несколько завышенную оценку разрешающей способности системы, так как в качестве максимального значения МПФ использует большую величину. Всё это интересно, но на практике почти всегда игнорируется.

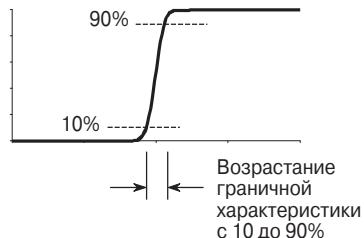
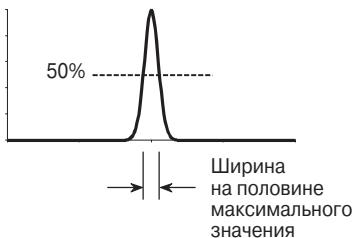
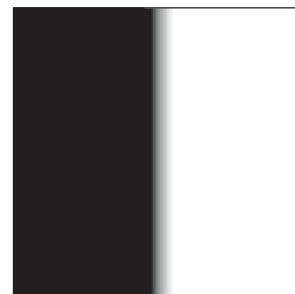
Так как для измерения МПФ наряду с синусоидальными используются прямоугольные волны, потребовалась специальная терминология. Вместо слова «период» применяют термин «пара линий» (т. е. чередование тёмной и светлой линий). Соответственно пространственная частота может измеряться значением, например, 25 пар линий на миллиметр, а не 25 периодов на миллиметр.

Определение разрешающей способности по ширине ФРТ плохо согласуется с субъективным восприятием изображений человеком и ставит проблемы, связанные с измерениями. Методы, основанные на МПФ, работают не в той области, в которой следовало бы проводить анализ влияния разрешающей способности на качество закодированной информации. Так имеется ли более удачный подход? Да, и основан он на использовании *функции расширения линии (ФРЛ)* и *граничной характеристики*. Как показано на Рис. 25.3, функция расширения линии — это реакция системы на входное изображение в виде тонкой линии, пересекающей изображение в вертикальном направлении. В свою очередь граничная характеристика — это реакция системы на ступенчатое изменение яркости изображения (границу). Так как линия — это производная (или первая разность) от границы, то ФРЛ — это производная (или первая разность) от граничной характеристики. Здесь используется измерение единственного параметра — расстояния, за которое кривая граничной характеристики возрастает с 10 до 90% своего максимума.

а) Функция расширения линии (ФРЛ)



б) Граничная характеристика

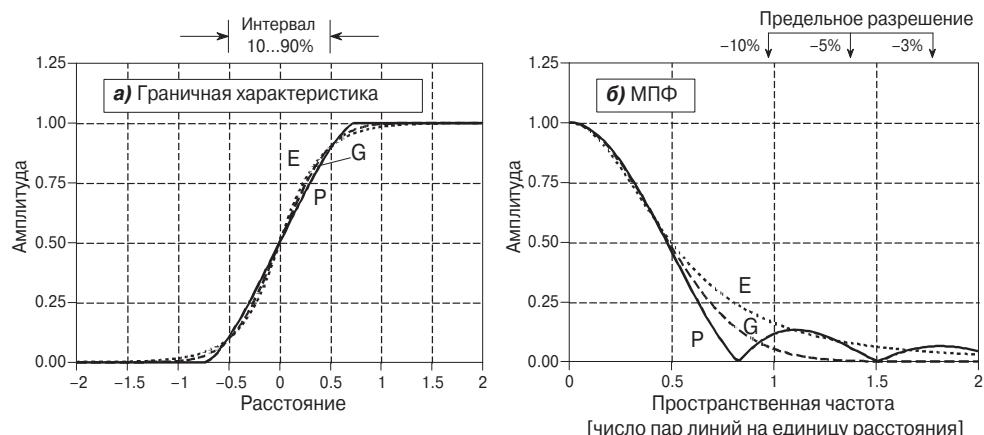


**Рис. 25.3.** Функция расширения линии и граничная характеристика. Функция расширения линии (ФРЛ) — это производная от граничной характеристики. Ширина ФРЛ обычно выражается как ширина на уровне половины максимального значения. Ширина граничной характеристики обычно принимается равной расстоянию, за которое кривая возрастает с 10 до 90% своего максимального значения.

В использовании граничной характеристики для измерения пространственного разрешения есть множество достоинств. Во-первых, измерение производится в той же области, в которой закодирована информация, т. е. в пространственной. Фактически основной причиной интереса к разрешающей способности системы является необходимость оценки чёткости границ на изображении. Второе преимущество состоит в том, что граничную характеристику достаточно просто измерить, ведь сформировать входное изображение-границу очень просто. При необходимости можно легко найти ФРЛ взятием первой разности от граничной характеристики.

Третьим преимуществом является то, что все используемые граничные характеристики имеют похожую форму, даже если им соответствуют абсолютно разные ФРП. Это проиллюстрировано на Рис. 25.4а, где изображены граничные характеристики систем с круглой гауссовой и экспоненциальной ФРП. Так как кривые граничных характеристик имеют схожие формы, то интервал возрастания этих кривых с уровня 10 до 90% оказывается отличным единым параметром, характеризующим разрешающую способность систем. Четвёртое преимущество состоит в том, что по ФРЛ с помощью вычисления одномерного БПФ можно непосредственно найти МПФ (подобно тому, как для вычисления МПФ по ФРП необходимо использовать двумерное преобразование Фурье). На (б) показаны МПФ, соответствующие граничным характеристикам, изображённым на (а). То есть кривые на (б) получаются из кривых на (а) вычислением первой разности (чтобы найти ФРЛ) и последующим БПФ.

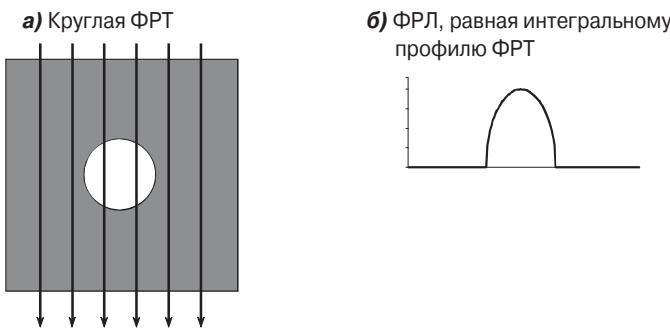
Пятым преимуществом является то, что подобным граничным характеристикам соответствуют подобные кривые МПФ, что проиллюстрировано на (а и б). Это позволяет легко переходить от одних характеристик к другим. Например, система, у которой интервал возрастания граничной характеристики с уровня 10 до 90% равен  $x$ , характеризуется предельным разрешением (измеряемым по уровню МПФ 10%), равным приблизительно 1 паре линий на  $x$ . При этом едини-



**Рис. 25.4.** Граничная характеристика и МПФ. На (а) показаны граничные характеристики, соответствующие трём ФРП: круглой (Р), гауссовой (G) и экспоненциальной (Е). Каждая характеристика имеет интервал возрастания с 10 до 90%, равный единице. На (б) показаны соответствующие кривые МПФ, которые совпадают выше 10-процентного уровня. Предельное разрешение является несколько неопределённым показателем, равным частоте, на которой МПФ имеет амплитуду 3...10%.

цы измерения этого интервала могут быть различны, в зависимости от конкретного типа системы. Пусть, например, имеются три системы обработки изображений с интервалом возрастания граничной характеристики с 10 до 90%, равным 0.05 мм, 0.2 мрад и 3.3 пикселя. 10-процентный уровень МПФ будет достигнут для таких систем на пространственных частотах 20 пар линий/мм, 5 пар линий/мрад и 0.33 пар линий/пиксель соответственно.

**Рис. 25.5** иллюстрирует математические соотношения между ФРТ и ФРЛ. На **(а)** показана круглая ФРТ: небольшой круг в центре, внутри которого все пиксели имеют значение 1 (показан белым цветом), окружённый со всех сторон пикселями со значением 0 (показаны серым цветом). Если взять только значения пикселей, расположенных вдоль одной прямой, проходящей через центр изображения (это так называемый профиль ФРТ), то мы получим прямоугольный импульс. На **(б)** показана соответствующая ФРЛ. Видно, что ФРЛ получается как *интегральный профиль ФРТ*, вычисляемый по следующей схеме. Следует «пробежать» в одном из направлений по пикселям изображения, расположенным вдоль лучей, как показано на **Рис. 25.5**. Каждое значение интегрального профиля ФРТ есть сумма значений пикселей, расположенных вдоль соответствующего луча.



**Рис. 25.5.** Связь между ФРТ и ФРЛ. Круглая ФРТ показана на **(а)**. Любая строка или столбец, включающие пиксели из белого круга, окажется прямоугольным импульсом. На **(б)** показана соответствующая ФРЛ, равная интегральному профилю ФРТ. Это означает, что ФРЛ (каждый её отсчёт) вычисляется «перемещением» по изображению в одном из направлений и суммированием (интегрированием) значений пикселей, расположенных вдоль каждого из лучей. Если перемещение осуществляется в направлении, показанном на рисунке, то интегрирование сводится к суммированию значений всех пикселей в каждом столбце.

В рассматриваемом примере лучи построены вертикально, в результате чего каждая точка интегрального профиля ФРТ находится суммированием всех значений пикселей в каждом столбце. Это соответствует ФРЛ для случая вертикальной линии, выбранной в качестве «входного» изображения. ФРЛ для случая горизонтальной линии будет находиться как сумма всех значений пикселей в каждой строке. Для непрерывных изображений ФРЛ вычисляется аналогично, но суммирование заменяется интегрированием.

В данном примере показано, что ФРЛ может быть непосредственно вычисляемо по ФРТ. Однако ФРТ не всегда можно найти по ФРЛ, т. к. ФРТ содержит информацию о пространственном разрешении во всех направлениях, а ФРЛ — только в одном. Система имеет только одну ФРТ, но бесконечное множество ФРЛ — по одной для каждого угла наклона прямой. Возьмём, например, систему,

имеющую ФРТ вытянутой вдоль горизонтальной оси формы. Пространственное разрешение такой системы окажется различным в вертикальном и горизонтальном направлениях, обуславливая различные ФРЛ в этих направлениях. Очевидно, что ФРЛ при одном угле наклона не несёт достаточно информации, чтобы вычислить ФРТ. Есть, однако, одно исключение: ФРТ, обладающая свойством круговой симметрии. Если же ФРТ не обладает этим свойством, то можно вычислить ФРТ на основе множества ФРЛ, измеренных под разными углами. Это, однако, очень запутанная процедура, требующая использования сложной математики. Задача вычисления ФРТ по большому числу измерений ФРЛ фактически аналогична проблеме формирования изображения в компьютерной томографии, о которой пойдёт речь ниже.

На практике ФРЛ и ФРТ не слишком сильно отличаются друг от друга для большинства систем, и часто можно встретить использование одной из них в качестве аппроксимации другой. Более того, существуют два распространённых случая, когда они оказываются идентичны: прямоугольной ФРТ соответствует прямоугольная ФРЛ (с той же шириной), и гауссовой ФРТ соответствует гауссова ФРЛ (с тем же СКО).

Обобщение сказанного выше даёт нам возможность ответить на два вопроса: как характеризует систему указанная в спецификации разрешающая способность и как измерить разрешающую способность, чтобы составить спецификацию самому? Предположим, вам на глаза попадается рекламный проспект, в котором утверждается: «Разрешающая способность нашей системы — 40 пар линий на миллиметр». Вы должны интерпретировать это следующим образом: «Амплитуда синусоиды с частотой 40 пар линий/мм уменьшится системой до 3...10% своего исходного значения и окажется еле заметной на изображении». Вы также можете вычислить в уме, что 40 пар линий/мм при использовании 10-процентного уровня означают, что интервал возрастания граничной характеристики от 10 до 90% составит  $1/40 = 0.025$  мм. Если же спецификация была рассчитана для 3-процентного уровня МПФ, то интервал возрастания граничной характеристики окажется в 1.5...2 раза больше.

Если вы проводите измерение пространственного разрешения собственной системы, то нужно выполнить те же шаги, но в обратном порядке. «Подайте» на вход системы изображение-границу и измерьте на выходе граничную характеристику. Интервал возрастания этой кривой с 10- до 90-процентного уровня — это лучший единий показатель, характеризующий разрешающую способность системы. Чтобы ваш шеф и работники рекламного отдела остались довольны, возьмите первую разность от граничной характеристики и найдите ФРЛ, а затем с помощью БПФ рассчитайте МПФ.

## 25.2. Шаг и апертура выборки

На Рис. 25.6 показаны два крайних случая дискретизации изображений, которые мы будем называть *идеальным детектором* и *смазывающим детектором*. Представьте, что на (а) изображена поверхность датчика изображений, например такого, как ПЗС. Свет, падающий в любую точку внутри одного из квадратов-пикселей, будет влиять только на значение этого пикселя и не будет влиять

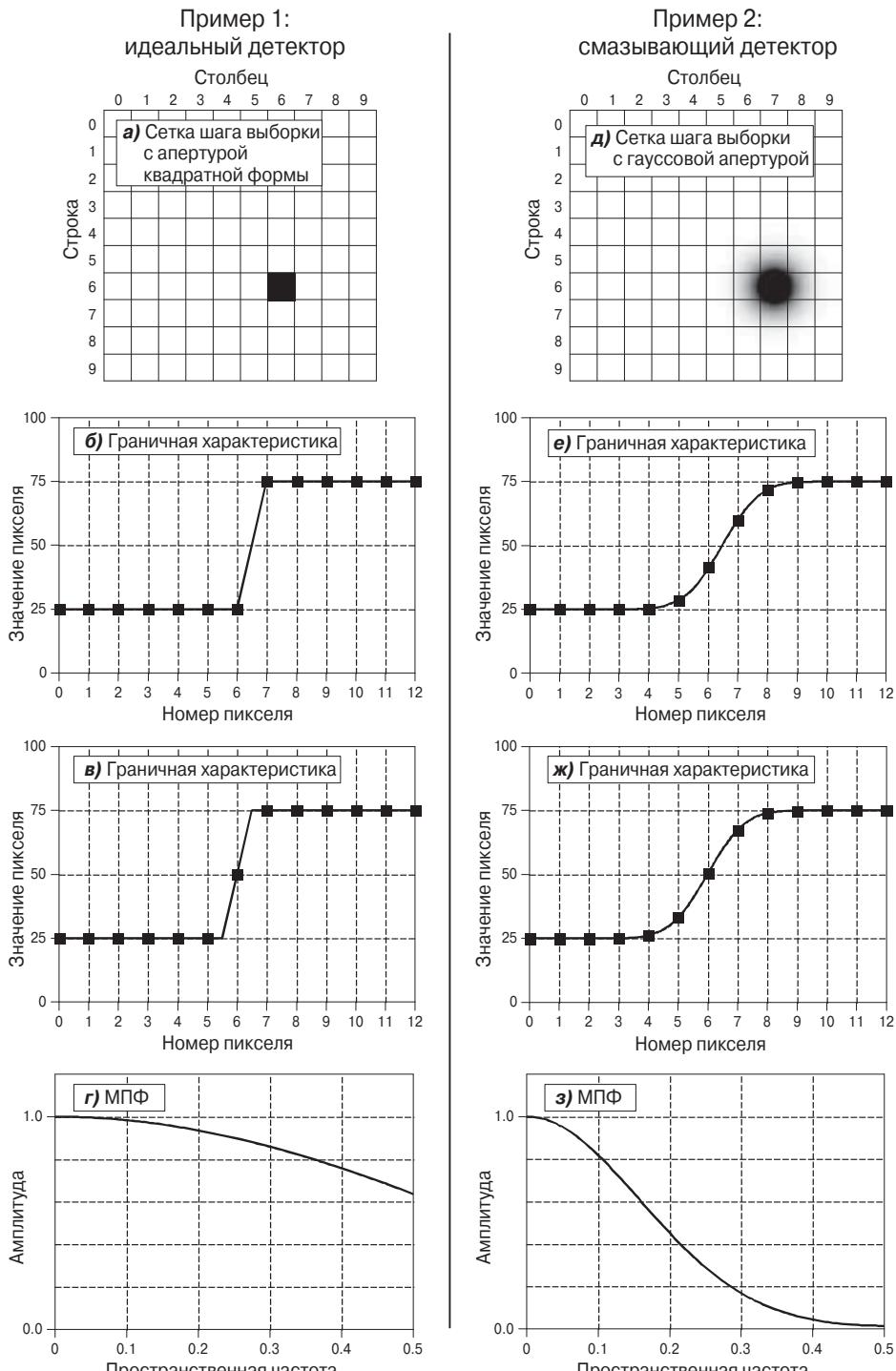
на другие. На рисунке это отражено точным совпадением квадрата-пикселя и апертуры выборки, окрашенной в чёрный цвет. Это оптимальная ситуация при дискретизации изображений: весь падающий свет участвует в формировании значения пикселя, и в то же время нет перекрытий или перекрёстных помех между соседними пикселями. Это ситуация, когда *апертура выборки* точно совпадает с *шагом выборки*.

Противоположная ситуация показана на (д). Апертура выборки значительно превышает шаг выборки и имеет гауссово распределение. В этом случае на значение каждого пикселя на выходе детектора изображений влияет свет, падающий и на соседние пиксели. Знакомая ситуация: она аналогична рассмотрению свёртки со стороны выхода. Ей соответствует рассмотрение со стороны входа: узкий луч света падает на детектор изображений и определяет значение нескольких соседних пикселей в соответствии с гауссовым распределением.

Теперь обратим наше внимание на граничные характеристики, соответствующие рассматриваемым примерам. Квадратиками на графиках отмечены действительные значения пикселей, которые могут быть найдены в регистрируемом изображении. Эти квадратики соединяются аппроксимирующей линией, представляющей исходную кривую, из которой были сделаны выборки. Важным здесь является то, что форма этой линии определяется только апертурой выборки. Это означает, что разрешающая способность в конечном изображении ограничивается двумя факторами. Во-первых, аппроксимирующая линия может иметь плохое разрешение из-за слишком большой апертуры выборки. Во-вторых, шаг выборки может оказаться слишком велик, из-за чего мелкие детали изображения, расположенные между моментами выборки, окажутся утеряны. На рисунке представлены два случая граничной характеристики для каждого из рассматриваемых примеров. Они показывают, что моменты выборки могут располагаться в произвольных точках вдоль кривой. При этом граница может точно совпадать с одним из пикселей, а может оказаться между двумя соседними пикселями. Обратите внимание, что в случае идеального детектора на интервал возрастания граничной характеристики в момент скачка будет находиться максимум один пиксель, а в случае смазывающего детектора — три или четыре.

Что ограничивает разрешающую способность в рассматриваемых примерах? Обратимся к *теореме отсчётов*. Как говорилось в Главе 3, дискретизация оставляет только частотные компоненты, лежащие ниже половины частоты дискретизации. Все другие частоты «заворачиваются» в эту область и отсутствуют в дискретном сигнале. Посмотрите теперь на кривую МПФ, показанную на (з). Апертура выборки смазывающего детектора удаляет все частотные компоненты, лежащие выше половины частоты дискретизации, и, следовательно, предотвращает потери информации, связанные с наложениями. Это означает, что разрешающая способность системы полностью определяется апертурой, а не шагом выборки. Апертура выборки фактически выступает в качестве *антиэлайзингового фильтра*, используемого для устранения эффекта наложений и соответствующих искажений информации.

По-другому ведёт себя кривая МПФ на (г). Она показывает, что в случае идеального детектора и апертура, и шаг выборки ограничивают пространственное разрешение системы. Спад кривой МПФ на высоких частотах соответствует потере информации, определяемой апертурой выборки, а то, что кривая МПФ не



**Рис. 25.6.** Два крайних случая дискретизации изображений — идеальный и смазывающий детекторы.

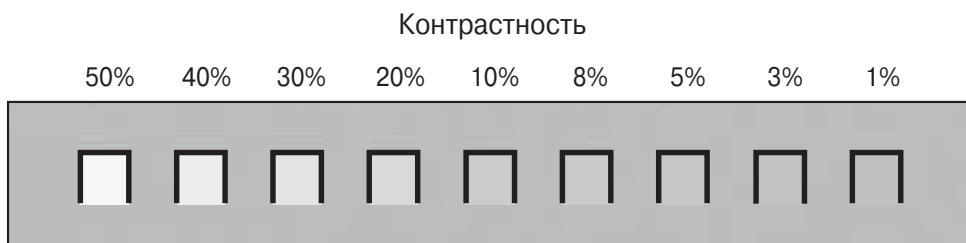
спадает до нуля к значению частоты, равному 0.5, вызывает потери информации в процессе дискретизации, обусловленные конечным значением шага выборки. Какой из факторов ограничивает пространственное разрешение в большей степени? На этот вопрос трудно ответить, т. к. численно оценить влияние данных факторов на разрешающую способность оказывается достаточно сложно, ведь они ухудшают качество изображений различными путями. Однако можно сказать, что разрешение в случае идеального детектора (**пример 1**) обычно ограничивается шагом выборки.

Хотя рассмотренные принципы могут казаться запутанными, на практике они сводятся к очень простым правилам. Рассмотрим систему с некоторым интервалом 10...90%-го возрастания граничной характеристики, равным, например, 1 мм. Если шаг выборки больше, чем 1 мм (на границу приходится менее одной выборки), разрешение будет ограничиваться шагом выборки. Если шаг выборки меньше 0.33 мм (на границу приходится более трёх выборок), разрешение будет ограничиваться апертурой выборки. Когда характеристика системы имеет 1...3 выборки на границу, разрешение ограничивается обоими факторами.

## 25.3. Отношение сигнал/шум

Отдельные объекты видны на изображении, потому что их яркость отличается от яркости других окружающих точек. Таким образом, контрастность объекта (будем говорить, что объект — это сигнал, т. е. часть изображения, несущая полезную информацию) должна быть высокой с учётом наличия шума. Здесь можно выделить два класса ограничений: ограничения зрительного восприятия и ограничения данных.

На **Рис. 25.7** проиллюстрирован способ определения возможностей глаза человека воспринимать слабые сигналы. В зависимости от условий наблюдения глаз способен различать объекты с контрастностью 0.5...5%, что соответствует способности различать от 20 до 200 оттенков серого (от абсолютно чёрного до абсолютно белого). Точные значения определить сложно, так как они зависят от различных факторов, таких как яркость окружающего освещения, расстояние между двумя сравниваемыми областями, тем, как изображение было получено (например, вывод на видеэкран, фотография и т. п.).

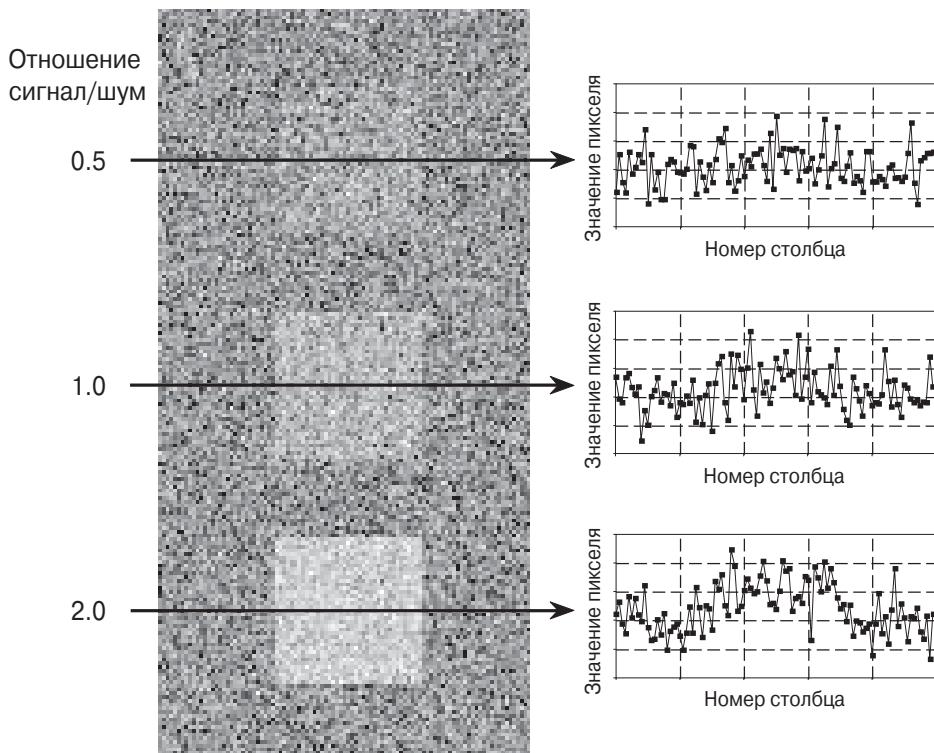


**Рис. 25.7.** Чувствительность глаза человека к контрастности объектов. Глаз способен обнаруживать объекты при минимальной контрастности 0.5...5% в зависимости от условий наблюдения. Значение контрастности 100% — это разница между абсолютно чёрным и абсолютно белым.

Преобразование шкалы серого, описанное в Главе 23, можно использовать для того, чтобы повысить контрастность выбранного диапазона значений пикселей. Это достаточно мощный инструмент, позволяющий преодолеть ограничения зрения человека. Контрастность объектов с одним уровнем яркости повышается за счёт уменьшения контрастности других уровней. Этот приём, однако, может работать только при условии, что объекты не теряются в шумах. Иначе ситуация окажется весьма серьёзной: в сигнале (изображении) не содержится достаточной информации, чтобы выявить объект вне зависимости от способностей глаза.

На Рис. 25.8 показано изображение, на котором размещены три квадрата с контрастностью 5, 10 и 20%. Фоном служит нормально распределённый шум с дисперсией на уровне 10% значения контрастности. Отношение сигнал/шум определяется как контрастность, делённая на девиацию шума. Трём квадратам на Рис. 25.8 соответствуют отношения сигнал/шум 0.5, 1.0 и 2.0. В общем случае проблемы могут возникнуть, когда отношение сигнал/шум оказывается менее 1.0.

Точное значение минимального отношения сигнал/шум, при котором объект ещё может быть замечен, зависит от размера объекта: чем больше объект, тем легче его обнаружить. Чтобы это доказать, представим, что мы сгладили изображения, представленные на Рис. 25.8, с помощью фильтра с ядром  $3 \times 3$  пикселя. В ре-



**Рис. 25.8.** Минимально необходимое отношение сигнал/шум. Объект может быть замечен на изображении, только если его контрастность достаточно велика, чтобы преодолеть влияние шумов. Например, на представленном рисунке показаны три квадрата, характеризуемые отношением сигнал/шум 2.0, 1.0 и 0.5 (отношение сигнал/шум определяется отношением контрастности объекта к дисперсии шума).

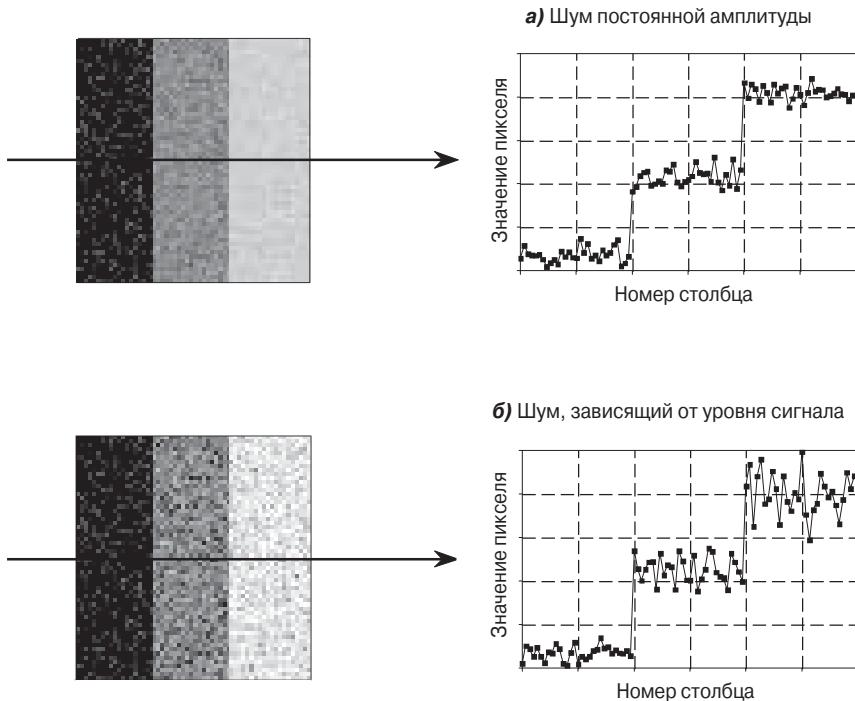
зультате этой процедуры контрастность осталась без изменений, а уровень шума уменьшился в 3 раза (корень квадратный из числа пикселей в ядре фильтра). Отношение сигнал/шум увеличилось в 3 раза, и низкоконтрастные объекты стали заметнее. Чтобы увидеть совсем слабо выраженные объекты, размер ядра фильтра можно увеличить. Например, ядро  $5 \times 5$  пикселей повысит отношение сигнал/шум в  $\sqrt{25} = 5$  раз. Увеличивать размер ядра можно до тех пор, пока оно не станет равнозначимо размеру обнаруживаемого объекта. А это означает, что способность объекта быть обнаруженным пропорциональна квадратному корню из его площади. Так, если объект увеличился в диаметре в 2 раза, его можно будет обнаружить при наличии вдвое большего шума.

Обработка видимого изображения, осуществляемая мозгом человека, реализуется во многом так же: изображение сглаживается с помощью ядер фильтров разных размеров в попытке распознать объект с низкой контрастностью. **Рис. 25.8** иллюстрирует, как люди различают объекты на фоне шумов. Несмотря на то что выявить чёткие контуры объектов на этих рисунках достаточно сложно, они все же являются для нас очевидными. Чтобы действительно понять, насколько велики способности зрительной системы человека, попытайтесь разработать алгоритм, который бы распознавал объекты в такой среде с низким отношением сигнал/шум. Вы будете очень удивлены: ваш мозг легко может делать то, что далеко не под силу программному коду!

Обычно изображения сопровождаются шумами двух видов. Первый из них, показанный на **Рис. 25.9а**, характеризуется постоянной амплитудой. То есть уровень шума одинаков как для тёмных, так и для светлых областей изображения. Второй, проиллюстрированный на **(б)**, соответствует шуму, уровень которого зависит вместе с уровнем сигнала, так что яркие области будут более зашумлены, чем тёмные. В большинстве случаев на изображении присутствуют сразу два шума, но один из них превалирует. В частности, уровень шума обычно падает вместе с уменьшением уровня сигнала до тех пор, пока не будет достигнут уровень шума постоянной амплитуды.

Типовой источник шума постоянной амплитуды — предусилитель сигнала изображения. Все аналоговые электронные цепи генерируют шум, но особенно существенным его влияние оказывается там, где мал уровень сигнала, т. е. непосредственно на ПЗС или другом датчике изображения. Шум предусилителя возникает из-за хаотического движения электронов в транзисторах. Это делает уровень шума зависимым от конструкции электронной схемы, а не от уровня усиливаемого сигнала. К примеру, типичная ПЗС-камера характеризуется отношением сигнал/шум от 300 до 1000 (40...60 дБ), определяемым как отношение полного размаха амплитуды сигнала к среднему отклонению шума с постоянной амплитудой.

Шум, который растёт с уровнем сигнала, возникает, когда изображение представлено малым числом отдельных частиц. Например, это могут быть рентгеновские лучи, прошедшие сквозь тело пациента, фотоны света, попадающие в фотокамеру, или электроны в источнике ПЗС. Математически эти процессы описываются *счётной статистикой*, или *статистикой Пуассона*. Предположим, что поверхность ПЗС равномерно освещена, так что в среднем 10 000 электронов генерируется в каждой потенциальной яме. По чистой случайности некоторые ямы будут иметь больше электронов, а некоторые меньше. Говоря точнее, число электронов будет нормально распределено со средним значением, равным 10 000,



**Рис. 25.9.** Шумы, характерные для изображений. Обычно изображения сопровождаются шумами двух видов. На (а) показан шум, уровень которого остаётся постоянным при изменении уровня сигнала. Это типичный электронный шум. На (б) уровень шума растёт пропорционально корню квадратному из уровня сигнала. Этот тип шума порождается детектированием малого числа частиц, таких как фотоны света, электроны, рентгеновские лучи.

и некоторым СКО, описывающим степень разброса значений от одной потенциальной ямы к другой. Ключевым свойством статистики Пуассона является то, что СКО оказывается равным корню квадратному из числа отдельных частиц. То есть если имеется  $N$  частиц в каждом пикселе, то среднее значение равно  $N$ , а СКО равно  $\sqrt{N}$ . Это делает отношение сигнал/шум равным  $N/\sqrt{N}$  или просто  $\sqrt{N}$ . В форме уравнения:

$$\mu = N, \sigma = \sqrt{N}, C/I = \sqrt{N} \quad (25.1)$$

Статистика Пуассона. Сигнал, распределённый по Пуассону, имеет среднее значение  $\mu$ , равное среднему числу отдельных частиц  $N$ . СКО  $\sigma$  равно корню квадратному из среднего числа частиц. Отношение сигнал/шум ( $C/I$ ) равно среднему значению, делённому на СКО.

В примере с ПЗС СКО равно  $\sqrt{10000} = 100$ . Также и отношение сигнал/шум равно  $\sqrt{10000} = 100$ . Если среднее число электронов на источник (потенциальную яму) вырастет до одного миллиона, то и СКО, и отношение сигнал/шум увеличится до 1000. То есть шум станет больше, если сигнал будет больше (б). Однако сигнал растёт быстрее, чем шум, поэтому в целом отношение сигнал/шум возраст-

тает. Только не думайте, что более слабый сигнал будет сопровождаться меньшим шумом и, следовательно, окажется информативнее. Помните, что ваша цель — не уменьшить уровень шума, а извлечь из смеси сигнала и шума полезную составляющую. Это делает отношение сигнал/шум ключевым параметром.

Многие системы отображения функционируют путём преобразования одних частиц в другие. Например, рассмотрим, что происходит в медицинских рентгеновских приборах. Внутри рентгеновской лучевой трубы электроны ударяются о металлическую поверхность, формируя рентгеновские лучи. После прохождения через пациента рентгеновские лучи ударяют в вакуумный детектор — усилитель изображения. Здесь рентгеновские лучи последовательно преобразуются в фотоны света, затем в электроны и затем опять в фотоны. Эти фотоны света проходят в фотокамеру, где они преобразуются в электроны на ПЗС. В каждой из этих промежуточных форм изображение представляется конечным числом частиц, сопровождаемых аддитивным шумом в соответствии с выражением (25.1). Результирующее отношение сигнал/шум формируется как сумма шумов всех ступеней. При этом обычно доминирующей является одна ступень. Это та ступень, на которой отношение сигнал/шум минимально из-за наименьшего числа частиц. Эта ступень является ограничивающей.

В системах ночного видения таким ограничивающим фактором является количество фотонов света, которые могут быть зафиксированы камерой. Чем темнее ночь, тем больше уровень шума в конечном изображении. Таким же примером является медицинская рентгеновская аппаратура. Ограничение здесь — это число рентгеновских лучей, падающих на детектор. Более высокий уровень излучения обеспечивает меньший уровень шума на изображении за счёт большего уровня облучения пациента.

Когда шум со статистикой Пуассона будет являться основным в системе? Он доминирует всегда, когда шум, являющийся результатом описанного ограничивающего фактора, окажется больше, чем шумы от других источников, например аналоговой электроники. Рассмотрим, к примеру, типичную ПЗС-камеру с отношением сигнал/шум, равным 300. Это означает, что шум от предуслителя ПЗС равен  $1/300$  полной шкалы сигнала. Эквивалентный пуассоновский шум формируется, если ограничивающая ступень системы содержит 90 000 частиц на пиксель. Если число частиц больше, то шум предуслителя будет доминирующим. По этой причине большинство разрабатываемых ПЗС-матриц имеют зарядовую ёмкость потенциальной ямы (full-well capacity) от 100 000 до 1 000 000 электронов для минимизации шума Пуассона.

## 25.4. Морфологическая обработка изображений

Распознавание объектов на изображении может быть очень трудной задачей. Одним из способов её упрощения является переход от изображения в шкале серого к бинарному изображению, каждый пиксель которого может принимать лишь значение нуля или единицы. Для бинарных изображений применяются различные методы обработки, в том числе: *анализ пятен* (blob analysis), *анализ связности* (connectivity analysis) и *морфологическая обработка* (morphological image

processing), берущая своё название от греческого слова «*morphe*», что значит «форма». Морфологическая обработка основывается на строгих математических выкладках теории множеств, однако на практике сложная математика редко бывает необходима. Большинство алгоритмов морфологической обработки используют лишь простые, ориентированные на конкретную задачу логические операции. Каждая частная задача требует своего решения, найденного методом проб и ошибок. Это в большей степени искусство, чем наука. Чаще приходится использовать набор хитростей и алгоритмических фокусов, чем стандартные схемы расчётов и математические правила. Приведём несколько примеров.

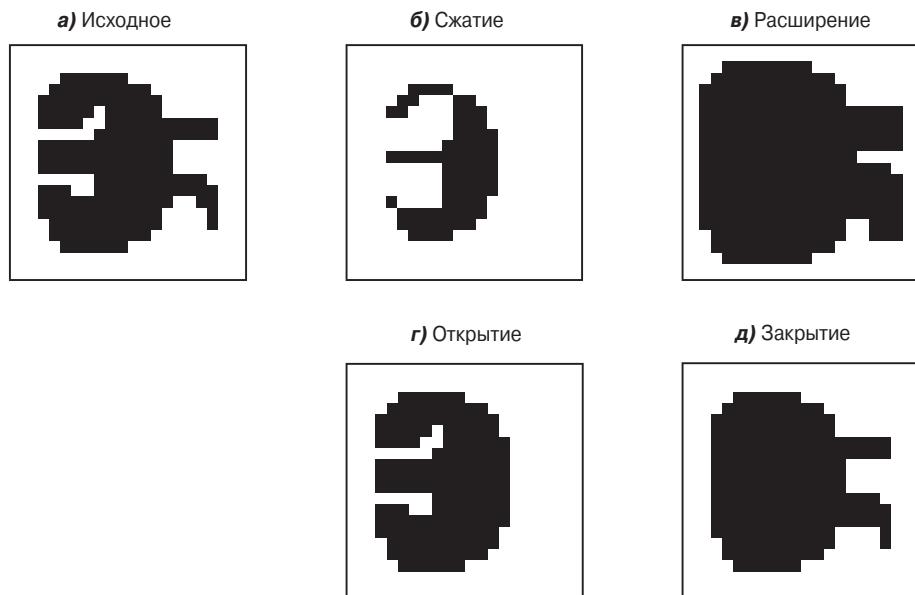
**Рис. 25.10а** показывает пример бинарного изображения. Это может быть, например, инфракрасное изображение танка неприятеля, или астероид, запечатленный на космической фотографии, или подозрительная опухоль на медицинском рентгеновском снимке. Каждый пиксель фона имеет белый цвет, а каждый пиксель объекта — чёрный. Обычно бинарные изображения формируются с помощью пороговой классификации исходного изображения в шкале серого. При этом пикселям со значением, большим порога, присваивается значение единицы, а пикселям со значением, меньшим порога, — нуля. Часто изображения в шкале серого перед преобразованием их в бинарные с использованием пороговой классификации предварительно обрабатывают линейными методами. Например, осуществляют выравнивание освещённости (о котором шла речь в Главе 24), что даёт улучшение качества бинарного изображения.

На **(б и в)** показано, как изображение изменяется при использовании двух типовых морфологических операций: сжатия и расширения. При сжатии каждый пиксель объекта, который «соприкасается» с пикселием фона, преобразуется в пиксель фона. При расширении каждый пиксель фона, который соприкасается с пикселием объекта, преобразуется в пиксель объекта. Сжатие делает объекты меньше и может разбить один объект на несколько. Расширение делает объекты больше и может привести к слиянию нескольких объектов в одно целое.

На **(г)** проиллюстрирована операция открытия, определяемая как сжатие, за которой следует расширение. На **(д)** показан результат другой операции — закрытия, определяемого как растяжение, за которым следует сжатие. Эти примеры показывают, что открытие удаляет маленькие островки и тонкие полоски пикселей объекта. В свою очередь закрытие удаляет островки и полоски фона. Описанные действия оказываются очень полезны при работе с зашумлёнными изображениями, отдельные пиксели которых имеют неверное бинарное значение. Например, таким образом можно подкорректировать изображение, если известно, что на нём не может быть «дыр», а границы объектов должны быть гладкими.

**Рис. 25.11** показывает пример морфологической обработки. Изображение на **(а)** — это бинарное изображение отпечатка пальца. Необходимо разработать алгоритм анализа таких образов, позволяющий сопоставить индивидуальный отпечаток пальца с имеющимися в базе данных. Общепринятым приёмом в этих алгоритмах является операция, называемая *скелетонизация* **(б)**. Она упрощает изображение, удаляя избыточные пиксели, меняя их цвет с чёрного на белый. В результате все полоски превращаются в линии шириной в один пиксель.

**Программы 25.1 и 25.2** реализуют операцию скелетонизации. Несмотря на то, что изображение отпечатка пальца является бинарным, оно представляется массивом, в котором каждый элемент (пиксель) может иметь значения 0...255. Чёр-



**Рис. 25.10.** Морфологические преобразования. При морфологической обработке бинарных изображений используются четыре основных действия: сжатие, расширение, открытие и закрытие. На (а) показан пример бинарного изображения. Результат применения этих операций к изображению на (а) иллюстрируют (б...е).



**Рис. 25.11.** Бинарная скелетонизация. Бинарное изображение отпечатка пальца (а) состоит из полосок различной толщины. Скелетонизация (б) приводит к тому, что все линии оказываются шириной только в один пиксель.

ному пикселию соответствует значение 0, а белому — 255. Как видно из **Программы 25.1**, алгоритм состоит из 6 итераций, которые постепенно превращают полосы в тонкие линии. Число итераций подбирается методом проб и ошибок. Количество итераций можно считать достаточным, если очередная итерация уже не вносит изменений.

На одной итерации каждый пиксель изображения оценивается с точки зрения необходимости удаления. При этом применяется целый ряд критериев, по которым принимается решение об изменении цвета пикселя с чёрного на белый. Строки 200...240 — это цикл по всем пикселям изображения, а оценка необходимости удаления пикселя принимается в **Программе 25.2** (подпрограмма). Если пиксель нужно оставить, то в указанной подпрограмме с ним не выполняется никаких действий. Если же пиксель нужно удалить, подпрограмма меняет его значение с 0 на 1. Это указывает главной программе, что данный пиксель пока чёрный, но должен быть изменён на белый в конце итерации. После оценки всех пикселей происходит изменение значений отмеченных пикселей с 1 на 255, что выполняется в строках 260...300. Результатом такой двухэтапной процедуры является уменьшение толщины линий равномерно со всех направлений, а не только с тех сторон, которые определялись бы порядком сканирования изображения по строкам и столбцам.

### **Программа 25.1**

```

100 'ПРОГРАММА СКЕЛЕТОНИЗАЦИИ
110 'Пиксели объекта имеют значение 0 (чёрный цвет)
120 'Пиксели фона имеют значение 255 (белый цвет)
130 '
140 DIM X%[149,149] 'X%[ , ] хранит обрабатываемое изображение
150 '
160 GOSUB XXXX 'Предполагаемая подпрограмма загрузки X%[ , ]
170 '
180 FOR ITER% = 0 TO 5 'Цикл из 6 итераций
190 '
200 FOR R% = 1 TO 148 'Цикл по каждому пикселию изображения.
210 FOR C% = 1 TO 148 'Подпрограмма 5000 (Программа 25.2) указывает, какой
220 GOSUB 5000 'пиксель можно изменить с чёрного на белый,
230 NEXT C% 'помечая такие пиксели значением 1.
240 NEXT R%
250 '
260 FOR R% = 0 TO 149 'Цикл по каждому пикселию изображения, изменяющий
270 FOR C% = 0 TO 149 'цвет помеченных пикселей с чёрного на белый.
280 IF X%(R%,C%) = 1 THEN X%(R%,C%) = 255
290 NEXT C%
300 NEXT R%
310 '
320 NEXT ITER%
330 '
340 END

```

### **Программа 25.2**

```

5000 ' Подпрограмма, определяющая, может ли быть удалён пиксель X%[R%,C%].
5010 ' Если удовлетворены все четыре условия, то X%(R%,C%) присваивается значение 1,
5020 ' указывающее, что он должен быть удалён в конце главной программы.
5030 '
5040 'Правило 1: Ничего не делать, если пиксель белый.
5050 IF X%(R%,C%) = 255 THEN RETURN
5060 '

```

```

5070 '
5080 'Правило 2: Ничего не делать, если все соседние пиксели чёрные.
5090 IF X%[R% -1,C% ] <> 255 AND X%[R% ,C%+1] <> 255 AND
X%[R%+1,C% ] <> 255 AND X%[R% ,C% -1] <> 255 THEN RETURN
5100 '
5110 '
5120 'Правило 3: Ничего не делать, если только один соседний пиксель чёрный.
5130 COUNT% = 0
5140 IF X%[R% -1,C% -1] = 0 THEN COUNT% = COUNT% + 1
5150 IF X%[R% -1,C% ] = 0 THEN COUNT% = COUNT% + 1
5160 IF X%[R% -1,C%+1] = 0 THEN COUNT% = COUNT% + 1
5170 IF X%[R% ,C%+1] = 0 THEN COUNT% = COUNT% + 1
5180 IF X%[R%+1,C%+1] = 0 THEN COUNT% = COUNT% + 1
5190 IF X%[R%+1,C% ] = 0 THEN COUNT% = COUNT% + 1
5200 IF X%[R%+1,C% -1] = 0 THEN COUNT% = COUNT% + 1
5210 IF X%[R% ,C% -1] = 0 THEN COUNT% = COUNT% + 1
5220 IF COUNT% = 1 THEN RETURN
5230 '
5240 '
5250 'Правило 4: Ничего не делать, если соседние пиксели не связаны между собой
5260 'Это определяется подсчётом перепадов цвета с чёрного на белый
5270 'при движении по часовой стрелке по 8 соседним пикселям.
5280 COUNT% = 0
5290 IF X%[R% -1,C% -1] = 0 AND X%[R% -1,C% ] > 0 THEN COUNT% = COUNT% + 1
5300 IF X%[R% -1,C% ] = 0 AND X%[R% -1,C%+1] > 0 AND X%[R% ,C%+1] > 0
THEN COUNT% = COUNT% + 1
5310 IF X%[R% -1,C%+1] = 0 AND X%[R% ,C%+1] > 0 THEN COUNT% = COUNT% + 1
5320 IF X%[R% ,C%+1] = 0 AND X%[R%+1,C%+1] > 0 AND X%[R%+1,C% ] > 0
THEN COUNT% = COUNT% + 1
5330 IF X%[R%+1,C%+1] = 0 AND X%[R%+1,C% ] > 0 THEN COUNT% = COUNT% + 1
5340 IF X%[R%+1,C% ] = 0 AND X%[R%+1,C% -1] > 0 AND X%[R% ,C%-1] > 0
THEN COUNT% = COUNT% + 1
5350 IF X%[R%+1,C% -1] = 0 AND X%[R% ,C% -1] > 0 THEN COUNT% = COUNT% + 1
5360 IF X%[R% ,C% -1] = 0 AND X%[R% -1,C% -1] > 0 AND X%[R%-1,C% ] > 0
THEN COUNT% = COUNT% + 1
5370 IF COUNT% > 1 THEN RETURN
5380 '
5390 '
5400 'Если все условия выполняются, то пиксель помечается и изменяется на белый в
5410 'конце итерации очередной итерации вызвавшей программы
5420 X%(R%,C%) = 1
5430 '
5440 RETURN

```

Решение об удалении пикселя принимается при соблюдении четырёх правил, которые перечислены в **Программе 25.2**. Пиксель должен удовлетворять всем этим условиям, чтобы его цвет сменился с чёрного на белый. Первые три условия достаточно просты. А вот четвёртое оказывается гораздо сложнее. Посмотрите на **Рис. 25.12**. Здесь пиксель с координатами  $[R, C]$  имеет восемь «соседей»: четыре соседних пикселя, расположенные на одной горизонтали или вертикали (пиксели 2, 4, 6, 8), называют ближайшими соседями; четыре пикселя, расположенные

по диагонали (пиксели 1, 3, 5, 7), называют отдалёнными соседями. Перечислим четыре условия изменения цвета пикселя.

*Первое.* Рассматриваемый пиксель должен быть чёрным. Если он белый, то никаких действий не предпринимается.

*Второе.* По крайней мере один из ближайших соседей рассматриваемого пикселя должен быть белым. Этот факт гарантирует, что сжатие широкой полосы производится с внешней стороны. Другими словами, если пиксель и все окружающие его пиксели чёрные, то на данной итерации менять его не следует. Почему используются только ближайшие соседи, а не все окружающие пиксели? Ответ прост: если выполнить алгоритм и первым, и вторым способами, то окажется, что использование только ближайших соседей даёт лучший результат. Помните, что для морфологической обработки изображений такой подход является широко распространённым: для определения того, какой алгоритм будет лучше работать, используется метод проб и ошибок.

*Третье.* Должно быть более одного соседнего чёрного пикселя. Если же он только один, то это означает, что данный пиксель является концом линии и удалять его нельзя.

*Четвёртое.* Пиксель не может быть удалён, если в результате удаления его соседи окажутся разъединены. Это нужно, чтобы полосы превращались в непрерывные линии, а не в группу отдельных отрезков. «Соединённые» — означает, что все чёрные соседи касаются друг друга (**б**), а «разъединённые» — означает, что чёрные пиксели образуют две (или более) группы (**в**).

Алгоритм проверки того, соединены или разъединены соседние пиксели, основан на подсчёте перепадов цвета с чёрного на белый между смежными соседними пикселями в направлении по часовой стрелке. Например, если пиксель 1 — чёрный, а пиксель 2 — белый, то это рассматривается как перепад цвета. Аналогично если пиксель 2 — чёрный, а пиксели 3 и 4 — белые, то это также перепад цвета. Всего бывает 8 случаев, соответствующих перепаду цвета с чёрного на белый. На (**б** и **в**) такие перепады цвета отмечены звёздочками. Суть алгоритма в следующем: получиться может либо 0, либо единственный перепад цвета с чёрного на белый, если соседние пиксели являются соединёнными. Если число перепадов цвета больше одного, то это означает, что пиксели разъединены.

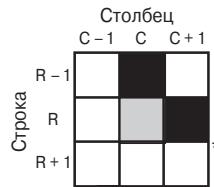
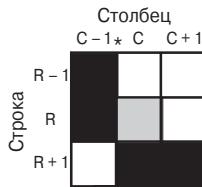
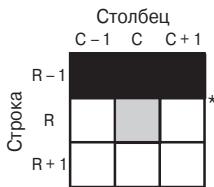
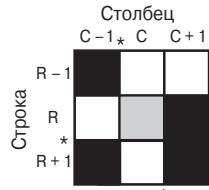
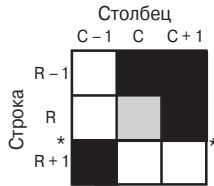
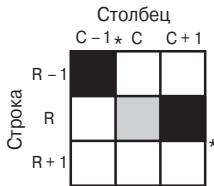
В качестве дополнительных примеров бинарной обработки изображений рассмотрим ещё несколько алгоритмов, которые могут быть полезны после скелетонизации отпечатков пальцев. Недостатком рассмотренного алгоритма является то, что в данном случае скелетонизация оставляет достаточно много коротких волосков по бокам линий. Для уменьшения такого рода помех существует ряд других подходов. Например, в программе можно циклически удалять все пиксели на концах всех линий. Это те пиксели, у которых только один соседний чёрный пиксель. Если произвести такую процедуру несколько раз, то ненужные волоски будут удалены, правда, одновременно укоротятся и правильные линии. Более эффективным мог бы быть метод, сканирующий изображение и детектирующий «волоски» (пиксели, имеющие более двух соседей). Для каждого ответвления от линии считают число пикселей. Если число пикселей оказывается меньше некоторого значения (скажем, 5), значит, это ненужный «волосок», и его можно удалить, сменив цвета всех пикселей с чёрного на белый.

Другой возможный алгоритм основан на переходе от растрового формата изображения к векторному. Это означает, что формируется список линий, при-

**а) Нумерация пикселей**

Столбец			
C - 1	C	C + 1	
Строка	1	2	3
R - 1	8		4
R	7	6	5
R + 1			

**Рис. 25.12.** Соседние пиксели. Пиксель с координатами [R, C] имеет восемь соседей, обозначенных на (а) цифрами 1...8. На (б) и (в) показаны примеры, на которых соседние пиксели являются соединёнными и разъединёнными соответственно. Такое понимание терминов «соединённые» и «разъединённые» подразумевает правило номер четырёх алгоритма скелетонизации.

**б) Соединённые соседи****в) Разъединённые соседи**

существующих на изображении, и пикселей, из которых эти линии состоят. В векторном формате каждая линия отпечатка пальца выступает как объект, описываемый набором параметров. Это отличает его от растрового формата, в котором изображение состоит из множества независимых пикселей. Чтобы выполнить переход к векторному формату, можно использовать следующий подход. Сканируем изображение в поисках точек, соответствующих концам линий: это такие точки, которые имеют только один соседний чёрный пиксель. Далее, начиная с одного конца, пробегаем по всей линии, находя связанные пиксели. Достигнув противоположного конца линии, мы объявляем все её пиксели одним объектом и таким образом представляем её в векторном формате.

## 25.5. Компьютерная томография

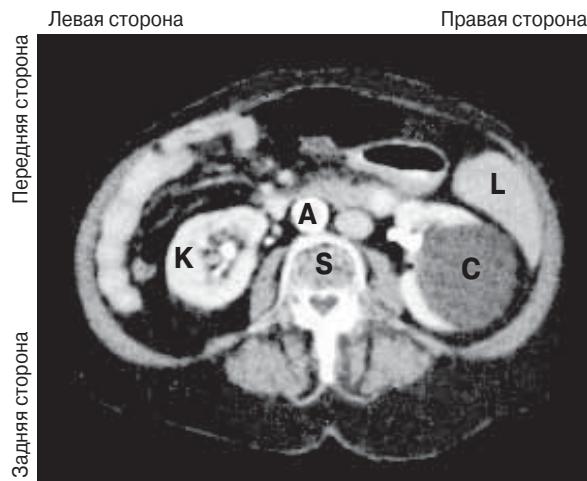
Серьёзной проблемой при анализе изображений, полученных с помощью рентгеновских лучей (или других проникающих излучений), является то, что они оказываются двумерными, хотя отображают трёхмерные объекты. Это означает, что на таком изображении различные детали могут быть перекрыты, несмотря на то, что в действительности они абсолютно разделены. Эта проблема особенно серьёзна в медицинской диагностике, где на одном снимке имеется много разных

анатомических структур, мешающих увидеть интересующий врачей объект. В 1930-х годах эту проблему пытались решить, совместно перемещая источник рентгеновских лучей и их приёмник в процессе формирования изображения. Из-за геометрии такого движения источника и приёмника только одна плоскость внутри исследуемой области остаётся в фокусе, а объекты, находящиеся вне этой плоскости, отображаются размытыми. Эффект размытия аналогичен изображению, получаемому фотоаппаратом, если он сфокусирован на расстояние в 5 футов, а снимает объекты, расположенные на расстоянии, например, 1 или 50 футов. Этот метод построения изображения и многочисленные его разновидности, основанные на перемещении источника и приёмника, теперь называют *классической томографией*. Слово «томография» означает «изображение плоскости».

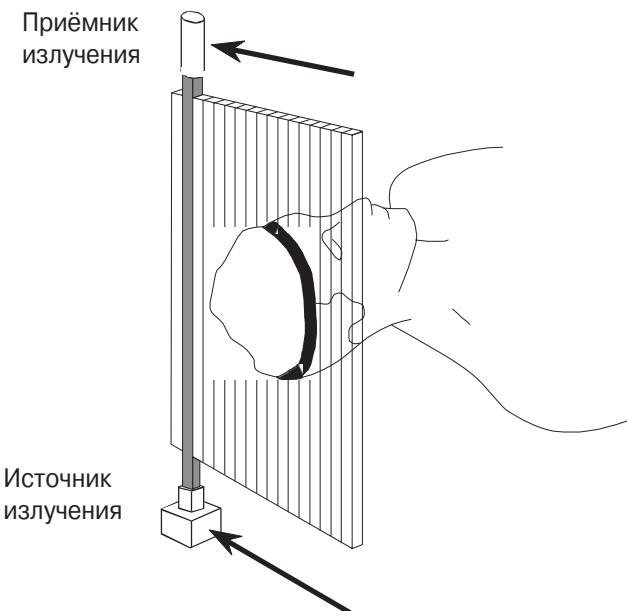
Несмотря на то что классическая томография является широко известной уже более 50 лет, она редко используется на практике. Дело в том, что для этой методики характерно существенное ограничение: объекты, не представляющие интереса, не удаляются из изображения, а только размываются. Поэтому качество получаемого изображения оказывается слишком низким для практического использования. Решением, которое долго не удавалось найти, явилась система, формирующая двумерное изображение одного, представляющего интерес, слоя трёхмерного исследуемого объекта независимо от других слоёв.

Решение было найдено в начале 1970-х годов и названо *компьютерной томографией*. Компьютерная томография произвела революцию в медицинской рентгенографии, предоставив беспрецедентную возможность «заглянуть» внутрь тела человека. На Рис. 25.13 показано типовое изображение, обеспечиваемое компьютерной томографией.

На Рис. 25.14 изображена простая схема получения снимка одного сечения в центре головы человека с использованием компьютерной томографии. Узкий луч рентгеновского излучения идёт от источника к детектору. При этом регистрируе-



**Рис. 25.13.** Изображение, получаемое с помощью компьютерной томографии. Это изображение брюшной полости человека на уровне пупка. Видны многие органы, такие как печень (L), почка (K), аорта (A), позвоночник (S) и киста (C) на правой почке. Компьютерная томография даёт гораздо больше возможностей визуализации внутренних органов человека по сравнению с обычным рентгеном.



**Рис. 25.14.** Регистрация данных в процессе компьютерной томографии. В простой системе компьютерной томографии узкий рентгеновский луч проходит через тело человека от источника к детектору. Источник и детектор перемещаются, что приводит к формированию полного вида одного сечения. Затем источник и детектор поворачиваются с шагом  $1^\circ$ , и формирование изображения сечения повторяется под другим углом.

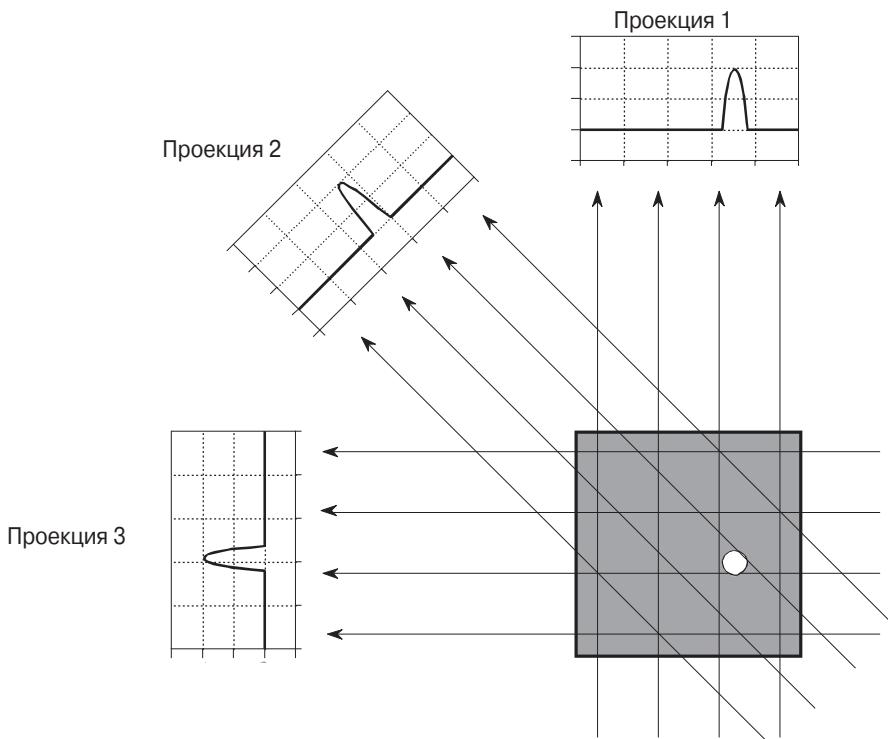
мое детектором излучение определяется всей совокупностью объектов, оказавшихся на пути луча. Такие материалы, как кости или зубы, больше задерживают излучение и дают более слабый сигнал по сравнению с мягкими тканями или жиром. Как показано на рисунке, источник и детектор затем перемещаются, чтобы сформировать один полный вид (проекцию) под данным углом наблюдения. На рисунке показан только один такой вид, но в действительности для построения изображения методом компьютерной томографии необходимо сформировать 300...1000 таких частных изображений, наблюдаемых под разными углами с шагом поворота от  $0.3$  до  $1.0^\circ$ . Для этого источник и детектор монтируются на врашающейся основе так, чтобы исследуемый пациент мог находиться внутри. Ключевым моментом в компьютерной томографии является то, что рентгеновские лучи проходят только через тонкий слой исследуемого объекта. Это отличает её от классической томографии, в которой лучи идут через все структуры, а затем падавляются изображения мешающих объектов. Компьютерная томография не позволяет лишней информации даже попасть в регистрируемые данные.

Перед тем как выполнить построение конечного изображения, обычно необходимы некоторые шаги предварительной обработки. Например, требуется вычислить логарифм каждого результата измерений рентгеновских лучей. Это нужно, поскольку интенсивность рентгеновского излучения убывает экспоненциально при прохождении через ткани. Взятие логарифма обеспечивает линеаризацию зависимости сигнала от характеристик исследуемых тканей. Другие варианты предварительной обработки могут использоваться, например, чтобы компенсировать использование полихроматических рентгеновских лучей или многоэлементных де-

текторов (в отличие от единственного элемента детектора, показанного на Рис. 25.14). Хотя все эти шаги являются очень важными для реализации метода в целом, они не относятся непосредственно к алгоритмам восстановления изображений, поэтому мы не будем их обсуждать.

**Рис. 25.15** иллюстрирует взаимосвязь между отдельными проекциями, получаемыми описанным образом, и соответствующим изображением. Каждый отсчёт, регистрируемый системой компьютерной томографии, равен сумме всех значений пикселей изображения, расположенных на одном луче, «указывающем» на этот отсчёт. Например, проекция 1 находится сложением пикселей всех строк. Аналогично проекцию 3 находят сложением пикселей всех столбцов. Другие проекции, например 2, находятся суммированием пикселей вдоль лучей, которые идут под некоторым углом.

Существует четыре подхода к вычислению изображения сечения, заданного множеством его видов под разными углами. В компьютерной томографии они называются *алгоритмами восстановления*. Первый подход совершенно непригоден для практического применения, но он позволяет лучше понять суть проблемы. Этот подход основан на решении большого числа совместных линейных уравнений. Одно уравнение составляется для каждого измерения: каждая выборка каж-



**Рис. 25.15.** Отдельные проекции в компьютерной томографии. При компьютерной томографии регистрируется множество проекций интересующего сечения, и по ним восстанавливается искомое изображение. Каждый отсчёт одной проекции равен сумме значений пикселей восстанавливаемого изображения, расположенных вдоль луча, который указывает на данный отсчёт. В этом примере строится изображение яркой точки, окружённой нулями. Здесь показаны только три проекции, но обычно компьютерная томография использует сотни видов, немного отличающихся углом наблюдения.

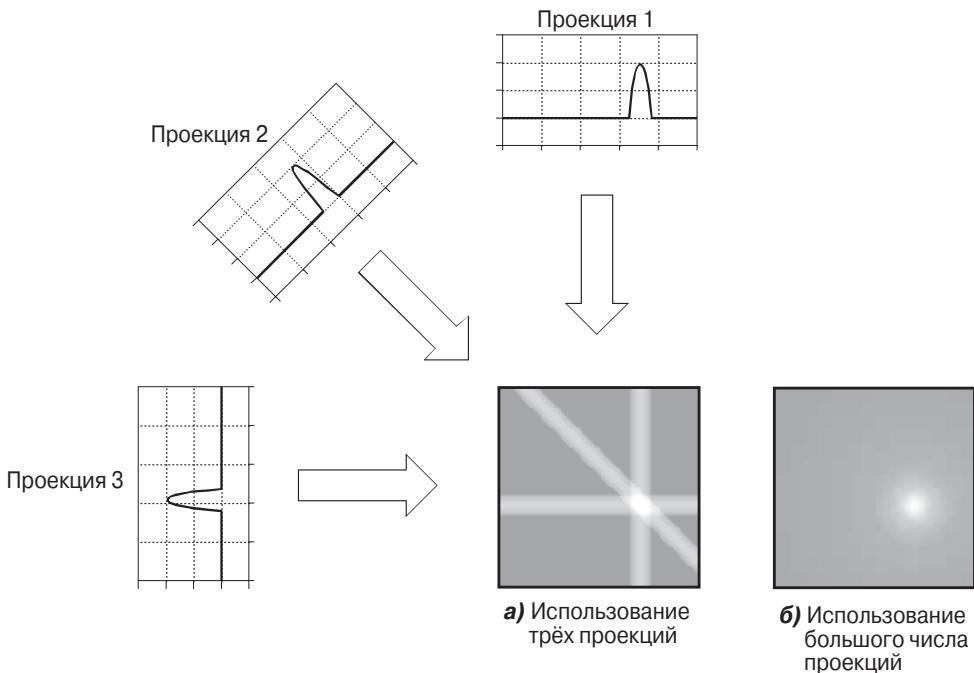
дого регистрируемого изображения (вида) выражается суммой определённого набора пикселей исходного изображения. Чтобы найти  $N^2$  неизвестных переменных (значений пикселей восстанавливаемого изображения), необходимо составить  $N^2$  независимых уравнений и, следовательно, провести  $N^2$  измерений. В большинстве случаев компьютерные томографы требуют использования на 50% большего числа выборок, чем следует из этого анализа. Например, для восстановления изображения размером  $512 \times 512$  пикселей томограф формирует 700 проекций, из 600 пикселей каждая. За счёт такой переопределённости системы при восстановлении изображения удается снизить уровень шумов и искажений. Недостатком описанного метода восстановления является огромное время вычислений. Решение системой нескольких сот тысяч уравнений — это пугающая задача.

Второй подход представляет собой итерационную процедуру восстановления изображения на основе повторяющихся этапов коррекции. Существует несколько вариаций этого метода: *алгебраический метод восстановления* (Algebraic Reconstruction Technique), *метод совместной итерационной реконструкции* (Simultaneous Iterative Reconstruction Technique) и *среднеквадратический итерационный метод* (Iterative Least Squares Technique). Разница между ними состоит в том, каким образом выполняется коррекция на каждой итерации: луч за лучом, пиксель за пикселеем или одновременно по всему множеству данных соответственно. Мы рассмотрим только *алгебраический метод восстановления*.

Перед тем как начать выполнение алгоритма восстановления, всем пикселям изображения присваивается некоторое произвольное значение. Затем используется итерационная процедура, постепенно изменяющая элементы массива изображения в соответствии с каждым новым измерением. На каждой итерации происходит опрос всех точек измеренного изображения и задаётся вопрос: как изменить значения пикселей в массиве восстанавливаемого изображения, чтобы они соответствовали данному измерению? При этом измеренная выборка сравнивается с суммой пикселей изображения вдоль луча, указывающего на эту выборку. Если сумма оказывается меньше выборки, то значения всех пикселей вдоль луча увеличиваются. Аналогично, если сумма больше выборки, значения пикселей уменьшаются. После завершения первой итерации остаётся ошибка между суммами пикселей восстанавливаемого изображения и измеренными значениями, потому что корректировка пикселей, выполняемая для очередного измерения, сводит на нет все ранее внесённые изменения. Однако метод предполагает, что эта ошибка становится меньше от итерации к итерации, пока не будет найдено точное решение — восстановленное изображение.

Итерационные методы в общем случае достаточно медленны, но они оказываются полезны, когда другие алгоритмы не применимы. Алгебраический метод восстановления был основой первого коммерческого компьютерного томографа EMI Mark I, выпущенного в 1972 году. Мы ещё вернёмся к итерационным методам в следующей главе, посвящённой нейронным сетям. Теперь расскажем ещё о двух методах восстановления изображений, которые практически полностью заменили итерационные методы в коммерческих томографах.

В основе этих двух методов лежит строгая математическая теория. Они могут служить прекрасным примером применения цифровой обработки сигналов. Один из них называется *методом обратных проекций с фильтрацией*. Он является модификацией ранее известного *метода обратных проекций*. Рис. 25.16 показывает



**Рис. 25.16.** Метод обратных проекций. Данный метод восстанавливает изображение, «размазывая» каждую из зарегистрированных проекций по изображению в направлении противоположном тому, с которого она была получена. Восстановленное таким способом изображение оказывается размытой копией действительного изображения.

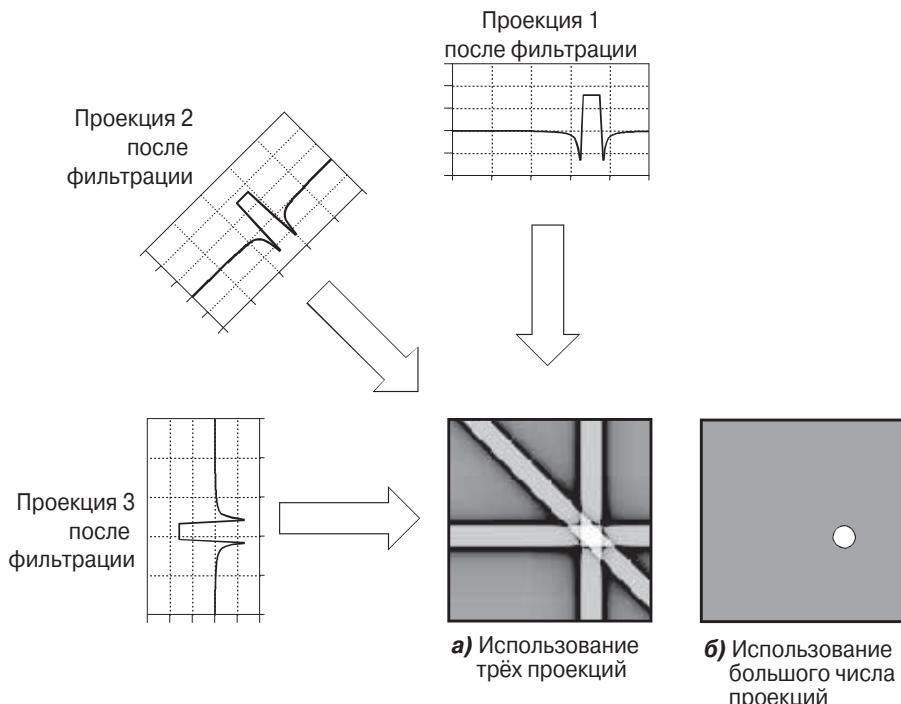
зывает, что метод обратных проекций использует интуитивно понятный принцип и является достаточно простым. Для каждого имеющегося вида находится обратная проекция — всем пикселям восстанавливаемого изображения, расположенным вдоль луча, направленного на данную выборку, присваивается значение самой выборки. В более простой формулировке: обратная проекция формируется «размазыванием» каждого вида по изображению в направлении, противоположном тому, с которого он был первоначально получен. Конечное восстановленное изображение получается в этом случае суммированием всех обратных проекций.

Метод обратных проекций достаточно прост, однако решение задачи с его помощью оказывается не совсем корректным. Как показано на (б), восстановленное данным методом изображение оказывается очень нечётким. Одна точка исходного изображения превращается в целую область в форме кружка, в которой яркость пикселей падает с удалением от центра. Можно сказать, что *функция рассеяния точки* в данном случае является кругообразно симметричной и уменьшается гиперболически с увеличением радиуса.

*Метод обратных проекций с фильтрацией* — это метод, использующий коррекцию изображения, восстановленного с помощью обычного метода обратных проекций. **Рис. 25.17** показывает, что каждая проекция (вид) перед вычислением обратной проекции фильтруется так, чтобы ФРТ оказалась в результате не

размытой. То есть для каждой проекции осуществляется свёртка с одномерным ядром фильтра, в результате чего образуется большое число фильтрованных проекций. По ним восстанавливается исходное изображение методом обратных проекций. Качество восстановления оказывается в этом случае достаточно высоким: исходное и восстановленное изображения будут полностью идентичны, если использовать бесконечное множество проекций и бесконечное число точек в каждой из них.

О том, каким должно быть ядро фильтра, используемого в данном методе, мы кратко скажем позднее. Сейчас лишь отметим, какой эффект даёт фильтрация. Исходное изображение в рассматриваемом примере представляет собой белый кружок с одинаковой яркостью во всех точках, окруженный чёрным фоном (кнопка). Каждая из полученных проекций состоит из горизонтальной линии, соответствующей фону, и всплеска с закруглённой вершиной, представляющего собой белый кружок. Фильтрация изменяет форму проекций в двух существенных аспектах. Во-первых, вершина всплесков становится плоской, что приводит при вычислении обратной проекции к постоянному уровню яркости точек внутри круга. Во-вторых, по сторонам от основного пика появляются небольшие отрицательные выбросы. Благодаря им нечёткость конечного восстанавливаемого изображения удаётся устранить.

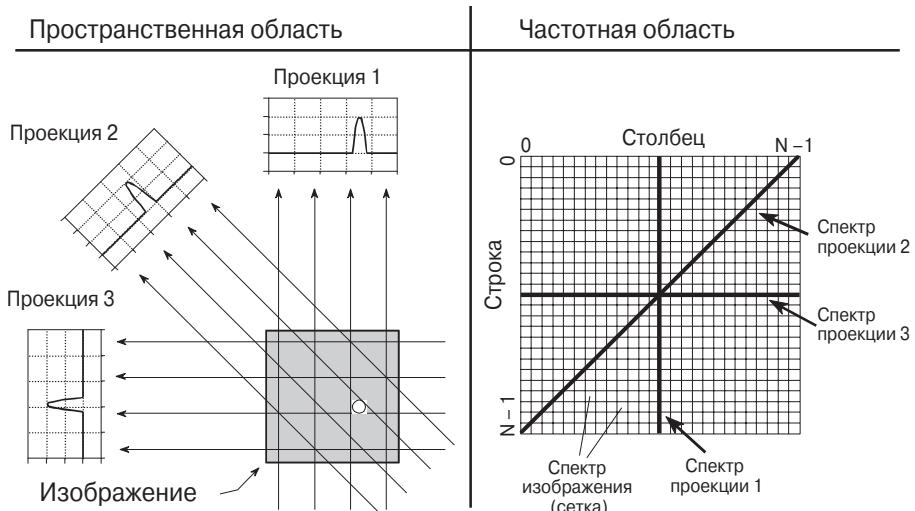


**Рис. 25.17.** Метод обратных проекций с фильтрацией. Данный метод восстанавливает изображение, предварительно фильтруя каждую проекцию перед вычислением обратных проекций. Это позволяет устраниить нечёткость изображения, восстанавливаемого простым методом обратных проекций. Данный метод даёт математически точное восстановление исходного изображения. Метод обратных проекций с фильтрацией является наиболее часто используемым алгоритмом в системах компьютерной томографии.

И наконец, последний, четвёртый метод восстановления называется *восстановлением Фурье*. Напомним, что в пространственной области восстановление изображения при компьютерной томографии состоит в том, чтобы найти взаимосвязь между двумерным исходным изображением и множеством его одномерных проекций. Вычислением двумерного преобразования Фурье исходного изображения и одномерного преобразования Фурье каждой из его проекций задача может быть перенесена в частотную область. Оказывается, что связь исходного изображения с его проекциями в частотной области гораздо проще, чем в пространственной. Переход к анализу проблемы восстановления изображения в частотной области — это большой шаг в развитии технологии компьютерной томографии. В его основе лежит *теорема срезов Фурье* (Fourier slice theorem).

**Рис. 25.18** иллюстрирует проблему одновременно в частотной и пространственной областях. В пространственной области каждая проекция находится интегрированием изображения вдоль лучей, построенных под определённым углом. В данном примере спектр изображения в частотной области представляется как двумерная сетка. Спектр каждой проекции (одномерный сигнал) — это толстая линия на сетке. *Теорема срезов Фурье* утверждает, что спектр проекции представляет собой совокупность значений спектра изображения, расположенных вдоль соответствующей линии (среза). Эти линии показаны на рисунке. Например, спектр проекции 1 — это центральный столбец спектра изображения, а спектр проекции 3 — центральная строка спектра изображения. Заметим, что линия спектра каждой проекции располагается на сетке под тем же углом, под которым проекция была получена из исходного изображения. Все эти частотные спектры включают область отрицательных частот и изображаются с нулевой частотой в центре.

Восстановление Фурье выполняется с помощью трёх действий. Во-первых, вычисляется одномерное БПФ каждой проекции. Во-вторых, полученные спектры проекций используются, чтобы вычислить двумерный частотный спектр изоб-



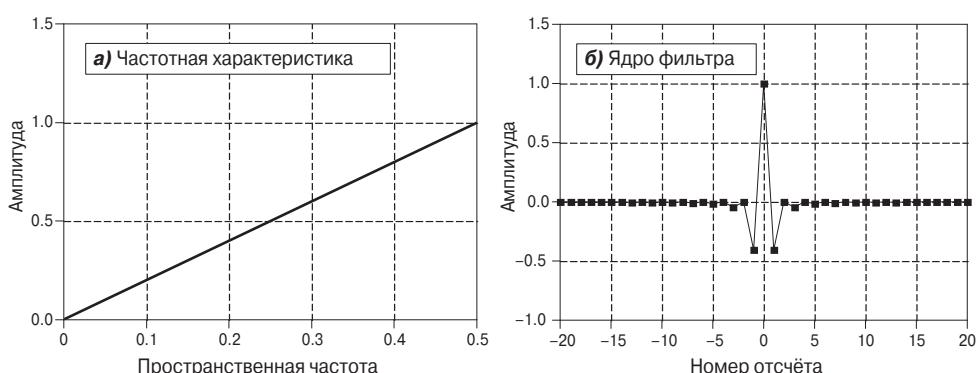
**Рис. 25.18.** Теорема срезов Фурье. Теорема срезов Фурье описывает взаимосвязь между изображением и его проекциями в частотной области. В пространственной области каждая проекция находится интегрированием изображения вдоль лучей под конкретным углом. В частотной области спектр каждой проекции есть одномерный срез двумерного спектра изображения.

ражения, что выполняется на основе теоремы срезов Фурье. Поскольку спектры проекций располагаются радиально, а спектр изображения представляется в прямоугольных координатах, то для такого вычисления дополнительно требуется интерполяция. Третье действие — это вычисление обратного БПФ спектра изображения, позволяющее получить искомое восстановленное изображение.

Преобразование изображения из радиального в прямоугольное можно считать ключом к пониманию метода обратных проекций с фильтрацией. Радиальное расположение характерно для спектров обратных проекций, а прямоугольная сетка — это спектр исходного изображения. Если мы сравним малую область радиального спектра с соответствующей областью прямоугольной сетки, то мы обнаружим, что значения их пикселей идентичны. Однако они имеют разную плотность. Спектр исходного изображения имеет равномерно распределённые пиксели, что соответствует сетке со всеми строками и столбцами одного размера. А вот спектры обратных проекций имеют более высокую плотность точек вблизи центра из-за их радиального расположения. Это можно пояснить таким примером: спицы велосипедного колеса располагаются теснее друг к другу вблизи ступицы. Данный эффект не влияет на результат восстановления Фурье, потому что интерполяция осуществляется по значениям соседних отсчётов независимо от их плотности.

Фильтр в методе обратных проекций с фильтрацией устраняет неравную плотность отсчётов. То есть частотная характеристика фильтра является обратной по отношению к функции плотности отсчётов. При этом, поскольку спектр обратных проекций имеет плотность  $1/f$ , соответствующий фильтр должен иметь частотную характеристику  $H[f] = f$ . Эта частотная характеристика показана на Рис. 25.19а. Ядро фильтра вычисляется как обратное преобразование Фурье (б). Математически фильтр задаётся выражениями

$$\begin{aligned} h[0] &= 1, \\ h[k] &= 0, \quad \text{для чётных значений } k; \\ h[k] &= -4\pi^2/k^2, \quad \text{для нечётных значений } k. \end{aligned} \quad (25.2)$$



**Рис. 25.19.** Фильтр обратных проекций. Частотная характеристика фильтра обратных проекций показана на (а), а ядро фильтра — на (б). Выражение (25.2) представляет собой формулу расчёта ядра фильтра.

Прежде чем завершить тему компьютерной томографии, отметим, что в медицине имеется ещё ряд похожих методов обработки изображений. Все они используют цифровую обработку сигналов. *Томография с позитронной эмиссией* основана на введении в тело человека слаборадиоактивного компонента, который излучает позитроны. Сразу после эмиссии позитроны взаимодействуют с электронами, генерируя два гамма-луча, которые выходят из тела человека в противоположных направлениях. Детекторы радиации, расположенные вокруг тела пациента, обнаруживают эти гамма-лучи и определяют линию, вдоль которой они распространялись. Точка, где были сформированы гамма-лучи, лежит на этой линии, и, чтобы её найти, требуется использовать некоторый алгоритм восстановления подобно методу компьютерной томографии. Получаемое изображение выглядит подобно результату, полученному методом компьютерной томографии, но яркость изображения в данном случае определяется количеством радиоактивного вещества, находящегося в каждой точке исследуемого объекта. Отсюда вытекает основное преимущество томографии с позитронной эмиссией: радиоактивные компоненты могут добавляться к веществам, естественно присутствующим в теле человека, например к глюкозе, и восстанавливаемое изображение позволяет получать информацию о количестве данного вещества в том или ином месте. Таким образом, в данном случае изображение отражает не столько анатомию тела человека, сколько его физиологию. В частности, применение данного метода может позволить сформировать изображение, несущее информацию о том, какая часть мозга человека выполняет те или иные задачи.

Ещё более похожим на компьютерную томографию методом является *магнитно-резонансная диагностика*, которая в настоящее время проводится в большинстве крупных больниц. Данный метод первоначально назывался *ядерным магнитным резонансом*. Однако название пришлось изменить под влиянием общественного мнения: слово *ядерный* настораживает и пугает. Объяснить людям, что термин *ядерный* означает лишь то, что все атомы содержат ядра, оказалось очень непростой задачей. При магнитно-резонансном исследовании пациент помещается в центр мощного магнита. С помощью радиоволн и магнитного поля можно заставить резонировать выборочные ядра, которые начинают излучать вторичные волны. Эти вторичные радиоволны оцифровываются и формируют множество данных, на основе которых удается произвести восстановление изображения. Результатом являются изображения, сильно напоминающие компьютерную томографию. Достоинствами магнитно-резонансной диагностики являются хорошая различимость мягких тканей, простой выбор интересующего сечения исследуемого объекта и отсутствие потенциально опасной рентгеновской радиации. К недостаткам метода относятся относительная дороговизна по сравнению с компьютерной томографией и низкое качество изображения костей и других твёрдых тканей. Компьютерная томография и магнитно-резонансная диагностика останутся основными методами построения изображений в медицине ещё многие годы.

# Глава 26

---

## НЕЙРОННЫЕ СЕТИ

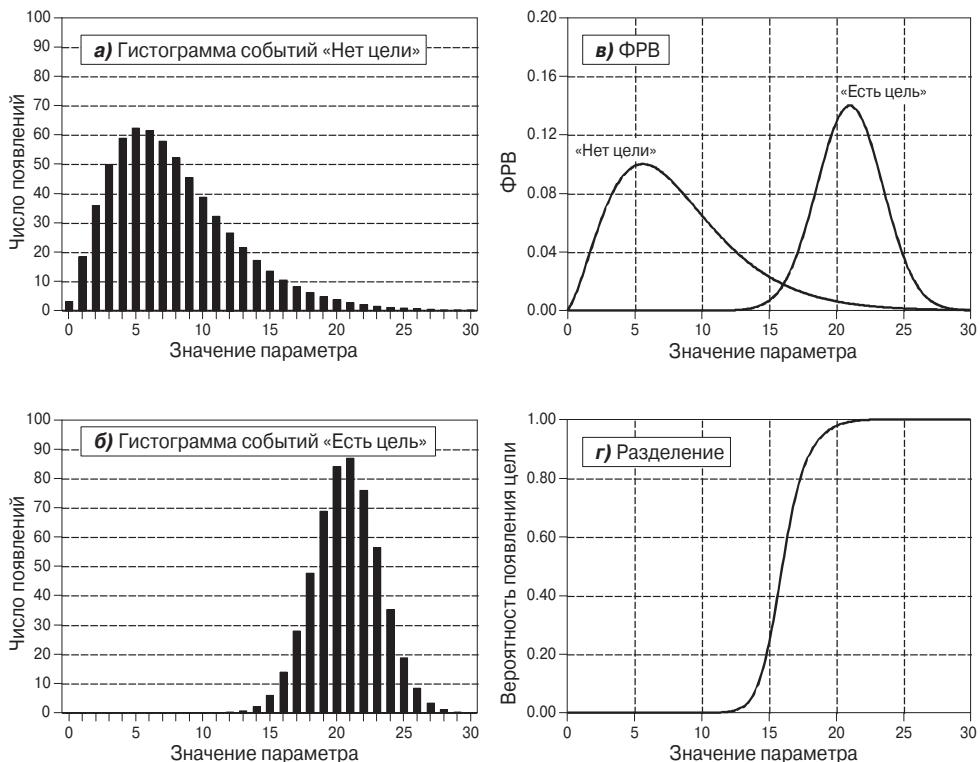
Традиционная ЦОС основана на алгоритмах, преобразующих данные из одной формы в другую с помощью пошаговых процедур. Для работы большинства из них требуется определённые параметры. Например, рекурсивные фильтры используют коэффициенты рекурсии; обнаружение особенностей сигнала может осуществляться с использованием корреляции и пороговой классификации; качество изображений зависит от установленной яркости и контрастности и т. д. Алгоритмы описывают, что должно быть сделано, а параметры задают условия и требования к их реализации. Правильный выбор параметров часто оказывается более важен, чем сам алгоритм. *Нейронные сети* доводят эту идею до крайности — они используют очень простые алгоритмы и множество оптимизируемых параметров. В этом проявляется революционный подход данного направления, отличный от традиционных положений науки и техники: сначала математическая логика и аналитические выкладки, а затем эксперимент. Нейронные сети заменяют эту стратегию решения проблем на метод проб и ошибок, практические действия и методологию: «это работает лучше, чем то». Настоящая глава посвящена вопросам, связанным с выбором параметров нейронных сетей, а также традиционных алгоритмов ЦОС.

### 26.1. Задача обнаружения

Перед учёными и инженерами часто возникает необходимость ответить на вопрос: присутствует некоторый объект или нет, имеет ли место то или иное событие? Например, геофизики исследуют землю с целью поиска нефти; врачи ставят больному диагноз; астрономы изучают вселенную в поиске неземных цивилизаций и т. д. Эти задачи обычно включают сравнение регистрируемых данных с некоторым пороговым значением. Если порог превышен, то считается, что цель (объект или событие) присутствует.

Предположим, например, что вы изобрели устройство для диагностики раковых опухолей у людей. Аппаратура обследует пациента и выводит на дисплей числа от 0 до 30. Маленькие числа означают отсутствие заболевания, а большие указывают, что раковые ткани присутствуют. В целом ваше устройство работает правильно, но иногда оно делает ошибки. Вопрос в том, как вы используете эту систему, чтобы осуществить проверку с пользой для пациента?

**Рис. 26.1** иллюстрирует систематический способ анализа этой ситуации. Предположим, что устройство проверяется на двух группах добровольцев: нескольких сотнях здоровых людей (не цель) и нескольких сотнях больных раком (цель). На **(а и б)** показаны результаты тестирования в виде гистограмм.



**Рис. 26.1.** Вероятность обнаружения цели. На (а и б) показаны гистограммы, характерные для двух групп с заранее известным состоянием «Есть цель» или «Нет цели», построенные относительно некоторого условно выбранного параметра. На основе этих гистограмм могут быть найдены функции распределения вероятности (в). Используя только эту информацию, можно рассчитать и построить кривую (г), которая показывает вероятность события «Есть цель» для конкретных значений параметра.

Как показано в Главе 2, эти гистограммы могут использоваться в качестве оценки функции распределения вероятности (ФРВ) (в). Например, предположим, что устройство апробируется на случайно выбранных здоровых субъектах. По рисунку видно, что имеется примерно 8%-й шанс, что результат тестирования равен 3, и 1%-й шанс, что он будет равен 18. (Этот пример не различает, будет ли число целым, требующим использования функции распределения целых чисел (ФРЧ), или вещественным, требующим ФРВ. Впрочем, сейчас это не столь важно.)

Теперь подумаем о том, что случится, когда это устройство будет использоваться для пациента, состояние здоровья которого неизвестно. Например, если человек, которого мы раньше никогда не видели, получил оценку 15, то что можно заключить? Есть рак или нет? Мы знаем, что вероятность того, что здоровый человек получит оценку 15, равна 2.1%. В то же время вероятность того, что больной раком получит оценку 15, равна 0.7%. Если нет никакой другой информации, мы должны заключить, что человек имеет в 3 раза больше шансов не быть больным раком, чем наоборот. То есть результат тестирования 15 подразумевает, что субъект входит в группу «Есть цель» с вероятностью 25%. Этот метод можно обобщить, сформи-

ровав кривую  $(\varrho)$  вероятности того, что субъект болен раком только по числу, выдаваемому устройством (математически:  $\Phi\text{PB}_{\text{ЕЦ}} / (\Phi\text{PB}_{\text{ЕЦ}} + \Phi\text{PB}_{\text{НЦ}})$ , где  $\Phi\text{PB}_{\text{ЕЦ}}$  — это функция распределения вероятности события «Есть цель»,  $\Phi\text{PB}_{\text{НЦ}}$  — функция распределения вероятности события «Нет цели»).

Если мы остановим анализ в этом пункте, то сделаем одну из наиболее распространённых (и серьёзных) ошибок в обнаружении цели. Для того чтобы сделать кривую  $(\varrho)$  содержательной, должна всегда приниматься в расчёт другая исходная информация. Это предполагаемое отношение числа событий «Есть цель» к числу событий «Нет цели», характерное для тестируемой популяции. Например, нам может быть известно, что из тысячи человек болен раком один, которого мы пытаемся обнаружить. Чтобы учесть этот факт, значения  $\Phi\text{PB}$  события «Нет цели» на  $(\varrho)$  устанавливаются так, чтобы площадь под кривой была равна 0.999. Кроме того, значения  $\Phi\text{PB}$  события «Есть цель» устанавливаются таким образом, чтобы площадь под кривой вероятности стала равной 0.001. Как и прежде,  $(\varrho)$  отображает результат вычислений вероятности того, что пациент болен раком.

Пренебрежение этой информацией — серьёзная ошибка, потому что она сильно влияет на интерпретацию результатов тестирования. Другими словами, кривая на  $(\varrho)$  радикально изменяется, когда учитывается эта предварительная информация. Например, т.к. только один человек из тысячи болен раком (0.001 от всех тестируемых), то результат тестирования, равный 15, соответствует 0.025%-й вероятности того, что пациент имеет рак. Это сильно отличается от 25%-й вероятности, найденной при первоначальном анализе.

Этот метод преобразования выходного значения устройства тестирования в вероятность может быть полезен для понимания проблемы, но не является главным способом обнаружения цели. Большинство применений требует принятия решения «да»/«нет» о присутствии цели, так как «да» приведёт к одним действиям, а «нет» — к другим. Это делается сравнением выходного значения устройства тестирования с порогом. Если выходное значение выше порога, то говорят, что тест *положительный*, указывающий на присутствие цели. Если выходное значение ниже порога, говорят, что тест *отрицательный*, т. е. цель отсутствует. В нашем примере отрицательный результат тестирования означает, что пациент здоров, и его можно отправить домой. Когда результат тестирования положительный, будет выполнено дополнительное тестирование, такое как, например, получение образца ткани введением иглы для биопсии.

Так как распределения событий «Есть цель» и «Нет цели» перекрываются, некоторые результаты тестирования будут неточны. То есть некоторые пациенты, отправленные домой, в действительности больны раком, а некоторые пациенты, отправленные на дополнительное тестирование, в действительности здоровы. С позиции теории обнаружения правильное решение называется *истиной*, а неправильное — *ложью*. Например, если пациент болен раком и тест правильно обнаружил это состояние, говорят, что это *истина положительная*. Аналогично, если пациент не болен раком и тест показывает на его отсутствие, говорят, что это *истина отрицательная*. *Ложь положительная* появляется, когда пациент не болен раком, но тест ошибочно ставит диагноз «да». Этот результат вызывает напрасную тревогу и боль, а также расходы на дополнительное тестирование. Ещё худший сценарий случается при *ложном отрицании*, когда пациент болен раком, а

тест указывает, что он здоров. К сожалению, если рак не лечить, это может вызвать много проблем со здоровьем, включая и преждевременную смерть.

Человеческие страдания, являющиеся результатом этих двух типов ошибок, делают пороговую селекцию деликатным и требующим особой сбалансированности процессом. Как много ложных положительных результатов можно допустить, чтобы снизить число ложных отрицательных? Рис. 26.2 показывает графический метод оценки этой проблемы: *рабочую характеристику приёмника (РХП)*. Кривая РХП показывает процент событий, ошибочно принятых за положительные (чем ниже, тем лучше) для различных значений порога. Другими словами, каждая точка на кривой РХП представляет вероятную альтернативу истинно положительной и должно отрицательной характеристик.

**Рис. 26.2а...г** показывают положение порога в нашем примере с обнаружением рака. Например, посмотрите на **(б)**, где порог установлен на уровне 17. Каждое тестирование, которое даёт выходное значение выше порога, признаётся имеющим положительный результат. Около 13% площади под кривой распределения вероятности события «Нет цели» выше порога (т. е. справа от порога). Из всех пациентов, у которых нет рака, 87% имеют отрицательный результат (т. е. истинно отрицательный), а 13% будут признаны положительными (т. е. должно положительными). Для сравнения: около 80% площади под кривой функции распределения события «Есть цель» выше порога. Это значит, что 80% тех, кто болен раком, будут иметь положительный результат (т. е. истинно положительный). Другие 20% пациентов, имеющих рак, будут неправильно признаны здоровыми (т. е. результат можно отрицательный). Как показано на кривой РХП **(б)**, этот порог даёт точку на кривой: [положительные события «Нет цели»] = 13% и [положительные события «Есть цель»] = 80%.

Чем более эффективен процесс обнаружения, тем более круто изгибаются кривая РХП по направлению к верхнему левому углу графика. Если кривая представлена диагональной линией, идущей под углом  $45^\circ$ , то это соответствует простому угадыванию с вероятностью 50 на 50%. Установка относительно низкого порога, как показано на **(а)**, приводит к тому, что почти все цели обнаруживаются. Это достигается ценой большого уровня ложных тревог (ложный положительный). Как иллюстрируется на **(г)**, установка относительно высокого порога даёт обратную ситуацию: низкий уровень ложных тревог, но много пропущенных целей.

Эти методы анализа полезны для понимания последствий выбора порога, но окончательное решение принимается на основе того, что могут допустить люди. Предположим, что вы первоначально установили порог аппаратуры обнаружения рака на некоторое значение, которое вам кажется вполне обоснованным. После того как многие пациенты прошли исследование, вы поговорили с десятком пациентов, результат тестирования которых был признан должно положительным. Разговоры о том, как исследования на вашей системе усложняют жизнь этих людей, так сильно воздействуют на вас, что вы повышаете порог. Затем вы сталкиваетесь с ситуацией, которая ещё более негативно отражается на вас: вы сообщаете пациенту, который действительно болен раком, что ваша система тестирования не гарантирует правильный результат. В ответ на эту тяжёлую ситуацию вы значительно снижаете порог. Такие события повторяются не раз, и порог постепенно сдвигается к уравновешенному значению. То есть должно положительная оценка, умноженная на коэффициент значимости (снижающий порог), уравновешивает-

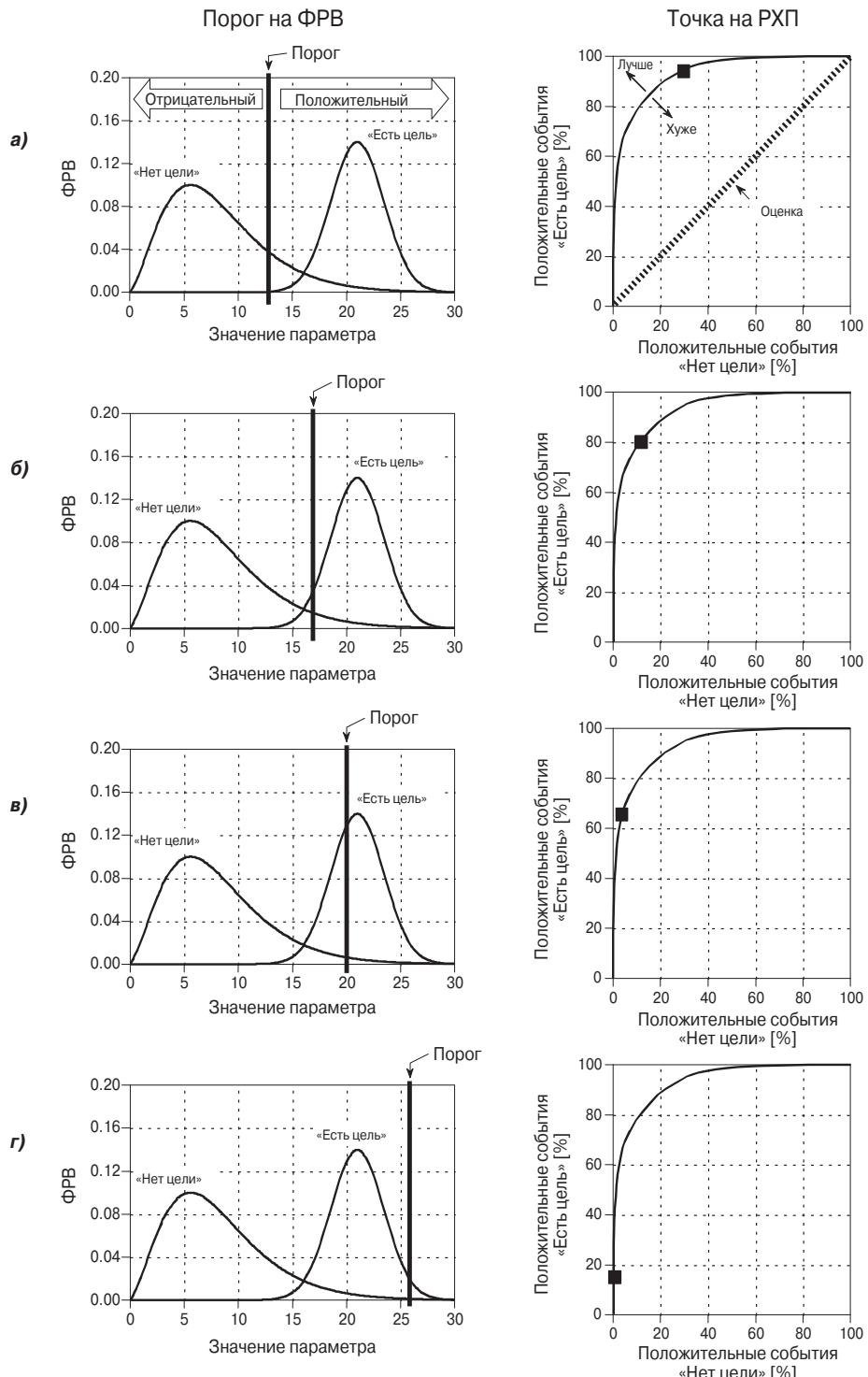


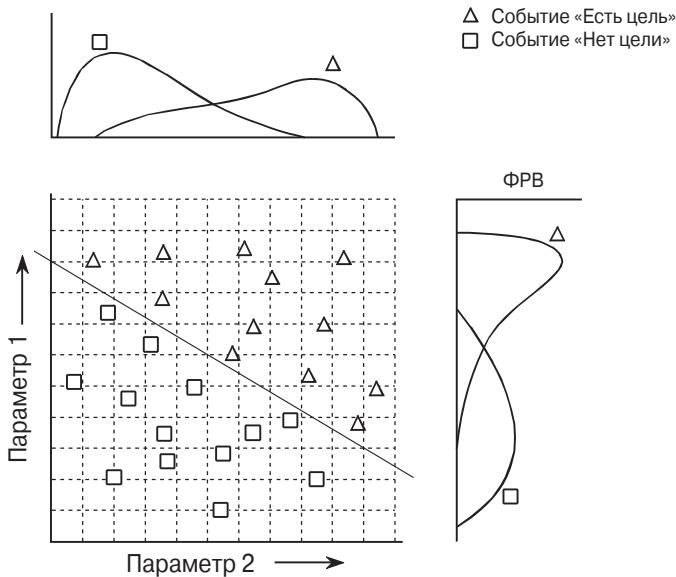
Рис. 26.2 Соотношения между кривыми РХП и ФРВ.

ся должно отрицательной оценкой, умноженной на другой коэффициент значимости (увеличивающий порог).

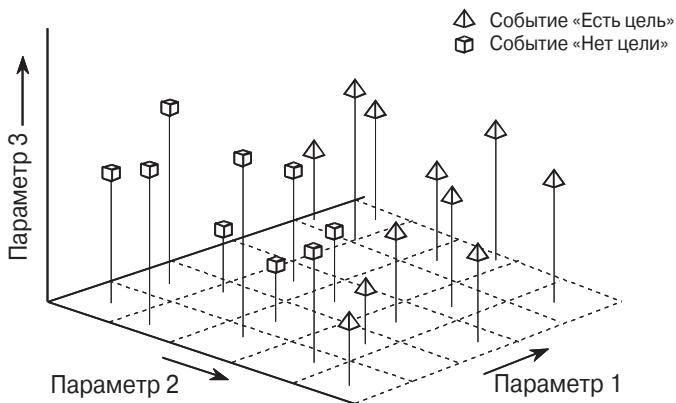
Рассмотренный анализ можно распространить на устройства, которые имеют более чем один выход. Например, предположим, что система диагностики рака работает на основе рентгеновского снимка пациента и автоматически анализирует изображение с целью выявления опухоли. Алгоритмы идентифицируют подозрительную область и затем измеряют ключевые характеристики, которые влияют на окончательную оценку. Например, предположим, что мы измерили диаметр подозрительной области (параметр 1) и её яркость на изображении (параметр 2). Далее предположим, что наше исследование показывает, что опухоль вообще больше и ярче, чем нормальный образец. Сначала мы можем применить ранее представленный РХП-анализ для каждого параметра и найти приемлемый для них порог. Затем можно классифицировать тест как положительный, только если он отвечает двум критериям: параметр 1 больше некоторого порога и параметр 2 больше другого порога.

Этот метод раздельного применения порога к каждому параметру и применения затем логических функций (И, ИЛИ и т. п.) широко распространён. Тем не менее он очень неэффективен. **Рис. 26.3** показывает, почему это так. На этом рисунке каждый треугольник соответствует появлению одного события «Есть цель» (у пациента рак), изображённого в том месте, которое определяют значения двух её параметров. Каждый квадрат соответствует появлению одного события «Нет цели» (у пациента нет рака). Как показано на графике ФРВ, имеет место большое перекрытие между распределениями событий «Есть цель» и «Нет цели». Другими словами, каждый параметр, взятый индивидуально, является плохим предсказателем рака. Комбинируя два параметра с помощью простых логических функций, можно обеспечить только небольшое улучшение. Это особенно интересно, так как два параметра вместе содержат информацию, достаточную для совершенного разделения событий «Есть цель» и «Нет цели». На **Рис. 26.3** это иллюстрируется проведением диагональной линии между двумя группами событий.

Такой тип системы координат называется «параметрическое пространство». Например, двумерная плоскость в этом примере может называться пространство «диаметр-яркость». Идея состоит в том, что события «Есть цель» будут занимать одну область параметрического пространства, а события «Нет цели» — другую. Раздел между двумя областями может быть простым, как прямая линия, или это может быть замкнутая кривая сложной формы. **Рис. 26.4** показывает следующий уровень сложности: трёхпараметрическое пространство, представленное тремя осями  $x$ ,  $y$ ,  $z$ . Например, это может соответствовать системе диагностирования рака, которая измеряет диаметр, яркость и некоторый третий параметр, скажем, резкость границы. Как и в двумерном случае, идея заключается в том, что группы событий «Есть цель» и «Нет цели» (можно надеяться) займут разные области пространства, позволяя разделить их на две. В трёх измерениях области разделяются плоскостями и поверхностями. Термин *гиперпространство* (сверх, выше или позади обычного пространства) часто используется для описания параметрического пространства с более чем тремя измерениями. Математически гиперпространство не отличается от пространств с одним, двумя и тремя измерениями. Однако на практике нет возможности изобразить их в графической форме в трёхмерном пространстве.



**Рис. 26.3** Пример двухпараметрического пространства. Событие «Есть цель» ( $\Delta$ ) и событие «Нет цели» (□) группируются раздельно в двухпараметрическом пространстве, однако они перекрываются по каждому отдельному параметру. Это перекрытие показано одномерной ФРВ вдоль оси каждого параметра.



**Рис. 26.4.** Пример трёхпараметрического пространства. Так же, как двухпараметрическое пространство формируется на плоскости, трёхпараметрическое пространство может быть графически представлено с использованием обычных осей  $x$ ,  $y$ ,  $z$ . Разделение трёхпараметрического пространства на области производится плоскостями или кривыми поверхностями.

Выбор порога для задачи с одним параметром не может (обычно) быть определён как верный или неверный, потому что каждое пороговое значение приводит к уникальной комбинации ложных положительных и ложных отрицательных решений, т. е. некоторой точке на кривой РХП. А это всего лишь обмен одной цели на другую, и такой подход не даст абсолютно точного ответа. С другой стороны, пространства с двумя или более параметрами могут иметь неправильный раз-

дел между областями. Например, предположим, что увеличение числа точек данных на Рис. 26.3 позволяет обнаружить малое перекрытие между группами событий «Есть цель» и «Нет цели». Должна быть возможность передвигать пороговую линию между двумя группами событий, чтобы осуществлять обмен между должно положительными и должно отрицательными решениями. То есть диагональная линия должна иметь возможность перемещаться вверх вправо или вниз влево. Однако было бы неверно поворачивать линию, потому что это увеличивало бы ошибки обоих типов.

Как подсказывают эти примеры, удобным путём к обнаружению события «Есть цель» (иногда называемым распознаванием образов) является двухшаговый процесс. Первый шаг называется *выбор признаков*. Он использует алгоритмы для уменьшения множества данных до нескольких параметров, таких как диаметр, яркость, резкость границ и т. п. Эти параметры часто называются *признаками* или *классификаторами*. Выбор признаков нужен для уменьшения количества данных. Например, медицинские рентгеновские изображения содержат более миллиона пикселей. Целью выбора признаков является извлечение информации в более концентрированном и управляемом виде. Разработка алгоритмов этого типа является скорее искусством, чем наукой. Она придаёт большое значение практике и искусству видеть проблему и говорит: «Это те признаки, по которым лучше всего можно определить основную информацию». Здесь важную роль играет метод проб и ошибок.

На втором шаге выполняется оценка признаков, чтобы определить, какое событие имеет место. Другими словами, используются некоторые методы, чтобы разделить параметрическое пространство на области, которые соответствуют событию «Есть цель», и области, которые соответствуют событию «Нет цели». Это прямая для одно- или двухпараметрических пространств. Известные точки данных отмечаются на графике (так, как на Рис. 26.3), и область разделяется «на глаз». Такое разделение описывается в компьютерной программе уравнением или каким-либо другим способом, определяющим эти области. В принципе, такой же метод можно применить к трёхмерному параметрическому пространству. Проблема в том, что трёхмерные графики очень трудны для восприятия человека (см. Рис. 26.4).

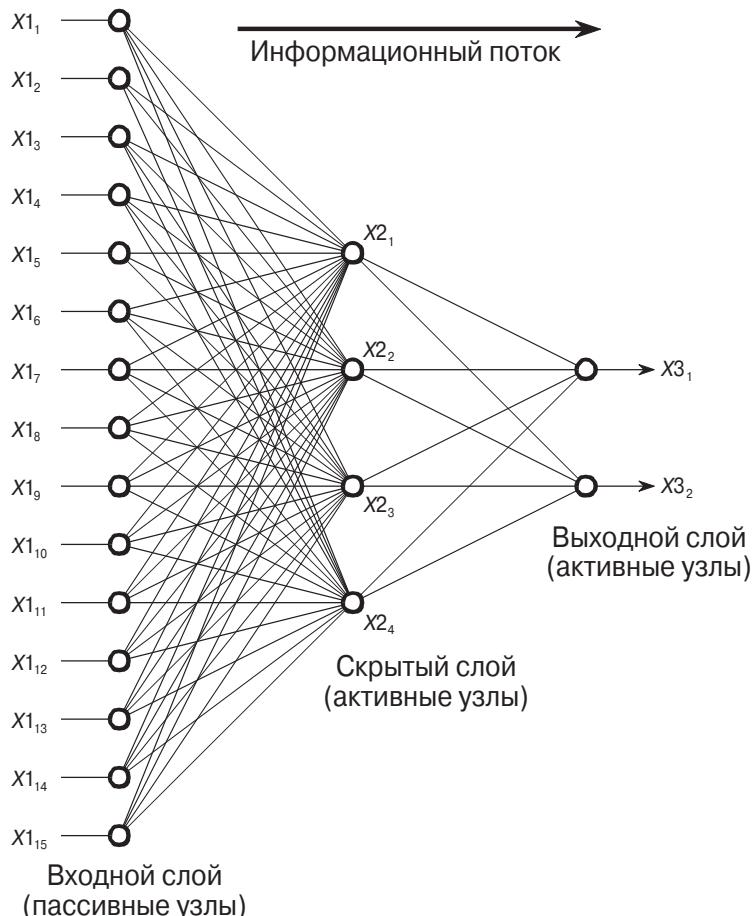
Одним словом, необходим механизм, способный выполнять разделение много-параметрического пространства в соответствии с заданными требованиями. Эта идеальная система обнаружения цели в высшей степени близка к главной теме данной главы — *нейронным сетям*.

## 26.2. Архитектура нейронных сетей

Люди и животные обрабатывают информацию с помощью нейронных сетей, которые сформированы миллиардами *нейронов* (нервных клеток), обменивающихся короткими электрическими импульсами. Эти импульсы называют *биоэлектрическими потенциалами*. Компьютерные алгоритмы, которые имитируют эти биологические структуры, формально называют *искусственными нейронными сетями*, чтобы отличать их от подобных биологических систем. Однако большинство учёных и инженеров используют термин *нейронные сети* по отношению к биологическим, и к небиологическим системам.

Исследование нейронных сетей мотивируется двумя желаниями: лучше понять принципы работы мозга человека и разработать компьютер, который будет работать с абстрактными и плохо обусловленными задачами. Например, обычный компьютер плохо «понимает» речь и распознаёт человеческие лица, в то время как люди очень хорошо решают эти задачи.

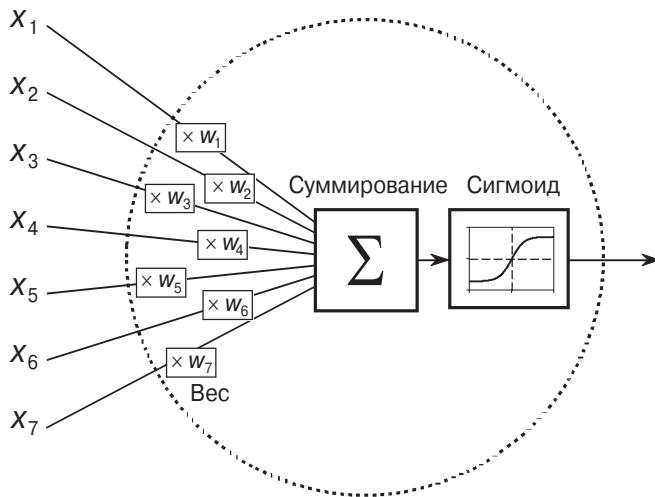
Были испытаны многие структуры нейронных сетей. Некоторые основаны на имитации того, что видит под микроскопом биолог, другие — больше на математическом анализе процессов. Наиболее распространённая структура нейронной сети показана на Рис. 26.5. Она сформирована из трёх слоев, называемых входным, скрытым и выходным слоями соответственно. Каждый слой состоит из одного или более узлов, представленных на этой диаграмме маленькими кружками. Линии между узлами показывают поток информации от одного узла к другому.



**Рис. 26.5.** Архитектура нейронной сети. Это самая общая структура нейронной сети: три слоя с полной взаимосвязью. Узлы входного слоя пассивные, не изменяющие данные, но ретранслирующие значения от своего единственного входа на многие выходы. Для сравнения: узлы скрытого и выходного слоёв активны и изменяют сигналы в соответствии с Рис. 26.6. Действие этой нейронной сети определено весами, применёнными в скрытом и выходном слоях.

В некоторых типах нейронных сетей информация передаётся только от входа к выходу (т. е. слева направо). Другие типы нейронных сетей имеют более сложные связи, включая обратные.

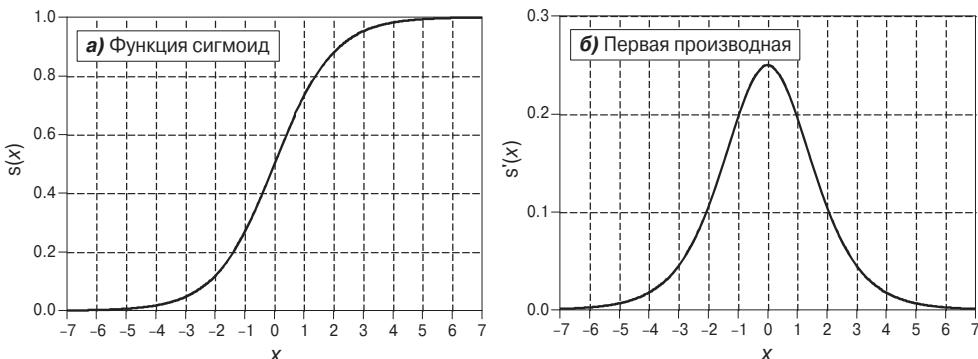
Узлы входного слоя пассивны, т. е. не изменяют данные. Они принимают сигнал со своего входа и передают его на множество выходов. Для сравнения: узлы скрытого и выходного слоёв активны. Это значит, что они изменяют данные, как показано на **Рис. 26.6**. Переменные  $X_{11}, X_{12}, \dots, X_{15}$  содержат данные, которые должны оцениваться (см. **Рис. 26.5**). Например, они могут быть значениями пикселей изображения, выборками из аудиосигнала, ценами на фондовом рынке за какой-либо период времени и т. п. Они также могут быть выходными параметрами какого-нибудь алгоритма, например, признаками в нашем примере с диагностированием рака: диаметр, яркость, резкая граница и т. п.



**Рис. 26.6.** Активный узел нейронной сети. Это диаграмма активных узлов, используемых скрытым и выходным слоями нейронной сети. Каждый вход умножается на весовой коэффициент ( $w_n$ ) и затем суммируется. Так формируется одно значение, которое проходит через нелинейный элемент с передаточной функцией  $s$ -образной формы, называемый сигмоидом. Функция сигмоид показана в деталях на **Рис. 26.7**.

Каждое значение входного слоя посыпается ко всем скрытым узлам. Это называется полностью взаимосвязанной структурой. Как показано на **Рис. 26.6**, значения, переданные в скрытый узел, умножаются на весовые коэффициенты — множество предварительно определённых чисел, хранящихся в программе. Взвешенные входы затем складываются для получения одного результата. Это показано на рисунке символом « $\Sigma$ ». Прежде чем оставить узел, это число проходит через нелинейную математическую функцию, называемую сигмоид. Она имеет форму « $s$ », которая ограничивает выход узла. То есть вход сигмоида — это значение числа  $-\infty \dots +\infty$ , а его выход может принимать только значения  $0 \dots 1$ .

Выходы скрытого слоя представлены в диаграмме (**Рис. 26.5**) переменными  $X_{21}, X_{22}, X_{23}, X_{24}$ . Как и прежде, каждое из этих значений передаётся к следующему



**Рис. 26.7.** Функция сигмоид и её производная. Выражения (26.1) и (26.2) описывают эти функции.

му слою. Активные узлы выходного слоя объединяют и изменяют данные для получения двух выходных значений этой сети:  $X_3_1$  и  $X_3_2$ .

Нейронные сети могут иметь любое число слоёв и любое число узлов в каждом слое. Большинство приложений используют трёхслойную структуру, включающую до нескольких сот входных узлов. Скрытый слой обычно имеет размер около 10% от размера входного слоя. В случае обнаружения цели выходной слой должен иметь только один узел. Выход этого узла является порогом для обеспечения положительной или отрицательной индикации присутствия или отсутствия цели во входных данных.

**Программа 26.1** реализует диаграмму, представленную на **Рис. 26.5**. Ключевой момент состоит в том, что эта архитектура очень простая и очень обобщённая. Такая диаграмма может использоваться для многих задач, несмотря на их частные особенности. Способность нейронных сетей обеспечивать полезные манипуляции с данными лежит в правильном выборе весов. Такой подход существенно отличается от обычного порядка обработки информации, когда поиск результата описывается пошаговыми процедурами.

В качестве примера рассмотрим нейронную сеть для распознавания объектов в гидроакустическом сигнале. Предположим, что 1000 выборок этого сигнала запоминаются в компьютере. Как сделать так, чтобы компьютер определил, что представляют эти данные: подводную лодку, кита, подводную гору или совсем ничего? Обычная ЦОС может подойти к решению этой задачи с позиции математических методов и алгоритмов, таких как корреляция и анализ спектра частот. В нейронных сетях 1000 выборок просто подаются на входной слой, а результирующее значение поступает из выходного слоя. Выбором подходящих весов можно сконфигурировать выход так, чтобы обеспечить получение широкого диапазона информации. Например, это могут быть выходы для следующих объектов: подводная лодка (да/нет), кит (да/нет), подводная гора (да/нет) и т. п.

При использовании других весов выходы сети могут классифицировать объекты как: металл или не металл, биологический или не биологический, вражеский или свой и т. п. Никаких алгоритмов, никаких правил, только отношения между входом и выходом, определяемые значениями выбранных весов.

**Программа 26.1**

```

100 'НЕЙРОННАЯ СЕТЬ (ДЛЯ СТРУКТУРНОЙ СХЕМЫ НА РИС. 26.5)
110
120 DIM X1[15]      'хранит входные значения
130 DIM X2[4]        'хранит значения выхода скрытого слоя
140 DIM X3[2]        'хранит значения выхода выходного слоя
150 DIM WH[4,15]    'хранит веса скрытого слоя
160 DIM W0[2,4]     'хранит веса выходного слоя
170 '
180 GOSUB XXXX      'некоторая подпрограмма для загрузки X1[ ] из входных данных
190 GOSUB XXXX      'некоторая подпрограмма для загрузки весов WH[ , ] и W0[ , ]
200 '
210 '                Найти значения узлов X2[ ]
220 FOR J% = 1 TO 4 'цикл по каждому узлу скрытого слоя
230 ACC = 0          'очистить аккумуляторную переменную ACC
240 FOR I% = 1 TO 15 'взвешивать и суммировать каждый входной узел
250 ACC = ACC + X1[I%] * WH[J%,I%]
260 NEXT I%
270 X2[J%] = 1 / (1 + EXP(-ACC)) 'пропустить суммируемое значение через сигмоид
280 NEXT J%
290 '
300 '                НАЙТИ ЗНАЧЕНИЯ ВЫХОДНЫХ УЗЛОВ X3[ ]
310 FOR J% = 1 TO 2 'цикл по каждому узлу выходного слоя
320 ACC = 0          'очистить АККУМУЛЯТОРНУЮ ПЕРЕМЕННУЮ ACC
330 FOR I% = 1 TO 4 'взвешивать и суммировать каждый узел скрытого слоя
340 ACC = ACC + X2[I%] * W0[J%,I%]
350 NEXT I%
360 X3[J%] = 1 / (1 + EXP(-ACC)) 'пропустить суммируемые значения через сигмоид
370 NEXT J%
380 '
390 END

```

**Рис. 26.7** показывает вид функции сигмоида, математически описываемой уравнением:

$$s(x) = 1/(1+e^{-x}). \quad (26.1)$$

Функция сигмоид. Она используется в нейронных сетях как гладкий порог.

Эта функция представлена на графике **Рис. 26.7**.

Точная форма сигмоида не важна, важно только то, что это гладкий порог. Для сравнения: простой порог даёт значение 1, когда  $x > 0$ , и значение 0, когда  $x < 0$ . Сигмоид формирует такую же основную пороговую функцию, но только дифференцируемую, как показано на **Рис. 26.7б**. Хотя производная не использовалась в диаграмме (**Рис. 26.5**), она является основным способом нахождения подходящих весов. Далее расскажем о ней кратко. Достоинство сигмоида в том, что её производная вычисляется очень просто:

$$s'(x) = s(x) [1 - s(x)]. \quad (26.2)$$

Первая производная функции сигмоид.

Она вычисляется, используя значение самой функции.

Например, если  $x = 0$ , то  $s(x) = 0.5$  (в соответствии с выражением (26.1), а первая производная вычисляется как  $s'(x) = 0.5(1 - 0.5) = 0.25$ . Выражение (26.2), являющееся производной от выражения (26.1), используется для упрощения вычислений.

Не будет ли нейронная сеть более гибкой, если сигмоид получит возможность смещаться влево или вправо, так что его центр окажется в некоторой точке, отличной от  $x = 0$ ? Ответ — да, и большинство нейронных сетей позволяют это выполнять. Это делается очень просто: к входному слою добавляется дополнительный узел со значением входного сигнала, всегда равным 1. Когда он умножается на весовой коэффициент в скрытом слое, это обеспечивает смещение каждому сигмоиду. Дополнительный узел называется узлом смещения. Он идентичен другим узлам сети, за исключением того, что имеет постоянное единичное входное значение.

Может ли нейронная сеть быть создана без сигмоида или подобной нелинейности? Чтобы ответить на этот вопрос, посмотрите на трёхслойную сеть Рис. 26.5. Если сигмоида нет, три слоя превращаются только в два слоя. То есть суммирование и взвешивание в скрытом и выходном слоях будут объединены в один слой, давая в результате только двухслойную сеть.

## 26.3. Почему это работает?

Веса, необходимые, чтобы нейронная сеть выполняла конкретные задачи, находят с помощью *алгоритма обучения* на примерах того, как система должна работать. Применительно к гидроакустическим системам база данных может состоять из нескольких сот (или больше) сегментов по 1000 выборочных значений. Некоторые из этих сегментов могут соответствовать подводным лодкам, другие — китам или случайному шуму и т. п. Обучающий алгоритм использует эти примеры, чтобы вычислить веса, соответствующие текущей задаче. Термин *обучающий* широко используется для описания нейронных сетей. Однако можно дать лучшее описание путём определения оптимального множества весов, основанного на статистике примеров. Несмотря на то что метод определён, полученные веса ничего не дают для его понимания. Можно иметь нейронную сеть, в которой использованы веса, определяющие отношения между входом и выходом, но почему эти конкретные веса работают, совершенно непонятно. Это «мистическое» качество нейронных сетей заставляет многих учёных и инженеров относиться к ним с большой осторожностью. Вспомните многочисленные научно-фантастические фильмы о том, как машины восстали против людей и захватили Землю.

В ответ на это раздаются голоса сторонников нейронных сетей: «Нейронные сети просты и понятны». Чтобы проверить это заявление, покажем сначала, что можно подобрать веса нейронной сети с помощью традиционных методов ЦОС. Далее продемонстрируем, что обучающие алгоритмы обеспечивают лучшее решение, чем традиционные методы. Хотя до конца необъяснимо, почему конкретное множество весов даёт ожидаемый результат, метод вызывает доверие.

С наиболее искусшённой точки зрения нейронная сеть — это метод разметки различных областей в параметрическом пространстве. Например, рассмотрим нейронную сеть гидроакустической системы с 1000 входами и единственным выходом. При выборе подходящих весов выход системы должен быть близок к единице, если входной сигнал представляет собой эхо от подлодки, и близок к нулю,

если входной сигнал — это только шум. Это формирует параметрическое гиперпространство из 1000 измерений. Нейронная сеть представляет собой метод определения соответствующего значения каждому положению в этом гиперпространстве. То есть 1000 входных значений определяют положение в гиперпространстве, а выход нейронной сети присваивает значение этому положению. Таблица преобразования может правильно решить эту задачу, имея сохранённое выходное значение для каждого возможного входного адреса. Разница состоит в том, что нейронная сеть вычисляет значение для каждого положения (адреса), а не запоминает невероятно большое количество значений. В действительности архитектура нейронной сети часто оценивается по тому, как хорошо она разделяет гиперпространство для данного числа весовых коэффициентов.

Этот подход — ключ к нахождению количества узлов в скрытом слое. Параметрическое пространство  $N$  измерений требует  $N$  чисел для определения местоположения. Идентификация области в гиперпространстве требует  $2N$  значений (т. е. минимальное и максимальное значения по каждой оси определяют гиперпространственный прямоугольный параллелепипед). Например, эти простые соотношения указывают, что нейронная сеть с 1000 входов требует 2000 весов для отделения одной области гиперпространства от другой. В полностью взаимосвязанной сети потребуется два скрытых узла. Количество необходимых областей зависит от конкретной задачи, но можно ожидать, что оно будет много меньше, чем число измерений в параметрическом пространстве. Хотя это только грубая аппроксимация, она объясняет, почему большинство нейронных сетей может работать со скрытым слоем, размер которого составляет от 2 до 30% размера входного слоя.

Совершенно другой способ понимания нейронных сетей даёт концепция корреляции. Как описано в Главе 7, корреляция является оптимальным способом обнаружения в сигнале известного образа. Корреляция определяется умножением сигнала на известный образец и сложением полученных произведений. Чем больше сумма, тем больше сигнал похож на эталон. Далее рассмотрим Рис. 26.5 и поразмыслим над каждым скрытым узлом и над тем, как выглядит этот специфичный объект во входных данных. Каждый скрытый узел вычисляет корреляцию входных данных со множеством весов, связанных с этим скрытым узлом. Если известный образ присутствует, то сумма, проходящая к сигмоиду, будет достаточно большой, в противном случае она будет маленькой.

С этой точки зрения действие сигмоида очень интересно. Посмотрим на Рис. 26.1г и заметим, что вероятностная кривая, отделяющая два колоколообразных распределения, похожа на сигмоид. Если мы будем вручную проектировать нейронную сеть, мы можем сделать выход каждого скрытого узла условной вероятностью того, что конкретный образ представлен во входных данных. Выходной слой повторяет эту операцию, делая выходную трёхслойную структуру сетью, которая выполняет корреляцию корреляций.

Традиционная ЦОС основана на двух методах: свёртке и анализе Фурье. Нейронные сети могут выполнить обе эти операции, а также многие другие. Рассмотрим сигнал из  $N$  выборок, который отфильтрован для создания другого сигнала из  $N$  выборок. В соответствии с описанием свёртки со стороны выхода каждая выборка выходного сигнала — это взвешенная сумма входных выборок. Далее рассмотрим двухслойную нейронную сеть с  $N$  узлами в каждом слое. Значение, созданное каждым узлом выходного слоя, — это также взвешенная сумма вход-

ных значений. Если каждый узел выходного слоя использует такие же веса, как все другие выходные узлы, сеть будет выполнять линейную свёртку. Аналогично ДПФ можно вычислить с помощью двухслойной нейронной сети с  $N$  узлами в каждом слое. Каждый узел выходного слоя находит амплитуду одной частотной составляющей. Это делается установкой весов каждого узла выходного слоя равными значениям искомой синусоиды. Результирующая сеть вычисляет корреляцию входного сигнала с каждой базисной синусоидальной функцией, вычисляя таким образом ДПФ. Конечно, двухслойная нейронная сеть намного менее мощная, чем стандартная трёхслойная архитектура. Это значит, что нейронная сеть может выполнять нелинейную обработку так же хорошо, как и линейную.

Предположим, что эти традиционные алгоритмы ЦОС используются для нахождения весов нейронной сети. Можно ли утверждать, что сеть оптимальна? Традиционные алгоритмы ЦОС обычно основаны на некоторых предположениях о характеристиках входного сигнала. Например, винеровская фильтрация оптимальна с позиции получения максимального отношения сигнал/шум в предположении, что спектры сигнала и шума известны; корреляция является оптимальной для обнаружения целей в предположении, что шум белый; деконволюция (обращение свёртки) «препятствует» нежелательной свёртке, предполагая, что ядро деконволюции является инверсией первоначального ядра свёртки, и т. д. Проблема в том, что учёные и инженеры редко имеют полную информацию о входных сигналах и шумах. Хотя соответствующая математика может быть элегантной, все достижимые возможности ограничиваются тем, насколько адекватно интерпретируются данные.

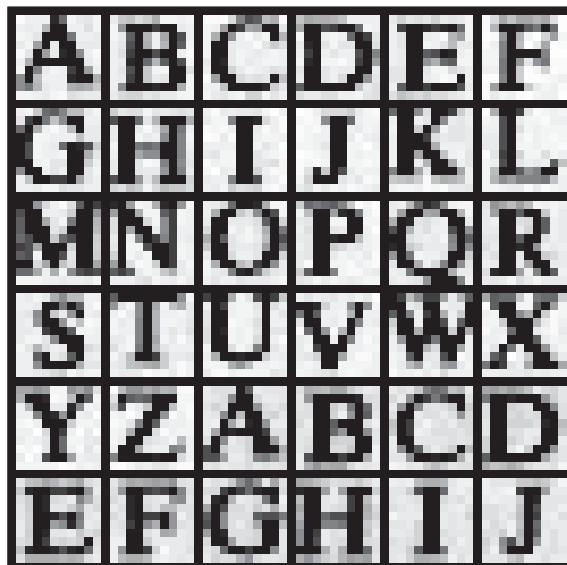
Для примера рассмотрим тестирование традиционных алгоритмов ЦОС с помощью реальных входных сигналов. Затем повторим тест со слабо изменённым алгоритмом, скажем, увеличив один из параметров на один процент. Если второй тест будет иметь результат лучше, чем первый, то исходный алгоритм не является оптимальным для данной задачи. Почти все традиционные алгоритмы могут быть значительно улучшены на основе анализа небольших изменений параметров алгоритмов и процедур методом проб и ошибок. Это и есть стратегия нейронных сетей.

## 26.4. Обучение нейронной сети

Конструкцию нейронной сети можно объяснить с помощью примера. Рис. 26.8 показывает решаемую задачу: идентификацию отдельных букв в изображении текста. Эта задача распознавания образов привлекла большое внимание. Многие методики частично достигли успеха, но, к сожалению, нет идеального решения. Многие успешные коммерческие продукты были основаны на решении этой проблемы: чтение адресов на письмах для маршрутизации почты, ввод документов в компьютеры и т. д.

Первым шагом в разработке нейронной сети является создание обучающей базы данных. Для задачи распознавания текста это выполняется печатью 26 заглавных букв: A, B, C, D, ..., Z 50 раз на листе бумаги. Далее эти 1300 букв преобразуются в цифровое изображение с помощью сканера, подключённого к компьютеру. Цифровое изображение большого размера затем делится на малые изображения  $10 \times 10$  пикселей, каждое содержащее одну букву. Эта информация хранится как

1.3МБ-я база данных: 1300 изображений, 100 пикселей на изображение, 8 бит на пиксель. Будем использовать первые 260 изображений в этой базе данных для обучения нейронной сети (т. е. для определения весов), а остальные — для тестирования её характеристик. База данных должна также содержать способ идентификации букв, содержащихся в каждом изображении. Например, к каждому изображению  $10 \times 10$  пикселей может быть добавлен ASCII-код этой буквы. В другой схеме на то, какая это буква, может указывать положение каждого изображения  $10 \times 10$  пикселей в базе данных. Например, изображения 0...49 могут быть буквой «А», изображения 50...99 — буквой «В» и т. д.



**Рис. 26.8.** Пример изображения текста. Идентификация букв в изображении текста — это классическая проблема распознавания образов. В этом примере каждая буква содержится в изображении размером  $10 \times 10$  пикселей с 256 уровнями серого на пиксель. База данных используется для обучения и тестирования образца нейронной сети, состоящей из 50 множеств по 26 букв, всего 1300 изображений. Изображения, показанные здесь, являются частью базы данных.

С целью демонстрации была разработана нейронная сеть для следующей задачи: определить, какое из изображений  $10 \times 10$  пикселей содержит гласную букву, т. е. А, Е, И, О или У. Эта задача не может иметь какого-либо практического приложения, но иллюстрирует способность нейронной сети решать очень абстрактные задачи распознавания образов. Включив десять примеров каждой буквы в обучающее множество, сеть будет изучать ключевые свойства, которые отличают выбранные изображения от остальных.

Нейронная сеть, используемая в этом примере, — это традиционная трёхслойная полностью связанный архитектура, как показано на **Рис. 26.5** и **26.6**. Имеется 101 узел во входном слое (100 значений пикселей плюс узел смещения), 10 узлов в скрытом слое и 1 узел в выходном слое. Когда на вход сети подаётся 100-пиксельное изображение, мы хотим, чтобы выходное значение было близко к единице, если присутствует гласная, и близко к нулю, если гласная отсутствует. Не обращайте внимание на то, что входной сигнал был принят как двумерный

массив ( $10 \times 10$ ), хотя вход нейронной сети — это одномерный массив. Это ваш взгляд на то, как взаимосвязаны значения пикселей. Нейронная сеть будет находить связи в её собственном понимании.

**Программа 26.2** представляет собой главную программу для вычисления весов нейронной сети. Она работает совместно с **Программой 26.3**, в которой содержатся три подпрограммы, вызываемые из главной программы. Элементы массива  $X1[1] \dots X1[100]$  хранят значения входного слоя. В дополнение  $X1[101]$  всегда хранит значение 1, обеспечивая входной слой узлом смещения. Выходные значения от скрытого слоя содержатся в элементах массива  $X2[1] \dots X2[10]$ . Переменная  $X3$  содержит выходное значение сети. Веса скрытого слоя содержатся в массиве  $WH[,]$ , где первый индекс указывает скрытый узел (1...10), а второй индекс — узел входного слоя (1...101). Веса выходного слоя хранятся в  $WO[1] \dots WO[10]$ . Так создаётся 1020 значений весов, которые и определяют, как сеть будет работать.

### Программа 26.2

```

100 'ОБУЧЕНИЕ НЕЙРОННОЙ СЕТИ (Определение весов)
110 '
120                               'ИНИЦИАЛИЗАЦИЯ
130 MU = .000005                 'размер шага итерации
140 DIM X1[101]                  'хранит сигнал + смещение входного слоя
150 DIM X2[10]                    'хранит сигнал скрытого слоя
160 DIM WH[10,101]                'хранит веса скрытого слова
170 DIM WO[10]                    'хранит веса выходного слоя
180 '
190 FOR H% = 1 TO 10             'ИНИЦИАЛИЗАЦИЯ ВЕСОВ СЛУЧАЙНЫМИ ЗНАЧЕНИЯМИ
200 WO[H%] = (RND-0/5)          'веса выходного слоя: от -0.5 до 0.5
210 FOR I% = 1 TO 101            'веса скрытого слоя: от -0.0005 до 0.0005
220 WH[H%,I%] = (RND-0.5)/1000
230 NEXT I%
240 NEXT H%
250 '
260                               'ОСНОВНОЙ ЦИКЛ
270 FOR ITER% = 1 TO 800         'цикл по 800 итерациям
280 '
290 ESUM = 0                     'очистить аккумулятор ошибки ESUM
300 '
310 FOR LETTER% = 1 TO 260       'цикл по каждой букве в обучающем множестве
320 GOSUB 1000                   'загрузить  $X1[ ]$  с обучающим множеством
330 GOSUB 2000                   'найти ошибку для этой буквы
340 ESUM = ESUM + ELET^2        'накопить ошибку для этой итерации
350 GOSUB 3000                   'найти новые веса
360 NEXT LETTER%
370 '
380 PRINT ITER% ESUM           'отображать процесс на экране
390 '
400 NEXT ITER%
410 '
420 GOSUB XXXX                  'некоторая подпрограмма хранения весов
430 END

```

**Программа 26.3**

```
1000 'ПОДПРОГРАММА ЗАГРУЗКИ X1[ ] С ИЗОБРАЖЕНИЕМ ИЗ БАЗЫ ДАННЫХ
1010 'Переменная входа программы: LETTER%
1020 'Переменные выхода программы: X1[1] – X1[100], X1[101] = 1, CORRECT
1030 '
1040 'Переменная LETTER% со значением от 1 до 260 указывает,
    'какой образ из базы данных должен быть
1050 'возвращён в X1[1]..X1[100]. Узел смещения X1[101] всегда имеет значение 1.
1060 'Переменная CORRECT имеет значение 1, если возвращаемое изображение
    'является гласной буквой, и 0 – в противном случае
1070 '(Детали этой подпрограммы не важны и не приводятся здесь).
1900 RETURN

2000 'ПОДПРОГРАММА ВЫЧИСЛЕНИЯ ОШИБКИ С ТЕКУЩИМИ ВЕСАМИ
2010 'Переменные входа программы : X1[ ], X2[ ], WI[ , ], WH[ ], CORRECT'
2020 'Переменная выхода программы: ELET'
2030 '
2040 '          'НАЙТИ ЗНАЧЕНИЯ СКРЫТЫХ УЗЛОВ X2[ ] '
2050 FOR H% = 1 TO 10   'цикл по каждому скрытому узлу'
2060 ACC = 0            'очистить аккумулятор
2070 FOR I% = 1 TO 101  'взвесить и просуммировать каждый входной узел
2080 ACC = ACC + X1[I%] * WH[H%,I%]
2090 NEXT I%
2100 X2[H%] = 1 / (1 + EXP(-ACC) ) 'пропустить суммированное значение
    'через сигмоид
2110 NEXT H%
2120 '
2130 '          'НАЙТИ ВЫХОДНОЕ ЗНАЧЕНИЕ: X3
2140 ACC = 0            'очистить аккумулятор
2150 FOR H% = 1 TO 10   'взвесить и просуммировать каждый скрытый узел

2160 ACC = ACC + X2[H%] * WO[H%]
2170 NEXT H%
2180 X3 = 1 / (1 + EXP(-ACC) ) 'пропустить суммированное значение через сигмоид

2190 '
2200 '          НАЙТИ ОШИБКУ ДЛЯ ЭТОЙ БУКВЫ, ELET
2210 ELET = (CORRECT - X3) 'найти ошибку
2220 IF CORRECT = 1 THEN ELET = ELET*5 'задать дополнительный весовой
    'коэффициент для цели
2230 '
2240 RETURN

3000 'ПОДПРОГРАММА НАХОЖДЕНИЯ НОВЫХ ВЕСОВ
3010 Переменные входа программы: X1[ ], X2[ ], X3, WI[ , ], WH[ ], ELET, MU
3020 'Переменные выхода программы: WI[ , ], WH[ ]
3030 '
3040 '          НАЙТИ НОВЫЕ ВЕСА ДЛЯ СКРЫТЫХ УЗЛОВ
```

```

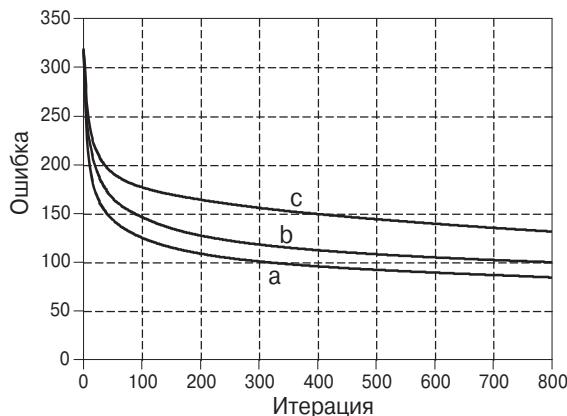
3050 FOR H% = 1 TO 10
3060 FOR I% = 1 TO 101
3070 SLOPEO = X3 * (1 - X3)
3080 SLOPEH = X2(H%) * (1 - X2[H%])
3090 DX3DW = X1[I%] * SLOPEH * WO[H%] * SLOPEO
3100 WH[H%,I%] = WH[H%,I%] + DX3DW * ELET * MU
3110 NEXT I%
3120 NEXT H%
3130 '
3140 'НАЙТИ НОВЫЕ ВЕСА ДЛЯ ВЫХОДНЫХ УЗЛОВ
3150 FOR H% = 1 TO 10
3160 SLOPEO = X3 * (1 - X3)
3170 DX3DW = X2(H%) * SLOPEO
3180 WO[H%] = WO[H%] + DX3DW * ELET * MU
3190 NEXT H%
3200 '
3210 RETURN

```

Первое действие программы — установить начальное значение каждого веса, используя генератор случайных чисел. Как показано в строках 190...240, веса скрытого слоя получают начальные значения между  $-0.0005$  и  $0.0005$ , а выходного слоя — между  $-0.5$  и  $0.5$ . Эти диапазоны выбраны так, чтобы амплитуда была такого же порядка, каким должен быть окончательный вес. Выбор основан: 1) на диапазоне значений входного сигнала; 2) на количестве входов, суммируемых в каждом узле и 3) на том, что диапазон значений перед каждым сигмоидом является активным (вход изменяется от  $-5$  до  $5$ , а выход — от  $0$  до  $1$ ). Например, когда 101 вход с заданным значением 100 умножается на типичный вес, равный  $0.0002$ , сумма произведений примерно равна 2, т. е. находится в активном диапазоне входа сигмоида.

Если оценивать характеристики нейронной сети, используя случайные веса, можно ожидать, что они должны быть также случайными. Обучающий алгоритм улучшает характеристики сети, постепенно изменения каждый вес в нужном направлении. Это называется *итеративной* процедурой и управляется в программе с помощью цикла FOR–NEXT в строках 270...400. Каждая итерация делает веса всё более эффективными для принятия правильного решения. Цикл итераций обычно выполняется до тех пор, пока не будет ощутимого улучшения. В типичных нейронных сетях количество итераций может быть от 10 до 10 000, но обычно несколько сот. Рассмотренный пример выполнялся за 800 итераций.

Для того чтобы эта итеративная стратегия успешно работала, должен быть простой параметр, который определяет качество работы системы на каждой итерации. Переменная ESUM (суммарная ошибка) выполняет в программе эту функцию. Первым действием внутри цикла является установка переменной ESUM в нуль (строка 290), чтобы она могла использоваться как аккумулятор. В каждой итерации полученное значение ошибки ESUM выводится на монитор (строка 380), что иллюстрирует процесс формирования окончательного результата. Значение ESUM начинается с большого числа и постепенно уменьшается в процессе обучения распознаванию цели. **Рис. 26.9** показывает примеры того, как ошибка ESUM уменьшается при итеративном процессе.



**Рис. 26.9.** Сходимость нейронной сети. Этот график показывает, как ошибка нейронной сети (значение ESUM) уменьшается, когда выполняются итерации. Показаны три разных попытки, каждая из которых начинается с разными начальными весами.

Все 260 изображений в обучающем множестве оцениваются в течение каждой операции, управляемые циклом FOR-NEXT в строках 310...360. Встроенная подпрограмма (строка 1000) используется для получения изображения из обучающей базы данных. Так как здесь нет ничего интересного, то мы опишем только параметры, передаваемые в эту подпрограмму и получаемые из неё. Входной параметр LETTER% данной подпрограммы имеет значение 1...260. При возврате из подпрограммы входные узлы X1[1]...X1[100] будут содержать значения пикселей изображения в базе данных, соответствующие LETTER%. Узел смещения X1[101] всегда имеет постоянное значение — *один*. Подпрограмма также возвращает другой параметр — CORRECT. Этот параметр содержит желаемое выходное значение сети для частной буквы. То есть если буква в изображении гласная, то CORRECT будет возвращаться со значением *один*. Если буква в изображении не гласная, CORRECT будет возвращаться со значением *нуль*.

После того как отрабатываемое изображение загружено в массив X1[1]...X1[100], другая подпрограмма (строка 2000) пропускает данные через текущую нейронную сеть для создания значения выходного узла X3. Другими словами, данная подпрограмма такая же, как **Программа 26.1**, за исключением разного числа узлов в каждом слое. Эта подпрограмма вычисляет также оценку качества идентификации сетью требуемой буквы. В строке 2210 переменная ELET (ошибка в определении буквы) вычисляется как разность между действительно сгенерированным выходным значением X3 и желаемым значением CORRECT. Это делает значение ELET лежащим между −1 и 1. Все 260 значений для ELET объединяются (строка 340), чтобы сформировать ESUM, общую квадратичную ошибку сети для введённого обучающего множества.

Строка 2220 отображает операцию, которая часто используется при вычислении ошибки: назначение приоритетов различным видам ошибок. Вспомним пример с болезнью раком, рассмотренный ранее в этой главе, и последствия выбора между ложной положительной и ложной отрицательной ошибками. В настоящем примере произвольно предположим, что ошибка в обнаружении цели в 5 раз хуже, чем ошибка противоположного характера. В действительности это говорит о

том, что сеть должна в первую очередь эффективно работать по критерию правильного обнаружения цели.

Следующая подпрограмма (строка 3000) является сердцем стратегии нейронной сети. Это реализация алгоритма изменения весов на каждой итерации. Будем использовать аналогию, чтобы объяснить лежащую в основе алгоритма математику. Рассмотрим ситуацию с военным парашютистом-десантником, сброшенным в тылу врага. Он приземлился на землю на неизвестной территории в кромешной темноте, так что невозможно что-либо видеть далее нескольких футов. Ему приказано спуститься в самый низ ближайшей долины, чтобы начать свою миссию. Перед ним возникает проблема: как, находясь в полной темноте, найти путь до самой низкой точки долины? Другими словами, ему нужен алгоритм изменения координат  $x$  и  $y$  на поверхности земли, чтобы минимизировать свою высоту. Это аналогично задаче изменения весов нейронной сети так, чтобы минимизировать ошибку сети ESUM.

Рассмотрим два алгоритма решения этой проблемы: *метод перебора (проб и ошибок)* и *метод наискорейшего спуска*. В методе «проб и ошибок» парашютист проходит определённое расстояние в некотором случайном направлении. Если новая высота больше, чем предыдущая, он возвращается на стартовую позицию и делает новую попытку. Если новая высота ниже, он повторяет процесс из новой точки. В конце концов он достигнет дна долины, хотя очень неэффективным и случайным путём. Этот метод называется методом «проб и ошибок», потому что такой же тип алгоритма используется природой в биологической эволюции. Каждое новое поколение вида имеет случайные отклонения от предыдущего. Если эти отличия выгодны для вида, они, по-видимому, должны остаться и передаться следующему поколению. В результате это улучшение позволяет животным получать больше пищи, лучше спасаться от врагов, давать больше потомков и т. п. Если новая черта вредна, животные с недостатками становятся пищей хищников и постепенно исчезают. В этом смысле каждое новое поколение является итерацией процедуры эволюционной оптимизации. Когда эволюция используется как обучающий алгоритм, каждый вес в нейронной сети слегка изменяется добавлением значения от генератора случайных чисел. Если изменённые веса делают сеть лучше (т. е. более низкое значение ошибки ESUM), изменения остаются, в противном случае они отбрасываются. Хотя этот метод работает, он слишком медленно сходится. Это термин, используемый для описания того, что постоянное улучшение делается в направлении оптимального решения (самой низкой точки долины). Подчас программе нужны дни для достижения решения, а не минуты или часы.

К счастью, алгоритм наискорейшего спуска работает гораздо быстрее. Так, парашютист, естественно, скажет: сначала надо оценить, какой путь является спуском, и затем двигаться в этом направлении. Подумайте об этом способе. Парашютист может сделать один шаг на север и записать изменение высоты. После возвращения в исходное положение он может сделать один шаг на восток и записать другое изменение высоты. Используя эти два значения, он может определить, какой путь является спуском. Предположим, что парашютист снизился на 10 см, когда он двигался на один шаг в северном направлении, и на 20 см, когда он шёл в восточном направлении. Чтобы идти прямо вниз, ему нужно двигаться вдоль каждой оси с шагом пропорционально наклону вдоль этой оси. В этом случае он может пройти 10 шагов на север и 20 шагов на восток. Это передвигает его

вниз по направлению наискорейшего спуска на расстояние  $\sqrt{10^2 + 20^2} = 22.4$  шага. То есть он мог бы двигаться по прямой линии в новое местоположение, используя 22.4 шага по диагонали. Ключевой момент — наискорейший спуск достигается при движении вдоль каждой оси на расстояние, пропорциональное наклону вдоль этой оси.

Рассматриваемая подпрограмма реализует такой же алгоритм наискорейшего спуска для весов сети. Прежде чем запустить подпрограмму 3000, выбранное изображение прикладывается к входному слою и информация распространяется к выходу. Это значит, что входы  $X[1]$ ,  $X[2]$  и  $X3$  определены так же, как и текущие значения весов  $WH[ , ]$  и  $WO[ , ]$ . Кроме того, мы знаем ошибку ELET, которую сеть даёт для конкретного изображения. Веса скрытого слоя изменяются в строках 3050...3120, а веса выходного слоя модифицируются в строках 3150...3190. Это делается вычислением наклона для каждого веса и затем изменением каждого веса на значение, пропорциональное этому наклону. В случае с парашютистом наклон вдоль оси находится продвижением по этой оси на малое расстояние (скажем,  $\Delta x$ ) измерением изменения высоты (скажем,  $\Delta E$ ) и затем делением ( $\Delta E / \Delta x$ ). Наклон веса нейронной сети может быть найден таким же способом: добавить малое приращение к значению веса ( $\Delta w$ ), найти результирующее изменение выходного сигнала ( $\Delta X3$ ) и разделить одно на другое ( $\Delta X3 / \Delta w$ ). Далее мы рассмотрим пример, который вычисляет наклон подобным образом. Однако в настоящем примере будем использовать более эффективный метод.

Ранее мы сказали, что нелинейность (сигмоид) должна быть дифференцируемой. Здесь мы будем использовать это свойство. Если мы знаем наклон в каждой точке нелинейности, мы можем непосредственно написать уравнение для наклона каждого веса ( $\Delta X3 / \Delta w$ ). Рассмотрим определённый вес, например  $WO[1]$ , соответствующий первому входу выходного узла. Посмотрите на структуру на Рис. 26.5 и 26.6 и спросите, как выход  $X3$  будет работать, если этот частный вес ( $w$ ) немного изменится, но все остальные останутся такими же? Ответ:

$$\Delta X3 / \Delta w = X2[1] \text{ НАКЛОН}_0. \quad (26.3)$$

Наклон весов выходного слоя. Это выражение записано для веса  $WO[1]$ .

В выражении (26.3)  $\text{НАКЛОН}_0$  — это первая производная сигмоида выходного слоя, оценивающая, на какой точке кривой мы находимся. Другими словами,  $\text{НАКЛОН}_0$  описывает, насколько изменяется *выход* сигмоида в ответ на изменение его *входа*. Из выражения (26.3)  $\text{НАКЛОН}_0$  может быть вычислен, исходя из текущего выходного значения сигмоида  $X3$ . Это вычисление показано в строке 3160. В строке 3170 вычисляется наклон этого веса по выражению (26.3) и запоминается в переменной  $DX3DW$  (т. е.  $\Delta X3 / \Delta w$ ).

Используя подобный анализ, можно найти наклон для веса в скрытом слое, например, для  $WH[1, 1]$ :

$$\Delta X3 / \Delta w = X1[1] \text{ НАКЛОН}_{H1} WO[1] \text{ НАКЛОН}_0. \quad (26.4)$$

Наклон весов скрытого слоя. Это выражение записано для веса  $WH[1, 1]$ .

$\text{НАКЛОН}_{H1}$  — это первая производная сигмоида скрытого слоя в рабочей точке на его кривой. Другие значения  $X1[1]$  и  $WO[1]$  — это простые константы. В строках 3070 и 3080 наклоны сигмоидов вычисляются с использованием выраже-

ния (26.2). Наклон веса скрытого слоя DX3DW вычисляется в строке 3090 по выражению (26.4).

Теперь, когда мы знаем наклон каждого веса, мы можем посмотреть как каждый вес изменяется к следующей итерации. Новое значение для каждого веса находится из текущего веса добавлением значения, пропорционального наклону:

$$w_{\text{нов}} = w_{\text{стар}} + (\Delta X_3 / \Delta w) ELET MU. \quad (26.5)$$

Изменение весов. Каждый вес изменяется добавлением значения, пропорционального наклону веса.

Это вычисление выполняется в строке 3100 для скрытого слоя и в строке 3180 для выходного слоя. Константа пропорциональности состоит из двух коэффициентов: ELET — ошибка сети для конкретного входа и MU — множество констант в начале программы. Чтобы понять необходимость ELET в этом вычислении, предположим, что изображение, находящееся на входе, даёт малую ошибку в выходном сигнале. Далее предположим, что другое изображение, поступающее на вход, даёт большую выходную ошибку. Когда изменяются веса, мы хотим подтолкнуть сеть в большей степени ко второму изображению, чем к первому. Если что-то работает плохо, мы хотим изменить его, если работает хорошо, мы хотим оставить его. Это выполняется изменением каждого веса пропорционально текущей ошибке ELET.

Чтобы понять, как на систему воздействует MU, вернёмся к примеру с парашютистом. Однажды определив направление спуска, он должен решить, как далеко пройти до переоценки наклона местности. Делая это расстояние коротким, например, один метр, он будет способен точно следовать профилю местности и всегда будет двигаться в оптимальном направлении. Проблема состоит в том, что он потеряет время в основном на оценку наклона, а не на движение вниз холма. Для сравнения: он может выбрать расстояние большим, скажем, 1000 метров. Хотя это позволит парашютисту быстро передвигаться по местности, он может проскочить путь вниз. Слишком большое расстояние делает его передвижения малоэффективными.

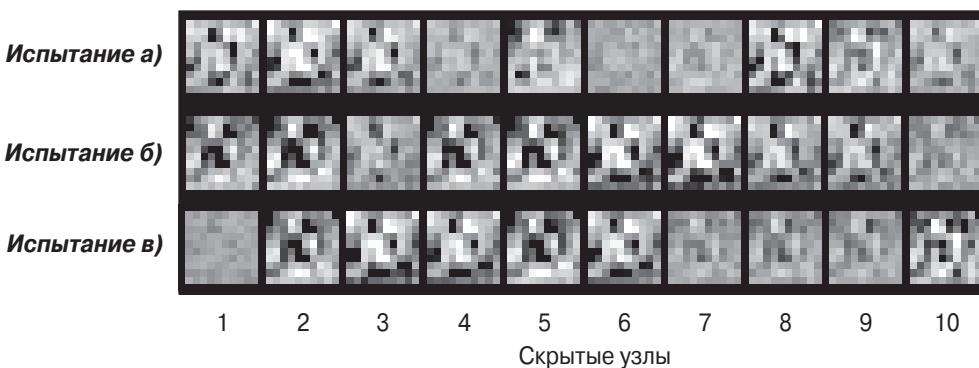
В нейронной сети константа MU управляет величиной изменения веса на каждой итерации. Значение её зависит от конкретной проблемы и может быть таким малым, как  $10^{-6}$ , или таким большим, как 0.1. По аналогии с парашютистом можно ожидать, что слишком малое значение вызовет медленную сходимость нейронной сети. Для сравнения: слишком большое значение будет вызывать неопределённую сходимость и демонстрировать хаотические колебания вокруг окончательного решения. К сожалению, способ реагирования нейронных сетей на различные значения MU может быть трудным для понимания или предсказания. Из-за этого многие подвергают критике необходимость мониторинга за поведением ошибки сети (т. е. ESUM) во время обучения, например, её распечаткой (выводом на экран) в конце каждой итерации. Если система сходится плохо, остановите программу и испытайте другое значение константы MU.

## 26.5. Оценка результатов

Так как же она работает? Программа обучения для распознавания гласных исполнялась 3 раза, используя различные случайные значения для начальных весов. Около 10 минут потребовалось для завершения 800 итераций на персональном

компьютере Pentium с частотой 1 ГГц. **Рис. 26.9** показывает, как ошибка сети изменяется после каждого периода. Неуклонное снижение ошибки показывает, что сеть решает поставленную задачу обучения и достигает близкого к оптимальному значения после нескольких сот итераций. Каждое испытание даёт разное решение проблемы с разными конечными характеристиками. Это аналогично старту парашютиста с различных мест и, следовательно, приземлению на дне разных впадин. Так же как одни впадины глубже, чем другие, некоторые решения нейронных сетей лучше, а другие хуже. Это значит, что обучающий алгоритм необходимо использовать несколько раз, а в качестве окончательного решения выбирается самый лучший результат из группы.

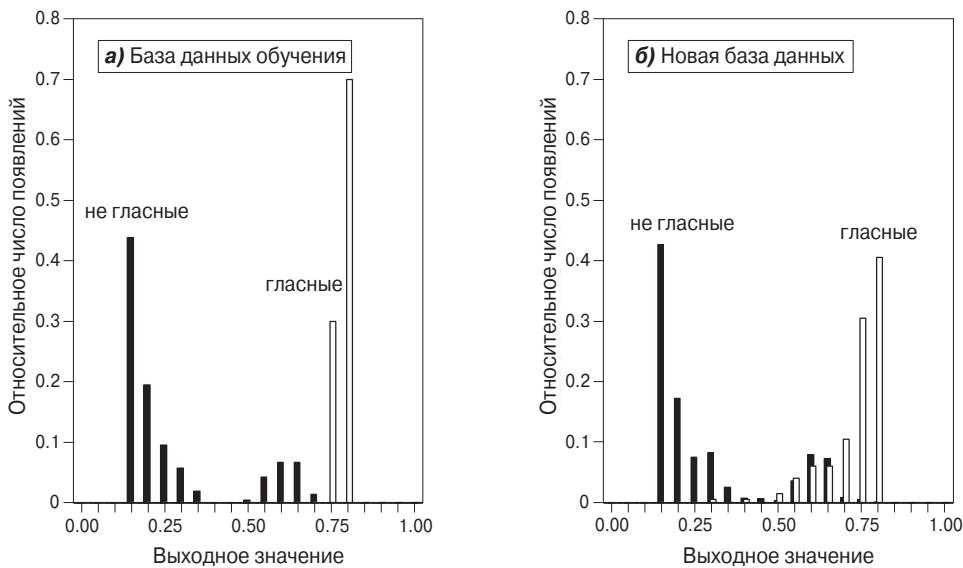
На **Рис. 26.10** показаны изображения весов скрытого слоя трёх решений. Это означает, что первым действием, предпринятым нейронной сетью, является корреляция (умножение и суммирование) этих изображений с входным сигналом. Они выглядят как случайный шум! Можно показать, что эти значения весов решают поставленную задачу, но почему это происходит — остаётся загадкой. Здесь есть над чем подумать. Мозг человека состоит примерно из 100 триллионов нейронов, каждый в среднем с 10 000 связей. Если мы не можем понять простую нейронную сеть в данном примере, как мы можем изучать то, что по крайней мере в 100 000 000 000 000 раз более сложно? Это исследование XXI в.



**Рис. 26.10.** Пример весов нейронной сети. На этом рисунке изображены веса скрытого слоя в виде изображений. Все три решения воспринимаются глазом человека как случайные сигналы.

**Рис. 26.11а** показывает гистограмму выхода нейронной сети для 260 букв в обучающей последовательности. Помните, веса были выбраны для того, чтобы сделать выход для изображения гласных букв близким к единице, а для не гласных — близким к нулю. Разделение оказывается полным: отсутствует перекрытие между двумя распределениями. Заметим также, что распределение гласных более узкое, чем распределение согласных, потому что ошибка принятия решения «есть цель» в 5 раз более важна, чем ошибка принятия решения «нет цели» (см. строку 2220).

Для сравнения: **(б)** показывает гистограмму изображений 261...1300 в этой базе данных. Хотя распределения рассматриваемых событий вполне понятны, они не полностью разделены. Почему нейронная сеть лучше работает с первыми 260 буквами, чем с последними 1040? Просто пройти тестирование гораздо легче, если ты уже знаешь ответы. Сеть осуществляет распознавание известных образов из



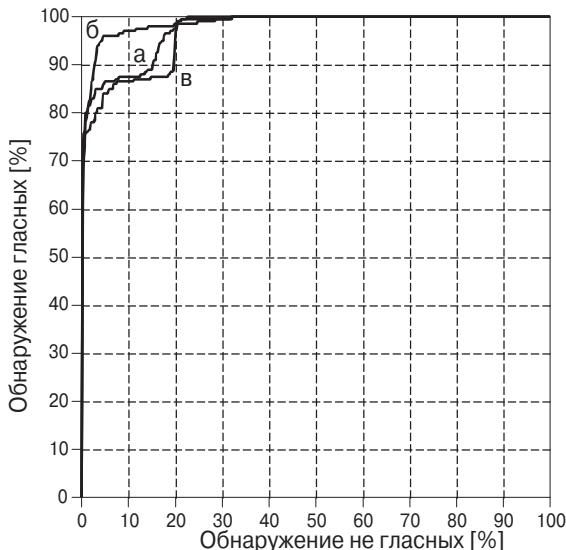
**Рис. 26.11.** Характеристика нейронной сети. Гистограммы выходных значений нейронной сети: **а)** для обучающих данных; **б)** для остальных изображений. Нейронная сеть работает лучше с обучающими данными, потому что уже знает ответы на этот тест.

обучающей последовательности, а не выделяет некоторые общие признаки, отличающие гласные буквы от согласных.

**Рис. 26.12** показывает характеристику трёх решений, изображённых в виде кривых РХП. Испытание на **(б)** обеспечивает значительно лучшую сеть, чем два других. Такой результат получен случайным образом и зависит от используемых начальных весов. При одной установке порога нейронная сеть, созданная в испытании «б», может обнаружить 24 из 25 целей, т. е. 96% изображений гласных, с числом ложных тревог 1 из 25 решений (т. е. 4% изображений не гласных). Это совсем неплохо, если учесть абсолютно абстрактный подход к описанию задания и обобщённость решения.

Некоторые заключительные комментарии о нейронных сетях. Достижение сходимости алгоритмов обучения нейронной сети может быть непростым. Если ошибка сети убывает не монотонно, программа должна быть остановлена, изменена и запущена вновь. Возможно, придётся делать несколько попыток, прежде чем вы достигнете успеха. Чтобы добиться сходимости, можно изменить три параметра: 1) константу MU; 2) амплитуду начальных случайных весов и 3) количество скрытых узлов (изменять их следует именно в этом порядке).

Наиболее критичным элементом при разработке нейронной сети является достоверность обучающих примеров. Например, когда создаются новые коммерческие продукты, единственные данные, имеющиеся в распоряжении разработчиков, — это тестовые данные прототипов, результаты моделирования, приблизительные оценки и т. д. Если нейронная сеть обучена на основе только этой предварительной информации, она не может точно работать в готовых продуктах. Любое расхождение между примерами обучающей базы данных и реальными данными будет ухудшать характеристики сети (закон Мерфи для нейронных сетей). Не пытайтесь помочь своей нейронной сети в этом случае. Это невозможно!



**Рис. 26.12.** РХП-анализ примеров нейронных сетей. Эти кривые сравнивают три нейронные сети, созданные для обнаружения гласных букв. Кривая испытания «б» — это наилучшее решение. Эта кривая расположена ближе других к верхнему левому углу графика. Данная сеть может корректно обнаруживать 24 из 25 целей, гарантируя всего 1 ложную тревогу на каждые 25 не целей. Координаты этой точки на кривой РХП:  $x = 4\%$  и  $y = 96\%$ .

## 26.6. Проектирование рекурсивных фильтров

Главы 19 и 20 показывают, как проектировать рекурсивные фильтры со стандартными частотными характеристиками: высокочастотный, низкочастотный, полосовой и т. д. Что, если вам нужно что-нибудь специфическое? Ответ: проектируйте рекурсивный фильтр так же, как нейронную сеть, т. е. начните со множества рекурсивных коэффициентов и используйте итерационную обработку, чтобы постепенно «слепить» из них то, что вы хотите. Этот метод занимает важное место по двум причинам. Во-первых, он позволяет разрабатывать заказные рекурсивные фильтры без использования сложностей математики  $Z$ -преобразования. Во-вторых, он показывает, что идеи традиционной ЦОС и нейронных сетей можно комбинировать для создания замечательных алгоритмов.

### Программа 26.4

```

100 'ИТЕРАТИВНОЕ ПРОЕКТИРОВАНИЕ РЕКУРСИВНЫХ ФИЛЬТРОВ
110 '
120 'ИНИЦИАЛИЗАЦИЯ
130 N% = 256           'число точек в БПФ
140 NP% = 8            'число полюсов в фильтре
150 DELTA = .00001     'изменение приращения
160 MU = .2             'размер шага итерации
170 DIM REX[255]        'действительная часть сигнала при БПФ

```

```

180 DIM IMX[255]           'мнимая часть сигнала при БПФ
190 DIM T[128]              'желаемая частотная характеристика (только амплитуда)
200 DIM A[8]                'коэффициент рекурсии "а"
210 DIM B[8]                'коэффициент рекурсии "б"
220 DIM SA[8]              'наклон для коэффициентов "а"
230 DIM SB[8]              'наклон для коэффициентов "б"
240 '
250 GOSUB XXXX             'предполагаемая подпрограмма загрузки Т[ ]
260 '
270 FOR P% = 0 TO NP%      'инициализация коэффициентов системы
280 A[P%] = 0
290 B[P%] = 0
300 NEXT P%
310 A[0] = 1
320 '
330 '                       'ЦИКЛ ИТЕРАЦИЙ
340 FOR ITER% = 1 TO 100    'цикл для желаемого числа итераций
350 GOSUB 2000              'вычисление новых коэффициентов
360 PRINT ITER% ENEW MU    'печатать текущее состояние
370 IF ENEW > EOLD THEN MU = MU/2 'изменить значение MU
380 NEXT ITER%
390 '
400 '
410 FOR P% = 0 TO NP%      'ПЕЧАТАТЬ КОЭФФИЦЕНТЫ
420 PRINT A[P%] B[P%]
430 NEXT P%
440 '
450 END

```

**Программа 26.4** — это главная программа метода, работающая совместно с двумя подпрограммами, приведёнными в **Программе 26.5**. Массив  $T[]$  хранит желаемую частотную характеристику в форме некоторого вида кривой, которую мы должны воспроизвести. Так как эта программа основана на БПФ, то длины сигналов должны быть степенью двух. Как видно, **Программа 26.4** использует БПФ длины 256, что определено переменной  $N\%$  в строке 130. Это значит, что переменные  $T[0]...T[128]$  соответствуют частотам 0...0.5 частоты дискретизации. В этом массиве содержится только амплитуда; фаза при этом не учитывается.

Коэффициентами рекурсии является множество их начальных значений в строках 270...310. Не используйте для этой цели случайные числа, иначе фильтры могут оказаться неустойчивыми. Коэффициенты рекурсии хранятся в массивах  $A[]$  и  $B[]$ . Переменная  $NP\%$  устанавливает число полюсов в проектируемом фильтре. Например, если  $NP\% = 5$ , то коэффициент « $a$ » пробегает массив от  $A[0]$  до  $A[5]$ , а коэффициент « $b$ » — от  $B[0]$  до  $B[5]$ .

Как было показано ранее, итеративная процедура требует единственного значения показателя качества, которое определяет, как хорошо функционирует система в текущем состоянии. Это обеспечивается переменной  $ER$  (текущая ошибка), которая вычисляется в подпрограмме (строка 3000). Строки 3040...3080 загружают единичный импульс в массив  $IMX[]$ . Далее строки 3100...3150 используют этот импульс как входной сигнал для проектирования рекурсивного фильтра с помощью текущих значений  $A[]$  и  $B[]$ . Выход фильтра — это импульсная

характеристика текущей системы, она хранится в массиве REX [ ]. Частотная характеристика системы находится вычислением БПФ от импульсной характеристики, как показано в строке 3170. Подпрограмма в строке 1000 — это программа БПФ, приведённая в Главе 12 (**Программа 12.3**). Подпрограмма БПФ вычисляет частотную характеристику в прямоугольной системе координат, переписывая массивы REX [ ] и IMX [ ].

Строки 3200...3250 вычисляют ошибку ER — среднеквадратическую ошибку между амплитудами текущей воспроизведимой и желаемой частотными характеристиками. Обратите особое внимание на то, как эта ошибка находится. Итеративные действия этой программы минимизируют данную ошибку, поэтому способ, которым она определяется, является очень важным. Цикл FOR — NEXT исполняется по всем частотам частотной характеристики. Для каждой частоты строка 3220 вычисляет амплитуду текущей воспроизведимой частотной характеристики по данным в прямоугольной системе координат. В строке 3230 ошибка на этой частоте находится вычитанием желаемой амплитуды T [ ] от текущей амплитуды MAG. Эта ошибка затем возводится в квадрат и добавляется к аккумулируемой переменной ER. После завершения цикла переменная ER определяет среднеквадратическую ошибку всей частотной характеристики.

Строки 340...380 управляют итерационным циклом программы. Подпрограмма в строке 2000 выполняет изменение рекурсивных коэффициентов. Первым действием этой подпрограммы является определение текущего значения ошибки ER и сохранение его в другой переменной EOLD (строки 2040...2050). После того как подпрограмма изменит коэффициенты, значение переменной ER будет снова определено и соответственно назначено переменной ENEW (строки 2270 и 2280).

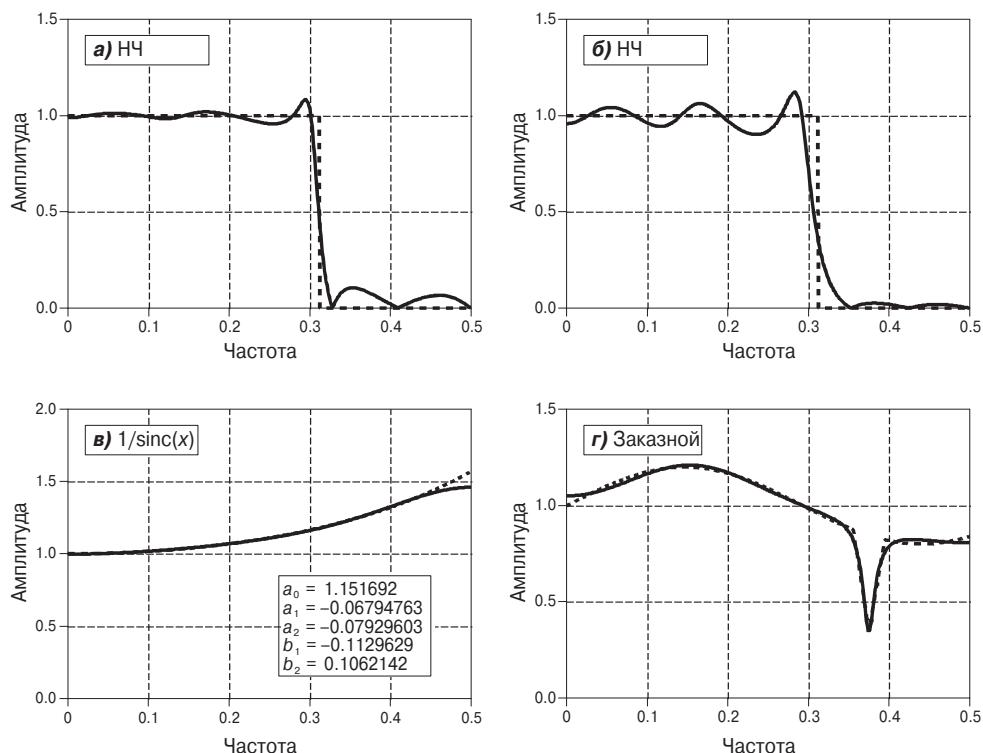
Константа MU управляет размером шага итерации, как и в предыдущей программе нейронной сети. Полезное свойство, используемое в этой программе, — автоматическое регулирование значения константы MU. По этой причине используются две переменные — EOLD и ENEW. Когда программа стартует, MU устанавливается на относительно высокое значение 0.2 (строка 160). Это позволяет обеспечить быструю сходимость, но будет ограничивать то, насколько близко фильтр подойдёт к оптимальному решению. Когда в процессе выполнения итераций будут достигнуты те точки, в которых нет дальнейшего прогресса в минимизации ошибки, идентифицируемые тем, что ENEW будет выше, чем EOLD, обучение приостанавливается. Каждый раз, когда это происходит, строка 370 уменьшает значение константы MU.

Подпрограмма в строке 2000 изменяет коэффициенты рекурсии в соответствии с методом наискорейшего спуска: вычисляет наклон каждого коэффициента, а затем изменяет его пропорционально наклону. Строки 2080...2130 вычисляют наклоны коэффициентов «а», сохраняя их в массиве SA [ ]. Подобно этому строки 2150...2200 вычисляют наклоны коэффициентов «б», сохраняя их в массиве SB [ ]. Строки 2220...2250 модифицируют каждый рекурсивный коэффициент на значение, пропорциональное их наклону. В этой программе константа пропорциональности — это просто размер шага MU.

В заключение рассмотрим то, как программа вычисляет наклоны коэффициентов рекурсии. В примере с нейронной сетью с этой целью выводится уравнение для наклона. Однако эта процедура не может использоваться в нашем случае, потому что она требует взятия производной через ДПФ. Вместо этого применен ме-

тод решения проблемы «в лоб»: изменение коэффициентов рекурсии на небольшое приращение и затем непосредственно вычисление нового значения ошибки ER. Наклон находится как изменение ошибки ER, делённое на значение приращения. Текущее значение ошибки ER находят в строках 2040...2050 и запоминают в переменной EOLD. Цикл в строках 2080...2130 выполняется по всем коэффициентам «а». Первое действие внутри этого цикла — это добавление малых приращений DELTA к коэффициенту рекурсии, который рассчитан в строке 2090. Подпрограмма в строке 3000 вызывается в строке 2100, чтобы найти значение ошибки ER с модифицированными коэффициентами. Затем строка 2110 вычисляет наклон этого коэффициента:  $(ER - EOLD)/\text{DELTA}$ , а строка 2120 возвращает модифицированные коэффициенты, вычитая значения DELTA.

**Рис. 26.13** показывает некоторые примеры фильтров, спроектированных с использованием этой программы. Пунктирная линия — это желаемая частотная характеристика, а сплошная линия — воспроизведенная частотная характеристика проектируемого фильтра. Каждый из этих фильтров требует примерно минуту для вычислений на сходимость на ПК с процессором Pentium 1 ГГц. На (а) изображен восьмиполосный низкочастотный фильтр, где ошибка «равновзвешена» по всему



**Рис. 26.13** Итеративное проектирование рекурсивных фильтров. На (а) изображен восьмиполосный низкочастотный фильтр с ошибкой, равномерно распределенной между 0 и 0.5. На (б) ошибка взвешена, чтобы улучшить характеристику в полосе заграждения за счет ошибки в полосе пропускания. На (с) показан двухполюсный фильтр, используемый для коррекции  $1/\text{sinc}(x)$  в цифро-аналоговом преобразовании. Частотная характеристика на (д) полностью заказная. На каждом рисунке желаемая частотная характеристика показана пунктирной линией, а реальная воспроизведенная частотная характеристика — сплошной линией.

частотному спектру. На (б) показан точно такой же фильтр, за исключением того, что ошибка в полосе непрозрачности умножается на восемь, после того как вычислена ошибка ER. Это улучшает фильтр, т. к. уменьшаются колебания в зоне непрозрачности за счёт больших колебаний в полосе пропускания.

### **Программа 26.5**

```

2000 'ПОДПРОГРАММА ВЫЧИСЛЕНИЯ НОВЫХ КОЭФФИЦИЕНТОВ РЕКУРСИИ
2010 'Переменные входа программы: A[ ], B[ ], DELTA, MU
2020 'Переменные выхода программы: A[ ], B[ ], EOLD, ENEW
2030 '
2040 GOSUB 3000           'НАЙТИ ТЕКУЩУЮ ОШИБКУ
2050 EOLD = ER            'хранит текущую ошибку в переменной EOLD
2060 '
2070 'НАЙТИ НАКЛОНЫ ОШИБОК
2080 FOR P% = 0 TO NP%    'цикл по каждому коэффициенту "а"
2090 A[P%] = A[P%] + DELTA 'прибавить небольшое приращение к коэффициенту
2100 GOSUB 3000           'найти ошибку с изменением
2110 SA[P%] = (ER-EOLD)/DELTA 'вычислить наклон ошибки и сохранить в SA[ ]
2120 A[P%] = A[P%] - DELTA 'вернуть коэффициент к первоначальному значению
2130 NEXT P%
2140 '
2150 FOR P% = 1 TO NP%    'повторить процесс для каждого коэффициента "б"
2160 B[P%] = B[P%] + DELTA
2170 GOSUB 3000
2180 SB[P%] = (ER-EOLD)/DELTA 'вычислить наклон ошибки, сохранить в SB[ ]
2190 B[P%] = B[P%] - DELTA
2200 NEXT P%
2210 '                   'ВЫЧИСЛИТЬ НОВЫЕ КОЭФФИЦИЕНТЫ
2220 FOR P% = 0 TO NP%    'цикл по каждому коэффициенту
2230 A[P%] = A[P%] - SA[P%] * MU 'изменить коэффициенты в направлении «спуска»
2240 B[P%] = B[P%] - SB[P%] * MU
2250 NEXT P%
2260 '
2270 GOSUB 3000           'НАЙТИ НОВУЮ ОШИБКУ
2280 ENEW = ER            'запомнить новую ошибку в переменной ENEW
2290 '
2300 RETURN

3000 'ПОДПРОГРАММА ВЫЧИСЛЕНИЯ ОШИБКИ В ЧАСТОТНОЙ ОБЛАСТИ
3010 'Переменные входа программы: A[ ], B[ ], T[ ]
3020 'Переменная выхода программы: ER
3030 '
3040 FOR I% = 0 TO N%-1   'ЗАГРУЗИТЬ СДВИНУТЫЕ ИМПУЛЬСЫ В IMX[ ]
3050 REX[I%] = 0
3060 IMX[I%] = 0
3070 NEXT I%
3080 IMX[12] = 1
3090 '                   'ВЫЧИСЛИТЬ ИМПУЛЬСНУЮ ХАРАКТЕРИСТИКУ
3100 FOR I% = 12 TO N%-1
3110 FOR J% = 0 TO NP%
3120 REX[I%] = REX[I%] + A[J%] * IMX[I%-J%] + B[J%] * REX[I%-J%]

```

```

3130 NEXT J%
3140 NEXT I%
3150 IMX[12] = 0
3160 '
3170 GOSUB 1000      ' ВЫЧИСЛИТЬ БПФ
3180 '
3190 '
3200 ER = 0           ' НАЙТИ ОШИБКУ В ЧАСТОТНОЙ ОВЛАСТИ
3210 FOR I% = 0 TO N%/2 ' обнулить ER и использовать как аккумулятор
3220 MAG = SQR(REX[I%]^2 + IMX[I%]^2) 'цикл по каждой положительной частоте
3230 ER = ER + ( MAG - T[I%] )^2      'преобразование прямоугольного в полярное
3240 ER = ER + ( MAG - T[I%] )^2      'вычисление и накопление квадрата ошибки
3240 NEXT I%
3250 ER = SQR( ER/(N%/2+1) )          'конец вычисления ошибки ER
3260 '
3270 RETURN

```

На (в) изображён двухполюсный фильтр для  $1/\text{sinc}(x)$ . Как обсуждалось в Главе 3, он может быть использован для нейтрализации звена нулевого порядка при цифро-аналоговом преобразовании (см. Рис. 3.6). Ошибка в этом фильтре была сосредоточена между 0 и 0.45, давая наилучшее совпадение в этом диапазоне за счёт плохого согласования между 0.45 и 0.5. Наконец, (г) — это крайне неравномерная шестиполюсная частотная характеристика, которая включает очень резкий провал. Чтобы достигнуть сходимости, коэффициенты рекурсии были изначально установлены на такие же значения, как в режекторном фильтре.

## СЖАТИЕ ДАННЫХ

Передача и хранение данных стоят определённых денег. Чем больше поток информации, тем больше денежные затраты. Однако в большинстве случаев цифровые данные не хранятся в максимально компактной форме. Скорее информация хранится в любом виде, который оказывается самым удобным для использования, например: ASCII-код текстовых редакторов; бинарный код, исполняемый на компьютерах; отдельные выборки в системах сбора данных и т. д. Обычно эти часто используемые методы кодирования требуют почти в 2 раза большего объёма файлов данных, чем действительно необходимо для представления информации. Сжатие данных — это общий термин, употребляемый для разных алгоритмов и программ, разработанных с целью решения указанной проблемы. Программа сжатия используется для того, чтобы преобразовать данные из формата, удобного в использовании, в формат, оптимизированный с точки зрения компактности. И наоборот, программа декомпрессии возвращает информацию к её первоначальной форме. Мы рассмотрим в этой главе пять методов сжатия данных. Первые три — это простые методы кодирования, называемые кодированием длин серий, кодированием Хаффмана и дельта-кодированием соответственно. Последние два метода представляют собой специально разработанные процедуры, которые стали общепринятыми стандартами, — LZW и JPEG.

### 27.1. Стратегии сжатия данных

Табл. 27.1 показывает два разных подхода к классификации алгоритмов сжатия данных. В левой части таблицы приведено деление на методы *без потерь* и методы *с потерями*. Определение «без потерь» означает, что восстановленный после сжатия файл оказывается *идентичным* оригиналу. Это необходимо для многих типов данных, например исполняемого кода программ, файлов текстовых редакторов, файлов табличных данных и т. п. Вы не можете позволить себе поставить не на своё место даже один бит этого типа информации. Для сравнения: файлы данных, которые представляют изображения и другие принимаемые сигналы, не обязательно должны быть строго неизменны при хранении и передаче. Все реально существующие измеряемые сигналы неизбежно содержат шумы определённого уровня. Если изменения, произошедшие с этими сигналами, подобны наложению небольшого аддитивного шума, то никакого ущерба это не нанесёт. Методы сжатия, которые допускают такой тип искажений, называются методами *с потерями*. Это различие важно, потому что методы с потерями значительно более эффективны при сжатии, чем методы без потерь. Чем выше степень сжатия, тем больший шум добавляется к данным.

Таблица 27.1. Классификация методов сжатия

Способы классификации				
Без потерь или с потерями		Метод	Фиксированный или переменный размер группы	
Методы без потерь	Методы с потерями		Размер группы	На входе
Кодирование длин серий	CS&Q	CS&Q	фиксированный	фиксированный
Кодирование Хаффмана	JPEG	Кодирование Хаффмана	фиксированный	переменный
Дельта-кодирование	MPEG	Арифметический	переменный	переменный
LZW		Кодирование длин серий, LZW	переменный	фиксированный

Изображения, передаваемые через Интернет, являются блестящим примером того, почему важно сжатие данных. Предположим, что нам нужно загрузить цифровую цветную фотографию через компьютерный modem со скоростью 33.6 Кбит/с. Если изображение не сжато (например, TIFF-файл), оно будет представлено примерно 600 Кбайтами данных. Если файл сжат с использованием метода без потерь (например, в формате GIF), он будет почти вдвое меньше, т. е. 300 КБ. Если же использовать сжатие с потерями (JPEG-файл), то объём составит около 50 КБ. Время загрузки этих трёх эквивалентных файлов будет 142, 71 и 12 с соответственно. Это очень большая разница! JPEG — наилучший выбор для цифровых фотографий, в то время как GIF используется с рисованными изображениями, такими как, например, логотип компании, который имеет большую площадь одного цвета.

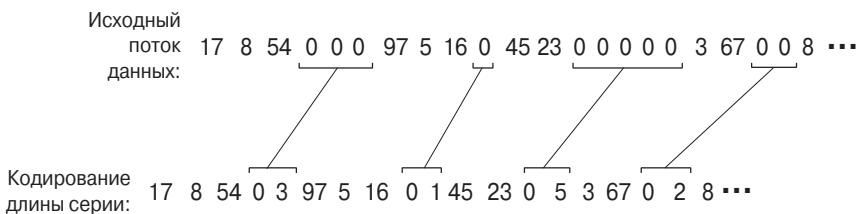
Второй способ классификации методов сжатия данных показан в правой части **Табл. 27.1**. Большинство программ сжатия данных работают с группами данных: получают очередную группу данных из исходного файла, сжимают её некоторым способом и затем записывают сжатую группу в выходной файл. Например, один из способов, представленных в этой таблице, — это CS&Q-сжатие (сокращение от слов грубая выборка и/или квантование — coarser sampling and/or quantization). Предположим, что мы сжимаем аудиосигнал, который был оцифрован 12-битным кодом. Мы можем прочитать две соседние выборки из исходного файла (24 бита), отбросить одну выборку полностью, отбросить последние значения 4 бита из другой выборки и затем записать оставшиеся 8 бит в выходной файл. При 24 входных и 8 выходных битах мы имеем коэффициент сжатия 3:1, используя алгоритм с потерями. Хотя это грубо само по себе, но оказывается очень эффективно при использовании совместно с методом, называемым *сжатием с преобразованием* (*transform compression*). Как будет показано далее, это основа JPEG-формата.

**Табл. 27.1** показывает, что CS&Q относится к методам с фиксированным размером входа и фиксированным размером выхода. То есть фиксированное число бит читается из входного файла, и меньшее фиксированное число бит записывается в выходной файл. Другие методы сжатия позволяют читать или писать переменное число бит. При изучении того или иного из этих методов сжатия будет полезно вернуться к этой таблице и посмотреть, к чему относится этот метод в представленной классификации. Почему JPEG и MPEG не приведены в этой таблице? Это сложные алгоритмы, которые объединяют много других методов. Они слишком изощрённые, чтобы быть классифицированными по этим простым категориям.

## 27.2. Кодирование длин серий

Файлы данных часто содержат одинаковые символы, повторяющиеся много раз в одной строке. Например, в текстовых файлах применяется большое количество пробелов, используемых для разделения фраз, выделения отступов в начале абзацев, формирования таблиц и графиков и т. п. Оцифрованные сигналы также могут включать фрагменты, на которых значение сигнала не меняется. Например, изображение ночного неба может содержать большие фрагменты, представляющие чёрный фон. Другой пример — цифровая музыка, для которой характерно наличие длинной последовательности нулей-пауз между песнями. *Кодирование длин серий* — это самый простой метод сжатия файлов такого типа.

**Рис. 27.1** иллюстрирует кодирование длин серий для последовательностей данных, имеющих часто встречающиеся сегменты нулевых значений. Каждый раз, когда нуль встречается во входных данных, он заменяется двумя значениями в выходном файле. Первое из этих значений — нуль, являющийся флагом, указывающим, что начинается сжатие длин серий. Второе значение — длина последовательности нулей. Если средняя длина нулевых фрагментов больше двух, то сжатие будет успешным. С другой стороны, если в файле оказывается большое число нулей, стоящих отдельно, то «сжатый» таким образом файл может стать объёмнее исходного.



**Рис. 27.1.** Пример кодирования длины серий. Каждая серия нулей заменяется двумя символами в сжатом файле: нуль указывает, что сжатие начинается; последующее число соответствует количеству нулей в серии.

Разработано много разных схем кодирования длин серий. Например, входные данные могут рассматриваться как отдельные байты или группы байтов; кодирование длин серий может использоваться по отношению только к одному символу (как рассмотренный пример с нулём), к нескольким символам или ко всем символам.

Хорошим примером обобщённой схемы кодирования длин серий является формат PackBits, созданный для пользователей Macintosh. Каждый байт (8 бит) входного файла заменяется девятью битами в сжатом файле. Добавленный девятый бит интерпретируется как *знак* числа. То есть каждый символ, читаемый из входного файла, — это число от 0 до 255, а каждый символ, записанный в кодированный файл, — число от -255 до 255. Чтобы понять, как это может быть использовано, рассмотрим входной файл: 1, 2, 3, 4, 2, 2, 2, 2, 4 и сжатый файл алгоритмом «PackBits»: 1, 2, 3, 4, 2, -3, 4. Программа сжатия просто преобразовала каждое число входного файла в сжатый файл, за исключением чисел 2, 2, 2, 2. Эта последовательность представляется в выходном файле двумя числами: 2, -3. Первое число (<2>) показывает, какой символ входит в серию. Второй — число «-3» — позволяет найти количество символов в серии взятием его абсолютного значения и добавлением единицы. Например, последовательность 4, -2 означает: 4, 4, 4, а последовательность 21, -4 означает: 21, 21, 21, 21, 21 и т. д.

Недостаток формата PackBits состоит в том, что девять битов должны быть преформатированы в стандартные восьмибитные байты, используемые в компьютерной памяти и при передаче информации. Можно модифицировать эту схему, если считать, что на входе будет текст в кодировке ASCII<sup>1)</sup>. Как показывает Табл. 27.2, каждый ASCII-символ обычно хранится как полный байт, но в действительности он использует всего семь битов, идентифицирующих символ. Другими словами, значения от 127 до 255 не заняты какими-либо стандартными символами и не обязаны передаваться или храниться. Это позволяет использовать восьмой бит как индикатор процесса кодирования длин серий.

Таблица 27.2. Коды ASCII

0	null – байт, содержащий код 00h	32	space – пробел	64	@	96	`
1	start heading – начало заголовка	33	!	65	A	97	a
2	start of text – начало текста	34	"	66	B	98	b
3	end of text – конец текста	35	#	67	C	99	c
4	end of xmit – конец передачи (процесса)	36	\$	68	D	100	d
5	enquiry – запрос	37	%	69	E	101	e
6	acknowledge – подтверждение приёма	38	&	70	F	102	f
7	bell, beep – «звонок»	39	'	71	G	103	g
8	backspace – забой	40	(	72	H	104	h
9	horz. tab – горизонтальная табуляция	41	)	73	I	105	i
10	line feed – перевод строки	42	*	74	J	106	j
11	vertical tab, home – вертикальная табуляция	43	+	75	K	107	k
12	form feed, cls – перевод страницы, новая страница	44	,	76	L	108	l
13	carriage return – возврат каретки	45	-	77	M	109	m
14	shift out – сдвигать без сохранения выдыхаемых битов	46	.	78	N	110	n
15	shift in – сдвигать с сохранением выдыхаемых битов	47	/	79	O	111	o
16	data line escape – смена (активного) канала данных	48	0	80	P	112	p
17	device control 1 – управление устройством 1	49	1	81	Q	113	q
18	device control 2 – управление устройством 2	50	2	82	R	114	r
19	device control 3 – управление устройством 3	51	3	83	S	115	s
20	device control 4 – управление устройством 4	52	4	84	T	116	t
21	negative acknowledge – отсутствие подтверждения приёма	53	5	85	U	117	u
22	synchronous idle – синхронное прерывание	54	6	86	V	118	v
23	end xmit block – конец передачи блока	55	7	87	W	119	w
24	cancel – отмена	56	8	88	X	120	x
25	end of medium – конец носителя	57	9	89	Y	121	y
26	substitute – замена	58	:	90	Z	122	z
27	escape – переход, выход, отключаться	59	;	91	[	123	{
28	file separator – разделитель файлов	60	<	92	\	124	
29	group separator – разделитель групп (данных)	61	=	93	]	125	}
30	record separator – разделитель записей, специальный символ, код и т. п., позволяющий определять конец текущей записи	62	>	94	^	126	~
31	unit separator – разделитель элементов (данных)	63	?	95	_	127	del – удалить

<sup>1)</sup> ASCII — это давно установленный стандарт кодировки букв и чисел для представления в цифровой форме. Каждый печатаемый символ обозначается числом от 32 до 127, а числа от 0 до 31 используются для различных управляющих действий. Хотя зарезервировано только 128 кодов, символы ASCII обычно хранятся как полный байт (8 бит). Неопределённые значения (от 128 до 255) часто используются для греческих букв, математических символов и различных геометрических образов; однако они не стандартизованы. Многие из управляющих символов (от 0 до 31) основаны на более старых связных сетях и не применимы для компьютерной технологии.

## 27.3. Кодирование Хаффмана

Этот метод назван в честь Д. А. Хаффмана, который разработал данную процедуру в 1950-е годы. Рис. 27.2 показывает гистограмму значений байтов из большого ASCII-файла. Более 96% этого файла состоит только из 31 символа: букв нижнего регистра, пробела, точки с запятой, запятой, точки и возврата каретки. Этот факт можно использовать, чтобы создать соответствующую схему сжатия файла. Для начала мы обозначим каждый из 31 общих символов пятиразрядным бинарным кодом: 00000 = «а», 00001 = «б», 00010 = «с» и т. д. Это позволяет 96% файла сократить до  $5/8$  его размера. Последний код 11111 может быть флагом, указывающим, что символ, который нужно передать, не является одним из 31 общих символов. Следующие 8 бит указывают, какой символ представлен ASCII-кодировкой. В результате этого 4% символов во входном файле требуют  $5 + 8 = 13$  бит. Идея состоит в том, чтобы часто используемые символы обозначить малым числом бит, а редко используемые символы — большим числом бит. В этом примере среднее число бит, требуемое на исходный символ, равно  $0.96 \times 5 + 0.04 \times 13 = 5.32$ . Другими словами, общий коэффициент сжатия  $8/5.32$  бит, или около 1.5 : 1.

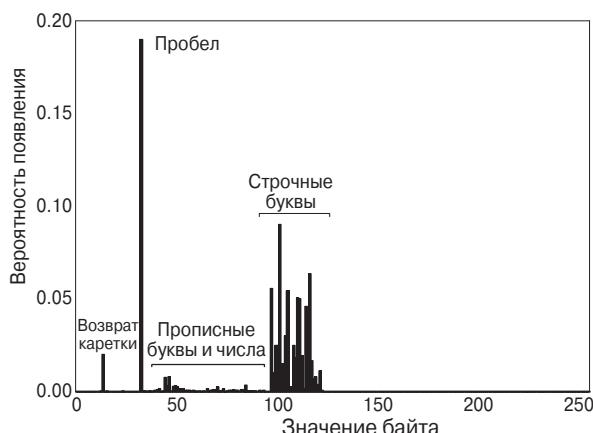


Рис. 27.2. Гистограмма текста. Это гистограмма значений ASCII-кода символов, встречающихся в одной из глав этой книги. Больше всего оказывается строчных букв, пробелов и возвратов каретки.

Кодирование Хаффмана доводит эту идею «до крайности». Символы, которые появляются более часто, такие как точка и пробел, могут обозначаться одним или двумя битами. Нечасто используемые символы, такие как !, @, #, \$, % могут требовать дюжины или более бит. В математических терминах оптимальная ситуация достигается, когда число бит, используемых на каждый символ, пропорционально логарифму вероятности появления символа.

Достоинство кодирования Хаффмана состоит в том, что оно позволяет упаковать вместе коды разной длины. Рассмотрим приём потока последовательных данных из единиц и нулей. Если каждый символ представлен восемью битами, вы можете непосредственно отделить один символ от следующего «отламыванием» 8-битных «ломтиков». Теперь рассмотрим поток данных, закодированный по

Хаффману, где каждый символ может иметь переменное число бит. Как вы отделите один символ от другого? Ответом являются особые свойства кодов Хаффмана, позволяющие выполнить точное разделение. Как это работает, рассмотрим на примере.

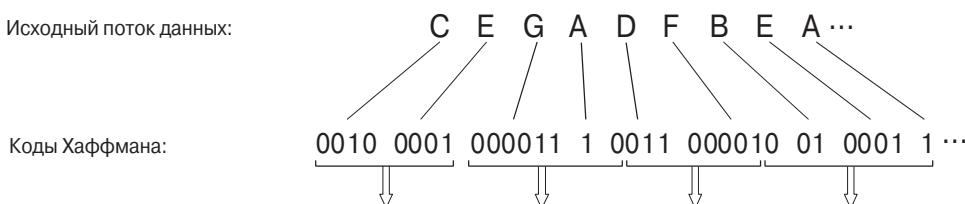
**Рис. 27.3** показывает упрощённую схему кодирования Хаффмана. Символы от А до G появляются в исходном потоке данных с указанной вероятностью. Так как символ А встречается наиболее часто, мы будем представлять его одним битом, кодом 1. Следующий наиболее вероятный символ В получает два бита, т. е. код 01. Так продолжается до самого редкого символа G, обозначенного шестью битами — 000011. Как показано на рисунке, коды переменной длины объединяются в 8-битные группы, стандартные для компьютерного использования.

Когда происходит декомпрессия, все 8-битные группы заменяются от начала до конца длинной последовательной строкой из единиц и нулей. Посмотрите внимательно в кодирующую таблицу **Рис. 27.3** и заметьте, что каждый код состоит из двух частей: некоторое число нулей перед *единицей* и некоторый бинарный код после *единицы*. Это позволяет разбить непрерывный поток двоичных данных на отдельные коды без необходимости в разграничителях или каких-либо специальных символах между кодами в принимаемом потоке. Программа декомпрессии принимает входной поток единиц и нулей, пока не сформируется очередной код, а затем начинает поиск следующего символа. Способ формирования кодов гарантирует, что никакой неопределённости в их последующем разделении не будет.

Пример таблицы кодирования

Буква	Вероятность	Код Хаффмана
A	0.154	1
B	0.110	01
C	0.072	0010
D	0.063	0011
E	0.059	0001
F	0.015	000010
G	0.011	000011

Исходный поток данных:



Коды Хаффмана: 0010 0001 000011 1 0011 000010 01 0001 1 ...

Коды, сгруппированные в байты: 00100001 00001110 01100001 00100011 ...

Байт 1

Байт 2

Байт 3

Байт 4

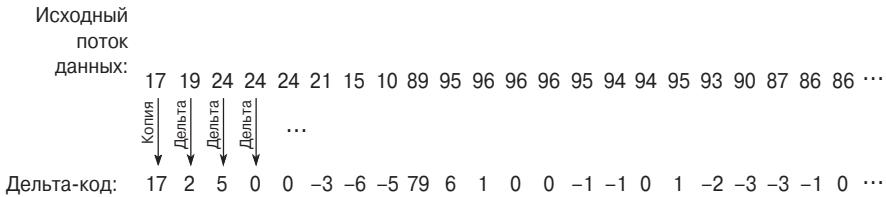
**Рис. 27.3.** Кодирование Хаффмана. Таблица кодирования обозначает каждую из семи букв в этом примере двоичным кодом различной длины, зависящей от вероятности появления этой буквы. Исходный поток данных, состоящий из семи букв, транслируется с помощью этой таблицы в данные, закодированные по Хаффману. Так как каждый из кодов Хаффмана имеет различную длину, двоичные данные группируются в стандартные 8-битные байты для хранения и передачи.

Более сложная версия кодирования Хаффмана называется *арифметическим кодированием*. Такое кодирование ставит в соответствие последовательностям символов коды, длина которых зависит от вероятности появления последовательности. Это даёт лучшее сжатие данных, скажем, на 5...10%. Кодирование длин серий с последующим кодированием Хаффмана или арифметическим кодированием также является общеизвестной стратегией. Но эти алгоритмы достаточно сложны, и мы оставим их для тех, кто специализируется в области сжатия данных.

Чтобы выполнить кодирование Хаффмана или арифметическое кодирование, алгоритмы сжатия и декомпрессии должны быть согласованы друг с другом по кодам, которые используются для представления каждого символа (или группы символов). Это может быть выполнено одним из двух способов. Простейший вариант — использовать определённую таблицу кодирования, которая всегда одинакова независимо от сжимаемой информации. Более сложные схемы используют кодирование, оптимизированное для частных случаев используемых данных. Это требует, чтобы таблица кодирования включалась в сжимаемый файл и затем использовалась программой декомпрессии. Оба метода широко применяются.

## 27.4. Дельта-кодирование

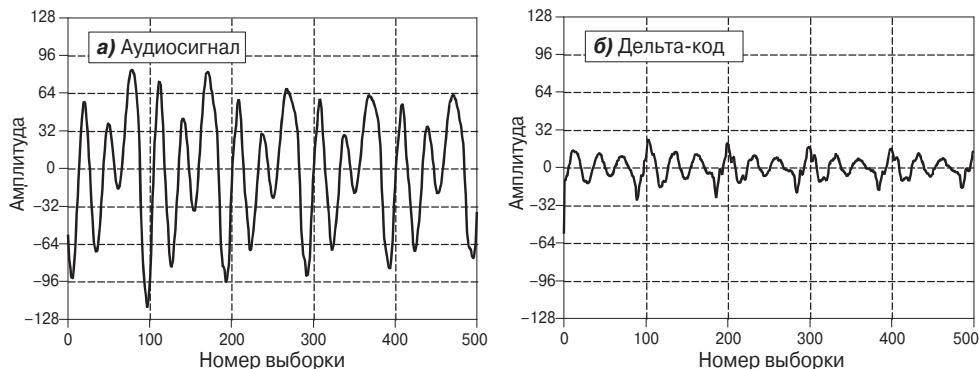
В науке, технике и математике греческой буквой «дельта» ( $\Delta$ ) принято обозначать изменение (*приращение*) переменной. Термин *дельта-кодирование* относится к ряду методик, которые представляют данные через разность между последовательными выборками (или символами), а не непосредственно сами выборки. Рис. 27.4 показывает пример того, как это делается. Первое значение в дельта-кодированном файле является таким же, как первое значение в исходных данных. Все следующие значения в кодированном файле равны разности (дельта) между текущим и предыдущим значениями во входном файле.



**Рис. 27.4.** Пример дельта-кодирования. Первое значение закодированного файла точно такое же, как первое значение из исходного файла. После этого каждая выборка в закодированном файле формируется как разность между текущей и предыдущей выборкой в исходном файле.

Дельта-кодирование может использоваться для сжатия данных, когда значения в исходном файле изменяются достаточно гладко, т. е. между соседними значениями разница относительно невелика. Дельта-кодирование не используют для сжатия ASCII-текстов или исполняемых кодов, но оно широко применяется, когда дело касается сигналов. Например, Рис. 27.5а показывает сегмент аудиосигнала, оцифрованного с точностью 8 бит, при значении –127...127. На (б) показана дельта-кодированная версия этого сигнала. Ключевым здесь является то, что этот сжатый сигнал имеет меньшую амплитуду, чем исходный. Другими словами, дельта-кодирование увеличило вероятность того, что значение каждой выборки

близко к нулю, и уменьшило вероятность больших отклонений от нуля. Эта неравномерная вероятность как раз то, что нужно для кодирования Хаффмана. Если исходный сигнал не изменяется или изменяется по линейному закону, в результате дельта-кодирования будут получаться отрезки с постоянным значением. А это то, что нужно для кодирования длин серий. Таким образом, применение дельта-кодирования совместно с кодированием Хаффмана или кодированием длин серий является общераспространённым методом сжатия данных.



**Рис. 27.5.** Пример дельта-кодирования. На (а) показан аудиосигнал, оцифрованный с точностью 8 бит. На (б) изображена дельта-кодированная версия этого сигнала. Дельта-кодирование полезно для сжатия данных, если кодируемый сигнал медленно изменяется от выборки к выборке.

Идея, используемая в дельта-кодировании, расширена в более сложном методе, называемом *линейным предсказанием*. Чтобы понять, что это за метод, предположим, что первые 99 выборок входного сигнала уже закодированы, и мы работаем с выборкой номер 100. Мы спрашиваем себя: «Какое значение выборки номер 100 наиболее вероятно, опираясь на первые 99 выборок?» В дельта-кодировании ответ состоит в том, что наиболее подходящее значение для выборки номер 100 такое же, как предыдущее значение, т. е. выборки номер 99. Это ожидаемое значение используется как основа для кодирования выборки 100. В кодированный файл помещается разность между выборкой и ожиданием. Линейное предсказание усовершенствует этот подход, делая предположение о самом вероятном значении более эффективным способом. Это достигается за счёт рассмотрения нескольких предыдущих значений, а не только самого последнего. Алгоритмы, лежащие в основе линейного предсказания, похожи на алгоритмы рекурсивной фильтрации и используют Z-преобразование и другой сложный математический аппарат.

## 27.5. LZW-сжатие

*LZW-сжатие* получило своё название по именам его разработчиков: A. Lempel, J. Ziv, а также Terry A. Welch, внесший впоследствии некоторые модификации. Это передовой метод сжатия данных общего назначения, отличающийся простотой и универсальностью. В типовом случае LZW обеспечивает сжатие текстового файла, исполняемого кода или другого подобного файла данных при-

мерно наполовину их исходного размера. LZW также хорошо работает с экстремально избыточными файлами данных, такими как табулированные числа, исходные компьютерные коды и принимаемые сигналы. В этом случае типовой степенью сжатия является отношение 5:1. LZW положен в основу ряда утилит персональных компьютеров, которые обещают: «Удвоим ёмкость ваших жёстких дисков!»

LZW-сжатие всегда используется в GIF-файлах изображений и предлагается как опция в файлах типов TIFF и PostScript. LZW-сжатие защищено патентом США № 4558302 от 10 декабря 1985 г. на Sperry Corporation (теперь Unisys Corporation). За информацией о коммерческой лицензии обращайтесь по адресу: Welch Licensing Department, Law Department, M/SC2SW1, Unisys Corporation, Blue Bell, Pennsylvania, 19424-0001.

LZW-сжатие использует таблицу кодирования, которая показана на Рис. 27.6. Чаще всего выбирают таблицу на 4096 входных значений. В этом случае LZW-кодированные данные содержат полные 12-битные коды, которые указывают на один из входов кодовой таблицы. Декомпрессия достигается выборкой каждого кода из сжатого файла и трансляцией его с помощью кодовой таблицы, чтобы определить, какой символ (или символы) представлен. Коды 0...255 в кодовой таблице всегда выделены для представления одиночных байтов во входном файле. Поэтому, если бы для кодирования использовались только первые 256 кодов, каждый байт исходного файла преобразовывался бы в 12 бит кодированного фай-

**Рис. 27.6.** Пример сжатия кодовой таблицей.

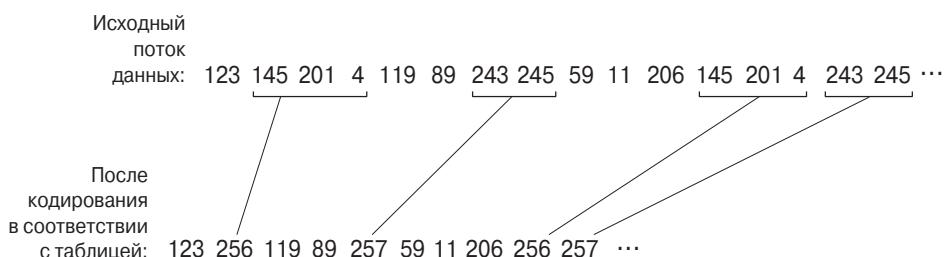
Это основа популярного метода LZW-сжатия.

Процесс кодирования начинается с поиска в сжимаемом файле таких комбинаций байтов, которые имеются в кодовой таблице. Далее эти комбинации заменяются 12-битным кодом, который записывается в файл сжатых данных.

Первые 256 кодов в таблице соответствуют одиночным байтам, а остальные коды относятся к их последовательностям. LZW-алгоритм — это эффективный способ формирования кодирующей таблицы в случае конкретных, заранее известных типов сжимаемых данных. Отметим, что кодовая таблица на этом рисунке является упрощённым примером, а не таблицей, реально созданной алгоритмом LZW.

**Пример таблицы кодирования**

Номер кода	Трансляция
0000	0
0001	1
:	:
0254	254
0255	255
0256	145 201 4
0257	243 245
:	:
4095	xxx xxx xxx



ла, в результате чего размер файла получился бы на 50% больше. В процессе декомпрессии каждый 12-битный код транслировался бы с помощью таблицы кодировки обратно в единственный байт. Конечно, это бесполезная процедура.

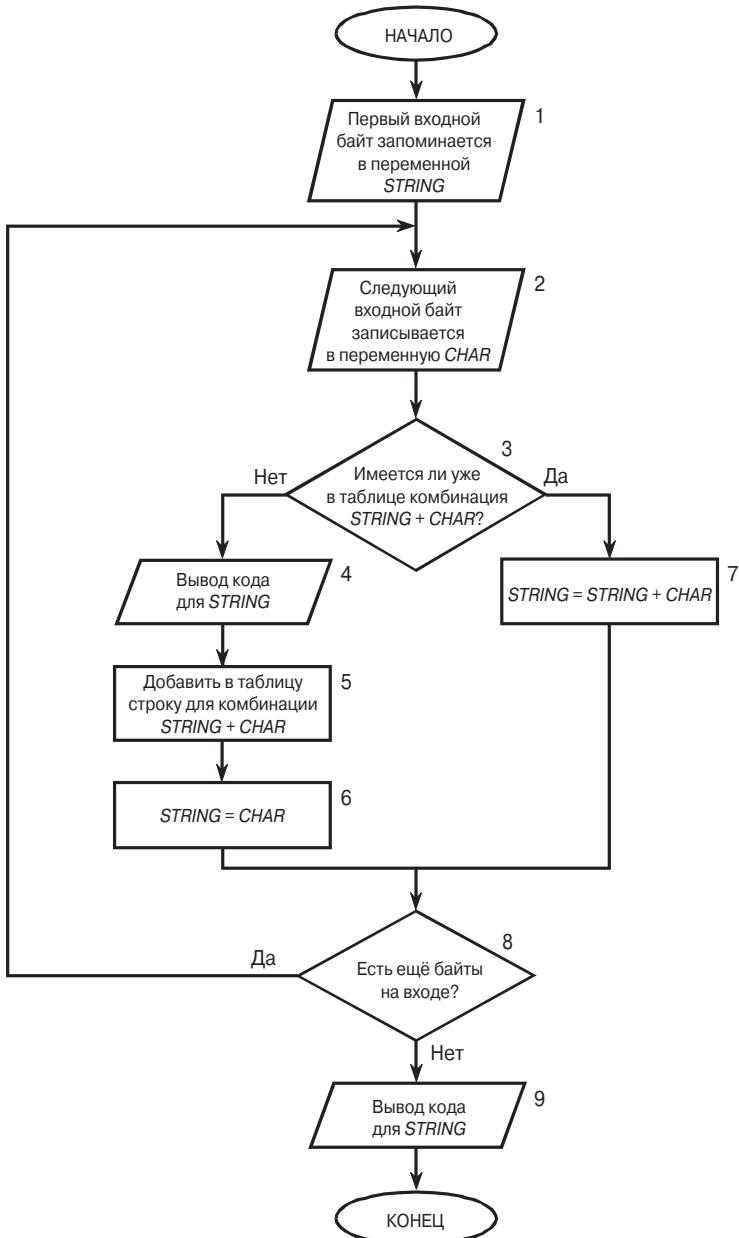
Метод LZW достигает сжатия, используя для представления последовательности данных коды 256...4095. Например, код 523 может представлять последовательность из трёх байтов: 231 124 234. Каждый раз, когда алгоритм сжатия находит эту последовательность в исходном файле, код 523 помещается в кодированный файл. При декомпрессии код 523 транслируется с помощью кодовой таблицы для воссоздания истинной 3-байтовой последовательности. Чем более длинные и более часто встречающиеся последовательности заменяются одним байтом, тем выше достигаемое сжатие.

Хотя это простой подход, имеется два главных затруднения, которые необходимо преодолеть: 1) как определить, какая последовательность должна попасть в кодовую таблицу и 2) как обеспечить программу декомпрессии такой же кодовой таблицей, какая использовалась программой сжатия? LZW-алгоритм легко решает обе эти проблемы.

Когда LZW-программа начинает кодировать файл, кодовая таблица содержит только первые 256 строк, а остальная таблица остаётся пустой. Это означает, что первые коды, проходящие в сжатый файл, являются просто одним байтом из входного файла, преобразованными в 12 бит. Когда продолжается кодирование, алгоритм LZW идентифицирует повторяющиеся последовательности в данных и добавляет их в кодовую таблицу. Сжатие начинается, когда последовательность отыскивается во второй раз. Ключевой момент здесь состоит в том, что последовательность из входного файла не может оказаться в кодовой таблице, пока она не попадёт в сжатый файл в виде отдельных символов (коды 0...255). Это важно, потому что за счёт этого у программы декомпрессии появляется возможность восстановить кодовую таблицу непосредственно из сжатых данных без специальной передачи кодовой таблицы.

**Рис. 27.7** показывает структурную схему алгоритма LZW-сжатия. **Табл. 27.3** иллюстрирует последовательные шаги, осуществляемые в примере сжатия входного файла, содержащего 45-байтовую текстовую строку в коде ASCII: «*the/rain/in/Spain/falls/mainly/on/the/plain*». Когда мы говорим, что алгоритм LZW читает символ «*a*» из входного файла, мы знаем, что он читает значение 01100001 (97, выраженное восемью битами, где 97 — это «*a*» в ASCII). Когда мы говорим, что записываем символ «*a*» в кодированный файл, мы обозначаем эту запись 000001100001 (97, выраженное 12 битами).

Алгоритм сжатия использует две переменные — STRING и CHAR. Переменная CHAR содержит один символ, т. е. однобайтовое значение 0...255. Переменная STRING — это строка переменной длины, т. е. группа из одного (или более) символов (каждый символ — один байт). В блоке 1 **Рис. 27.7** программа начинается с выбора одного байта из входного файла и записи его в переменную STRING. **Табл. 27.3** показывает это действие в строке 1. За ней следует циклический алгоритм по каждому добавляемому байту во входном файле, управляемому блоком 8 структурной схемы. Каждый раз байт читается из входного файла (блок 2) и сохраняется в переменной CHAR. В таблице данных затем осуществляется поиск для определения того, имеется ли конкатенация из двух переменных STRING + CHAR, обозначенная кодом (блок 3).



**Рис. 27.7.** Структурная схема алгоритма LZW-сжатия. Переменная **CHAR** — это один байт. Переменная **STRING** — это последовательность символов произвольной длины. Данные читаются из входного файла (блоки 1 и 2) как простые байты и записываются в сжатый файл (блок 4) как 12-битный код. **Табл. 27.3** показывает пример этого алгоритма.

Таблица 27.3. Пример LZW-кодирования. Приводится пример сжатия фразы  
 «the/rain/in/Spain/falls/mainly/on/the/plain»

	CHAR	STRING +CHAR	Уже присутствует в таблице?	Выход	Добавить в таблицу	Новое значение STRING	Комментарий
1	t	t				t	Первый символ – нет действий
2	h	th	нет	t	256 = th	h	
3	e	he	нет	h	257 = he	e	
4	/	e/	нет	e	258 = e/	/	
5	r	/r	нет	/	259 = /r	r	
6	a	ra	нет	r	260 = ra	a	
7	i	ai	нет	a	261 = ai	i	
8	n	in	нет	i	262 = in	n	
9	/	n/	нет	n	263 = n/	/	
10	i	/i	нет	/	264 = /i	i	
11	n	in	да (262)			in	Найдено первое совпадение
12	/	in/	нет	262	265 = in/	/	
13	S	/S	нет	/	266 = /S	S	
14	p	Sp	нет	S	267 = Sp	p	
15	a	pa	нет	p	268 = pa	a	
16	i	ai	да (261)			ai	Совпадение ai, ain ещё нет в таблице
17	n	ain	нет	261	269 = ain	n	ain добавляется в таблицу
18	/	n/	да (263)			n/	
19	f	n/f	нет	263	270 = n/f	f	
20	a	fa	нет	f	271 = fa	a	
21	l	al	нет	a	272 = al	l	
22	l	ll	нет	l	273 = ll	l	
23	s	ls	нет	l	274 = ls	s	
24	/	s/	нет	s	275 = s/	/	
25	m	/m	нет	/	276 = /m	m	
26	a	ma	нет	m	277 = ma	a	
27	i	ai	да (261)			ai	Совпадение ai
28	n	ain	да (269)			ain	Совпадает более длинная строка ain
29	l	ainl	нет	269	278 = ainl	l	
30	y	ly	нет	l	279 = ly	y	
31	/	y/	нет	y	280 = y/	/	
32	o	/o	нет	/	281 = /o	o	
33	n	on	нет	o	282 = on	n	
34	/	n/	да (263)			n/	
35	t	n/t	нет	263	283 = n/t	t	
36	h	th	да (256)			th	Совпадение th, the ещё нет в таблице
37	e	the	нет	256	284 = the	e	the добавляется в таблицу
38	/	e/	да			e/	
39	p	e/p	нет	258	285 = e/p	p	
40	l	pl	нет	p	286 = pl	l	
41	a	la	нет	l	287 = la	a	
42	i	ai	да (261)			ai	Совпадение ai
43	n	ain	да (269)			ain	Совпадает более длинная строка ain
44	/	ain./	нет	269	288 = ain/	/	
45	EOF	/		/			Конец файла, вывод STRING

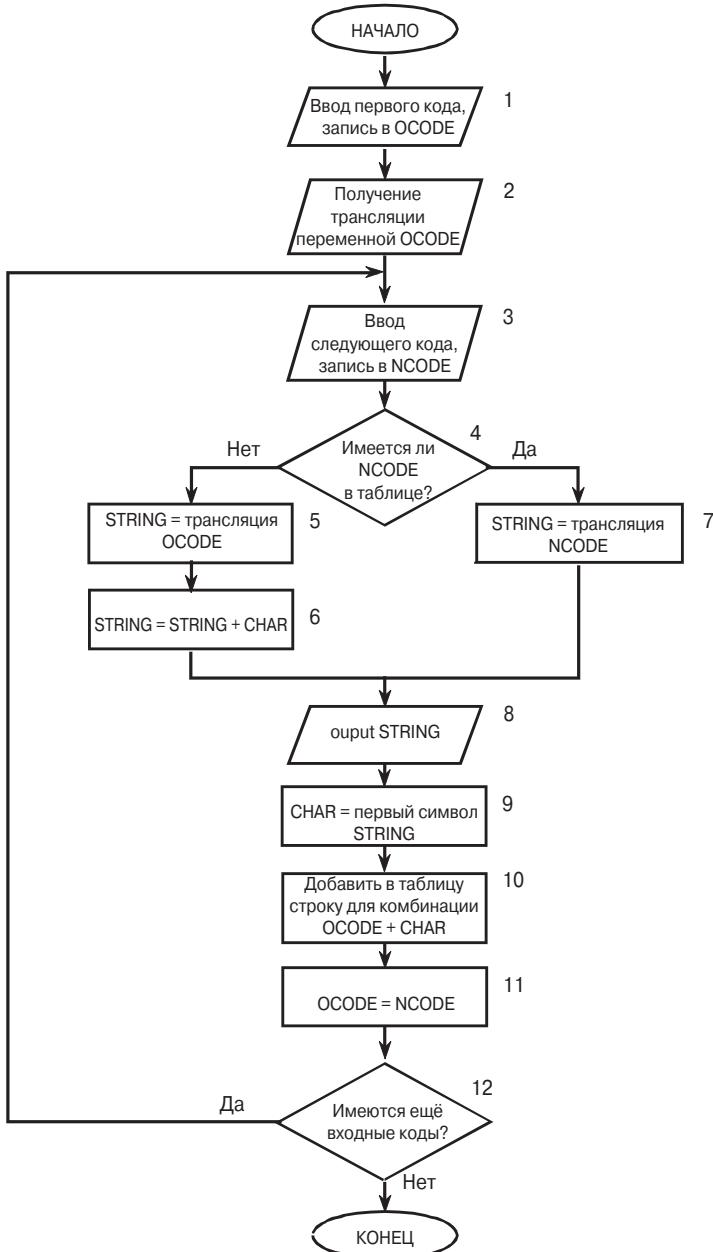
Если в кодовой таблице совпадение кодов не найдено, выполняются три действия, как показано в блоках 4, 5 и 6. В блоке 4 12-битный код, соответствующий содержанию переменной STRING, записывается в сжатый файл. В блоке 5 создаётся новый код в таблице для конкатенации STRING + CHAR. В блоке 6 переменная STRING принимает значение переменной CHAR. Пример этого действия показан в строках 2...10 в **Табл. 27.3** для первых десяти байтов файла.

Если же совпадение в кодовой таблице найдено (блок 3), конкатенация STRING + CHAR сохраняется в переменной STRING без каких-либо иных действий (блок 7). То есть, если совпадающая последовательность найдена в кодовой таблице, то никаких действий не предпринимается, пока не будет определено, имеется ли ещё более длинная совпадающая последовательность в таблице. Пример этого показан в строке 11, где последовательность STRING + CHAR = *in* определена как уже имеющаяся в кодовой таблице. В строке 12 следующий символ из входного файла «/» добавляется к последовательности, и в кодовой таблице ищется «*in/*». Так как более длинная последовательность отсутствует в таблице, программа добавляет её к таблице, выводит код для более короткой последовательности, которая имеется в таблице (код 262), и начинает поиск последовательности, начиная с символа «/». Этот поток событий продолжается до тех пор, пока во входном файле не исчерпаются все символы. Программа выходит из цикла и записывает текущее значение STRING в выходной файл (блок 9 **Рис. 27.7** или строка 45 **Табл. 27.3**).

Структурная схема алгоритма декомпрессии LZW показана на **Рис. 27.8**. Каждый код читается из сжатого файла и сравнивается с элементами кодовой таблицы для обеспечения трансляции. В процессе такой обработки каждого кода кодовая таблица постепенно изменяется так, что она оказывается идентична таблице, использованной при сжатии. Однако имеется небольшая тонкость. Существуют некоторые комбинации данных, которые приводят к появлению кодов, ещё не сформированных в кодовой таблице в процессе выполнения алгоритма декомпрессии. Эта ситуация управляется блоками 4, 5 и 6.

Всего несколько десятков строк программного кода требуется для реализации наиболее простых алгоритмов LZW. Основное затруднение вызывает эффективное формирование кодовой таблицы. Если решать проблему «в лоб», потребуется большой объём памяти, а скорость выполнения задачи окажется невысокой. В коммерческих LZW-программах производители используют ряд хитростей, позволяющих повысить их эффективность. Например, проблема с памятью возникает из-за того, что длина каждой из строк символов для каждой кодовой комбинации неизвестна заранее. Большинство программ LZW преодолевают эту проблему, используя избыточность кодовой таблицы. Например, рассмотрим строку 29 в **Табл. 27.3**, где код 278 определён для *ainl*. Вместо того чтобы записывать в файл эти 4 байта, код 278 может быть сохранён как 269 + *l*, где код 269 был ранее поставлен в соответствие *ain* в строке 17. Точно так же код 269 может быть сохранён как 261 + *n*, где код 261 определён для *ai* в строке 7. Это остаётся справедливым во всех подобных случаях: каждый код может быть выражен как предыдущий код плюс один новый символ.

Время выполнения алгоритма сжатия ограничивается поиском в кодовой таблице совпадений. Проведём аналогию со следующей ситуацией. Предположим, вы хотите найти фамилию вашего друга в телефонном справочнике. Сложность в



**Рис. 27.8.** Структурная схема алгоритма декомпрессии LZW. Переменные OCODE и NCODE (старый код и новый код) хранят 12-битные коды сжатого файла, CHAR хранит один байт, STRING хранит строку байтов.

том, что список упорядочен по номерам телефонов, а не по алфавиту. Это потребует от вас просмотра страниц одной за другой в попытке найти имя, которое вам нужно. Конечно, это очень неэффективно и прямо соответствует ситуации поиска среди 4096 кодов одного, соответствующего определённой символьной стро-

ке. Сделаем вывод: необходимо организовать кодовую таблицу так, чтобы удобно было осуществлять поиск по требуемому параметру (подобно упорядоченному в алфавитном порядке телефонном справочнике). Другими словами, не следует размещать 4096 кодов в памяти последовательно. Лучше разбить память на секции по принципу, где какие последовательности будут находиться. Например, предположим, что мы хотим определить, имеется ли в кодовой таблице последовательность вида «код 329 + x». Кодовая таблица должна быть организована так, чтобы символ «x» указывал, где начинать поиск. Существует большое количество различных схем такой работы с кодовыми таблицами, и многие из них отличаются очень высокой сложностью.

Заканчивая обсуждение LZW и подобных схем сжатия, отметим, что это сфера большой конкуренции. Основы сжатия данных относительно просты, но существует огромное число различных типов программ, реализуемых как коммерческие продукты и являющихся достаточно сложными. Компании делают деньги, продавая вам программы, которые выполняют сжатие, и бдительно защищают свои фирменные секреты патентами и свидетельствами. Не думайте, что вам удастся достигнуть такого же уровня производительности программы сжатия, созданной за несколько часов работы.

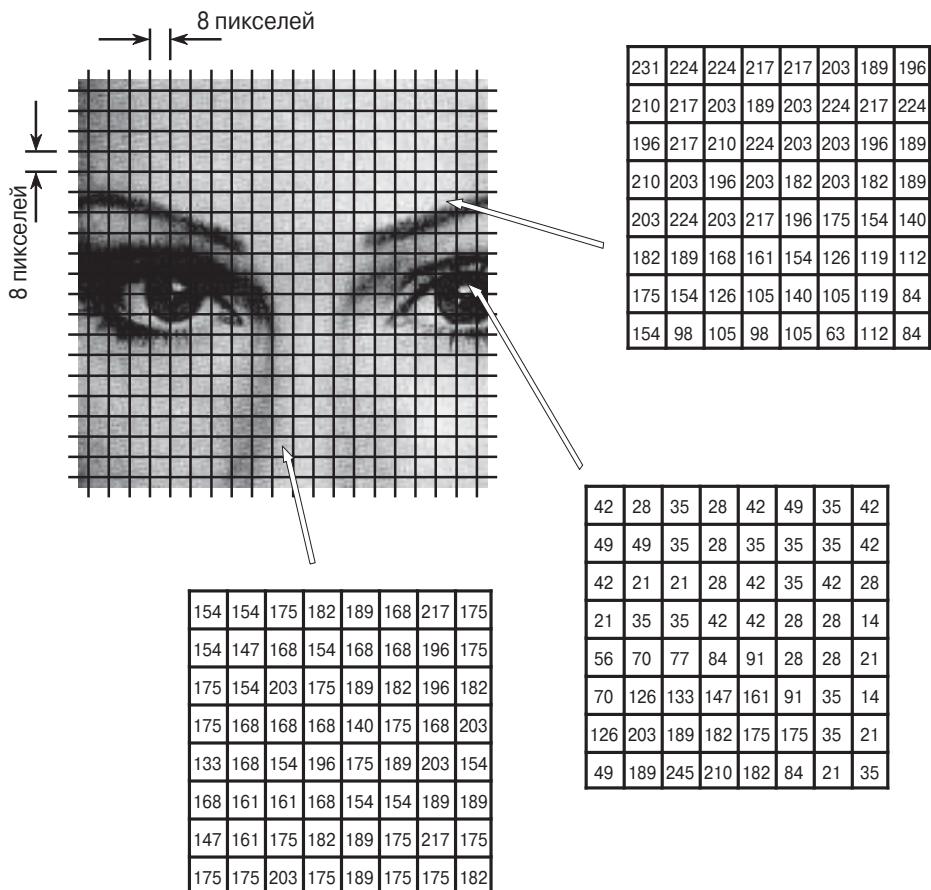
## 27.6. Сжатие с преобразованием

Разработано большое число методов сжатия с потерями, однако семейство методов, называемое *сжатие с преобразованием*, доказало свою наибольшую ценность. Самый лучший пример преобразующего сжатия реализован в популярном стандарте кодирования изображений *Joint Photographic Experts Group (JPEG)*. Мы опишем работу JPEG, чтобы проиллюстрировать, как работает сжатие с потерями.

Мы уже обсуждали простой метод сжатия данных с потерями, именуемый «грубая выборка и/или квантование» (CS&Q в **Табл. 27.1**). Он включает в себя уменьшение числа бит на выборку или полное удаление некоторых выборок. Обе эти процедуры несут желаемый эффект: файл данных становится меньше, но за счёт ухудшения качества сигнала. Как вы могли ожидать, эти простые методы работают не очень хорошо.

Сжатие с преобразованием основано на простом допущении: когда сигнал подвергается преобразованию Фурье (или другому преобразованию), результирующие данные не будут соответствовать исходным по переносимой ими информации. В частности, низкочастотные компоненты сигнала более важны, чем высокочастотные. Удаление, скажем, 50% бит высокочастотных компонентов уменьшает количество закодированной информации только на 5%.

Как показано на **Рис. 27.9**, сжатие JPEG начинается с разбиения изображения на группы размером  $8 \times 8$  пикселей. Полный алгоритм JPEG может выделять большое число бит на пиксель, включая использование информации о цвете. В рассматриваемом примере каждый пиксель представлен одним байтом, значение которого лежит в шкале серого 0...255. Эти группы размером  $8 \times 8$  пикселей обрабатываются независимо. Каждая группа первоначально представлена 64 байтами. После преобразования и удаления данных каждая группа оказывается пред-



**Рис. 27.9.** Разбиение изображения при JPEG-сжатии. Преобразующее сжатие JPEG начинается с разбиения изображения на группы размером 8×8, т. е. 64 пикселя. Три таких группы показаны на рисунке в увеличенном масштабе, отображая значение каждого пикселя — однобайтное число 0...255.

ставлена байтами в количестве примерно от 2 до 20. В течение декомпрессии обратное преобразование осуществляется над группами, включающими от 2 до 20 байт, и формируется аппроксимация первоначальных групп размером 8×8. Эти группы затем «монтируются» вместе и образуют восстановленное изображение. Почему используют группы 8×8 пикселей, а не, например, 16×16? Группирование 8×8 было основано на максимальном размере, принятом в технологии интегральных схем в то время, когда разрабатывался стандарт. Во всяком случае размер 8×8 работает хорошо, и он может как измениться, так и не измениться в будущем.

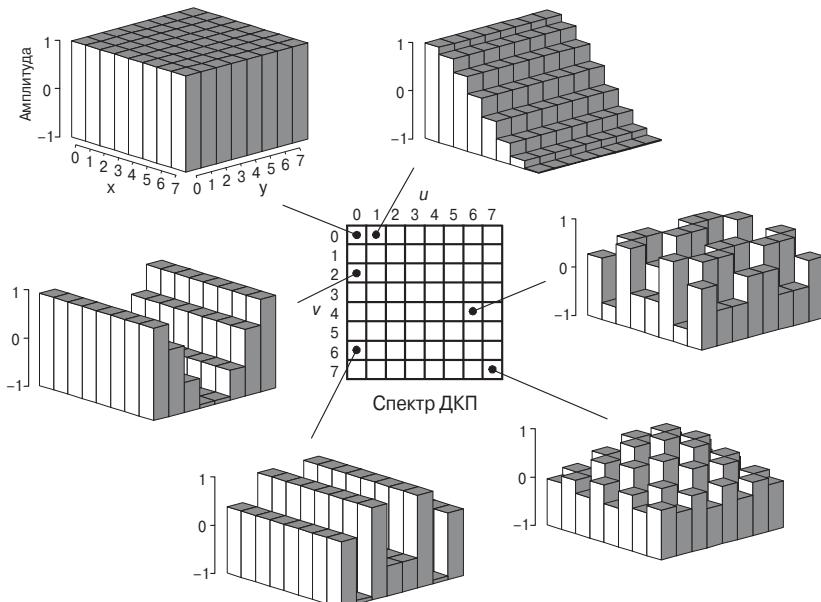
Было исследовано много разных видов преобразований на предмет возможности использования для сжатия данных. Некоторые из них были даже специально изобретены для этих целей. Например, *преобразование Кархунена—Лоева* обеспечивает наибольшую возможную степень сжатия, но оно трудно в применении. *Преобразование Фурье* легко использовать, но оно не обеспечивает адекватное сжатие. В результате соперничества победителем оказалось родственное преобразованию Фурье *дискретное косинусное преобразование (ДКП)*.

В то время как преобразование Фурье использует синусные и косинусные волны для представления сигнала, ДКП использует только косинусные волны. Имеется несколько версий ДКП с небольшими различиями в математике. Как пример одной из версий рассмотрим 129-точечный сигнал, включающий выборки 0...128. Теперь создадим 256-точечный сигнал, дублируя выборки 1...127, дополняя ими сигнал в точках 255...130, т. е. 1, 2, ..., 127, 128, 127, ..., 2, 1. Выполнив преобразование Фурье 256-точечного сигнала, получим спектр из 129 точек 0...128. Так как сигнал во временной области был искусственно сделан симметричным, мнимая часть спектра будет состоять из одних нулей. Другими словами, мы начинали со 129 точками во временной области и закончили с частотным спектром из 129 точек, каждая из которых представляет собой амплитуду косинусной волны. Вуаля: вот вам ДКП!

Когда ДКП берётся от группы  $8 \times 8$  пикселей, в результате получается спектр  $8 \times 8$ . Другими словами, 64 числа преобразуются в другие 64 числа. Все эти значения *действительные*, комплексной математики здесь нет. Так же, как и в анализе Фурье, каждое значение в спектре есть амплитуда *базисной функции*. **Рис. 27.10** показывает 6 из 64 базисных функций, используемых в ДКП  $8 \times 8$ , с указанием того, где размещены соответствующие амплитуды. Базисные функции ДКП  $8 \times 8$  задаются уравнением:

$$b[x, y] = \cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right]. \quad (27.1)$$

Базисные функции ДКП. Переменные  $x$  и  $y$  являются индексами в пространственной области,  $u$  и  $v$  — индексами в частотной области. Для ДКП  $8 \times 8$  все индексы проходят от 0...7.



**Рис. 27.10.** Базисные функции ДКП. Спектр ДКП состоит из массива  $8 \times 8$  элементов. Каждый элемент является амплитудой одной из 64 базисных функций. Шесть из этих базисных функций показаны здесь с указанием того, где размещены соответствующие амплитуды.

Низкие частоты находятся в верхнем левом углу спектра, а высокие частоты — в правом нижнем углу. Постоянная составляющая в точке  $[0, 0]$  — самое верхнее левое значение. Базисная функция для  $[0, 1]$  — это половина периода косинусной волны в одном направлении и постоянное значение в другом. Базисная функция для  $[1, 0]$  аналогична, но повернута на  $90^\circ$ .

ДКП вычисляет спектр, коррелируя группу  $8 \times 8$  пикселей с каждой из базисных функций. То есть значение точки спектра находится умножением соответствующей базисной функции на группу  $8 \times 8$  пикселей и суммированием произведений. После этого необходимо выполнить две корректировки, завершающие вычисление ДКП (так же, как и при преобразовании Фурье). Во-первых, надо разделить 15 спектральных значений в строке 0 столбца 0 на два. Во-вторых, необходимо разделить все 64 значения в спектре на 16. При вычислении обратного ДКП для каждой амплитуды в спектре выбирается соответствующая базисная функция и производится суммирование, восстанавливающее изображение в пространственной области. Никаких сложных шагов не требуется. Точно такая же концепция, как и при анализе Фурье, но с другими базисными функциями.

**Рис. 27.11** иллюстрирует кодирование JPEG для трёх групп  $8 \times 8$  пикселей, выбранных на **Рис. 27.9**. В левой колонке (**а**, **б** и **в**) показаны исходные значения пикселей. В средней колонке (**г**, **д** и **е**) приводятся ДКП-спектры выбранных групп. В правой колонке (**ж**, **з** и **и**) иллюстрируется влияние уменьшения числа битов, выделяемых на представление каждого компонента частотного спектра. Например, (**ж**) сформирован сокращением каждой выборки (**г**) до 10 бит выполнением инверсии ДКП и затем вычитанием реконструированного изображения из оригинала. Аналогично сформированы (**з** и **и**) при сокращении количества битов каждой выборки в спектре до 8 и 5 бит соответственно. Как и ожидалось, ошибка восстановления возрастает при уменьшении числа бит, используемых для представления данных. Как пример этого сокращения числа бит, спектры, показанные в средней колонке, имеют по 8 бит на значение спектра, лежащего в диапазоне 0...255 для постоянной составляющей и  $-127\dots 127$  — для остальных значений.

Второй метод сжатия в частотной области состоит в отбрасывании некоторых из 64 спектральных значений. Как показывают спектры на **Рис. 27.11**, основная мощность сигналов сконцентрирована в области низких частот. Это означает, что наиболее высокие частотные компоненты могут быть исключены, что не приведет к сколько-нибудь существенному ухудшению качества. **Рис. 27.12** показывает, как изменяется изображение, если осуществлять отбрасывание разного количества высокочастотных компонент. Группа пикселей размером  $8 \times 8$ , используемая в этом примере, — это изображение глаза на **Рис. 27.10**. **Рис. 27.12г** показывает точную реконструкцию, использующую все 64 спектральных значения. Остальные рисунки иллюстрируют восстановление только на основе указанного числа самых низких частотных компонентов. Как показано на **Рис. 27.12в**, даже при удалении  $3/4$  высокочастотных составляющих спектра ошибка восстановления остаётся небольшой. Она скорее похожа на небольшой белый шум.

JPEG — это хороший пример того, как различные схемы сжатия данных могут комбинироваться для достижения большей эффективности. Полная процедура JPEG реализуется следующими основными шагами. Во-первых, изображение разбивается на группы размером  $8 \times 8$  пикселей. Во-вторых, от каждой группы бе-

Исходные группы пикселей

ДКП-спектры

Ошибка квантования

**а) Бровь**

231	224	224	217	217	203	189	196
210	217	203	189	203	224	217	224
196	217	210	224	203	203	196	189
210	203	196	203	182	203	182	189
203	224	203	217	196	175	154	140
182	189	168	161	154	126	119	112
175	154	126	105	140	105	119	84
154	98	105	98	105	63	112	84

**г) Спектр брови**

174	19	0	3	1	0	-3	1
52	-13	-3	-4	-4	-4	5	-8
-18	-4	8	3	3	2	0	9
5	12	-4	0	0	-5	-1	0
1	2	-2	-1	4	4	2	0
-1	2	1	3	0	0	1	1
-2	5	-5	-5	3	2	-1	-1
3	5	-7	0	0	0	-4	0

**ж) При использовании 10 бит**

0	0	0	0	-1	0	0	0
-1	0	0	0	0	0	0	-1
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	1	0	0	0	-1	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

**б) Глаз**

42	28	35	28	42	49	35	42
49	49	35	28	35	35	35	42
42	21	21	28	42	35	42	28
21	35	35	42	42	28	28	14
56	70	77	84	91	28	28	21
70	126	133	147	161	91	35	14
126	203	189	182	175	175	35	21
49	189	245	210	182	84	21	35

**д) Спектр глаза**

70	24	-28	-4	-2	-10	-1	0
-53	-35	43	13	7	13	1	3
23	9	-10	-8	-7	-6	5	-3
6	2	-2	8	2	-1	0	-1
-10	-2	-1	-12	2	1	-1	4
3	0	0	11	-4	-1	5	6
-3	-5	-5	-4	3	2	-3	5
3	0	4	5	1	2	1	0

**з) При использовании 8 бит**

0	-3	-1	-1	1	0	0	-1
1	0	-1	-1	0	0	0	-1
-1	-2	1	0	-2	0	-2	-2
-1	-2	-1	2	0	2	0	1
0	-2	1	0	0	1	0	0
0	-4	-1	0	1	0	0	0
0	-2	0	1	-1	-1	1	-1
-1	-3	1	1	1	-3	-2	-1

**в) Нос**

154	154	175	182	189	168	217	175
154	147	168	154	168	168	196	175
175	154	203	175	189	182	196	182
175	168	168	168	140	175	168	203
133	168	154	196	175	189	203	154
168	161	161	168	154	154	189	189
147	161	175	182	189	175	217	175
175	175	203	175	189	175	175	182

**е) Спектр носа**

174	-11	-2	-3	-3	6	-3	4
-2	-3	1	2	0	3	1	2
3	0	-4	0	0	0	-1	9
-4	-6	-2	1	-1	4	-10	-3
1	2	-2	0	0	-2	0	-5
3	-1	3	-2	2	1	1	0
3	5	2	-2	3	0	4	3
4	-3	-13	3	-4	3	-5	3

**и) При использовании 5 бит**

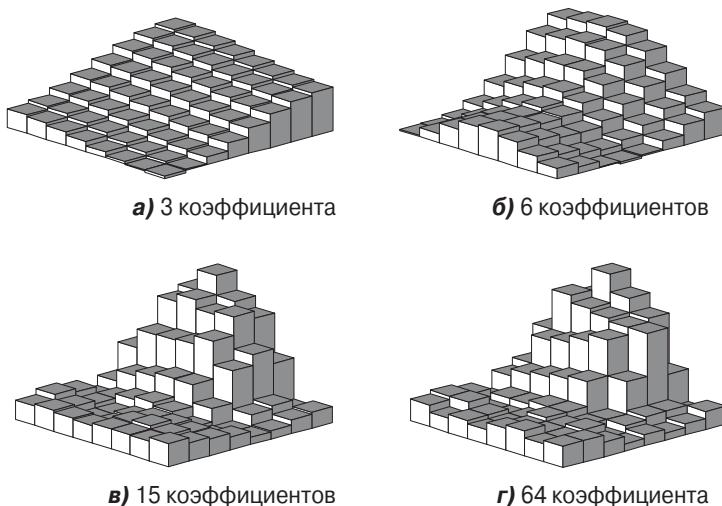
-13	-7	1	4	0	0	10	-2
-22	6	-13	5	-5	2	-2	-13
-9	-15	0	-17	-8	8	12	25
-9	16	1	9	1	-5	-5	13
-20	-3	-13	-16	-19	-1	-4	-22
-11	6	-8	16	-9	-3	-7	6
-14	10	-9	4	-15	3	3	-4
-13	19	12	9	18	5	-5	10

**Рис. 27.11.** Пример кодирования JPEG. В левой колонке показаны исходные группы 8×8 пикселей, выбранные на Рис. 27.9. В средней колонке приводятся ДКП-спектры этих трёх групп пикселей. Третья колонка иллюстрирует ошибку восстановления исходного изображения, являющуюся следствием конечности числа битов, выделяемых для представления каждого спектрального отсчёта.

рётся ДКП-преобразование. В-третьих, каждый полученный спектр размером 8×8 точек сжимается указанными выше методами (сокращением числа битов и удалением ряда компонентов). Это делается одним действием с использованием таблицы квантования. Два примера таблицы квантования приведены на Рис. 27.13. Каждое значение в спектре делится на соответствующее значение в таблице квантования, и результат округляется до ближайшего целого числа. Например, верхнее левое значение в таблице квантования равно единице. Поэтому значение постоянной составляющей не изменяется. Для сравнения: нижний пра-

вый элемент на (а) равен 16, означая, что исходный диапазон значений  $-127\dots127$  уменьшится до интервала  $-7\dots7$ . Другими словами, точность значения уменьшится с восьми до четырёх бит. В самом крайнем случае, которому соответствует значение нижнего правого элемента таблицы на (б), равное 256, спектральное значение полностью удаляется.

На четвёртом шаге JPEG-кодирования модифицированный спектр преобразуется из массива  $8\times8$  элементов в линейную последовательность. На этом шаге используется кривая, изображённая на Рис. 27.14, помогая размещать все высокочастотные компоненты в конце линейной последовательности. При этом нули, оставшиеся от удалённых компонентов, располагаются в виде единых фрагментов (серий) большой длины. Пятый шаг процедуры кодирования сжимает эти фрагменты нулей по схеме кодирования длин серий. На шестом шаге полученная



**Рис. 27.12.** Пример восстановления JPEG. Группа пикселей размером  $8\times8$ , используемая в данном примере, соответствует изображению глаза на Рис. 27.9. Как показано на рисунке, менее  $1/4$  значений оказывается достаточно для хорошей аппроксимации исходного изображения.

**а)** Низкая степень сжатия

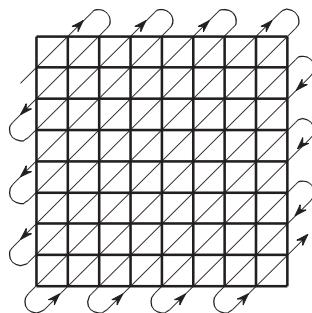
1	1	1	1	1	2	2	4
1	1	1	1	1	2	2	4
1	1	1	1	2	2	2	4
1	1	1	1	2	2	4	8
1	1	2	2	2	2	4	8
2	2	2	2	2	4	8	8
2	2	2	4	4	8	8	16
4	4	4	4	8	8	16	16

**б)** Высокая степень сжатия

1	2	4	8	16	32	64	128
2	4	4	8	16	32	64	128
4	4	8	16	32	64	128	128
8	8	16	32	64	128	128	256
16	16	32	64	128	128	256	256
32	32	64	128	128	256	256	256
64	64	128	128	256	256	256	256
128	128	128	256	256	256	256	256

**Рис. 27.13.** Таблицы квантования JPEG. Приводятся два примера таблиц квантования, которые могут использоваться в процессе сжатия данных. Каждое значение ДКП-спектра делится на соответствующее значение в таблице квантования, и результат округляется до ближайшего целого числа.

**Рис. 27.14.** Последовательность процедур JPEG-кодирования. Показанная кривая используется для преобразования ДКП-спектра в виде массива 8×8 элементов в линейную последовательность 64 значений. Такой вид кривой размещает все высокочастотные компоненты вместе, после чего большое число идущих подряд нулевых элементов может быть эффективно сжато с использованием кодирования длин серий.



последовательность кодируется или кодом Хаффмана, или арифметическим кодированием, в результате чего формируется конечный сжатый файл.

Степень сжатия и результирующие потери качества изображения могут быть проанализированы при выполнении программы JPEG-сжатия. **Рис. 27.15** пока-



**а)** Исходное изображение



**б)** Сжатие 10:1



**в)** Сжатие 45:1

**Рис. 27.15.** Пример искажений при JPEG-сжатии. На (а) показано исходное изображение, а (б и в) показывают восстановленное изображение при степени сжатия 10:1 и 45:1 соответственно. Высокая степень сжатия, используемая в случае (в), представляет группу пикселей 8×8 менее чем 12 битами.

зыает, как искажается изображение, если степень сжатия выбрана слишком большой. При степени сжатия 45:1, что соответствует иллюстрируемому случаю, каждая группа  $8 \times 8$  пикселей заменяется только 12 битами. Подробный анализ изображения позволяет сделать вывод, что здесь в некоторой степени присутствуют лишь шесть самых низкочастотных базисных функций.

Почему для сжатия изображений ДКП оказывается лучше, чем преобразование Фурье? Главная причина состоит в том, что ДКП использует половины периодов базисных функций, т. е.  $S[0, 1]$  и  $S[1, 0]$ . Как показано на Рис. 27.10, они медленно спадают с одного конца массива к другому. Для сравнения: самые низкие частоты при преобразовании Фурье имеют один полный период. Изображения почти всегда содержат области, где яркость изменяется плавно. Использование базисной функции, которая ведёт себя аналогичным образом, позволяет получить лучшее сжатие.

## 27.7. MPEG

*MPEG* — это стандарт сжатия цифровых видеопоследовательностей, которые используются в компьютерном видео и сетях цифрового телевидения. К тому же MPEG обеспечивает сжатие саундтреков, обычно связанных с видео. Наименование стандарта происходит от названия разработавшей его организации *Moving Pictures Experts Group*. Если вы думаете, что JPEG — это сложно, то MPEG — это просто кошмар. MPEG — это то, что вы можете только купить, даже не пытаясь написать программу самому. Будущее этой технологии — использование для реализации алгоритмов сжатия и декомпрессии непосредственно на интегральных схемах. Потенциал MPEG огромен. Представьте себе тысячи видеоканалов, совмещённых в одном оптическом кабеле, подходящем к вашему дому. MPEG — это одна из ключевых технологий XXI в.

В дополнение к уменьшению требуемой скорости данных MPEG имеет ещё несколько важных свойств. Он позволяет проигрывать кино и вперёд, и назад с нормальной или с повышенной скоростью. К кодированной информации есть прямой доступ, т. е. любой конкретный кадр в последовательности может быть легко показан как неподвижная картинка. Это делает фильм редактируемым, т. е. короткие сегменты из фильма могут быть закодированы только со ссылкой на них самих, а не во всей последовательности. MPEG создан устойчивым к ошибкам. Вам ведь не хочется, чтобы небольшие ошибки в нескольких битах вызвали повреждение всего видеофильма.

Подход, используемый MPEG, может быть разделён на два типа сжатия: сжатие внутри кадра и сжатие между кадрами. Сжатие внутри кадра означает, что конкретные кадры, составляющие видеопоследовательность, кодируются, как обычные неподвижные картинки. Это сжатие выполняется на основе JPEG-стандартта с небольшими модификациями. В MPEG-терминологии кадр, который нужно закодировать этим способом, называется *I-картинкой* (*I-picture* — *intra-coded*).

Большинство пикселей в видеопоследовательности изменяется очень мало от одного кадра к следующему. Если камера не движется, то большинство изображений включает фон, который остаётся постоянным в течение десятков кадров.

MPEG использует этот факт и с помощью усложненной формы дельта-кодирования сжимает избыточную информацию между кадрами. После сжатия одного из кадров как I-картинки MPEG кодирует последовательность кадров. Это называется предсказывающее кодирование или *P-картинка* (*P-picture — predictive-coded*). При этом только те пиксели, которые изменились по сравнению с I-картинкой, включаются в P-картинку.

Хотя эти две схемы сжатия формируют костяк стандарта MPEG, в действительности он гораздо более сложен, чем то, что описано в этой книге. Например, P-картинка может ссылаться на I-картинку, сдвинутую так, чтобы учесть движение объектов, произошедшее за время смены кадров в последовательности. Существует также двунаправленное предсказывающее кодирование или *B-картинки*. Они ссылаются как на предыдущую, так и на будущую I-картинку. Такая схема кодирования применяется к фрагментам изображений, которые постепенно изменяются в течение многих кадров. Каждый кадр может также быть сохранён не в том порядке, в котором он следует в исходной последовательности изображений, делая возможным более рациональное построение последовательности I-, P- и B-картинок. Добавление информации о цвете и звуке делает стандарт ещё более сложным.

Основные искажения, которые допускает MPEG, происходят, когда большие фрагменты изображений быстро изменяются от одного кадра к следующему. В действительности, чтобы успеть отследить быстрое изменение сцены, нужно резкое увеличение количества информации. Но если скорость данных фиксирована, зритель заметит «зависание» образов при переходе от одной сцены к другой. Этот эффект может быть сведён к минимуму, если используются сети, способные одновременно передавать много видеоканалов, как, например, сети кабельного телевидения. Резкий «прилив» информации, нужный для поддержания быстро меняющихся сцен в одном видеоканале, обеспечивается за счёт невысоких требований сравнительно медленно меняющихся сцен в других каналах.

## ЦИФРОВЫЕ СИГНАЛЬНЫЕ ПРОЦЕССОРЫ

Цифровая обработка сигналов реализуется с помощью математических операций, а, например, текстовые редакторы и другие похожие программы выполняют только реорганизацию хранимых данных. Это означает, что компьютеры, предназначенные для бизнеса или других применений общего назначения, не могут быть оптимальны для реализации таких алгоритмов, как цифровая фильтрация и анализ Фурье. *Цифровые сигнальные процессоры* представляют собой класс микропроцессоров, специально разработанный для решения задач цифровой обработки сигналов. Эти устройства претерпели стремительное развитие в последние десятилетия и нашли применение повсюду, начиная с мобильного телефона и кончая суперсовременным научным оборудованием. Примечательно, что в английском языке аббревиатура DSP расшифровывается инженерами—разработчиками аппаратного обеспечения как *Digital Signal Processor — цифровой сигнальный процессор* (ЦСП), а учёными, имеющими дело с математикой и алгоритмами, — как *Digital Signal Processing* — цифровая обработка сигналов (ЦОС). В этой главе рассматривается, чем ЦСП отличаются от других типов микропроцессоров, как определить, что именно ЦСП необходим для решения конкретной задачи, и как начать работать с этими замечательными устройствами. Следующая глава будет посвящена более детальному рассмотрению одного из семейств ЦСП — семейству SHARC фирмы Analog Device.

### 28.1. Чем ЦСП отличаются от других микропроцессоров

В 1960-е годы делались прогнозы относительно того, что искусственный интеллект внесёт революционные перемены в способы взаимодействия человека и компьютера. В те годы полагали, что к концу XX в. роботы будут убирать наши дома, бортовые компьютеры — управлять машинами, а управлять записью и воспроизведением информации можно будет с помощью голоса. Этого, однако, не случилось: такие абстрактные задачи оказались намного сложнее, чем ожидалось, и их очень тяжело реализовать с использованием пошаговой логики, характерной для цифровых компьютеров.

В то же время последние сорок лет показали, что компьютеры хорошо приспособлены для использования в двух широких областях применения: 1) в работе с данными, такой как редактирование текста и управление базами данных и 2) математических вычислениях, используемых в науке, технике и цифровой обработке сигналов. Все микропроцессоры могут выполнять оба этих класса задач, однако очень трудно (дорого) сделать такое устройство, которое было бы одина-

ково хорошо приспособлено и к тем, и к другим задачам. Это происходит потому, что указанные классы задач по ряду направлений предъявляют к процессорам противоположные требования. К таким требованиям, например, относятся объём набора команд и организация обработки прерываний. Ещё более существенными оказываются условия рынка: стоимость разработки и изготовления, конкурентоспособность, время жизни изделия и т. д. Эти факторы сделали традиционные микропроцессоры, такие как Pentium, в основном ориентированными на работу с данными. В свою очередь ЦСП разрабатываются специально для выполнения математических вычислений, необходимых в цифровой обработке сигналов.

В Табл. 28.1 перечислены наиболее важные различия между двумя классами задач. Работа с данными включает хранение и классификацию информации. Рассмотрим, например, программу обработки слов (текстовый редактор). Основная задача здесь заключается в хранении информации (напечатанной оператором), организации информации (удалении и вставке, проверке правильности написанная, разметке страниц и т. д.) и сохранении информации (записи документа на носитель информации или вывод на печать). Эти задачи выполняются перемещением данных с одного места на другое и проверкой неравенств ( $A = B$ ,  $A < B$  и т. п.). В качестве примера, представим себе сортировку списка слов в алфавитном порядке. Каждое слово представлено 8-битным числом — значением ASCII-кода первой буквы. Расстановка по алфавиту состоит в изменении порядка слов до тех пор, пока соответствующие значения ASCII-кодов не будут непрерывно возрастать от начала до конца списка. Это может быть выполнено повторением двух шагов один за другим, пока не закончится расстановка по алфавиту. Первый шаг проверяет, в алфавитном ли порядке расположены соседние слова? (ЕСЛИ  $A > B$ , ТОГДА ...). Второй шаг выполняется, если слова оказались не в алфавитном порядке, и заключается он в том, что слова меняются местами. Этот процесс повторяется многократно для всех пар слов.

Таблица 28.1. Работа с данными и математические вычисления

	Манипуляция данными	Математические вычисления
Типичные приложения	Обработка текста, управление базой данных, большие таблицы, операционные системы и т. п.	Цифровая обработка сигналов, управление движением, научное и техническое моделирование и т. п.
Основные операции	Перемещение данных ( $A \rightarrow B$ ) Проверка значений (Если $A = B$ , то ...)	Сложение ( $A + B = C$ ) Умножение ( $A \times B = C$ )

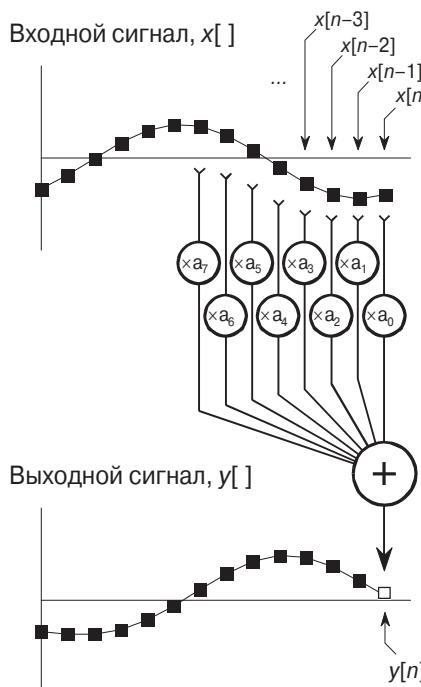
В качестве другого примера рассмотрим, как из текстового редактора печатается документ. Компьютер непрерывно проверяет устройство ввода (мышь или клавиатуру) на наличие бинарного кода, означающего команду «распечатать документ». Когда этот код обнаружен, программа пересыпает данные из памяти компьютера на принтер. Здесь мы имеем те же две базовые операции: перемещение данных и проверку неравенств. Хотя математические операции иногда используются в этом типе приложений, встречаются они не так часто и незначительно влияют на скорость решения задачи.

Для сравнения: скорость выполнения алгоритмов ЦОС почти полностью ограничивается требуемым числом операций умножения и сложения. В качестве примера, Рис. 28.1 иллюстрирует реализацию цифрового фильтра с *конечной импуль-*

*сной характеристикой (КИХ) — одну из наиболее часто встречающихся в ЦОС задач. Входной сигнал обозначен через  $x[n]$ , а выходной — через  $y[n]$ . Задача состоит в вычислении значения отсчёта выходного сигнала в момент времени  $n$ , т. е.  $y[n]$ . КИХ-фильтр выполняет это вычисление, умножая соответствующие отсчёты входного сигнала на группу коэффициентов, обозначенных  $a_0, a_1, a_2, \dots$  с последующим суммированием произведений. Можно рассчитать  $y[n]$  по формуле:*

$$y[n] = a_0x[n] + a_1x[n - 1] + a_2x[n - 2] + a_3x[n - 3] + \dots$$

Говорят, что входной сигнал сворачивается с ядром фильтра (т. е. с импульсной характеристикой), состоящим из коэффициентов  $a_0, a_1, a_2, \dots$ . В зависимости от приложения ядро фильтра может содержать как 5...10, так и несколько тысяч коэффициентов. Этот алгоритм вычисления включает ряд операций пересылки данных и оценки неравенств, что необходимо, в частности, для использования промежуточных результатов и управления циклами, однако математические операции здесь доминируют с точки зрения времени выполнения.



**Рис. 28.1.** Цифровой КИХ-фильтр. При реализации КИХ-фильтра отсчёты выходного сигнала  $y[n]$  рассчитываются перемножением отсчётов входного сигнала  $x[n], x[n - 1], x[n - 2], \dots$  на коэффициенты ядра фильтра  $a_0, a_1, a_2, \dots$  и последующим суммированием произведений.

Кроме очень быстрого выполнения математических вычислений, для ЦСП очень важным является также предсказуемость времени выполнения задачи. Предположим, что вы запустили ваш компьютер на решение некоторой задачи, скажем, преобразования текстового документа из одного формата в другой. Не имеет значения, займёт этот процесс 10 мс или 10 с, вы просто ждёте окончания выполнения задачи, прежде чем дать компьютеру следующее задание.

В отличие от компьютеров общего назначения ЦСП в большинстве случаев используются в таких приложениях, где обработка идёт непрерывно, не имея строго определённого момента начала и окончания. Например, возьмём систему на базе ЦСП, предназначенную для обработки аудиосигнала в слуховом аппарате. Если на вход ЦСП непрерывно поступает цифровой сигнал со скоростью 20 000 отсчётов в секунду, то процессор должен обеспечивать обработку этих 20 000 отсчётов не более чем за секунду. Однако имеются достаточно важные причины выполнять обработку не быстрее, чем это необходимо. Увеличение скорости вызывает увеличение стоимости, потребляемой мощности, сложности проектирования и т. д. Это делает точное знание времени выполнения задачи очень критичным для выбора соответствующих устройств, а также применяемых алгоритмов.

## 28.2. Циклическая буферизация

Цифровые сигнальные процессоры спроектированы специально для быстрого выполнения задач КИХ-фильтрации и подобных алгоритмов. Поэтому, чтобы понять архитектуру ЦСП, сначала нужно разобраться с алгоритмами. В данном разделе мы составим детальный список действий, необходимых для реализации КИХ-фильтра. После этого можно будет изучать, как проектируются ЦСП, так чтобы выполнять все эти действия максимально эффективно.

Для начала нам нужно понять различие между автономной обработкой (*off-line*) и обработкой в *реальном времени* (*real-time*). При автономной обработке входной сигнал целиком находится в компьютере. Например, геофизики могут регистрировать с помощью сейсмометра колебания земной поверхности во время землетрясения. После землетрясения информация может быть записана в компьютер и проанализирована каким-либо способом. Другой пример автономной обработки — медицинское обследование, например компьютерная томография. Регистрируется набор показателей в то время, пока пациент находится внутри аппарата, а построение изображения может быть сделано позднее. Ключевым моментом является то, что вся информация одновременно доступна для программы обработки. Такая ситуация является широко распространённой в научных исследованиях и технике, но не в устройствах широкого потребления. Автономная обработка — это сфера персональных компьютеров и больших ЭВМ.

При обработке в реальном времени выходной сигнал формируется в то же время, как в систему поступает входной сигнал. Например, это необходимо в телефонной связи, слуховых аппаратах и радарах. В этих приложениях информация должна быть доступна незамедлительно, хотя она может быть задержана на небольшое время. Например, 10-миллисекундную задержку в телефонной линии не различают ни говорящий, ни слушающий. Также не произойдёт ничего страшного, если радиолокационный сигнал задержится на несколько секунд перед тем, как отобразится на дисплее оператора. В приложениях реального времени принимается входной отсчёт, выполняется определённый алгоритм и формируется отсчёт на выходе, и так повторяется снова и снова. Возможен и другой вариант: принимается группа отсчётов, выполняется алгоритм и на выходе формируется также группа отсчётов. Цифровые сигнальные процессоры ориентированы именно на такие задачи.

Теперь вернёмся к Рис. 28.1 и рассмотрим, что собой представляет КИХ-фильтр, работающий в реальном времени. Чтобы вычислить выходной отсчёт, мы должны иметь доступ к определённому числу самых последних (свежих) входных отсчётов. Например, предположим, что мы используем фильтр с восемью коэффициентами  $a_0, a_1, a_2 \dots a_7$ . Это значит, что мы должны знать значения восьми самых последних отсчётов сигнала  $x[n], x[n - 1], \dots, x[n - 7]$ . Эти восемь отсчётов должны быть сохранены в памяти и непрерывно изменяться при поступлении новых отсчётов. Как управлять этим процессом наилучшим образом? Ответ: с помощью циклической буферизации.

Рис. 28.2 иллюстрирует работу циклического буфера на восемь отсчётов. Этот буфер занимает в памяти 8 последовательно расположенных ячеек 20041...20048. На (а) показаны восемь входных отсчётов, находящихся в буфере в некоторый момент времени. На (б) иллюстрируется изменение буфера после поступления следующего отсчёта. Идея циклической буферизации состоит в том, что конец линейного массива связан с его началом, т. е. ячейка памяти 20041 рассматривается как следующая за ячейкой 20048, так же как за 20044 следует 20045. Вы «перемещаетесь» по массиву с помощью *указателя* (переменной, значением которой является адрес), который указывает, где размещён последний (самый новый) входной отсчёт. Например, на (а) указатель содержит адрес 20044, а на (б) — 20045. Когда поступает новый отсчёт, он помещается на место самого старого отсчёта в массиве, и указатель передвигается на один адрес вперёд. Циклический буфер эффективен, потому что необходимо изменить только одно значение при поступлении нового отсчёта.

Чтобы организовать циклический буфер, нужны четыре параметра. Во-первых, должен быть указатель на начало циклического буфера, т. е. его адрес в памяти (в нашем примере это ячейка с номером 20041). Во-вторых, должен быть указатель на конец буфера (20048) или переменная, содержащая его длину (8). В-третьих, должен быть определен шаг, с которым происходит перемещение



Рис. 28.1. Цифровой КИХ-фильтр. При реализации КИХ-фильтра отсчёты выходного сигнала  $y[n]$  рассчитываются перемножением отсчётов входного сигнала  $x[n], x[n - 1], x[n - 2], \dots$  на коэффициенты ядра фильтра  $a_0, a_1, a_2, \dots$  и последующим суммированием произведений.

внутри буфера. В примере на Рис. 28.2 размер шага равен 1, т. е., например, если по адресу 20043 расположен один отсчёт, то следующий отсчёт будет размещаться по адресу 20044 и т. д. На практике такая ситуация встречается нечасто. Обычно адресация ведётся по байтам, а каждый отсчёт занимает в памяти два или четыре байта. В этом случае размер шага будет равен соответственно двум или четырём.

Перечисленные три параметра определяют размер и конфигурацию циклического буфера и не изменяются в течение работы программы. Четвёртый параметр — указатель на самый последний (самый «свежий») отсчёт — должен постоянно модифицироваться в процессе поступления новых данных. Поэтому должна быть предусмотрена определённая программная логика, управляющая тем, как этот четвёртый параметр изменяет своё значение на основе первых трёх параметров. Логика эта достаточно проста, поэтому должна реализовываться очень быстро. ЦСП необходимо строить так, чтобы оптимизировать управление циклическими буферами, обеспечивая максимально высокую скорость адресации.

Отметим, что циклическая буферизация может быть полезной и при обработке не в реальном времени. Рассмотрим, например, программу вычисления свёртки, входной и выходной сигналы которой целиком размещаются в памяти. Циклическая буферизация не является здесь необходимой, так как любой отсчёт может быть непосредственно доступен. Однако обычно алгоритмы обработки сигналов предполагают поэтапное выполнение с формированием промежуточных сигналов при переходе от одного этапа к другому. Таким образом, например, реализуется рекурсивный фильтр, который представляет собой последовательность биквадратных фильтров. При решении задачи «в лоб» каждый промежуточный сигнал также целиком записывается в память. Однако циклическая буферизация способна предложить более эффективный вариант: хранить только те промежуточные отсёты, которые необходимы для текущих вычислений. Это уменьшает требуемый объём памяти за счёт усложнения алгоритма. Важно помнить, что при автономной обработке циклическая буферизация может быть полезна, а в приложениях реального времени она просто необходима.

Теперь мы можем перечислить шаги, необходимые для реализации КИХ-фильтра, использующего циклические буфера, и для хранения входного сигнала, и для коэффициентов фильтра. Список этих шагов может показаться вам тривиальным, не представляющим на первый взгляд интереса, но это не совсем так. Эффективная реализация всех перечисляемых этапов — это то, что отличает ЦСП от обычных микропроцессоров. Итак, для каждого нового входного отсчёта необходимо выполнить действия, перечисленные в Табл. 28.2.

Необходимо сделать так, чтобы все эти шаги выполнялись как можно быстрее. Так как пункты 6...12 повторяются многократно (один раз для каждого коэффициента фильтра), особое внимание необходимо уделить этим операциям. Обычный микропроцессор выполняет приведённые 14 шагов последовательно один за другим. ЦСП же проектируется так, чтобы выполнять их параллельно. В некоторых случаях все операции внутри цикла (шаги 6...12) могут даже быть выполнены за один *такт*. Рассмотрим внутреннюю архитектуру ЦСП, позволяющую достигать такой замечательной производительности.

Таблица 28.2. Шаги реализации КИХ-фильтра

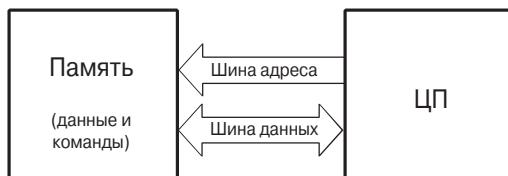
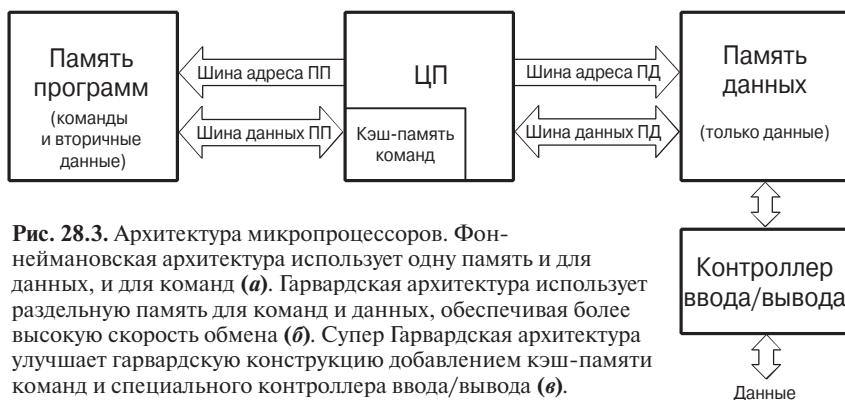
№ п/п	Шаг реализации КИХ-фильтра
1	Получить отсчёт с АЦП, сгенерировать прерывание
2	Обнаружить прерывание и отреагировать на него
3	Записать поступивший отсчёт в циклический буфер входного сигнала
4	Изменить указатель циклического буфера входного сигнала
5	Обнулить аккумулятор
6	Организовать цикл по всем коэффициентам фильтра
7	Взять коэффициент из циклического буфера коэффициентов фильтра
8	Изменить указатель циклического буфера коэффициентов
9	Взять отсчёт из циклического буфера входного сигнала
10	Изменить указатель циклического буфера входного сигнала
11	Умножить коэффициент фильтра на отсчёт сигнала
12	Добавить результат умножения к аккумулятору
13	Передать выходной отсчёт (аккумулятор) в буфер хранения
14	Передать выходной отсёт из буфера хранения в ЦАП

## 28.3. Архитектура цифрового сигнального процессора

Одним из наиболее критичных для скорости выполнения алгоритмов ЦОС факторов является передача информации в память и из памяти. Сюда входят не только данные, такие как выборки входного сигнала и коэффициенты фильтра, но и команды программы, т. е. двоичные коды, поступающие в *программный автомат*. Предположим, например, что нам нужно умножить два числа, которые находятся где-то в памяти. Чтобы сделать это, мы должны извлечь три двоичных значения из памяти: числа, которые нужно умножить, плюс команду программы, указывающую, что нужно сделать.

**Рис. 28.3а** иллюстрирует, как эта простая на вид задача выполняется в традиционном микропроцессоре. Используемая в данном случае архитектура обычно называется *фон Неймановской*, по имени выдающегося американского математика Джона фон Неймана (1903—1957). Фон Нейман положил начало многим открытиям, сделанным в науке и технике в начале XX в. Его достижения включают разработку концепции программируемого компьютера, формализацию математического аппарата квантовой механики и работы над созданием атомной бомбы. Если было что-то новое и интересное в те времена, фон Нейман в этом обязательно участвовал!

Как показано на **(а)**, архитектура фон Неймана содержит единственный модуль памяти и единственную шину для обмена данными с центральным процессором (ЦП). Умножение двух чисел в этом случае требует по крайней мере трёх тактов: по одному на пересылку каждого из двух чисел и кода команды по шине данных из памяти в ЦП. Мы не учитываем здесь время на пересылку результата обратно в память, потому что предполагаем, что он остаётся в ЦП для последую-

**а) Архитектура фон Неймана (один модуль памяти)****б) Гарвардская архитектура (два модуля памяти)****в) Супергарвардская архитектура**  
(два блока памяти, кэш-память команд, контроллер ввода/вывода)

**Рис. 28.3. Архитектура микропроцессоров.** Фон-неймановская архитектура использует одну память и для данных, и для команд (**а**). Гарвардская архитектура использует раздельную память для команд и данных, обеспечивая более высокую скорость обмена (**б**). Супер Гарвардская архитектура улучшает гарвардскую конструкцию добавлением кэш-памяти команд и специального контроллера ввода/вывода (**в**).

щих преобразований (таких как суммирование произведений в КИХ-фильтре). Фоннеймановская архитектура вполне подходит в том случае, когда речь идёт о последовательном выполнении всех команд. Большинство современных компьютеров построено фактически по архитектуре фон Неймана. Иная архитектура требуется только тогда, когда необходима очень быстрая обработка, и мы готовы заплатить за неё увеличением сложности устройства.

Таким требованиям отвечает *гарвардская архитектура*, проиллюстрированная на (**б**). Она называется так благодаря разработкам, сделанным в гарвардском университете в 1940-х годах под руководством Говарда Айкина (1900–1973). Айкин настаивал на разделении памяти на два блока: память данных и память программ с выделением отдельных шин для каждой из них. Так как шины работают независимо, то и команды, и данные могут выбираться из памяти и загружаться в ЦП

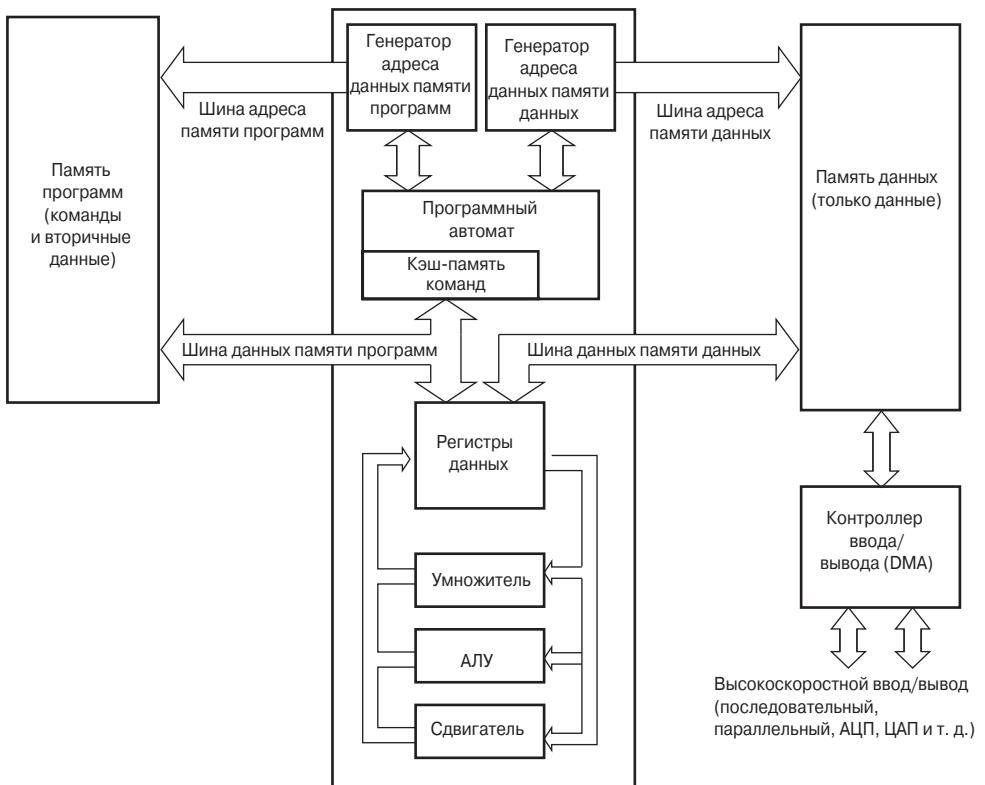
одновременно, повышая тем самым скорость обработки. Большинство современных ЦСП используют эту двухшинную архитектуру.

На (в) показан следующий уровень сложности организации обмена с памятью — *супергарвардская архитектура*. Этот термин был введён фирмой Analog Devices для описания работы цифровых сигнальных процессоров семейств ADSP-2106x и ADSP-211xx, выпускаемых компанией. ЦСП, построенные по такой архитектуре, назвали *SHARC*, что является сокращением от слов *Super Harvard ARChitecture*. Идея состояла в том, чтобы модернизировать гарвардскую архитектуру, наделив её дополнительными свойствами, улучшающими производительность. Архитектура ЦСП семейства SHARC оптимизирована по десяткам направлений, однако два из них играют особую роль — это наличие кэш-памяти команд и контроллера ввода/вывода (в).

В первую очередь рассмотрим, как наличие кэш-памяти команд усовершенствует гарвардскую архитектуру. Недостатком базовой гарвардской архитектуры является то, что шина памяти данных занята больше, чем шина памяти программ. При перемножении двух чисел два двоичных значения (числа) должны поступить в ЦП по шине данных, и только одно двоичное значение (команда) загружается по шине программ. Чтобы улучшить ситуацию, мы начнём с того, что переместим часть данных (чисел) в память программ. Например, мы можем разместить в памяти программ коэффициенты фильтра, а отсчёты входного сигнала записывать по-прежнему в память данных (перемещённые данные на рисунке названы вторичными). На первый взгляд ситуация нисколько не улучшилась. Теперь мы должны передавать одно значение по шине данных (отсчёт входного сигнала) и два значения по шине команд (команду и коэффициент). Если бы мы выполняли случайную последовательность команд, то это действительно было бы так.

Однако известно, что алгоритмы ЦОС в общем случае основное время обработки тратят на выполнение циклов, таких как команды 6...12 в Табл. 28.2. Но цикл подразумевает, что из памяти программ в ЦП поступают одни и те же команды. Супергарвардская архитектура использует этот факт, дополняя ЦП *кэш-памятью программ*. Это память малого объёма, которая запоминает около 32 последних выполненных команд. При первой итерации цикла команды загружаются в ЦП по шине программ. Это приводит к замедлению работы из-за конфликта с коэффициентами фильтра, которые тоже должны пересыпаться по этой шине. Однако при дальнейшем выполнении цикла команды могут выбираться из кэш-памяти. А это означает, что вся информация может загружаться из памяти в ЦП за один такт: отсчёт входного сигнала приходит по шине данных, коэффициент фильтра поступает по шине программ, а команда выбирается из кэш-памяти. На языке ЦОС о такой эффективной передаче данных говорят «высокая пропускная способность интерфейса с памятью».

На Рис. 28.4 архитектура процессоров SHARC показана более детально. Архитектура таких процессоров содержит *контроллер ввода/вывода*, соединённый с памятью данных и организующий ввод сигналов в систему и вывод из неё. В частности, для обмена данными с внешними устройствами ЦСП семейства SHARC содержат и параллельный, и последовательный порты, обеспечивающие очень высокую скорость передачи. Например, если тактовая частота процессора 40 МГц, то два последовательных порта обеспечивают скорость передачи 40 Мбит/с каждый, а шесть параллельных портов — 40 МБ/с каждый. Когда все



**Рис. 28.4.** Типовая архитектура ЦСП. Цифровые сигнальные процессоры спроектированы так, чтобы выполнять различные задачи параллельно (одновременно). На рисунке приведена упрощённая схема одного из процессоров семейства SHARC фирмы Analog Devices. Сопоставьте эту архитектуру с теми задачами, которые необходимы для реализации КИХ-фильтра (см. **Табл. 28.2**). Все действия, выполняемые внутри цикла, показанного в **Табл. 28.2**, могут быть выполнены таким процессором за один такт.

шесть параллельных портов используются вместе, скорость передачи данных становится невероятной — 240 МБ/с.

Этого достаточно, чтобы передать весь текст этой книги всего за 2 миллисекунды! Очень важным является то, что специальные аппаратные средства позволяют передавать эти потоки данных непосредственно в память (*прямой доступ в память* — ПДП, или англоязычная аббревиатура DMA) без задействования регистров ЦП. Другими словами, шаги 1-й и 14-й в нашем списке (**Табл. 28.2**) выполняются независимо и одновременно с остальными шагами, не требуя от ЦП никаких временных затрат. Главные шины (шина программ и шина данных) являются доступными для внешних устройств, обеспечивая дополнительный интерфейс с внешней памятью и периферией. Это позволяет ЦСП семейства SHARC использовать до 4 Гслов (16 ГБ) памяти со скоростью доступа 40 Мслов/с (160 МБ/с) при передаче 32-битных данных. Это очень много!<sup>1)</sup>

<sup>1)</sup> Современные процессоры семейства SHARC имеют тактовые частоты до 400 МГц.

Способность поддерживать высокоскоростной интерфейс ввода/вывода является ключевой особенностью ЦСП. Ведь главная задача ЦСП — загрузить входные данные, выполнить необходимые математические преобразования и вывести результат, прежде чем на входе системы появится следующий отсчёт. Всё остальное вторично. Некоторые ЦСП имеют на кристалле встроенные аналого-цифровой или цифро-аналоговый преобразователи. Такие процессоры называются *смешанными (mixed-signal)*, т. е. совмещающими сигналы разных типов — аналоговые и цифровые. Однако все ЦСП имеют возможность взаимодействовать с внешними АЦП и ЦАП через последовательные или параллельные порты.

Теперь заглянем внутрь ЦП. В верхней части схемы (Рис. 28.4) мы видим два блока, называемых *генераторами адреса данных*, по одному на каждый модуль памяти. Эти блоки управляют адресами, направляемыми в память программ и память данных и указывающими, из какой ячейки памяти должна быть прочитана информация или в какую ячейку её необходимо записать. В более простых микропроцессорах решение этой задачи возлагается на программный автомат и является совершенно очевидным для программиста. Однако ЦСП спроектированы для работы с циклическими буферами и используют дополнительные схемные решения, позволяющие делать это эффективно. Такие устройства освобождают ЦП от необходимости тратить драгоценные такты на то, чтобы отслеживать, куда записывать или откуда считывать данные. Каждый из двух генераторов адреса процессора SHARC может управлять восемью циклическими буферами. Это означает, что каждый генератор адреса должен хранить 32 значения (по четыре на каждый буфер) и обеспечивать необходимую логику работы.

Зачем нужно так много циклических буферов? Ряд алгоритмов цифровой обработки реализуется эффективнее, если они разбиваются на несколько этапов. Например, БИХ-фильтры оказываются более устойчивыми, если реализуются в виде последовательного соединения нескольких биквадратных фильтров (т. е. фильтров, имеющих два полюса и не более двух нулей). Наличие нескольких этапов обработки требует наличия такого же числа циклических буферов, чтобы обеспечить возможность максимально быстрого выполнения алгоритма. Кроме возможности поддерживать большое число циклических буферов, генераторы адреса данных процессоров SHARC оптимизированы также для эффективного выполнения алгоритма *быстрого преобразования Фурье (БПФ)*. Они позволяют адресовать циклические буфера в режиме *бит-реверсной адресации*, которая необходима для реализации БПФ. Отметим, что большое число циклических буферов существенно упрощает процесс генерации программного кода как в случае его разработки вручную (программистом), так и при автоматическом создании кода средствами компилятора языков программирования высокого уровня, например языка Си.

Набор регистров данных ЦП используется так же, как и в обычных микропроцессорах. ЦСП семейства ADSP-2106x SHARC включают 16 40-битных регистров общего назначения. Они предназначены для хранения промежуточных результатов вычислений, загрузки входных операндов математических процедур, а также служат как буфера при пересылке данных, выступают в качестве флагов, управляющих программой, и т. д. При необходимости эти регистры могут также использоваться для управления циклами и выступать в качестве счётчиков, однако в

ЦСП SHARC предусмотрен специальный набор регистров, предназначенных для подобных задач.

Математическая обработка в процессоре осуществляется в трёх вычислительных блоках: *умножителе, арифметико-логическом устройстве (АЛУ) и параллельном сдвигателе*. Умножитель загружает из регистров два операнда, находит их произведение и результат записывает в регистр. АЛУ служит для выполнения операций сложения, вычитания, вычисления модуля, логических операций (И, ИЛИ, ИСКЛЮЧАЮЩЕЕ ИЛИ, НЕ), преобразования чисел в формате с фиксированной точкой в формат с плавающей точкой или наоборот и других функций. Элементарные двоичные операции выполняются параллельным сдвигателем. К ним относятся: сдвиг, циклический сдвиг, извлечение или запись фрагментов слова данных и др. Важным свойством процессоров SHARC, делающим их архитектуру действительно мощной, является возможность параллельной (одновременной и независимой) работы умножителя и АЛУ. За один такт умножителем могут быть обработаны операнды из регистров 0...7, а АЛУ — операнды из регистров 8...15, а два полученных результата записаны в любой из 16 регистров.

Развитием базовой архитектуры SHARC ADSP-2106x является семейство процессоров SHARC ADSP-2116x, а также платформа ADSP-TS101 TigerSHARC<sup>1)</sup>. Эти процессоры построены по так называемой SIMD-архитектуре (Single Instruction Multiple Data: одна команда — много данных). Принцип SIMD состоит в том, что одни и те же инструкции выполняются одновременно для разных наборов данных.

Там, где процессор семейства ADSP-2106x включал один элемент обработки, процессоры ADSP-2116x и ADSP-TS101 имеют два элемента, обозначаемые РЕ x и РЕ y в 2116x и просто X и Y в TS101. Элемент РЕ x функционирует обычным образом так же, как тот же элемент в процессоре 2106x, но в режиме SIMD одни и те же команды выполняются и элементом РЕ x (над данными в регистрах R0...R15), и элементом РЕ y (над данными в регистрах S0...S15). Дальнейшим усовершенствованием архитектуры является поддержка режима MIMD — Multiple Instructions Multiple Data (много команд — много данных). В этом режиме вычислительные блоки выполняют различные потоки команд над разными наборами данных в пределах одного такта. Этот режим характерен только для процессоров ADSP-TS101 TigerSHARC фирмы Analog Devices (SIMD-режим ими также поддерживается как частный случай MIMD-режима).

Другое высокопроизводительное семейство ЦСП фирмы Analog Devices — семейство Blackfin — спроектировано так, чтобы за один такт выполнять параллельно как можно больше математических операций. Сердцем архитектуры Blackfin является блок арифметических операций над данными (Data Arithmetic Unit), который включает два 16-битных умножителя-накопителя (MAC), два 40-битных АЛУ, четыре 8-битных видео-АЛУ и один 40-битный параллельный сдвигатель. С помощью такой архитектуры обработка 8-, 16- или 32-битных данных производится с максимальной эффективностью.

Сравнивая эти разные архитектуры, можно сделать определённые важные выводы. Известно, что большинство алгоритмов ЦОС требует для выполнения

<sup>1)</sup> В 2006 г. семейство процессоров SHARC фирмы Analog Devices насчитывало уже 3 поколения, включая процессоры ADSP-212xx и ADSP-213xx, а семейство TigerSHARC включало процессоры ADSP-TS20x.

4...400 команд, что для процессоров SHARC эквивалентно 4...400 тактам выполнения обработки одного отсчёта входного сигнала. Тогда для второго поколения ЦСП SHARC ADSP-2116x, работающего на частоте 100 МГц, будет характерна скорость обработки от 250 тысяч до 25 миллионов отсчётов в секунду (в зависимости от сложности алгоритма). Аналогично для ЦСП ADSP-TS101 на частоте 250 МГц данные будут обрабатываться со скоростью от 625 тысяч до 62.5 миллиона отсчётов в секунду (в зависимости от сложности алгоритма). Здесь мы не принимали во внимание возможности функционирования в режиме SIMD, позволяющем вдвое сократить время обработки, т. е. повысить в 2 раза скорость обработки входного потока данных.

## 28.4. Процессоры с фиксированной и плавающей точкой

Цифровые сигнальные процессоры можно разделить на две категории: *процессоры с фиксированной точкой* и *процессоры с плавающей точкой*. Различие между ними состоит в способе представления и обработки чисел внутри устройства. ЦСП с фиксированной точкой обычно представляют каждое число с помощью 16 бит, хотя могут использоваться и другие варианты длины слова. Например, фирма Motorola выпускает семейство ЦСП с фиксированной точкой, в котором используется 24-битное представление чисел. Существуют четыре общепринятых способа использования выделенных 16 бит, т. е.  $2^{16} = 65536$  возможных двоичных комбинаций для представления чисел. Тип данных *беззнаковое целое* (*unsigned integer*) позволяет числам принимать целые значения в диапазоне 0...65535. Тип *целое со знаком* (*signed integer*) использует дополнительный код и включает отрицательные числа, охватывая диапазон  $-32768...32767$ . Число в формате *дробное без знака* (*unsigned fraction*) может принимать одно из 65536 дробных значений, равномерно распределённых между 0 и 1. И наконец, тип данных *дробное со знаком* позволяет представлять числа, равномерно распределённые между  $-1$  и 1.

В отличие от процессоров с фиксированной точкой, ЦСП с плавающей точкой обычно используют для представления чисел 32 бита. Число возможных двоичных кодовых комбинаций при этом существенно возрастает:  $2^{32} = 4294967296$ . Ключевой особенностью формата с плавающей точкой является то, что представляемые этим форматом числа распределяются неравномерно. Так для наиболее распространённого формата ANSI/IEEE Std. 754-1985 самое большое по модулю число оказывается равным  $\pm 3.4 \times 10^{38}$ , а самое маленькое —  $\pm 1.2 \times 10^{-38}$ . При этом значения между этими двумя границами диапазона распределены неравномерно и формируются так, что любые два «соседних» числа отличаются друг от друга на величину, в десять миллионов раз меньшую значения самих чисел. Это очень важно, потому что в результате большие по модулю числа отличаются друг от друга существенно, а малые числа имеют гораздо меньшие «интервалы» между соседними значениями. Более детально формат чисел с плавающей точкой обсуждался в Главе 4.

Все ЦСП с плавающей точкой могут также оперировать числами в формате с фиксированной точкой, что необходимо для использования счётчиков, организации циклов, получения целочисленных данных с АЦП и передачи данных на ЦАП. Однако это не значит, что математические операции с фиксированной точ-

кой будут выполняться так же быстро, как и операции с плавающей точкой. Это зависит от внутренней архитектуры процессора. Например, ЦСП SHARC построены так, что они могут одинаково эффективно обрабатывать как данные в формате с плавающей точкой, так и данные в формате с фиксированной точкой. По этой причине процессоры SHARC часто называют «32-битными ЦСП», а не просто «ЦСП с плавающей точкой».

**Рис. 28.5** иллюстрирует проблему выбора между ЦСП с фиксированной и плавающей точкой. В Главе 3 мы подчёркивали, что в компьютерах общего назначения арифметика с фиксированной точкой реализуется намного быстрее вычислений с плавающей точкой. Однако для ЦСП характерно примерное равенство этих скоростей, так как архитектура процессоров в высокой степени оптимизирована для математических операций. При этом архитектура ЦСП с плавающей точкой оказывается более сложной, чем архитектура процессоров с фиксированной точкой. Все регистры и шины данных должны быть не 16-, а 32-битными, умножитель и АЛУ должны быть приспособлены к быстрому выполнению арифметики с плавающей точкой, набор команд должен быть шире (чтобы выполнять операции над числами как в формате с фиксированной, так и в формате с плавающей точкой) и т. д. Для формата с плавающей точкой (32 бита) характерна большая точность и более широкий динамический диапазон, чем для формата с фиксированной точкой (16 битов). Кроме того, программы для процессора с плавающей точкой обычно гораздо проще и быстрее создаются, так как программисту не нужно беспокоиться о таких проблемах, как переполнение, потеря значимости и ошибки округления.

С другой стороны, ЦСП с фиксированной точкой обычно имеют меньшую стоимость, чем ЦСП с плавающей точкой. Ничто не меняется так быстро, как цены на электронную продукцию; поэтому все цифры, которые вы встретите в этой книге, устареют раньше, чем она будет напечатана. Тем не менее, стоимость является ключевым фактором в понимании того, как эволюционируют ЦСП, и мы должны с вами понять общую идею. Когда эта книга была подготовлена к печати в 2002 г.<sup>1)</sup>, ЦСП с фиксированной точкой стоили 5...100\$, а ЦСП с плавающей точкой — 10...300\$<sup>2)</sup>. Эта разница в цене может рассматриваться как мера относительной сложности устройств.

Теперь обратим наше внимание на эффективность: что 32-битные процессоры с плавающей точкой могут делать такого, чего не могут 16-битные ЦСП с фиксированной точкой? Ответ на этот вопрос: *отношение сигнал/шум*. Предположим, ЦСП хранит некоторое 32-битное число в формате с плавающей точкой. Как было сказано ранее, интервал между этим числом и его «ближайшим соседом» составит одну десятимиллионную долю значения самого числа. Чтобы записать в память произвольное число, оно должно быть округлено в большую или меньшую сторону, изменяясь максимум наполовину указанного интервала. То есть

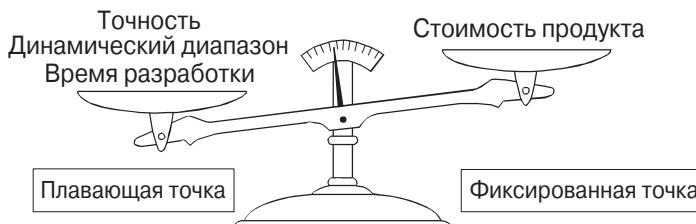
---

<sup>1)</sup> Имеется в виду оригинальное издание на английском языке. — Примеч. ред.

<sup>2)</sup> На сегодняшний день диапазон цен на процессоры с фиксированной точкой 2...300\$, на процессоры с плавающей точкой — 5...300\$. С одной стороны, технологии дешевеют, и плавающая точка стремится стать дешёвой. С другой стороны, процессоры с фиксированной точкой стали «утяжелять» дополнительными возможностями, поэтому некоторые их модели не уступают по цене процессорам с плавающей точкой. — Примеч. науч. ред.

каждый раз, когда мы записываем число в память в формате с плавающей точкой, мы добавляем к сигналу *шум*.

То же самое происходит и тогда, когда число записывается как 16-битное в формате с фиксированной точкой, однако уровень шума при этом намного больше. Шум возрастает потому, что интервалы между соседними числами в этом случае гораздо больше. Предположим, мы хотим представить число 10000 как целое со знаком (диапазон  $-32768\dots32767$ ). Интервал между этим и соседним числом составит одну десятитысячную долю от значения запоминаемого числа. Если мы захотим в этом же формате записать в память число 1000, интервал между числами составит уже всего одну тысячную долю.



**Рис. 28.5.** Фиксированная и плавающая точка: что лучше? ЦСП с фиксированной точкой обычно дешевле, а процессоры с плавающей точкой отличаются более высокой точностью, широким динамическим диапазоном и меньшим временем разработки программ.

Обычно уровень шума в сигнале характеризуется *среднеквадратическим отклонением (СКО)*. Об этом шла речь в Главе 2. Сейчас нам важно то, что СКО шума *квантования* равно примерно одной третьей величины интервала между соседними представляемыми числами. Это значит, что отношение сигнал/шум при хранении данных в формате с плавающей точкой составляет около тридцати миллионов к одному, а при записи в формате с фиксированной точкой — всего около десяти тысяч к одному. Таким образом, для формата с плавающей точкой характерен приблизительно в 3000 раз меньший шум квантования, чем для формата с фиксированной точкой.

В этом проявляется одно из существенных отличий ЦСП от обычных микропроцессоров. Предположим, мы реализуем КИХ-фильтр на процессоре с фиксированной точкой. Мы организуем цикл, в котором «пробегаем» последовательно по всем коэффициентам фильтра, умножая их на соответствующие отсчеты входного сигнала и накапливая результат в аккумуляторе. Здесь, однако, возникает проблема. В микропроцессорах общего назначения этот аккумулятор представляет собой обычную 16-битную переменную в формате с фиксированной точкой, такую же, как входные операнды. Чтобы избежать переполнения, необходимо осуществлять нормировку суммируемых чисел и добавлять таким образом шум квантования на каждой итерации цикла. В худшем случае этот шум квантования будет постоянно нарастать, значительно уменьшая отношение сигнал/шум. Например, при реализации КИХ-фильтра с 500 коэффициентами уровень шума в каждом выходном отсчете может оказаться в 500 раз больше уровня шума каждого входного отсчета. Отношение сигнал/шум десять тысяч к одному падает до ужасной величины — двадцать к одному. Несмотря на то что это предельный случай, он хорошо иллюстрирует важный факт: когда каждый отсчет сигнала участвует в

большом количестве вычислений, результат оказывается очень плохим. Более детально эта проблема рассмотрена в Главе 3.

В ЦСП проблема переполнения при накоплении решена с помощью использования *аккумулятора повышенной точности*. Для целей накопления выделяется специальный регистр, число битов которого в 2...3 раза больше по сравнению с другими элементами памяти. Например, в 16-битных ЦСП обычно используется 32- или 40-битный аккумулятор, а в процессорах SHARC для вычислений в формате с фиксированной точкой предусмотрен 80-битный аккумулятор. Такая расширенная разрядность аккумулятора позволяет практически полностью исключить ошибки округления в процессе накопления. Ошибка округления проявляется лишь тогда, когда значение аккумулятора записывается в 16-битную память и происходит соответствующая нормировка. Описанный приём работает очень хорошо, но он накладывает определённые ограничения на то, как должны выполняться алгоритмы. В отличие от процессоров с фиксированной точкой процессоры с плавающей точкой имеют настолько малые шумы квантования, что для них обычно нет необходимости в применении таких методов.

Кроме преимуществ малых шумов квантования, процессоры с плавающей точкой отличаются также упрощением процесса разработки программ. Большинство методов ЦОС основано на повторяющихся процедурах умножения и сложения. Программа, реализующая эти методы на процессоре с фиксированной точкой, должна после каждой операции проверять, не произошло ли переполнение или потеря значимости. Программисту нужно постоянно следить за амплитудой чисел, процессом накопления ошибки квантования и тем, как следует производить нормировку, чтобы избежать переполнений и потери значимости. Все эти проблемы исчезают при использовании процессоров с плавающей точкой. Числа в формате с плавающей точкой «могут сами о себе позаботиться» (бывают, правда, и редкие исключения).

Чтобы лучше разобраться с этим вопросом, взгляните на Рис. 28.6, на котором приведена таблица, взятая из руководства пользователя по процессору SHARC. Она описывает способы реализации операции умножения для форматов с фиксированной и с плавающей точкой. Сначала посмотрим, как умножаются числа в формате с плавающей точкой. Здесь есть только один способ:  $F_n = F_x \times F_y$ , где  $F_n$ ,  $F_x$  и  $F_y$  — это любые из имеющихся 16 регистров. Проще быть не может. А теперь посмотрите, как много команд может использоваться для реализации этой же операции в формате с фиксированной точкой. Различных вариантов так много, потому что существует большой набор разных условий, которые необходимо учитывать, чтобы написать действительно эффективный программный код.

На Рис. 28.6  $R_n$ ,  $R_x$  и  $R_y$  — это любые из 16 регистров данных, а MRF и MRB — 80-битные аккумуляторы. Вертикальными линиями отмечены альтернативные варианты. Например, запись в верхней левой позиции таблицы означает, что все нижеперечисленные команды являются возможными:  $R_n = R_x * R_y$ ,  $MRF = R_x * R_y$  и  $MRB = R_x * R_y$ , т. е. значения любых двух регистров могут быть перемножены, а результат помещён в третий регистр или в один из аккумуляторов повышенной точности. Эта таблица также показывает, что числа могут быть знаковыми или беззнаковыми (S или U), дробными или целыми (F или I). Опции RND и SAT используются для управления режимами округления и переполнения регистров.

Фиксированная точка		Плавающая точка
$R_n = Rx * Ry$	$(\begin{array}{ c c c } \hline S & S & F \\ \hline U & U & I \\ \hline \end{array})$	$F_n = Fx * Fy$
$\begin{array}{l} MRF \\ MRB \end{array}$		
$R_n = MRF$	$+ Rx * Ry$	$(\begin{array}{ c c c } \hline S & S & F \\ \hline U & U & I \\ \hline \end{array})$
$R_n = MRB$		$(\begin{array}{ c c c } \hline S & S & F \\ \hline U & U & I \\ \hline FR & & \end{array})$
$MRF = MRF$		
$MRB = MRB$		
$R_n = MRF$	$- Rx * Ry$	$(\begin{array}{ c c c } \hline S & S & F \\ \hline U & U & I \\ \hline FR & & \end{array})$
$R_n = MRB$		
$MRF = MRF$		
$MRB = MRB$		
$R_n = SAT MRF$	$(SI)$	
$R_n = SAT MRB$	$(UI)$	
$MRF = SAT MRF$	$(SF)$	
$MRB = SAT MRB$	$(UF)$	
$R_n = RND MRF$	$(SF)$	
$R_n = RND MRB$		
$MRF = RND MRF$		
$MRB = RND MRB$		
$MRF = 0$		
$MRB = 0$		
$MRF = Rn$		
$MRB = Rn$		
$R_n = \begin{array}{ c } \hline MRF \\ \hline MRB \\ \hline \end{array}$		

**Рис. 28.6.** Сравнение команд, используемых при использовании формата с фиксированной и форматом с плавающей точкой. Рассмотрен пример команды умножения, выполняемый на процессоре SHARC. В то время как при работе с плавающей точкой достаточно всего одной команды, при использовании фиксированной точки приходится учитывать большое количество дополнительных условий.  $R_n$ ,  $R_x$  и  $R_y$  — это любые из 16 регистров данных, а MRF и MRB — 80-битные аккумуляторы. Вертикальными линиями отмечены альтернативные варианты.

В таблице перечислено много других деталей и альтернативных вариантов, не представляющих в данный момент для нас интереса. Основная идея, которую вы должны вынести из этих рассуждений, состоит в том, что программист, работающий с числами в формате с фиксированной точкой, должен проанализировать десятки разных способов выполнения такой базовой вычислительной процедуры, как умножение. В противоположность этому программист, работающий с числами в формате с плавающей точкой, может экономить своё время, концентрируясь на самом алгоритме.

Ознакомившись со всеми этими достоинствами и недостатками процессоров с фиксированной и плавающей точкой, какому из них вы отдастите предпочтение? Чтобы правильно выбрать подходящий класс ЦСП, следует рассмотреть несколько моментов. Во-первых, посмотрите, сколько битов используется в АЦП и ЦАП. Для многих приложений 12...14 бит — это граница, отделяющая фиксиру-

ванную точку от плавающей. Например, телевизионные или иные видеосигналы обычно обрабатываются 8-битными АЦП и ЦАП, и точности процессоров с фиксированной точкой оказывается вполне достаточно. С другой стороны, в профессиональных цифровых аудиосистемах звуковые сигналы представлены с точностью от 20 до 24 бит, что практически безальтернативно заставляет использовать для этих приложений процессоры с плавающей точкой, позволяющие охватить требуемый динамический диапазон.

Следующим фактором, на который следует обратить внимание, является сложность реализуемого алгоритма. Если алгоритм прост — лучше использовать фиксированную точку; если относительно сложен — отдайте предпочтение плавающей. Например, КИХ-фильтрация и другие методы обработки во временной области реализуются всего несколькими десятками строк программного кода и хорошо подходят для применения фиксированной точки. В отличие от них алгоритмы обработки сигналов в частотной области, такие как спектральный анализ и БПФ-свёртка, имеют массу деталей, и их программная реализация оказывается достаточно сложна. Хотя они и могут быть выполнены в формате с фиксированной точкой, время разработки можно существенно сократить, если использовать плавающую точку.

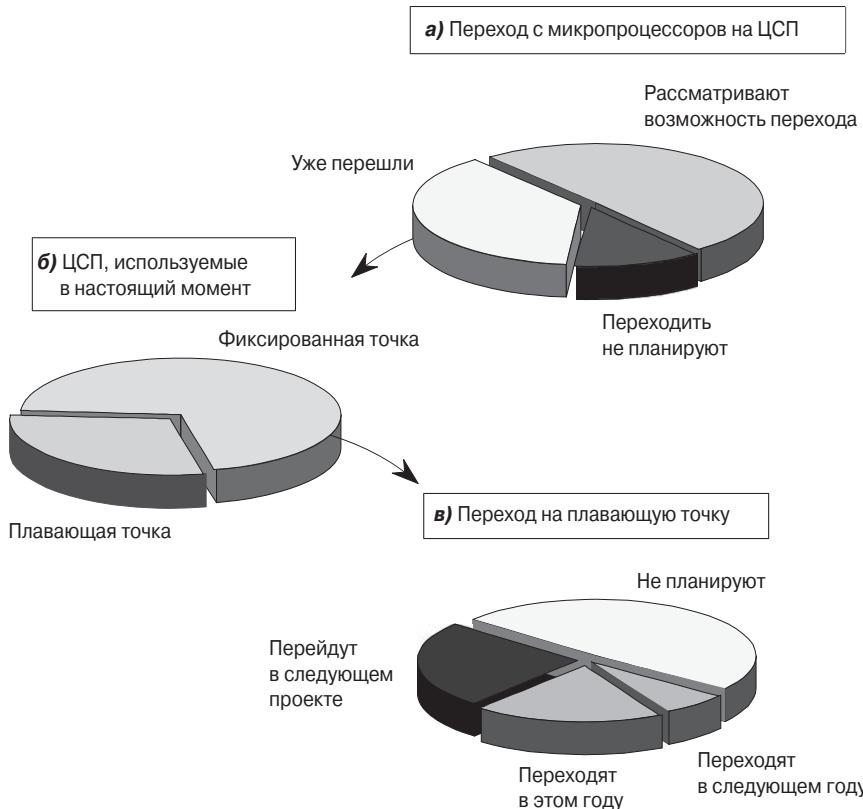
Наконец, подумайте о деньгах: насколько важна стоимость конечного продукта и какую роль играет для вас стоимость разработки? Фиксированная точка обычно позволяет снизить стоимость изделия, но цена разработки при этом существенно возрастает из-за необходимости создавать более сложные программы. При использовании плавающей точки, наоборот, разработка проходит проще, быстрее и дешевле, но стоимость конечного продукта увеличивается.

**Рис. 28.7** иллюстрирует некоторые основные тенденции в развитии ЦСП. На **(а)** показано, какое место занимают сейчас ЦСП на рынке *встраиваемых систем*. Так называются приложения, в которых микропроцессоры используются для непосредственного контроля и управления некоторой более сложной системой. Это, например, сотовые телефоны, микроволновые печи, бортовые компьютеры автомобилей и т. д. Вычислительные средства в таких системах часто называют *микроконтроллерами*, чтобы отличать их от *микропроцессоров*, используемых в персональных компьютерах. Как показано на **(а)**, около 38% разработчиков встраиваемых систем уже начали использовать ЦСП, а ещё 49% рассматривают возможность перехода на сигнальные процессоры<sup>1)</sup>. За счёт высокой пропускной способности и огромной вычислительной мощности ЦСП являются идеальным выбором для встраиваемых систем.

Как показано на **(б)**, ЦСП с фиксированной точкой используют почти в 2 раза большее число разработчиков по сравнению с ЦСП с плавающей точкой. Однако выбор во многом зависит от приложения. Фиксированная точка более популярна в продуктах широкого потребления с характерной для этого сегмента рынка высокой конкуренцией, где стоимость электроники должна поддерживаться максимально низкой. Хорошим примером здесь могут служить сотовые телефоны. Если вы являетесь производителем миллионов устройств и хотите победить в борьбе с конкурентами, разница в цене всего в несколько долларов может оказаться гранью между успехом и провалом. Плавающая точка, в отличие от фиксированной,

---

<sup>1)</sup> Эти данные автор приводит на момент подготовки книги (книга издана в 2003 г.). — Примеч. ред.



**Рис. 28.7.** Основные тенденции развития ЦСП. Как показано на (а), около 38% разработчиков встраиваемых систем уже перешли от использования обычных микропроцессоров к ЦСП, а ещё 49% рассматривают такую возможность. На (б) проиллюстрировано, что ЦСП с фиксированной точкой использует почти в 2 раза большее число разработчиков по сравнению с ЦСП с плавающей точкой. Это обусловлено главным образом наличием приложений, рассчитанных на широкий круг потребителей, таких как, например, мобильные телефоны. Здесь стоимость продукта имеет существенное значение. В то же время, как показано на (в), популярность плавающей точки возрастает очень быстро: более половины разработчиков, использующих настоящее время 16-битные устройства с фиксированной точкой, планируют перейти на ЦСП с плавающей точкой.

используется преимущественно тогда, когда наиболее критичной является вычислительная производительность, а стоимость продукта оказывается второстепенной. Предположим, вы проектируете медицинскую диагностическую систему, такую как компьютерный томографический сканер. Продано может быть всего несколько сот моделей по цене несколько сот тысяч долларов каждая. Для этого приложения стоимость ЦСП несущественна; вычислительная эффективность играет первостепенную роль. Несмотря на преобладание использования ЦСП с фиксированной точкой, рынок процессоров с плавающей точкой является наиболее быстро растущим. Как показано на (в), свыше половины инженеров, использующих 16-битные устройства с фиксированной точкой, планируют перейти на использование плавающей точки в ближайшем будущем.

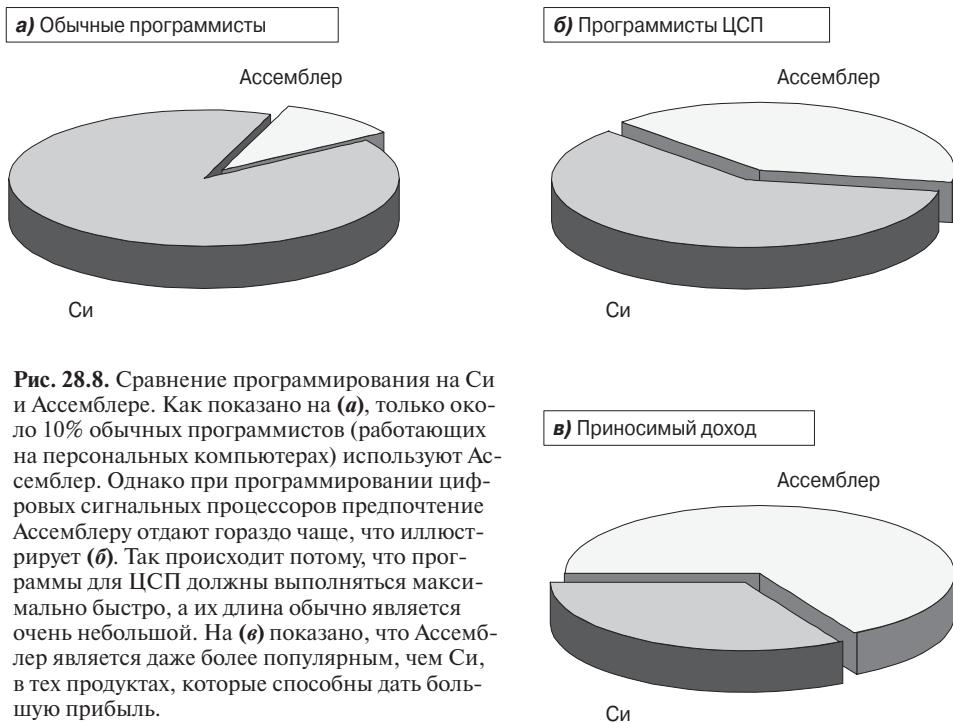
Прежде чем завершить эту тему, хотелось бы отметить, что процессоры с плавающей и с фиксированной точкой обычно являются соответственно 32- и 16-битными, но справедливо это не всегда. Например, процессоры семейства SHARC могут представлять числа в 32-битном формате с фиксированной точкой — типовом формате для цифровых систем обработки аудиосигналов. Числа в этом случае представляются  $2^{32}$  уровнями квантования, равномерно распределёнными на относительно малом интервале, скажем, от  $-1$  до  $1$ . Для сравнения: представление чисел в формате с плавающей точкой распределяет  $2^{32}$  уровней квантования логарифмически на огромном интервале  $\pm 3.4 \times 10^{38}$ . В результате 32-битная фиксированная точка оказывается более точной, т. е. ошибка квантования в любом одном отсчёте будет меньше. Однако плавающая точка имеет более широкий динамический диапазон, т. е. большую разницу между самым большим и самым маленьким числами, которые могут быть представлены в этом формате.

## 28.5. Си или Ассемблер?

ЦСП программируются на традиционных языках научных и инженерных приложений — *Ассемблере* и *Си*. Программы, написанные на Ассемблере, исполняются быстрее, а программы на Си легче разрабатывать и обслуживать. В традиционных приложениях, таких как программы, исполняемые на персональных компьютерах, Си почти всегда оказывается единственным выбором. Если Ассемблер и используется, то только в небольших фрагментах, где необходимо обеспечить максимально возможную скорость. Это отражено на Рис. 28.8а: на одного программиста, работающего с Ассемблером в приложениях общего назначения, приходится десять программистов, использующих Си.

Однако программы ЦСП отличаются от традиционного программного обеспечения двумя важными особенностями. Во-первых, эти программы обычно намного короче: скажем, сто против десяти тысяч строк кода. Во-вторых, скорость выполнения часто играет решающую роль. Привлекательность ЦСП состоит в первую очередь в их невероятной скорости обработки. Эти два фактора объясняют, почему многие программисты сигнальных процессоров предпочитают переходить с Си на Ассемблер. Взгляните на (б): почти одинаковое число программистов использует Ассемблер и Си.

Эффективность применения процессоров с точки зрения полученной прибыли иллюстрирует (в). На каждый доллар, полученный при использовании ЦСП, запрограммированного на языке Си, приходятся два доллара, «заработанные» ЦСП, запрограммированного на Ассемблере. Причина этого проста: доход возрастает тогда, когда удается превзойти конкурентов. С точки зрения вычислительной производительности и стоимости изготовления, Ассемблер всегда будет иметь преимущество перед Си. Программы на Си, в частности, обычно требуют большего объёма памяти, по сравнению с Ассемблером, что приводит к необходимости использовать более дорогие аппаратные решения. Однако рынок ЦСП непрерывно меняется. Он стремительно расширяется, и производители отвечают на это разработкой таких ЦСП, архитектура которых оптимизирована для программирования на Си. Например, программные коды Си оказываются гораздо



**Рис. 28.8.** Сравнение программирования на Си и Ассемблере. Как показано на (а), только около 10% обычных программистов (работающих на персональных компьютерах) используют Ассемблер. Однако при программировании цифровых сигнальных процессоров предпочтение Ассемблеру отдают гораздо чаще, что иллюстрирует (б). Так происходит потому, что программы для ЦСП должны выполняться максимально быстро, а их длина обычно является очень небольшой. На (в) показано, что Ассемблер является даже более популярным, чем Си, в тех продуктах, которые способны дать большую прибыль.

более эффективными, если на процессоре имеется большой набор регистров общего назначения и единое пространство памяти. Такой подход позволяет минимизировать разницу между языками Си и Ассемблера по скорости выполнения программ и расширяет, таким образом, возможности применения языка высокого уровня для программирования ЦСП.

Чтобы лучше понять проблему выбора между Си и Ассемблером, рассмотрим типовую задачу, решаемую ЦСП с помощью программы, написанной на каждом из двух языков. Пример задачи, который мы рассмотрим, представляет собой вычисление скалярного произведения двух векторов-массивов  $x[ ]$  и  $y[ ]$ . Это достаточно простая математическая процедура: мы умножаем каждый элемент одного массива на соответствующий элемент другого массива и результаты суммируем, т. е.  $x[0] \times y[0] + x[1] \times y[1] + x[2] \times y[2] \dots$ . Это должно выглядеть знакомым для вас, так как это основная операция при реализации КИХ-фильтра: каждый отсчёт выходного сигнала рассчитывается умножением хранящихся в памяти отсчётов входного сигнала (один массив) на коэффициенты фильтра (другой массив) и суммированием произведений.

**Программа 28.1** показывает, как вычисление скалярного произведения реализуется на Си. В строках 001...004 мы определяем два массива  $x[ ]$  и  $y[ ]$ , длиной по 20 элементов. Программа вычисляет скалярное произведение этих двух массивов. Мы также определяем переменную *result*, в которую записывается вычисленное скалярное произведение по завершении выполнения программы. Стока 011 организует цикл из 20 итераций, за которые реализуется процедура, используя в качестве счётчика цикла переменную *n*. В цикле мы видим единственную строку —

строку 012, в которой осуществляется перемножение соответствующих элементов двух массивов и добавление результата умножения к аккумулятору — переменной *s* (если вы не знакомы с Си, то уточняем: команда *s += x[n]\*y[n]* означает то же самое, что *s = s + x[n]\*y[n]*). По окончании цикла в строке 013 значение аккумулятора *s* переписывается в выходную переменную *result*.

### **Программа 28.1. Скалярное произведение на Си**

```

001 #define LEN 20
002     float dm x[LEN];
003     float dm y[LEN];
004     float result;
005
006 main()
007 {
008     int n;
009     float s;
010     for (n=0;n<LEN;n++)
011         s+=x[n]*y[n];
012     result=s
013 }
```

Главным достоинством использования языка высокого уровня (такого как Си, Фортран или Бейсик) является то, что программист не нуждается в глубоком знании архитектуры используемого микропроцессора; эта функция переносится на компилятор. Например, в приведённой программе использовано несколько переменных: *n*, *s*, *result*, а также массивы *x[ ]*, *y[ ]*. Для всех этих переменных и массивов в памяти процессора должно быть выделено своё место, в которое будут записываться присваиваемые им значения. В зависимости от конкретного микропроцессора в качестве таких областей памяти могут выступать регистры общего назначения, основная память, специальные регистры, выделенные под определённые функции, и т. д. Однако тот, кто пишет программу на языке высокого уровня, знает мало или совсем ничего об этих особенностях процессора. Задача распределения памяти была решена тем, кто создавал компилятор. При этом программист процессора и разработчик компилятора не знают друг друга и находят взаимопонимание через набор предопределённых правил. Программирование на языке высокого уровня проще, чем на Ассемблере, потому что вы перекладываете половину работы на кого-то другого. Однако оно оказывается менее эффективным, потому что вы не можете полностью гарантировать, что эта чужая работа выполнена максимально хорошо.

**Программа 28.2**, написанная для ЦСП SHARC на Ассемблере, реализует вычисление скалярного произведения и позволяет сравнить языки высокого и низкого уровней. Язык Ассемблер, разработанный фирмой Analog Devices для выпускаемых ею ЦСП (и 16-битных с фиксированной точкой, и 32-битных с плавающей точкой), известен своим простым и удобным алгебраическим синтаксисом. Мы не будем рассматривать все детали, а опишем только общие идеи. Заметим, что всё, что вы видите в программе, относится непосредственно к аппаратным ресурсам процессора; здесь нет абстрактных переменных, а только имена регистров и адреса ячеек памяти.

### Программа 28.2. Скалярное произведение на Ассемблере (не оптимизировано)

```

001    i12=_y          /*i12 указывает на начало y[]*/
002    i4=_x          /*i4 указывает на начало x[]*/
003
004    lcntr=20, do(pc,4) until lce; /*цикл для 20-элементного массива*/
005    f2=dm(i4,m6);      /*загрузить значение x[] в регистр f2*/
006    f4=pm(i12,m14);    /*загрузить значение y[] в регистр f4*/
007    f8=f2*f4;         /*умножить два значения, запомнить в f8*/
008    f12=f8+f12;       /*прибавить произведение к аккумулятору f12*/
009
010    dm(_result)=f12   /*записать аккумулятор в память*/

```

Эта программа вычисляет скалярное произведение двух массивов  $x[ ]$  и  $y[ ]$ , формируя результат в переменной *result*. Каждое выражение, отделённое точкой с запятой, представляет собой команду, выполняемую за один такт. Массивы  $x[ ]$  и  $y[ ]$  организованы как циклические буферы и размещены в основной памяти. В строках 001 и 002 в регистры *i4* и *i12* записываются адреса этих массивов. Далее выполняются 20 итераций цикла, организуемого в строке 004. В формате этой команды проявляется одно из достоинств ЦСП семейства SHARC — *циклы с нулевыми издержками* (zero-overhead looping). Это означает, что все переменные, необходимые для управления циклом, хранятся в специальных регистрах, работа с которыми идёт параллельно с основным вычислительным процессом и не требует дополнительных затрат. В нашем примере в регистр *lcntr* (*счётчик цикла*) загружается начальное значение 20, и содержимое этого регистра уменьшается на единицу на каждой итерации цикла. Цикл заканчивается, когда значение *lcntr* становится равным нулю (это определено выражением *lce*, что является сокращением от «*loop counter expired*» — значение счётчика цикла достигло нуля). Тело цикла заключено в строках 004...008, что определено оператором (*pc*, 4). То есть цикл заканчивается через четыре строки после текущего программного счётчика.

Внутри тела цикла строка 005 загружает значение из массива  $x[ ]$  в регистр данных *f2*, а строка 006 загружает значение из массива  $y[ ]$  в регистр данных *f4*. Символы *dm* и *pm* указывают, что значения пересыпаются по «шине данных» и по «шине команд» соответственно. Переменные *i4*, *m6*, *i12* и *m14* — это регистры генератора адресов данных, которые управляют циклическими буферами — массивами  $x[ ]$  и  $y[ ]$ . Два значения в регистрах *f2* и *f4* перемножаются в строке 007, и результат запоминается в регистре данных *f8*. В строке 008 произведение из регистра *f8* складывается с аккумулятором — регистром данных *f12*. После окончания цикла аккумулятор *f12* пересыпается в память.

Эта программа правильно вычисляет скалярное произведение, но она не использует преимуществ высокопараллельной архитектуры процессора SHARC. Программа 28.3 представляет собой оптимизированный вариант этой программы, в котором многие операции выполняются параллельно. Во-первых, заметьте, что в строке 007 цикл организуется только на 18 итераций, а не на 20. И также заметьте, что сам цикл содержит теперь единственную строку (008), в которой осуществляется целый ряд операций. Основная стратегия при оптимизации — сделать код цикла максимально эффективным, т. е. в данном случае реализовать его в виде одной строки, выполняемой за один такт процессора. Для этого необходим не-

большой фрагмент программного кода, который подготавливает регистры к реализации первой итерации (строки 004 и 005), а также ещё один фрагмент кода, который завершает выполнение последней итерации (строки 010 и 011).

### **Программа 28.3. Скалярное произведение на Ассемблере (оптимизировано).**

**Это оптимизированная версия Программы 28.2, разработанная с учётом преимуществ высокопараллельной архитектуры SHARC**

```

001    i12=_y                      /*i12 указывает на начало y[]*/
002    i4=_x                      /*i4 указывает на начало x[]*/
003
004    f2=dm(i4,m6); f4=pm(i12,m14);      /*настроить регистры*/
005    f8=f2*f4, f2=dm(i4,m6), f4=pm(i12,m14);
006
007    lcntr=18, do(pc,1)until lce;        /*высокоэффективный главный цикл*/
008    f12=f8+f12, f8=f2*f4, f2=dm(i4,m6), f4=pm(i12,m14)
009
010   f12=f8+f12, f8=f2*f4;            /*завершить последний цикл*/
011   f12=f8+f12;
012
013   dm(_result)=f12                /*запомнить результат в памяти*/

```

Чтобы понять, как всё это работает, рассмотрим строку 008 — единственную строку в теле цикла. В этой одной строке параллельно выполняются четыре действия: 1) значение  $x[ ]$  читается из циклического буфера, расположенного в памяти программ, и загружается в регистр f2; 2) значение  $y[ ]$  читается из циклического буфера, расположенного в памяти данных, и загружается в регистр f4; 3) предыдущие значения, загруженные в регистры f2 и f4, перемножаются, и результат помещается в регистр f8; 4) предыдущее значение регистра f8 добавляется в аккумулятор — регистр f12.

Например, когда строка 008 выполняется в пятый раз, из памяти читаются значения  $x[7]$  и  $y[7]$  и загружаются в регистры f2 и f4. Одновременно значения  $x[6]$  и  $y[6]$  (которые находились в регистрах f2 и f4 перед началом выполнения строки) перемножаются, и результат помещается в регистр f8. Кроме того, произведение  $x[5] \times y[5]$  (результат которого находился в регистре f8 в начале выполнения команды) добавляется к содержимому регистра f12.

Сравним число тактов, которое затрачивается на выполнение неоптимизированной и оптимизированной программ. Вспомним, что неоптимизированная программа использовала 20 итераций цикла и 4 строки в теле цикла. Таким образом, ей потребовалось 80 тактов, чтобы выполнить цикл, плюс 5 тактов дополнительных затрат, т. е. всего 85 тактов. А оптимизированная программа выполняет 18 итераций цикла за 18 тактов, но требует ещё 11 дополнительных тактов, необходимых для подготовки первой и завершения последней итерации. Таким образом, всего на выполнение оптимизированной программы затрачивается 29 тактов, т. е. почти втрое меньше, чем на программу без оптимизации.

Здесь возникает важный вопрос: как быстро выполняется программа на Си по сравнению с программой на Ассемблере? Когда мы откомпилируем Программу 28.1, будет ли полученный ассемблерный код соответствовать оптимизированному варианту? Ответ: в данном случае — да, но следует заметить, что скалярное произведение — это слишком простой пример. Для компилятора гораз-

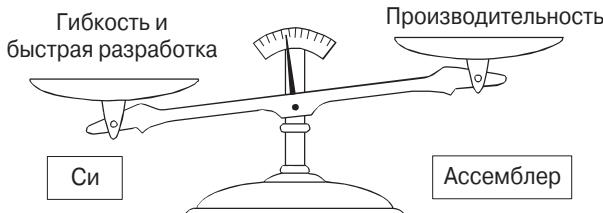
до сложнее сформировать оптимальный ассемблерный код, когда Си-программа оказывается не такой простой, а включает, например, вложенные циклы или вызовы подпрограмм. Если ваш код имеет простую, типовую структуру, можно ожидать, что компилятор даст результат, близкий к оптимальному. Если же ваша программа на Си неординарна или сложна, то имейте в виду, что ассемблерный код мог бы оказаться гораздо эффективнее. В худшем случае программа, написанная на Си, будет выполняться в 2...3 раза медленнее по сравнению с той же программой, написанной на Ассемблере. Как было отмечено выше, конкретный выигрыш от использования Ассемблера вместо Си сильно зависит от типа используемого процессора. В общем случае процессоры с плавающей точкой программируются на языках высокого уровня более эффективно, чем процессоры с фиксированной точкой. Конечно же в этом процессе важную роль играют соответствующие программные средства, такие как отладчик, поддерживающий возможность измерения времени выполнения отдельных фрагментов программы.

Есть также метод, который позволит вам получить лучшее от обоих языков программирования: пишите программы на Си и используйте Ассемблер для тех фрагментов кода, в которых наиболее важна скорость вычислений. Язык Си так популярен среди учёных и инженеров в частности потому, что он, будучи языком высокого уровня, позволяет в то же время непосредственно манипулировать нужными аппаратными ресурсами. Даже если вы планируете программировать только на Си, вам всё равно потребуются некоторые знания об архитектуре процессора и его наборе ассемблерных команд. Вернёмся, например, к строкам 002 и 003 в **Программе 28.1** — программе скалярного произведения, написанной на Си. Символ `dm` означает, что массив `x[ ]` следует разместить в памяти данных, а `rtm` — что массив `y[ ]` следует разместить в памяти программ. Несмотря на то что программа написана на языке высокого уровня, базовые знания архитектуры ЦСП всё же необходимы для того, чтобы добиться от устройства большей производительности.

Какой язык лучше подходит для вашего приложения? Это зависит от того, что для вас более важно. Если вам нужна гибкость и скорость разработки, выбирайте Си. С другой стороны, если вам нужны наилучшие характеристики, используйте Ассемблер. Как показано на **Рис. 28.9**, этот выбор потребует от вас взвесить все «за» и все «против» и прийти к определённому компромиссному решению. Здесь можно привести ряд моментов, на которые вы должны обратить внимание в первую очередь:

- насколько сложна программа? Если она большая и сложная, вам, вероятно, лучше использовать Си. Если она маленькая и простая, правильным выбором будет Ассемблер;
- требуется ли вам добиться от ЦСП максимальной скорости? Если да, Ассемблер «выжмет» вам до последней капли наилучшие характеристики из устройства. Для менее требовательных приложений Ассемблер может оказаться избыточным, и вам следует рассматривать возможность использования Си;
- сколько программистов работают вместе над данным приложением? Если проект достаточно большой и в нём задействовано несколько программистов, то лучшее взаимодействие между ними обеспечит Си, а Ассемблер в этом случае можно использовать в виде «*in-line*»-вставок (вставок фрагментов ассемблерного кода прямо в тексте Си-программы) в тех фрагментах программы, где быстродействие является действительно критичным;

- что более важно: стоимость продукта или стоимость разработки? Если стоимость продукта, то лучше подойдёт Ассемблер; если стоимость разработки — выбирайте Си;
- каков ваш личный опыт программирования? Если вы работали с Ассемблером (пусть даже на другом микропроцессоре), выбирайте Ассемблер. Если вы раньше работали только на Си, отдавайте предпочтение Си;
- руководствуйтесь тем, что вам советует производитель ЦСП.



**Рис. 28.9.** Ассемблер или Си? Программы на Си более гибкие и быстрые в разработке. В отличие от них программы на Ассемблере обычно имеют лучшие характеристики: они быстрее выполняются и используют меньше памяти, что в результате сказывается на снижении стоимости системы.

Последний пункт очень важен. Предположим, что вы спросили производителя ЦСП, какой язык использовать, и вам сказали: «Можно использовать Си или Ассемблер, но мы рекомендуем Си». Лучше следовать их рекомендациям! В действительности эта фраза расшифровывается так: «Наш ЦСП так труден для программирования на Ассемблере, что вам понадобится 6 месяцев, чтобы только научиться его использовать». С другой стороны, некоторые ЦСП легко программируются на Ассемблере. Например, к этой категории относятся продукты фирмы Analog Devices. Спросите их инженеров — они очень гордятся этим.

Один из лучших способов принять решение о ЦСП и программном обеспечении — поговорить с инженерами, которые уже использовали его. Узнайте у производителя о компаниях, которые работают с интересующим вас продуктом, или найдите компетентных людей с помощью Интернета и электронной почты. Не сомневайтесь — инженеры любят высказывать своё мнение о продуктах, которые они используют. Им польстит то, что вы с ними советуетесь.

## 28.6. Насколько быстры ЦСП?

Основной причиной использования ЦСП вместо микропроцессоров широкого назначения является их *скорость* — способность быстро принимать отсчёты входного сигнала, выполнять с ними необходимые математические операции и выводить обработанные данные. Здесь возникает вопрос: как оценить скорость ЦСП? Обычно ответ дают *тестовые задачи* (*benchmarks*), позволяющие выразить скорость конкретного ЦСП в виде числа. Например, производительность систем с фиксированной точкой принято оценивать в миллионах целочисленных операций, выполняемых ею в секунду, — *MIPS* (*Million Integer operations Per Second*). Аналогично скорость устройств с плавающей точкой оценивается в миллионах операций с плавающей точкой в секунду — *MFLOPS* (*Million Floating point Operations Per Second*).

Сто пятьдесят лет назад британский премьер-министр Бенджамин Дизраэли сказал, что существует три типа лжи: ложь, грубая ложь и статистика. Если бы Дизраэли жил сегодня и работал с микропроцессорами, он бы добавил *тестовые задачи* как четвёртую категорию. Идея, стоящая в основе *тестовых задач*, — дать возможность непосредственного сравнения устройств, чтобы показать, какое из них лучше. К сожалению, на практике такой подход часто оказывается неработоспособным, так как разные процессоры часто могут опережать друг друга по разным показателям. Представьте, что вас спрашивают: «Какой автомобиль лучше — «кадиллак» или «феррари»? Как здесь ответить, ведь они несопоставимы. Всё зависит от того, для чего вам нужен автомобиль!»

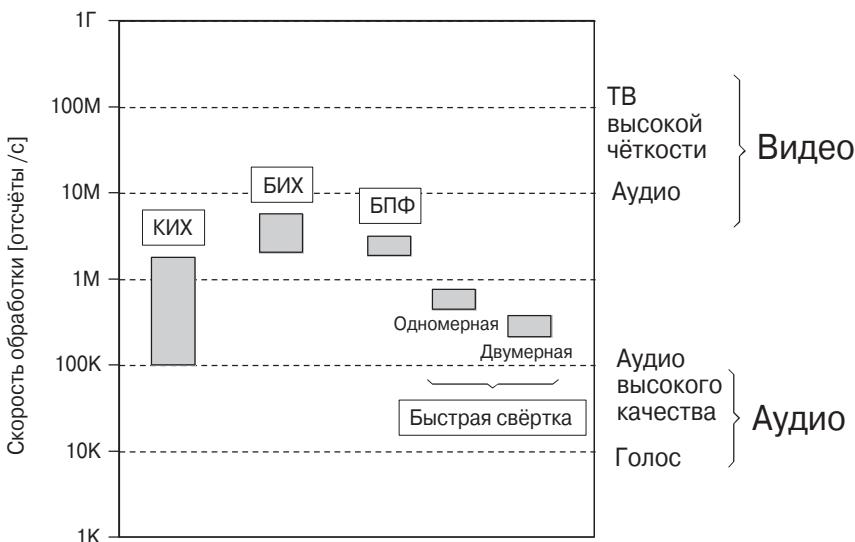
Сложности с использованием *тестовых задач* усугубляются конкурентной природой электронной промышленности. Производители хотят показать, что их продукт лучше всех других, и они будут использовать любую неопределённость в процедуре тестирования так, чтобы подчеркнуть достоинства своих изделий. В электронике есть старая поговорка: «Тот, кто пишет спецификацию, может «выжать» из устройства вдвое большую производительность, чем тот, кто это устройство разработал». Эти люди не лгут, просто им платят за их богатое воображение. Тестовые задачи следует рассматривать как инструмент для решения более сложной задачи оценки эффективности процессора. Если вы не умеете обращаться с этим инструментом, то вы, вполне вероятно, придёте к ошибочным выводам о характеристиках устройства. Лучшим подходом будет найти информацию о скорости выполнения на интересующем вас процессоре тех алгоритмов, которые вы намереваетесь использовать в вашей системе. Например, если вам необходимо реализовывать КИХ-фильтр, найдите информацию о точном числе тактов, которое требуется процессору для решения задачи КИХ-фильтрации.

Иллюстрируя этот подход, определим время, требуемое для выполнения различных алгоритмов на выбранном нами в качестве примера ЦСП семейства SHARC фирмы Analog Devices. Имейте в виду, что скорость работы микропроцессоров удваивается каждые три года. Это означает, что вам необходимо обратить особое внимание на сам метод оценки производительности, а не на конкретные числа. Эти числа постоянно меняются, и вам придётся повторять вычисления каждый раз в начале работы над новым проектом. В мире технологий XXI в. стоит моргнуть, и вы уже далеко отстали от жизни!

Процессоры семейства SHARC оказываются одними из наиболее удобных для демонстрации того, как оценивается время выполнения различных задач. Дело в том, что операции умножения и накопления производятся в этих процессорах в одном такте. Например, если фильтры класса КИХ обычно включают 25...400 коэффициентов, то и время выполнения задачи КИХ-фильтрации на процессоре SHARC будет лежать в диапазоне 25...400 тактов на каждый отсчёт обрабатываемого сигнала. Как указывалось ранее, при реализации фильтра существуют небольшие дополнительные затраты времени, требуемые, в частности, для организации цикла свёртки (в первую очередь эти издержки появляются на первой и последней итерациях цикла), но обычно ими можно пренебречь. Чтобы рассчитать пропускную способность процедуры фильтрации, мы можем разделить тактовую частоту процессора (на момент написания книги — 40 МГц) на число тактов, приходящихся на обработку одного отсчёта. Таким образом, мы получим максимальную пропускную способность КИХ-фильтра, лежащую в диапазоне от 100 тысяч до

1.6 миллиона отсчётов в секунду. Как видите, расчёты — проще некуда! Полученные нами значения скорости реализации КИХ-фильтрации приведены на Рис. 28.10.

Расчёт времени выполнения также прост и для рекурсивного фильтра. БИХ-фильтры обычно используют 5...17 коэффициентов. Так как циклы, реализующие фильтрацию, будут в этом случае достаточно короткими (малое число итераций), мы не сможем пренебречь издержками и добавим немного времени (скажем, 3 такта) на обработку одного входного отсчёта. В результате получим, что на обработку одного отсчёта требуется 8...20 тактов процессора. При тактовой частоте 40 МГц максимальная пропускная способность процедуры БИХ-фильтрации составит от 1.8 до 3.1 миллиона отсчётов в секунду. Эти значения также приведены на Рис. 28.10.



**Рис. 28.10** Скорость ЦСП. Скорость выполнения конкретного алгоритма ЦОС может быть найдена делением частоты тактовых импульсов на требуемое для выполнения число тактов на один отсчёт входного сигнала. Эта иллюстрация показывает диапазон скоростей выполнения четырёх общезвестных алгоритмов, выполняемых на ЦСП SHARC с тактовой частотой 40 МГц.

Теперь нам следует обратиться к методам, работающим в частотной области и основанным на алгоритме быстрого преобразования Фурье. Подпрограммы БПФ, эффективно реализуемые на конкретном процессоре, почти всегда предоставляются производителем. Это высокооптимизированные программные коды, написанные на Ассемблере. В спецификации ЦСП ADSP-21062 семейства SHARC указано, что комплексное 1024-точечное БПФ выполняется за 18221 такт, или приблизительно за 0.46 мс при тактовой частоте 40 МГц. Для расчёта пропускной способности процедуры БПФ лучше представить показатель на обработку одного отсчёта — 17.8 тактов. Затраты времени на один отсчёт слабо зависят от того, малой или большой размерности реализуется БПФ. Например, 256-точечное БПФ требует около 14.2 тактов на отсчёт, а 4096-точечное — 21.4 тактов. Действительное БПФ может быть вычислено примерно на 40% быстрее, чем комплексное БПФ при тех же данных. Таким образом, вся совокупность программ

БПФ характеризуется диапазоном скоростей обработки от 10 до 22 тактов на отсчёт, что соответствует пропускной способности процедур БПФ от 1.8 до 3.3 миллиона отсчётов в секунду.

Быстрая свёртка (свёртка в частотной области с использованием БПФ) является быстрым способом реализации КИХ-фильтров. В типовом случае из входного сигнала выбирается сегмент из 512 отсчётов, которые дополняются ещё 512 нулевыми отсчётами и преобразуются в частотную область с помощью 1024-точечного БПФ. После умножения этого спектра сигнала на желаемую частотную характеристику фильтра полученный сигнал переносится во временную область с помощью обратного 1024-точечного БПФ. Полученные таким образом 1024 отсчёта объединяются со смежными сегментами обработанного сигнала с использованием перекрытия с накоплением. В результате этого получаются 512 отсчётов выходного обработанного сигнала.

Сколько тактов это занимает? На каждый 512-точечный сегмент приходятся два 1024-точечных БПФ-преобразования плюс небольшие издержки. Округляя, можно сказать, что требуется в 5 раз больше тактов, чем на выполнение одного 512-точечного БПФ. Так как действительное БПФ характеризуется скоростью около 12 тактов на отсчёт, то время обработки одного отсчёта при реализации быстрой свёртки составит приблизительно 60 тактов. Для ЦСП 2106x SHARC с частотой 40 МГц это соответствует пропускной способности процедуры свёртки в частотной области, равной около 660 тысяч отсчётов в секунду.

Заметьте, это фактически столько же, сколько требует КИХ-фильтр с 60-ю коэффициентами, реализуемый по обычному алгоритму во временной области. Другими словами, если КИХ-фильтр насчитывает менее 60 коэффициентов, то его лучше реализовывать обычной свёрткой. Если же число коэффициентов превышает 60, быстрее будет работать быстрая свёртка. Ключевым преимуществом свёртки в частотной области является то, что время её выполнения возрастает пропорционально логарифму числа коэффициентов. Например, 4096-точечное ядро фильтра потребует только на 30% больше времени по сравнению с 512-ю коэффициентами.

Быстрая свёртка может также использоваться и в случае двумерных сигналов, например при обработке изображений. Предположим, что мы хотим обработать изображение размером  $800 \times 600$  пикселей в частотной области. Сначала дополним изображение нулями, чтобы получить размер  $1024 \times 1024$  пикселей. Двумерный частотный спектр вычисляется взятием БПФ от каждой строки и последующим БПФ от каждого получившегося столбца. После умножения этого спектра на желаемую частотную характеристику дважды выполняется обратное БПФ. Сначала выполняется обратное БПФ по каждой строке, а затем по каждому получившемуся столбцу. Просуммировав число тактов, приходящихся на каждое частное преобразование, и разделив результат на число отсчётов, мы получим, что вся процедура потребует приблизительно 150 тактов на один пиксель. Для процессора ADSP-2106 с частотой 40 МГц это соответствует пропускной способности процедуры двумерной быстрой свёртки, равной около 260 тысяч отсчётов в секунду.

Сравнивая все эти разные алгоритмы на Рис. 28.10, мы можем сделать важное заключение. Почти все методы ЦОС используют 4...400 команд программного кода (равных числу тактов в случае процессоров SHARC). Для ЦСП SHARC, ра-

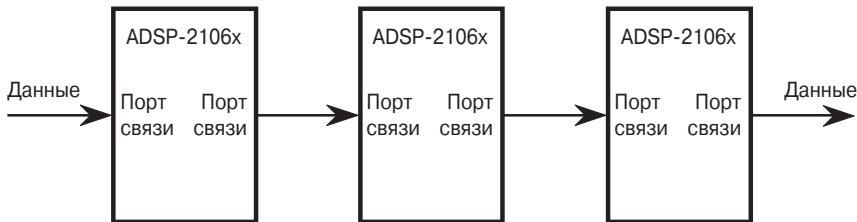
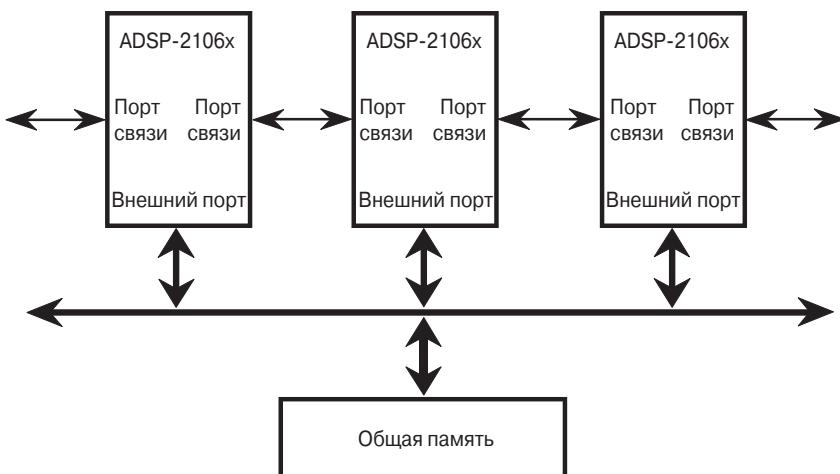
ботающего на частоте 40 МГц, мы можем сразу же определить, что его пропускная способность будет лежать в пределах от 100 тысяч до 10 миллионов отсчётов в секунду в зависимости от сложности алгоритма.

Теперь, когда мы научились оценивать, как быстро ЦСП может обрабатывать цифровые сигналы, обратим наше внимание на другую сторону того же вопроса: как быстро нам нужно обрабатывать данные? Конечно, это будет зависеть от конкретного приложения. Мы рассмотрим два самых распространённых примера — аудио- и видеосистемы.

Скорость обработки, необходимая при реализации аудиосистем, зависит от требуемого качества воспроизведимого звука. С одной стороны, передача голоса по телефонным сетям использует только частотный диапазон 10 Гц...3.2 кГц, и для неё достаточно скорости обработки на уровне 8000 отсчётов в секунду. С другой стороны, аудиосистемы высокого качества требуют использования всего слышимого диапазона звуковых частот 20 Гц...20 кГц. При типовой в этом случае частоте дискретизации 44.1 кГц для левого и правого каналов требуется скорость обработки 88.2 тысячи отсчётов в секунду для полного стереосигнала. Как семейство SHARC соответствует этим требованиям? **Рис. 28.10** показывает, что эти процессоры могут легко справиться с задачами воспроизведения аудио высокого качества, либо за то же время обработать до нескольких десятков голосовых сигналов.

С видеосигналами другая история. Они требуют примерно в 1000 раз большую скорость обработки по сравнению с аудиосигналами. Хорошим примером видео низкого качества является стандарт CIF (Common Interface Format), применяемый для видеотелефонов. Он использует  $352 \times 288$  пикселей, три цвета на пиксель и 30 кадров в секунду, что соответствует общей скорости обработки данных, равной 9.1 миллиона отсчётов в секунду. С другой стороны, высокое качество видео обеспечивается стандартом HDTV (телевидение высокой чёткости), использующим  $1920 \times 1080$  пикселей, три цвета на пиксель и 30 кадров в секунду. Этот стандарт требует скорости обработки данных выше 186 миллионов отсчётов в секунду. Такая скорость лежит за пределами возможностей одного ЦСП SHARC, как показано на **Рис. 28.10**. Существуют и другие приложения, требующие очень высокой скорости обработки. Это, например, радиолокация, гидролокация и применения в оборонной промышленности, в частности в задачах наведения ракет.

Чтобы справиться с подобными чрезвычайно ресурсоёмкими задачами, несколько ЦСП объединяют в единую систему, таким образом переходя к так называемой *многопроцессорной обработке*. ЦСП семейства SHARC проектировались с учётом возможной необходимости создания на их основе многопроцессорных систем, и их архитектура предусматривает ряд специальных средств, делающих создание многопроцессорной системы максимально удобным. В частности, чтобы соединить вместе внешние шины нескольких ЦСП SHARC, не требуется никакой дополнительной логики. Все элементы, осуществляющие управление взаимодействием процессоров через эти шины, содержатся прямо на кристаллах. Альтернативный вариант объединения нескольких процессоров в единую систему обработки обеспечивают *порты связи* (link-ports — 4-битные параллельные порты). **Рис. 28.11** иллюстрирует типовые методы объединения процессоров SHARC в многопроцессорную систему. В случае, показанном на **(а)**, алгоритм обработки разбивается на ряд последовательных шагов, и каждый процессор реали-

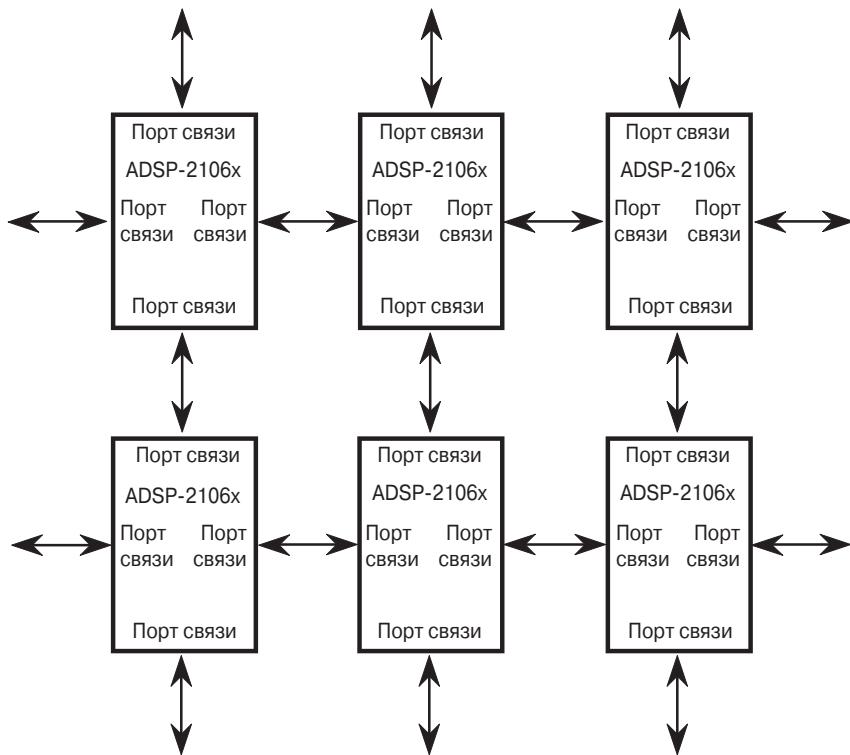
**а) Структура на основе разбиения потока данных****б) Кластерная структура**

**Рис. 28.11.** Конфигурации многопроцессорных систем. Обычно многопроцессорные системы используют одну из двух схем взаимодействия между процессорными элементами: выделенные линии связи точка-точка (**а**) или общую глобальную память, доступ к которой осуществляется через параллельную шину (**б**).

зует один из этих этапов конвейера обработки сигнала. На (**б**) процессоры взаимодействуют через одну общую глобальную память, доступ к которой они получают по параллельнойшине (т. е. через внешний порт). **Рис. 28.12** иллюстрирует другой способ, с помощью которого большое число процессоров может быть объединено в единую систему — двумерную или трёхмерную ячеистую структуру. Каждая из конфигураций будет иметь свои достоинства и недостатки в зависимости от конкретной задачи.

Чтобы сделать жизнь программиста легче, процессоры семейства SHARC используют *единое адресное пространство*. Это означает, что все адресное пространство объёмом 4 Глов, адресуемое 32-битной адреснойшиной, разделяется между разными процессорами, работающими в системе совместно. Чтобы передать данные с одного процессора на другой, достаточно просто обратиться (считать или записать данные) к соответствующей области памяти. Внутренняя логика SHARC сама позаботится обо всём остальном, организуя передачу данных

между процессорами на скорости примерно 240 МБ/с (при частоте процессора 40 МГц).

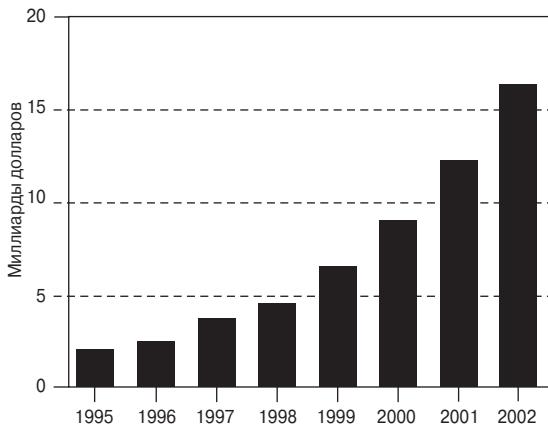


**Рис. 28.12.** Ячеистая конфигурация многопроцессорных систем. В таких задачах, как, например, формирование радиолокационных изображений, двумерные и трёхмерные структуры могут быть наиболее эффективным способом объединения большого числа процессоров в одну систему.

## 28.7. РЫНОК ЦИФРОВЫХ СИГНАЛЬНЫХ ПРОЦЕССОРОВ

Рынок ЦСП очень большой и быстрорастущий. Как показано на Рис. 28.13, в 2000 г. он составлял около 8...10 миллиардов долларов в год и в последующий период увеличивался на 30...40% каждый год. Такой темп роста поддерживается непрерывно возрастающим спросом на всё более функциональные и одновременно более дешёвые продукты массового потребления, такие как сотовые телефоны, мультимедийные компьютеры, цифровые аудиоплееры. Эти высокодоходные приложения формируют рынок ЦСП; при этом менее прибыльные области, такие как системы научных измерений, подхватывают волну новых технологий.

Покупка ЦСП может осуществляться в трёх формах: *ядро*, *процессор* или *встраиваемый продукт*. Термин «ядро» применительно к ЦСП обозначает часть процессора, в которой выполняются основные задачи. В него входят регистры данных, умножитель, АЛУ, генераторы адреса и счётчик команд. Процессор объединяет ядро с памятью и блоками периферии, обеспечивающими интерфейс с внешним миром. Ядро процессора и его периферия разрабатываются отдельно, но в результате они выполняются на одном кристалле кремния и образуют одну микросхему.



**Рис. 28.13.** Рынок ЦСП. В 2000 г. рынок ЦСП составлял 8...10 миллиардов долларов в год и в последующий период увеличивался на 30...40% ежегодно.

Предположим, вы собираетесь выпустить сотовый телефон и хотите включить ЦСП в его конструкцию. Наиболее вероятно, что вы приобретёте ЦСП в виде процессора — микросхемы, которая содержит ядро, память и другие блоки. Например, процессор ADSP-21060 семейства SHARC представляет собой микросхему в 240- выводном корпусе типа PQFP размером 35×35×4 мм. Чтобы включить эту микросхему в ваш продукт, необходимо спроектировать печатную плату, на которой она будет размещаться наряду с другими необходимыми вам компонентами. Это наиболее распространённый вариант использования ЦСП.

Теперь предположим, что вы работаете в компании, которая производит свои собственные интегральные схемы. В этом случае вам может оказаться интересен не весь процессор, а только архитектура его ядра. После заключения соответствующего лицензионного соглашения вы сможете начать изготовление микросхем, которые оптимальны для вашего конкретного приложения. Вы получаете свободу в выборе размеров внутренней памяти, способов приёма и передачи данных, типа корпуса и т. д. Такие заказные устройства образуют сегмент, играющий все большую роль на рынке ЦСП.

И наконец, существуют несколько десятков компаний, у которых вы можете приобрести ЦСП, уже размещённый на некоторой печатной плате. Здесь вы сразу получаете возможность использования дополнительных компонентов, таких как внешние микросхемы памяти, АЦП и ЦАП, разъёмы перепрограммиру-

емых ПЗУ, другие процессоры на той же плате и т. д. Некоторые из таких плат предназначены для использования в качестве автономных модулей, но большинство встраивается в хост-систему, например, персональный компьютер. Компании, которые производят такие типы плат, называются *разработчиками третьей стороны* (*third party developers*). Узнать о таких компаниях лучше всего у производителя того ЦСП, который вы собираетесь использовать. Загляните на веб-сайт. Если информации там не окажется, свяжитесь с ними по электронной почте. Они будут счастливы рассказать вам, кто использует их продукты и как с ними связаться.

На сегодняшний день на рынке цифровых сигнальных процессоров доминируют две компании — Analog Devices и Texas Instruments. Ниже приведён список их продуктов и общая схема, которую они используют для обозначения своих процессоров.

#### Analog Devices ([www.analog.com/dsp](http://www.analog.com/dsp))

- ADSP-218x 16-битный, фиксированная точка;
- ADSP-219x 16-битный, фиксированная точка, вдвое большая производительность по сравнению с 218x;
- ADSP-2106x 32-битный, плавающая точка;
- ADSP-2116x 32-битный, плавающая точка, SIMD-ядро;
- ADSP-2153x 16-битный, высокая вычислительная производительность;
- ADSP-TSxxx 8-, 16- и 32-битный, фиксированная и плавающая точка, MIMD, сверхвысокая вычислительная производительность;

#### Texas Instruments ([www.ti.com](http://www.ti.com))

- TMS320C6xx 16/32-битный, фиксированная/плавающая точка;
- TMS320C5xx 16-битный, фиксированная точка;
- TMS320C2xx 16-, 32-битный, фиксированная точка.

Имейте в виду, что граница между ЦСП и другими микропроцессорами не всегда оказывается чёткой. Взгляните, например, как Intel описывает свою технологию MMX, которая усовершенствует процессор Pentium:

*«Инженеры Intel добавили 57 новых мощных команд, специально разработанных для эффективной обработки видео, аудио и графических данных. Эти команды ориентированы в основном на параллельные, повторяющиеся последовательности данных, которые обычно характерны для мультимедийных приложений».*

В будущем мы будем наблюдать всё большее количество традиционных для ЦСП функций, появляющихся у микропроцессоров широкого назначения и микроконтроллеров. Интернет и другие мультимедийные приложения оказывают большое влияние на подобные перемены. Эти приложения развиваются так быстро, что вполне возможно через двадцать лет цифровой сигнальный процессор станут уже называть обычным микропроцессором.

Как угнаться за этими быстро меняющимися технологиями? Наилучший вариант — читать профессиональные журналы, которые посвящены рынку ЦСП, такие, как EDN (Electronic Design News, [www.ednmag.com](http://www.ednmag.com)) и ECN (Electronic Component News, [www.ecnmag.com](http://www.ecnmag.com)). Они свободно распространяются и содержат новейшую информацию о текущем состоянии и о направлениях развития микропроцессорных технологий. Читать такие отраслевые журналы просто необходимо каждому, кто всерьёз занят в сфере ЦОС-технологий. Также желательно числить-

ся в списках e-mail-рассылок ряда основных производителей. Это позволит получать информацию о появлении новых продуктов, ценах и специальных предложениях (свободном программном обеспечении, снижении цен на средства разработки и т. д.). Некоторые фирмы также рассылают еженедельные или ежемесячные письма со списком новостей от производителя. Analog Devices, в частности, имеет в Интернете свой сайт, который постоянно обновляется последней информацией.

Вы можете подписаться на некоторые рассылки на сайте: [http://www.analog.com/misc/enewsletters\\_jumpage.html](http://www.analog.com/misc/enewsletters_jumpage.html). Всегда следует начинать с поиска сайта производителя в Интернете, а затем отправлять ему по почте запрос на включение вас в список рассылки новостей.

## НАЧИНАЕМ РАБОТАТЬ С ЦСП

Если вы решили, что *цифровой сигнальный процессор* необходим для вашего приложения, у вас возникает вопрос: с чего начать? Обычно производители предлагаю в таких случаях приобрести у них так называемый *стартовый набор*, позволяющий оценить их продукт «из первых рук». Это великолепный инструмент для обучения. Не имеет значения, новичок вы или профессионал, — такой набор является лучшим способом знакомства с конкретным ЦСП. Например, фирма Analog Devices для ознакомления потенциальных покупателей с семейством ЦСП SHARC разработала стартовый набор EZ-KIT® Lite. За небольшую цену вы получаете все аппаратные и программные средства, необходимые для того, чтобы опробовать ЦСП в действии. В набор также часто входят примеры программного обеспечения, которые вы можете выполнять на процессоре или встраивать в собственные приложения на Си или Ассемблере. Предположим, вы приобрели один из таких стартовых наборов от Analog Devices и планируете поработать с ним в ближайшие дни. В этой главе мы расскажем о том, что вы сможете использовать, что понять и реализовать.

### 29.1. Семейство ADSP-2106x

В предыдущей главе мы рассмотрели семейство *цифровых сигнальных процессоров* ADSP-2106x SHARC в общих чертах. В **Табл. 29.1** перечислены разные типы процессоров этого семейства. Все они используют одну и ту же архитектуру, но имеют различный объём внутренней памяти, что часто является ключевым фактором при выборе конкретного устройства. Дело в том, что обмен с памятью — одно из самых слабых мест в системах с ЦСП. Процессоры SHARC частично снимают эту проблему с помощью размещённой на кристалле *двухпортовой памяти* типа *SRAM* (*статическое ОЗУ*). При этом у вас совершенно не будет желания покупать память большего объёма, чем вам нужно, ведь ЦСП очень часто входят в состав сильно чувствительных к цене конечных продуктов, таких как, например, сотовые телефоны или CD-плееры. Таким образом, номенклатура процессоров семейства SHARC ADSP-2106x определяется не только технологией, но и маркетингом<sup>1)</sup>.

Отметим, что даже модели, находящиеся на самых нижних позициях в **Табл. 29.1**, содержат значительный объём внутренней памяти. Например, процессор ADSP-21065L включает SRAM объёмом 544 Кбит. Этого достаточно, чтобы хранить 6...8 секунд оцифрованной речи (8000 отсчётов в секунду, 8 бит на один

<sup>1)</sup> Названия SHARC, EZ-KIT, EZ-LAB, VisualDSP, EZ-ICE, а также логотипы SHARC, Analog Device и VisualDSP — это зарегистрированные торговые марки фирмы Analog Devices.

отсчёт). В то же время передовой процессор ADSP-21060 содержит память объёмом 4 Мбита. Этого более чем достаточно, чтобы полностью записать цифровое изображение ( $512 \times 512$  пикселей, 8 бит на пиксель). Если вам требуется память ещё большего объёма, можно легко добавить внешнюю микросхему SRAM-памяти (или памяти более медленного типа) к любому из устройств.

Процессоры отличаются не только объёмом памяти, но и устройствами ввода/вывода — *периферией*. ЦСП ADSP-21060 и ADSP-21062 (наиболее мощные) включают шесть *портов связи* (*link-портов*) каждый. Это 4-битные параллельные порты, используемые для объединения ЦСП в мультипроцессорные системы, а также для других задач, требующих гибкого механизма ввода/вывода на высокой скорости. Процессоры ADSP-21061 и ADSP-21065L (наименее мощные) не имеют портов связи, но включают большее число каналов *прямого доступа к памяти* — ПДП (DMA — Direct Memory Access), которые помогают повысить эффективность работы последовательного порта. Кроме того, вы можете увидеть рядом с обозначением этих устройств буквы «L» или «M», например ADSP-21060L. Это говорит о том, что устройства работают с напряжением меньше традиционных 5 В. Например, процессор ADSP-21060L питается от напряжения 3.3 В, а ADSP-21060M — от напряжения 2.5 В.

Таблица 29.1. Процессоры семейства SHARC

Процессор	Объём памяти	Примечания
ADSP-TS101	6 Мбит	SIMD-ядро (одна команда, много данных); очень быстрый и функциональный; оптимизирован для мультипроцессорных систем; ядро с низким напряжением питания и малой рассеиваемой мощностью
ADSP-21160	4 Мбит	SIMD-ядро; оптимизирован для мультипроцессорных систем: включает набор портов связи, 64-битную внешнюю шину и 14-канальный контроллер ПДП
ADSP-21161	1 Мбит	Более дешёвый вариант процессора ADSP-21160 с меньшим числом портов связи и меньшим объёмом внутренней памяти, но с дополнительными протоколами последовательных портов (SPI, I <sub>2</sub> S)
ADSP-21060	4 Мбит	SISD-ядро (одна команда, одни данные); порты связи, обеспечивающие высокую скорость передачи данных и объединение в мультипроцессорную систему; 48-битная внешняя шина
ADSP-21062	2 Мбит	Те же особенности, что и у ADSP-21060, но объём внутренней памяти (SRAM) уменьшен, чтобы снизить стоимость процессора
ADSP-21061	1 Мбит	Недорогой вариант, используемый в стартовых наборах EZ-KIT Lite: меньший объём памяти, портов связи нет, но зато добавлены дополнительные возможности ПДП, поддерживающие работу последовательного порта
ADSP-21065	544 Кбит	Более быстрый и менее дорогой, чем другие процессоры ADSP-2106x; должен привлечь к семейству SHARC внимание тех, кто ведёт разработку систем на процессорах с фиксированной точкой или проектирует системы обработки аудиосигналов

В июне 1998 года фирма Analog Devices разработала второе поколение архитектуры SHARC с анонсированием процессора ADSP-21160. Процессоры этого класса отличают SIMD-архитектура ядра, работающего на частоте 100 МГц, модернизированная шина памяти с пропускной способностью 1600 Мбит в секунду, две 64-битные шины данных и четыре 80-битных аккумулятора для вычислений с фиксированной точкой. Процессор ADSP-21160M выполняет 1024-точечное БПФ за 46 микросекунд. SIMD-ядро содержит двойной набор вычислительных блоков (арифметико-логические устройства, сдвигатели, регистровые файлы и

умножители), позволяя фирме Analog Devices обеспечивать совместимость по программным кодам старых процессоров семейства SHARC с новыми, превышающими своих предшественников по производительности в 10 раз.

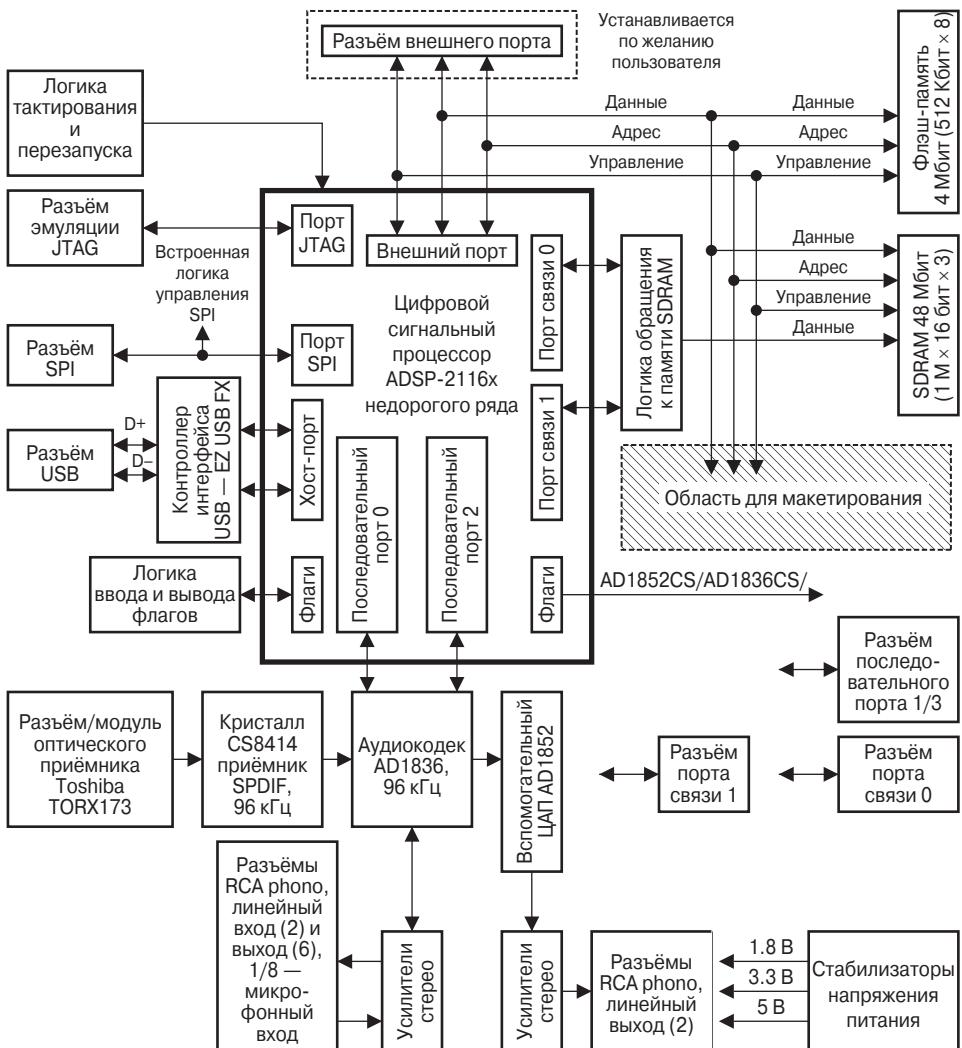
## 29.2. Стартовый набор SHARC EZ-KIT Lite

*EZ-KIT Lite* предлагает вам всё, что необходимо для изучения ЦСП SHARC, включая аппаратное и программное обеспечение, а также документацию. На Рис. 29.1 показана структура модуля EZ-KIT Lite, ядром которого является ЦСП ADSP-21161. Этот модуль представляет собой печатную плату размером 13.5×20 см. Вам лишь нужно позаботиться о соединении платы с источником питания постоянного тока, USB-интерфейсе с персональным компьютером, а также подсоединении входного и выходного сигналов. Адаптер для подключения к сети питания и USB-кабель поставляются в комплекте. В качестве входного и выходного сигналов могут использоваться аудиосигналы. В этом случае ко входу подключается микрофон, а к выходу — небольшие колонки (такие, которые обычно используются с персональными компьютерами). В результате получается система, с помощью которой можно достаточно эффективно изучать различные алгоритмы ЦОС, воспринимая на слух влияние обработки на входные сигналы.

Аналого-цифровое и цифро-аналоговое преобразования выполняют размешённый на плате кодек (кодер-декодер) AD1836 фирмы Analog Devices. Он ориентирован в первую очередь на работу с аудиосигналами и обеспечивает частоту дискретизации до 96 кГц. В его состав входит также ЦАП AD1852.

На плате имеется ряд кнопок, позволяющих пользователю сгенерировать прерывание, выполнить перезапуск процессора или установить какие-либо флаги, значения которых могут быть прочитаны системой. Шесть светодиодов, смонтированных на плате, можно программно включать и выключать, задавая различные значения управляющих бит. Если вы хотите использовать ресурсы платы на более серьёзном уровне, для вас предусмотрен ряд разъёмов, открывающих доступ к последовательным портам, портам связи, оптическому интерфейсу SPDIF и шине процессора.

Рассмотрим, как работает плата. Когда на неё подаётся питание, в процессор из размещённой на плате флэш-памяти загружается программа, устанавливающая связь с персональным компьютером через интерфейс USB и разъём JTAG. Затем вы запускаете на своём ПК среду VisualDSP++, позволяющую загружать в процессор программы, пересыпать данные между ЦСП и ПК, выполнять компиляцию и ассемблирование программных кодов, осуществлять построение проектов для реализации на плате. Имеется возможность использовать утилиту программирования флэш-памяти, что позволяет делать собственные программы загружаемыми автоматически при включении и создавать автономные приложения (не нуждающиеся в подключении к ПК). С набором EZ-KIT Lite поставляется ряд заранее написанных типовых программ обработки сигналов. Например, программный проект, реализующий полосовой фильтр, сразу позволит вам увидеть результат фильтрации вашей речи, произносимой в подключённый к плате микрофон. Такие программы очень полезны по двум причинам: 1) они позволяют быстро построить систему, выполняющую что-то интересное и демонстрирующую работоспособность технологии; 2) они обеспечивают вас работающим шаб-



**Рис. 29.1.** Структурная схема платы EZ-KIT Lite. Необходимо лишь организовать несколько соединений: на плате предусмотрен ряд различного типа разъёмов ввода/вывода аудиосигналов, разъёмы RCA на несколько каналов, интерфейс SPDIF. Необходимо также обеспечить питание и подключить кабель USB для интерфейса с персональным компьютером (всё, что для этого нужно, поставляется в комплекте).

лоном, на основе которого вы можете создавать собственные программные решения. Здесь мы подходим к теме нашего следующего раздела — примеру проектирования с использованием EZ-KIT Lite. Отметим, что приводимый пример проектирования был разработан для ранее выпущенного набора на базе процессора ADSP-21061. Однако этот процессор совместим по кодам с более новым ADSP-21161, что делает пример справедливым и для платы EZ-KIT Lite ADSP-21161, а также позволяет увидеть преимущество в скорости нового процессора за счёт меньшего времени выполнения команд (большой тактовой частоты).

## 29.3. Пример проектирования: КИХ-фильтр обработки аудиосигналов

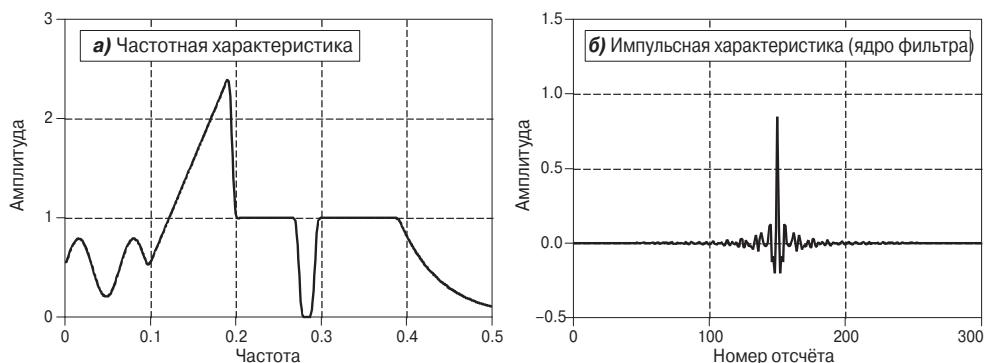
После экспериментов с готовыми примерами программ, поставляемыми с платой, вы очень скоро захотите что-нибудь в них изменить, так чтобы они выполняли то, что хочется вам. Программы могут быть написаны на Ассемблере или на Си; EZ-KIT Lite обеспечивает программные инструменты для поддержки обоих языков. Ниже мы рассмотрим процесс программирования более детально, включая работу с *симулятором*, процесс отладки и работу в *интегрированной среде разработки*. Сейчас же мы рассмотрим простейший вариант подготовки собственной программы к выполнению. Сделаем первый маленький шаг.

Исходные коды имеющегося в нашем распоряжении программного обеспечения записаны в простой ASCII-кодировке. Поэтому, чтобы внести в них какие-либо изменения и создать собственную программу, достаточно простого текстового редактора. **Программа 29.1** является примером программы, реализующей КИХ-фильтр, написанной на Ассемблере. Это пока единственный программный код, который должен иметь для вас значение. Однако помните, что для работы всего программного обеспечения необходимы и другие файлы. К ним, в частности, относятся «файл описания архитектуры» (определяет конфигурацию устройства и распределение памяти), файл определения векторов прерываний, программа инициализации кодека и т. д. Иногда у вас будет возникать необходимость разбираться с содержимым этих файлов, но сейчас достаточно просто скопировать их в свой проект из ранее разработанных программных продуктов.

В самых первых строках **Программы 29.1** определяются три объекта, которые необходимо установить до перехода к основной части программы. Это число коэффициентов ядра фильтра NR\_COEF, циклический буфер, содержащий последние отсчёты входного сигнала dline[], и циклический буфер коэффициентов фильтра coef[]. В программу необходимо также передать ещё два параметра: частоту дискретизации кодека и имя файла, содержащего ядро фильтра, которое нам надо считать в массив coef[]. Все эти действия очень просты — они требуют только по одной строке программного кода. Мы не приводим их в примере, потому что они содержатся в той части кода, который мы для простоты опускаем.

**Рис. 29.2** показывает ядро фильтра, с помощью которого мы будем тестировать программу. Это тот же специальный фильтр, который мы спроектировали в Главе 17. Вспомним, что отличительной особенностью данного фильтра является неравномерная частотная характеристика, подчёркивающая возможность КИХ-фильтров представлять системы практически с любой частотной характеристикой. На (а) показана частотная характеристика нашего тестового фильтра, а на (б) — соответствующая импульсная характеристика (т. е. ядро фильтра). Ядро фильтра, включающее 301 коэффициент, хранится в ASCII-файле и объединяется с другими частями программы во время компоновки, формируя единый исполняемый код.

В основной части программы выполняются два действия. Во-первых, в строках 6...13 осуществляется организация циклических буферов dline[] и coef[] путём конфигурирования регистров в генераторах адреса данных. Как было описано в предыдущей главе, на каждый буфер требуется три параметра: адрес начала буфера (b0 и b8), длина буфера (l0 и l8) и шаг выборки данных из буфера (m0 и m8). Значения этих параметров, которые и управляют циклической буферизацией, записываются в регистры генераторов адресов данных.



**Рис. 29.2.** Пример КИХ-фильтра. На (а) показана частотная характеристика специального фильтра, для которого характерна строго заданная форма АЧХ. Соответствующая ей импульсная характеристика (ядро фильтра) изображена на (б). Фильтр был спроектирован нами в Главе 17, когда мы хотели показать, что практически любая частотная характеристика может рассматриваться как характеристика цифрового КИХ-фильтра.

### Программа 29.1. Программа КИХ-фильтрации на Ассемблере<sup>1)</sup>

```

001 ****
002 ***** MAIN PROGRAM ****
003 ****
004 main:
005
006 /* ОРГАНИЗАЦИЯ ЦИКЛИЧЕСКИХ БУФЕРОВ */
007
008 b0 = dline; /* буфер dline[ ], содержащий последние отсчеты входного сигнала */
009 l0 = @dline;
010 m0 = 1;
011 b8 = coef; /* буфер коэффициентов фильтра coef[ ] */
012 l8 = @coef;
013 m8 = 1;
014
015 /* БЕСКОНЕЧНЫЙ ЦИКЛ. ОЖИДАНИЕ ПРЕРЫВАНИЯ ГОТОВНОСТИ ВХОДНОГО ОТСЧЕТА */
016
017 wait:
018 idle;
019 jump wait;
020
021
022 ****
023 ***** ПОДПРОГРАММА ОБРАБОТКИ ОДНОГО ОТСЧЕТА ****
024 ****
025 sample_ready:
026

```

<sup>1)</sup> Перед выполнением основной программы должны быть определены следующие объекты: NR\_COEF — число коэффициентов ядра фильтра (в нашем примере 301); dline[NR\_COEF] — циклический буфер, в который записываются последние входные отсчеты (память данных); coef[NR\_COEF] — циклический буфер, содержащий коэффициенты фильтра (память программ).

```

027 /* СЧИТЫВАНИЕ НОВОГО ОТСЧЁТА, ПЕРЕВОД ЕГО В ФОРМАТ С ПЛАВАЮЩЕЙ ТОЧКОЙ */
028
029 r0 = dm(rx_buf + 1); /* запись входного отсчёта в регистр r0 */
030 r0 = lshift r0 by 16; /* сдвиг влево на 16 бит с сохранением знака */
031 r1 = -31; /* установка масштаба преобразования */
032 f0 = float r0 by r1; /* выполняем преобразование в формат с плавающей точкой */
033 dm(i0,m0) = f0; /* запись нового отсчёта в буфер dline[ ] и обнуление f12 */
034
035 /* ВЫЧИСЛЕНИЕ ВЫХОДНОГО ОТСЧЁТА КИХ-ФИЛЬТРА */
036
037 f12 = 0;
038 f2 = dm(i0,m0), f4 = pm(i8,m8); /* загрузка регистров */
039 f8 = f2*f4, f2 = dm(i0,m0), f4 = pm(i8,m8);
040 /* основной цикл */
041 lcntr = NR_COEF-2, do (pc,1) until lce;
042 f8 = f2*f4, f12 = f8+f12, f2 = dm(i0,m0), f4 = pm(i8,m8);
043
044 f8 = f2*f4, f12 = f8+f12; /* окончание последней итерации */
045 f12 = f8+f12;
046
047 /* ПРЕОБРАЗОВАНИЕ ВЫХОДНОГО ОТСЧЁТА В ФОРМАТ С ФИКСИРОВАННОЙ ТОЧКОЙ
   /* И ВЫВОД РЕЗУЛЬТАТА */
048
049 r1 = 31; /* установка масштаба преобразования */
050 r8 = fix f12 by r1; /* переход от плавающей точки к фиксированной */
051 rti(db); /* возврат из прерывания с предварительным исполнением
   /* следующих двух строк*/
052 r8 = lshift r8 by -16; /* сдвиг вправо на 16 бит*/
053 dm(tx_buf + 1) = r8; /* вывод полученного отсчёта */

```

Во-вторых, в главной программе в строках 15...19 организуется бесконечный цикл ожидания прерывания от устройства ввода, указывающего, что поступил новый входной отсчёт. Обработка в рассматриваемом примере осуществляется поотсчётно. Со входа считывается один отсчёт и рассчитывается один отсчёт выходного сигнала. Большинство алгоритмов, осуществляющих обработку во временной области, таких как КИХ- или БИХ-фильтрация, работают именно так. Альтернативой является обработка по кадрам, которая преимущественно используется при обработке в частотной области. При этом на обработку поступает сразу группа входных отсчётов (кадр), для них выполняются вычисления и формируется кадр отсчётов выходного сигнала.

Подпрограмма, которая обслуживает прерывание по готовности входного отсчёта, разбита на три части. В первой части (строки 27...33) новый отсчёт считывается с кодека как число в формате с фиксированной точкой и осуществляется преобразование этого числа в формат с плавающей точкой. В языке Ассемблер процессоров SHARC регистры данных, хранящие числа в формате с фиксированной точкой, принято обозначать буквой «г» (r0, r8, r15), а регистры, хранящие значения в формате с плавающей точкой, — буквой «f» (f0, f8, f15). Поэтому, например, строку 32 следует интерпретировать так: число в формате с фиксированной точкой из регистра данных 0 (т. е. r0) преобразовать в число в формате с пла-

вающей точкой и записать его в тот же регистр данных 0 (но уже f0). Это преобразование реализуется в соответствии с масштабом, задаваемым целым числом, записанным в регистр r1. В третьей части (строки 47...53) осуществляются обратные шаги: значение отсчёта выходного сигнала преобразуется из формата с плавающей точкой в формат с фиксированной точкой и выдаётся на кодек.

Вторая часть, которую мы пропустили, образует ядро программы — процедуру свёртки, преобразующую входные отсчёты сигнала в выходные (строки 35...45). Все вычисления выполняются в формате с плавающей точкой, исключая возможность переполнений и снимая необходимость в нормировке чисел. Как было описано в предыдущей главе, эта часть кода оптимизирована так, чтобы максимально использовать достоинства архитектуры ЦСП SHARC, состоящее в способности выполнять несколько команд за один такт.

Итак, мы разработали программу КИХ-фильтрации на Ассемблере и рассчитали ядро требуемого фильтра. Таким образом, мы готовы сформировать программное обеспечение, которое может быть выполнено на ЦСП. Чтобы это осуществить, необходимо запустить программы *компиляции, ассемблирования и компоновки (линковки)* — три программы, поставляемые вместе с EZ-KIT Lite. Компилятор переводит программу с языка Си на Ассемблер процессора SHARC. Если вы написали программу непосредственно на Ассемблере, как в рассматриваемом примере, то этап компиляции может быть опущен. Ассемблирование и компоновка преобразуют разработанную программу и внешние подключаемые файлы (такие как файл конфигурации процессора, программа инициализации кодека, файл коэффициентов ядра фильтра и т. п.) в конечный исполняемый код. Всё это занимает около 30 секунд с окончательным результатом в виде программы SHARC, размещенной на жёстком диске вашего ПК. Специальный интерфейс между EZ-KIT Lite и ПК используется затем, чтобы перенести и выполнить программу непосредственно на плате. Достаточно просто «кликнуть» кнопкой мыши на нужном файле с исполняемым кодом, и вся работа по загрузке программы и её запуске будет выполнена автоматически.

Здесь возникают два вопроса: как можно протестировать систему, чтобы быть уверенными, что наш фильтр аудиосигналов работает корректно, и почему компания Analog Devices выпускает **цифровые** сигнальные процессоры?

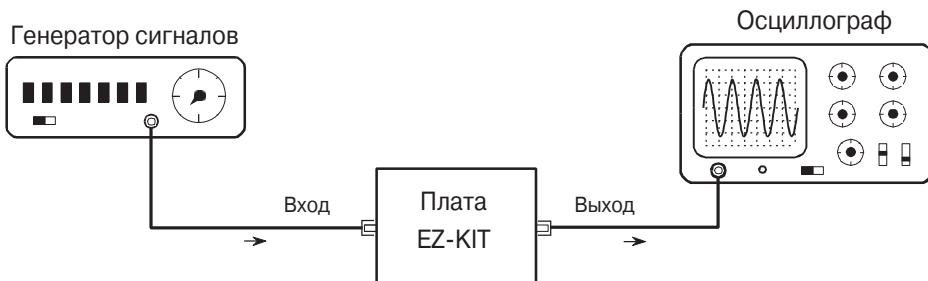
## 29.4. Аналоговые измерения с помощью цифровых систем

Забудьте на мгновение, что вы изучаете цифровые технологии. Давайте посмотрим на те же вопросы с точки зрения инженера, специализирующегося на аналоговой технике. Его не беспокоит, что находится внутри платы EZ-KIT Lite. Для него есть только аналоговый вход и аналоговый выход. Как проиллюстрировано на Рис. 29.3, он будет использовать традиционные аналоговые методы анализа системы, представляющей собой «чёрный ящик»: подключит генератор сигналов на вход и будет наблюдать состояние выхода системы с помощью осциллографа.

Что же увидит этот «аналоговый эксперт»? Во-первых, то, что *система линейна* (по крайней мере, в рамках того, что может сказать такой простой тест). Если

амплитуда или частота на входе изменяются, соответствующие изменения видны и на выходе. Если входная частота медленно увеличивается, то возникает момент, в который амплитуда выходной синусоидальной волны начинает быстро падать до нуля. Это происходит в непосредственной близости от точки половины частоты дискретизации и является следствием действия антиэлайзингового фильтра, расположенного в АЦП.

Далее наш эксперт замечает нечто нехарактерное для аналоговых устройств: оказывается, система имеет строго линейную *фазовую характеристику*. То есть задержка между событием на входе и реакцией на выходе системы всё время остаётся постоянной. Рассмотрим, например, импульсную характеристику нашего фильтра, представленную на Рис. 29.2. Так как центром симметрии является отсчёт с номером 150, то выходной сигнал будет задержан относительно входного на 150 дискретных отсчётов. Если частота дискретизации составляет, например, 8 кГц, то эта задержка будет равна 18.75 миллисекунды. Небольшую дополнительную задержку будет давать также сигма-дельта конвертер.

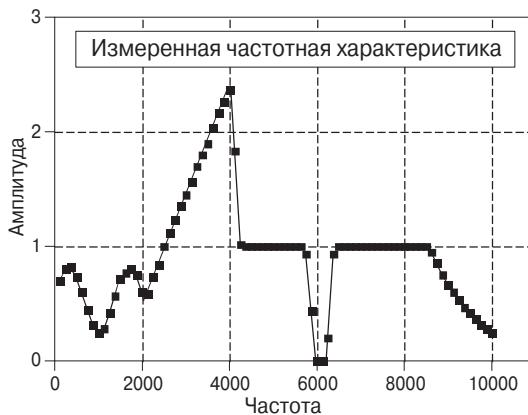


**Рис. 29.3.** Тестирование платы EZ-KIT Lite. Специалисты по аналоговым устройствам анализируют неизвестную им систему, подключая генератор сигналов к входу и осциллограф — к выходу. Если цифровую систему (в частности, EZ-KIT Lite) тестировать таким образом, то она будет выглядеть фактически как аналоговая система.

Специалист, имеющий дело с аналоговыми системами, будет обескуражен этим фактом. Сигналы на выходе ведут себя не так, как должны! Он начнёт нажимать на кнопки, переключать тумблеры, утверждать, что синхронизация выполнена неправильно, и бормотать: «Это не имеет смысла», «Что произошло?», «Кто трогал мой осциллограф?» Характеристики ЦОС-систем настолько хороши, что ему потребуется несколько минут, чтобы понять, что он видит.

Чтобы произвести на нашего специалиста ещё большее впечатление, мы попросим его измерить «вручную» частотную характеристику системы. Чтобы сделать это, он будет настраивать генератор сигналов последовательно на все частоты — от 125 Гц до 10 кГц с шагом 125 Гц и на каждой частоте измерять амплитуду выходного сигнала и вычислять её отношение к амплитуде входного сигнала (при этом удобнее всего оставлять амплитуду входного сигнала постоянной). Для нашего эксперимента мы установим частоту дискретизации на плате EZ-KIT Lite, равной 22 кГц. Это будет означать, что приведённым частотам, лежащим в диапазоне 0...0.5 (Рис. 29.2а), будут соответствовать реальные частоты в диапазоне 0...11 кГц.

**Рис. 29.4** показывает результат измерений, проведённых на EZ-KIT Lite. Великолепный результат! Точки измеренной кривой полностью согласуются с формой теоретически рассчитанной АЧХ с точностью до ошибки измерения. Такого наш



**Рис. 29.4.** Измерение частотной характеристики. Этот график показывает измеренные точки частотной характеристики КИХ-фильтра из рассматриваемого примера. Измеренные точки имеют значительно меньшую точность, чем рассчитанная при проектировании частотная характеристика Рис. 29.2а.

эксперт в области аналоговых систем никогда не видел, работая только с фильтрами, построенными на резисторах, конденсаторах и катушках индуктивности.

Однако это ещё не всё, на что способны ЦСП-системы. Для аналоговых измерений с использованием осциллографа или вольтметра характерна точность порядка 0.1...1.0%. Для сравнения: наша ЦСП-система ограничена лишь 0.001%-й ошибкой округления 16-битного кодека, при условии, что внутренние вычисления осуществляются в формате с плавающей точкой. Другими словами, оцениваемое устройство в 1000 раз точнее, чем используемые измерительные средства. Более точное измерение частотной характеристики требует использования специальных средств, таких как компьютеризированная система сбора данных с 20-битным АЦП. Поэтому неудивительно, что ЦСП часто используются в измерительных приборах для достижения высокой точности.

Теперь мы можем ответить на вопрос, почему фирма Analog Devices производит цифровые сигнальные процессоры. Если полтора-два десятилетия назад передовые методы обработки сигналов основывались на использовании высокоточных операционных усилителей и транзисторов, то сегодня в подавляющем большинстве случаев аналоговая обработка наивысшего качества выполняется с помощью цифровой техники. Analog Devices может служить примером для других фирм и частных лиц: будьте верны своим принципам и целям, но не бойтесь адаптироваться к изменениям перспективных технологий.

## 29.5. К вопросу о выборе фиксированной или плавающей точки

В рассмотренном выше примере мы использовали одну из ключевых особенностей процессоров семейства SHARC — способность осуществлять вычисления в *формате с плавающей точкой*. Хотя отсчёты входного и выходного сигналов поступают с кодека и на кодек в *формате с фиксированной точкой*, при реализации

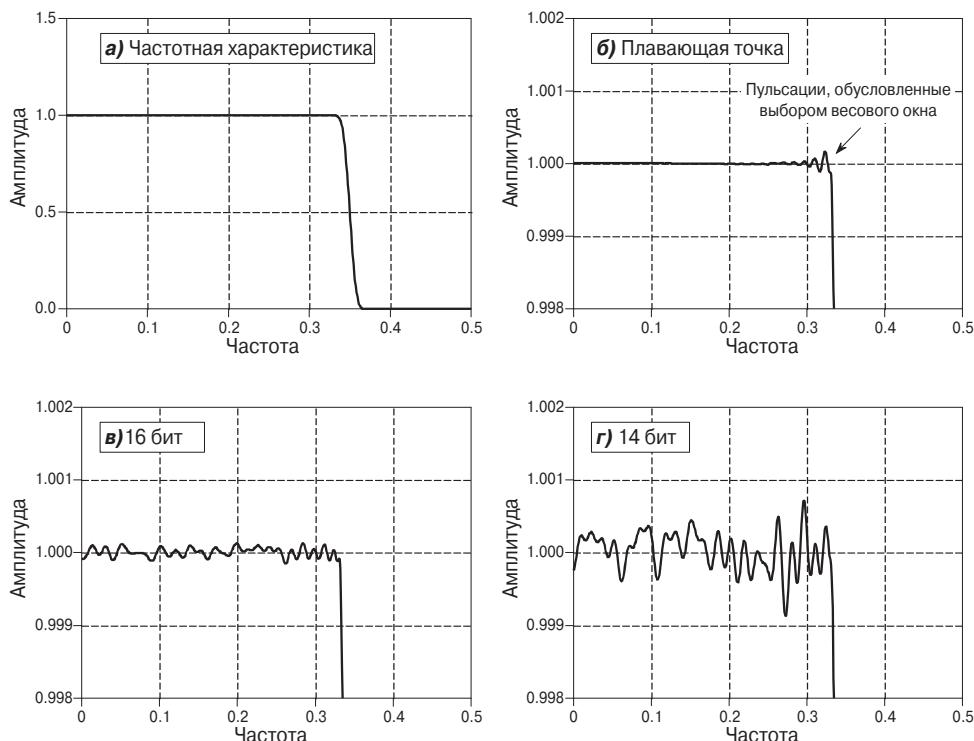
самого алгоритма обработки — КИХ-фильтрации — мы преобразуем эти сигналы к формату с плавающей точкой. Как было сказано в предыдущей главе, есть две причины, по которым обработка данных в формате с плавающей точкой оказывается предпочтительнее: удобство программирования и достижимая точность. Но так ли существенны различия двух форматов?

Для программиста — да, различия просто огромны. Программу для процессора с плавающей точкой написать значительно проще. Вернитесь к ассемблерному коду **Программы 29.1**. Основная процедура фильтрации реализуется здесь всего в двух строках (41 и 42). Если же вы пишете программу для ЦСП с фиксированной точкой, то придётся добавлять команды управления данными, сопровождающие каждое математическое действие. Чтобы избежать переполнений или потери значимости, все числа необходимо постоянно проверять на соответствие допустимому диапазону значений и при необходимости масштабировать их. Кроме того, промежуточные результаты необходимо хранить в аккумуляторах с повышенной точностью, чтобы избежать разрушительного эффекта накопления ошибок округления.

Достоинство повышенной точности формата с плавающей точкой гораздо менее очевидно. Взгляните, например, на **Рис. 29.5а**, на котором приведена амплитудно-частотная характеристика низкочастотного КИХ-фильтра со средней крутизной спада (см. Главу 16). При рассмотрении в таком масштабе данная кривая будет выглядеть одинаково как для плавающей, так и для фиксированной точки. Чтобы почувствовать разницу между этими двумя форматами, мы должны увеличить масштаб отображения в несколько сот раз, как показано на **(б...г)**. Теперь различия очевидны. При использовании формата с плавающей точкой шум округления становится настолько мал, что эффективность фильтра оказывается ограниченной только методикой проектирования фильтра. Наличие 0,02%-го перегулирования в начале переходной зоны обусловлено характеристиками весового окна Блэкмана, использованного при проектировании фильтра. То есть, если мы хотим улучшить характеристики фильтра, нам нужно обращать внимание на сам алгоритм, а не на его реализацию. Кривые **(в и г)** демонстрируют шумы округления, характерные для 16- и 14-битного представления каждого коэффициента фильтра соответственно. Повышение качества алгоритма в этом случае никак не сможет улучшить ситуацию. Действительная форма частотной характеристики просто «утонула» в шумах.

**Рис. 29.6** иллюстрирует различия представления сигналов во временной области в формате с фиксированной и с плавающей точкой. На **(а)** показан пример гармонического колебания с экспоненциально уменьшающейся амплитудой. Это может быть, например, звук колеблющейся струны или колебания земной поверхности, вызванные удалённым взрывом. Как и в рассмотренном выше случае, если масштаб отображения не очень велик, сигналы, представленные в формате с фиксированной и плавающей точкой, выглядят одинаково. Чтобы увидеть различие между ними, требуется увеличить масштаб, что показано на **(б...г)**. Становятся видны эффекты квантования, о которых шла речь в Главе 3, состоящие в добавлении к сигналу аддитивного шума, ограничивающего различимость слабых компонентов сигнала.

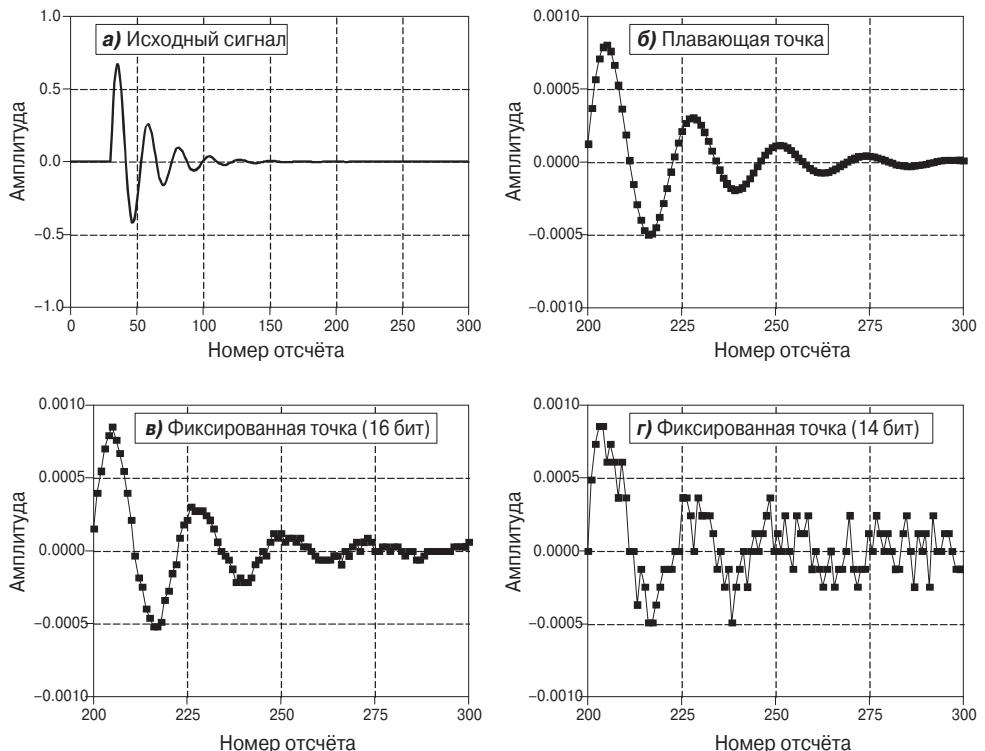
Эти различия между форматами с фиксированной и с плавающей точкой часто бывают несущественны. Например, на графиках без масштабирования (**Рис. 29.5а** и **Рис. 29.6а**) их даже незаметно. Однако в ряде приложений возмож-



**Рис. 29.5.** Шумы округления в частотной характеристике. На (а) показана частотная характеристика низкочастотного оконного фильтра, использующего весовую функцию Блэкмана и имеющего порядок, равный 150. На (б...г) эта же кривая показана более детально. Если ядро фильтра представлено в формате с плавающей точкой (б), шум округления незначителен по сравнению с неравномерностями, вносимыми в АЧХ фильтра оконной функцией. Как показано на (в и г), представление ядра фильтра в формате с фиксированной точкой делает шум округления доминирующим, определяющим неравномерность формы АЧХ.

ности более точного представления чисел формата с плавающей точкой оказываются очень полезны или даже просто необходимы. Например, качественные аудиосистемы широкого потребления, такие как CD-плееры, используют 16-битное представление сигналов в формате с фиксированной точкой. В большинстве случаев это превышает способности человеческого слуха. Однако профессиональные аудиосистемы наивысшего качества представляют сигналы с 20- и 24-битной точностью, не оставляя никаких шансов шумам, способным исказить музыку. Формат с плавающей точкой практически идеально подходит для алгоритмов обработки подобных цифровых сигналов повышенной точности.

Другой ситуацией, в которой необходимо использовать точность представления сигналов в формате с плавающей точкой, является случай особой чувствительности алгоритма к шумам. Например, КИХ-фильтры очень устойчивы к эффектам округления. Как показано на Рис. 29.5, наложение шума округления в целом не изменяет формы частотной характеристики; она просто становится более неравномерной. С БИХ-фильтрами ситуация совершенно иная. Округление может вызвать самые разные искажения, включая потерю устойчивости фильтра.



**Рис. 29.6.** Шумы округления во временной области. На (а) показан исходный сигнал — гармоническое колебание с экспоненциально затухающей амплитудой. На (б...г) дают более детальное изображение за счёт укрупнения масштаба по оси ординат. При представлении сигнала в формате с плавающей точкой (б) шум округления так мал, что на графике его не видно. В то же время при представлении сигнала в формате с фиксированной точкой (в и г) уровень шумов округления оказывается достаточно велик.

Плавающая точка существенно расширяет возможности подобных алгоритмов: крутизна спада частотной характеристики может быть значительно выше, затухание в полосе подавления — больше, а перерегулирование переходной характеристики — меньше.

## 29.6. Более мощные программные инструменты

Рассмотренный нами пример реализации специального фильтра показывает самый простой путь получения программ, исполняемых на ЦСП SHARC: необходимо просто пройти этапы редактирования, ассемблирования, компоновки и загрузки, каждый из которых выполняется отдельной программой. Этот метод прекрасен для простых задач, однако есть более мощные программные инструментальные средства, используемые опытными программистами. Давайте посмотрим, какие инструменты будет полезно использовать, если вы захотите решать с помощью ЦСП серьёзные задачи.

Таблица 29.2. Основные функции библиотеки языка Си

Математические функции		Манипуляции строками и символами	
abs	абсолютное значение	atoi	преобразовать строку в целое
acos	арккосинус	bsearch	поиск в двоичном массиве
asin	арксинус	isalnum	обнаружить букву или цифру
atan	арктангенс	isalpha	обнаружить букву
atan2	арктангенс отношения	iscntrl	обнаружить управляющий символ
cabsf	комплексное абсолютное значение	isdigit	обнаружить десятичную цифру
cexpf	комплексная экспонента	isgraph	обнаружить печатаемый символ
cos	косинус	islower	обнаружить символ нижнего регистра
cosh	косинус гиперболический	isprint	обнаружить печатаемый символ
cot	котангенс	ispunct	обнаружить символ пунктуации
div	деление	isspace	обнаружить пробел
exp	экспонента	isupper	обнаружить символ верхнего регистра
fmod	модуль	isxdigit	обнаружить шестнадцатеричную цифру
log	логарифм натуральный	memchr	найти первое появление символа
log10	логарифм десятичный	memcpy	копировать символ
matadd	сложение матриц	streat	конкатенация строк
matmul	умножение матриц	strcmp	сравнить строки
pow	возвведение в степень	strerror	получить сообщение об ошибке
rand	генератор случайных чисел	strlen	длина строки
sin	синус	strncmp	сравнить символы
sinh	синус гиперболический	strchr	найти последнее появление символа
sqrt	квадратный корень	strstr	найти строку в строке
srand	начальное случайное число	strtok	разбить строку на части
tan	тангенс	system	передать строку операционной системе
tanh	тангенс гиперболический	tolower	преобразовать верхний регистр в нижний
		toupper	преобразовать нижний регистр в верхний

Управление программой	
abort	аномальное завершение программы
calloc	выделить/инициализировать память
free	освободить память
idle	команда ожидания процессора
interrupt	функции обработки прерываний
poll_flag_in	проверить входной флаг
set_flag	установка флагов процессора
timer_off	отключить таймер
timer_on	включить таймер
timer_set	инициализировать таймер

Обработка сигнала	
a_compress	А-закон сжатия
a_expand	А-закон восстановления
autocorr	автокорреляция
biquad	секция биквадратного фильтра
cfftN	комплексное БПФ
crosscorr	взаимная корреляция
fir	КИХ-фильтр
histogram	гистограмма
ifftN	обратное комплексное БПФ
iir	БИХ-фильтр
mean	среднее в массиве
mu_compress	μ-закон сжатия
mu_expand	μ-закон восстановления
rfftN	действительное БПФ
rms	среднеквадратическое значение массива

Прежде всего это *компилятор* языка Си. Как говорилось в предыдущей главе, язык Си используется для программирования ЦСП наряду с Ассемблером. При этом существенным достоинством языка Си является наличие библиотеки стандартных функций и функций, реализующих базовые алгоритмы ЦОС. В Табл. 29.2 приведён частичный список библиотечных функций языка Си, используемых для программирования ЦСП семейства SHARC. Набор математических функций включает тригонометрические функции (синус, косинус, тангенс и т. д.), логарифм и экспоненту, вычисление которых в алгоритмах ЦОС используется достаточно часто. Если некоторые из этих функций оказались необходимы при реализации вашего проекта, то это может уже быть достаточным основанием для того, чтобы использовать Си, а не Ассемблер. В Табл. 29.2 особое место занимают функции обработки сигналов. Среди них вы найдёте базовые алгоритмы, используемые в ЦОС, включая действительное и комплексное БПФ, КИХ- и БИХ-фильтрации и функции статистики — вычисление среднего и среднеквадратического значений и дисперсии. Все эти функции написаны конечно же на Ассемблере, что делает их очень эффективными с точки зрения скорости выполнения и требуемых затрат памяти.

**Программа 29.2** представляет собой пример программы на языке Си, взятый из руководства пользователя по компилятору языка Си для процессоров семейства SHARC фирмы Analog Devices. Эта программа генерирует эхо, прибавляя к входному сигналу его задержанную копию. Последние 4000 отсчётов входного сигнала запоминаются в циклическом буфере. При поступлении каждого нового отсчёта циклический буфер обновляется. При этом самый новый отсчёт суммируется с умноженным на постоянный коэффициент самым старым отсчётом, и результат пересыпается на выход системы.

### Программа 29.2. Пример программы на языке Си

```
/* CIRCBUF.C */
/* Это программа генерации эха, написанная на Си для ADSP-21061 EZ-KIT Lite */
/* Эта программа использует базовую программу прямой передачи входных данных */
/* (talkthrough) на выход и дополнительно использует схему циклической буферизации */
/* Циклический буфер инициализируется функциями CIRCULAR_BUFFER, BASE и LENGTH */
/* Эхо формируется добавлением к текущему входному отсчёту самого старого */
/* записанного в циклический буфер входного отсчёта */
/* Задержка эха может регулироваться значением параметра BUFF_LENGTH */
*/
#include <21020.h> /* для команды idle() */
#include <signal.h> /* для команды interrupt */
#include <macros.h> /* для функций CIRCULAR_BUFFER() и segment() */

#define BUFF_LENGTH 4000

CIRCULAR_BUFFER (float,1,echo) /* инициализирует переменную echo,
                                 * как указатель на */
/* циклический буфер в памяти данных. Значение указателя хранится в регистре i1 */
/* генератора адреса данных DAG1 */

volatile float in_port segment (hip_reg0); /* объекты hip_reg0 и hip-reg2 */
volatile float out_port segment (hip_reg2); /* используются в файле архитектуры */
```

```

void process_input (int);
void main (void)

{/* Длину циклического буфера можно менять. Если остановить выполнение программы в */
/* теле функции main и в окне отображения памяти dm изменить параметр _BUFF_LENGTH */
/* то задержка эха изменится. Но не делайте BUFF_LENGTH больше, чем размер стека! */

float data_buff [BUFF_LENGTH];
interrupt (SIG IRQ3, process_input);
BASE (echo) = data_buff; /* Загружает в b1 и i1 начальный адрес буфера */
LENGTH (echo) = BUFF_LENGTH; /* Загружает в L1 длину буфера */
/* когда циклический буфер полностью заполнен, n-й его элемент содержит самый */
/* новый отсчёт, а (n + 1)-й элемент содержит самый старый отсчёт */

while (1)
{
float oldest, newest; /* Добавление эха означает суммирование самого нового */
idle(); /* и самого старого отсчётов */

/* После обработки прерывания переменная echo указывает на n-й элемент */
/* циклического буфера. Новый отсчёт записывается в переменную 'newest'. */
/* Затем указатель увеличивается на единицу и переходит к элементу n + 1 */

CIRC_READ (echo, 1, newest, dm);

/* Теперь переменная echo указывает на (n + 1)-й элемент. Происходит чтение */
/* переменной 'oldest'. Указатель не изменяется, так как теперь он указывает */
/* на элемент, в который после следующего прерывания будет записан новый отсчёт */

CIR_READ (echo, 0, oldest, dm);

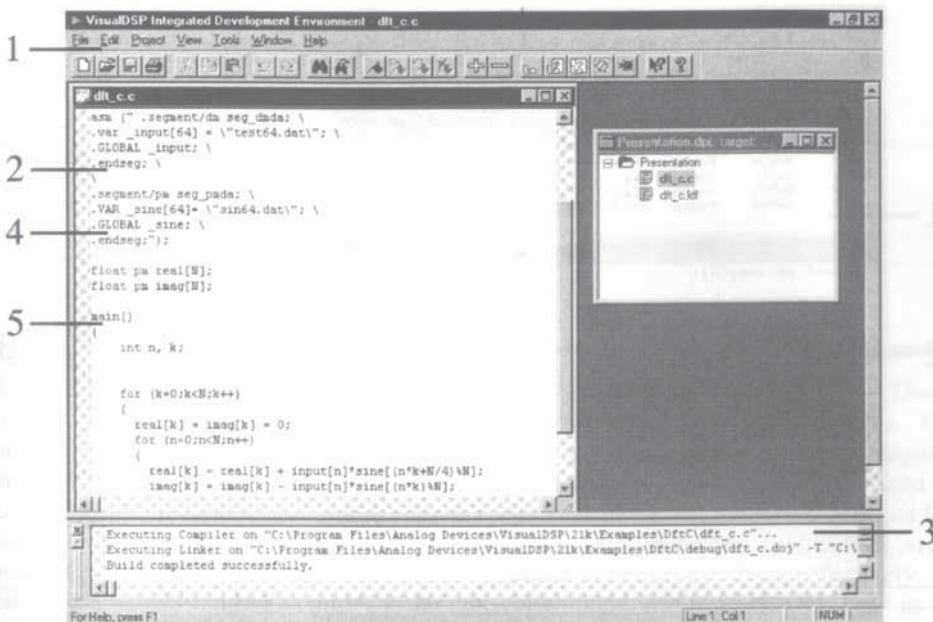
/* Суммируем самый старый и самый новый отсчёты и результат отправляем на выход */
out_port=oldest+newest;
}

void process_input (int int_number)
{
/* Новый входной отсчёт записывается в циклический буфер на место самого старого */
/* отсчёта в n-й элемент. Указатель не изменяется */

CIRC_WRITE (echo, 0, in_port, dm);
}

```

Следующий программный инструмент ещё более высокого уровня, который широко используется при работе с ЦСП, — это *интегрированная среда разработки*. Таким причудливым термином обозначается программная среда, внутри которой объединено все, что необходимо для программирования и отладки работы ЦСП. Фирма Analog Devices обеспечивает поддержку своих процессоров интегрированной средой разработки **VisualDSP®**, работающей под операционной системой Windows®. На Рис. 29.7 показан пример главного рабочего окна среды VisualDSP, обеспечивающего в едином интерфейсе средства для редактирования, построения и отладки проектов.

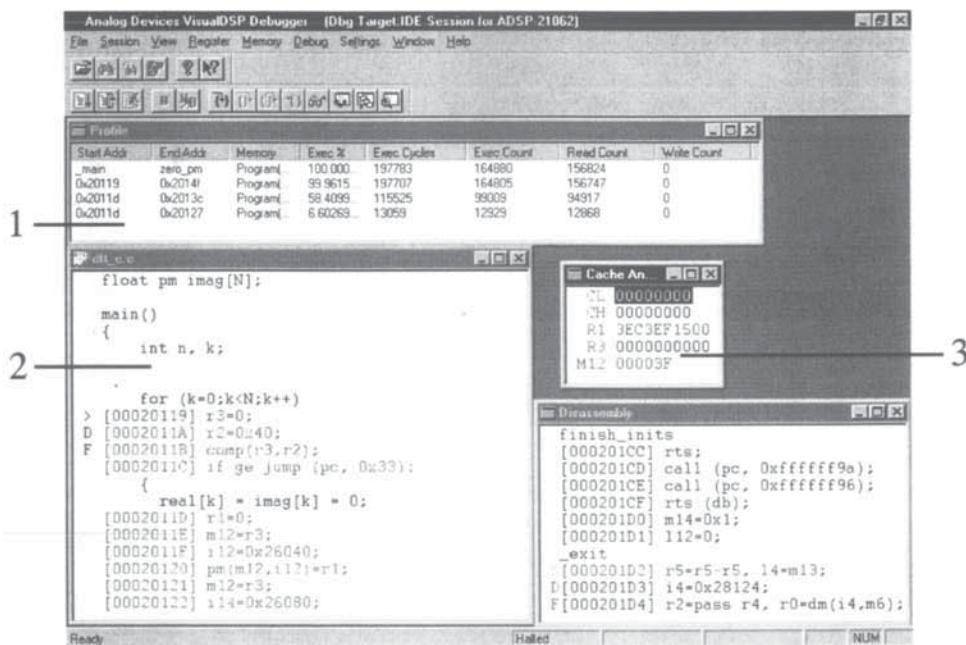


1. Простой переход от редактирования к построению проекта и отладке.
2. Смесь Ассемблера и Си в одном общем исходном файле.
3. Отображение результатов построения.
4. Мощный редактор «понимает» синтаксис языков.
5. Удобство переходов через систему закладок.

**Рис. 29.7.** Пример рабочего окна среды VisualDSP. Это интегрированная среда разработки, используемая для создания программ для процессоров SHARC. В рамках единого интерфейса могут быть доступны следующие функции: редактирование, компиляция, ассемблирование, компоновка, симуляция, отладка, загрузка в процессор и прошивка ПЗУ.

Здесь имеется ряд ключевых, характерных для VisualDSP моментов, иллюстрирующих огромную важность интегрированной среды для ускорения процесса разработки программного обеспечения. Редактор программ специализирован для написания кодов и на языке Си, и на Ассемблере или их комбинации. Например, он «понимает» синтаксис языков и отображает различные типы операторов разным цветом. Есть также возможность работы более чем с одним файлом. Это очень удобно. Выходной код формируется компоновкой всех файлов проекта.

На **Рис. 29.8** показан пример окна среды VisualDSP в процессе отладки кода. Отладчик VisualDSP непосредственно связан ещё с двумя инструментальными средствами разработки: симулятором и эмулятором. *Симулятор* позволяет провести программные коды на персональном компьютере. При этом в наличии самого ЦСП нет никакой необходимости. Это обычно первый шаг отладки программы после её написания. Симулятор создает модель архитектуры ЦСП и имитирует её функционирование, включая процессы ввода потоков данных, возникновения и обслуживания прерываний и т. д. *Эмуляторы* (например, EZ-ICE® фирмы Analog Devices) дают возможность проверить работу программы на реальном оборудовании. Для этого необходимо, чтобы программное обеспечение эмулятора, установленное на ПК, было способно контролировать электрические сигналы внутри ЦСП. Для обеспечения такой возможности процессоры семейства SHARC вклю-



1. Статистика временных затрат (*profiling*), позволяющая выявить «узкие места» программы.
2. Совместное отображение листингов программы на языках Си и Ассемблер.
3. Создание пользовательского окна регистров.

**Рис. 29.8.** Окно отладчика среды VisualDSP. Это общий интерфейс, характерный и для симулятора, и для эмулятора. Он позволяет отображать совместно текст программы на языке Си и генерированный на её основе ассемблерный код; показывать ход выполнения команд; отображать содержимое регистров (аппаратных, программных и сформированных в памяти); следить за активностью использования шин и решать многие другие задачи.

чают в свой состав порт контрольного доступа IEEE 1140.1, позволяющий внешнему устройству следить за процессами, происходящими внутри ЦСП.

После того как вы ознакомились с оценочным комплектом и пришли к мысли о необходимости приобретения современных инструментов программного обеспечения, вам следует также рассмотреть возможность обучения работе с ЦСП на специальных тренировочных курсах. Такие программы обучения предусмотрены многими производителями ЦСП. Например, фирма Analog Devices предлагает 3-дневные занятия, проводимые ею несколько раз в год в различных городах и странах. Это уникальная возможность узнать о процессоре от экспертов. Чтобы узнать подробнее о таких программах, заходите на сайты производителей.

# Глава 30

## КОМПЛЕКСНЫЕ ЧИСЛА

*Комплексные числа* образуют более широкое множество чисел по сравнению с вещественными (действительными) числами, которыми мы привыкли пользоваться. Уникальным свойством комплексных чисел является способность представлять две переменные и обращаться с ними как с единым целым. Такая особенность прекрасно подходит для гармонического анализа (анализа Фурье), в котором сигналы в частотной области представляются состоящими из двух компонент — вещественной и мнимой. Комплексные числа позволяют упростить запись уравнений, используемых в ЦОС. Кроме того, появляется возможность применять методы, которые сложно или вообще невозможно реализовать с помощью одних только вещественных чисел; например, комплексные числа лежат в основе алгоритма быстрого преобразования Фурье. К сожалению, применение комплексных методов требует более углублённого знания математики. Перед тем как применять их на практике, необходимо потратить немало сил на их изучение и получить определённый навык использования. Условно можно сказать, что через комплексные числа «проходит граница», отделяющая тех, кто использует ЦОС как вспомогательный инструмент, от тех, для кого использование ЦОС — это профессия. В этой главе мы рассмотрим математические действия с комплексными числами, а также простейшие способы применения этих чисел в науке и технике. В последующих трёх главах обсуждаются важные методы, основанные на комплексных числах: комплексное преобразование Фурье, преобразование Лапласа и Z-преобразование. Эти комплексные преобразования легли в основу теории ЦОС. Готовьтесь — начинается математика!

### 30.1. Основные понятия

Что такое *комплексные числа*, поясним на примере с ребёнком, подбрасывающим мяч в воздух. Предположим, что мяч летит строго вертикально с начальной скоростью 9.8 м/с. Двигаясь с ускорением свободного падения, через секунду после броска мяч достигнет высоты 4.9 метра, а его скорость станет равна нулю. Затем мяч начнёт двигаться вниз с тем же ускорением и, спустя две секунды после момента броска, снова вернётся в руки ребенка. Уравнение движения мяча при этом выглядит следующим образом:

$$h = \frac{-gt^2}{2} + v_0 t,$$

где  $h$  — высота над поверхностью земли (м),  $g$  — ускорение свободного падения ( $9.8 \text{ м/с}^2$ ),  $v_0$  — начальная скорость ( $9.8 \text{ м/с}$ ),  $t$  — время (с).

Теперь предположим, что нужно узнать, в какой момент времени мяч находился на заданной высоте. Подставив известные величины, выразим  $t$ :

$$t = 1 \pm \sqrt{1 - h / 4.9}.$$

Например, на высоте трёх метров мяч оказывается дважды:  $t_1 = 0.38$  с (при подъёме) и  $t_2 = 1.62$  с (при падении).

До тех пор, пока мы задаём высоту в разумных пределах, уравнение движения даёт разумные ответы. Но что произойдёт, если мы превысим этот предел? Например, попробуем узнать, в какой момент времени мяч достигнет высоты 10 метров? Ясно, что на этот вопрос ответить нельзя, поскольку мяч никогда не поднимется на такую высоту. Тем не менее подстановка значения  $h = 10$  в представленное выше уравнение приводит к двум результатам:

$$t_1 = 1 + \sqrt{-1.041},$$

$$t_2 = 1 - \sqrt{-1.041}.$$

В обоих равенствах присутствует квадратный корень из отрицательного числа, чего, как известно, в реальном мире просто не может быть. Это необычное свойство полиномиальных уравнений было впервые использовано итальянским математиком Джироламо Кардано (1501–1576). Двумя веками позже великий немецкий математик Карл Фридрих Гаусс (1777–1855) ввёл термин «комплексные числа» и проложил путь к современному их пониманию.

Любое комплексное число есть сумма двух составляющих — *вещественной части* и *мнимой части*. Вещественная часть комплексного числа представляет собой обычное *вещественное число*. Такие числа проходят в школе. Мнимая часть — это мнимое число, определяемое как квадратный корень из отрицательного вещественного числа. В стандартной форме записи комплексных чисел мнимая часть представляется произведением обычного (вещественного) числа на корень из минус единицы. Например, комплексное число

$$t = 1 + \sqrt{-1.041}$$

сначала следует преобразовать к виду

$$t = 1 + \sqrt{1.041} \sqrt{-1},$$

а затем записать в окончательной форме, принятой для комплексных чисел:

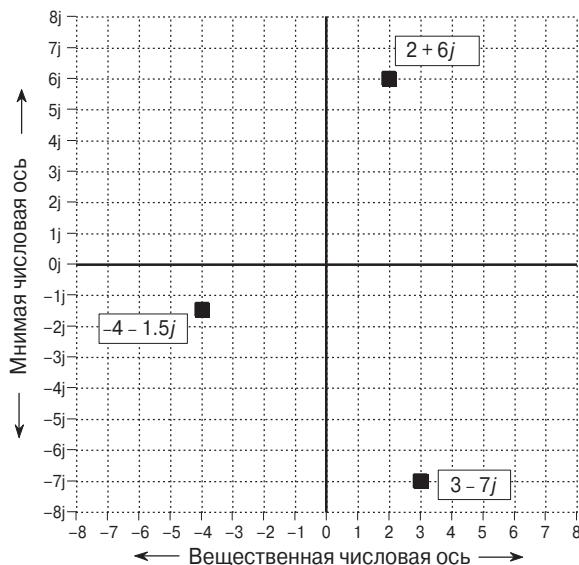
$$t = 1 + 1.02\sqrt{-1}.$$

Вещественная часть этого комплексного числа равна 1, а мнимая —  $1.02\sqrt{-1}$ .

Такая запись позволяет для обозначения абстрактного понятия  $\sqrt{-1}$  ввести специальный символ. У математиков укоренилась традиция использовать символ  $i$ , в то время как инженеры пользуются символом  $j$ , поскольку  $i$  они применяют для обозначения электрического тока. В ЦОС встречаются оба символа, но в этой книге мы будем пользоваться принятым в технике символом  $j$ .

Можно привести следующие примеры комплексных чисел:  $1 + 2j$ ;  $1 - 2j$ ;  $-1 + 2j$ ;  $3.14159 + 2.7183j$ ;  $(4/3) + (19/2)j$  и т. д. Любые обычные числа, такие как 2, 6.34 и  $-1.414$ , также можно рассматривать как комплексные числа с нулевой мнимой частью:  $2 + 0j$ ,  $6.34 + 0j$  и  $-1.414 + 0j$ .

Графическим отображением вещественных чисел является точка на числовой прямой; геометрической интерпретацией комплексных чисел являются точки двумерного пространства — *комплексной плоскости*. По горизонтальной оси откладывается вещественная часть комплексного числа, а по вертикальной — его мнимая часть (**Рис. 30.1**). Поскольку вещественные числа — это комплексные числа с нулевой мнимой частью, можно сказать, что числовая прямая вещественных чисел совпадает с осью абсцисс комплексной плоскости.



**Рис. 30.1.** Комплексная плоскость. Для примера произвольно выбраны три комплексных числа. Каждому из них на комплексной плоскости соответствует единственная точка. По горизонтальной оси откладывается вещественная часть, по вертикальной — мнимая.

В математических уравнениях комплексное число, хотя и состоит из двух частей, представляется как одна переменная. На **Рис. 30.1** изображены три значения комплексной переменной:

$$A = 2 + 6j,$$

$$B = -4 - 1.5j,$$

$$C = 3 - 7j.$$

Здесь проявляются как сильная, так и слабая сторона комплексных чисел. До-  
стоинством комплексных чисел является возможность учёта двух величин в од-  
ном символе. Недостаток состоит в том, что требуется постоянно помнить, какая  
переменная является комплексным числом, а какая — обычным (вещест-  
венным).

Для разделения комплексного числа на его вещественную и мнимую части используют математические операторы  $\operatorname{Re}(\ )$  и  $\operatorname{Im}(\ )$ . Например, для приведённых выше комплексных чисел можно записать:

$$\operatorname{Re} A = 2 \quad \operatorname{Im} A = 6$$

$$\operatorname{Re} B = -4 \quad \operatorname{Im} B = -1.5$$

$$\operatorname{Re} C = 3 \quad \operatorname{Im} C = -7$$

Обратите внимание на то, что значение, возвращаемое математическим оператором  $\operatorname{Im}(\ )$ , не содержит символа  $j$ :  $\operatorname{Im}(3 + 4j)$  равно 4, а не  $4j$ .

Для комплексных чисел вводятся те же алгебраические операции, что и для вещественных. Величина  $j$  рассматривается при этом как константа, поэтому сложение, вычитание, умножение и деление могут быть описаны следующими формулами.

*Сложение комплексных чисел:*

$$(a + bj) + (c + dj) = (a + c) + j(b + d) \quad (30.1)$$

*Вычитание комплексных чисел:*

$$(a + bj) - (c + dj) = (a - c) + j(b - d) \quad (30.2)$$

*Умножение комплексных чисел:*

$$(a + bj)(c + dj) = (ac - bd) + j(bc + ad) \quad (30.3)$$

*Деление комплексных чисел:*

$$\frac{(a + bj)}{(c + dj)} = \left( \frac{ac + bd}{c^2 + d^2} \right) + j \left( \frac{bc - ad}{c^2 + d^2} \right). \quad (30.4)$$

При решении подобных уравнений применяются две хитрости. Первая: как только встречается выражение  $j^2$ , его следует сразу заменить на  $-1$ . Равенство этих величин следует из определения мнимой единицы  $j$ :

$$j^2 = (\sqrt{-1})^2 = -1.$$

Вторая хитрость заключается в том, что из знаменателя дроби мнимую единицу всегда можно исключить, умножив числитель и знаменатель на сопряжённое знаменателю выражение. В левой части выражения (30.4) знаменатель дроби имеет вид  $(c + dj)$ . Умножив числитель и знаменатель на  $(c - dj)$ , получим дробь, в которой знаменатель — вещественное число. Два комплексных числа, отличающихся только арифметическим знаком мнимой части, называются *комплексно-сопряжёнными*. Комплексно-сопряжённую величину обозначают звёздочкой, которую располагают справа выше переменной. Например, если  $Z = a + bj$ , то  $Z^* = a - bj$ .

Перечисленные ниже свойства верны не только для вещественных, но и для комплексных чисел  $A, B, C$ . Справедливость этих равенств можно доказать, разделив переменные на вещественную и мнимую части и выполнив простые алгебраические преобразования.

*Свойство коммутативности:*

$$AB = BA. \quad (30.5)$$

*Свойство ассоциативности:*

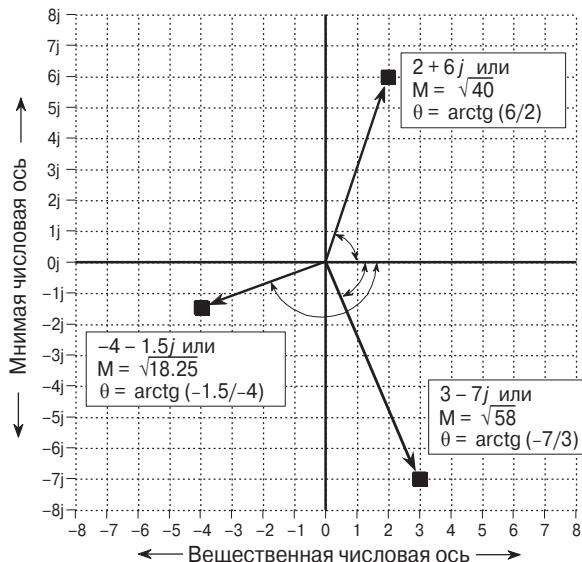
$$(A + B) + C = A + (B + C). \quad (30.6)$$

*Свойство дистрибутивности:*

$$A(B + C) = AB + AC. \quad (30.7)$$

## 30.2. Полярная форма записи комплексных чисел

Та форма записи комплексных чисел, в которой они вводились выше, называется алгебраической (или формой записи в прямоугольной системе координат). Теперь рассмотрим полярную форму записи комплексных чисел. На Рис. 30.2 показаны те же самые три точки, представляющие комплексные числа на комплексной плоскости. Модулем комплексного числа называется длина вектора, исходящего из начала координат и заканчивающегося в соответствующей этому числу точке. Фазой комплексного числа называется угол между данным вектором и положительным направлением оси абсцисс. Переход от одной формы записи комплексных чисел к другой выполняется по следующим правилам (обратите



**Рис. 30.2.** Полярная форма записи комплексных чисел. Три точки на комплексной плоскости представлены в полярной системе координат. Алгебраическая форма записи соответствует прямоугольной системе координат (Рис. 30.1).

внимание на те неприятности, связанные с полярной формой, о которых говорилось в Главе 8):

$$\begin{aligned} M &= \sqrt{(Re A)^2 + (Im A)^2}, \\ \theta &= \arctg\left(\frac{Im A}{Re A}\right). \end{aligned} \quad (30.8)$$

Переход от алгебраической формы записи комплексных чисел к полярной форме. Комплексную переменную  $A$  можно преобразовать из алгебраической формы  $\{Re A, Im A\}$  в полярную  $\{M, \theta\}$ .

$$\begin{aligned} Re A &= M \cos(\theta), \\ Im A &= M \sin(\theta). \end{aligned} \quad (30.9)$$

Переход от полярной формы записи комплексных чисел к алгебраической форме.  
Комплексное число преобразуется из  $\{M, \theta\}$  в  $\{Re A, Im A\}$ .

Эти преобразования играют большую роль в математике. Поэтому следует уделить им особое внимание. Комплексное число, записанное в алгебраической форме, имеет вид  $a + jb$ . Хотя информация содержится в переменных  $a$  и  $b$ , но комплексным числом называется всё выражение целиком. При полярной форме записи информация содержится в  $M$  и  $\theta$ . Встаёт вопрос: какое выражение объединяет модуль и фазу комплексного числа?

Ответ на поставленный вопрос может быть найден при анализе выражения (30.9). Начнём с того, что запишем алгебраическую форму комплексного числа в общем виде:  $a + jb$ , а затем выполним подстановку по правилам, определённым в выражении (30.9). В результате мы получим

$$a + jb = M(\cos \theta + j \sin \theta). \quad (30.10)$$

Алгебраическая и полярная формы записи комплексного числа. Слева число записано в алгебраической форме, справа — в полярной. Переход от одной формы к другой выполняется в соответствии с выражениями (30.8) и (30.9).

Выражение в левой части полученного уравнения представляет собой алгебраическую форму записи комплексного числа, а выражение в правой части — тоже число, но записанное в полярной форме.

Прежде чем пойти дальше, подведём итог сказанному ранее. Во-первых, мы дали графическое описание комплексного числа в алгебраической форме как некой точки на комплексной плоскости. Во-вторых, мы ввели два новых члена:  $M$  и  $\theta$ , которые связаны с алгебраической формой записи комплексных чисел такими же зависимостями, как полярная и прямоугольная системы координат связаны посредством правил (30.8) и (30.9). В-третьих, мы выявили математическую связь между двумя формами записи комплексных чисел, получив полярную форму записи вида  $M(\cos \theta + j \sin \theta)$ . Хотя логика подобных рассуждений проста и очевидна, такой результат труден для «интуитивного» восприятия. К сожалению, дальше будет сложнее.

Одним из самых важных уравнений в разделе математики, посвящённом комплексным числам, является *формула Эйлера*, названная в честь блестящего и творчески плодовитого швейцарского математика Леонарда Эйлера (1707–1783):

$$e^{jx} = \cos x + j \sin x. \quad (30.11)$$

Формула Эйлера. Основное уравнение для инженеров и ученых, пользующихся комплексными числами.

Если вам нравится заниматься доказательством математических формул, то рекомендуем воспользоваться разложением комплексной экспоненты в ряд Тейлора:

$$e^{jx} = \sum_{n=0}^{\infty} \frac{(jx)^n}{n!} = \left[ \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k}}{(2k)!} \right] + j \left[ \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!} \right].$$

Два члена выражения в правой части, заключенные в квадратные скобки, являются рядами Тейлора для функций  $\cos(x)$  и  $\sin(x)$ . Но не тратьте слишком много времени на доказательство: оно нам не пригодится.

Воспользовавшись формулой Эйлера, полярную форму записи комплексного числа выражения (30.10) можно представить в виде *комплексной экспоненты*:

$$a + jb = Me^{\theta}. \quad (30.12)$$

Полярная форма записи в виде комплексной экспоненты. Слева число записано в алгебраической форме, справа — выражено через комплексную экспоненту.

Экспоненциальная форма представления комплексных чисел — это основа математического аппарата ЦОС. Рекомендуем запомнить уравнения (30.8)...(30.12). Достоинством экспоненциальной формы записи является то, что она позволяет упростить операции умножения и деления комплексных чисел:

$$M_1 e^{j\theta_1} M_2 e^{j\theta_2} = M_1 M_2 e^{j(\theta_1 + \theta_2)}. \quad (30.13)$$

Умножение комплексных чисел.

$$\frac{M_1 e^{j\theta_1}}{M_2 e^{j\theta_2}} = \left[ \frac{M_1}{M_2} \right] e^{j(\theta_1 - \theta_2)}. \quad (30.14)$$

Деление комплексных чисел.

Таким образом, при умножении комплексных чисел их модули умножаются, а фазы складываются; при делении модули делятся, а фазы вычитаются. Операции сложения и вычитания комплексных чисел проще всего реализовать, преобразовав эти числа к алгебраической форме записи, выполнив соответствующую операцию и преобразовав результат обратно в полярную форму. В компьютерных программах комплексные числа обычно представляются в алгебраической форме, а в записях математических формул чаще всего в полярной форме. Подобно тому как  $Re()$  и  $Im()$  используются для извлечения двух составляющих алгебраической формы комплексного числа, операторы  $Mag()$  и  $Phase()$  используются для извлечения двух компонентов полярной формы записи. Например, если  $A = 5e^{j\pi/7}$ , то  $Mag(A) = 5$ , а  $Phase(A) = \pi/7$ .

### 30.2.1. Метод замещения комплексными числами

Подведём итог вышесказанному. Решение обычных алгебраических уравнений часто приводит к необходимости извлечения квадратного корня из отрицательного числа. Получаемые при этом числа называются комплексными и представляют собой величины, которые не существуют в известном нам физическом мире. Комплексные числа могут быть записаны в одной из двух форм:  $a + bj$  (алгебраическая) и  $Me^{j\theta}$  (полярная). Символом  $\langle j \rangle$  обозначается  $\sqrt{-1}$ . При любой из двух форм записи информация содержится в двух компонентах: в первом случае — в  $a$  и  $b$ , во втором — в  $M$  и  $\theta$ . Несмотря на некую иллюзорность, комплексные числа подчиняются тем же (или аналогичным) законам математики, что и обыкновенные (вещественные) числа.

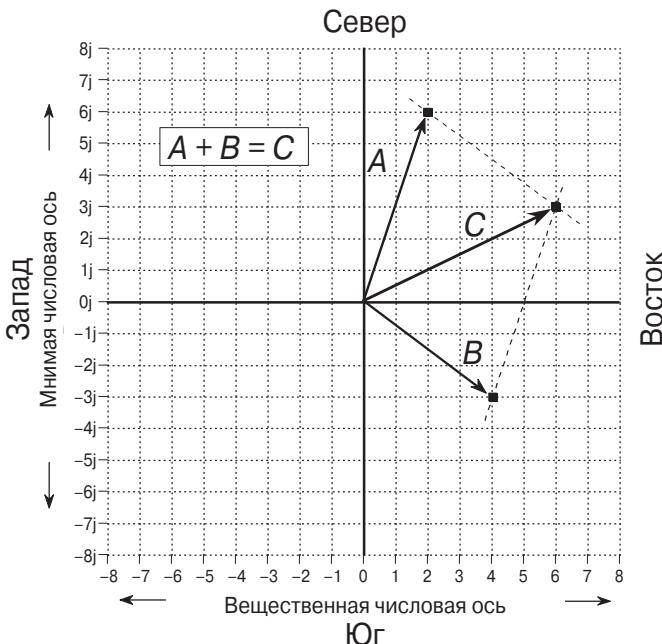
Итак, мы знаем, что такое комплексные числа и как они вписываются в мир теоретической математики. Теперь наша задача заключается в том, чтобы показать, как применить комплексные числа при решении инженерных и научных задач. Откуда возникает возможность использовать для решения повседневных практических задач математику комплексных чисел, которая кажется оторванной от реального мира? Ответ прост: если в нашем распоряжении оказался молоток, остаётся сделать из решаемой задачи гвоздь. Другими словами, задачу реального физического мира достаточно сформулировать с использованием терминологии комплексных чисел, а после того, как поставленная задача будет решена, вернуться в реальный мир.

Существуют два пути перехода от реальности к комплексным числам: простой метод — *метод замещения*, более изощрённый — *метод математической эквивалентности*. О втором методе поговорим в Главе 31, посвящённой комплексному преобразованию Фурье. В данной главе расскажем о методе замещения.

Метод замещения основан на использовании двух вещественных физических параметров, один из которых отождествляется с вещественной частью комплексного числа, а второй — с его мнимой частью. Это позволяет работать с двумя величинами как с единым комплексным числом. После выполнения необходимых математических операций полученное комплексное число разделяется на вещественную и мнимую части, которые снова соответствуют реальным физическим величинам.

Приведём простой пример. Из курса физики известно, что такие параметры, как сила, скорость, ускорение и т. п., могут быть представлены в виде векторов. Пусть имеется парусное судно, которое ветер несёт в одну сторону, а течение океана — в другую. Результирующая сила, действующая на судно, равна сумме двух векторов силы. На Рис. 30.3 сложение двух векторов —  $A$  и  $B$  по правилу параллограмма даёт результирующий вектор  $C$ .

Постараемся описать задачу с использованием комплексных чисел, для чего заменим силы их проекциями на координатные оси, одна из которых имеет направление восток-запад, а другая — север-юг, а потом отождествим эти проекции соответственно с вещественной и мнимой частями комплексных чисел. При таком подходе каждый вектор представляется одним комплексным числом, несмотря на то, что имеет две проекции. Например, вектор силы ветра  $A$  имеет протяжённость две единицы в сторону востока и 6 единиц в сторону севера, поэтому его можно представить комплексным числом  $(2 + 6j)$ . Аналогично вектор силы



**Рис. 30.3** Сложение векторов с помощью комплексных чисел. Каждый из векторов  $A$  и  $B$ , которые соответствуют складываемым силам, можно разложить на две компоненты: одну — действующую в направлении север-юг, и другую — действующую в направлении восток-запад. Направление восток-запад отождествляется с вещественной осью, а направление север-юг — с мнимой осью. Такое замещение позволяет применить комплексную математику для решения задачи с вещественными величинами.

течения  $B$  имеет протяжённость 4 единицы в сторону востока и 3 единицы в сторону юга, что может быть выражено комплексным числом  $(4 - 3j)$ . Теперь сложим два вектора по правилу выражения (30.1) и получим комплексное число  $(6 + 3j)$ , представляющее собой вектор  $C$ . Выполнив обратный переход к реальным физическим величинам, находим действующую на судно результирующую силу, которая представляется вектором протяжённостью 6 единиц в сторону севера и 3 единицы — в сторону востока.

Можно ли было решить такую задачу без применения комплексных чисел? Конечно. Комплексные числа всего лишь делают использование двух проекций вектора более формализованным. Из рассмотренного примера следует уяснить главную идею: переход от постановки задачи, описанной через вещественные числа, к постановке, описанной через комплексные числа, заключается во введении символа  $j$  в одну из составляющих. И наоборот, отбрасывая  $j$ , мы возвращаемся к исходным физическим величинам. В этом и заключается суть метода замещения.

Но здесь есть одна трудность. Как узнать, что правила и законы, применимые к комплексной математике, являются теми же правилами и законами, которые применимы к исходной физической задаче? Допустимо ли использовать выражение (30.1) для сложения векторов сил при решении задачи с парусником? Как убедиться в том, что сложение комплексных чисел даёт тот же результат, что и

сложение векторов силы? В большинстве случаев о применимости комплексной математики к решению конкретной задачи известно с чьих-то слов. Какой-то всеми признанный математик или инженер провел основательное исследование метода решения задачи и опубликовал результаты. Помните: нельзя преобразовать в комплексную форму любую задачу и быть уверенными, что получится правильный ответ. Следует выбирать только такие задачи, которые уже зарекомендовали себя с точки зрения применимости комплексного анализа.

Рассмотрим пример, в котором метод замещения комплексными числами не работает. Предположим, вы покупаете яблоки по цене 5 долларов за ящик, а апельсины — по цене 10 долларов за ящик. Вы пытаетесь объединить две цены одним комплексным числом  $(5 + 10j)$ . За неделю вы купили 6 ящиков яблок и 2 ящика апельсинов и точно так же объединяете эти числа одним комплексным числом  $(6 + 2j)$ . Стоимость любого товара равна его количеству, умноженному на цену. Пользуясь этим правилом, вы умножаете комплексные числа:  $(5 + 10j)(6 + 2j) = = 10 + 70j$ . Получается, что надо уплатить за яблоки 10 долларов, а за апельсины — 70 долларов. Результат явно неправильный. Оказывается, что в данном случае правила комплексной математики не соответствуют правилам реальной задачи.

### 30.3. Комплексное представление синусоидальных функций

Комплексные числа нашли свою нишу в задачах электротехники и обработки сигналов благодаря компактности записи и простоте выполнения операций с наиболее широко применяемыми функциями — синусом и косинусом. Простейшей формой представления гармонических функций (синусов и косинусов) являются выражения вида  $M\cos(\omega t + \varphi)$  или  $A\cos(\omega t) + B\sin(\omega t)$ , которые соответствуют полярной и алгебраической формам комплексного числа. Здесь  $\omega$  — круговая (циклическая) частота, измеряемая в радианах в секунду (рад/с). Если вам удобнее использовать обычную линейную частоту  $f$ , можете выполнить замену  $\omega$  на  $2\pi f$  (частота  $f$  измеряется в герцах — Гц). Тем не менее большинство расчётов в ЦОС производится с более краткой формой записи, и к этому надо привыкнуть. Поскольку гармонические функции (при заданной частоте) определяются двумя параметрами ( $A$  и  $B$  или  $M$  и  $\varphi$ ), то для их представления идеально подходят комплексные числа. Благодаря методу замещения переход от гармонических функций к комплексным числам и обратно не представляет труда. При использовании алгебраической формы:

$$A\cos(\omega t) + B\sin(\omega t) \leftrightarrow a - jb,$$

(обычное представление)      (комплексное число)

где  $A \leftrightarrow a$  и  $B \leftrightarrow -b$ .

То есть амплитуда косинусоиды представляется вещественной частью комплексного числа, а амплитуда синусоиды, взятая с обратным знаком, — мнимой частью. Важно понять, что это не уравнение, а лишь способ представления гармонической функции комплексным числом. Замещение можно также выполнить в полярной форме:

$$M \cos(\omega t + \varphi) \leftrightarrow M e^{j\theta},$$

(обычное представление) (комплексное число)

где  $M \leftrightarrow M$  и  $\theta \leftrightarrow -\varphi$ .

То есть в полярную форму амплитуда входит без изменений, а фаза меняет знак на противоположный. Зачем менять знак мнимой части и фазы? Затем, чтобы замещение приводило к тем же результатам, что и комплексное преобразование Фурье, речь о котором пойдёт в следующей главе. С точки зрения описываемого здесь метода замещения от перемены знака ничего не зависит, но зато появляется соответствие с более совершенными методами.

Представление синусоидального и косинусоидального колебаний комплексными переменными — способ, весьма распространённый в теории цепей и ЦОС. Объясняется это тем, что многие (хотя далеко не все) правила и законы, действующие для комплексных чисел, выполняются и для гармонических функций. Другими словами, синусоидальные и косинусоидальные колебания можно спокойно заменять комплексными числами, выполнять с этими числами необходимые операции, получая правильные конечные результаты.

Тем не менее надо в каждом случае проявлять внимание и пользоваться только теми математическими операциями, которые подходят для решаемой физической задачи. Предположим, что мы используем комплексные переменные  $A$  и  $B$  для обозначения двух гармонических сигналов равной частоты, но разных амплитуд и фаз. В результате сложения этих чисел мы получаем третье комплексное число. Точно так же при сложении двух синусоид мы получаем третью. Можно быть уверенным в том, что полученное третье комплексное число соответствует третьей синусоиде. Выполненное нами сложение комплексных чисел находится в полном соответствии с работой данной физической системы.

А теперь предположим, что комплексные числа  $A$  и  $B$  перемножаются, образуя третье комплексное число  $C$ . Соответствует ли данная операция умножения комплексных чисел тому, что происходит при перемножении гармонических функций? Нет! Дело в том, что перемножение двух гармонических функций не приводит к образованию третьей. То есть умножение комплексных чисел не находит соответствия в данной физической системе и, следовательно, не может быть использовано.

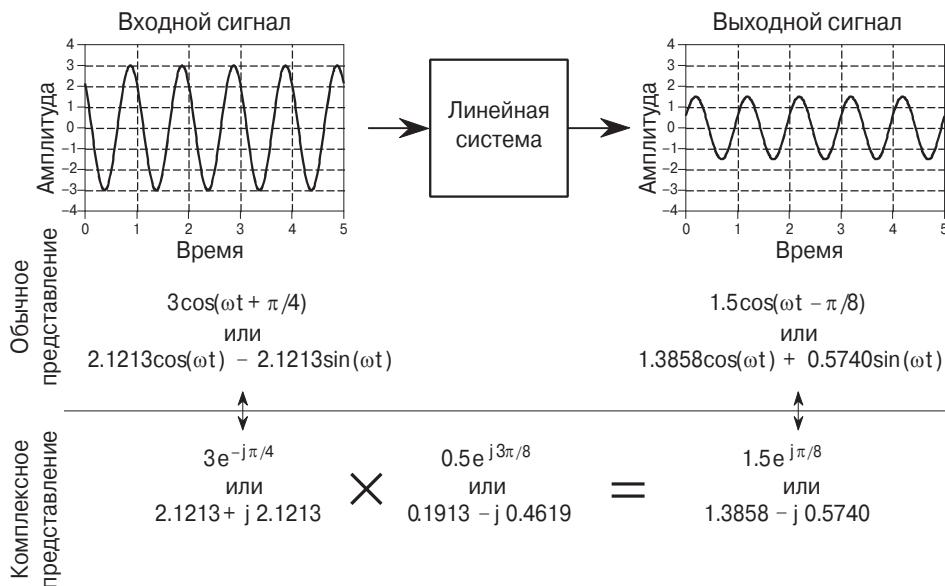
К счастью, множество допустимых операций хорошо известно. Достаточно выполнения двух условий. Во-первых, все гармонические функции должны иметь одинаковую частоту. Например, если комплексные числа  $(1 + 1j)$  и  $(2 + 2j)$  соответствуют функциям одной и той же частоты, то их сумма  $(3 + 3j)$  тоже будет соответствовать гармонической функции этой частоты. Однако если под теми же комплексными числами подразумеваются гармонические функции разных частот, то аппарат комплексных чисел применить не удастся. Полученный в этом случае результат суммирования оказывается бессмысленным.

Несмотря на это, переменная частоты сигнала может явно присутствовать в записях комплексных выражений. Но во всех выражениях переменная частоты должна быть одной и той же. Допускается, например, складывать  $(2\omega + 3\omega j)$  и  $(3\omega + 1j)$ , в результате чего получается  $(5\omega + (3\omega + 1)j)$ . В этих выражениях учитывается зависимость амплитуды и фазы от частоты сигнала. Хотя частота непосредственно не задана, можно точно сказать, что она одинакова во всех выражениях и равна  $\omega$ .

Второе требование заключается в том, что выполняемые с помощью комплексных чисел действия должны быть линейными, о чём говорилось в Главе 5. Это значит, что гармонические сигналы допускается складывать и вычитать, но ни в коем случае не умножать и не делить. Другими словами, комплексные числа могут использоваться при анализе усилителей, НЧ- и ВЧ-фильтров и т. п., но нельзя применять их при описании таких операций, как возведение в квадрат, ограничение по времени и сравнение с порогом. Запомните, что такие операции, как свёртка и преобразование Фурье, являются линейными.

## 30.4. Описание систем с использованием комплексных чисел

На Рис. 30.4 показан пример, в котором процесс прохождения гармонического сигнала через линейную систему описан на основе аппарата комплексных чисел. В нём рассматривается обработка непрерывного сигнала. При переходе к дискретным сигналам принцип остаётся тем же. Поскольку на вход поступает гармоническое воздействие, а система линейна, то реакция будет гармонической функцией той же частоты. Для примера выбран сигнал  $(3\cos(\omega t + \pi/4))$ . Эквивалентная форма записи такого сигнала имеет вид  $(2.1213\cos(\omega t) - 2.1213\sin(\omega t))$ . Данному сигналу соответствует комплексное число  $3e^{-j\pi/4}$  в полярной форме или  $(2.1213 + j2.1213)$  в алгебраической форме. Аналогичным образом выходному сигналу



**Рис. 30.4.** Представление гармонических сигналов комплексными числами. Комплексные числа распространены в электротехнике и ЦОС, благодаря удобству, которое они обеспечивают при описании гармонических сигналов и при проведении операций над ними. Данный пример показывает, что входной и выходной сигналы можно описать комплексными числами, записанными либо в полярной, либо в алгебраической форме. Кроме того, сам процесс изменения сигнала системой можно описать комплексным числом.

$(1.5\cos(\omega t - \pi/8))$ , или  $(1.3858\cos(\omega t) + 0.5740\sin(\omega t))$ , соответствует комплексное число  $1.5e^{j\pi/8}$ , или  $(1.3858 - j 0.5740)$ .

Комплексными числами можно описать характеристики системы. При этом модуль комплексного числа выражает отношение амплитуды выходного сигнала к амплитуде входного:  $M_{\text{вых}}/M_{\text{вх}}$ , а его фаза равна разности фаз входного и выходного сигналов:  $-\left[\Phi_{\text{вых}} - \Phi_{\text{вх}}\right]$ . В рассматриваемом примере систему можно описать комплексным числом  $0.5e^{j3\pi/8}$ . Иными словами, амплитуда синусоиды уменьшается до уровня 0.5 от первоначального значения, а фаза изменяется на  $-3\pi/8$  рад.

Комплексное число, представляющее данную систему, можно записать в алгебраической форме как  $(0.1913 - j 0.4619)$ . Но будьте внимательны: данная запись не означает, что синусоида, проходящая через систему, изменяет свою амплитуду с коэффициентом 0.1913 или что косинусоида изменяется в 0.4619 раза. На самом деле обе составляющих исходного сигнала порождают на выходе линейной системы смесь косинусоидальной и синусоидальной компонент.

К счастью, при вычислениях в комплексной форме подобный взаимный переход между синусоидальными и косинусоидальными компонентами учитывается автоматически. Когда гармонический сигнал проходит через линейную систему, то перемножение двух комплексных чисел, одно из которых представляет входной сигнал, а второе — систему, приводит к получению третьего комплексного числа, представляющего выходной сигнал. Если известны какие-либо два комплексных числа, всегда можно найти третье. Как показано на Рис. 30.4, вычисления можно выполнять как в полярной, так и в алгебраической форме.

В предыдущих главах мы рассказали о том, как с помощью преобразования Фурье разложить сигнал на косинусные и синусные компоненты. Амплитуды косинусных компонент называют вещественной составляющей спектра, а амплитуды синусных компонент — мнимой составляющей спектра. Мы обратили особое внимание на тот факт, что эти амплитуды являются обычными вещественными числами и называются вещественной и мнимой частью спектра только для того, чтобы различать их между собой. Теперь, когда мы ввели понятие комплексного числа, становится понятным происхождение названий этих составляющих. Пусть сигнал из 1024 отсчётов разложен на 513 синусных и 513 косинусных компонент. Воспользовавшись методом замещения, его спектр можно представить состоящим из 513 комплексных чисел. Однако не запутайтесь: такое разложение сигнала вовсе не является комплексным преобразованием Фурье, речь о котором пойдет в Главе 31. Это по-прежнему вещественное преобразование Фурье, для которого использован метод комплексного замещения.

## 30.5. Анализ электрических цепей

Рассмотренный нами метод замещения комплексными числами синусоидальных и косинусоидальных сигналов называется *векторным преобразованием*. Это преобразование является основным средством анализа электрических схем, содержащих сопротивления, ёмкости и индуктивности. Говоря более строго, векторное преобразование определяется в инженерной практике как умножение на комплексную экспоненту вида  $e^{j\omega t}$  с последующим извлечением вещественной части. Такой подход позволяет записать преобразование в виде более простого уравнения. Метод замещения приводит к такому же результату, но имеет менее элегантную форму.

Прежде всего нужно установить связь между током и напряжением для каждого из элементов электрической цепи. Для сопротивления такая связь формулируется законом Ома:  $u = iR$ , где  $i$  — мгновенное значение тока через сопротивление,  $u$  — мгновенное значение напряжения, падающего на этом сопротивлении,  $R$  — значение сопротивления. В отличие от сопротивления для ёмкости и индуктивности взаимосвязь тока и напряжения выражается дифференциальными уравнениями: для ёмкости —  $i = C \frac{du}{dt}$ , для индуктивности —  $u = L \frac{di}{dt}$ . Здесь  $C$  — значение ёмкости, а  $L$  — значение индуктивности. При самом общем подходе к анализу электрических цепей эти малоприятные дифференциальные уравнения объединяют в соответствии со структурой конкретной электрической цепи, а затем находят из них нужные параметры. Хотя таким способом мы сможем получить всю интересующую нас информацию о схеме, математические действия могут оказаться очень сложными.

Всю процедуру можно упростить, если ограничиться рассмотрением гармонических сигналов. Если представить такие сигналы комплексными числами, появляется возможность перейти от сложных дифференциальных уравнений к намного более простым алгебраическим уравнениям. Всё сказанное поясняется на Рис. 30.5. Будем рассматривать каждый из трёх основных элементов электрических цепей — сопротивление, ёмкость и индуктивность — как систему, для которой входным сигналом является мгновенное значение тока, изменяющееся по гармоническому закону, а выходным — мгновенное значение напряжения на элементе, которое тоже меняется по гармоническому закону. Это означает, что вход и выход системы могут быть описаны при помощи двух комплексных переменных:  $I$  — комплексный ток,  $U$  — комплексное напряжение. Соотношение величин входного и выходного сигналов также выражается комплексным числом. Это число называется *полным сопротивлением* (или *импедансом*) и обозначается символом  $Z$ :

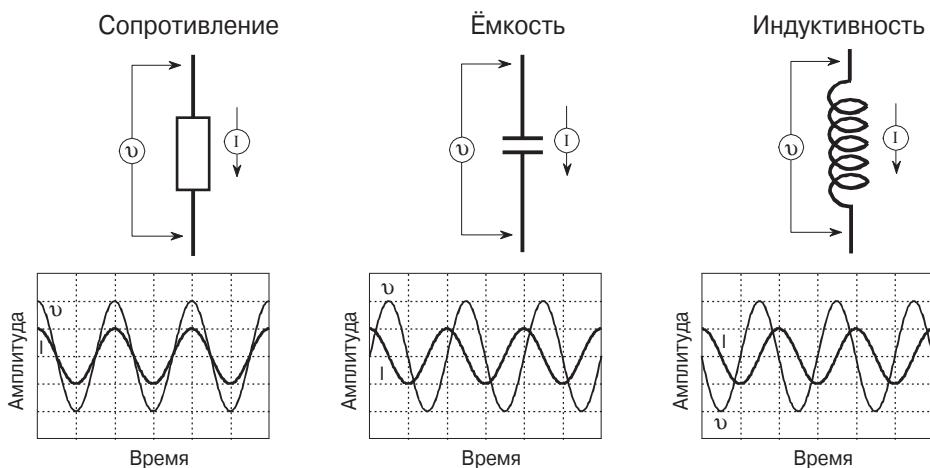
$$I \times Z = U.$$

Данное уравнение означает, что комплексные числа, описывающие гармонический закон изменения напряжения, равны произведению соответствующего комплексного числа, описывающего гармонический закон изменения тока, на комплексное число, описывающее полное сопротивление. При любых двух известных величинах всегда можно найти третью. В полярной форме записи модуль полного сопротивления равен отношению модулей  $U$  и  $I$ , а фаза полного сопротивления есть разность фаз  $U$  и  $I$ .

Данное выражение является обобщённой формой закона Ома. Закон Ома связывает напряжение и ток на активном сопротивлении:  $u = iR$ . При этом коэффициент пропорциональности называется сопротивлением. При рассмотрении гармонических сигналов, представляемых комплексными числами, связь между током и напряжением приобретает вид  $U = IZ$ , в котором напряжение и ток снова связывает полное сопротивление (импеданс). Сопротивление в обычном законе Ома является вещественным числом, поскольку связывает две вещественные величины. Полное сопротивление — это комплексное число, связывающее две комплексные величины. Полное сопротивление несёт больше информации, чем обычное (активное) сопротивление, поскольку учитывает не только амплитуду, но и фазу.

Из приведённых выше дифференциальных уравнений, описывающих взаимосвязь тока и напряжения, можно получить выражения импеданса всех трёх основ-

ных элементов — сопротивления ( $R$ ), ёмкости ( $-j/\omega C$ ) и индуктивности ( $j\omega L$ ). Предположим, что через каждый из основных элементов электрической цепи протекает ток, величина которого изменяется по закону косинуса с амплитудой, равной единице (Рис. 30.5). При использовании терминологии комплексных чисел такой сигнал можно обозначить выражением  $(1 + 0j)$ . Напряжение на сопротивлении связано с током следующим выражением:  $U = IZ = (1 + 0j)R = R + 0j$ , т. е. является косинусоидой с амплитудой  $R$ . Теперь находим напряжение на ёмкости:  $U = IZ = (1 + 0j)(-j/\omega C) = 0 - j/\omega C$ . То есть напряжение на ёмкости изменяется по синусоидальному закону, причём амплитуда зависит от частоты и равна  $1/\omega C$ . Аналогично выражаем напряжение на индуктивности:  $U = IZ = (1 + 0j)(j\omega L) = 0 + j\omega L$ . То есть напряжение на индуктивности изменяется по синусоидальному закону с противоположным знаком и амплитудой  $\omega L$ .



**Рис. 30.5.** Определение полного сопротивления. Если напряжение и ток, изменяющиеся по гармоническому закону, представить комплексными числами, то отношение этих чисел будет называться полным сопротивлением (или импедансом). Обозначается оно комплексной переменной  $Z$ . Сопротивление, ёмкость и индуктивность характеризуются импедансом, равным  $R$ ,  $-j/\omega C$  и  $j\omega L$  соответственно.

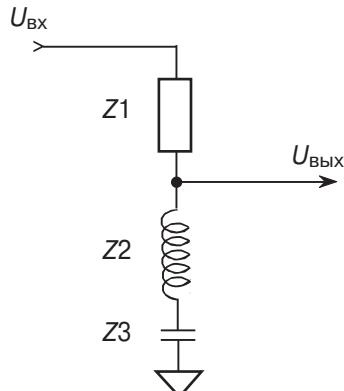
Красота описанного метода состоит в том, что появляется возможность анализировать  $RLC$ -цепи, не прибегая к использованию дифференциальных уравнений. С импедансом сопротивлений, ёмкостей и индуктивностей можно работать точно так же, как с сопротивлением в цепях постоянного тока. Сказанное относится ко всем основным комбинациям элементов в цепях: последовательному, параллельному соединениям элементов и делителям напряжения.

На Рис. 30.6 показана  $RLC$ -схема, применяемая для вырезания узкой полосы частот из спектра, поступающего на вход сигнала, и называемая *узкополосным резонансным фильтром*. С помощью такой схемы можно, например, вырезать помеху, наведённую сетью питания в измерительной и звуковоспроизводящей аппаратуре. Если бы все три элемента в этой схеме были обычными резисторами, то отношение входного и выходного сигналов выражалось бы формулой делителя напряжения:  $u_{\text{вых}}/u_{\text{вх}} = (R2 + R3)/(R1 + R2 + R3)$ . Но так как в схеме присутствуют ёмкость и индуктивность, придётся воспользоваться понятием импеданса:

$$\frac{U_{\text{вых}}}{U_{\text{вх}}} = \frac{Z2 + Z3}{Z1 + Z2 + Z3},$$

где  $U_{\text{вых}}$ ,  $U_{\text{вх}}$ ,  $Z1$ ,  $Z2$  и  $Z3$  — комплексные переменные. Подставив соответствующее выражение для импеданса каждого элемента схемы, получим

$$\frac{U_{\text{вых}}}{U_{\text{вх}}} = \frac{j\omega L - \frac{j}{\omega C}}{R + j\omega L - \frac{j}{\omega C}}.$$



**Рис. 30.6.** RLC-схема узкополосного режекторного фильтра. Схема предназначена для подавления в спектре узкой полосы частот. Использование метода замещения комплексными числами значительно упрощает анализ данной схемы и других подобных ей схем.

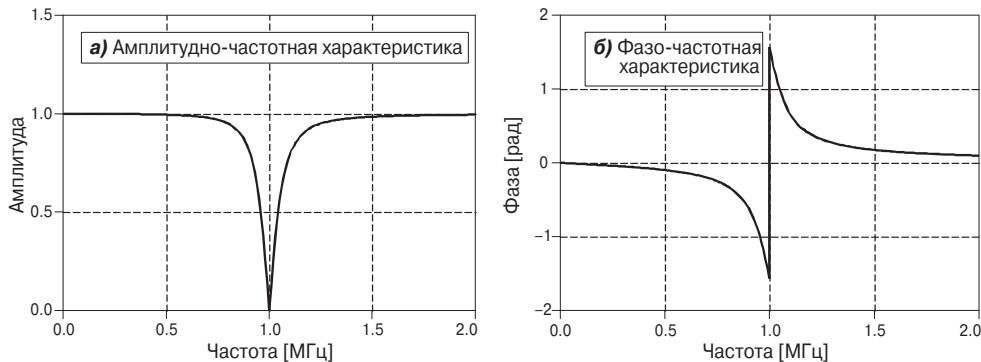
Далее выполним алгебраические действия, чтобы отделить все, что содержит  $j$ , от всего, что не содержит  $j$ . Другими словами, разделим уравнение на действительную и мнимую части. Используемая алгебраическая процедура может получиться долгой и утомительной, но её альтернатива ещё хуже, так как заключается в решении дифференциальных уравнений. После разделения на вещественную и мнимую части комплексное уравнение для режекторного фильтра принимает следующий вид:

$$\frac{U_{\text{вых}}}{U_{\text{вх}}} = \frac{k^2}{R^2 + k^2} + j \frac{Rk}{R^2 + k^2},$$

$$\text{где } k = \omega L - \frac{1}{\omega C}.$$

Теперь данное выражение остаётся преобразовать в полярную форму записи, чтобы отобразить в виде двух основных частотных характеристик на графиках (**Рис. 30.7**):

$$\left| \frac{U_{\text{вых}}}{U_{\text{вх}}} \right| = \frac{\omega L - 1/\omega C}{\left( R^2 + [\omega L - 1/\omega C]^2 \right)^{1/2}}, \quad \arg \left( \frac{U_{\text{вых}}}{U_{\text{вх}}} \right) = \arctg \left( \frac{R}{\omega L - 1/\omega C} \right).$$



**Рис. 30.7.** Частотные характеристики узкополосного режекторного фильтра. Графики построены при следующих значениях параметров элементов цепи:  $R = 50 \text{ Ом}$ ,  $C = 470 \text{ пФ}$ ,  $L = 54 \text{ мкГн}$ .

Главное, что следует уяснить из приведенных выше примеров, — это сам принцип метода замещения, позволяющий перейти от физической реальности к комплексной абстракции. В следующей главе будет рассмотрен более совершенный метод применения комплексных чисел в науке и технике — метод математической эквивалентности.

## КОМПЛЕКСНОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ

Хотя комплексные числа сами по себе никак не связаны с реальным физическим миром, их можно использовать для решения научных и технических задач двумя способами. Первый способ заключается в замещении параметров задачи, возникшей в реальном мире, комплексными числами, о чём говорилось в предыдущей главе. Второй способ является более изящным и более мощным, он связан с таким применением комплексных чисел, которое обеспечивает математическую эквивалентность действительной физической задаче. Этот подход связан с *комплексным преобразованием Фурье*, представляющим собой более сложный вариант вещественного преобразования Фурье, описанного в Главе 8. Комплексное преобразование Фурье важно не только само по себе, но и как предпосылка к ещё более мощным комплексным методам, таким как преобразование Лапласа и *Z*-преобразование. Эти комплексные преобразования составляют основу теории ЦОС.

### 31.1. Вещественное ДПФ

Любое преобразование, входящее в четвёрку *преобразований Фурье* (дискретное преобразование Фурье (ДПФ), преобразование Фурье дискретного времени, преобразование Фурье и ряд Фурье), можно выполнить, используя как вещественные, так и комплексные числа. Поскольку в ЦОС чаще всего приходится иметь дело с ДПФ, большинство рассуждений проводится на примере этого преобразования. Перед тем как погрузиться в комплексную математику, вспомним, что такое вещественное ДПФ, особо уделяя внимание тем его сторонам, которые являются не вполне удачными с точки зрения математики. В Главе 8 мы описали вещественное ДПФ при помощи следующей пары уравнений:

$$\begin{aligned} \operatorname{Re} X[k] &= \frac{2}{N} \sum_{n=0}^{N-1} x[n] \cos(2\pi kn/N), \\ \operatorname{Im} X[k] &= -\frac{2}{N} \sum_{n=0}^{N-1} x[n] \sin(2\pi kn/N). \end{aligned} \tag{31.1}$$

ДПФ в вещественной форме. Данное преобразование описывает переход от временной области к частотной. Несмотря на использование выражений «вещественная часть» и «мнимая часть», в уравнениях присутствуют только обычные (вещественные) числа. Индекс частоты  $k$  изменяется в диапазоне  $0...N/2$ . Уравнения не отличаются от выражения (8.4), за исключением того, что в них присутствует коэффициент  $2/N$ .

Данные уравнения описывают процесс разложения сигнала  $x[n]$ , состоящего из  $N$  дискретных отсчётов, взятых во временной области, на совокупность  $(N/2 + 1)$  синусных и  $(N/2 + 1)$  косинусных компонент с частотами, определяемыми индек-

сом  $k$ .  $\text{Re}X[k]$  — амплитуды косинусных составляющих дискретного спектра,  $\text{Im}X[k]$  — амплитуды синусных составляющих. Суть уравнений заключается в вычислении корреляции временной формы сигнала с соответствующими косинусами и синусами. Несмотря на использование таких терминов, как «вещественная часть» и «мнимая часть», в этих уравнениях нет комплексных чисел. Символа  $j$  здесь не найти! Кроме того, в этих уравнениях присутствует коэффициент нормировки  $2/N$ . Напомним, что при обсуждении выражения (8.3) говорилось о возможности переноса такого коэффициента из уравнения анализа в уравнение синтеза и обратно, а также о возможности вынесения операции нормировки в отдельный этап. Вы должны помнить эти уравнения из предыдущих глав книги. Но если вы уже забыли их, лучше вернуться назад и повторить данный материал прежде, чем читать дальше. Кто не понимает вещественное ДПФ, вряд ли поймёт комплексное.

В то время как вещественное ДПФ использует только вещественные числа, метод замещения позволяет описать частотную область ДПФ с помощью комплексных чисел. В соответствии с обозначением массивов  $\text{Re } X[k]$  — вещественная часть комплексного частотного спектра, а  $\text{Im } X[k]$  — его мнимая часть. Можно сказать также, что к мнимой части уравнения приписывается символ  $j$ , а затем обе части складываются. Тем не менее не стоит думать, что это и есть комплексное ДПФ. На самом деле это не что иное, как всё то же вещественное ДПФ, к которому применили метод замещения комплексными числами.

Хотя вещественное ДПФ подходит для решения многих научных и технических задач, оно является несколько неудачным с точки зрения математики по трём причинам. Первая причина: все преимущества комплексных чисел ограничиваются в случае вещественного ДПФ лишь использованием метода комплексного замещения. Это создаёт неудобства для математиков, потому что удобнее сказать, что «это равно тому-то», а не что «это заменяет то-то». Например, математическое утверждение « $A$  равно  $B$ » сразу же начинает порождать в нашем сознании бесконечное множество следствий:  $5A = 5B$ ,  $1 + A = 1 + B$ ,  $A/x = B/x$  и т. д. А теперь представьте, что мы располагаем утверждением « $A$  заменяет  $B$ ». При отсутствии дополнительной информации мы не можем добавить к нему абсолютно ничего! Когда речь идёт о равенстве величин, мы опираемся на четыре тысячелетия, прошедшие с начала развития математики. Когда величины только заменяют друг друга, приходится начинать с введения новых определений. Например, если речь идёт о замене гармонических функций комплексными числами, мы допускаем операции сложения и вычитания, исключая умножение и деление.

Второй недостаток вещественного ДПФ связан с понятием *отрицательных частот* спектра. Как говорилось в Главе 10, одна и та же синусоида или косинусоида может быть представлена как положительной, так и отрицательной частотой. Поскольку оба таких представления равноправны, при рассмотрении вещественного ДПФ обычно не принимают во внимание отрицательные частоты. Но существуют такие практические приложения, в которых необходимо учитывать влияние отрицательных частот. Примером является их проявление в области положительных частот, вызванное смещением спектра, что можно сравнить с «материализацией призраков». Подобные эффекты наблюдаются при наложении спектров, круговой свёртки и амплитудной модуляции. Поскольку вещественное преобразование Фурье не учитывает наличия отрицательных частот, его возможности в такого рода задачах весьма ограничены.

Третьим недостатком является особая обработка первой и последней точек частотного спектра:  $\operatorname{Re} X[0]$  и  $\operatorname{Re} X[N/2]$ . Если последовательность  $x[n]$  представлена  $N$  отсчётыами, то полученный с помощью ДПФ дискретный спектр сигнала содержит отсчёты вида  $\operatorname{Re} X[k]$  и  $\operatorname{Im} X[k]$ , где  $k$  изменяется в пределах  $0 \dots N/2$ . Однако амплитуды компонент такого сигнала не позволяют выполнить непосредственный обратный переход во временную область: полученные значения отсчётов  $\operatorname{Re} X[0]$  и  $\operatorname{Re} X[N/2]$  необходимо уменьшить вдвое, как говорилось при описании выражения (8.3). Эту операцию деления легко выполнить при помощи компьютерной программы, но при записи математических уравнений она приносит много неудобств.

Более изящное решение перечисленных здесь проблем достигается в комплексном преобразовании Фурье. Можно сказать, что при использовании комплексного ДПФ отрицательные частоты идут рука об руку с комплексными числами. Посмотрим, как работает это преобразование.

## 31.2. Математическая эквивалентность

Сначала покажем, как выразить синус и косинус через комплексные числа. Решением этой задачи является формула Эйлера, о которой говорилось в предыдущей главе:

$$e^{jx} = \cos(x) + j \sin(x). \quad (31.2)$$

Формула Эйлера.

На первый взгляд может показаться, что данное уравнение мало чем способно помочь нам: в нём всего лишь отражено равенство двух комплексных выражений. И всё же с помощью несложной алгебры его можно преобразовать в два других уравнения:

$$\cos(x) = \frac{e^{jx} + e^{-jx}}{2}, \quad \sin(x) = \frac{e^{jx} - e^{-jx}}{2j}. \quad (31.3)$$

Формула Эйлера для синуса и косинуса.

Данный результат очень важен, поскольку полученные правила связывают гармонические тригонометрические функции и комплексные экспоненты. Хотя выражение (31.3) является стандартной формой, выражающей такую связь, для нашей дальнейшей работы удобнее перейти к следующей форме записи:

$$\begin{aligned} \cos(\omega t) &= \frac{1}{2} e^{j(-\omega)t} + \frac{1}{2} e^{j\omega t}, \\ \sin(\omega t) &= \frac{1}{2} j e^{j(-\omega)t} - \frac{1}{2} j e^{j\omega t}. \end{aligned} \quad (31.4)$$

Связь гармонических функций с комплексными экспонентами. При использовании комплексных чисел косинус и синус можно разложить на сумму комплексных экспонент с положительным и отрицательным показателями.

Каждое выражение представляет собой сумму двух комплексных экспонент, из которых одна соответствует положительной частоте, а вторая — отрицательной. Другими словами, при записи синуса и косинуса в комплексной форме автоматически учитывается отрицательная часть спектра. Положительные и отрицательные частоты полностью равноправны и поэтому входят в формулу с половинными амплитудами.

### 31.3. Комплексное ДПФ

Прямое комплексное ДПФ при использовании полярной формы записи выражается следующим соотношением:

$$X[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N}. \quad (31.5)$$

Прямое комплексное ДПФ.  $X[k]$  — отсчёты сигнала в частотной области,  $x[n]$  — отсчёты сигнала во временной области. Индексы  $k$  и  $n$  изменяются в пределах  $0\dots(N-1)$ . Данная полярная форма записи уравнения получила широкое распространение в ЦОС.

Используя формулу Эйлера, можно перейти к алгебраической форме записи прямого ДПФ:

$$X[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] (\cos(2\pi kn/N) - j \sin(2\pi kn/N)). \quad (31.6)$$

Прямое комплексное ДПФ (алгебраическая форма записи).

Для начала сравним уравнение комплексного ДПФ с уравнением вещественного ДПФ (31.1). На первый взгляд кажется, что различия между выражениями (31.1) и (31.6) совершенно незначительны и могут быть устранены с помощью нескольких простых алгебраических преобразований. Однако подобные выводы оказываются слишком опрометчивыми, т. к. отличия между этими двумя уравнениями имеют очень большое значение. Опишем их более подробно.

Во-первых, вещественное ДПФ превращает вещественный сигнал, заданный во временной области  $x[n]$ , в два вещественных сигнала в частотной области —  $\operatorname{Re}X[k]$  и  $\operatorname{Im}X[k]$ . Благодаря методу комплексного замещения полученные в частотной области сигналы можно представить одним массивом комплексных чисел  $X[k]$ . Что касается комплексного ДПФ, то в нём комплексными массивами являются как  $x[n]$ , так и  $X[k]$ . Обратите внимание на то, что временная область является комплексной, ничто не заставляет нас использовать мнимую часть массивов. Допустим, нужно обработать вещественный сигнал, например последовательность отсчётов напряжения, полученных в дискретные моменты времени. Эти вещественные значения присваиваются вещественным частям отсчётов сигнала, а мнимым частям отсчётов присваиваются нулевые значения.

Во-вторых, вещественное ДПФ учитывает только положительные частоты. То есть индекс частоты  $k$  изменяется на интервале  $0\dots N/2$ . Отличием комплексного ДПФ является то, что оно учитывает дополнительно как положительные, так и отрицательные частоты. Другими словами, в случае вещественного ДПФ индекс  $k$  принимает значения  $0\dots(N-1)$ . Индексы  $0\dots(N/2)$  соответствуют поло-

жительным частотам, а  $N/2 \dots (N - 1)$  — отрицательным частотам. Напомним, что частотный спектр дискретного сигнала является периодическим и отрицательные частоты на интервале  $N/2 \dots (N - 1)$  совпадают с частотами интервала  $-N/2 \dots 0$ . Отсчёты с индексами 0 и  $N/2$  являются границами, разделяющими положительный и отрицательный диапазоны частот. Если вы хотите повторить этот материал, вернитесь к Главам 10 и 12.

В-третьих, при использовании метода замещения для описания вещественного ДПФ к синусным составляющим спектра добавляют символ  $j$ , что позволяет представить частотный спектр с помощью комплексных чисел. Выполняя обратный переход к обычным синусам и косинусам, символ  $j$  можно просто отбросить. Такое небрежное обращение оказывается возможным только потому, что одно из понятий всего лишь заменяет другое. В комплексное ДПФ (31.5)  $j$  входит как неотъемлемая часть, и его нельзя произвольно вводить или удалять, как нельзя вводить или удалять любую из переменных.

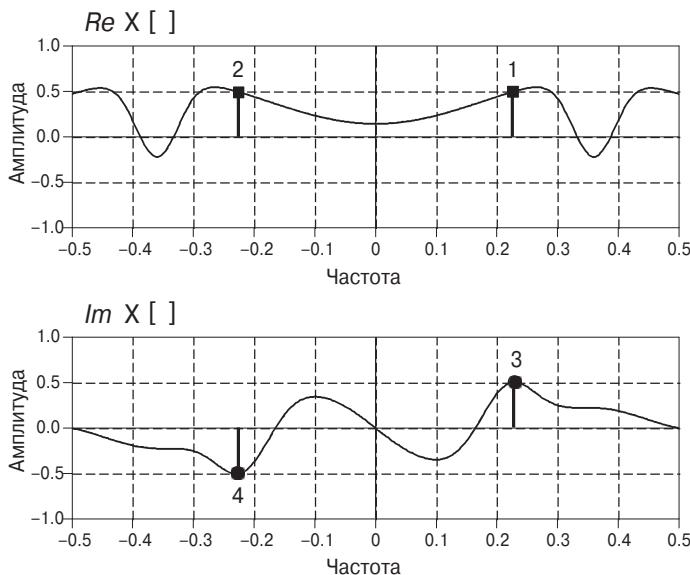
В-четвёртых, в начале уравнения вещественного ДПФ присутствует коэффициент масштабирования, равный двум, который отсутствует в уравнении комплексного ДПФ. Если выполнить вещественное ДПФ косинусоиды с единичной амплитудой, то соответствующая косинусная компонента в спектре сигнала тоже окажется равной единице. Выполнение комплексного ДПФ приводит к иному результату. В спектре косинусоидального сигнала будут присутствовать две компоненты: одна — в положительной области частот, а другая — в отрицательной. Значение обеих компонент равно  $1/2$ . Другими словами, компонента амплитудой  $1/2$  в диапазоне положительных частот, сложенная с компонентой такой же амплитуды, но с противоположной по знаку частотой, даёт косинусоиду единичной амплитуды (при рассмотрении во временной области).

В-пятых, при использовании вещественного ДПФ требуется специальная обработка двух частотных компонент спектра:  $ReX[0]$  и  $ReX[N/2]$ , в то время как при использовании комплексного ДПФ все частоты обрабатываются по одним и тем же правилам. Пусть имеется сигнал, представленный своими дискретными отсчётами во временной области. С помощью ДПФ можно найти спектр такого сигнала. Для выполнения обратной операции следует выполнить обратное ДПФ, в результате чего будет восстановлен исходный временной сигнал. Но для правильного восстановления исходного сигнала нужно вводить коэффициент масштабирования. В комплексном ДПФ масштабирующий коэффициент одинаков для всех компонент частотного спектра и равен  $1/N$ . Его можно вводить либо в уравнение прямого ДПФ, либо в уравнение обратного ДПФ, либо вообще вынести в отдельную операцию. В формулах вещественного ДПФ масштабирующий коэффициент в 2 раза больше, чем в комплексном, и равен  $2/N$ , о чём говорилось выше. Более того, для вещественного ДПФ вводится дополнительное масштабирование, в результате которого  $ReX[0]$  и  $ReX[N/2]$  делятся на два. Иными словами, для этих двух компонент спектра масштабирующий коэффициент равен  $1/N$ , а для всех остальных —  $2/N$ . Ранее уже упоминался этот недостаток, связанный с неудобством математического описания вещественного ДПФ.

Почему же вещественное и комплексное ДПФ по-разному работают с двумя указанными отсчётами в спектре сигнала? Чтобы ответить на поставленный вопрос, вспомним, что косинусоидальной (или синусоидальной) функции, определённой во временной области, комплексное ДПФ ставит в соответствие две ком-

поненты в спектре сигнала. Исключение из этого правила составляют два отсчёта спектра: 0 и  $N/2$ . Эти отсчёты соответствуют нулевой частоте (постоянной составляющей сигнала) и частоте Найквиста (половине частоты дискретизации). Поскольку эти отсчёты находятся на границах, разделяющих положительные и отрицательные частоты, то они не имеют соответствующих пар. В результате они вносят вдвое меньший вклад по сравнению с другими отсчётами спектра.

На Рис. 31.1 показан спектр, полученный с помощью комплексного ДПФ. Предполагается, что во временной области сигнал является полностью вещественным, т. е. мнимая часть равна нулю. К вопросу представления временных сигналов, имеющих нулевую мнимую часть, обратимся чуть позже. Обычно используют два способа графического отображения комплексного частотного спектра. Один из них использован на рисунке: нулевая частота располагается в середине отображенного интервала, положительные частоты — справа, отрицательные — слева. Этот способ наилучшим образом подходит для восприятия всего спектра целиком и является единственным способом графического отображения апериодического спектра.



**Рис. 31.1.** Комплексный частотный спектр. Кривые соответствуют вещественному (во временной области) сигналу, так как вещественная часть спектра обладает чётной симметрией, а мнимая — нечётной. Две точки вещественной части спектра, отмеченные на графике квадратиками, соответствуют одной косинусоидальной составляющей с частотой, равной 0.23, и единичной амплитудой. Две точки мнимой части спектра, отмеченные кружками, соответствуют одной синусоидальной составляющей с такими же значениями частоты и амплитуды.

Проблема выбора способа отображения спектра связана с периодичностью спектра дискретного сигнала (независимо от того, получен он с помощью ДПФ или с помощью преобразования Фурье дискретного времени). Это означает, что все отсчёты спектра, расположенные в диапазоне частот  $-0.5 \dots 0.5$ , периодически повторяются при смещении вправо и влево от этого интервала. Следовательно, интервал частот  $0 \dots 1.0$  содержит ту же информацию, что и интервал  $-0.5 \dots 0.5$ .

При построении графиков, подобных тому, что изображён на Рис. 31.1, обычно рассматривают интервал  $-0.5 \dots 0.5$ . Однако во многих уравнениях и программах используется интервал  $0 \dots 1.0$ . Например, в выражениях (31.5) и (31.6) индекс частоты  $k$  принимает значения  $0 \dots (N - 1)$ , что соответствует диапазону частот  $0 \dots 1.0$ . Но эти уравнения легко переписать в такой форме, при которой  $k$  меняется в интервале  $N/2 \dots (N/2 - 1)$  (что соответствует диапазону частот  $-0.5 \dots 0.5$ ).

С помощью изображения спектра, приведённого на Рис. 31.1, можно проанализировать, как происходит восстановление временной формы сигнала при выполнении обратного ДПФ. Формула обратного комплексного ДПФ имеет следующий вид:

$$x[n] = \sum_{k=0}^{N-1} X[k] e^{j2\pi kn/N}. \quad (31.7)$$

Обратное комплексное ДПФ. Это уравнение соответствует прямому ДПФ (31.5).

Формула Эйлера позволяет перейти к алгебраической форме записи:

$$x[n] = \sum_{k=0}^{N-1} \operatorname{Re} X[k] (\cos(2\pi kn/N) + j \sin(2\pi kn/N)) - \sum_{k=0}^{N-1} \operatorname{Im} X[k] (\sin(2\pi kn/N) - j \cos(2\pi kn/N)). \quad (31.8)$$

Обратное комплексное ДПФ. Это уравнение эквивалентно выражению (31.7), но позволяет описать влияние на формирование сигнала во временной области к каждой из компонент комплексного спектра в отдельности.

Обычно обратное ДПФ записывают в более компактной форме (31.7), хотя, возможно, развернутое уравнение (31.8) является более наглядным. Из него следует, что каждый отсчёт вещественной части спектра порождает во временной области вещественную косинусоиду и мнимую синусоиду. Подобно этому любой отсчёт мнимой части спектра порождает вещественную синусоиду и мнимую косинусоиду. Сигнал во временной области складывается из всех этих вещественных и мнимых гармонических функций. Важным моментом является то, что любой отсчёт в частотном представлении сигнала порождает одновременно и вещественную, и мнимую гармонические функции во временной области.

Предположим, например, что с помощью обратного ДПФ воспроизводится косинусоида единичной амплитуды с частотой  $2\pi k/N$ . В частотной области косинусоида представляется двумя отсчётами в вещественной части спектра. На Рис. 31.1 косинусоида изображена двумя квадратиками, причём  $k/N = 0.23$ . Положительная частота со значением 0.23 (точка номер 1 на Рис. 31.1) порождает при переходе к временной области вещественную косинусоиду и мнимую синусоиду:

$$1/2 \cos(2\pi 0.23n) + 1/2 j \sin(2\pi 0.23n).$$

Точно так же и отрицательная частота со значением  $-0.23$  (точка номер 2 на Рис. 31.1) порождает при переходе к временной области вещественную косинусоиду и мнимую синусоиду:

$$1/2 \cos(2\pi (-0.23)n) + 1/2 j \sin(2\pi (-0.23)n).$$

Знак «минус» можно вынести из функций синуса и косинуса, руководствуясь следующими тригонометрическими тождествами:  $\cos(-x) = \cos(x)$  и  $\sin(-x) = -\sin(x)$ . В результате выражение, полученное для отрицательной частоты, примет вид

$$1/2 \cos(2\pi 0.23n) + 1/2 j \sin(2\pi 0.23n).$$

Складывая выражения, полученные для отрицательной и положительной частот, находим искомый временной сигнал:

$$\begin{array}{l} \text{выражение для положительной частоты} \rightarrow 1/2 \cos(2\pi 0.23n) + 1/2 j \sin(2\pi 0.23n) \\ \text{выражение для отрицательной частоты} \rightarrow 1/2 \cos(2\pi 0.23n) + 1/2 j \sin(2\pi 0.23n). \\ \hline \text{результатирующий временной сигнал} \rightarrow \cos(2\pi 0.23n) \end{array}$$

Аналогичным образом можно синтезировать синусоиду. В этом случае в спектре сигнала должны присутствовать две мнимые составляющие: одна — на положительной частоте и одна — на отрицательной. На Рис. 31.1 такие составляющие отмечены кружками. Из выражения (31.8) следует, что каждый из мнимых отсчётов спектра порождает вещественную синусоиду и мнимую косинусоиду во временной области. Мнимые косинусоиды сокращаются, вещественные синусоиды складываются:

$$\begin{array}{l} \text{выражение для положительной частоты} \rightarrow -1/2 \sin(2\pi 0.23n) - 1/2 j \cos(2\pi 0.23n) \\ \text{выражение для отрицательной частоты} \rightarrow -1/2 \sin(2\pi 0.23n) + 1/2 j \cos(2\pi 0.23n) \\ \hline \text{результатирующий временной сигнал} \rightarrow -\sin(2\pi 0.23n) \end{array}$$

Обратите внимание на то, что, хотя значения отсчётов в спектре были положительными, после перехода к временной области получилась синусоида со знаком «минус». Такая смена знака является характерной особенностью математического аппарата комплексного ДПФ. Как мы помните, смена знака встречается и в вещественном ДПФ. То есть положительная амплитуда компонент мнимой части частотного спектра соответствует отрицательной синусоиде. Во многих научных работах смена знака включается в определение вещественного ДПФ, для того чтобы это преобразование было согласовано с комплексным. Разница в том, что в комплексном ДПФ смена знака является необходимостью, тогда как в вещественном ДПФ она вводится искусственно.

Для комплексного ДПФ очень важна симметрия. Как мы видели на Рис. 31.1, вещественному временному сигналу соответствует частотный спектр с чётной симметрией вещественной части и нечётной симметрией мнимой части. Это значит, что составляющие комплексного спектра с равной по модулю и противоположной по знаку частотой подчиняются следующему правилу: если эти составляющие принадлежат вещественной части спектра, то они равны друг другу (точки 1 и 2 на Рис. 31.1), а если они принадлежат мнимой части спектра, то равны по модулю и противоположны по знаку (точки 3 и 4).

Тут мы подходим к вопросу о мнимой части сигнала, представленного во временной области. До сих пор мы считали, что во временной области сигнал явля-

ется полностью вещественным, т. е. имеет равную нулю мнимую часть. Но для комплексного ДПФ такого условия не требуется. Каков физический смысл мнимого во временной области сигнала? Как правило, у такого сигнала нет физического смысла. Он является чистой абстракцией комплексной математики и не имеет соответствия в реальном мире. Тем не менее существует целый ряд практических приложений, в которых введение мнимых сигналов упрощает математические преобразования.

В Главе 12 мы уже встречались с подобным примером. Мнимая часть временной области даёт частотный спектр с чётной симметрией вещественной части и нечётной симметрией мнимой части. Всё это в точности обратно тем принципам, по которым строится спектр вещественного сигнала (**Рис. 31.1**). Когда во временном представлении сигнала присутствуют одновременно вещественная и мнимая части, частотный спектр является суммой их спектров. В Главе 12 рассказывается, как этим свойством можно воспользоваться, для того чтобы с помощью алгоритма БПФ вычислить спектры двух вещественных сигналов одновременно. Для этого один сигнал отождествляют с вещественной частью комплексного сигнала, а другой — с его мнимой частью. После выполнения алгоритма БПФ спектры двух сигналов разделяют, используя декомпозицию на основе чётной и нечётной симметрии (см. Главу 5).

## 31.4. Семейство преобразований Фурье

Наличие двух форм — вещественной и комплексной — свойственно не только ДПФ, но и другим алгоритмам, принадлежащим семейству преобразований Фурье. Это порождает ряд уравнений, перечисленных в **Табл. 31.1**. Прежде чем начать подробное изучение каждого из алгоритмов в отдельности, постарайтесь разобраться с принципами классификации, позволяющими сделать систему алгоритмов хорошо упорядоченной и симметричной. Ниже приводится ряд комментариев, которые помогут понять принципы организации семейства преобразований Фурье. Комментарии выглядят слишком подробными, многословными и утомительными. Тем не менее в результате их изучения создаются те условия, на основе которых возможно правильное понимание теоретических положений ЦОС. Постарайтесь хорошенко разобраться с комментариями.

### 31.4.1. Четвёрка преобразований Фурье

При рассмотрении во временной области сигнал может быть описан либо непрерывной, либо дискретной функцией и одновременно являться либо периодическим, либо апериодическим. В результате такой классификации получается четыре типа преобразований Фурье: *дискретное преобразование Фурье* (дискретное, периодическое), *преобразование Фурье дискретного времени* (дискретное, апериодическое), *ряд Фурье* (непрерывное, периодическое) и *преобразование Фурье* (непрерывное, апериодическое).

Сопоставляя временную и частотную формы сигнала, можно указать следующий принцип: сигналу, дискретному в одной области, соответствует периодический сигнал в другой области, и, наоборот, непрерывному сигналу соответствует апериодический сигнал в другой области. Для описания непрерывных сигналов

используют круглые скобки, для описания дискретных — квадратные. При таких обозначениях сразу ясно, является ли спектр сигнала периодическим или апериодическим.

### 31.4.2. Вещественные и комплексные преобразования

Каждое из четырёх преобразований имеет комплексную и вещественную формы. Комплексные преобразования связывают комплексный сигнал во временной области и комплексный сигнал в частотной области. Вещественные преобразования связывают вещественный сигнал во временной области и пару вещественных сигналов в частотной области. В комплексных преобразованиях используются как положительные, так и отрицательные частоты, а в вещественных — только положительные. Комплексные преобразования обычно записываются в экспоненциальной форме, но при необходимости, воспользовавшись формулой Эйлера, можно перейти к синусам и косинусам.

### 31.4.3. Анализ и синтез

Для каждого преобразования вводятся два уравнения: уравнение анализа (уравнение прямого преобразования) и уравнение синтеза (уравнение обратного преобразования). Уравнения анализа описывают переход от временной области к частотной, уравнения синтеза — обратный переход.

### 31.4.4. Система обозначений во временной области

Непрерывный во времени сигнал обозначается  $x(t)$ , дискретный —  $x[n]$ . При комплексных преобразованиях эти сигналы считаются комплексными, при вещественных преобразованиях — вещественными. Все временные сигналы имеют бесконечную область определения. Однако если сигнал является периодическим, то достаточно ограничиться одним его периодом, поскольку всё, что расположено за пределами этого периода, несёт избыточную информацию. Переменными  $T$  и  $N$  во временной области обозначают период непрерывных и дискретных сигналов соответственно.

### 31.4.5. Система обозначений в частотной области

Непрерывные в частотной области сигналы обозначаются  $X(\omega)$ , если они являются комплексными, и  $Re X(\omega)$  и  $Im X(\omega)$ , если они являются вещественными. Дискретные в частотной области сигналы обозначаются  $X[k]$ , если они комплексные, и  $Re X[k]$  и  $Im X[k]$ , если они вещественные. Область определения сигнала, полученного в результате комплексного преобразования, содержит кроме положительных частот ( $0 \dots +\infty$ ) ещё и отрицательные частоты ( $-\infty \dots 0$ ), тогда как в случае вещественного преобразования ограничиваются рассмотрением только положительного частотного диапазона. Если сигнал описывается в частотной области

периодической функцией, то ограничиваются одним периодом, поскольку остальная часть сигнала несёт избыточную информацию. Для сигналов с непрерывным спектром период изменения независимой переменной  $\omega$  рассматривается на интервале  $-\pi \dots \pi$ . Для сигналов с дискретным спектром переменная  $k$  «пробегает» все целые числа в диапазоне  $0 \dots (N - 1)$ .

### 31.4.6. Уравнения анализа

Уравнения анализа сводятся к вычислению корреляции, т. е. к умножению сигнала, описанного во временной области, на гармонические функции с последующим интегрированием (для непрерывных сигналов) или суммированием (для дискретных сигналов). В случае аperiодических сигналов корреляция вычисляется на бесконечном интервале  $(-\infty \dots +\infty)$ . Если сигнал является периодическим, корреляция вычисляется на любом конечном интервале, протяжённость которого равна одному периоду. В приведенных далее уравнениях интегрирование (суммирование) выполняется на интервале  $0 \dots T$  ( $0 \dots (N - 1)$ ). Хотя выбор любого другого периода приводит к точно таким же результатам, т. е. можно выбрать интервалы  $-T \dots 0$ ,  $-T/2 \dots T/2$  и т. д.

### 31.4.7. Уравнения синтеза

Уравнения синтеза устанавливают соответствие между гармоническими компонентами, из которых складывается временная форма сигнала, и порождающими их частотами, содержащимися в спектре этого сигнала. Процедура синтеза сводится к умножению сигнала, представленного в частотной области, на гармонические функции разной частоты с последующим интегрированием (для сигналов с непрерывным спектром) или суммированием (для сигналов с дискретным спектром). Если спектр сигнала является комплексным и аperiодическим, то интервал интегрирования или суммирования имеет бесконечно большую длину  $(-\infty \dots +\infty)$ . Если спектр является комплексным и периодическим, то протяжённость такого интервала сокращается до периода  $-\pi \dots \pi$  для непрерывного спектра и  $0 \dots (N - 1)$  — для дискретного. Если спектр является вещественным и аperiодическим, то интервал интегрирования или суммирования ограничивается неотрицательными числами  $(0 \dots +\infty)$ . Если спектр вещественный и периодический, этот интервал сокращается либо до  $0 \dots \pi$  в случае непрерывного спектра, либо до  $0 \dots N/2$  в случае дискретного спектра.

### 31.4.8. Масштабирование

Для того чтобы последовательное использование уравнения анализа и уравнения синтеза позволяло получить исходный сигнал, необходимо вводить масштабирование. В уравнениях, представленных в Табл. 31.1, коэффициент масштабирования содержится в уравнениях анализа. Для комплексных преобразований коэффициенты масштабирования выглядят следующим образом:  $1/N$ ,  $1/T$ ,  $1/(2\pi)$ . Так как вещественные преобразования не учитывают отрицательных частот, те же коэффициенты для них в 2 раза больше:  $2/N$ ,  $2/T$ ,  $1/\pi$ . При вычислении мнимой части спектра в уравнениях вещественных преобразований присутствует отрица-

тельный знак, который нужен только для того, чтобы получить более точное соответствие вещественного и комплексного преобразований. Наконец, в уравнениях синтеза вещественного ДПФ и вещественного ряда Фурье вводятся специальные коэффициенты масштабирования для отсчётов  $\operatorname{Re} X[0]$  и  $\operatorname{Re} X[N/2]$ , что указано в строках 5, 10, 15 и 20 Табл. 31.1.

### 31.4.9. Другие формы записи уравнений

Уравнения, описывающие преобразования Фурье, могут иметь и другие формы записи, отличающиеся от представленных выше. Перечислим несколько отличий, на которые следует обратить внимание:

- использование  $f$  вместо  $\omega$ , где  $\omega = 2\pi f$ ;
- отличие в выборе пределов интегрирования:  $-T \dots 0, -T/2 \dots T/2, 0 \dots T$ ;
- перемещение коэффициента масштабирования или его части в уравнение синтеза;
- замена параметра периода параметром частоты дискретизации:  $f_0 = 1/T$ ;
- использование других обозначений переменных: например, в случае апериодического ДПФ  $\omega$  может обозначаться как  $\Omega$ , а  $\operatorname{Re} X[k]$  и  $\operatorname{Im} X[k]$  могут входить в уравнение ряда Фурье как  $a_k$  и  $b_k$ .

## 31.5. Зачем использовать комплексное преобразование Фурье?

Из всего сказанного в этой главе совершенно очевидно, что комплексное ДПФ гораздо сложнее вещественного. Оправдывают ли те преимущества, которые можно получить с помощью комплексного ДПФ, необходимость изучения трудных математических формул? Ответ на этот вопрос зависит от того, кто вы и какие цели преследуете. Основное достоинство данной книги в том, что большинство рассмотренных в ней методов ЦОС описано без привлечения комплексных чисел, что нисколько не мешает правильно понять эти методы и научиться использовать их на практике. Если вы изучаете ЦОС, чтобы повысить эффективность проводимых исследований или при решении каких-либо других инженерных задач в областях, напрямую не связанных с ЦОС, то комплексное ДПФ окажется излишне мощным оружием в ваших руках.

Для тех же, кто специализируется в области ЦОС, комплексная математика является основным языком описания алгоритмов обработки. Если вы не владеете этим языком, вам не удастся общаться с профессионалами, работающими в этой области. Сюда же следует отнести возможность чтения литературы по ЦОС: книги, научные труды, статьи в технических журналах и т. д. Почему же комплексные методы столь популярны среди специалистов в ЦОС?

Тому есть целый ряд причин, которые здесь уже упоминались: компактность форм записи уравнений, симметрия между уравнениями анализа и синтеза, симметрия между временной и частотной областями, учёт отрицательного диапазона частот. Кроме того, комплексные преобразования являются первым шагом к преобразованию Лапласа и  $Z$ -преобразованию.

Ещё одна причина, о которой мы пока не упоминали, имеет философские корни и связана с понятием правильности наших представлений о мире. В самом начале главы говорилось о том, что вещественное преобразование Фурье по целому ряду аспектов является несколько неудачным. Введение комплексного преобразования Фурье позволяет избавиться от возникших неудобств. Отлично, говорим мы, — комплексное преобразование Фурье решает все проблемы!

Такое умозаключение является справедливым, однако оно далеко не полностью отражает значение комплексного преобразования Фурье. Посмотрим на данную ситуацию несколько с другой стороны. Несмотря на свой абстрактный характер, комплексное преобразование Фурье очень точно описывает суть поведения физической системы. Когда мы ограничиваем математические действия областью вещественных чисел, возникают проблемы. Можно сказать, что не комплексное преобразование Фурье решает проблемы, а что вещественное преобразование Фурье эти проблемы создаёт. С точки зрения математики комплексное преобразование Фурье оказывается ближе к истинному описанию мира, чем вещественное. Именно это привлекает математиков и других ученых, для которых стремление приблизиться к правильному пониманию окружающего мира важнее простого решения какой-либо частной задачи.

Таблица 31.1. Преобразования Фурье

Ряд Фурье	
Комплексное преобразование	Вещественное преобразование
Уравнение синтеза: $x(t) = \sum_{k=-\infty}^{+\infty} X[k] e^{j2\pi kt/T}$	Уравнение синтеза: $x(t) = \sum_{k=0}^{+\infty} Re X[k] \cos(2\pi kt/T) - Im X[k] \sin(2\pi kt/T)$
Уравнение анализа: $X[k] = \frac{1}{T} \int_0^T x(t) e^{-j2\pi kt/T} dt$	Уравнение анализа: $Re X[k] = \frac{2}{T} \int_0^T x(t) \cos(2\pi kt/T) dt$ $Im X[k] = \frac{-2}{T} \int_0^T x(t) \sin(2\pi kt/T) dt$
Временная область: $x(t)$ — комплексная, непрерывная, периодическая, $t$ изменяется в пределах одного периода: $0 \dots T$ .	Временная область: $x(t)$ — вещественная, непрерывная, периодическая, $t$ изменяется в пределах одного периода: $0 \dots T$ .
Частотная область: $X[k]$ — комплексная, дискретная, апериодическая, $k$ изменяется в бесконечных пределах $-\infty \dots +\infty$ , $k > 0$ — диапазон положительных частот, $k < 0$ — диапазон отрицательных частот	Частотная область: $Re X[k]$ — вещественная, дискретная, апериодическая, $Im X[k]$ — вещественная, дискретная, апериодическая, $k$ изменяется в полубесконечных пределах $0 \dots +\infty$ .
<b>Примечание.</b> Перед тем как воспользоваться уравнением синтеза, $Re X[0]$ требуется разделить на два	

(продолжение)

Преобразование Фурье	
Комплексное преобразование	Вещественное преобразование
Уравнение синтеза: $x(t) = \int_{-\infty}^{+\infty} X(\omega) e^{j\omega t} d\omega$	Уравнение синтеза: $x(t) = \int_0^{+\infty} Re X(\omega) \cos(\omega t) - Im X(\omega) \sin(\omega t) dt$
Уравнение анализа: $X(\omega) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} x(t) e^{-j\omega t} dt$	Уравнение анализа: $Re X(\omega) = \frac{1}{\pi} \int_{-\infty}^{+\infty} x(t) \cos(\omega t) dt$ $Im X(\omega) = \frac{-1}{\pi} \int_{-\infty}^{+\infty} x(t) \sin(\omega t) dt$
Временная область: $x(t)$ — комплексная, непрерывная, апериодическая, $t$ изменяется в бесконечных пределах $-\infty \dots +\infty$ . Частотная область: $X(\omega)$ — комплексная, непрерывная, апериодическая, $\omega$ изменяется в бесконечных пределах $-\infty \dots +\infty$ , $\omega > 0$ — диапазон положительных частот, $\omega < 0$ — диапазон отрицательных частот	Временная область: $x(t)$ — вещественная, непрерывная, апериодическая, $t$ изменяется в бесконечных пределах $-\infty \dots +\infty$ . Частотная область: $Re X(\omega)$ — вещественная, непрерывная, апериодическая, $Im X(\omega)$ — вещественная, непрерывная, апериодическая, $\omega$ изменяется в полубесконечных пределах $0 \dots +\infty$
Дискретное преобразование Фурье (ДПФ)	
Комплексное преобразование	Вещественное преобразование
Уравнение синтеза: $x[n] = \sum_{k=0}^{N-1} X[k] e^{j2\pi kn/N}$	Уравнение синтеза: $x[n] = \sum_{k=0}^{N/2} Re X[k] \cos(2\pi kn/N) - Im X[k] \sin(2\pi kn/N)$
Уравнение анализа: $X[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/T}$	Уравнение анализа: $Re X[k] = \frac{2}{T} \sum_{n=0}^{N-1} x[n] \cos(2\pi kn/N)$ $Im X[k] = \frac{-2}{T} \sum_{n=0}^{N-1} x[n] \sin(2\pi kn/N)$
Временная область: $x[n]$ — комплексная, дискретная, периодическая, $n$ принимает значения в интервале $0 \dots (N-1)$ . Частотная область: $X[k]$ — комплексная, дискретная, периодическая, $k$ изменяется в пределах одного периода: $0 \dots (N-1)$ , $k = 0 \dots N/2$ — диапазон положительных частот, $k = N/2 \dots (N-1)$ — диапазон отрицательных частот	Временная область: $x[n]$ — вещественная, дискретная, периодическая, $n$ принимает значения в интервале $0 \dots (N-1)$ . Частотная область: $Re X[k]$ — вещественная, дискретная, периодическая, $Im X[k]$ — вещественная, дискретная, периодическая, $k$ изменяется на интервале, соответствующем половине периода: $0 \dots N/2$ .
<b>Примечание.</b> Перед тем как воспользоваться уравнением синтеза, $Re X[0]$ и $Re X[N/2]$ требуется разделить на два	

(продолжение)

Преобразование Фурье дискретного времени	
Комплексное преобразование	Вещественное преобразование
Уравнение синтеза: $x[n] = \int_0^{2\pi} X(\omega) e^{j\omega n} d\omega$	Уравнение синтеза: $x[n] = \int_0^{\pi} Re X(\omega) \cos(\omega n) - Im X(\omega) \sin(\omega n) d\omega$
Уравнение анализа: $X(\omega) = \frac{1}{2\pi} \sum_{n=-\infty}^{+\infty} x[n] e^{-j\omega n}$	Уравнение анализа: $Re X(\omega) = \frac{1}{\pi} \sum_{n=-\infty}^{+\infty} x[n] \cos(\omega n)$ $Im X(\omega) = \frac{-1}{\pi} \sum_{n=-\infty}^{+\infty} x[n] \sin(\omega n)$
Временная область: $x[n]$ — комплексная, дискретная, апериодическая, $n$ изменяется в бесконечных пределах $-\infty \dots +\infty$ .	Временная область: $x[n]$ — вещественная, дискретная, апериодическая, $n$ изменяется в бесконечных пределах $-\infty \dots +\infty$ .
Частотная область: $X(\omega)$ — комплексная, непрерывная, периодическая, $\omega$ изменяется в пределах одного периода $0 \dots 2\pi$ , $0 < \omega < \pi$ — диапазон положительных частот, $\pi < \omega < 2\pi$ — диапазон отрицательных частот	Частотная область: $Re X(\omega)$ — вещественная, непрерывная, периодическая, $Im X(\omega)$ — вещественная, непрерывная, периодическая, $\omega$ изменяется на интервале, соответствующем половине периода $0 \dots \pi$

## ПРЕОБРАЗОВАНИЕ ЛАПЛАСА

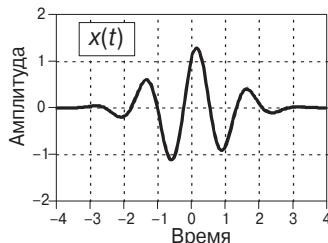
Две основные концепции обработки сигналов — *свёртка* и *Фурье-анализ* — учат тому, что как *импульсная*, так и *частотная характеристика* несут в себе исчерпывающую информацию о свойствах любой *линейной системы*. Такой подход к исследованию систем является слишком обобщенным в силу того, что импульсная и частотная характеристики могут принимать почти любые возможные формы. И действительно, для многих инженерных и научных исследований подобное обобщение оказывается излишним. Параметры многих явлений окружающего нас мира связаны посредством *дифференциальных уравнений*. Например, напряжение на катушке индуктивности пропорционально производной от протекающего через неё тока. Сила, прикладываемая к объекту некоторой массы, пропорциональна производной от приобретаемой им скорости. Вся физика наполнена такими зависимостями. В системах, основу которых составляют данные уравнения, формы частотных и импульсных характеристик не могут принимать произвольный вид, а должны соответствовать характеру решаемых дифференциальных уравнений. Это означает, что импульсные характеристики могут быть составлены только из экспоненциальных или синусоидальных функций. При анализе систем, предназначенных для обработки непрерывных сигналов, пользуются *преобразованием Лапласа*; при анализе дискретных систем — *Z-преобразованием*.

### 32.1. Понятие *S*-плоскости

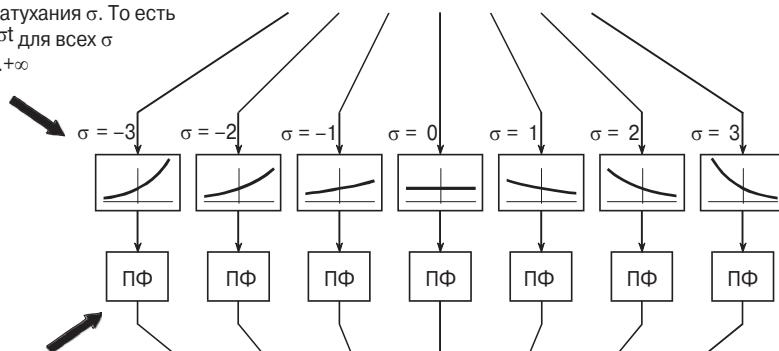
*Преобразование Лапласа* — это метод решения *дифференциальных уравнений*, давно и надёжно утвердившийся в математике. Своим именем метод обязан великому французскому математику Пьеру Симону де Лапласу (1749–1827). В основе преобразования Лапласа, как и в основе других преобразований над сигналами, лежит набор уравнений, определяющих правило перехода от одной функции к другой (от оригинала к изображению). Как показано на **Рис. 32.1**, преобразование Лапласа переводит сигнал из временной области в *S-область*, которую также называют *S-плоскостью*. Временной сигнал является непрерывным, определяется на интервале  $-\infty \dots +\infty$  и может быть либо периодическим, либо апериодическим. Преобразование Лапласа допускает описание временной области с использованием комплексных чисел, однако при обработке сигналов такое встречается нечасто. В этой книге, как и во множестве практических приложений, сигнал во временной области считается вещественным.

Как видно по **Рис. 32.1**, *S*-область является комплексной плоскостью, т. е. вещественная часть числа откладывается по горизонтальной оси, а мнимая — по вертикальной. Расстояние по вещественной оси выражается переменной  $\sigma$ , которая в греческом алфавите называется строчной буквой «сигма». Для измерения

**Шаг 1.**  
Выбираем сигнал  $x(t)$ ,  
определенный во временной области



**Шаг 2.**  
Умножаем сигнал  $x(t)$  на бесконечное  
число экспонент, различающихся  
коэффициентом затухания  $\sigma$ . То есть  
вычисляем  $x(t)e^{-\sigma t}$  для всех  $\sigma$   
на интервале  $-\infty \dots +\infty$

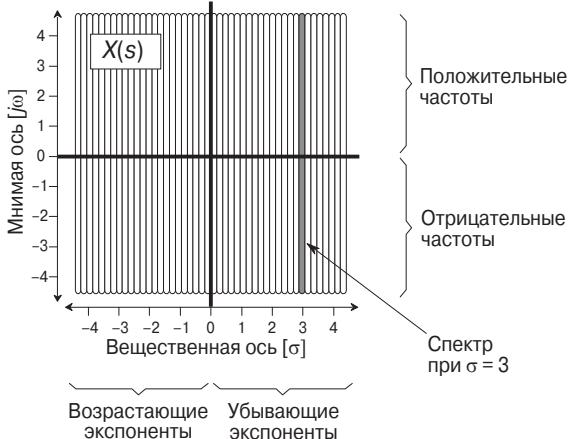


**Шаг 3.**  
Вычисляем преобразование  
Фурье (ПФ) для каждого  
взвешенного сигнала. То есть  
вычисляем

$$\int_{-\infty}^{\infty} [x(t)e^{-\sigma t}]e^{-j\omega t}dt$$

для всех  $\sigma$  на интервале  $-\infty \dots +\infty$

**Шаг 4.**  
Размещаем каждый спектр  
вдоль вертикальной прямой  
на S-плоскости: вверху —  
положительные частоты,  
внизу — отрицательные



**Рис. 32.1.** Преобразование Лапласа. Преобразование Лапласа переносит сигнал из временной области  $x(t)$  в S-область  $X(s)$  или  $X(\sigma, \omega)$ . Комплексные значения, соответствующие точкам любой произвольно выбранной в S-области вертикальной прямой, могут быть получены в результате умножения временного сигнала на экспоненциальную функцию с коэффициентом затухания  $\sigma$  с последующим выполнением комплексного преобразования Фурье. Если во временной области сигнал является вещественным, то верхняя полуплоскость S-плоскости окажется зеркальным отражением нижней полуплоскости.

расстояний, откладываемых по мнимой оси, используется переменная  $\omega$  — круговая частота. Такая координатная система позволяет описать положение любой точки с помощью  $\sigma$  и  $\omega$ . Пользуясь терминологией комплексных чисел, каждой точке можно поставить в соответствие комплексное число  $s = \sigma + j\omega$ . По аналогии с преобразованием Фурье сигналы, заданные в  $S$ -области, обозначают заглавными буквами. Например, определённому во временной области сигналу  $x(t)$  ставится в соответствие сигнал  $X(s)$ , определённый в  $S$ -области, который можно также записать в виде функции двух переменных —  $X(\sigma, \omega)$ . В любом из четырёх возможных направлений  $S$ -плоскость простирается на бесконечно большое расстояние.

Кроме того, что каждая точка имеет комплексную координату на плоскости, ей соответствует ещё и некоторая комплексная величина. Другими словами, каждой точке комплексной плоскости соответствуют вещественная и мнимая компоненты. Как известно, от алгебраической формы записи комплексных чисел можно перейти к полярной форме записи, выразив число через амплитуду и фазу.

Аналогично тому как в случае преобразования Фурье процедура анализа основана на использовании синусоидальных функций, анализ на основе преобразования Лапласа опирается на синусоидальные и экспоненциальные функции. С точки зрения математики, преобразование Фурье представляет собой частный случай преобразования Лапласа. Взаимосвязь  $S$ -области и временной области поясняется на Рис. 32.1. Чтобы определить значения комплексных величин, соответствующих точкам произвольно выбранной на  $S$ -плоскости вертикальной оси, временному сигналу сначала умножают на экспоненту  $e^{-\sigma t}$ . Левой полуплоскости соответствуют монотонно возрастающие экспоненты ( $\sigma < 0$ ), а правой — монотонно убывающие ( $\sigma > 0$ ). Затем к найденному путём экспоненциального взвешивания сигналу применяют комплексное преобразование Фурье. Полученный спектр соответствует точкам, образующим на  $S$ -плоскости вертикальную прямую, причём в верхней части параметр  $\omega$  принимает положительные значения, а в нижней — отрицательные. Обратите внимание, что значения, получаемые в точках  $S$ -плоскости, лежащих на оси ординат ( $\sigma = 0$ ), совпадают с результатами преобразования Фурье.

Как говорилось в предыдущей главе, комплексное преобразование Фурье имеет вид:

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

От преобразования Фурье легко перейти к преобразованию Лапласа. Для начала следует умножить сигнал  $x(t)$  на экспоненциальный множитель:

$$X(\sigma, \omega) = \int_{-\infty}^{\infty} [x(t)e^{-\sigma t}] e^{-j\omega t} dt.$$

Хотя это и не самая простая форма записи преобразования Лапласа, она скорее всего наиболее точно выражает суть выполняемых действий. Чтобы перейти к более короткой форме записи, объединим две входящие в него экспоненты:

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

Положение точки на комплексной плоскости задается комплексной переменной  $s$ , где  $s = \sigma + j\omega$ . Это позволяет сделать запись еще более компактной:

$$X(s) = \int_{-\infty}^{\infty} x(t)e^{-st} dt. \quad (32.1)$$

**Преобразование Лапласа.** Данное уравнение устанавливает соответствие между сигналом во временной области  $x(t)$  и его изображением в  $S$ -области  $X(s)$ . Определённые в  $S$ -области  $s$  и  $X(s)$  являются комплексными величинами. Хотя преобразование Лапласа и допускает комплексную форму сигналов во временной области, чаще всего эти сигналы являются вещественными.

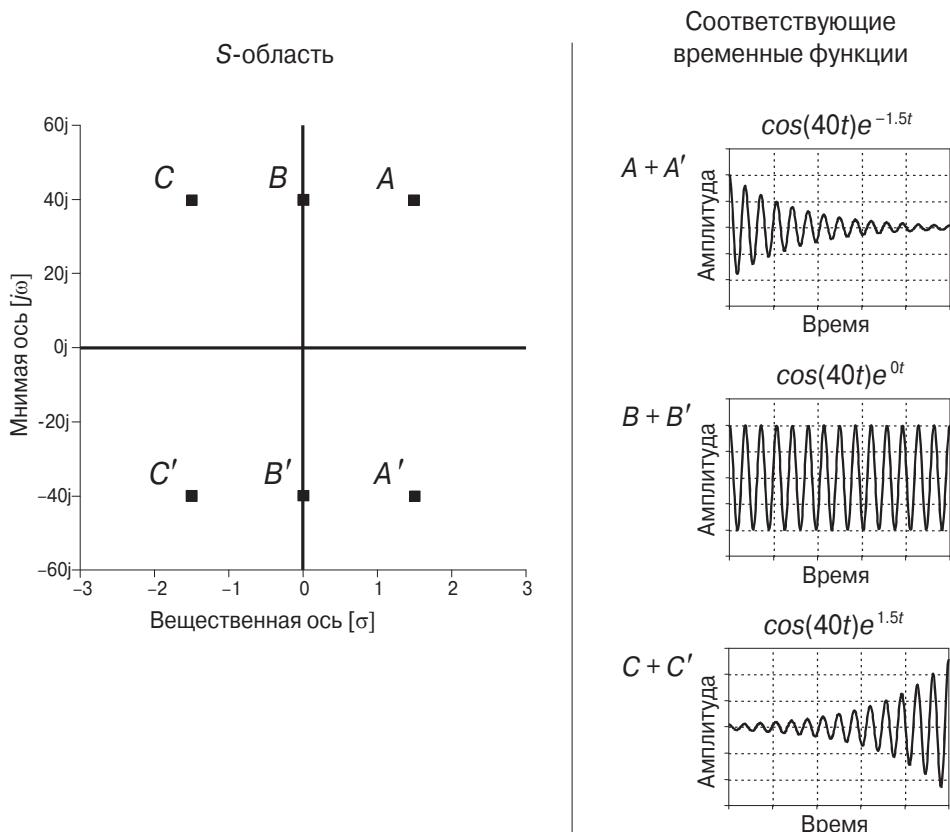
*Полученное выражение является основной формой записи преобразования Лапласа и одним из наиболее важных уравнений в обработке сигналов и электротехнике.* Обратите внимание на множитель  $e^{-st}$ , который называется комплексной экспонентой. Из вывода формулы следует, что весовые коэффициенты, заданные при помощи комплексных экспонент, представляют собой более компактный способ представления синусоид и вещественных экспоненциальных функций.

Несмотря на то что, объясняя суть преобразования Лапласа, мы представили его в виде двух последовательных этапов — умножения на экспоненциальную функцию и преобразования Фурье, — имейте в виду, что такое разделение выражения (32.1) на более простые компоненты было сделано только с целью объяснения материала. На самом деле преобразование Лапласа — это одно уравнение, связывающее сигналы  $x(t)$  и  $X(s)$ , а не пошаговая процедура. В выражении (32.1) содержится правило вычисления значений комплексных чисел, соответствующих каждой из точек  $S$ -плоскости, на основе двух координат точки — параметров  $\sigma$  и  $\omega$  — и заданного во временной области сигнала  $x(t)$ . Использование преобразования Фурье для вычисления комплексных значений во всех точках вертикальных прямых необходимо лишь для удобства, но вовсе не является обязательным требованием при построении схемы преобразования Лапласа. Тем не менее следует помнить, что на оси ординат комплексной  $S$ -плоскости ( $\sigma = 0$ ) результаты, полученные с помощью преобразования Лапласа, совпадают с результатами, найденными с помощью преобразования Фурье. Далее в этой главе мы расскажем о том, что именно в этих рассуждениях содержится ключ к пониманию того, почему преобразование Лапласа столь полезно.

Чтобы лучше разобраться с выражением (32.1), выберем несколько точек, лежащих в  $S$ -области, и постараемся исследовать взаимосвязь комплексных величин, вычисленных в этих точках, с сигналом, заданным во временной области. Прежде всего вспомним, каким образом отдельные точки частотной области связаны с временным сигналом. Каждой точке частотной области, определяемой параметром частоты  $\omega$ , соответствуют две гармонические функции:  $\cos(\omega t)$  и  $\sin(\omega t)$ . Вещественная часть комплексного числа, соответствующего данной точке, может быть найдена в результате умножения сигнала, заданного во временной области, на косинус частоты  $\omega$  и последующего интегрирования полученной функции на интервале  $-\infty \dots +\infty$ . Мнимая часть этого комплексного числа находится аналогично, за исключением того, что вместо косинуса используется синус. Если речь идет о комплексном преобразовании Фурье, то значения, полученные для отрицательных частот  $-\omega$ , являются комплексно-сопряжёнными по отношению к значениям, полученным для соответствующих положительных частот  $\omega$ , т. е. име-

ют равные вещественные и противоположные мнимые части. Преобразование Лапласа связано с дальнейшим обобщением этого принципа. На Рис. 32.2 изображены три пары точек  $S$ -плоскости:  $A$  и  $A'$ ,  $B$  и  $B'$ ,  $C$  и  $C'$ . Как и в случае комплексного частотного спектра, точки  $A$ ,  $B$  и  $C$  (положительные частоты) являются комплексно-сопряжёнными по отношению к точкам  $A'$ ,  $B'$  и  $C'$  (отрицательные частоты). Верхняя полуплоскость  $S$ -плоскости является зеркальным отражением нижней, причём обе полуплоскости находятся в однозначном соответствии с вещественным сигналом во временной форме. Другими словами, введение принципа равенства значений, вычисленных в комплексно-сопряжённых парах точек, гарантирует, что соответствующие сигналы во временной области являются вещественными.

Поскольку каждая такая пара характеризуется отличными от всех других пар значениями  $\sigma$  и  $\pm\omega$ , то ей соответствуют во временной области две уникальные функции:  $\cos(\omega t)e^{-\sigma t}$  и  $\sin(\omega t)e^{-\sigma t}$ . Например, паре  $A$  и  $A'$ , положение которой харак-



**Рис. 32.2.** Соответствие временных функций точкам  $S$ -плоскости. Каждая точка на  $S$ -плоскости определяется двумя параметрами:  $\sigma$  и  $\omega$ . Эти величины одновременно являются параметрами двух функций, определённых во временной области. Если мы накладываем принцип равенства на значения, вычисленные в комплексно-сопряжённых точках (таких как  $A$  и  $A'$ ,  $B$  и  $B'$ ,  $C$  и  $C'$ ), то двумя такими функциями во временной области будут синусоида и косинусоида частоты  $\omega$  с амплитудой, изменяющейся с течением времени по экспоненциальному закону, причём  $\sigma$  — показатель экспоненты.

теризуется координатами  $\sigma = 1.5$  и  $\omega = \pm 40$ , ставятся в соответствие функции  $\cos(40t)e^{-1.5t}$  и  $\sin(40t)e^{-1.5t}$ , которые представляют собой гармонические функции с убывающей с течением времени амплитудой, как это видно по **Рис. 32.2**. Синусоида и косинусоида, которым соответствует пара точек  $B$  и  $B'$ , имеют постоянную амплитуду, что объясняется равенством нулю параметра  $\sigma$ . Амплитуды синусоиды и косинусоиды, соответствующие паре точек  $C$  и  $C'$ , возрастают со временем по экспоненциальному закону, так как параметр  $\sigma$  является в данном случае отрицательным.

Значение комплексной величины, вычисленной в любой точке *S*-плоскости, разбивается на вещественную и мнимую части. Вещественная часть находится путём умножения временного сигнала на косинусоиду, амплитуда которой изменяется по экспоненциальному закону, и последующего интегрирования в интервале  $-\infty \dots \infty$ . Мнимая часть находится аналогично, с той лишь разницей, что вместо косинусоиды используется синусоида. Например, вещественная часть комплексной величины, соответствующей паре  $A$  и  $A'$ , определяется следующей формулой:

$$\operatorname{Re} X(\sigma = 1.5, \omega = \pm 40) = \int_{-\infty}^{\infty} x(t) \cos(40t) e^{-1.5t} dt.$$

На **Рис. 32.3** показан пример сигнала, определённого во временной области, а также его частотный спектр и результат, полученный при переходе в *S*-область. Во временной области сигнал представляет собой прямоугольный импульс единичной амплитуды и шириной две единицы. Как видим, спектр такого сигнала, полученный с помощью преобразования Фурье, имеет вещественную часть вида  $\sin x/x$  и нулевую мнимую часть. В *S*-области такому сигналу соответствует волнобразная двумерная функция, которая графически отображается в виде двух топографических поверхностей, соответствующих вещественной и мнимой частям комплексной функции. Математическая зависимость выражается следующим образом:

$$X(s) = \int_{-\infty}^{\infty} x(t) e^{-st} dt = \int_{-1}^1 e^{-st} dt.$$

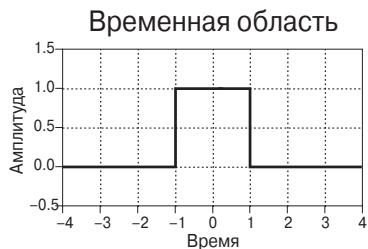
То есть в уравнении, описывающем преобразование Лапласа в форме выражения (32.1),  $x(t)$  заменяется единичной константой, а пределы интегрирования сужаются до границ отрезка, внутри которых сигнал  $x(t)$  отличен от нуля. После проведения интегрирования по переменной  $t$  получаем выражение, связывающее  $X(s)$  с комплексной координатой  $s$ , которая характеризует положение точки на *S*-плоскости:

$$X(s) = \frac{e^s - e^{-s}}{s}.$$

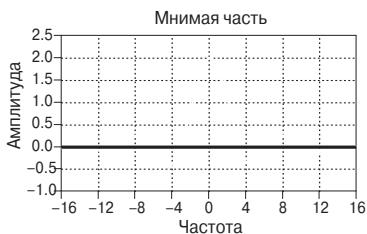
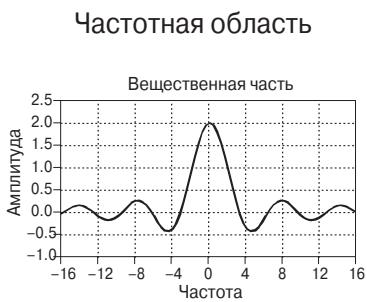
Хотя полученная запись является наиболее компактной, использование комплексных переменных затрудняет её понимание и не позволяет получить графическое представление, подобное показанному на **Рис. 32.3**. Поэтому удобно перейти от комплексной переменной  $s$  к эквивалентной сумме  $\sigma + j\omega$ , а затем разделить вещественную и мнимую части получившегося выражения:

$$\operatorname{Re} X(\sigma, \omega) = \frac{\sigma \cos(\omega) [e^\sigma - e^{-\sigma}] + \omega \sin(\omega) [e^\sigma + e^{-\sigma}]}{\sigma^2 + \omega^2},$$

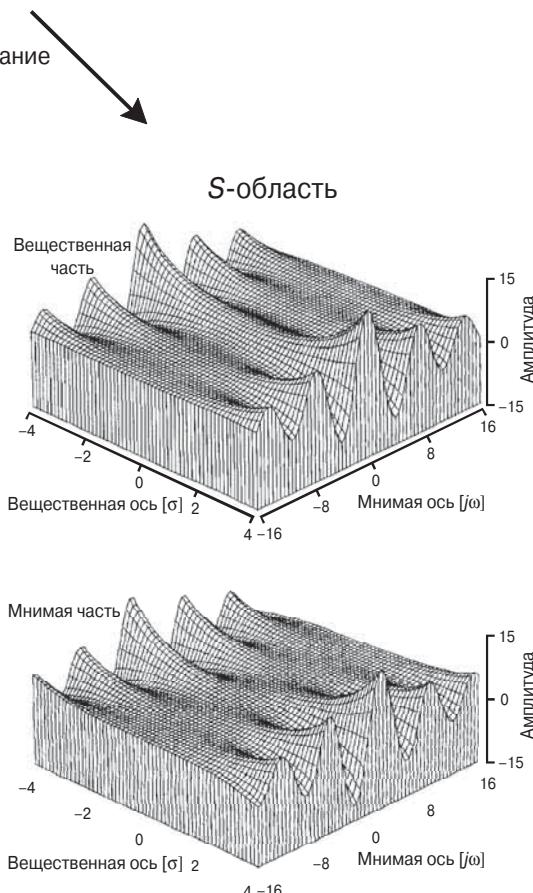
$$\operatorname{Im} X(\sigma, \omega) = \frac{\sigma \sin(\omega) [e^\sigma + e^{-\sigma}] - \omega \cos(\omega) [e^\sigma - e^{-\sigma}]}{\sigma^2 + \omega^2}.$$



Преобразование  
Фурье



**Рис. 32.3.** Представление сигнала во временной, частотной и *S*-области. От временной формы описания сигнала (в данном примере это прямоугольный импульс) с помощью преобразования Фурье выполняется переход в частотную область, а с помощью преобразования Лапласа — в *S*-область.



Топографические поверхности, изображённые на **Рис. 32.3**, являются графическим отображением этих уравнений. Уравнения получаются очень длинными, а их вывод оказывается весьма сложным. Как проверить, что применение данных

уравнений приведёт к правильным результатам? Один из возможных способов состоит в проверке того, что данные уравнения вырождаются в преобразование Фурье при ограничении области определения осью ординат на комплексной плоскости. Для этого достаточно задать нулевое значение параметру  $\sigma$  и упростить выражения:

$$\left. \operatorname{Re} X(\sigma, \omega) \right|_{\sigma=0} = \frac{2\sin(\omega)}{\omega}, \quad \left. \operatorname{Im} X(\sigma, \omega) \right|_{\sigma=0} = 0.$$

Полученные результаты полностью согласуются с графиками на Рис. 32.3 для частотной области, полученными с помощью преобразования Фурье.

## 32.2. Стратегия преобразования Лапласа

Чтобы вам было легче понять принцип использования преобразования Лапласа при решении практических задач, приведем пример, взятый из конкретной жизненной ситуации. Представьте, что вам пришлось ехать ночью из одного города в другой поездом. На карте путь кажется идеально прямым, но за окном та-кая тьма, что невозможно оценить, так ли это на самом деле. Не имея других за-нятий, вы обнаруживаете на стене высотомер и принимаете решение следить за изменениями его показаний в течение своей ночной поездки.

Спустя несколько часов вам эти наблюдения порядком надоедают и вы пытае-тесь завязать беседу с проводником. «Забавная местность, — говорите вы. — Получается так, что мы преимущественно всё время поднимаемся только выше и выше. Но несколько раз я наблюдал интересные отклонения от этого правила». Хотя со-вершенно очевидно, что проводнику совсем неинтересны ваши рассуждения, вы продолжаете: «В самом начале пути мы преодолели резкий подъём, а затем почти столь же резкий спуск. В дальнейшем наблюдалась неглубокая впадина». Предпо-лагая, что вы опасный, а может быть, просто чокнутый, проводник решает, что луч-ше ответить на ваши рассуждения: «Полагаю, вы абсолютно правы. Пункт назна-чения находится у основания большого горного массива. Этим объясняется по-стоянный подъём. По пути нам встретился небольшой выступ, расположенный в предгорье, а потом мы пересекли долину по самому её центру».

А теперь задумайтесь о том, насколько велика разница между вашим пред-ставлением о зависимости уровней высоты от пройденного поездом расстояния и картикой, которую нарисовал для вас проводник. Так как вы опираетесь на полу-ченные при наблюдении за высотомером результаты измерений, вы вправе ут-верждать, что обладаете полной информацией о характере такой зависимости. В свою очередь проводник обладает столь же полной информацией, но в более простой и интуитивно понятной форме: расположение гор и долин точно совпа-дает с обнаруженными вами повышениями и понижениями уровня земной по-верхности. В то время как ваше описание сигнала может состоять из тысяч отде-льных результатов измерений, проводник обходится знанием о наличии всего нескольких объектов.

Теперь перенесёмся в область электротехники. Допустим, целью нашего исследования является построение импульсной и частотной характеристик системы. Как уже говорилось ранее, любая из этих характеристик содержит исчерпывающую информацию о линейной системе. Но это вовсе не означает, что информация представлена в них в наиболее простой форме. В частности, частотная характеристика представляется как множество результатов измерения некоторого параметра, зависящих от частоты. Используя пример с поездом, частотную характеристику можно представить как путь, проходящий по меняющейся ландшафту земной поверхности, в таком случае рельеф этой окружающей поверхности можно отождествить с  $S$ -плоскостью.

Опираясь, как и ранее, на пример с поездом, вернёмся к Рис. 32.3 и зададим вопрос: «Как может помочь знание «рельефа»  $S$ -области в понимании частотной характеристики?» — Никак не поможет! На  $S$ -плоскости построены замечательные графики, но они не позволяют заглянуть в суть процессов, наблюдаемых в частотной области. Дело в том, что преобразование Лапласа создано для анализа особого класса временных сигналов, образованного множеством синусоидальных функций с изменяющейся по экспоненциальному закону амплитудой. Если преобразование Лапласа применить к сигналу какой-либо иной формы, скажем, к прямоугольному импульсу, изображённому на Рис. 32.3, то полученный в  $S$ -области результат оказывается бессмысленным.

Как упоминалось в начале главы, системы, для описания которых используется данный класс функций, широко распространены в науке и технике. Это связано с тем, что синусоидальные и экспоненциальные зависимости являются результатом решения дифференциальных уравнений, т. е. того самого раздела математики, который позволяет описать большую часть явлений окружающего нас физического мира. На дифференциальных уравнениях основаны, в частности, законы, которым подчиняются электрические схемы, распространение радиоволн, поступательное и вращательное движение, электрические и магнитные поля, теплопередача.

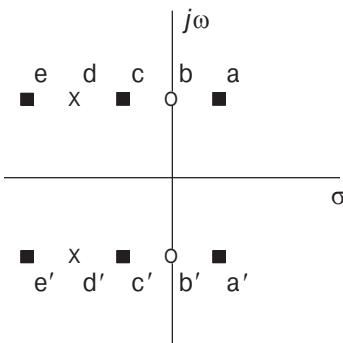
Предположим, мы пытаемся понять работу некоторой линейной системы, которая может быть описана с помощью дифференциальных уравнений, например электрической схемы. Решение дифференциальных уравнений представляет собой аналитический способ нахождения импульсной характеристики. С другой стороны, импульсная характеристика может быть найдена экспериментальным путём с использованием генератора импульсов, осциллографа, записывающего устройства и ряда других приборов. Перед тем как начать исследовать полученную импульсную характеристику, зададимся вопросом: «Что мы собираемся найти?» Несколько свойств импульсной характеристики известно без всяких дополнительных исследований. Во-первых, импульсная характеристика удовлетворяет условию каузальности, т. е. до поступления входного воздействия в момент времени  $t = 0$  реакция системы должна равняться нулю, чтобы выполнялся принцип причинно-следственной связи, распространяющийся на весь наш мир. Во-вторых, импульсная характеристика состоит из синусоидальных и экспоненциальных функций, потому что именно такие функции являются решением дифференциальных уравнений, описывающих работу системы. Как бы мы ни старались, мы никогда не найдем в реальном мире систему с импульсной характеристикой прямоугольной или треугольной формы. В-третьих, импульсная характеристика

должна иметь бесконечную длину, т. е. отличаться от нуля при изменении времени  $t$  в диапазоне  $0...+\infty$ . Это связано с тем, что амплитуды синусоид и косинусоид, затухающие по экспоненте, стремятся к нулю, но никогда его не достигают. Система считается *устойчивой*, если амплитуда импульсной характеристики уменьшается со временем, стремясь к нулю при  $t = +\infty$ . В некоторых случаях система может оказаться *неустойчивой*. Например, слишком сильное увеличение коэффициента обратной связи может приводить к самовозбуждению усилителей. При этом амплитуда импульсной характеристики теоретически увеличивается до бесконечности. Малейшие возмущения на входе таких систем вызывают неограниченные выходные реакции.

В целом математический аппарат преобразования Лапласа очень похож на аппарат преобразования Фурье: функции заданного вида умножаются на сигнал, определённый во временной области, а затем полученное выражение подвергается интегрированию. На первый взгляд стратегия преобразования Лапласа та же, что и у преобразования Фурье, — вычисление корреляции сигнала со множеством базисных функций, цель которого состоит в разложении сигнала по этому базису. Но на самом деле это не так! Несмотря на всю схожесть математического описания, логическое обоснование преобразований сильно отличается. Преобразование Лапласа определяет степень соответствия сигнала, представленного во временной области, синусоидальным сигналам разных частот и с разными показателями экспоненциального затухания амплитуд. Рассмотрим этот вопрос на примере.

На Рис. 32.5 на всех графиках центральной колонки показана импульсная характеристика узкополосного режекторного *RLC*-фильтра, о котором говорилось в Главе 30. В ней присутствует дельта-импульс в момент  $t = 0$ , за которым следует синусоидальная функция с убывающей по экспоненциальному закону амплитудой. На (a...e) иллюстрируется процесс определения степени соответствия импульсной характеристики синусоидальным функциям, различающимся скоростью затухания амплитуды. Каждый из испытательных сигналов определяется двумя параметрами — частотой колебаний  $\omega$  и показателем экспоненциальной огибающей  $\sigma$ . Другими словами, все эти сигналы соответствуют разным точкам *S*-плоскости (Рис. 32.4). Проверка соответствия заключается в умножении импульсной характеристики на разные испытательные сигналы и последующем интегрировании по переменной  $t$  на интервале  $-\infty...+\infty$ . Полученные результаты показаны в правой колонке. Наша цель заключается в том, чтобы найти такие комбинации параметров  $\sigma$  и  $\omega$ , при которых происходит вырождение передаточной функции. Вырождение передаточной функции может проявляться в двух случаях: при равенстве площади под кривой нулю или на границе области сходимости интеграла Лапласа. Все остальные возможные результаты не представляют интереса для анализа. Точки *S*-плоскости, для которых интегрирование приводит к получению нулевого результата, называются *нулями* системы. Точки, для которых значение интеграла достигает бесконечности (граница области сходимости преобразования Лапласа), называются *полюсами*. Полюсы и нули — это, по сути, те же горы и впадины в примере с поездом; по ним можно составить наглядное представление о свойствах «рельефа поверхности» по обе стороны от частотной характеристики.

Для начала посмотрим, что произойдёт, если подать на вход системы сигнал с убывающей амплитудой (a). В этом случае параметр  $\sigma > 0$ , что соответствует точ-

Схематическое изображение  $S$ -плоскости

**Рис. 32.4.** Пример, поясняющий смысл нулей и полюсов. Узкополосный  $RLC$ -фильтр имеет два полюса (обозначаются  $\times$ ) и два нуля (обозначаются  $o$ ). На данном рисунке показаны пять пар точек, соответствующих пяти испытательным сигналам. (Сам процесс испытания иллюстрируется на **Рис. 32.5**.)

кам правой полуплоскости  $S$ -плоскости. Так как с увеличением времени как импульсная характеристика, так и испытательный сигнал затухают, то их произведение тоже является функцией с убывающей амплитудой. Если функцию с экспоненциально убывающей амплитудой проинтегрировать на интервале  $-\infty \dots +\infty$ , то получится некоторая конечная величина. Этот случай не представляет особого интереса, так как при нём интеграл не равен нулю, а значит, не наблюдается вырождения передаточной функции. Кроме того, из рассмотренного примера видно, что устойчивая система не может иметь полюсы при  $\sigma > 0$ . То есть область расположения полюсов устойчивой системы ограничена левой полуплоскостью. Присутствие полюсов в правой полуплоскости говорит о неустойчивости системы, так как в этом случае импульсная характеристика неограниченно возрастает с течением времени.

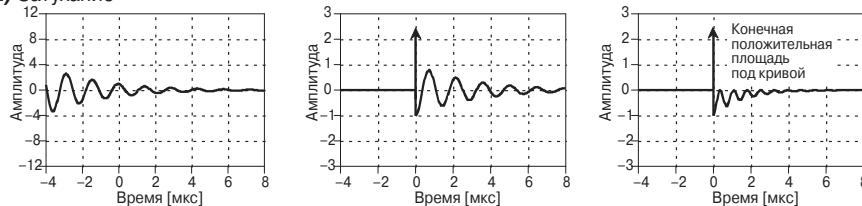
На **(б)** показан один из тех особых случаев, которые мы собирались обнаружить. Умножение такого испытательного сигнала на импульсную характеристику и последующее интегрирование позволяют получить равный нулю результат. Нулевой результат объясняется тем, что площадь под кривой, расположенная сверху от оси абсцисс (площадь, ограниченная дельта-функцией), равна по модулю и противоположна по знаку площади, расположенной снизу от оси абсцисс (площадь, ограниченной синусоидой). Пара значений  $\sigma$  и  $\omega$  задаёт координаты точки, называемой нулём системы. На  $S$ -плоскости (**Рис. 32.4**) нули обозначаются кружками ( $o$ ).

Следующее испытание иллюстрируется на **(в)**. Испытательный сигнал в этом случае представляет собой синусоидальную функцию, амплитуда которой возрастает со временем, но скорость её возрастания меньше, чем скорость затухания амплитуды импульсной характеристики. В результате произведения этих функций получается функция с убывающей амплитудой. Поэтому, как и в случае, показанном на **(а)**, значение интеграла является некоторой конечной величиной и не представляет особого интереса.

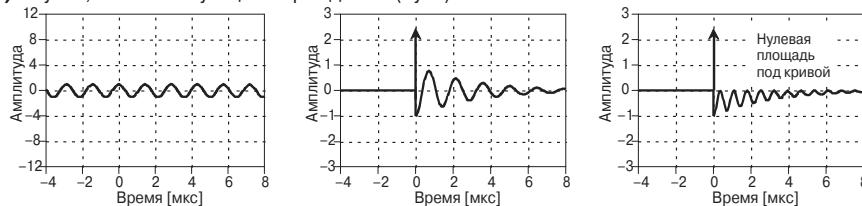
Пропустим пока **(г)** и перейдём к **(д)**. Скорость возрастания амплитуды испытательного сигнала, представленного на этом рисунке, больше, чем скорость убы-

Испытательный сигнал  $p(t)$  Импульсная характеристика  $h(t)$  Результат умножения  $p(t) \times h(t)$

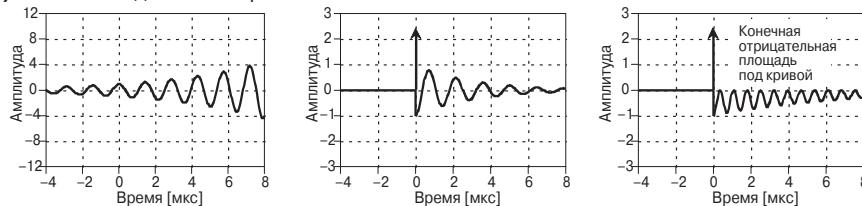
**а) Затухание**



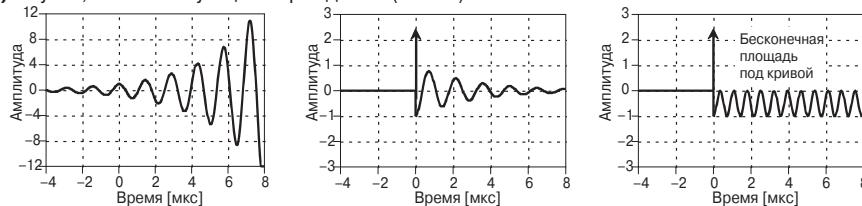
**б) Случай, соответствующий вырождению (нуль)**



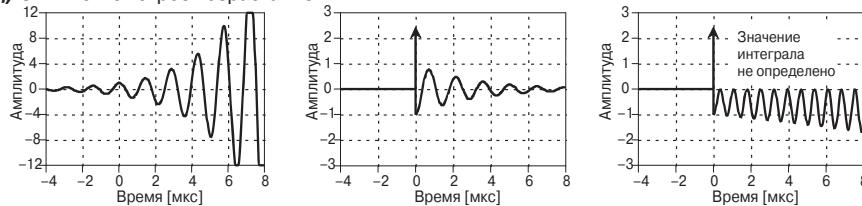
**в) Слишком медленное возрастание**



**г) Случай, соответствующий вырождению (полюс)**



**д) Слишком быстро возрастание**



**Рис. 32.5.** Исследование импульсной характеристики. Преобразование Лапласа можно рассматривать как исследование степени соответствия импульсной характеристики синусоидальным функциям с экспоненциально затухающей амплитудой. Испытательные сигналы, при которых достигается вырождение передаточной функции, называются либо нулями, либо полюсами. Слева показано пять испытательных сигналов, которые обрабатываются узкополосным режекторным фильтром (импульсная характеристика фильтра изображена на графиках в среднем столбце). Точки  $S$ -плоскости, соответствующие этим сигналам, показаны на Рис. 32.4.

вания амплитуды импульсной характеристики. После умножения получаем сигнал с неограниченно возрастающей амплитудой. Это означает, что с каждым очередным колебанием график ограничивает всё большую и большую область. В результате при рассмотрении функции на всей области определения  $t$ , т. е. в пределах  $-\infty \dots +\infty$ , её площадь под кривой оказывается не определена. Математики в таком случае говорят, что интеграл не сходится. Другими словами, преобразование Лапласа определено не на всей  $S$ -плоскости. Часть  $S$ -плоскости, где интеграл сходится, называется *областью сходимости*. Для некоторых математических задач важно знать область сходимости интеграла Лапласа. Однако для тех задач, которые рассмотрены в данной книге, такой информации не требуется. Достаточно только найти такие точки  $S$ -плоскости, где наблюдается один из двух вариантов вырождения передаточной функции.

На (г) скорость увеличения амплитуды испытательного сигнала точно соответствует скорости уменьшения амплитуды импульсной характеристики. Результатом произведения таких функций является функция, в которой присутствуют колебания постоянной амплитуды. Такой случай является промежуточным случаем между вариантами, показанными на (в и д), и соответствует границе области сходимости интеграла Лапласа, когда площадь подграфика «достигает бесконечности». Как уже говорилось, пары величин  $\sigma$  и  $\omega$ , соответствующих данному случаю, называются полюсами системы. На  $S$ -плоскости (Рис. 32.4) нули обозначаются крестиками (х).

### 32.3. Анализ электрических схем

До сих пор мы говорили о преобразовании Лапласа, используя графический метод описания. Теперь вы имеете представление о том, как выглядят базисные функции, по которым производится разложение сигнала, и о том, как они применяются. Такой подход позволяет получить самое интуитивное представление о преобразовании Лапласа. Преобразование Лапласа — математический метод, и поэтому его строгое описание не может быть выполнено без записи и преобразования математических формул. Но проблема состоит в том, что при выводе сложных алгебраических уравнений легко заблудиться в абстрактном мире комплексной алгебры и потерять при этом всякую связь с задачами реального мира. Поэтому в нашем дальнейшем повествовании нужно постараться избежать этих крайностей. Преобразование Лапласа — основной метод анализа электрических схем. Но имейте в виду, что любую систему, в основе описания которой лежат дифференциальные уравнения, можно анализировать при помощи преобразования Лапласа, а электрические схемы, о которых мы будем говорить далее, — только частный пример.

Самый примитивный способ анализа заключается в решении описывающих систему дифференциальных уравнений и получении импульсной характеристики этой системы, после чего эту характеристику остаётся преобразовать в  $S$ -область с помощью выражения (32.1). К счастью, имеется способ лучше: от дифференциальных уравнений переходят в  $S$ -область, а затем преобразуют получившиеся соотношения по установленным для этого правилам. Данный способ очень похож на метод векторного преобразования (Глава 30), в котором сопротивление, ин-

индуктивность и ёмкость описывались величиной полного сопротивления:  $R, j\omega L$  и  $1/j\omega C$ . В преобразовании Лапласа сопротивлению, индуктивности и ёмкости ставятся в соответствие следующие комплексные выражения:  $R, sL$  и  $1/sC$ . Обратите внимание на то, что векторное преобразование является частным случаем преобразования Лапласа, когда если в  $s = \sigma + j\omega$  параметр  $\sigma = 0$ , то  $R$  остаётся  $R$ ,  $sL$  переходит в  $j\omega L$ , а  $1/sC$  переходит в  $1/j\omega C$ .

Как и в Главе 30, будем рассматривать каждый из трёх элементов как отдельную систему, входным сигналом которой является закон изменения тока, протекающего через этот элемент, а выходным — закон изменения напряжения. Когда для обозначения сопротивления, индуктивности и ёмкости пользуются символическими обозначениями  $R, sL$  и  $1/sC$ , то подразумевают под ними отношение изображения выходного сигнала по Лапласу к изображению входного сигнала. Другими словами, символическое обозначение элемента электрической цепи есть отношение изображения напряжения на элементе к изображению тока, протекающего через этот элемент, вычисленных по Лапласу.

Пусть ток, протекающий через индуктивность, изменяется по косинусоидальному закону с единичной амплитудой и частотой  $\omega_0$ . Тогда напряжение на индуктивности находится в результате решения дифференциального уравнения:

$$u(t) = L \frac{d}{dt} i(t) = L \frac{d}{dt} \cos(\omega_0 t) = \omega_0 L \sin(\omega_0 t).$$

Если входное воздействие в виде тока поступает на вход системы, начиная с момента  $t = 0$ , то выходное воздействие, представляющее собой напряжение, тоже появляется, начиная с этого момента времени, т. е. при  $t < 0$   $i(t) = 0$  и  $u(t) = 0$ . Перейти от таких функций напряжения и тока, определённых во временной области, к их изображениям в  $S$ -области позволяют следующие уравнения, полученные из выражения (32.1):

$$I(s) = \int_0^{\infty} \cos(\omega_0 t) e^{-st} dt = \frac{\omega_0}{\omega_0^2 + s^2},$$

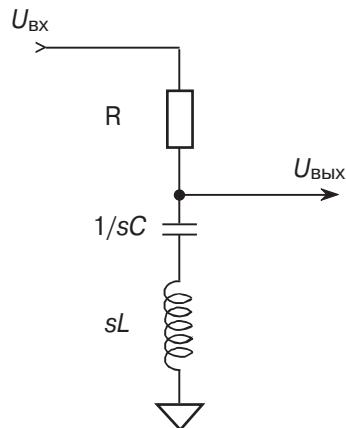
$$U(s) = \int_0^{\infty} \omega_0 L \sin(\omega_0 t) e^{-st} dt = \frac{\omega_0 L s}{\omega_0^2 + s^2}.$$

В завершение примера разделим изображение напряжения на изображение тока подобно тому, как это делается в законе Ома ( $R = U / I$ ):

$$\frac{U(s)}{I(s)} = \frac{\frac{\omega_0 L s}{\omega_0^2 + s^2}}{\frac{\omega_0}{\omega_0^2 + s^2}} = sL.$$

Итак, мы доказали, что отношение преобразования Лапласа напряжения на индуктивности к преобразованию Лапласа тока через эту индуктивность равно  $sL$ . Это утверждение верно также для любых других входных воздействий. Аналогичным образом можно доказать, что отношение изображений напряжения и тока для сопротивления равно  $R$ , а для ёмкости —  $1/sC$ .

На Рис. 32.6 показан пример схемы, для которой мы собираемся провести анализ на основе преобразования Лапласа. Это тот же самый RLC-фильтр из Главы 30. Поскольку такой анализ применим для всех электрических схем, рассмотрим его подробно, разбив на несколько шагов.



**Рис. 32.6.** Анализ режекторного фильтра в  $S$ -области. Первый шаг анализа заключается в переходе к символическим обозначениям сопротивления, индуктивности и ёмкости.

Шаг 1: Введение символьических выражений для каждого элемента схемы:  $R$  — сопротивление,  $sL$  — индуктивность,  $1/sC$  — ёмкость (Рис. 32.6).

Шаг 2: Нахождение отношения изображения выходного сигнала к входному сигналу —  $H(s)$ . Как говорилось в Главе 30, на этом шаге все действия полностью аналогичны тем, которые используются для схем, к которым применим закон Ома. «Сопротивления» элементов выражаются величинами  $R$ ,  $sL$  и  $1/sC$ , что позволяет использовать стандартные формулы преобразования последовательного и параллельного соединений, делителей напряжения и т. д. Рассматривая выбранную в этом примере RLC-схему как делитель напряжения (так же, как в Главе 30), находим  $H(s)$ :

$$H(s) = \frac{U_{\text{вых}}(s)}{U_{\text{вх}}(s)} = \frac{sL + 1/sC}{R + sL + 1/sC} = \frac{sL + 1/sC}{R + sL + 1/sC} \left[ \frac{s}{s} \right] = \frac{Ls^2 + 1/C}{Ls^2 + Rs + 1/C}.$$

Как вы помните из анализа Фурье, деление частотного спектра выходного сигнала на частотный спектр входного сигнала даёт частотную характеристику системы  $H(\omega)$ . Записанное выше уравнение является обобщением частотной характеристики на  $S$ -область. Величина  $H(s)$  называется *передаточной функцией* системы, которая определяется как отношение изображений по Лапласу выходного и входного сигналов, найденное при нулевых начальных условиях. Более того,  $H(s)$  представляет собой преобразование Лапласа импульсной характеристики системы точно так же, как  $H(\omega)$  представляет собой преобразование Фурье этой импульсной характеристики.

Пока всё сказанное повторяет методы, изложенные в предыдущей главе, за исключением того, что вместо  $j\omega$  используется  $s$ . Именно благодаря последнему факту возникает самое большое различие двух методов. Если бы мы связали дальнейший путь рассуждений с параметром  $j\omega$ , мы бы смогли построить график частотной характеристики, а затем постараться его исследовать. Но, с точки зрения математики, это тупиковый путь. При выборе преобразования Лапласа, напро-

тив, с этого момента только начинают обнаруживаться самые интересные факты. Нахождение  $H(s)$  является всего лишь началом анализа на основе преобразования Лапласа. Однако для дальнейших рассуждений  $H(s)$  требуется выразить в конкретной частной форме. С этой целью используются следующие два шага, в ходе которых выполняется ряд алгебраических преобразований.

Шаг 3: Приведение передаточной функции к стандартной форме в виде отношения многочленов. Передаточная функция должна иметь вид:

$$H(s) = \frac{as^2 + bs + c}{as^2 + bs + c}. \quad (32.2)$$

Передаточная функция в форме отношения многочленов.

Передаточную функцию можно выразить в форме выражения (32.2) всегда, когда система описывается линейными дифференциальными уравнениями. Прямоугольный импульс, изображённый на Рис. 32.3, не может быть описан с использованием дифференциальных уравнений, поэтому преобразование Лапласа прямоугольного импульса нельзя представить в форме выражения (32.2). Зато передаточную функцию любой электрической схемы, элементами которой являются сопротивление, ёмкость и индуктивность, можно выразить в форме отношения многочленов комплексной переменной  $s$ . В результате выполненных на шаге двух алгебраических преобразований для  $RLC$ -фильтра, выбранного нами в качестве примера, передаточная функция уже приобрела вид отношения двух многочленов второй степени:

$$H(s) = \frac{as^2 + bs + c}{as^2 + bs + c} = \frac{Ls^2 + 1/C}{Ls^2 + Rs + 1/C},$$

где  $a = L$ ,  $b = 0$ ,  $c = 1/C$ ,  $a = L$ ,  $b = R$ ,  $c = 1/C$ .

Шаг 4: Разложение многочленов числителя и знаменателя на множители. В результате числитель и знаменатель должны принять форму произведения одночленов. Если перемножить полученные одночлены, то числитель и знаменатель должны вновь приобрести форму первоначальных многочленов. Другими словами, уравнение преобразуется к следующему виду:

$$H(s) = \frac{(s - z_1)(s - z_2)(s - z_3) \dots}{(s - p_1)(s - p_2)(s - p_3) \dots}. \quad (32.3)$$

Передаточная функция, разложенная на одночлены. Такая форма записи передаточной функции позволяет выразить её через нули и полюсы.

Корни числителя  $z_1, z_2, z_3, \dots$  называются *нулями* передаточной функции, а корни знаменателя  $p_1, p_2, p_3, \dots$  — *полюсами*. Это те же самые нули и полюсы, с которыми мы уже встречались в данной главе. О том, как их использовать, расскажем в следующем разделе.

Разложение многочленов числителя и знаменателя не представляет проблем, когда порядки этих многочленов не превышают двух. То есть присутствие в уравнениях множителей  $s$  и  $s^2$  не создаёт таких трудностей, как наличие в них множителей  $s^3, s^4, s^5, \dots$ . Это связано с тем, что корни многочлена второго порядка вида

$ax^2 + bx + c$  выражаются достаточно простой и всем известной формулой, предназначенной для поиска корней квадратного уравнения:  $x = -b \pm \sqrt{b^2 - 4ac} / 2a$ . Используя только что описанный подход, можно преобразовать передаточную функцию режекторного фильтра, рассматриваемого в нашем примере, к следующему виду:

$$H(s) = \frac{(s - z_1)(s - z_2)}{(s - p_1)(s - p_2)},$$

где

$$z_1 = j/\sqrt{LC}, \quad p_1 = \frac{-R + \sqrt{R^2 - 4L/C}}{2L},$$

$$z_2 = -j/\sqrt{LC}, \quad p_2 = \frac{-R - \sqrt{R^2 - 4L/C}}{2L}.$$

Как видно из данного примера, любая система второго порядка имеет не более двух нулей и не более двух полюсов. Число полюсов системы равно числу элементов схемы, способных накапливать энергию, — индуктивностей и ёмкостей. Резисторы не обладают такой способностью. Число нулей не превышает числа полюсов.

Для разложения на множители многочленов более высоких порядков чаще всего не удается воспользоваться алгебраическими формулами, поэтому приходится обращаться к сложным численным методам. Одним из принципов построения электрических схем является каскадная структура, состоящая из последовательно соединенных звеньев второго порядка. Пример такой структуры представляет семейство аналоговых фильтров, рассмотренных в Главе 3. В частности, восьмиполюсный фильтр может быть представлен в форме последовательного соединения четырех каскадов, каждый из которых представляет собой фильтр второго порядка. При этом надо иметь в виду, что главной причиной перехода к каскадной структуре является ограниченность средств математического аппарата, а не электротехники.

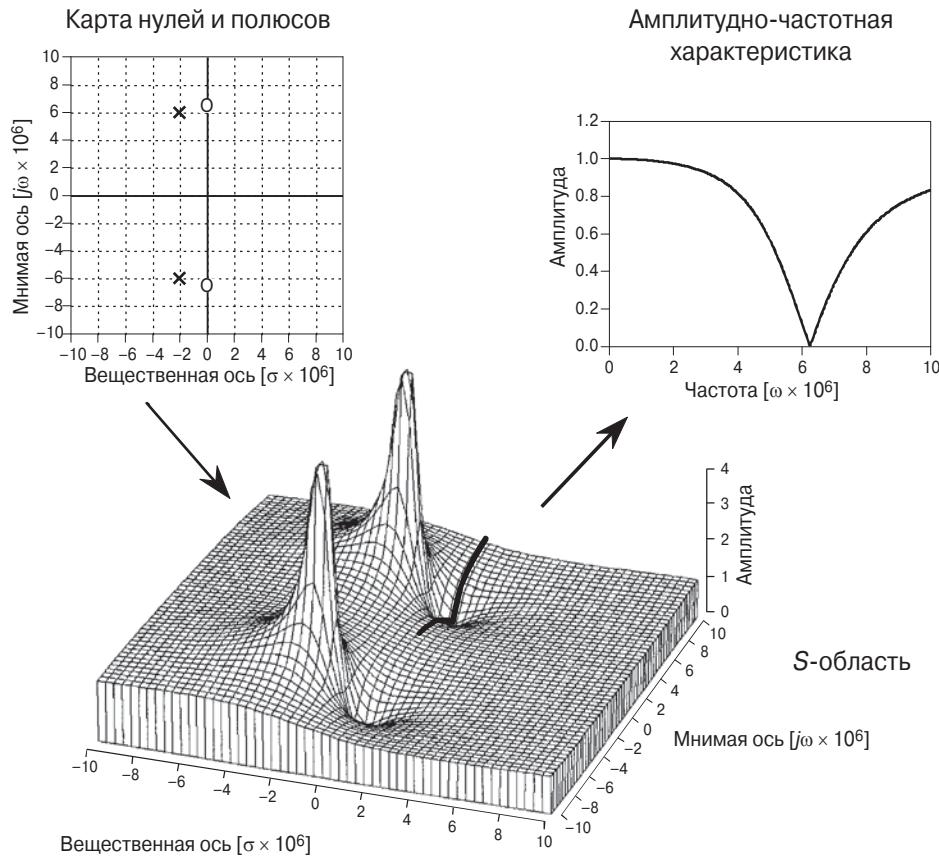
## 32.4. Значение нулей и полюсов

Чтобы сделать рассуждения менее абстрактными, подставим в полученные уравнения конкретные значения параметров схемы:  $R = 220$  Ом,  $L = 54$  мГн,  $C = 470$  пФ:

$$z_1 = 0 + j 6.277 \times 10^6, \quad p_1 = -2.037 \times 10^6 + j 5.937 \times 10^6,$$

$$z_2 = 0 - j 6.277 \times 10^6, \quad p_2 = -2.037 \times 10^6 - j 5.937 \times 10^6.$$

Расположение нулей и полюсов показано на Рис. 32.7. Нули изображены кружками, а полюсы — крестиками. Такое представление называют *картой нулей и полюсов*, представляющей наиболее широко распространенный способ графи-



**Рис. 32.7.** Нули и полюсы в  $S$ -области. На данном рисунке отражена связь между картой нулей и полюсов,  $S$ -областью и частотной характеристикой на примере узкополосного режекторного  $RLC$ -фильтра. Графики построены для схемы со следующими значениями параметров:  $R = 220 \text{ Ом}$ ,  $C = 470 \text{ пФ}$ ,  $L = 54 \text{ мкГн}$ . Частота подавления такого фильтра  $\omega = 6.277 \times 10^6 \text{ рад/с}$ , что соответствует примерно 1 МГц.

ческого отображения  $S$ -области. На Рис. 32.7 показано также топографическое представление  $S$ -плоскости. Для простоты график построен только для модуля преобразования Лапласа, но не забывайте, что есть ещё и фаза. Подобно тому как высокие горы и глубокие ущелья определяют форму поверхности земли, полюсы и нули определяют форму  $S$ -области. Но в отличие от гор и ущелий, форма и размеры искажений поверхности одинаковы для всех нулей и для всех полюсов. Единственное отличие любого нуля или полюса от всех других — это его положение на плоскости. Нули и полюсы имеют большое значение при описании свойств систем, поскольку в них содержится в краткой форме информация обо всех точках  $S$ -плоскости. То есть всего несколько числовых параметров несут информацию о свойствах всех характеристик системы. В случае  $RLC$ -фильтра достаточно четырёх комплексных параметров:  $z_1, z_2, p_1, p_2$ .

Наглядное представление о том, что такое нули и полюсы, позволяет получить следующий пример. Представьте, что по  $S$ -плоскости ползёт муравей. Лю-

бой точке плоскости, в которой может оказаться муравей (т. е. любому  $s$ ), соответствует некоторое значение передаточной функции  $H(s)$ . Значение передаточной функции является комплексным числом, и его можно выразить через модуль и фазу или через вещественную и мнимую части. Предположим, муравей приполз в точку, где расположен нуль передаточной функции. Результат измерения  $H(s)$  в этой точке имеет нулевые вещественную и мнимую части. То же самое получается из выражения для передаточной функции  $H(s)$ , записанного в форме выражения (32.3): при совпадении переменной  $s$  с любым из нулей передаточной функции один из множителей числителя обращается в нуль, что означает равенство нулю всего произведения.

Продолжая путешествовать, муравей приближается к точке, где расположен один из полюсов. По мере его приближения к этой точке вещественная и мнимая части измеряемой величины  $H(s)$  всё время увеличиваются. Снова обратимся к выражению (32.3). Если  $s$  равно любому из значений  $p$ , знаменатель обращается в нуль, а деление на нуль даёт бесконечность.

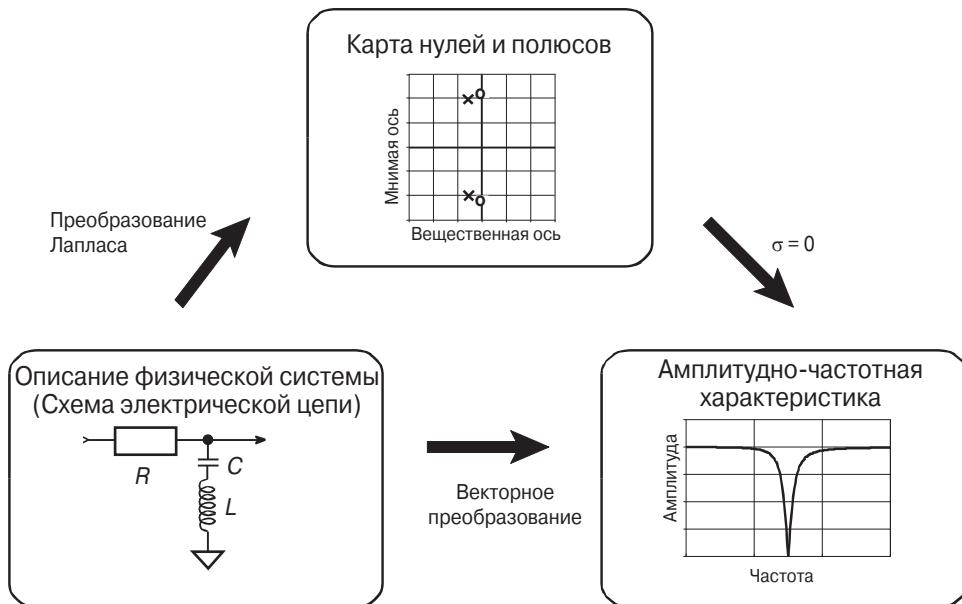
Исследовав особые точки, наш муравей начинает ползать по некоторой произвольно выбранной траектории. Величина  $H(s)$  в любой точке  $S$ -плоскости полностью определяется размещением нулей и полюсов, потому что эта странная местность не допускает присутствия никаких иных особенностей ландшафта. Рядом с полюсами наблюдается подъём, рядом с нулями — спуск.

С помощью выражения (32.3) можно оценить влияние множества нулей и полюсов на величину  $H(s)$  в любой точке  $S$ -плоскости. Вспомним, что расстояние между двумя точками равно разности их координат. Например,  $(s - z_0)$  — это расстояние между произвольно выбранной точкой плоскости  $s$  и нулем  $z_0$ . Из выражения (32.3) следует, что значение  $H(s)$  в любой точке  $S$ -плоскости равно произведению расстояний от неё до каждого из нулей, делённому на произведение расстояний от неё до каждого из полюсов.

Мы подошли к самому важному вопросу данной главы: каким образом знание нулей и полюсов способно дать более полное понимание частотной характеристики системы? Частотная характеристика совпадает со значениями  $H(s)$ , взятыми вдоль мнимой оси, что отмечено на топографической схеме (Рис. 32.7) жирной линией. Представьте, что наш муравей начинает своё путешествие из начала координат и ползёт по этой жирной полоске. В начале координат расстояние до нулей приблизительно равно расстоянию до полюсов. Из-за этого числитель и знаменатель выражения (32.3) взаимно компенсируются, в результате чего на низких частотах АЧХ равна единице. Ситуация почти не меняется до тех пор, пока муравей не окажется слишком близко к расположению нулей и полюсов. По мере приближения к нулю значение  $H(s)$  резко уменьшается, достигая нулевой величины, когда муравей приходит в нуль. После того как муравей пройдёт точку, в которой находится нуль, значение  $H(s)$  снова постепенно возвращается к единице. На этом наглядном примере можно показать, что ширина провала зависит от расстояния между полюсом и нулем.

Принцип использования преобразования Лапласа схематически показан на Рис. 32.8. Анализ начинается с описания работы некоторой физической системы, например электрической цепи. При желании можно с помощью векторного преобразования сразу перейти к частотной характеристике системы, как это было сделано в Главе 30. Альтернативой данному подходу является анализ цепи на ос-

нове преобразования Лапласа. Процедура такого анализа, состоящая из четырёх шагов, была описана выше. С помощью этого метода можно получить математическое выражение для передаточной функции  $H(s)$ , которая в графической форме представляется в виде карты нулей и полюсов. Вычисление значений  $H(s)$  вдоль мнимой оси, т. е. замена  $s$  на  $j\omega$ , позволяет найти частотную характеристику. Хотя оба метода приводят к одинаковым результатам, преобразование Лапласа имеет преимущество: промежуточный этап, связанный с построением карты нулей и полюсов, позволяет понять, почему система ведёт себя именно так и какие изменения следует в неё внести.



**Рис. 32.8.** Принцип использования преобразования Лапласа. Описанный в Главе 30 метод векторного преобразования (метод, в котором используются обозначения  $R$ ,  $j\omega L$ , и  $-j/\omega C$ ) позволяет рассчитать частотную характеристику непосредственно, если известны параметры всех элементов физической системы. В отличие от этого метода, преобразование Лапласа служит для перехода в  $S$ -область, для графического отображения которой чаще всего пользуются картой нулей и полюсов. Частотную характеристику можно получить на основе преобразования Лапласа, вычислив значения  $H(s)$  в точках  $S$ -области, расположенных на мнимой оси.

## 32.5. Расчёт фильтров в S-области

Наиболее важное практическое значение преобразования Лапласа связано с возможностью расчёта системы непосредственно в  $S$ -области. Такой расчёт разбивается на два этапа. На первом этапе определяют количество и местоположение нулей и полюсов. Это исключительно математическая задача, цель которой заключается в получении наилучшей частотной характеристики. На втором этапе разрабатывают схему электрической цепи, которая соответствует полученным в  $S$ -области математическим выражениям. Это во многом творческий этап, пос-

кольку для одного и того же размещения нулей и полюсов можно разработать множество самых разных схемотехнических решений.

Как уже упоминалось, если передаточная функция системы содержит более чем два полюса и два нуля, выполнить шаг 4 в методе анализа, основанном на преобразовании Лапласа, становится очень сложно. Выходом из такого затруднения служит переход к каскадной структуре. Например, фильтр, имеющий шесть полюсов, представляется в виде последовательного соединения трёх каскадов, каждый из которых содержит не более двух полюсов и двух нулей. Вследствие того что каждое из последовательно соединённых звеньев описывается передаточной функцией, содержащей квадратный многочлен в числитеle и квадратный многочлен в знаменателе, данный метод называется *расчётом с использованием биквадратных фильтров*.

На Рис. 32.9 изображена схема биквадратного фильтра в общем виде (ранее мы уже использовали такую схему в Главе 3). Это так называемая *схема Саллена–Ки*, которую впервые подробно описали в своих работах в середине 1950-х годов Р.П. Сален и И.Л. Ки. Хотя схема имеет несколько модификаций, изображённый на Рис. 32.9 вариант наиболее распространён: в нём присутствуют два сопротивления, две ёмкости и амплитудный усилитель с коэффициентом передачи 1...3. Недоступный во времена Салена и Ки элемент — амплитудный усилитель — сейчас легко реализуется на базе относительно дешёвого операционного усилителя при использовании нескольких дополнительных резисторов для организации обратных связей. Если проделать 4 шага описанной выше процедуры анализа, то можно выразить координаты нулей и полюсов через параметры компонентов электрической схемы:

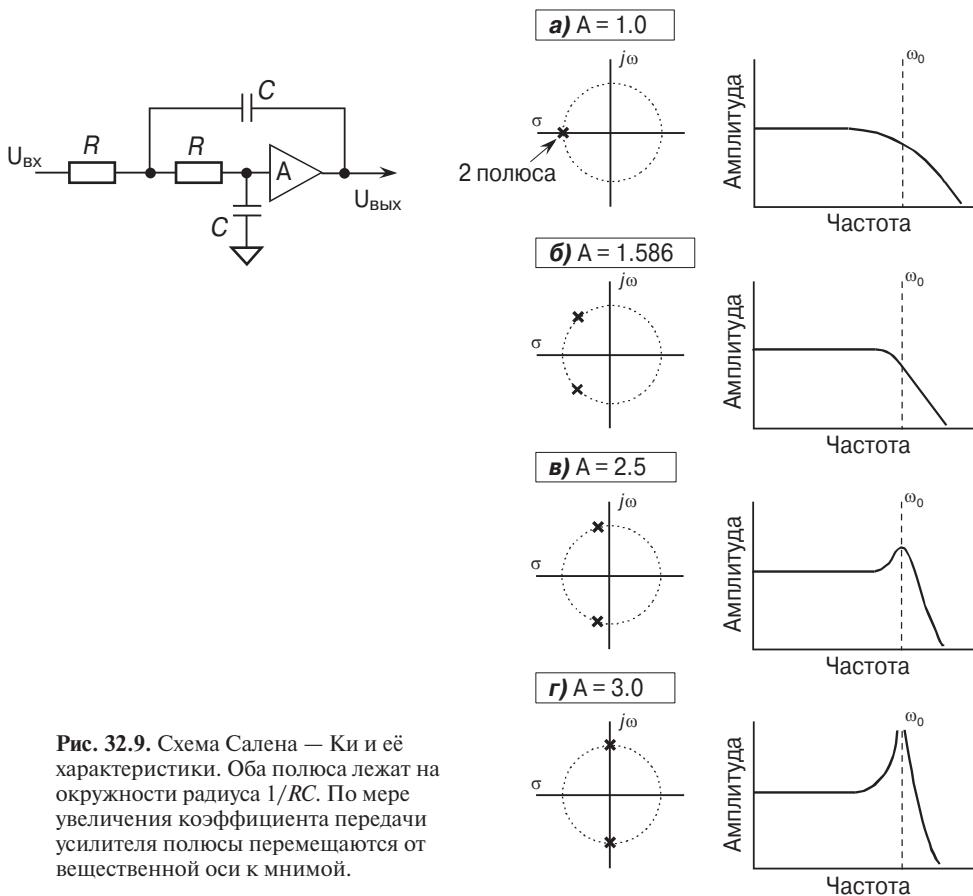
$$\sigma = \frac{A - 3}{2RC},$$

$$\omega = \frac{\pm \sqrt{-A^2 + 6A - 5}}{2RC}.$$
(32.4)

Расположение полюсов схемы Салена–Ки. Данные уравнения связывают координаты пары полюсов  $\omega$  и  $\sigma$  с параметрами элементов цепи: коэффициентом усиления  $A$ , сопротивлением  $R$  и ёмкостью  $C$ .

Из уравнений следует, что оба полюса всегда лежат на окружности радиуса  $1/RC$ . Их точное местоположение зависит от коэффициента передачи усилителя. На (а) показано, что при коэффициенте усиления, равном единице, оба полюса на карте нулей и полюсов совмещаются в одну точку, лежащую на вещественной оси. Судя по частотной характеристике, полученный фильтр является НЧ-фильтром с относительно медленным спадом в переходной зоне, разделяющей полосу пропускания и зону подавления. Частота среза для данной схемы, определяемая по уровню 0.707 ( $-3$  дБ), соответствует точке пересечения окружности с мнимой координатной осью:  $\omega_0 = 1/RC$ .

По мере увеличения коэффициента передачи усилителя полюсы перемещаются по окружности от вещественной оси к мнимой, что отражается на частотной характеристике системы. При коэффициенте передачи, равном 1.586, полюсы расходятся на угол 45 градусов (б), что делает спад АЧХ более крутым. Дальнейшее увеличение коэффициента передачи приводит к дальнейшему перемещению

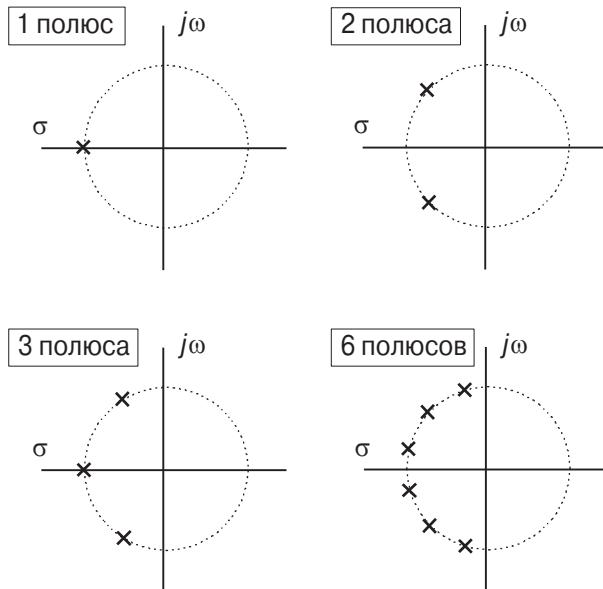


**Рис. 32.9.** Схема Салена — Ки и её характеристики. Оба полюса лежат на окружности радиуса  $1/RC$ . По мере увеличения коэффициента передачи усилителя полюсы перемещаются от вещественной оси к мнимой.

полюсов фильтра и, кроме того, к возникновению всплеска АЧХ на границе полосы пропускания: на Рис. 32.9в коэффициент передачи равен 2.5. Амплитуда всплеска продолжает расти вплоть до достижения коэффициентом передачи значения, равного 3. К этому моменту полюсы фильтра достигают мнимой оси (Рис. 32.9г), а всплеск, наблюдавшийся на АЧХ фильтра, увеличивается до бесконечности. На практике такая схема представляет собой генератор, способный вырабатывать собственные колебания. Если продолжить увеличение коэффициента передачи усилителя, полюсы попадут в правую полуплоскость S-плоскости. Как уже говорилось, это приведет к потере фильтром устойчивости и самовозбуждению схемы.

Используя схему Салена—Ки в качестве основного строительного блока, можно построить фильтры самых разных типов. Например, для получения низкочастотного фильтра *Баттерворта* все полюсы должны быть распределены равномерно на левой половине окружности (Рис. 32.10). Каждой паре комплексно-сопряжённых полюсов соответствует один каскад Салена—Ки. В Главе 3 говорилось, что фильтр Баттерворта является максимально гладким, что означает отсутствие в его АЧХ колебаний. При этом, чем больше полюсов у фильтра, тем круче спадает его характеристика. Поскольку все полюсы фильтра лежат

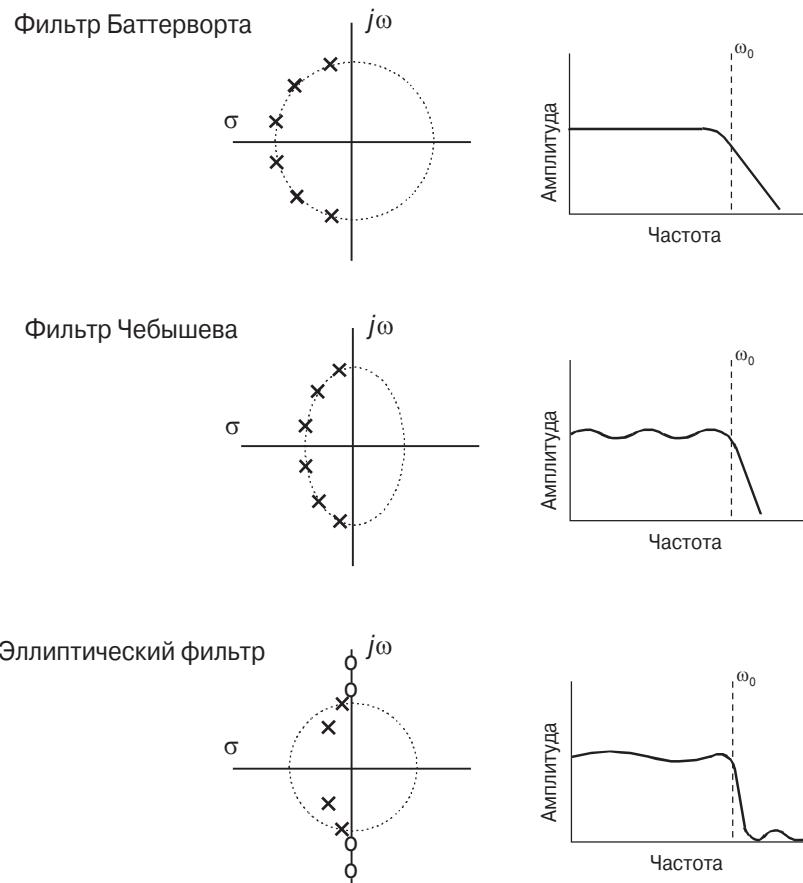
на одной окружности, то параметры  $R$  и  $C$  одинаковы для всех каскадов. Различие между каскадами наблюдается только в коэффициентах передачи усилителей. Почему размещение полюсов по окружности приводит к максимально гладкой частотной характеристики? Не старайтесь обнаружить какой-либо очевидный или интуитивно понятный ответ, такое свойство всего лишь вытекает из математических преобразований.



**Рис. 32.10.** Представление фильтра Баттервортса на  $S$ -плоскости. НЧ-фильтр Баттервортса получается в результате равномерного размещения полюсов вдоль левой половины окружности. Чем больше полюсов, тем круче спадает АЧХ.

На Рис. 32.11 показано, как следует изменить расположение полюсов фильтра Баттервортса, чтобы получить *фильтр Чебышева*. В Главе 3 говорилось, что фильтр Чебышева имеет более крутой спад АЧХ в переходной зоне, чем фильтр Баттервортса, но при этом допускает наличие колебаний АЧХ в полосе пропускания. В  $S$ -области фильтру Чебышева соответствует расположение полюсов по эллипсу. Чем больше этот эллипс отличается от окружности, тем сильнее колебания АЧХ в полосе пропускания фильтра и тем больше оказывается крутизна спада АЧХ в переходной зоне. При использовании каскадной формы построения фильтра на основе схемы Салена–Ки параметры сопротивлений и ёмкостей, используемых в разных каскадах, получаются разными.

Построение *эллиптического фильтра* — это ещё более сложный уровень. Эллиптическому фильтру соответствует самый крутой спад АЧХ, какой только возможен при заданном числе полюсов, но это достигается благодаря разрешению колебаний АЧХ как в полосе пропускания, так и в зоне подавления фильтра. В  $S$ -области построение эллиптического фильтра выражается в размещении нулей на мнимой оси, причём первые нули располагаются в непосредственной близости от частоты среза. Существует несколько разновидностей эллиптических фильтров, рассчитывать которые гораздо сложнее, чем фильтры Баттервортса и Чебышева. Дело в том, что размещение нулей и полюсов эллиптического фильтра



**Рис. 32.11.** Классические примеры размещения нулей и полюсов. Здесь показано размещение нулей и полюсов, характерное для трёх наиболее распространённых групп фильтров. Фильтры Баттервортта имеют полюсы, расположенные по окружности, что позволяет достичь максимально гладкой АЧХ. Фильтры Чебышева имеют полюсы, расположенные по эллипсу, что позволяет увеличить крутизну спада АЧХ, но приводит к возникновению колебаний АЧХ в полосе пропускания. У эллиптических фильтров на частотах, соответствующих зоне подавления, присутствуют нули. Это даёт возможность сделать спад АЧХ в переходной зоне ещё более крутым, но пульсации АЧХ у эллиптического фильтра присутствуют одновременно и в полосе пропускания, и в зоне подавления.

не может быть описано при помощи простейших геометрических фигур, а подчиняется математическим формулам, содержащим эллиптические функции и интегралы (отсюда происходит название этой группы фильтров).

Поскольку каждый биквадратный фильтр имеет два полюса, то при последовательном соединении получаются только *фильтры чётного порядка* (с двумя, четырьмя, шестью и т. д. полюсами). Для построения *фильтров нечётного порядка* (с одним, тремя, пятью и т. д. полюсами) требуется дополнительный элемент. Этот элемент — обычная *RC*-цепь, которой соответствует передаточная функция с единственным полюсом, расположенным на вещественной оси. Например, 9-полюсный фильтр можно составить из пяти каскадов: четырёх биквадратных фильт-

тров Салена–Ки и дополнительного каскада, состоящего из конденсатора и резистора.

Рассмотренные нами примеры размещения нулей и полюсов соответствуют НЧ-фильтрам, однако от них нетрудно перейти к схемам для фильтров с другими частотными характеристиками. Обычно сначала рассчитывают НЧ-фильтр, а затем производят математическое преобразование в  $S$ -области. Сначала вычисляют координаты полюсов НЧ-фильтра и записывают передаточную функцию  $H(s)$  в форме выражения (32.3). Тогда передаточная функция ВЧ-фильтра получается в результате замены каждого « $s$ » на « $1/s$ » и последующего упрощения выражения, которое требуется снова привести к форме выражения (32.3). Рассчитанные таким способом новые координаты полюсов и нулей соответствуют ВЧ-фильтру. Расчёт полосовых и режекторных фильтров в  $S$ -области связан с более сложными преобразованиями. О том, какие именно математические преобразования применяются в таких случаях, рассказывается в других книгах, посвящённых преобразованию Лапласа. Имейте в виду, что расчёт аналогового фильтра — это на 90% математическая задача и только на 10% задача электротехники.

К счастью, расчёт ВЧ-фильтров на основе каскадов Салена–Ки не требует сложных математических преобразований. Замена « $s$ » на « $1/s$ » в  $S$ -области соответствует в электрической схеме замене сопротивлений на конденсаторы и наоборот. На  $S$ -плоскости такая замена приводит к изменению расположения полюсов и появлению двух нулей в начале координат. В итоге частотная характеристика равна нулю на постоянной составляющей (на нулевой частоте), что и требуется для ВЧ-фильтра. При этом возможности схемы Саллена–Ки используются максимально: схема воспроизводит не только два полюса, но и два нуля.

# Глава 33

## Z-ПРЕОБРАЗОВАНИЕ

В теории цифровых систем  $Z$ -преобразование играет такую же важную роль, как и преобразование Лапласа в теории аналоговых. Оба преобразования возникли на основе одной и той же общей для них идеи поиска: корреляции исходной функциональной зависимости (оригинала) с разными функциями, образованными путём комбинации синусоидальных и экспоненциальных зависимостей, что позволяет выявить в дальнейшем нули и полюсы. Преобразование Лапласа определяется в  $S$ -области (на  $S$ -плоскости) и связано с дифференциальными уравнениями.  $Z$ -преобразование определяется в  $Z$ -области (на  $Z$ -плоскости) и связано с разностными уравнениями. Так что эти два подхода нельзя считать абсолютно идентичными: на  $S$ -плоскости задана прямоугольная система координат, а на  $Z$ -плоскости — полярная система. В то же время расчёту рекурсивного цифрового фильтра во многих случаях предшествует расчёт одного из классических аналоговых фильтров — Баттервортса, Чебышева или эллиптического, — после чего, в результате ряда математических преобразований, выполняется переход от аналоговой формы к цифровой. Такой подход уже был использован в программе расчёта фильтра Чебышева в Главе 20, а теперь будет рассмотрен нами подробнее на основе  $Z$ -преобразования.

### 33.1. Понятие Z-плоскости

Чтобы подчеркнуть сходство *преобразования Лапласа* и *Z-преобразования*, покажем сначала, как осуществить переход от одного из них к другому. По данному в предыдущей главе определению, преобразование Лапласа связывает временную область и  $S$ -область описания сигнала:

$$X(s) = \int_{-\infty}^{\infty} x(t)e^{-st} dt,$$

где  $x(t)$  — сигнал во временной области (оригинал), а  $X(s)$  — его изображение на  $S$ -плоскости. Как было показано в предыдущей главе, преобразование Лапласа позволяет перенести анализ сигнала из временной области в область синусоид и косинусоид с экспоненциально спадающими амплитудами. Это не сложно понять, если представить комплексную переменную  $s$  эквивалентным выражением  $\sigma + j\omega$ . При этом преобразование Лапласа определяется в иной форме:

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

Когда мы имеем дело с вещественными сигналами (т. е. в большинстве встречающихся на практике случаев), изображения, полученные в верхней и нижней полуплоскостях в  $S$ -области, являются симметричными по отношению друг к другу. Множитель  $e^{-j\omega}$  включает в себя синусоиду и косинусоиду. Присутствующие в данном уравнении параметры  $\sigma$  и  $\omega$  являются координатами некоторой точки на комплексной  $S$ -плоскости. Значение интегральной зависимости в каждой точке  $S$ -плоскости тоже является комплексным числом. Чтобы найти вещественную часть значения функции в точке  $\sigma + j\omega$ , нужно умножить сигнал  $x(t)$  на косинусоидальную функцию частотой  $\omega$ , амплитуда которой спадает по экспоненциальному закону с коэффициентом затухания  $\sigma$ . Тогда вещественная часть  $X(\sigma, \omega)$  находится путём интегрирования полученной функции. Минимая часть  $X(\sigma, \omega)$  находится аналогично, но косинус следует заменить синусом. Если эти рассуждения кажутся слишком сложными, то перед тем, как читать дальше, рекомендуем повторить материал предыдущей главы.

Переход от преобразования Лапласа к  $Z$ -преобразованию можно выполнить в три этапа. Первый шаг наиболее понятен и выражается в преобразовании непрерывного сигнала в дискретный путём замены переменной непрерывного времени  $t$  переменной дискретного времени  $n$  и одновременной замены интеграла суммой:

$$X(\sigma, \omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-\sigma n} e^{-j\omega n}.$$

Не запутайтесь: круглые скобки в записи  $X(\sigma, \omega)$  показывают, что данная функция после выполненного преобразования не становится дискретной, а остается, как и раньше, непрерывной. Хотя  $x[n]$  — дискретная функция, переменные  $\sigma$  и  $\omega$  по-прежнему определяются на непрерывном множестве.

Второй шаг — преобразование формы записи экспоненциального множителя, причём возможны два варианта:

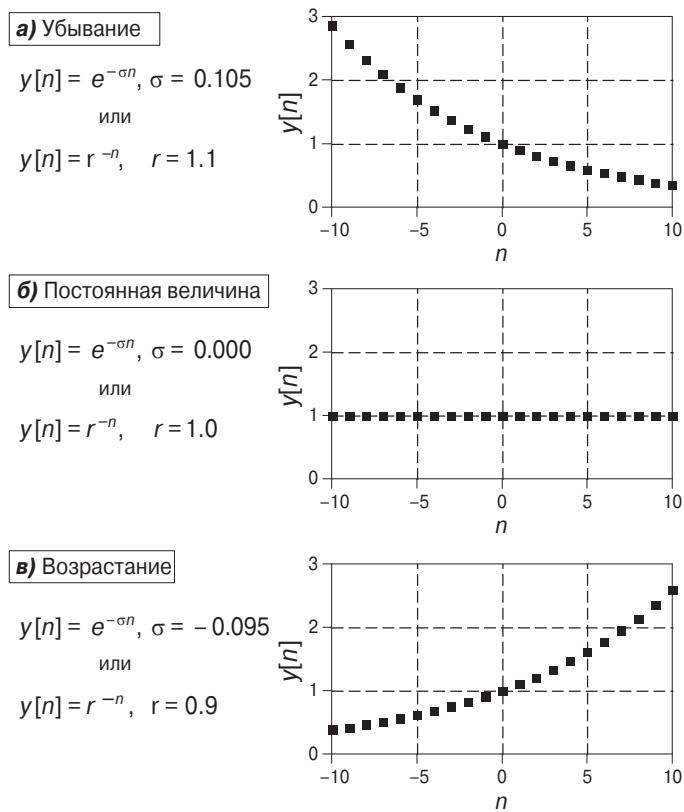
$$y[n] = e^{-\sigma n} \text{ или } y[n] = r^{-n}.$$

Как показано на Рис. 33.1, обеим функциям соответствуют экспоненциальные кривые. Параметр  $\sigma$  представляет собой коэффициент затухания. Если  $\sigma$  — положительная величина, то, чем больше этот коэффициент, тем быстрее убывает функция. Если параметр  $\sigma$  оказывается отрицательной величиной, то, напротив, чем он меньше, тем быстрее функция возрастает. Если  $\sigma = 0$ , то функция равна единице на всей области определения.

При использовании второй формы записи коэффициент затухания определяется параметром  $r$ . Если  $r > 1$ , зависимость убывает; если  $r < 1$ , зависимость возрастает. При  $r = 1$  все отсчёты последовательности становятся одинаковыми и равны 1. Оба соотношения позволяют задать одни и те же зависимости, но в различной форме. Переход от одной формы записи к другой осуществляется следующим образом:

$$r^{-n} = [e^{\ln(r)}]^{-n} = e^{-n \ln(r)} = e^{-\sigma n},$$

где  $\sigma = \ln(r)$ .



**Рис. 33.1.** Экспоненциальные сигналы. Экспоненциальная зависимость может быть представлена в двух разных формах, одна из которых используется в преобразовании Лапласа, а другая — в Z-преобразовании.

После выполнения второго шага на пути от преобразования Лапласа к Z-преобразованию получаем следующее уравнение:

$$X(r, \omega) = \sum_{n=-\infty}^{\infty} x[n] r^{-n} e^{-j\omega n}.$$

Хотя оно вполне соответствует Z-преобразованию, имеется возможность его упростить введением комплексной переменной. Ранее для преобразования Лапласа мы использовали переменную  $s = \sigma + j\omega$ . Для Z-преобразования комплексная переменная вводится следующим равенством:

$$z = re^{j\omega}.$$

Третий шаг связан с заменой двух вещественных переменных  $r$  и  $\omega$  комплексной переменной  $z$ , в которую  $r$  входит как амплитуда, а  $\omega$  — как фаза. В итоге, Z-преобразование определяется уравнением

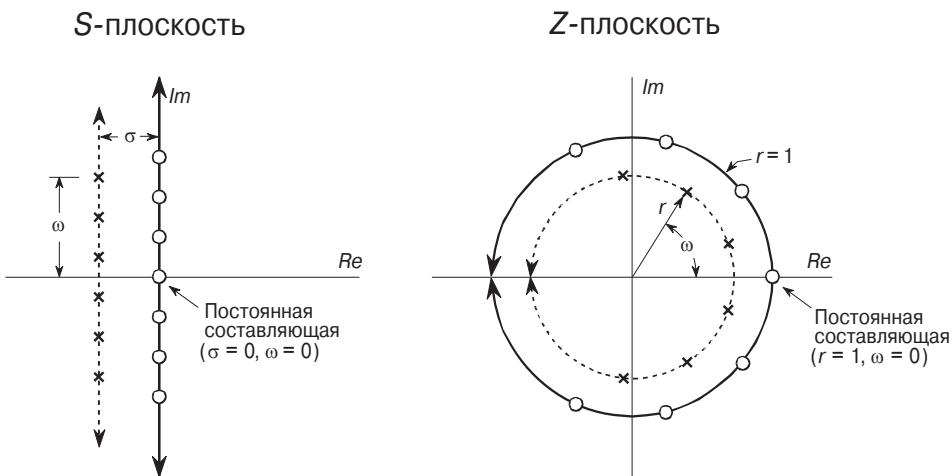
$$X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n}. \quad (33.1)$$

Z-преобразование. Z-преобразование связывает сигнал  $x[n]$ , представленный во временной форме (оригинал), и его изображение  $X(z)$  на Z-плоскости.

Почему  $Z$ -преобразование использует  $r^n$  вместо  $e^{-\sigma n}$  и  $z$  вместо  $s$ ? В главе 19 был описан процесс расчёта рекурсивных цифровых фильтров, основная цель которого заключается в определении их весовых коэффициентов. Для анализа полученных систем в  $Z$ -области нам необходимо иметь возможность осуществлять переход от набора весовых коэффициентов к передаточной функции, определённой в  $Z$ -области, а затем — обратный переход. Как будет показано ниже, использование в  $Z$ -преобразовании  $r^n$  и  $z$  позволяет достичь наиболее простого перехода между этими двумя важнейшими формами описания цифровых звеньев.

Разница между  $S$ -плоскостью, которую использует преобразование Лапласа, и  $Z$ -плоскостью, которое использует  $Z$ -преобразование, поясняется на Рис. 33.2. Координатами точки на комплексной  $S$ -плоскости являются два параметра преобразования Лапласа: по оси абсцисс  $\sigma$  — коэффициент затухания экспоненты, по оси ординат  $\omega$  — круговая частота колебания. Причем эти два параметра задают прямоугольную систему координат на плоскости, так как комплексная переменная представлена в алгебраической форме:  $s = \sigma + j\omega$ .

Параметры  $r$  и  $\omega$  задают полярную систему координат на  $Z$ -плоскости: расстояние точки от начала координат  $r$  непосредственно связано с коэффициентом затухания экспоненты, а угловая координата  $\omega$ , отсчитываемая от положительного направления оси абсцисс, является круговой частотой колебания. Такая система координат соответствует выражению  $z = re^{-j\omega}$ . Таким образом, система координат определяется способом объединения двух вещественных параметров, входящих в состав комплексной переменной.



**Рис. 33.2.** Различие между  $S$ -плоскостью и  $Z$ -плоскостью. На  $S$ -плоскости задана прямоугольная система координат, в которой по вещественной оси (абсцисс) откладывается параметр  $\sigma$ , а по мнимой оси (ординат) — параметр  $\omega$ . На  $Z$ -плоскости задана полярная система координат, в которой  $r$  — расстояние от начала координат, а  $\omega$  — угол, отложенный от положительного направления оси абсцисс. Для примера на  $S$ -плоскости показано несколько нулей и полюсов, расположенных вдоль вертикальных прямых; на  $Z$ -плоскости данным вертикальным прямым соответствуют концентрические окружности.

В результате такого отличия вертикальным линиям  $S$ -плоскости соответствуют концентрические окружности на  $Z$ -плоскости. Для примера на Рис. 33.2 показано некоторое произвольно заданное размещение нулей и полюсов на  $S$ -плоскости, при котором они располагаются вдоль вертикальных прямых. На  $Z$ -плоскости этим прямым соответствуют концентрические окружности. Данный факт легко проверить, используя выведенное ранее уравнение:  $\sigma = \ln(r)$ . Точкам вертикальной оси на  $S$ -плоскости, в которых  $\sigma = 0$ , соответствует на  $Z$ -плоскости *единичная окружность*, где  $r = 1$ . Аналогично вертикальные линии, лежащие в левой полуплоскости, могут быть сопоставлены концентрическим окружностям, расположенным внутри круга единичного радиуса, а вертикальные линии правой полуплоскости — окружностям, расположенным вне этого круга. Другими словами, левая полуплоскость переходит в область, ограниченную единичной окружностью, а правая — в область, находящуюся вне единичной окружности. Кроме того, аналогичным образом можно получить и условие устойчивости цифровых систем: цифровая система является устойчивой, если все её полюсы расположены внутри единичной окружности. Если сигнал является чисто вещественным (что справедливо в большинстве практических ситуаций), то верхняя и нижняя полу平面 на  $Z$ -плоскости оказываются симметричными по отношению друг к другу, точно так же как это было на  $S$ -плоскости.

Обратите внимание на то, как задействован параметр  $\omega$ , имеющий смысл круговой частоты, в двух разных преобразованиях. Частота непрерывного синусоидального сигнала может меняться от нуля (постоянная составляющая) до бесконечности. Частота дискретного сигнала не превышает половины частоты дискретизации, т. е. его приведенная круговая частота меняется в диапазоне  $0\dots\pi$ , так как  $\omega = 2\pi f$ . Такой диапазон находится в точном соответствии с полярной системой координат  $Z$ -плоскости. При этом положительным частотам соответствуют точки с угловой координатой  $0\dots\pi$ , а отрицательным — точки с угловой координатой  $0\dots-\pi$ . Так как частота по-разному учитывается на  $S$ -плоскости и  $Z$ -плоскости, то в некоторых работах частота дискретного сигнала обозначается заглавной буквой  $\Omega$ , чтобы отличить её от частоты непрерывного сигнала  $\omega$ . В этой книге используется один и тот же символ  $\omega$  как для непрерывных, так и для дискретных сигналов.

На  $S$ -плоскости значения преобразования Лапласа, взятые в точках, лежащих на оси ординат, совпадают с комплексной частотной характеристикой системы. То есть преобразование Лапласа при  $\sigma = 0$  совпадает с преобразованием Фурье. Точно также для дискретных сигналов комплексная частотная характеристика системы и  $Z$ -преобразование, взятое на единичной окружности, совпадают друг с другом. Чтобы доказать это утверждение, достаточно подставить в выражении (33.1)  $r = 1$  и получить уравнение ДПФ апериодической последовательности. Нулевая частота соответствует точке  $(1,0)$ , положительные частоты отсчитываются против часовой стрелки, а отрицательные — в противоположном направлении. Частоты  $\omega = \pi$  и  $\omega = -\pi$  совпадают. Такое отображение частот идеально согласуется с периодической формой спектра дискретного сигнала. После того как угловая частота превышает  $\pi$ , начинают повторяться в обратном порядке частоты из диапазона  $0\dots\pi$ . Продолжая увеличивать частоту, мы снова и снова получаем одни и те же значения.

## 33.2. Анализ рекурсивных систем

В Главе 19 для описания *рекурсивных фильтров* использовалось разностное уравнение:

$$\begin{aligned} y[n] &= a_0x[n] + a_1x[n-1] + a_2x[n-2] + \dots \\ &\dots + b_1y[n-1] + b_2y[n-2] + b_3y[n-3] + \dots \end{aligned} \quad (32.2)$$

Разностное уравнение.

В этом выражении  $a$  и  $b$  — *весовые коэффициенты* фильтра, а  $x[ ]$  и  $y[ ]$  — входной и выходной сигналы соответственно. Очевидное достоинство такой формы записи состоит в том, что она позволяет легко описать последовательность арифметических операций, которую нужно воплотить в программу. Другое важное преимущество заключается в непосредственной связи отсчётов выходной и входной последовательностей. Разностные уравнения играют столь же важную роль при описании цифровых систем, как и дифференциальные уравнения при описании непрерывных. Располагая разностным уравнением, можно построить карту нулей и полюсов; найти все основные характеристики системы: импульсную, переходную, частотную и т. д.

Прежде всего выполним  $Z$ -преобразование обеих частей выражения (33.2), чтобы затем перейти к анализу фильтра в  $Z$ -области. После изрядного количества алгебраических преобразований из получившегося выражения можно выразить отношение  $Y(z)/X(z)$ , т. е. отношение  $Z$ -преобразования выходной последовательности к  $Z$ -преобразованию входной последовательности при нулевых начальных условиях. Данное отношение называется передаточной функцией системы и обозначается  $H(z)$ :

$$H(z) = \frac{a_0 + a_1z^{-1} + a_2z^{-2} + a_3z^{-3} + ?}{1 - b_1z^{-1} - b_2z^{-2} - b_3z^{-3} - ?}. \quad (33.3)$$

Передаточная функция в виде отношения полиномов. Коэффициенты полиномов равны соответствующим весовым коэффициентам фильтра.

Это первая из двух основных форм записи передаточной функции. Её достоинством является простота построения, так как весовые коэффициенты фильтра входят в неё непосредственно, без изменений. Пусть весовые коэффициенты фильтра известны и заданы в виде таблицы:

$$\begin{array}{ll} a_0 = 0.389 & \\ a_1 = -1.558 & b_1 = 2.161 \\ a_2 = 2.338 & b_2 = -2.033 \\ a_3 = -1.558 & b_3 = 0.878 \\ a_4 = 0.389 & b_4 = -0.161 \end{array}$$

Не нужно абсолютно никаких алгебраических преобразований, чтобы написать передаточную функцию:

$$H(z) = \frac{0.389 - 1.558z^{-1} + 2.338z^{-2} - 1.558z^{-3} + 0.389z^{-4}}{1 - 2.161z^{-1} + 2.033z^{-2} - 0.878z^{-3} + 0.161z^{-4}}.$$

Важно заметить, что коэффициенты  $b$  входят в выражение (33.3) с отрицательным знаком. Однако очень многие авторы в своих работах определяют передаточную функцию так, что коэффициенты  $b$  входят в неё с положительным знаком, но при этом они оказываются противоположными по знаку соответствующим весовым коэффициентам. В результате такого разнотечения с вероятностью 50:50 может возникнуть путаница, а фильтр окажется неустойчивым.

В выражении (33.3) присутствуют отрицательные степени переменной  $z$ :  $z^{-1}$ ,  $z^{-2}$ ,  $z^{-3}$  и т. д. После того как выражение для передаточной функции уже записано, полиномы числителя и знаменателя несложно преобразовать к обычной форме, в которой переменная  $z$  присутствует в положительной степени:  $z^1$ ,  $z^2$ ,  $z^3$  и т. д. Умножая числитель и знаменатель на  $z^4$ , получим:

$$H(z) = \frac{0.389z^4 - 1.558z^3 + 2.338z^2 - 1.558z + 0.389}{z^4 - 2.161z^3 + 2.033z^2 - 0.878z + 0.161}.$$

Когда многочлены числителя и знаменателя передаточной функции определяются через положительные степени переменной  $z$ , то с ними проще оперировать. К тому же для многих преобразований требуется именно такая форма записи. Почему же не переписать выражение (33.3) в другой форме, в которую  $z$  входит только в положительной степени? К сожалению, это невозможно. Чтобы выразить передаточную функцию через положительные степени  $z$ , нужно предварительно узнать наибольший порядок многочленов числителя и знаменателя (в нашем примере порядки числителя и знаменателя равны 4), а для этого нужно знать, сколько весовых коэффициентов содержит фильтр. Следовательно, выразить передаточную функцию только через положительные степени переменной  $z$  невозможно для произвольного числа весовых коэффициентов.

Использование передаточной функции по сравнению с разностным уравнением, намного расширяет возможности разработчика цифровых фильтров. С её помощью можно объединять фильтры малых порядков при каскадной и параллельной формах построения в единую структуру, проектировать фильтры на основе карты нулей и полюсов, осуществлять переход от аналоговых фильтров к эквивалентным цифровым структурам и т. д. Все эти действия производятся в  $Z$ -области путём выполнения простых алгебраических операций над комплексными числами: умножения, сложения и разложения на множители. После всех преобразований новая передаточная функция снова имеет форму выражения (33.3), позволяя легко перейти к разностному уравнению.

Важное свойство передаточной функции при описании цифровых систем в  $Z$ -области заключается также и в том, что она может быть легко выражена через *нули* и *полюсы*. В результате появляется вторая из двух основных форм записи передаточной функции:

$$H(z) = \frac{(z - z_1)(z - z_2)(z - z_3)\dots}{(z - p_1)(z - p_2)(z - p_3)\dots}. \quad (33.4)$$

Передаточная функция, выраженная через нули и полюсы.

Все нули ( $z_1, z_2, z_3, \dots$ ) и полюсы ( $p_1, p_2, p_3, \dots$ ) являются комплексными числами. От выражения (33.4) легко перейти к выражению (33.3): достаточно перемен-

жить все одночлены числителя, а потом все одночлены знаменателя и представить получившиеся выражения в форме многочленов. Хотя такое преобразование требует большого количества алгебраических действий, его реализация в виде машинной программы очень проста. Переход от выражения (33.3) к выражению (33.4) осуществлять намного труднее, так как в этом случае приходится раскладывать многочлены высоких порядков на множители. Как говорилось в Главе 32, разложение многочленов на множители можно легко выполнить, воспользовавшись известной формулой вычисления корней квадратного уравнения, но только в тех случаях, когда порядок передаточной функции не превышает двух (т.е. независимая переменная  $z$  присутствует в степенях не более 2). Если порядок передаточной функции превышает 2, то для разложения многочленов на множители алгебраические методы неприменимы и приходится использовать сложные численные методы. К счастью, на практике необходимость в таком преобразовании возникает очень редко, поскольку расчёт цифровых фильтров начинается с указания значений нулей и полюсов в форме выражения (33.4), а завершается получением весовых коэффициентов передаточной функции вида выражения (33.3).

Как и любое комплексное число, нули и полюсы могут быть записаны и в полярной, и в алгебраической форме. Полярная форма больше соответствует полярной системе координат  $Z$ -плоскости. С другой стороны, при выполнении многих математических действий предпочтительнее оказывается алгебраическое представление комплексных чисел, т.е. с числами вида  $\sigma + j\omega$  проще проводить вычисления, чем с числами вида  $re^{j\phi}$ .

Рассмотрим данные уравнения на примере расчёта узкополосного режекторного фильтра второго порядка. Расчёт выполняется в четыре этапа:

- 1) размещаем нули и полюсы на  $Z$ -плоскости;
- 2) записываем передаточную функцию в форме выражения (33.4);
- 3) преобразуем передаточную функцию к форме выражения (33.3);
- 4) получаем весовые коэффициенты, необходимые для программной реализации фильтра.

Пусть режекторный фильтр содержит два нуля и два полюса, которые размещаются на карте нулей и полюсов так, как показано на Рис. 33.3. Эти нули и полюсы являются попарно комплексно-сопряжёнными и могут быть записаны в одной из следующих двух форм:

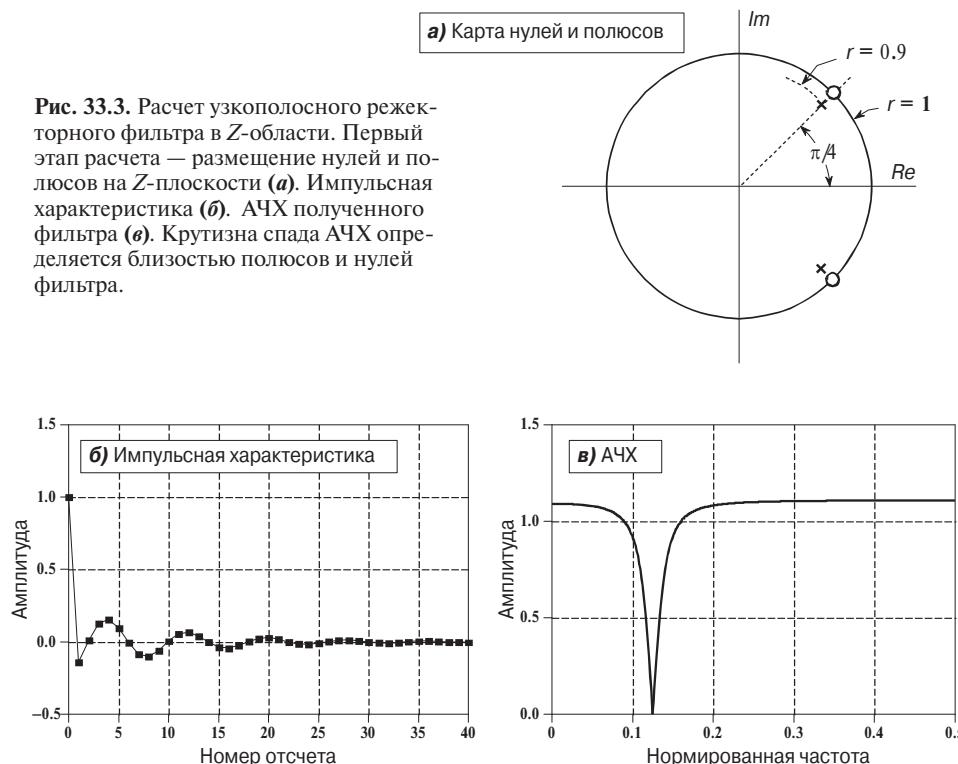
в полярной форме

$$\begin{aligned} z_1 &= 1.00 \cdot e^{j(\pi/4)} \\ z_2 &= 1.00 \cdot e^{j(-\pi/4)} \\ p_1 &= 0.90 \cdot e^{j(\pi/4)} \\ p_2 &= 0.90 \cdot e^{j(-\pi/4)} \end{aligned}$$

в прямоугольной форме

$$\begin{aligned} z_1 &= 0.7071 + j \cdot 0.7071 \\ z_2 &= 0.7071 - j \cdot 0.7071 \\ p_1 &= 0.6364 + j \cdot 0.6364 \\ p_2 &= 0.6364 - j \cdot 0.6364 \end{aligned}$$

Для доказательства того, что расположенные таким образом нули и полюсы определяют узкополосный режекторный фильтр, достаточно сравнить данное размещение с картой нулей и полюсов непрерывного узкополосного режекторного фильтра, показанной на Рис. 32.6. Принципиальное отличие состоит в том, что на  $Z$ -плоскости нули и полюсы размещаются вдоль единичной окружности, тогда как на  $S$ -плоскости они размешались вдоль вертикальной оси. Заданное размещение нулей и полюсов определяет приведенную круговую частоту наибольшего подавления, равную  $\pi/4$ , которая соответствует нормированной частоте, равной 0.125.



**Рис. 33.3.** Расчет узкополосного режек-  
торного фильтра в  $Z$ -области. Первый  
этап расчета — размещение нулей и по-  
люсов на  $Z$ -плоскости (a). Импульсная  
характеристика (б). АЧХ полученного  
фильтра (в). Крутизна спада АЧХ опре-  
деляется близостью полюсов и нулей  
фильтра.

Так как значения нулей и полюсов известны, то для построения передаточной функции в форме выражения (33.4) достаточно просто вписать в неё эти значения:

$$H(z) = \frac{[z - (0.7071 + j \cdot 0.7071)] \cdot [z - (0.7071 - j \cdot 0.7071)]}{[z - (0.6364 + j \cdot 0.6364)] \cdot [z - (0.6364 - j \cdot 0.6364)]}.$$

Чтобы найти весовые коэффициенты этого фильтра, преобразуем передаточную функцию к виду выражения (33.3). Для начала раскроем скобки, перемножая соответствующие величины:

$$\begin{aligned} H(z) = & \frac{z^2 - 0.7071z + j \cdot 0.7071z - 0.7071z + 0.7071^2 - j \cdot 0.7071^2 -}{z^2 - 0.6364z + j \cdot 0.6364z - 0.6364z + 0.6364^2 - j \cdot 0.6364^2 -} \\ & - j \cdot 0.7071z + j \cdot 0.7071^2 - j^2 \cdot 0.7071^2 \\ & - j \cdot 0.6364z + j \cdot 0.6364^2 - j^2 \cdot 0.6364^2. \end{aligned}$$

Теперь упростим полученное выражение. Слагаемые, содержащие мнимую единицу, исчезают, что объясняется зеркальной симметрией верхней и нижней полуплоскостей  $Z$ -плоскости (такая симметрия всегда присутствует при обработке вещественных сигналов):

$$H(z) = \frac{1.000 - 1.414 \cdot z + 1.000 \cdot z^2}{0.810 - 1.273 \cdot z + 1.000 \cdot z^2}.$$

Несмотря на то что теперь передаточная функция представлена в виде отношения двух многочленов, она все ещё не соответствует выражению (33.3), поскольку содержит вместо отрицательных степеней  $z$  положительные. Поэтому требуется разделить числитель и знаменатель на наибольшую степень  $z$ , а именно на  $z^2$ :

$$H(z) = \frac{1.000 - 1.414 \cdot z^{-1} + 1.000 \cdot z^{-2}}{1.000 - 1.273 \cdot z^{-1} + 0.810 \cdot z^{-2}}.$$

Теперь передаточная функция представлена в форме выражения (33.3) и содержит в своей записи непосредственно значения весовых коэффициентов БИХ-фильтра:

$$a_0 = 1.000$$

$$a_1 = -1.414 \quad b_1 = 1.273$$

$$a_2 = 1.000 \quad b_2 = -0.810$$

В приведенном нами примере были рассмотрены основные операции, с помощью которых осуществляется переход от карты нулей и полюсов к весовым коэффициентам рекурсивного фильтра. В ряде случаев удается получить более простые уравнения, устанавливающие связь между размещением нулей и полюсов и весовыми коэффициентами. Например, *биквадратному фильтру*, представляющему собой линейное звено, передаточная функция которого имеет два нуля и два полюса, соответствует следующая система уравнений:

$$\begin{aligned} a_0 &= 1, \\ a_1 &= -2r_0 \cos(\omega_0), \\ a_2 &= r_0^2, \\ b_1 &= 2r_p \cos(\omega_p), \\ b_2 &= -r_p^2. \end{aligned} \tag{33.5}$$

Система уравнений для расчёта биквадратного фильтра. Весовые коэффициенты  $a_0, a_1, a_2, b_1, b_2$  могут быть однозначно выражены через координаты полюсов ( $r_p, \omega_p$ ) и нулей ( $r_0, \omega_0$ ) фильтра.

Теперь, когда передаточная функция найдена, встаёт вопрос: как определить частотную характеристику фильтра? Существует три метода: один аналитический и два численных (выполнимых программно). Суть аналитического метода состоит в нахождении значений передаточной функции в точках, лежащих на единичной окружности в  $Z$ -плоскости, т. е. во всех точках, для которых  $r = 1$ . Точнее, при аналитическом методе требуется выполнять следующие действия: сначала необходимо записать передаточную функцию в форме выражения (33.3) или (33.4), а затем заменить  $z$  на  $e^{-j\omega}$  ( $re^{-j\omega}$  при  $r = 1$ ). В итоге получаем аналитическое выражение частотной характеристики  $H(\omega)$ . Главным недостатком является то, что результирующее выражение представлено в очень неудобной для его использования форме. Как правило, требуется выполнить значительное количество алгебраических

преобразований, прежде чем данное выражение примет удобный для дальнейших вычислений вид. Хотя данный метод позволяет точно выразить частотную характеристику, его трудно реализовать программно, а это имеет большое значение при создании компьютерных пакетов проектирования цифровых фильтров.

Второй метод определения частотной характеристики основан на непосредственном вычислении отдельно взятых точек на единичной окружности. Он отличается от первого метода лишь тем, что вместо выполнения сложных преобразований, позволяющих выразить частотную характеристику в аналитической форме, вычисляются дискретные отсчёты частотной характеристики. К примеру, компьютерная программа может вычислить значения частотной характеристики в 1000 точках, расположенных равномерно на интервале приведённых частот  $0\dots\pi$ . Представьте муравья, который пробегает поочерёдно 1000 точек, равномерно распределённых по той части единичной окружности, которая расположена в верхней половине  $Z$ -плоскости. Останавливаясь в каждой очередной точке, он вычисляет амплитуду и фазу частотной характеристики (т. е. АЧХ и ФЧХ), представляя координаты точки в уравнение передаточной функции.

Этот метод хорошо работает и поэтому широко используется в специализированных программных пакетах. Главным недостатком данного метода следует считать то, что он не учитывает влияние шума округления, приводящего к искажению характеристик системы. Даже если найденная с его помощью частотная характеристика кажется идеальной, при практической реализации рассчитанная система может оказаться неустойчивой!

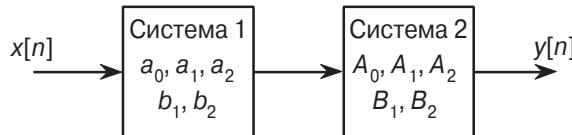
Для устранения такого недостатка был разработан третий метод, в котором для вычисления частотной характеристики используется не передаточная функция, а весовые коэффициенты фильтра. Прежде всего вычисляют отсчёты импульсной характеристики фильтра, для чего на его вход подают единичный импульс. Затем с помощью ДПФ полученной импульсной характеристики (на основе алгоритма БПФ) находят частотную характеристику. Очень важно взять достаточно большое число отсчётов импульсной характеристики, чтобы неучтённые отсчёты не оказали существенного влияния. О том, сколькими отсчётами импульсной характеристики можно ограничиться, написано во множестве разных книг. Но тем не менее на практике руководствуются очень простым правилом. Сначала выбирают столько отсчётов, сколько считают необходимым выбрать. После того, как частотная характеристика рассчитана, берут вдвое больше отсчётов и снова рассчитывают частотную характеристику. Если полученные характеристики почти не отличаются друг от друга, значит ошибка, вызванная усечением импульсной характеристики, не приводит к существенному искажению её формы.

### 33.3. Каскадное и параллельное соединения

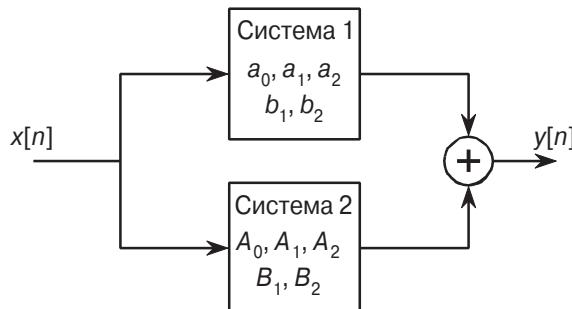
При расчёте сложных рекурсивных фильтров их обычно представляют в виде системы из простых блоков, чтобы избежать утомительных алгебраических преобразований. Две основных формы построения сложных фильтров — каскадная и параллельная — показаны на Рис. 33.4. Если необходимо получить полосовой фильтр, то достаточно последовательно (каскадно) соединить НЧ-фильтр и

**ВЧ-фильтр.** Параллельное соединение таких фильтров приводит к получению реекторного фильтра. Пусть соединяются система 1 и система 2 (Рис. 33.4) с весовыми коэффициентами соответственно  $a_0, a_1, a_2, b_1, b_2$  и  $A_0, A_1, A_2, B_1, B_2$ . Требуется найти весовые коэффициенты  $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \beta_1, \beta_2, \beta_3, \beta_4$  такого рекурсивного фильтра, который получается в результате соединения двух простых блоков (системы 1 и 2) в единый рекурсивный фильтр; назовём её системой 3.

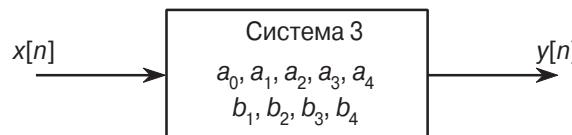
**а) Последовательное (каскадное) соединение**



**б) Параллельное соединение**



**в) Замена единой системой**



**Рис. 33.4.** Объединение отдельных блоков в единый фильтр. Z-преобразование позволяет объединить последовательно или параллельно соединённые блоки (системы 1 и 2) в единый рекурсивный фильтр (система 3).

Как вы помните из предыдущих глав, при последовательном соединении частотные характеристики умножаются, а при параллельном — складываются. Такое правило справедливо и для передаточных функций в Z-области. Переход к Z-области позволяет объединить две рекурсивных системы, воспользовавшись сложением или умножением Z-изображений, а затем остаётся выполнить обратный переход и получить результатирующую систему.

Для примера покажем, как объединить два последовательно включённых биквадратных фильтра в единое звено. Передаточные функции исходных звеньев

записываем в форме выражения (33.3). Тогда передаточная функция последовательного соединения может быть представлена в виде их произведения:

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 - b_1 z^{-1} - b_2 z^{-2}} \times \frac{A_0 + A_1 z^{-1} + A_2 z^{-2}}{1 - B_1 z^{-1} - B_2 z^{-2}}.$$

Перемножим многочлены и приведём подобные члены:

$$H(z) = \frac{a_0 A_0 + (a_0 A_1 + a_1 A_0)z^{-1} + (a_0 A_2 + a_1 A_1 + a_2 A_0)z^{-2} + (a_1 A_2 + a_2 A_1)z^{-3} + (a_2 A_2)z^{-4}}{1 - (b_1 + B_1)z^{-1} - (b_2 + B_2 - b_1 B_1)z^{-2} - (-b_1 B_2 - b_2 B_1)z^{-3} - (-b_2 B_2)z^{-4}}.$$

Так как полученное уравнение представлено в форме (33.3), то из него непосредственно выражаются коэффициенты результирующей системы:

$$\begin{aligned}\alpha_0 &= a_0 A_0, \\ \alpha_1 &= a_0 A_1 + a_1 A_0, & \beta_1 &= b_1 + B_1, \\ \alpha_2 &= a_0 A_2 + a_1 A_1 + a_2 A_0, & \beta_2 &= b_2 + B_2 - b_1 B_1, \\ \alpha_3 &= a_1 A_2 + a_2 A_1, & \beta_3 &= -b_1 B_2 - b_2 B_1, \\ \alpha_4 &= a_2 A_2, & \beta_4 &= -b_2 B_2.\end{aligned}$$

Практическая реализация описанного подхода требует множества алгебраических преобразований, связанных с умножением и дальнейшей перегруппировкой слагаемых. К счастью, программа для реализации алгоритма на компьютере оказывается очень короткой (**Программа 33.1**). В этой программе весовые коэффициенты объединяемых фильтров предварительно размещаются в массивах A1[ ], B1[ ] и A2[ ], B2[ ]. Коэффициенты результирующей системы программа записывает в массивы A3[ ], B3[ ]. Несмотря на то что объединение последовательно и параллельно соединённых систем требует разных математических операций, программа при этом почти не меняется, а меняется только одна её строка.

### **Программа 33.1. Объединение последовательно и параллельно соединённых рекурсивных фильтров**

```

100 'ОБЪЕДИНЕНИЕ ПОСЛЕДОВАТЕЛЬНО И ПАРАЛЛЕЛЬНО СОЕДИНЁННЫХ ФИЛЬТРОВ
110 '
120 '          'ОБЪЯВЛЕНИЕ МАССИВОВ
130 DIM A1[8], B1[8]      'Коэффициенты а и б системы 1
140 DIM A2[8], B2[8]      'Коэффициенты а и б системы 2
150 DIM A3[16], B3[16]    'Коэффициенты а и б системы 3 (результирующей)
160 '
170 'Выбор типа соединения: последовательное или параллельное
180 INPUT "Введите для каскадного соединения 0, для параллельного 1: ", CP%
190 '
200 GOSUB XXXX            'Подпрограмма загрузки A1[ ], B1[ ], A2[ ], B2[ ]
210 '
220 FOR I% = 0 TO 8        'Переход от коэффициентов к передаточной функции
230 B2[I%] = -B2[I%]
240 B1[I%] = -B1[I%]
250 NEXT I%
260 B1[0] = 1

```

```

270 B2[0] = 1
280 '
290 FOR I% = 0 TO 16      'Умножение полиномов с использованием свёртки
300 A3[I%] = 0
310 B3[I%] = 0
320 FOR J% = 0 TO 8
330 IF I%-J% < 0 OR I%-J% > 8 THEN GOTO 370
340 IF CP% = 0 THEN A3[I%] = A3[I%] + A1[J%]*A2[I%-J%]
350 IF CP% = 1 THEN A3[I%] = A3[I%] + A1[J%]*B2[I%-J%] + A2[J%]*B1[I%-J%]
360 B3[I%] = B3[I%] + B1[J%] * B2[I%-J%]
370 NEXT J%
380 NEXT I%
390 '
400 FOR I% = 0 TO 16      'Переход от передаточной функции к коэффициентам
410 B3[I%] = -B3[I%]
420 NEXT I%
430 B3[0] = 0
440 '                    'Весовые коэффициенты результирующего фильтра теперь
450 END                  'записаны в A3[ ] и B3[ ]

```

В строках 220...270 выполняется переход от весовых коэффициентов рекуррентных фильтров к коэффициентам полиномов передаточных функций в форме выражения (33.3). После соответствующей комбинации этих передаточных функций, производимой в строках 290...380, выполняется обратный переход, позволяющий выразить весовые коэффициенты результирующего фильтра (строки 400...430).

Суть данной программы состоит в том, чтобы правильно расположить коэффициенты фильтра в массивах и правильно произвести их комбинацию. Допустим, числитель передаточной функции первого фильтра имеет вид:  $a_0 + a_1z^{-1} + a_2z^{-2} + a_3z^{-3} + \dots$ . Коэффициенты этого полинома  $a_0, a_1, a_2, a_3, \dots$  сохраняются в соответствующих ячейках выделенного для них массива памяти:  $A1[0], A1[1], A1[2], A1[3], \dots$ . Коэффициенты числителя передаточной функции второго фильтра сохраняются аналогичным образом в ячейках второго массива:  $A2[0], A2[1], A2[2], A2[3], \dots$ . Найденные при выполнении программы коэффициенты размещаются в ячейках третьего массива:  $A3[0], A3[1], A3[2], A3[3], \dots$ . Все преобразования полиномов сводятся к простым операциям с их коэффициентами. Как должен выглядеть алгоритм, если все необходимые коэффициенты размещаются в массивах  $A1[ ], A2[ ]$  и  $A3[ ]$ ? На этот вопрос существует простой ответ: если полиномы перемножаются, то их коэффициенты подвергаются свёртке:  $A1[ ] * A2[ ] = = A3[ ]$ . То есть для вычисления коэффициентов числителя и знаменателя фильтра, образованного в результате объединения в одно звено двух последовательно соединённых фильтров, нужно выполнить свёртку коэффициентов числителей и коэффициентов знаменателей соответственно.

При объединении двух параллельно включённых фильтров процедура вычисления коэффициентов оказывается немного сложнее. Из курса алгебры известно, что сложение двух дробей происходит по следующему правилу:

$$\frac{w}{x} + \frac{y}{z} = \frac{w \cdot z + x \cdot y}{x \cdot z}.$$

Так как передаточные функции обоих фильтров представляют собой отношения полиномов, то при объединении параллельно включённых фильтров знаменатели передаточных функций, как и прежде, перемножаются, а числитель представляет собой сумму двух перекрёстных произведений числителя на знаменатель. Следовательно, процедура вычисления знаменателя получается одинаковой и для параллельного, и для последовательного соединения, а процедура вычисления числителя при параллельном соединении оказывается несколько сложнее. Последовательному соединению фильтров соответствует в предложенной программе строка 340 (свёртка коэффициентов двух числителей), а параллельному — строка 350 (сумма свёрток коэффициентов двух числителей с коэффициентами двух знаменателей). Процедура вычисления коэффициентов знаменателя не зависит от типа соединения фильтров и выполняется в строке 360.

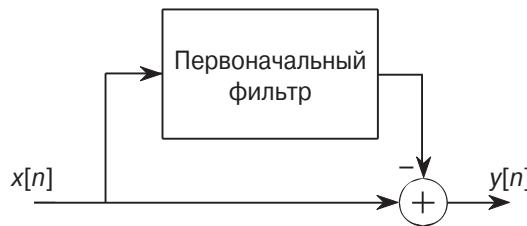
### 33.4. Инверсия АЧХ

В Главе 14 был рассмотрен метод *инверсии АЧХ* применительно к КИХ-фильтрам. Суть этого метода состоит в таком изменении весовых коэффициентов фильтра, за счёт которого достигается переворот АЧХ в направлении сверху вниз: полоса пропускания превращается в зону подавления и наоборот. НЧ-фильтр становится ВЧ-фильтром, полосовой фильтр переходит в режекторный и т. д. Метод инверсии АЧХ можно применить и для рекурсивных фильтров, хотя с меньшим успехом. Как показано на Рис. 33.5, инверсия АЧХ осуществляется путём вычитания выходного сигнала исходного фильтра из его входного сигнала. Данную структуру можно рассматривать как параллельное соединение двух фильтров, один из которых не вносит абсолютно никаких искажений в передаваемый сигнал. Опираясь на такое представление, нетрудно показать, что коэффициенты знаменателя остаются без изменений, а коэффициенты числителя подвергаются следующим преобразованиям:

$$\begin{aligned}\alpha_0 &= 1 - a_0, \\ \alpha_1 &= -a_1 - b_1, \\ \alpha_2 &= -a_2 - b_2, \\ \alpha_3 &= -a_3 - b_3, \\ &\vdots\end{aligned}\tag{33.6}$$

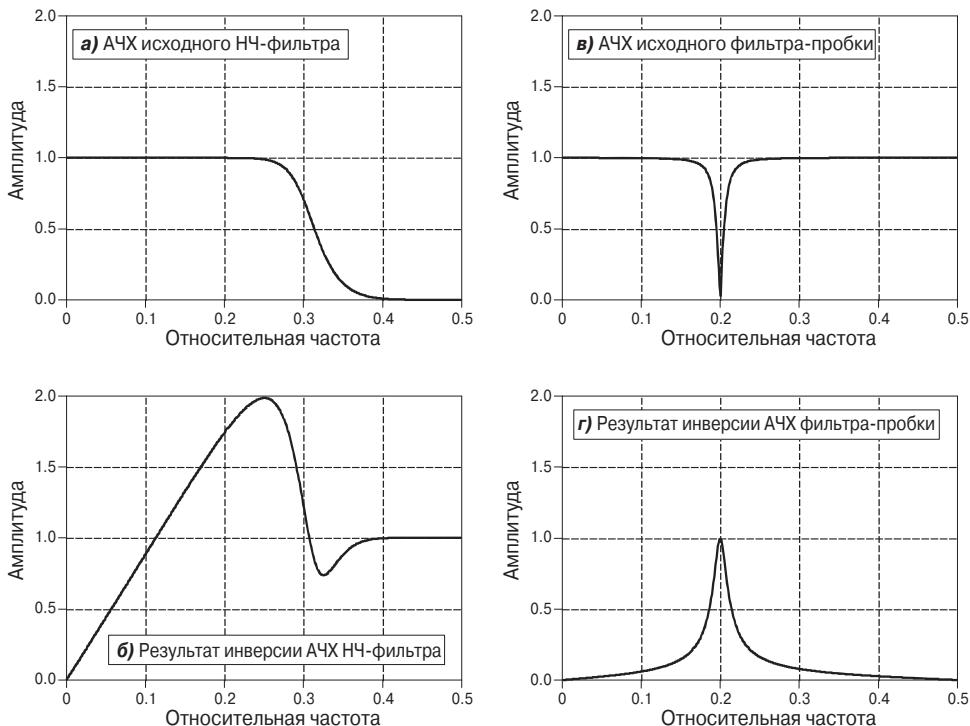
Инверсия АЧХ. Для переворота АЧХ достаточно изменить коэффициенты числителя передаточной функции. Коэффициенты числителя исходной передаточной функции обозначены латинскими буквами, а коэффициенты, полученные после инверсии, — греческими. Коэффициенты знаменателя не меняются. Метод инверсии АЧХ в случае рекурсивных фильтров мало эффективен.

На Рис. 33.6 метод инверсии АЧХ рассмотрен на примере двух широко используемых на практике рекурсивных фильтров: НЧ-фильтра Баттерворта (**а**) и узкополосного режекторного рекурсивного фильтра — фильтра-пробки (**в**). В результате инверсии получаются соответственно ВЧ-фильтр (**б**) и полосовой фильтр (**г**). Что можно сказать о качестве полученных АЧХ? ВЧ-фильтр получил просто отвратительным. С полосовым фильтром дело обстоит куда лучше, хотя



**Рис. 33.5.** Метод инверсии АЧХ. Фильтр, полученный методом инверсии АЧХ, можно рассматривать как систему, в которой выходной сигнал первоначального фильтра вычитается из его входного сигнала.

пик АЧХ уже не такой острый, каким был провал у режекторного фильтра. Наблюдаемые «удовлетворительные» результаты вызывают особенно сильное разочарование при сравнении с «отличными» результатами, полученными в Главе 14. Откуда же такая разница? Ответ кроется в ФЧХ фильтра, о которой часто забывают при расчёте.



**Рис. 33.6.** Два примера инверсии АЧХ. **а)** АЧХ фильтра Баттервортса (6 полюсов). **б)** Результат применения метода инверсии АЧХ для НЧ-фильтра Баттервортса — ничего хорошего! **в и г)** АЧХ узкополосного режекторного рекурсивного фильтра (фильтра-пробки) и результат, полученный при использовании метода инверсии АЧХ.

Чтобы показать, насколько сильный вред может причинить ФЧХ, рассмотрим систему, получившую название *преобразователя Гильберта*. Преобразователь Гильберта не является каким-то специальным техническим устройством. Любая система, имеющая единичный коэффициент передачи по амплитуде и обеспечивающая фазовый сдвиг 90 градусов во всем частотном диапазоне, считается преобразователем Гильберта. Другими словами, при прохождении через преобразователь Гильберта синусоидального сигнала любой частоты амплитуда сигнала остаётся неизменной, а фаза меняется на четверть периода. Преобразователи Гильберта, которые могут быть как аналоговыми, так и цифровыми (т. е. реализованными аппаратно или программно), широко используются в устройствах связи при выполнении модуляции и демодуляции.

Предположим теперь, что к преобразователю Гильберта применяется метод инверсии АЧХ, т. е. выходной сигнал преобразователя вычитается из входного сигнала. Если судить только по АЧХ, то можно заключить, что на выходе построенной системы всегда будет нуль: реакция на выходе преобразователя Гильберта равна по амплитуде входному воздействию и вычитается из него, приводя к полному подавлению сигнала. Разумеется, это совершенно неверно. Полная компенсация выходных синусоид возможна только при одновременном равенстве их амплитуд и фаз. Полученная путём инверсии АЧХ система имеет АЧХ с постоянным коэффициентом усиления, равным  $\sqrt{2}$ , и обеспечивает постоянный фазовый сдвиг — 45 градусов. В противоположность нашей наивной гипотезе о том, что сигнал на выходе должен быть полностью подавлен, этот сигнал даже превышает входное воздействие!

В Главе 14 метод спектральной инверсии замечательно работал только потому, что проверялся на фильтрах особого вида — фильтрах с нулевой фазой, т. е. на фильтрах с симметричной импульсной характеристикой. Если система не меняет фазу сигнала, то при вычитании может измениться только амплитуда. Поскольку нелинейность ФЧХ является общей «болезнью» рекурсивных фильтров, метод инверсии АЧХ чаще всего приводит к неудовлетворительным результатам.

## 33.5. Нормировка коэффициента усиления

Пусть имеется некоторый рекурсивный фильтр, и требуется так изменить его весовые коэффициенты, чтобы получить заданный коэффициент усиления. Например, может потребоваться фильтр с единичным усилением в полосе пропускания. Задача решается очень просто: достаточно умножить все коэффициенты  $a$  на такое число, на которое необходимо изменить коэффициент усиления фильтра, а коэффициенты  $b$  оставить без изменения.

Перед нормировкой коэффициента усиления требуется узнать его текущее значение. Поскольку нормировка производится в полосе пропускания, то должен учитываться тип фильтра: для НЧ-фильтров коэффициент усиления нормируется по нулевой частоте, для ВЧ-фильтров — на половине частоты дискретизации, т. е. на верхней границе основного диапазона частот дискретного фильтра. Определить коэффициент усиления на этих двух особых частотах очень легко. Поясним, как это сделать.

Сначала выведем уравнение для коэффициента усиления на нулевой частоте. Основная идея состоит в том, что при всех единичных отсчётах входной последо-

вательности все отсчёты выходной последовательности равны коэффициенту усиления фильтра —  $G$ . Взаимосвязь отсчётов входной и выходной последовательностей задаётся следующим уравнением:

$$y[n] = a_0x[n] + a_1x[n - 1] + a_2x[n - 2] + \dots + b_1y[n - 1] + b_2y[n - 2] + b_3y[n - 3] + \dots$$

Приравниваем входные отсчёты единице, а выходные отсчёты —  $G$ , получая упрощённое уравнение, описывающее фильтр на нулевой частоте:

$$G = a_0 + a_1 + a_2 + a_3 + \dots + b_1G + b_2G + b_3G + b_4G + \dots$$

Теперь остаётся выразить коэффициент усиления на нулевой частоте:

$$G = \frac{a_0 + a_1 + a_2 + a_3 + \dots}{1 - (b_1 + b_2 + b_3 + \dots)}. \quad (33.7)$$

Коэффициент передачи рекурсивного фильтра по постоянной составляющей, выраженный через весовые коэффициенты.

Чтобы получить фильтр с единичным коэффициентом передачи по постоянной составляющей, достаточно определить коэффициент изначально имеющегося фильтра  $G$ , а затем разделить все коэффициенты  $a$  на  $G$ .

Подобным образом определяется коэффициент усиления на частоте 0.5. Зададим отсчёты входной и выходной последовательностей так, чтобы они соответствовали отсчётам синусоиды с частотой 0.5, и упрощаем получившееся уравнение. На частоте 0.5 дискретная синусоида может быть представлена чередованием двух отсчётов: 1 и  $-1$ . То есть входная последовательность имеет вид: 1,  $-1$ , 1,  $-1$ , 1,  $-1$ , 1 и т. д. В последовательности, образующейся на выходе фильтра, тоже существует чередование знака:  $G$ ,  $-G$ ,  $G$ ,  $-G$ ,  $G$ ,  $-G$  и т. д. Заменяя эти значениями отсчёты входной и выходной последовательностей в уравнении рекурсивного фильтра:

$$G = a_0 - a_1 + a_2 - a_3 + \dots - b_1G + b_2G - b_3G + b_4G + \dots$$

Выражаем коэффициент усиления на частоте 0.5 через весовые коэффициенты:

$$G = \frac{a_0 - a_1 + a_2 - a_3 + \dots}{1 - (-b_1 + b_2 - b_3 + b_4 + \dots)}. \quad (33.8)$$

Коэффициент передачи рекурсивного фильтра на верхней границе основного диапазона частот, выраженный через весовые коэффициенты.

Как и прежде, для нормировки коэффициента усиления необходимо вычислить коэффициент усиления  $G$  и разделить на него все коэффициенты  $a$ . При создании программы для вычисления коэффициента усиления по выражению (33.8) потребуется методика, позволяющая инвертировать арифметический знак у весовых коэффициентов с нечётными индексами. Проще всего такая инверсия достигается умножением каждого весового коэффициента фильтра на  $(-1)^k$ , где  $k$  — индекс коэффициента, обрабатываемого на текущей итерации. По мере того как  $k$  пробегает значения 0, 1, 2, 3, 4, 5, 6 и т. д., вычисление  $(-1)^k$  приводит к последовательности 1,  $-1$ , 1,  $-1$ , 1,  $-1$ , 1 и т. д.

## 33.6. Расчёт фильтров. Метод Чебышева–Баттервортса

Общим подходом к расчёту цифровых рекурсивных фильтров считается метод Чебышева–Баттервортса. Программа для расчёта фильтров на основе этого метода подробно рассмотрена в Главе 20. Центральной идеей метода является переход от карты нулей и полюсов аналогового фильтра, т. е. от  $S$ -плоскости, к цифровому фильтру путём нескольких математических преобразований. Чтобы сделать алгебраические действия менее сложными, расчёт фильтра производится в каскадной форме, т. е. разбивается на несколько шагов. Каждый каскад представляет собой цифровое звено второго порядка и, следовательно, соответствует паре полюсов. На завершающем этапе все звенья малых порядков объединяются в одно звено высокого порядка. Этот алгоритм является очень сложным, и он идеально подходит для завершения книги. Разберёмся, как он работает.

### 33.6.1. Основной цикл программы

**Программа 33.2** представляет собой копию программы, которая уже была описана в Главе 20, а на Рис. 33.7 показана блок-схема алгоритма. После инициализации переменных и ввода параметров фильтра пользователем в основной части программы начинает выполняться цикл, число проходов которого равно числу пар полюсов фильтра. Цикл управляется условием 11 на блок-схеме и оператором цикла FOR-NEXT, расположенным в строках 320 и 460 кода программы. Например, для фильтра шестого порядка (3 пары полюсов) цикл выполняется 3 раза, а индекс  $P\%$  принимает значения 1, 2, 3. То есть фильтр шестого порядка при расчёте представляется в виде последовательного соединения трёх фильтров второго порядка.

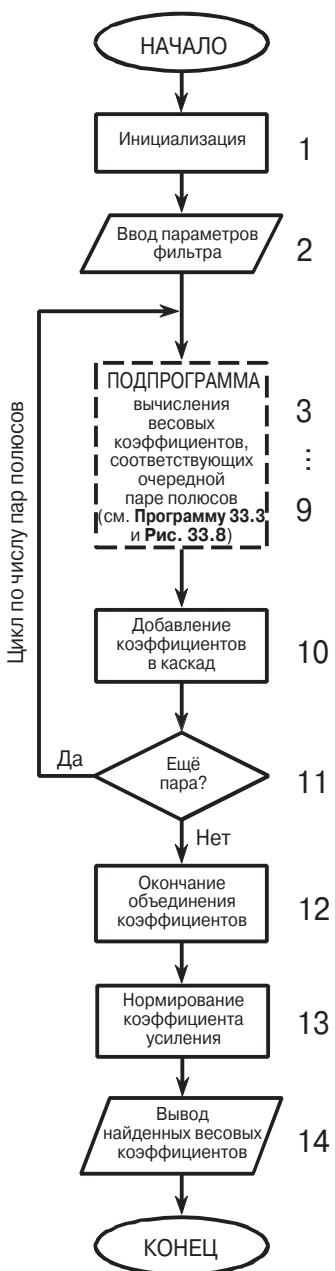
#### Программа 33.2

```

100 'ФИЛЬТР ЧЕБЫШЕВА – ВЫЧИСЛЕНИЕ ВЕСОВЫХ КОЭФФИЦИЕНТОВ
110 '
120           'НАЧАЛЬНЫЕ ЗНАЧЕНИЯ
130 DIM A[22]    'содержит коэффициенты а по завершении выполнения программы
140 DIM B[22]    'содержит коэффициенты b по завершении выполнения программы
150 DIM TA[22]   'используется внутри программы для объединения ступеней
160 DIM TB[22]   'используется внутри программы для объединения ступеней
170 '
180 FOR I% = 0 TO 22
190 A[I%] = 0
200 B[I%] = 0
210 NEXT I%
220 '
230 A[2] = 1
240 B[2] = 1
250 PI = 3.14159265
260           'ВВОД ЧЕТЫРЕХ ПАРАМЕТРОВ ФИЛЬТРА
270 INPUT "Введите частоту среза (от 0 до 0.5): ", FC
280 INPUT "Введите 0 для НЧ-фильтра, 1 для ВЧ-фильтра: ", LH

```

```
290 INPUT "Введите размах колебаний АЧХ в процентах (0...29): ", PR
300 INPUT "Введите количество полюсов (2,4,...20): ", NP
310 '
320 FOR P% = 1 TO NP/2          'ЦИКЛ ДЛЯ КАЖДОЙ ПАРЫ ПОЛЮСОВ
330 '
340 GOSUB 1000                'Подпрограмма (Программа 33.3)
350 '
360 FOR I% = 0 TO 22           'Добавление коэффициентов в каскад
370 TA[I%] = A[I%]
380 TB[I%] = B[I%]
390 NEXT I%
400 '
410 FOR I% = 2 TO 22
420 A[I%] = A0*TA[I%] + A1*TA[I%-1] + A2*TA[I%-2]
430 B[I%] = TB[I%] - B1*TB[I%-1] - B2*TB[I%-2]
440 NEXT I%
450 '
460 NEXT P%
470 '
480 B[2] = 0                  'Окончание объединения коэффициентов
490 FOR I% = 0 TO 20
500 A[I%] = A[I%+2]
510 B[I%] = -B[I%+2]
520 NEXT I%
530 '
540 SA = 0                   'НОРМИРОВАНИЕ КОЭФФИЦИЕНТА УСИЛЕНИЯ
550 SB = 0
560 FOR I% = 0 TO 20
570 IF LH = 0 THEN SA = SA + A[I%]
580 IF LH = 0 THEN SB = SB + B[I%]
590 IF LH = 1 THEN SA = SA + A[I%] * (-1)^I%
600 IF LH = 1 THEN SB = SB + B[I%] * (-1)^I%
610 NEXT I%
620 '
630 GAIN = SA / (1 - SB)
640 '
650 FOR I% = 0 TO 20
660 A[I%] = A[I%] / GAIN
670 NEXT I%
680 'Результирующие коэффициенты содержатся в A[ ] и B[ ]
690 END
```



**Рис. 33.7.** Блок-схема алгоритма расчёта фильтра Чебышева–Баттервортса. Блок-схема подпрограммы (**Программа 33.3**), вызываемой из строки 340, показана на **Рис. 33.8**.

### 33.6.2. Объединение коэффициентов

При каждом проходе цикла вызывается подпрограмма (**Программа 33.3**) вычисления весовых коэффициентов, соответствующих очередной паре полюсов. Найденные весовые коэффициенты сохраняются в переменных A0, A1, A2, B1, B2. На шаге 10 алгоритма (строки 360...440 программы) эти коэффициенты объединяются с коэффициентами звена, найденного на предыдущей итерации, и размещаются в ячейках массивов A[ ] и B[ ]. После первого прохода цикла в A[ ] и B[ ] содержатся весовые коэффициенты первого звена в последовательном соединении. После второго прохода в A[ ] и B[ ] содержатся весовые коэффициенты фильтра, соответствующего первым двум звеньям. После завершения цикла массивы A[ ] и B[ ] содержат весовые коэффициенты результирующего фильтра.

В целом принцип объединения весовых коэффициентов тот же, что был использован в **Программе 33.1**; отличие состоит лишь в нескольких доработках, делающих программу более компактной. Первое отличие состоит в начале индексации массивов A[ ] и B[ ] со второго элемента. Коэффициент  $a_0$  сохраняется в ячейке A[2],  $a_1$  и  $b_1$  сохраняются в ячейках A[3] и B[3] и т. д. Этот приём позволяет избежать возможного выхода при адресации за пределы массивов. Сдвиг на два элемента устраняется на шаге 12 (строки 480...520) при перезаписи найденных весовых коэффициентов фильтров.

Вторым отличием является инициализация массивов A[ ] и B[ ] (строки 180...240). Коэффициенты должны быть заданы так, чтобы образовать систему, не вносящую искажений в передаваемый сигнал. Если обнулить все элементы A[ ] и B[ ], то при объединении с коэффициентами звеньев, рассчитанных на следующих итерациях, все элементы массивов A[ ] и B[ ] так и останутся нулями. Третье отличие заключается во введении массивов TA[ ] и TB[ ] для временного хранения весовых коэффициентов. В этих массивах во время выполнения свёртки сохраняются старые коэффициенты из массивов A[ ] и B[ ], чтобы освободить место для записи в них новых коэффициентов.

Перед завершением программы производится нормирование коэффициента усиления. На шаге 13 (строки 540...670) весовые коэффициенты фильтра масштабируются таким образом, чтобы коэффициент передачи в полосе пропускания стал равным единице. Процедура нормирования, как и раньше, состоит из двух этапов: сначала по выражениям (33.7) или (33.8) определяется коэффициент передачи уже имеющегося фильтра, а затем все коэффициенты  $a$  делятся на этот коэффициент. При суммировании коэффициентов  $a$  и  $b$  используются промежуточные переменные SA и SB.

### 33.6.3. Вычисление координат полюсов фильтра на S-плоскости

Независимо от выбранного типа фильтра работа подпрограммы (**Программа 33.3**) начинается с вычисления на S-плоскости координат полюсов НЧ-фильтра Баттерворта с частотой среза  $\omega = 1$ . Как говорилось в предыдущей главе, полюсы фильтра Баттерворта располагаются с постоянным интервалом по окружности, лежащей на S-плоскости. Поскольку фильтр является низкочастотным, нули не принимаются во внимание. Частоте среза  $\omega = 1$  соответствует ок-

ружность единичного радиуса. На шаге 3 (строки 1080 и 1090) вычисляются декартовы координаты для очередной пары полюсов. Переменные  $RP$  и  $IP$  используются для хранения соответственно вещественной и мнимой компонент полюса. Если пару полюсов представить в аналитической форме  $\sigma \pm j\omega$ , то  $\sigma$  и  $\omega$  совпадают со значениями  $RP$  и  $IP$ . Исходными данными для вычисления координат полюса являются порядок фильтра (количество полюсов) и номер итерации (порядковый номер пары) —  $NP$  и  $P\%$ .

### 33.6.4. Преобразование окружности в эллипс

Чтобы получить фильтр Чебышева, окружность, на которой находятся полюсы, необходимо преобразовать в эллипс. Относительная гладкость эллипса определяет размах колебаний АЧХ в полосе пропускания фильтра. Полюс, имеющий на окружности координаты  $\sigma$  и  $\omega$ , переходит в результате такой деформации в точку эллипса с координатами  $\sigma'$  и  $\omega'$ :

$$\sigma' = \sigma \operatorname{sh}(v)/k,$$

$$\omega' = \omega \operatorname{ch}(v)/k,$$

где

$$v = \frac{\operatorname{arsh}(1/\xi)}{NP}, \quad (33.9)$$

$$k = \operatorname{ch}\left(\frac{1}{NP} \operatorname{arch} \frac{1}{\xi}\right),$$

$$\xi = \left[ \left( \frac{100}{100 - PR} \right)^2 - 1 \right]^{1/2}.$$

Преобразование окружности в эллипс. Данное преобразование описывает переход от размещения полюсов по окружности к размещению по эллипсу.  $NP$  — порядок фильтра,  $PR$  — размах колебаний АЧХ в полосе пропускания фильтра в процентах. Координаты  $\sigma$  и  $\omega$  определяют положение полюса на окружности, а  $\sigma'$  и  $\omega'$  — на эллипсе. Переменные  $\xi$ ,  $v$  и  $k$  введены для упрощения записи.

В (33.9) для описания эллипса использованы гиперболические синус и косинус, тогда как для выражения координат точек на окружности достаточно обычных синуса и косинуса. Гладкость эллипса определяется величиной параметра  $PR$ , численно равной размаху колебаний АЧХ в полосе пропускания фильтра, выраженному в процентах. Переменные  $\xi$ ,  $v$  и  $k$  введены с целью упрощения уравнений, в программе им соответствуют ES, VX и KX. Данные выражения не только преобразуют окружность в эллипс, но и позволяют переместить полюсы фильтра так, чтобы сохранить единичную частоту среза. Так как во многих языках программирования отсутствуют встроенные гиперболические функции, приведем здесь следующие тождества:

$$\operatorname{ch}(x) = \frac{e^x + e^{-x}}{2},$$

$$\operatorname{ch}(x) = \frac{e^x + e^{-x}}{2},$$

$$\operatorname{arsh}(x) = \ln \left[ x + (x^2 + 1)^{1/2} \right],$$

$$\operatorname{arch}(x) = \ln \left[ x + (x^2 - 1)^{1/2} \right].$$

При  $PR \geq 30$  и  $PR = 0$  описанная процедура расчёта приводит к запрещенным операциям (извлечению квадратного корня из отрицательного числа и делению на ноль). Чтобы программа позволяла рассчитывать фильтры Баттервортса, которые характеризуются нулевыми колебаниями АЧХ в полосе пропускания ( $PR = 0$ ), в строке 1120 (Программа 33.3) вводится оператор условного перехода.

### Программа 33.3

```

1000 ' ЭТА ПОДПРОГРАММА ВЫЗЫВАЕТСЯ ИЗ СТРОКИ 340 ПРОГРАММЫ 33.2
1010 '
1020 ' Входные переменные подпрограммы: PI, FC, LH, PR, HP, P%
1030 ' Выходные переменные подпрограммы: A0, A1, A2, B1, B2
1040 ' Внутренние переменные: RP, IP, ES, VX, KX, T, W, M, D, K,
1050 ' X0, X1, X2, Y1, Y2
1060 '
1070 '           'Вычисление координат полюса на единичной окружности
1080 RP = -COS(PI/(NP*2) + (P%-1) * PI/NP)
1090 IP = SIN(PI/(NP*2) + (P%-1) * PI/NP)
1100 '
1110 '           'Преобразование окружности в эллипс
1120 IF PR = 0 THEN GOTO 1210
1130 ES = SQR( (100 / (100-PR))^2 - 1 )
1140 VX = (1/NP) * LOG( (1/ES) + SQR( (1/ES^2) + 1 ) )
1150 KX = (1/NP) * LOG( (1/ES) + SQR( (1/ES^2) - 1 ) )
1160 KX = (EXP(KX) + EXP(-KX))/2
1170 RP = RP * ( (EXP(VX) - EXP(-VX)) / 2 ) / KX
1180 IP = IP * ( (EXP(VX) + EXP(-VX)) / 2 ) / KX
1190 '
1200 '           'Преобразование S-области в Z-область
1210 T = 2 * TAN(1/2)
1220 W = 2*PI*FC
1230 M = RP^2 + IP^2
1240 D = 4 - 4*RP*T + M*T^2
1250 X0 = T^2/D
1260 X1 = 2*T^2/D
1270 X2 = T^2/D
1280 Y1 = (8 - 2*M*T^2)/D
1290 Y2 = (-4 - 4*RP*T - M*T^2)/D
1300 '           'Преобразование НЧ-фильтра в НЧ-фильтр или
1310 '           'НЧ-фильтра в ВЧ-фильтр
1320 IF LH = 1 THEN K = -COS(W/2 + 1/2) / COS(W/2 - 1/2)
1330 IF LH = 0 THEN K = SIN(1/2 - W/2) / SIN(1/2 + W/2)
1340 D = 1 + Y1*K - Y2*K^2
1350 A0 = (X0 - X1*K + X2*K^2) / D
1360 A1 = (-2*X0*K + X1 + X1*K^2 - 2*X2*K) / D

```

```

1370 A2 = (X0*K^2 - X1*K + X2)/D
1380 B1 = (2*K + Y1 + Y1*K^2 - 2*Y2*K)/D
1390 B2 = (-(K^2) - Y1*K + Y2)/D
1400 IF LH = 1 THEN A1 = -A1
1410 IF LH = 1 THEN B1 = -B1
1420 '
1430 RETURN

```

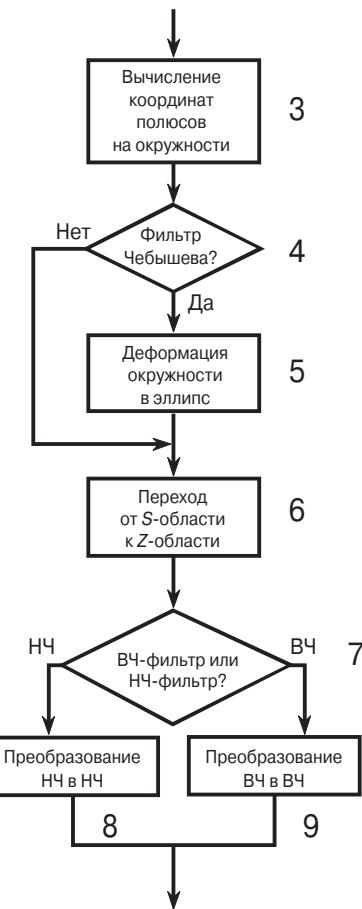


Рис. 33.8. Блок-схема подпрограммы, вызываемой из строки 340 основной программы (Программа 33.2).

### 33.6.5. Переход от аналогового фильтра к цифровому

Самым распространённым методом перехода от карты нулей и полюсов, заданной на  $S$ -плоскости, к карте нулей и полюсов, заданной на  $Z$ -плоскости, является *билинейное преобразование*. Оно представляет собой конформное отображение, переводящее точки одной комплексной плоскости в точки другой комплексной плоскости. При билинейном преобразовании осуществляется переход от  $H(s)$  к  $H(z)$  в результате следующей замены:

$$s \rightarrow \frac{2(1-z^{-1})}{T(1+z^{-1})}. \quad (33.10)$$

Билинейное преобразование. Данная замена переводит точки  $S$ -плоскости в соответствующие им точки  $Z$ -плоскости.

То есть при билинейном преобразовании сначала следует записать уравнение непрерывной передаточной функции  $H(s)$ , а потом заменить в нём все  $s$  указанным выражением. Как правило, выбирают  $T = 2\operatorname{tg}(1/2) = 1.093$ , потому что в этом случае все точки  $S$ -плоскости, соответствующие диапазону частот  $0 \dots \infty$  рад/с, переходят в точки  $Z$ -плоскости, соответствующие диапазону частот  $0 \dots \pi$  рад/с. Мы не будем здесь подробно рассказывать обо всех особенностях билинейного преобразования, делающих его столь привлекательным для перехода от  $S$ -плоскости к  $Z$ -плоскости, заметим лишь, что это преобразование переводит вертикальные линии в окружности. Теперь рассмотрим, как работает метод  $Z$ -преобразования, на конкретном примере. Передаточная функция непрерывной (аналоговой) системы с одной парой полюсов ( $p_1 = \sigma + j\omega$  и  $p_2 = \sigma - j\omega$ ) может быть записана в следующей форме:

$$H(s) = \frac{1}{(s - p_1)(s - p_2)}.$$

Путём замены выражения (33.10) получаем передаточную функцию дискретной (цифровой) системы, которая также содержит два полюса. Теперь проблема заключается в том, что найденное выражение является достаточно сложным:

$$H(z) = \frac{1}{\left(\frac{2(1-z^{-1})}{T(1+z^{-1})} - (\sigma + j\omega)\right)\left(\frac{2(1-z^{-1})}{T(1+z^{-1})} - (\sigma - j\omega)\right)}.$$

Выполнив множество утомительных алгебраических преобразований, получаем уравнение в стандартной форме выражения (33.3), где весовые коэффициенты определяются следующим образом:

$$\begin{aligned} a_0 &= T^2 / D, \\ a_1 &= 2T^2 / D, \\ a_2 &= T^2 / D, \\ b_1 &= (8 - 2MT^2) / D, \\ b_2 &= (-4 - 4\sigma T - MT^2) / D, \end{aligned} \quad (33.11)$$

где

$$\begin{aligned} M &= \sigma^2 + \omega^2, \\ T &= 2\operatorname{tg}(1/2), \\ D &= 4 - 4\sigma T - MT^2. \end{aligned}$$

Результат билинейного преобразования для передаточной функции с двумя полюсами. Пара полюсов на  $S$ -плоскости задана выражением  $\sigma \pm j\omega$ . Параметры  $a_0, a_1, a_2, b_1, b_2$  — весовые коэффициенты полученного цифрового фильтра.

Переменные  $M$ ,  $T$  и  $D$  не имеют физического смысла и введены только для упрощения записи уравнений.

В коде программы переходу от  $S$ -области к  $Z$ -области соответствуют строки 1200...1290. При этом координаты пары полюсов на  $S$ -плоскости задаются в программе переменными  $RP$  и  $IP$ , а весовые коэффициенты полученного цифрового фильтра сохраняются в переменных  $X0$ ,  $X1$ ,  $X2$ ,  $Y1$ ,  $Y2$ . На данном этапе работы программы были получены весовые коэффициенты рекурсивного НЧ-фильтра второго порядка с единичной частотой среза.

### 33.6.6. Преобразование НЧ-фильтра в НЧ-фильтр

Изменение частоты среза рекурсивного фильтра тоже осуществляется конформным отображением. Пусть известна передаточная функция рекурсивного НЧ-фильтра с единичной частотой среза. Тогда передаточную функцию НЧ-фильтра с частотой среза  $W$  можно получить путём *преобразования НЧ-фильтра в НЧ-фильтр*. Как и в случае билинейного преобразования, в данном отображении вводится своё правило замены. Сначала записывают передаточную функцию цифрового звена с единичной частотой среза, а затем заменяют  $z^{-1}$  следующим образом:

$$z^{-1} \rightarrow \frac{z^{-1} - k}{1 - kz^{-1}}, \quad (33.12)$$

где

$$k = \frac{\sin(1/2 - W/2)}{\sin(1/2 + W/2)}.$$

Преобразование НЧ-фильтра в НЧ-фильтр. Данный метод позволяет получить из НЧ-фильтра с единичной частотой среза НЧ-фильтр с частотой среза  $W = 0 \dots \pi$

В результате получается передаточная функция фильтра с новой частотой среза —  $W$ . В частности, для звена второго порядка такая замена приводит к следующим выражениям:

$$\begin{aligned} a_0 &= (a_0 - a_1 k + a_2 k^2) / D, \\ a_1 &= (-2a_0 k + a_1 + a_1 k^2 - 2a_2 k) / D, \\ a_2 &= (a_0 k^2 - a_1 k + a_2) / D, \\ b_1 &= (2k + b_1 + b_1 k^2 - 2b_2 k) / D, \\ b_2 &= (-k^2 - b_1 k + b_2) / D, \end{aligned} \quad (33.13)$$

где

$$\begin{aligned} D &= 1 + b_1 k - b_2 k^2, \\ k &= \frac{\sin(1/2 - W/2)}{\sin(1/2 + W/2)}. \end{aligned}$$

Результат преобразования НЧ-фильтра с единичной частотой среза в НЧ-фильтр с частотой среза  $W$  для звена второго порядка. Для обозначения весовых коэффициентов фильтра с частотой среза  $W$  использован прямой шрифт.

### 33.6.7. Преобразование НЧ-фильтра в ВЧ-фильтр

Немного изменив вид записи (33.12), можно получить выражение, описывающее *преобразование НЧ-фильтра в ВЧ-фильтр*:

$$z^{-1} \rightarrow \frac{-z^{-1} - k}{1 + kz^{-1}},$$

где

(33.14)

$$k = -\frac{\cos(W/2 + 1/2)}{\cos(W/2 - 1/2)}.$$

Преобразование НЧ-фильтра в ВЧ-фильтр. Данный метод позволяет получить из НЧ-фильтра с единичной частотой среза НЧ-фильтр с частотой среза  $W = 0\dots\pi$ .

Аналогично выражению (33.13) для звена второго порядка можно выразить каждый весовой коэффициент. В результате получаются такие же уравнения с двумя небольшими отличиями: по-другому определяется  $k$  (выражение (33.14)) и меняются арифметические знаки у выражений, позволяющих вычислять коэффициенты  $a_1$  и  $b_1$ . Преобразованию НЧ-фильтра с единичной частотой среза либо в НЧ-фильтр, либо в ВЧ-фильтр в программе соответствуют строки 1320...1410.

## 33.7. Две стороны ЦОС: хорошая и плохая

Цель данной книги заключается в том, чтобы *при минимальном использовании математики читатели смогли освоить и научиться достаточно хорошо применять на практике многие методы ЦОС*. Это позволит ученым и инженерам самостоятельно решать задачи ЦОС, которые могут возникать при исследованиях, проводимых ими даже в далёких от ЦОС областях науки и техники.

В данном контексте последние четыре главы позволяют увидеть обратную сторону медали, поскольку они посвящены сложным математическим преобразованиям. Примером может служить расчёт фильтра Чебышева–Баттервортса, о котором только что говорилось в этой главе. Хорошая сторона ЦОС, лицевая сторона медали, проявляется в изящных математических преобразованиях, позволяющих создать фильтр с оптимальными характеристиками. Однако существует и плохая сторона: процедура расчёта фильтра настолько сложна, что большинство инженеров, скорее всего, постараются найти ей какую-то замену.

Насколько глубоко вам следует изучать ЦОС? Всё зависит от вашего рода деятельности, т. е. от тех целей, которые вы перед собой ставите. Материал последних четырёх глав составляет теоретическую основу обработки сигналов. Если вы собираетесь сделать свою карьеру в области ЦОС, то вам необходимо более глубоко разобраться в использованном здесь сложном математическом аппарате. Что касается инженеров, специализирующихся в других областях, то им достаточно понять, как работают те или иные методы ЦОС, а их математическое обоснование в большинстве случаев имеет второстепенное значение.

# Глоссарий

- μ-закон** — Американский стандарт компандирования (сжатия динамического диапазона). Позволяет за счёт нелинейного квантования цифрового речевого сигнала сократить число информационных битов с 12 до 8. (См. также *A-закон*.)
- ASCII** (American Standard Code for Information Interchange) — Американский стандартный код для обмена информацией. Семибитный код для представления текстовой информации в двоичной форме. Широко применяется в компьютерах и системах связи.
- А-закон** — Европейский стандарт компандирования (сжатия динамического диапазона). Позволяет за счёт нелинейного квантования цифрового речевого сигнала сократить число информационных битов с 12 до 8. (См. также *μ-закон*.)
- C (Си)** — Язык программирования, получивший наиболее широкое распространение в науке и технике, а также в области цифровой обработки сигналов (ЦОС). Часто используется усовершенствованная версия этого языка, известная как C++.
- CVSD** (Continuously Variable Slope Delta modulation) — Дельта-модуляция с переменной крутизной. Применяется для преобразования речевого сигнала в поток двоичных данных.
- GIF** (Graphics Interchange Format) — Формат обмена графическими данными, использующий алгоритм сжатия (без потерь) LZW. Широко распространён в сети Интернет. (См. также *TIFF* и *JPEG*.)
- JPEG** (Joint Photographic Experts Group) — Метод сжатия изображений (с потерями) и соответствующий графический формат, разработанный Объединённой группой экспертов по машинной обработке фотографических изображений. Широко распространён в сети Интернет. (См. также *TIFF* и *GIF*.)
- LZW** — Метод сжатия данных, при котором используется алгоритм Лемпела–Зива–Велча.
- MFLOPS** (Million FLoating point Operations Per Second) — Единица быстродействия процессора. Миллион операций с плавающей точкой в секунду. (См. также *MIPS*.)
- MIPS** (Million Instructions Per Second) — Единица быстродействия процессора. Миллион операций (команд) в секунду. (См. также *MFLOPS*.)
- MPEG** (Motion Pictures Experts Group) — Стандарт сжатия движущихся изображений, разработанный Экспертной группой по вопросам движущихся изображений. Используется в цифровом телевидении.
- NTSC** — Стандарт цветного телевидения, принятый в США и Японии. (См. также *PAL* и *SECAM*.)
- PAL** — Формат цветного телевидения, принятый в странах Европы. (См. также *NTSC*.)
- RGB-кодирование** — Кодирование изображения, при котором каждый пиксель кодируется тремя параметрами, соответствующими яркости трёх первичных цветов: красного (R), зелёного (G) и синего (B).
- RISC-архитектура** (Reduced Instruction Set Computer) — Архитектура процессоров, использующих сокращённый набор команд (в отличие от архитектуры с полным на-

бором команд — Complex Instruction Set Computer). Обеспечивает быстрое и эффективное выполнение относительно небольшого набора встроенных команд.

**SECAM** — Стандарт цветного телевидения, получивший распространение в странах Европы. (См. также *NTSC*.)

**S-плоскость (область)** — Область, определяемая преобразованием Лапласа.

**TIFF** (Tagged Image File Format) — Формат представления графической информации, получивший широкое распространение при подготовке печатных документов. В данном формате сжатие обычно отсутствует, но допускается использование алгоритма сжатия *LZW*. (См. также *GIF* и *JPEG*.)

**Z-плоскость (область)** — Комплексная область, определяемая *Z*-преобразованием.

**Z-преобразование** — Метод математического анализа дискретных систем (например БИХ-фильтров), для описания которых используются разностные уравнения. Используется для перехода от временной области представления сигнала к *Z*-области.

**Автокорреляция** — Корреляция сигнала с самим собой. *Преобразование Фурье* функции автокорреляции даёт спектр мощности исходного сигнала.

**Аддитивность** — Математическое свойство, которым обязаны обладать линейные системы. Если  $p$  — это реакция системы на входное воздействие  $a$ , а  $q$  — реакция на входное воздействие  $b$ , то  $(p + q)$  — это реакция системы на входное воздействие  $(a + b)$ .

**Анализ** — Прямое преобразование Фурье, позволяющее перейти от временного представления сигнала к частотному. (См. также *Синтез*.)

**Антителайзинговый фильтр** — Аналоговый ФНЧ, включаемый перед АЦП. Предназначен для устранения эффектов наложения спектров при дискретизации. Подавляет все частоты, превышающие половину частоты дискретизации.

**Апертура выборки** — Размер участка непрерывного изображения, которому соответствует отдельный пиксель цифрового изображения. Обычно апертура выборки совпадает по размеру с интервалом дискретизации.

**Ассемблер** — Машино-ориентированный язык программирования. Является языком низкого уровня и служит для описания операций, выполняемых на уровне регистров и внутренних аппаратных средств микропроцессора. (См. также *Язык программирования высокого уровня*.)

**Ассоциативность операции свёртки** — Записывается следующим образом:  $(a[n] * b[n]) * c[n] = a[n] * (b[n] * c[n])$ . Свойство ассоциативности имеет большое значение при анализе последовательно соединённых цифровых фильтров.

**Бабочка** — Базовая операция алгоритма БПФ. Преобразование пары комплексных чисел в пару других комплексных чисел.

**Базисная функция** — Одна из функций, составляющих базис разложения сигнала. Например, базис разложения в ряд Фурье составляют синусоиды и косинусоиды единичной амплитуды.

**Бейсик** — Язык программирования высокого уровня, разработанный в 1964 г. Джоном Кемени и Томасом Курцем. Отличается простотой программирования, но вследствие этого имеет много недостатков. Большинство программ в данной книге написано на Бейсике.

**Белый шум** — Случайный шум, спектральная плотность которого не зависит от частоты, что возможно только при отсутствии взаимосвязи временных отсчётов сигнала. (См. также *Шум 1/f*.)

**Бесконечная импульсная характеристика (БИХ)** — Импульсная характеристика, имеющая бесконечное число ненулевых отсчётов. Примером является затухающая по экспоненте импульсная характеристика. Чаще всего данный термин применяют, чтобы подчеркнуть, что в основе фильтрации лежит рекурсивный алгоритм, а не операция свёртки.

**Биквадратный фильтр** — Аналоговое или цифровое звено, передаточная функция которого имеет два полюса и не более двух нулей. Используются для построения более сложных фильтров, для чего их соединяют последовательно.

**Билинейное преобразование** — Отображение  $S$ -плоскости на  $Z$ -плоскости, позволяющее осуществлять переход от аналоговых фильтров к цифровым.

**Бит-реверсная сортировка** — Адресация с инвертированием битов адреса, которая используется в алгоритме БПФ. Выполняется над двоичными битами, при этом биты переставляются в обратном порядке.

**Быстрая свёртка** — Метод вычисления свёртки сигналов путём перемножения их частотных спектров. Термин «быстрая» указывает на то, что в основе перехода от временной области к частотной лежит *быстрое преобразование Фурье*.

**Быстрое преобразование Фурье (БПФ)** — Эффективный алгоритм для выполнения *дискретного преобразования Фурье* (ДПФ). В ряде случаев позволяет сократить объём вычислительных затрат в сотни раз.

**Векторное преобразование** — Метод описания  $RLC$ -цепей в комплексной форме, позволяющий определить их частотную характеристику. Сопротивления присутствующих в схеме резисторов, индуктивностей и ёмкостей выражаются в виде следующих комплексных выражений:  $R$ ,  $j\omega L$ ,  $1/j\omega C$ .

**Взаимная корреляция** — Сигнал, который получается в результате применения операции корреляции к двум разным сигналам. Наличие пиков во взаимной корреляции указывает на имеющееся сходство между двумя данными сигналами. (См. также *Автокорреляция*.)

**Вокализованный звук** — Вокализованными называются звуки человеческой речи, которые образуются в результате колебания голосовых связок. Примером вокализованных звуков являются все гласные звуки. (См. также *Фрикативный звук*.)

**Восстановление сигнала** — Процесс возвращения искажённому сигналу его первоначальной формы. Одна из важнейших задач фильтрации.

**Восстановление Фурье** — Метод восстановления изображения по отдельным его видам (проекциям), применяемый в компьютерной томографии.

**Временная область** — Способ описания сигнала, при котором в качестве независимой переменной выступает время. Является основной областью, из которой выполняются преобразования в другие области представления сигналов.

**Выравнивание гистограммы** — Метод обработки изображения, при котором форма гистограммы используется для преобразования шкалы яркости. Выравнивание гистограммы увеличивает контрастность более крупных областей по сравнению с мелкими областями.

**Высота тона** — Субъективное человеческое ощущение основной частоты непрерывного гармонического сигнала. (См. также *Тембр*.)

**Гамма-кривая** — Математическая функция или таблица соответствия, связывающая числовые величины хранящихся в памяти данных и яркость соответствующих пикселей изображения. (См. также *Преобразование шкалы серого*.)

**Гарвардская архитектура** — Архитектура процессора, использующая для повышения производительности раздельные области памяти и соответствующие адресные шины для кода и данных; лежит в основе ПЦОС. (См. также *Фоннеймановская архитектура*.)

**Гармоника** — Одна из множества частотных составляющих периодического сигнала. Все гармоники кратны основной частоте: первая гармоника совпадает с основной частотой, частота второй гармоники в 2 раза выше, третьей — в 3 раза выше и т. д.

**Гидролокатор (сонар)** — Гидролокационная установка, предназначенная для обнаружения подводных лодок и прочих скрытых под водой объектов (от англ. sonar — SOund

**Navigation And Ranging).** Активный гидролокатор основан на активной эхо-локации, пассивный гидролокатор только прослушивает подводную обстановку.

**Гиперпространство** — этот термин вводится в приложениях, связанных с обнаружением целей и нейронно-сетевым анализом, при графической интерпретации одновременно более трёх параметров. При этом графической интерпретацией одного параметра является *числовая прямая*, двух параметров — *плоскость*, трёх параметров — *пространство*.

**Гистограмма** — Графическое отображение закона распределения одного из параметров сигнала. По оси абсцисс откладываются возможные значения этого параметра, а по оси ординат — число событий, когда данный параметр принимал соответствующие значения.

**Гомоморфная обработка** — Метод цифровой обработки для разделения сигналов, которые были объединены нелинейной операцией, такой как умножение или свёртка. При гомоморфной обработке нелинейный случай сводится к линейному.

**Граничная характеристика** — Реакция системы на входной сигнал с резким фронтом — границу (в обработке изображений). Кругизна граничной характеристики определяет разрешающую способность системы.

**Граничная частота (частота среза)** — Частота, разграничитывающая полосу пропускания и полосу подавления сигналов в аналоговых и цифровых фильтрах. Обычно соответствует точке, в которой АЧХ фильтра достигает уровня 0.707 (−3 дБ).

**Двойная точность** — 64-битное представление чисел. (См. также *Одинарная точность*.)

**Двунаправленная фильтрация** — Метод получения линейной ФЧХ при построении рекурсивного фильтра. Сначала сигнал пропускается через фильтр в одном направлении (слева направо), а затем — в обратном (справа налево).

**Действительная часть** — Одна из двух компонент в алгебраическом представлении комплексного числа, которая не содержит мнимой единицы  $j$  (например, действительная часть числа  $3 + 2j$  равна 3). Действительной частью действительного преобразования Фурье называется совокупность амплитуд косинусных компонент, несмотря на то что они не содержат  $j$ .

**Действительное БПФ** — Модификация алгоритма БПФ. Выполняется примерно на 30% быстрее стандартного алгоритма БПФ в тех случаях, когда сигнал во временной области представлен только действительными числами (т. е. мнимые части всех чисел равны нулю).

**Действительное ДПФ** — Дискретное преобразование Фурье, оперирующее только с действительными числами. Уступает по своим возможностям комплексному ДПФ, но проще в практической реализации. (См. также *Комплексное БПФ*.)

**Действительное преобразование Фурье** — Любая из форм семейства преобразований Фурье, в которой используются только действительные числа. (См. также *Комплексное преобразование Фурье*.)

**Декомпозиция с прореживанием** — Разложение, при котором отсчёты сигнала разбиваются на две группы: чётные и нечётные. Необходимо для алгоритма БПФ.

**Дельта-кодирование** — Кодирование сигналов, основанное на измерении разности между соседними отсчётами. Термин используется в аналого-цифровом преобразовании, сжатии данных и многих других практических приложениях.

**Дельта-сигма модуляция** — Наиболее распространённый в звукозаписи вид аналого-цифрового преобразования. Отличается очень высокой частотой дискретизации и использованием всего одного бита на отсчёт; затем полученный сигнал подвергается децимации.

**Дельта-функция** — Нормированный импульс. Дискретная дельта-функция принимает нулевые значения во всех точках, кроме нулевой, где она равна единице. Непре-

рывная дельта-функция довольно абстрактна, поскольку она представляет собой бесконечно короткий импульс бесконечно большой амплитуды, но имеющий при этом единичную площадь.

**Децимация** — Вторичная дискретизация, т. е. понижение частоты дискретизации дискретного сигнала. Как правило, перед децимацией сигнал пропускают через ФНЧ. (См. также *Интерполяция*.)

**Динамический диапазон** — Отношение наибольшей амплитуды входного сигнала, на обработку которой рассчитана данная система, к уровню собственных шумов системы. Используется при оценке необходимой разрядности АЦП. Кроме амплитуды, данный термин применим и к другим параметрам. (См. также *Динамический диапазон по частоте*.)

**Динамический диапазон по частоте** — Отношение верхней граничной частоты к нижней граничной частоте рабочего диапазона системы. Динамический диапазон у аналоговых систем намного шире, чем у цифровых.

**Дискретная производная** — Операция, применяемая для решётчатых функций, аналогичная нахождению производной непрерывной функции. Дискретную производную чаще называют *первой разностью*.

**Дискретное интегрирование** — Операция, аналогичная интегрированию непрерывных сигналов. Дискретное интегрирование называют также *суммированием с нарастающим итогом*.

**Дискретное косинусное преобразование (ДКП)** — Преобразование, близкое преобразованию Фурье. Позволяет произвести разложение сигнала на косинусоидальные компоненты. Применяется в алгоритмах сжатия изображений.

**Дискретное преобразование Фурье (ДПФ)** — Один из вариантов преобразования Фурье, при котором предполагается, что сигнал может быть описан *дискретной функцией времени* и является *периодическим*.

**Дискретный сигнал** — Сигнал, представляющий собой последовательность отсчётов, взятых с некоторым шагом по времени. При использовании нормированного времени областью определения является множество целых чисел, а сам сигнал описывается решётчатой функцией. Цифровая техника, такая как компьютер, работает с дискретными сигналами.

**Единичная окружность** — Окружность единичного радиуса на  $Z$ -плоскости. Частотная характеристика дискретной системы задаётся на единичной окружности.

**Единичный импульс** — То же, что *дельта-функция*.

**Зависимая переменная** — Параметр, участвующий в описании сигнала, который может быть выражен некоторым образом через другой параметр — независимую переменную. Например, напряжение сигнала может меняться со временем, т. е. напряжение — зависимая переменная, а время — независимая.

**Закрытие** — Морфологическая операция, заключающаяся в сжатии кадра и его последующем расширении.

**Затухание в полосе подавления** — Параметр фильтра, показывающий, во сколько раз уменьшается амплитуда сигнала в полосе подавления. Обычно измеряется в децибелах. Характеризует качество фильтра частотной селекции.

**Измеритель пространственного разрешения методом пар линий** — Устройство, предназначенное для измерения разрешающей способности систем формирования изображений. Используется набор светлых и тёмных полос, частота чередования которых постепенно увеличивается от одного края к другому.

**Импульс** — В данной книге термином «импульс» обозначен дельта-импульс — сигнал, принимающий нулевое значение везде, за исключением очень короткого времен-

ного интервала. В случае дискретных сигналов дельта-импульс содержит только один ненулевой отсчёт. Для непрерывных сигналов дельта-импульс — это сигнал, продолжительность которого несравненно меньше продолжительности переходных процессов всех внутренних узлов устройства.

**Импульсная декомпозиция** — Разложение, при котором дискретный сигнал длительностью  $N$  отсчётов разбивается на  $N$  коротких импульсов (дельта-импульсов). Импульсная декомпозиция лежит в основе операции свёртки.

**Импульсная характеристика** — Реакция системы на единичный импульс (дельта-функцию) при нулевых начальных условиях.

**Импульсный портрет** — Сигнал, представленный последовательностью дельта-импульсов, следующих с постоянным временным интервалом.

**Инвариантность к сдвигу** — Свойство системы, выражющееся в том, что сдвиг входного сигнала приводит к такому же сдвигу в выходном, остальные характеристики выходного сигнала при этом не меняются. Инвариантность к сдвигу означает, что свойства системы не зависят от времени (или другой независимой переменной).

**Инверсия АЧХ** — Метод пересчёта весовых коэффициентов КИХ-фильтра, при котором его АЧХ отражается в направлении сверху вниз. В результате инверсии АЧХ ФНЧ преобразуется в ФВЧ, а полосовой фильтр — в режекторный.

**Интеграл свёртки** — Математическое выражение, определяющее операцию свёртки применительно к непрерывным системам. (См. также *Сумма свёртки*.)

**Интерполяция** — Повышение частоты дискретизации цифрового сигнала. Как правило, заключается во введении дополнительных нулевых отсчётов между отсчётами исходного сигнала и последующей обработке сигнала фильтром низких частот. (См. также *Децимация*.)

**Искусственное зашумление** — Искусственное добавление шума к аналоговому сигналу на входе АЦП для предотвращения эффекта «залипания», возникающего при малых изменениях входного напряжения.

**Истина отрицательная** — Одно из четырёх возможных событий при обнаружении цели: цель отсутствует, и принимается верное решение о том, что цели нет.

**Истина положительная** — Одно из четырёх возможных событий при обнаружении цели: цель существует, и принимается верное решение о том, что цель обнаружена.

**Исходный текст программы** — Программа, написанная разработчиком в форме текстового документа. Программа в форме исходного текста ещё не готова к выполнению на компьютере, в отличие от *исполняемого кода* программы.

**Итеративный метод** — Метод нахождения решения путём постепенного изменения переменных в правильном направлении — в направлении сходимости алгоритма. Широко распространён в компьютерной томографии и нейронных сетях.

**Кадр** — Фрагмент телевизионного сигнала, которому соответствует отдельное изображение. В стандарте NTSC, принятом в США и Японии, в секунду передаётся около 30 кадров (точнее 29.97).

**Карта нулей и полюсов** — Графическое отображение нулей и полюсов на комплексной  $S$ - или  $Z$ -плоскости.

**Каузальный сигнал** — Любой сигнал, у которого отсчёты с отрицательными индексами принимают нулевые значения.

**Кепстр** — Термин образован перестановкой букв в слове «спектр». Заменяет в гомоморфной обработке спектр при частотно-временных преобразованиях.

**Классификатор** — Параметр, характеризующий большой массив данных. Примеры классификаторов: размер области, максимальная яркость, чёткость изображения и т. д. Классификаторы используются при распознавании образов.

**Кнопка** — Форма импульсной характеристики фильтров, применяемых в обработке изображений. Внутри круга заданного радиуса фильтр имеет постоянный коэффициент передачи, а за пределами данного круга коэффициент передачи равен нулю.

**Кодирование в частотной области** — Один из вариантов «размещения» информации в сигнале, при котором она содержится в амплитуде, частоте или фазе синусоидальных компонент сигнала, т. е. в частотной области. Наилучшим примером служит звуковой сигнал. (См. также *Кодирование во временной области*.)

**Кодирование во временной области** — Один из вариантов «размещения» информации в сигнале, при котором она содержится в форме огибающей. (См. также *Кодирование в частотной области*.)

**Кодирование длин серий** — Простой метод сжатия сигнала, при котором последовательность многократно повторяющихся символов заменяется кодом, содержащим информацию о значении повторяющихся символов и протяжённости цепочки.

**Кодирование Хаффмана** — Метод сжатия данных, при котором более часто встречающиеся символы представляются меньшим числом битов, чем символы, встречающиеся редко.

**Коммутативность операции свёртки** —  $a[n] * b[n] = b[n] * a[n]$ .

**Компандирование** — Схема кодирования речевых сигналов, позволяющая использовать 8 бит вместо 12 за счёт использования нелинейного преобразования *S*-типа. Распространение получили два варианта компандирования: А-закон получил распространение в странах Европы, а  $\mu$ -закон — в США.

**Комплексная плоскость** — Графическое представление множества комплексных чисел, при котором в декартовых координатах по оси абсцисс откладывается действительная составляющая комплексного числа, а по оси ординат — мнимая составляющая. Для сравнения: графическим представлением множества действительных чисел является *числовая ось*.

**Комплексная частотная характеристика (КЧХ)** — Зависимость комплексного коэффициента передачи линейной системы от частоты гармонического воздействия. КЧХ является функцией частоты и связана преобразованием *Фурье* с импульсной характеристикой системы.

**Комплексная экспонента** — Выражение вида  $e^{a+bi}$ . Широко применяется при расчетах в науке и технике, поскольку уравнение Эйлера позволяет выразить синус через комплексную экспоненту.

**Комплексное ДПФ** — Дискретное преобразование *Фурье*, при котором операции производятся над комплексными числами. Более сложный алгоритм по сравнению с *действительным ДПФ*.

**Комплексное преобразование Фурье** — Преобразование *Фурье*, в основе которого лежат операции над комплексными числами. Может быть представлено в одной из четырёх форм. (См. для сравнения *Действительное преобразование Фурье*.)

**Комплексное сопряжение** — Изменение арифметического знака мнимой части комплексного числа на противоположный. Символ комплексного сопряжения — звёздочка. Например, если  $A = 3 + 2j$ , то  $A^* = 3 - 2j$ .

**Комплексное число** — Число, в котором кроме обычной действительной компоненты присутствует также мнимая часть, содержащая  $j = \sqrt{-1}$ . Например,  $3 + 2j$ .

**Компьютерная томография** — Метод восстановления изображений внутренних органов по рентгеновским снимкам. Компьютерная томография в медицине является одним из самых первых практических приложений ЦОС.

**Конечная импульсная характеристика (КИХ)** — Импульсная характеристика, имеющая конечное число ненулевых отсчётов. Чаще всего данный термин применяют, что-

бы подчеркнуть, что в основе фильтрации лежит операция свёртки, а не рекурсивные алгоритмы.

**Контрастность** — Различие в яркости объекта и фона. (См. также *Яркость*.)

**Корреляция** — Математическая операция, подобная операции свёртки, но отличающаяся от неё тем, что отсчёты одного из участвующих в ней сигналов берутся в обратном порядке (отражаются слева направо). Корреляция является оптимальным способом обнаружения сигнала заданной формы.

**Коэффициент вариации** — Отношение среднеквадратического значения шума к его среднеарифметическому значению, выраженное в процентах.

**Коэффициенты рекурсивного фильтра** — Коэффициенты уравнения рекурсивного фильтра. Вид бесконечной импульсной характеристики рекурсивного или БИХ-фильтра определяется значениями его коэффициентов.

**Краевой эффект** — Ухудшение качества сигнала на выходе фильтра, вызванное усечением импульсной характеристики.

**Кривая Гаусса** — Колоколообразная кривая вида  $e^{-x^2}$ . Применяется для математического описания плотности вероятности *нормального распределения*.

**Круглая ФРТ** — Форма импульсной характеристики фильтров, применяемых в обработке изображений. Внутри круга заданного радиуса фильтр имеет постоянный коэффициент передачи, а за пределами данного круга коэффициент передачи равен нулю.

**Круговая свёртка** — Наложение сигналов во временной области, которое может происходить при перемножении спектров в частотной области. Выражается во взаимном проникновении соседних временных сегментов друг в друга.

**Круговая частота** — Частота, выраженная в радианах в секунду. Круговая частота равна умноженной на  $2\pi$  линейной частоте (в герцах).

**Крутизна спада АЧХ** — Характеризуется относительной шириной переходной полосы фильтра, разделяющей полосу пропускания и полосу подавления. Чем круче спад АЧХ, т. е. чем уже переходная полоса, тем лучше разрешающая способность фильтра.

**Кули и Тьюки** — Считается, что алгоритм БПФ впервые был разработан Дж. Кули и Дж. Тьюки, которые опубликовали его в США в 1965 году (алгоритм БПФ Кули—Тьюки).

**Линейная система** — Система, для которой справедливы свойства аддитивности и однородности.

**Линейная фаза** — ФЧХ системы, имеющая вид прямой линии. Во многих случаях требуется использовать системы с линейной ФЧХ, потому что для таких систем характерна симметрия импульсной характеристики и симметрия нарастающего и спадающего фронтов обработанного прямоугольного импульса. (См. также *Нулевая фаза*.)

**Линейность статической характеристики** — Линейность системы по отношению к статическим входным воздействиям, т. е. к таким воздействиям, параметры которых остаются постоянными на всем интервале наблюдения (например, постоянное входное напряжение). Свойство линейности выражается в данном случае в том, что реакция может быть представлена в виде произведения входного воздействия на некоторую константу.

**Ложь отрицательная** — Одно из четырёх возможных событий при обнаружении цели: цель присутствует, а принимается ошибочное решение о том, что цели нет.

**Ложь положительная** — Одно из четырёх возможных событий при обнаружении цели: цели нет, а принимается ошибочное решение о том, что цель обнаружена.

**ЛЧМ-система** — Системы с линейной частотной модуляцией (ЛЧМ) применяются в радиолокации и гидролокации. Передаваемый прямоугольный импульс преобра-

зуется в ЛЧМ-импульс большей длительности, а на приёмной стороне осуществляется обратное преобразование.

**Математическое ожидание** — Среднее значение сигнала или некоторого массива данных.

**Метод замещения** — Метод, основанный на введении комплексных чисел при решении различных физических задач, в том числе при расчёте электрических схем. После выполнения сложных преобразований в комплексной области выполняется обратный переход к действительным числам. (См. также *Метод математической эквивалентности*.)

**Метод математической эквивалентности** — Метод, основанный на использовании уравнений Эйлера для описания действительных функций в комплексной форме. (См. также *Метод замещения*.)

**Метод наискорейшего спуска** — Один из методов организации итеративных процедур нахождения минимума функции. Идея метода наискорейшего спуска аналогична постоянному движению вниз по склонам холмистой местности.

**Метод обратных проекций** — Метод восстановления изображения по отдельным его видам (проекциям), применяемый в компьютерной томографии. Не позволяет получать изображения высокого качества, однако служит основой для другого, более эффективного метода.

**Метод обратных проекций с фильтрацией** — Метод восстановления изображения по отдельным его видам (проекциям), применяемый в компьютерной томографии. Проекции предварительно фильтруются, а затем применяется метод обратных проекций.

**Мнимая часть** — Одна из двух компонент в алгебраическом представлении комплексного числа, которая представляется умноженной на мнимую единицу  $j$  (например, мнимая часть числа  $3 + 2j$  равна 2). Мнимой частью действительного преобразования Фурье называется совокупность амплитуд синусоидальных компонент, несмотря на то что они не содержат  $j$ .

**Многоскоростная обработка** — Цифровая обработка сигналов, при которой происходит изменение его частоты дискретизации. Используется, например, для повышения качества работы АЦП и ЦАП, и позволяет при этом упростить аналоговую часть.

**Модуляционная передаточная функция** — Термин, применяемый в области обработки изображений для обозначения *частотной характеристики*.

**Модуляция амплитудная** — Изменение амплитуды несущей частоты в соответствии с передаваемым сигналом (например, аудиосигналом). Как правило, амплитудная модуляция реализуется на основе умножения сигналов.

**Морфологическая обработка** — Нелинейное преобразование, производимое над двумерным изображением. Например, расширение и сжатие.

**Морфинг** — Плавное преобразование одного изображения в другое с помощью геометрических операций и цветовой интерполяции. Используется в компьютерной анимации для создания специальных эффектов, например для превращения фигуры человека в обратную.

**Мультиплексирование** — Объединение двух и более сигналов для совместной передачи. Может быть реализовано самыми разными способами.

**Наложение сигналов во временной области** — Перекрытие временных сегментов во временной области, возникающее в результате обработки сигнала в частотной области, например в результате выполнения *круговой свёртки*.

**Наложение спектров** — Проникновение спектра сигнала из одного частотного диапазона в другой, связанное с дискретизацией аналогового сигнала или другими нелинейными преобразованиями. Как правило, приводит к потере полезной информации.

**Наложение спектров в частотной области** — Наложение спектров, возникающее в результате различных преобразований во временной области. Например, наложение спектров при аналого-цифровых преобразованиях.

**Независимая переменная** — Параметр, через который может быть выражен другой параметр сигнала, называемый зависимым. Например, напряжение сигнала может меняться со временем, т. е. напряжение — зависимая переменная, а время — независимая.

**Неискажающая передача гармонических сигналов** — Важное свойство линейных систем, заключающееся в том, что реакцией на синусоидальный входной сигнал всегда является синусоидальный выходной сигнал той же частоты.

**Непрерывный сигнал** — Сигнал, заданный на непрерывном множестве чисел (в противоположность *дискретному сигналу*). Примером может служить постепенное изменение уровня напряжения. Непрерывный сигнал часто называют *аналоговым сигналом*.

**Несущая** — Синусоидальное колебание высокой частоты, которое используется для передачи низкочастотного сигнала, содержащего информационное сообщение, при амплитудной модуляции.

**Нормальное распределение** — Колоколообразная кривая вида  $e^{-x^2}$ . (См. также *кривая Гаусса*.)

**Нулевая фаза** — ФЧХ систем, равная нулю на любой частоте. Нулевая фаза возможна лишь в том случае, когда импульсная характеристика симметрична относительно нулевого отсчёта.

**Нуль** — *здесь*: Данный термин связан с преобразованием Лапласа и Z-преобразованием. Если передаточная функция некоторой системы представляется в S-области или в Z-области в виде отношения полиномов, то корни числителя называются *нулями* передаточной функции. (См. также *Полюс*.)

**Область описания** — Физическая величина, соответствующая независимой переменной. Например, для описания процесса изменения напряжения используется *временная область*. Областью определения при описании сигналов различного рода могут служить *частотная область* (Фурье-образы) и *пространственная область* (изображение).

**Область сходимости** — Применительно к преобразованию Лапласа или к Z-преобразованию область сходимости представляет собой множество точек S- или Z-плоскости соответственно, в которых преобразование определено.

**Обнаружение цели** — принятие решения о наличии или отсутствии обнаруживаемого объекта на основе некоторых измеренных параметров.

**Обработка в реальном времени** — Обработка данных по мере их поступления, в отличие от процедур сбора данных для последующей обработки. Примером обработки в реальном времени является подавление эхо-сигналов в телефонных линиях большой протяжённости при помощи цифровых устройств.

**Обратное преобразование** — Уравнение синтеза преобразования Фурье, позволяющее выполнить переход от частотной области к временной. (См. также *Прямое преобразование*.)

**Обращение АЧХ** — Метод пересчёта весовых коэффициентов КИХ-фильтра, при котором его АЧХ отражается в направлении слева направо, в результате чего АЧХ ФНЧ преобразуется в АЧХ ФВЧ.

**Обращение свёртки** — Выполнение обратной по отношению к свёртке операции. Если  $x[n] * h[n] = y[n]$ , то обращение свёртки заключается в поиске  $x[n]$  по известным  $h[n]$  и  $y[n]$ .

**Обучающий алгоритм** — Процедура нахождения набора весовых коэффициентов нейронной сети на основе имеющихся примеров правильной работы сети.

**Одинарная точность** — 32-битное представление чисел. (См. также *Двойная точность*.)

**Однородность** — Математическое свойство всех линейных систем: если  $y[n]$  — реакция на входное воздействие  $x[n]$ , то реакцией на  $kx[n]$  будет  $ky[n]$ , где  $k$  — произвольная константа.

**Однородный фильтр** — Фильтр, в котором каждый отсчёт выходного сигнала образуется в результате усреднения многих последовательно выбранных отсчётов входного сигнала.

**Окно Блэкмана** — Гладкая кривая, которая применяется при расчёте фильтров и в спектральном анализе и описывается функцией  $0.42 - 0.5\cos(2\pi n/M) + 0.08\cos(4\pi n/M)$ .

**Окно с плоской вершиной** — Оконная функция, используемая в спектральном анализе, которая позволяет обеспечить высокую точность при измерении амплитуд спектральных составляющих сигнала. Для данной цели можно воспользоваться оконным фильтром.

**Окно Хэмминга** — Гладкая кривая, которая применяется при расчёте фильтров и в спектральном анализе и описывается функцией  $0.54 - 0.46\cos(2\pi n/M)$ , где  $n = 0 \dots M$ .

**Оконный фильтр** — Одна из разновидностей цифровых фильтров частотной селекции.

**Октава** — Изменение частоты в 2 раза.

**Оптимальный фильтр** — Фильтр, параметры которого являются «наилучшими» для решения поставленной задачи. Например, фильтры Винера обеспечивают оптимальное отношение сигнал/шум; согласованные фильтры являются оптимальными в задачах обнаружения сигнала.

**Основная мембрана внутреннего уха** — Основная мембрана состоит из большого количества волокон различной длины. Более высокие частоты вызывают колебания более коротких волокон, и наоборот. Это позволяет нервным клеткам различать звуки разных частот, т. е. выполнять спектральный анализ.

**Основная частота** — Минимальная частота, с которой повторяется форма периодического сигнала. (См. также *Гармоника*.)

**Основной порядок строк** — Схема преобразования изображения в последовательность отсчётов, при которой элементы изображения читаются слева направо и сверху вниз аналогично чтению книги.

**Открытие** — Морфологическая операция, состоящая из двух этапов — расширения и сжатия.

**Отрицательная частота** — Математическая абстракция, используемая при описании комплексного преобразования Фурье.

**Ошибка квантования** — Ошибка, связанная с эффектом квантования в цифровых системах. В большинстве случаев ошибка квантования не превышает по модулю половину младшего бита двоичного числа. Среднеквадратическая ошибка квантования составляет  $1/\sqrt{12}$  младшего бита. Ошибку квантования называют также *шумом квантования*.

**Пара линий** — Термин, относящийся к обработке изображений. Эквивалентен термину «период». Например, 5 периодов/мм — это то же самое, что 5 пар линий/мм.

**Пара преобразований Фурье** — Пара преобразований, называемых прямым и обратным преобразованиями Фурье, связывающих между собой временное и частотное представления сигнала. Например, прямоугольный импульс связан парой преобразований Фурье с функцией вида  $\sin x/x$ .

**Параллельное соединение каскадов** — Схема соединения, при которой все каскады используют общий входной сигнал, а их выходные сигналы складываются.

**Пассивная гидролокация** — Обнаружение шумов, производимых подводными лодками и другими подводными объектами. Обеспечивает скрытое наблюдение.

**Первая разность** — Операция, аналогичная нахождению первой производной для аналоговых сигналов. Первую разность называют также *дискретной производной*.

**Передаточная функция** — Для аналоговых систем передаточная функция — это отношение преобразования Лапласа реакции системы к преобразованию Лапласа соответствующего входного воздействия при нулевых начальных условиях. Для цифровых систем вместо преобразования Лапласа используется  $Z$ -преобразование. Если передаточная функция представлена в виде отношения полиномов, то она может быть описана в виде совокупности нулей и полюсов.

**Переменный ток** — Термин, обозначающий часть сигнала, колеблющуюся относительно среднего значения (DC).

**Переходная полоса** — Полоса частот между полосой пропускания и полосой подавления фильтра частотной селекции. Переходная полоса соответствует спаду АЧХ.

**Переходная характеристика** — Реакция системы на единичный скачок при нулевых начальных условиях.

**Пиксель** — Минимальный элемент цифрового изображения (от англ. picture element).

**Плавающая точка** — Один из двух наиболее распространённых способов представления компьютерных данных. Формат чисел с плавающей точкой использует поле мантиссы и поле экспоненты. (См. также *Фиксированная точка*.)

**Плотность распределения вероятности** — Вероятность того, что случайная величина примет определённое значение.

**Погрешность измерения** — Постоянная составляющая ошибки измерения (или предсказания), одинаковая для всех экспериментов. Погрешность связана с *систематической* (повторяющейся от опыта к опыту) ошибкой измерения. (См. также *Точность измерения*.)

**Подчёркивание контуров** — Свойство алгоритма делать границы объекта более отчётливыми. (См. также *Увеличение чёткости*.)

**Поле** — При чересстрочной развёртке телевизионных сигналов сначала передаются все нечётные строки кадра, а затем — все чётные. При этом все нечётные строки составляют *нечётное поле* кадра, а все чётные — *чётное поле*.

**Полный видеосигнал** — Аналоговый видеосигнал, состоящий из сигналов цветности и яркости, сигнала гашения, а также кадровых и строчных синхроимпульсов.

**Полоса подавления** — Диапазон частот входного сигнала, которые требуется подавить.

**Полоса пропускания** — Диапазон частот, которые фильтр должен пропускать без искажений.

**Полутон** — Общепринятый метод вывода изображений на печать. Различные оттенки серого получаются в результате использования разных наборов чёрных точек. Для цветных полутона применяются красные, зелёные и синие точки.

**Полутоновое изображение** — Цифровое изображение, содержащее оттенки серого (от чёрного до белого). Также называется чёрно-белым изображением или изображением в шкале серого.

**Полюс** — термин, связанный с преобразованием Лапласа и  $Z$ -преобразованием. Если передаточная функция некоторой системы представляется в  $S$ -области или в  $Z$ -области в виде отношения полиномов, то корни знаменателя называются *полюсами*, а корни числителя — *нулями* передаточной функции.

**Последовательное (каскадное) включение** — Система, образованная двумя и более звеньями, в которой выходной сигнал одного звена является одновременно входным сигналом следующего.

**Потенциальная яма** — здесь: Участок ПЗС, воспринимающий световой поток.

**Представление комплексных чисел в полярной системе координат** — В полярной системе координат гармонический сигнал задаётся амплитудой и фазой:  $M\cos(\omega t + \phi)$ , где  $M$  — амплитуда,  $\phi$  — фаза. (См. также *Представление комплексных чисел в прямоугольной системе координат*.)

**Представление комплексных чисел в прямоугольной системе координат** — В прямоугольной системе координат гармонический сигнал задаётся в следующей форме:  $A\cos(\omega t) + B\sin(\omega t)$ , где  $A$  и  $B$  называются действительной и мнимой частями комплексного числа.

**Преобразование** — Правило, уравнение или алгоритм, описывающий переход от одной формы представления данных к другой.

**Преобразование Лапласа** — Метод математического анализа систем, описываемых дифференциальными уравнениями. Представляет собой мощное средство разработки электронных устройств, в том числе аналоговых фильтров. Преобразование Лапласа позволяет перейти от временного описания сигнала к описанию в  $S$ -области.

**Преобразование Фурье** — Семейство математических выражений, позволяющих представить сигнал в форме суммы синусоид. При использовании комплексных функций вместо синусоид используются комплексные экспоненты.

**Преобразование Фурье дискретного времени** — Один из вариантов преобразования Фурье, при котором предполагается, что сигнал может быть описан *дискретной функцией времени и не является периодическим*.

**Преобразование шкалы серого** — Функция, описывающая преобразование числовых величин, хранящихся в памяти данных, в яркость соответствующих пикселей изображения. (См. также *Гамма-кривая*.)

**Преобразователь Гильберта** — Система, обеспечивающая постоянный во всем рабочем диапазоне коэффициент передачи, равный 1, и постоянный фазовый сдвиг на  $90^\circ$ . Применяется в аппаратуре связи для выполнения модуляции. Может быть как аналоговым, так и цифровым.

**Преобразователь сигнала** — Любое устройство, позволяющее получить определённый выходной сигнал в ответ на входное воздействие.

**Прибор с зарядовой связью (ПЗС)** — Фоточувствительная полупроводниковая матрица. Используется в цифровых видеокамерах, сканерах и т. д.

**Просачивание спектральных составляющих** — Данный эффект проявляется в спектральном анализе и выражается в том, что из-за ограничения во времени выборки синусоидального сигнала при вычислении ДПФ в частотной области импульс получается немного «размазанным».

**Пространственная область** — Способ описания сигнала, при котором в качестве независимых переменных выступают пространственные протяжённости. Изображение — типичный сигнал, представленный в пространственной области.

**Пространство параметров** — Данный термин широко распространён среди специалистов по системам обнаружения целей. Графической интерпретацией одного параметра является *числовая прямая*, двух параметров — *плоскость*, трёх параметров — *пространство*, а более трёх параметров — *гиперпространство*.

**Прямое преобразование** — Уравнение анализа преобразования Фурье, позволяющее выполнить переход от временной области к частотной. (См. также *Обратное преобразование*.)

**Прямоугольное окно** — Функция, отсчёты которой принимают единичное значение внутри некоторого непрерывного интервала и равны нулю за его пределами. Умножение на прямоугольное окно используется для выделения сегмента сигнала, подлежащего обработке на текущем этапе.

**Рабочая характеристика приёмника** — Графическое отображение влияния выбора порога на эффективность обнаружения цели.

**Равенство Парсеваля** — Соотношение, связывающее энергию сигнала во временной области и энергию сигнала в частотной области.

**Радар** — Радиолокационная установка, предназначенная для обнаружения летательных аппаратов (от англ. RADAR — RAdio Detection And Ranging).

**Разложение** — Процесс разделения исходного сигнала на две и более аддитивных компонент. Термин часто используется при описании *прямого преобразования Фурье*, в результате которого сигнал раскладывается на синусоидальные компоненты.

**Разложение на основе сигналов с чётной и нечётной симметрией** — Способ разбиения сигнала на две составляющие, из которых одна подчиняется чётной, а другая — нечётной симметрии.

**Разностное уравнение** — Уравнение, связывающее текущий и предыдущие отсчёты выходного сигнала с текущим и предыдущими отсчётами входного сигнала. Называется также *рекурсивным уравнением*.

**Разрешение по частоте** — Способность различать (разделять) близко расположенные частоты.

**Растяжение шкалы серого** — Растигивание шкалы серого, позволяющее увеличить контрастность цифрового изображения и более чётко различать близкие по яркости участки. Однако уровни квантования, вышедшие при этом за пределы шкалы, сольются с белым или чёрным полями.

**Расширение** — Морфологическая операция. Применение операции растяжения к двоичным (бинарным) изображениям делает объекты крупнее и позволяет соединять разрозненные объекты в один.

**Реакция на импульс** — Реакция, появляющаяся на выходе фильтра в ответ на импульсное воздействие.

**Рекурсивное уравнение** — Уравнение, связывающее текущий и предыдущие отсчёты выходного сигнала с текущим и предыдущими отсчётами входного сигнала. Называется также *разностным уравнением*.

**Ряд Фурье** — Один из вариантов преобразования Фурье, при котором предполагается, что сигнал может быть описан *непрерывной функцией* времени и является *периодическим*.

**Свёртка в частотной области** — Алгоритм выполнения свёртки, основанный на умножении частотных спектров сигналов.

**Сдвиг с вычитанием** — Операция, позволяющая создавать трёхмерные изображения или изображения рельефной поверхности.

**Сейсмология** — Раздел геофизики, изучающий механические свойства земной поверхности.

**Сепарабельность** — Свойство ФРТ системы, которое позволяет представить её в виде произведения двух одномерных векторов: горизонтальной и вертикальной проекций. Данное свойство используется для сокращения вычислительных затрат при выполнении свёртки изображения и ФРТ.

**Сжатие** — Морфологическая операция. Применение операции растяжения к двоичным (бинарным) изображениям делает объекты мельче и может привести к разделению объекта на несколько разрозненных частей.

**Сжатие без потерь** — Обратимое сжатие данных, после которого возможно полностью восстановить исходные данные. Например, сжатие данных методом Лемпела–Зива–Велча (алгоритм LZW).

**Сжатие с потерями** — Методы сжатия данных, позволяющие лишь приблизительно восстановить исходные данные. Такие методы позволяют достичь более высокой степени сжатия. Примером служит сжатие в стандарте JPEG.

**Сжатие с преобразованием** — Сжатие данных, при котором для кодирования более высоких частот используется меньшее количество бит. Хорошим примером сжатия с преобразованием является алгоритм сжатия, используемый в формате JPEG.

**Сигмоид** — S-образная кривая, применяемая в нейронных сетях.

**Сигнал** — Процесс изменения какого-либо физического параметра в зависимости от изменений другого параметра. Например, изменение *напряжения* с течением времени.

**Синтез** — Обратное преобразование Фурье, позволяющее перейти от частотного представления сигнала к временному. (См. также *Анализ*.)

**Система без памяти** — Система, в которой текущее значение выходной величины зависит только от текущего значения входной величины и не зависит от истории.

**Систематическая ошибка** — Составляющая ошибки измерения или предсказания, сохраняющая постоянное значение в разных наблюдениях. Определяет *погрешность измерения*. (См. также *Случайная ошибка*.)

**Слуховой нерв** — Нерв, передающий возникающие в кортиевом органе раздражения в головной мозг.

**Случайная ошибка** — Составляющая ошибки измерения или предсказания, принимающая различные значения в разных наблюдениях. Определяет *точность измерения*. (См. также *Систематическая ошибка*.)

**Согласованная фильтрация** — Метод, позволяющий обнаружить место (или сам факт) присутствия в сигнале участка заданной формы. Данный метод основан на корреляции, а реализуется в цифровых фильтрах в виде алгоритма свёртки.

**Спектральный анализ** — Подход, при котором сигнал рассматривается в виде совокупности амплитуд, частот и фаз составляющих его синусоидальных компонент. Основным инструментом спектрального анализа является *преобразование Фурье*.

**Спектрограмма** — Графическое отображение процесса изменения спектра сигнала во времени. Как правило, представляет собой двумерное изображение. Иногда называется *сонограммой*.

**Среднеквадратическое значение (СКЗ)** — Выражает среднюю величину случайных отклонений сигнала от уровня, принятого за нулевой. Среднеквадратическое значение часто применяется в технических приложениях и выражается квадратным корнем из среднего арифметического значения квадрата отклонений сигнала. (См. также *Среднеквадратическое отклонение*.)

**Среднеквадратическое отклонение (СКО)** — Выражает среднюю величину случайных отклонений сигнала от его среднего уровня. Среднеквадратическое отклонение выражается квадратным корнем из среднего арифметического значения квадрата отклонений сигнала. (См. также *Среднеквадратическое значение*.)

**Стандарт Hi-Fi** — Высококачественное воспроизведение музыки, например в CD-плеерах.

**Статистика Пуассона** — Статистическое описание флюктуаций, связанных с малым числом частиц, например электронов или фотонов рентгеновского и светового диапазонов. Для данного явления часто используются понятия *пуассоновский шум* и *флюктуационный шум*.

**Сумма свёртки** — Математическое выражение, определяющее операцию свёртки применительно к дискретным системам. (См. также *Интеграл свёртки*.)

**Суммирование с нарастающим итогом** — Операция, аналогичная интегрированию непрерывных сигналов. Суммирование с нарастающим итогом называют также *дискретным интегрированием*.

**Суммирование с перекрытием** — Метод обработки, при котором сигналы большой длительности предварительно разбиваются на отдельные сегменты.

**Сходимость** — Свойство итеративной процедуры, выражющееся в стремлении с течением времени к некоторому установившемуся значению (в таких случаях говорят «алгоритм сходится»).

**Тембр** — Субъективное человеческое ощущение высших гармоник непрерывного гармонического сигнала. (См. также *Высота тона*.)

**Теорема отсчётов** — Если аналоговый сигнал ограничен верхней частотой  $f$ , то чтобы исключить возможность потери информации, его дискретизацию необходимо осуществлять с частотой  $2f$ . Теорему отсчётов часто называют теоремой отсчётов Шеннона, теоремой Найквиста или теоремой Котельникова.

**Точность измерения** — Случайная для каждого эксперимента составляющая ошибки измерения или предсказания. Точность отдельного измерения определяется случайной ошибкой. (См. также *Погрешность измерения*.)

**Увеличение чёткости** — Операция, позволяющая сделать границы объектов более чёткими.

**Указатель** — Переменная, значением которой является адрес другой переменной.

**Улитка** — Улитковый или кортиев орган. Расположен во внутреннем ухе и состоит из слуховых клеток, к которым подходят веточки слухового нерва. Имеет форму морской улитки.

**Уравнение Эйлера** — Важное математическое уравнение, позволяющее представить функции синуса и косинуса в виде суммы комплексных экспонент.

**Уровень звукового давления** — Логарифмическая величина, служащая для измерения громкости звука: 0 дБ — едва различимый звук, 60 дБ — обычная речь, 140 дБ — болевой порог.

**Устройство захвата кадра (фрейм-граббер)** — АЦП, предназначенный для оцифровки и ввода в память телевизионного кадра.

**Устройство смешанной обработки** — Устройство, работающее одновременно с аналоговыми и цифровыми сигналами. Очевидно, что простейшим примером устройства смешанной обработки является АЦП, основное назначение которого состоит в преобразовании аналогового сигнала в цифровой.

**Физически реализуемая система** — Система, реакция на выходе которой равна нулю до момента поступления входного воздействия. Входное воздействие является причиной выходного и предшествует ему по времени, поэтому физически реализуемая система называется также *каузальной системой*. Импульсная характеристика физически реализуемой (каузальной) системы является *каузальным сигналом*.

**Фиксатор нулевого порядка** — Элемент схемы, выходной сигнал которого равен входному в дискретные моменты времени, а в остальное время сохраняется неизменным. Фиксатор нулевого порядка используется для перехода от цифровой формы представления сигнала к аналоговой в ЦАП, формируя ступенчатый выходной сигнал.

**Фиксированная точка** — Один из двух наиболее распространённых способов представления компьютерных данных. Чаще всего числа в формате представления с фиксированной точкой рассматриваются как целые. (См. также *Плавающая точка*.)

**Фильтр Баттерворт** — Предназначен для частотной селекции сигналов. Обеспечивает наибольшую крутизну спада АЧХ при достаточно малой неравномерности в полосе пропускания. Может быть как аналоговым, так и цифровым. Фильтр Баттерворт называют также фильтром с *максимально гладкой АЧХ*.

**Фильтр Бесселя** — Аналоговый фильтр, позволяющий получить максимально линейную ФЧХ. Такой фильтр отличается почти нулевым перерегулированием переходной характеристики и вносит симметричные искажения в нарастающий и спадающий фронты прямоугольного импульса. Используется для сглаживания сигналов, кодированных во временной области.

**Фильтр Винера** — Фильтр, который при заданных свойствах сигнала и помехи является оптимальным с точки зрения увеличения отношения сигнал/шум.

**Фильтр восстанавливающий** — ФНЧ, сглаживающий ступенчатый аналоговый сигнал на выходе ЦАП. Восстанавливающий фильтр подавляет все частоты, превышающие половину частоты дискретизации.

**Фильтр на переключаемых конденсаторах** — Аналоговый фильтр, основанный на быстром переключении конденсаторов и не требующий использования резисторов. Выполняется в виде интегральной микросхемы. Очень часто применяется для устранения эффекта наложения спектров в АЦП и в качестве *восстанавливающего фильтра* в ЦАП.

**Фильтр нечётного порядка** — Фильтр с нечётным количеством полюсов.

**Фильтр Чебышева** — Предназначен для частотной селекции сигналов. Обеспечивает более крутой спад АЧХ, чем фильтр Баттервортса, за счёт допущения неравномерности в полосе пропускания. Может быть как аналоговым, так и цифровым.

**Фильтр чётного порядка** — Фильтр с чётным количеством полюсов.

**Флуктуационный шум** — Статистическое описание флуктуаций, связанных с малым числом корпускул, например электронов или фотонов рентгеновского и светового диапазонов. Для данного явления часто используются понятия *пуассоновский шум* и *статистика Пуассона*.

**Фоннеймановская архитектура** — Архитектура процессора, использующая единое адресное пространство для хранения команд и данных. (См. также *Гарвардская архитектура*.)

**Формат изображения** — Отношение ширины изображения к высоте в пикселях с учётом отношения линейных размеров экрана; для монитора ПК и телевизионного экрана обычно составляет 4:3, для стандарта HDTV (телевидения высокой чёткости) — 16:9.

**Фрикативный звук** — К фрикативным (щелевым) звукам относятся звуки в, ф, з, с, ж, щ, й, х. Фрикативные звуки представляют собой шум, образующийся в результате различного рода завихрений, возникающих в потоке выдыхаемого человеком воздуха. (См. также *Вокализованный звук*.)

**Функция вида  $\sin x/x$**  — Это так называемая sinc-функция,  $\text{sinc}(a) = \sin(\pi a)/\pi a$ . Коэффициент  $\pi$  часто присутствует неявно, как компонент другой переменной  $x$ :  $\sin(x)/x$ . Данная функция играет важную роль в технике, поскольку является преобразованием Фурье прямоугольного импульса.

**Функция рассеяния точки (ФРТ)** — Термин, который используется в обработке изображений и служит эквивалентом импульсной характеристики.

**Функция расширения линии** — Реакция системы обработки изображений на входное изображение в виде тонкой линии.

**Целочисленный формат** — Представление данных в виде целых чисел: ..., -2, -1, 0, 1, 2, ... . Целочисленный формат часто называют форматом с *фиксированной точкой*. (См. также *Плавающая точка*.)

**Центральная предельная теорема** — Важная теорема в математической статистике. Вот одна из формулировок: сумма большого числа случайных величин подчиняется гауссовскому распределению, несмотря на то что отдельно взятые величины могут описываться любым другим законом распределения.

**Центральная ямка** — Область наибольшей остроты зрения, расположенная в центре сетчатки, на оптической оси.

**Циклический буфер** — Метод размещения данных в памяти устройств обработки сигналов реального времени, при котором последний поступивший отсчёт сигнала занимает место наиболее старого из хранящихся в буфере отсчётов.

**Цикличность** — Свойство сигнала повторяться через определённый период. Для периодических сигналов можно считать, что за последним отсчётом периода снова следует первый его отсчёт.

**Цифровой однополюсный фильтр** — Простейший рекурсивный фильтр, подобный по своим свойствам аналоговой *RC*-цепи.

**ЦПОС** — Класс микропроцессорных устройств, предназначенных для быстрого выполнения операций, связанных с ЦОС. В основе ЦПОС обычно лежит конвейерная и/или гарвардская архитектура. Часто используется термин RISC-процессоры.

**Частота Найквиста** — Верхняя граница основной полосы частот цифрового сигнала, которая согласно теореме Найквиста—Котельникова равна половине частоты дискретизации. В ряде зарубежных изданий частоту Найквиста отождествляют с частотой дискретизации.

**Частота среза на уровне  $-3$  дБ** — Частота, разделяющая полосу пропускания и полосу подавления фильтра. Определяется по достижению АЧХ фильтра уровня  $-3$  дБ (0.707 от максимального значения коэффициента усиления фильтра).

**Частотная область** — Способ описания сигнала, при котором в качестве независимой переменной выступает частота. Переход к частотной области осуществляется с помощью преобразования *Фурье*.

**Частотное мультиплексирование** — Метод, позволяющий передавать одновременно несколько сигналов за счёт размещения их в разных частотных диапазонах.

**Чересстрочное видео** — Изображение, создаваемое с помощью чересстрочной развертки, при которой кадр передаётся в два приёма: сначала чётные строки, затем нечётные. Используется для уменьшения мерцания телевизионного изображения.

**Чётно-нечётное разложение** — Способ разбиения сигнала на две составляющие, из которых одна подчиняется чётной, а другая — нечётной симметрии.

**Шаг (интервал) выборки** — Интервал между дискретными отсчётами, которыми представляется изображение при его преобразовании в цифровую форму. Определяется расстоянием между центрами соседних пикселей цифрового изображения.

**Ширина на уровне половины амплитуды** — Широко распространённый метод оценки ширины импульса, при котором ширина импульса измеряется на уровне половины его амплитуды.

**Шум  $1/f$**  — Вид шума, для которого характерно резкое увеличение амплитуды на низких частотах. Этот шум широко распространён в физических системах, однако мало изучен. (См. также *Белый шум*.)

**Шум квантования (округления)** — Ошибка, возникающая в цифровых системах вследствие округления результатов математических операций до ближайшего квантованного значения.

**Эллиптический фильтр** — Предназначен для частотной селекции сигналов. Обеспечивает очень крутой спад АЧХ за счёт допущения неравномерности как в полосе пропускания, так и в полосе подавления. Может быть как аналоговым, так и цифровым.

**Эффект Гиббса** — Усечение сигнала в одной области (временной или частотной) приводит к возникновению колебаний в другой области.

**Ядро свёртки** — Весовые коэффициенты нерекурсивного фильтра или отсчёты его импульсной характеристики. Используется также термин *ядро фильтра*.

**Ядро фильтра** — См. *Ядро свёртки*.

**Язык программирования высокого уровня** — C, BASIC, FORTRAN и др.

**Яркость** — Общая освещённость или затемнённость изображения. (См. также *Контрастность*.)

# Предметный указатель

## C

CISC-процессор 108

## D

Dolby 411

## E

EFM 410

EZ-KIT Lite 594

## J

JPEG 548

## L

LZW-сжатие 541

## M

MFLOPS 582

MIPS 582

MPEG 555

$\mu$ -закон 412

## R

RGB-кодирование 429

RISC-процессор 108

## S

SHARC 565

sinc-функция 66, 156, 243, 248

SRAM 592

S-область 642, 667

S-плоскость 642

## Z

Z-область 670

Z-преобразование 132, 175, 364, 378, 667

## A

автокорреляция 165

алгебраический метод восстановления 497

алгоритмы восстановления 18

алгоритмы сжатия 18

АЛУ 104, 568

амплитудная модуляция 64, 239, 254

амплитудная характеристика 160

амплитудно-частотная характеристика (АЧХ) 377

анализ 177

анализ Фурье 170

аналоговый сигнал 26, 57, 64

аналоговый фильтр 53, 66, 67

аналого-цифровое преобразование 53, 64, 76, 79, 89, 234

аналого-цифровой преобразователь (АЦП) 53

антиэлайзинговый фильтр 67, 76

апертура выборки 425, 473, 481

аппроксимация 237

арифметико-логическое устройство (АЛУ) 104, 568

арифметическое кодирование 540

архитектура гарвардская 564

архитектура фонннеймановская 563

Ассемблер 99, 576

аудиосистемы высокого качества 407, 408

АЦП 53, 58, 67, 68

АЧХ 302, 307, 377

## Б

бабочка 269

базилярная мембрана 401

базисные функции 180, 209

безнаковый целочисленный формат 89

белый шум 54, 168, 204, 257, 349

билинейное преобразование 691

бит-реверсная адресация 266, 272, 567

БИХ-фильтр 301

БИХ-фильтры специальные 315

боковые лепестки 206, 235

БПФ 129, 169, 177, 187, 215, 262, 265, 277, 355, 360

БПФ с прореживанием по времени 272

БПФ с прореживанием по частоте 271

быстрая свёртка 169, 337, 466

быстрое преобразование Фурье (БПФ) 129, 169, 177, 187, 215, 355

## В

векторное преобразование 622

вероятность 38

верхняя боковая полоса 63

весовые коэффициенты 149

взаимная корреляция 165

восстановление Фурье 500

временная область 28, 63, 77, 176, 191, 245, 263

время установления 305

встраиваемые системы 574

выборка-хранение 53, 64  
 выравнивание гистограммы 443  
 выравнивание освещённости 458  
 высота 404  
 вычислительная производительность 168  
 вычислительная сложность 336  
 вычислительные затраты 322, 324, 360, 369

**Г**

гамма-кривая 437  
 гарвардская архитектура 108  
 гармоника 210, 256  
 гауссиан 450  
 генератор адреса данных 567  
 генератор случайных чисел 47  
 гидролокация 22  
 гистограмма 34, 36  
 гомоморфная обработка сигналов 133, 419, 458  
 граббер 434  
 граничная характеристика 477  
 громкость 403, 404

**Д**

двухуровневое кодирование Рида–Соломона 410  
 действительная часть 177  
 декомпозиция 124, 134, 171, 177, 227  
 декомпозиция на основе сигналов с чётной и нечётной симметрией 128  
 декомпозиция с прореживанием 129  
 декомпозиция Фурье 129, 130, 134, 174, 180  
 дельта-кодирование 540  
 дельта-модулятор 81  
 дельта-модулятор с непрерывным изменением крутизны 83  
 детектор идеальный 480  
 детектор смазывающий 480  
 деформирование изображения 444  
 десиммация 80, 237  
 динамический диапазон 89, 394  
 динамический диапазон амплитудный 394  
 дискретизация 57, 67, 76, 79, 300  
 дискретизация 53  
 дискретизирующая последовательность 63  
 дискретная дельта-функция 364  
 дискретная производная 154  
 дискретное косинусное преобразование (ДКП) 175, 549  
 дискретное преобразование Фурье (ДПФ) 173, 229  
 дискретный интеграл 154  
 дискретный отсчёт 57, 64, 157  
 дискретный отсчёт, номер дискретного отсчёта 27  
 дискретный сигнал 26  
 дисперсия 47, 56

дифференциальные уравнения 642  
 дифференцирование 153  
 ДКП 549  
 дополнение нулями 145  
 дополнительный код 91  
 ДПФ 177, 201, 229, 243, 262, 302  
 ДПФ действительное 270  
 ДПФ комплексное 270, 630  
 ДПФ прямое 191  
 дуальность 191, 192, 237, 246

**Е**

единичная окружность 671  
 единичный импульс 134, 136, 151, 157, 245, 300, 364  
 единичный импульс двумерный 447

**З**

заворачивание спектра 59  
 заворачивание частот 76  
 закон Гаусса 164  
 знаковый бит 90  
 зона переходная 306  
 зона подавления 306, 336, 378

**И**

измерение 50  
 измеритель пространственного разрешения методом пар линий 475  
 изображение цифровое 422  
 импульсная декомпозиция 126, 134  
 импульсная характеристика 134, 136, 151, 156, 161, 210, 213, 353  
 импульсный портрет 61, 64  
 инверсия АЧХ 310, 334, 681  
 интеграл свёртки 285  
 интегральная функция распределения 45  
 интегральный профиль ФРТ 479  
 интегрирование 153  
 интегрированная среда разработки 596, 607  
 интерполяция 80, 237, 411  
 интерполяция билинейная 445  
 интерполяция субпиксельная 445  
 интерпретатор 101  
 искусственно добавление шума 57  
 искусственное зашумление с вычитанием 57  
 исполняемый код 99  
 истинное значение 50  
 исходный код 99

**К**

карта нулей и полюсов 658  
 каскадное соединение 121  
 квантование 26, 53  
 КИХ-фильтр 301  
 кодирование в частотной области 76, 226, 237, 303, 314, 320

кодирование во временной области 76, 225, 237, 303, 314, 320  
 кодирование длин серий 536  
 кодирование Хаффмана 538  
 колебания Гиббса 238, 254  
 колебательный процесс 73  
 компандирование 408, 412  
 компилятор 100, 606  
 комплексная плоскость 612  
 комплексное ДПФ 192  
 комплексное сопряжение 228  
 комплексно-сопряжённое 613  
 комплексные числа 610  
 компоненты декомпозиции (разложения) 124  
 конвейерное выполнение команд 108  
 контрастность 437  
 контроллер ввода/вывода 565  
 коррекция частотной характеристики 342  
 корреляция 165, 228, 270  
 коэффициент вариации 33, 51  
 коэффициенты Фурье 185  
 кривая Гаусса 43, 252, 320  
 круговая симметрия 129, 456  
 круговая частота приведённая 179  
 крутизна спада 377  
 крутизна спада частотной характеристики 71, 75  
 кэш-память 106, 565

**Л**

ЛАЧХ 302  
 линейная система 120, 134, 136, 160, 195  
 линейное предсказание 408, 541  
 ЛПК 415  
 ЛЧМ-сигнал 259  
 ЛЧМ-система 259

**М**

мантийса 91  
 математическая статистика 26  
 математический сопроцессор 104  
 машинный код 99  
 метод замещения комплексными числами 617  
 метод математической эквивалентности 617  
 метод обратных проекций 497  
 метод обратных проекций с фильтрацией 497  
 микроконтроллер 574  
 микропроцессор 574  
 мнимая часть 177  
 многопроцессорная обработка 586  
 многоскоростная обработка 66, 237  
 многоскоростная обработка сигналов 68, 80, 81  
 модулирующий сигнал 239  
 модуляционная передаточная функция 475  
 модуляция восемь—четырнадцать 410  
 модуль 192, 202, 221, 245  
 модуль комплексного числа 614  
 морфин 444

морфологическая обработка 487  
 мультиплексирование в частотной области 241

**Н**

наложение в частотной области 232  
 наложение во временной области 230  
 наложение спектров 59, 67, 76, 352  
 наложения 235, 250  
 неискажаемая передача гармонических сигналов 195  
 нейронные сети 315, 503  
 нелинейная система 120  
 непрерывный сигнал 26, 53, 57, 67  
 неравномерность 307  
 неравномерность АЧХ 377  
 неравномерность в полосе пропускания 73  
 несущее колебание 239  
 нижняя боковая полоса 63  
 нормальное распределение 49, 57  
 нули 378, 651, 673  
 НЧФ 210

**О**

область временная 207, 210, 213, 219, 229, 243, 355  
 область пространственная 422  
 область частотная 207, 210, 213, 219, 229, 243, 245, 355  
 обработка звуковых сигналов 19  
 обработка изображений 23, 422  
 обратное ДПФ 177, 192, 230, 264  
 обратное преобразование Фурье 191  
 обращение АЧХ 310  
 обращение свёртки 213, 343  
 объектный код 99  
 гибающая 239  
 однобитные АЦП, ЦАП 81  
 ОЗУ статическое 592  
 окно Бартлетта 328  
 окно Блэкмана 208, 322, 327  
 окно П-образное 209  
 окно прямоугольное 208, 328  
 окно Хеннинга 328  
 окно Хэмминга 206, 327  
 октава 406  
 опорный сигнал 165  
 основная частота 256  
 отношение сигнал/шум 33, 51, 168, 202, 483  
 оттенки серого 424  
 оценка 50  
 ошибка квантования 54  
 ошибка округления 31, 94, 200, 276, 324, 361, 375, 394  
 ошибки систематические 51

**П**

память двухпортовая 592  
 память кэш 106, 565

пара Фурье-преобразований 245  
 параллельное соединение систем с суммируемыми выходами 163  
 ПДП 566  
 первая разность 154  
 передаточная функция 656  
 передискретизация 68, 80, 81  
 перерегулирование 73, 305, 386, 394  
 переходная зона 329  
 переходная характеристика 73, 75  
 переходный процесс 322, 398  
 период дискретизации 56  
 периферия 593  
 пиксель 422  
 погрешность 50  
 полная обратимость 260  
 полоса заграждения 71  
 полоса пропускания 71, 75, 306, 336, 377  
 полоски Maxa 449  
 полюсы 378, 651, 673  
 полярная система координат 192, 202  
 порт связи 586, 593  
 порядок 91  
 порядок фильтра 329  
 последовательное соединение 121, 162  
 предсказание 50  
 преобразование Гильберта 175  
 преобразование Кархунена—Лоева 549  
 преобразование Лапласа 132, 175, 378, 642, 667  
 преобразование Фурье 171, 210, 219, 235, 239, 243, 319, 627, 644  
 преобразование Фурье действительное 175  
 преобразование Фурье дискретного времени 173, 212, 241, 249  
 преобразование Фурье дискретное 302  
 преобразование Фурье комплексное 175, 627  
 преобразование шкалы серого 441  
 преобразователь аналого-цифровой 377  
 преобразователь Гильберта 683  
 преобразователь цифро-аналоговый 377  
 принцип суперпозиции 124, 153, 157  
 программное обеспечение 88  
 программный автомат 563  
 процесс 33, 38  
 процесс нестационарный 35  
 процесс случайный 43  
 процесс стационарный 35  
 процессор цифровой обработки сигналов (ПЦОС) 108  
 процессоры с плавающей точкой 569  
 процессоры с фиксированной точкой 569  
 прямое ДПФ 177, 192  
 прямой доступ в память 566, 593  
 прямоугольная система координат 192  
 псевдо случайная последовательность 50  
 ПЦОС 108

## P

радиолокация 21  
 разрядность 53, 66  
 распределение вероятностей 34, 38  
 распределение нормальное, Гауссово 43  
 расширение 235  
 реальное время 353, 560  
 рекурсивные разностные уравнения 156  
 ряд Фурье 173, 293

**C**  
 свёртка 66, 134, 153, 159, 165, 168, 187, 188, 192, 210, 213, 228, 300, 315, 360  
 свёртка быстрая 355, 360  
 свёртка двумерная 322  
 свёртка изображений 447, 469  
 свёртка круговая 216  
 свёртка при выполнении условия сепарабельности 454  
 свёртка с секционированием 353  
 свёртка со стороны входа 138  
 свёртка со стороны выхода 138  
 свёртка циклическая 216, 356  
 свойство аддитивности 114, 117, 119, 120, 219, 281  
 свойство ассоциативности 161  
 свойство дистрибутивности 162  
 свойство инвариантности к сдвигу 114, 117, 120, 221  
 свойство коммутативности 121, 161  
 свойство линейности 219  
 свойство линейности статической характеристики 117, 120  
 свойство неискажаемой передачи гармонических сигналов 117, 120  
 свойство однородности 114, 117, 120, 219  
 связь 17  
 сдвиг фазы 61  
 сдвигатель 568  
 сейсморазведка 22  
 секционирование 353, 356, 375  
 сепарабельность 454  
 сжатие 235  
 сжатие без потерь 534  
 сжатие данных 534  
 сжатие с потерями 534  
 сжатие с преобразованием 535, 548  
 Си 576  
 сигма-дельта АЦП 85  
 сигнал 14, 26, 38, 53, 76, 112, 124, 201, 210  
 сигнал апериодический 173, 241  
 сигнал дискретный 134, 153, 173, 235, 250  
 сигнал непрерывный 112, 173, 235, 250  
 сигнал обусловленный 158  
 сигнал периодический 173  
 сигнал с линейной фазой 159  
 сигнал с нелинейной фазой 159

сигнал с нечётной симметрией 128, 232  
 сигнал с нулевой фазой 159  
 сигнал с чётной симметрией 128, 232  
 сигнал телевизионный 434  
 сигнал циклический 230  
 сигнал цифровой 112  
 симулятор 596, 608  
 синтез 124, 177, 254  
 синтез и распознавание речи 19, 413  
 синтетическая речь 400  
 система 112, 124, 210  
 система без памяти 118  
 система линейная 114, 117, 132, 171, 210, 599  
 система нелинейная 132  
 система непрерывная 112  
 система неустойчивая 651  
 система устойчивая 651  
 система цифровая 112  
 системы связи 407  
 системы сжатия речи 407  
 скелетонизация 488  
 СКО 29  
 скорость вычислений 98  
 скорость нарастания переходной характеристики 317  
 слепое выравнивание 343  
 слоговый фильтр 84  
 случайный процесс 54, 202  
 случайный сигнал 33  
 смешённый двоичный формат 89  
 собственные функции 171  
 согласованный фильтр 168  
 спектр 60, 63, 64, 202, 228, 241  
 спектральная плотность 185  
 справочная таблица 110, 275, 376, 379  
 среднее значение 28, 34, 43, 50  
 среднеквадратическое отклонение 29, 34, 43, 50, 252, 276  
 стартовый набор 592  
 статистика 33  
 статистическая вариация 33  
 статистическая флюктуация 33  
 статистический шум 33, 34  
 статическая характеристика 118  
 ступенчатая декомпозиция 128  
 суперпозиция 126  
 схема Саллена–Ки 662

**Т**

таблица соответствия 376  
 текущая сумма 154  
 тембр 404, 405, 406  
 теорема отсчётов 59, 67, 77, 80  
 теорема отсчётов Найквиста 59  
 теорема отсчётов Шеннона 59  
 теорема срезов Фурье 500  
 теория вероятностей 26

тестовые задачи 582  
 томография классическая 494  
 томография компьютерная 494  
 томография с позитронной эмиссией 502  
 точность 50, 54, 67, 80, 89, 94, 336

**У**

улитка 400  
 умножитель 568  
 уравнение анализа 190, 192  
 уравнение анализа-синтеза 241  
 уравнение Парсеваля 243  
 уравнение синтеза 182, 192  
 устойчивость 364

**Ф**

фаза 192, 202, 221, 245  
 фаза комплексного числа 614  
 фаза линейная 222  
 фаза нелинейная 222  
 фаза нулевая 228  
 фазовая характеристика 160, 227  
 фиксатор нулевого порядка 64  
 фильтр 355  
 фильтр аналоговый 392  
 фильтр антиэлайзинговый 204  
 фильтр Баттерворта 68, 75, 76, 377, 386, 663  
 фильтр Бесселя 68, 75, 373  
 фильтр биквадратный 662, 676  
 фильтр БИХ 315, 323, 364, 395  
 фильтр верхних частот 156  
 фильтр Винера 351  
 фильтр восстановления 68  
 фильтр временной обработки 314  
 фильтр ВЧ 678  
 фильтр высокочастотный 310  
 фильтр Гаусса 322  
 фильтр КИХ 315, 323, 373, 395  
 фильтр линейно-фазовый 305, 371  
 фильтр нерекурсивный 300  
 фильтр нижних частот 64, 156  
 фильтр низкочастотный (НЧФ) 202, 210, 310  
 фильтр низкочастотный идеальный 252  
 фильтр НЧ 319, 325, 677  
 фильтр однополюсный 365, 398  
 фильтр однородный 315, 316, 319, 350, 369, 398  
 фильтр однородный модифицированный 320  
 фильтр оконный 210, 252, 310, 315, 325, 329, 392, 395  
 фильтр полосовой 310, 369  
 фильтр режекторный 310, 369  
 фильтр режекторный узкополосный 624  
 фильтр рекурсивный 300, 323, 337, 363, 375, 672  
 фильтр рекурсивный однополюсный 315  
 фильтр с бесконечной импульсной характеристикой 301, 364  
 фильтр с конечной импульсной

характеристикой 301, 323  
 фильтр с максимально гладкой АЧХ 377  
 фильтр с максимально плоской  
     характеристикой 73  
 фильтр с многократной обработкой 320  
 фильтр сглаживающий 319  
 фильтр скользящего среднего 316  
 фильтр согласованный 351  
 фильтр узкополосный 369  
 фильтр цифровой 299, 338, 373, 386, 392  
 фильтр частотной обработки 314  
 фильтр Чебышева 68, 75, 76, 315, 377, 386, 392,  
     395, 664  
 фильтр Чебышева–Баттерворта 685  
 фильтр эллиптический 664  
 фильтрация 355  
 фильтрация двунаправленная 375  
 фильтры на переключаемых конденсаторах 70  
 фильтры с переключаемыми конденсаторами 54  
 фильтры специальные 314  
 формантные частоты 415  
 формат с плавающей точкой 89, 91, 109, 323, 375,  
     601  
 формат с разделением поля знака и поля  
     значения 90  
 формат с фиксированной точкой 89, 109, 323, 601  
 формат чисел с плавающей точкой 40  
 формула Эйлера 615  
 ФРЛ 477  
 ФРТ 447  
 функция оконная 340  
 функция плотности вероятности 39  
 функция плотности вероятности  
     колоколообразная 43  
 функция приподнятого косинуса 328  
 функция распределения 45  
 функция распределения вероятности 39, 504  
 функция рассеяния точки (ФРТ) 136, 447, 498  
 функция расширения линии (ФРЛ) 477  
 Фурье-анализ 132, 461  
 ФЧХ 371, 373  
 ФЧХ линейная 371, 394  
 ФЧХ нелинейная 371  
 ФЧХ с нулевой фазой 371

**X**

характеристика амплитудно-частотная 245, 302  
 характеристика импульсная 243, 252, 300, 317,  
     320, 323, 364, 378  
 характеристика переходная 300, 304, 320, 394  
 характеристика фазовая 600  
 характеристика фазо-частотная (ФЧХ) 245  
 характеристика частотная 243, 300, 338

**Ц**

ЦАП 80  
 центральная предельная теорема 47, 164, 320  
 центральный процессор 104  
 циклическая свёртка 234  
 циклический буфер 561  
 циклы с нулевыми издержками 579  
 цифро-аналоговое преобразование 53, 64, 90  
 цифро-аналоговый преобразователь (ЦАП) 57,  
     68, 79  
 цифровая обработка сигналов (ЦОС) 14  
 цифровое изображение 422  
 цифровой сигнал 26, 58, 67  
 цифровой сигнальный процессор (ЦСП) 108,  
     557, 592  
 цифровой фильтр 68, 156  
 цифровые сигналы 53  
 ЦОС 14  
 ЦСП 108, 557

**Ч**

частота дискретизации 53, 58, 64, 66, 80, 178, 222,  
     235, 309  
 частота Найквиста 60, 64, 66  
 частота среза 64, 71, 156, 307, 325, 329  
 частотная избирательность 307  
 частотная область 28, 63, 77, 177, 191  
 частотная характеристика 66, 71, 160, 210  
 частотный спектр 220, 230, 235, 237, 245

**III**

шаг выборки 425, 473, 481  
 ширина полосы частот 67  
 шкала серого 422  
 шум 26, 47, 54, 67, 317  
 шум 1/f 205  
 шум квантования 66, 571  
 шум округления 301, 387  
 шум собственный 394

**Э**

экспонента комплексная 616  
 эллиптический фильтр 73, 76  
 эмулятор 608  
 энергия сигнала 243  
 эффект Гиббса 171, 327

**Я**

ядро 136, 213  
 ядро фильтра 156, 300  
 язык программирования 88, 98, 109  
 язык программирования высокого уровня 89  
 яркость 437

## Ошибки и опечатки

1. Стр. 183. Третий абзац сверху. Следует читать:

«Обратите внимание, что в выражении (8.2) для массивов амплитуд используются обозначения  $\operatorname{Re} \bar{X}[k]$  и  $\operatorname{Im} \bar{X}[k]$ , а не  $\operatorname{Re} X[k]$  и  $\operatorname{Im} X[k]$ . Дело в том, что массивы амплитуд, участвующие в процедуре синтеза и обозначенные нами  $\operatorname{Re} \bar{X}[k]$  и  $\operatorname{Im} \bar{X}[k]$ , немного отличаются от частотного представления сигнала  $\operatorname{Re} X[k]$  и  $\operatorname{Im} X[k]$ .»

2. Стр. 187. Отсутствует Рис. 8.7:

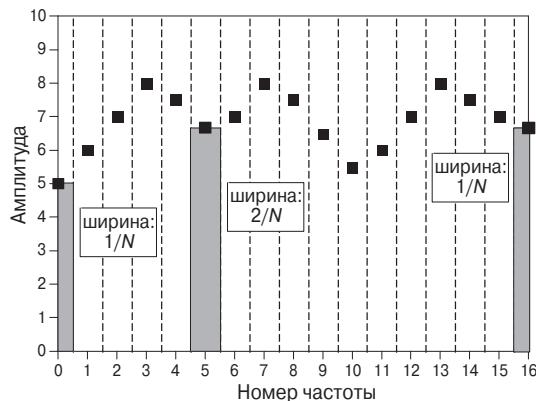


Рис. 8.7. Полосы частот, представляемые дискретными отсчетами в частотной области. Каждый дискретный отсчет в частотной области несет информацию о целой полосе частот шириной  $2/N$ , если выражать ее в долях от всей полосы дискретного сигнала  $N/2$ . Первый и последний отсчеты составляют исключение. Соответствующая им полоса частот имеет в два раза меньшую ширину  $1/N$ .

3. Стр. 188, заголовок 8.6.2. Вместо «Вычисление ДПФ с помощью свёртки» следует читать «Вычисление ДПФ с помощью корреляции».

4. Стр. 188, второй абзац снизу. Вместо «Использование свёртки...» следует читать «Использование корреляции...».

5. Стр. 188, первый абзац снизу. Вместо «С понятием свёртки...» следует читать «С понятием корреляции...».

6. Там же. Вместо «Напомним, что свёртка...» следует читать «Напомним, что корреляция...».

7. Стр. 189, первый абзац. Вместо «... алгоритм на основе свёртки...» следует читать «... алгоритм на основе корреляции...».

8. Стр. 190, последний абзац перед программой 8.2. Вместо «... сигналов с использованием свёртки...» следует читать «... сигналов с использованием корреляции...».

9. Стр. 193. Формула 8.6 должна выглядеть так:

$$\begin{aligned} Mag X[k] &= (Re X[k]^2 + Im X[k]^2)^{1/2}, \\ Phase X[k] &= \arctg(Im X[k] / Re X[k]). \end{aligned} \quad (8.6)$$

10. Стр. 228, третий абзац сверху. Вместо «спектр  $X^*[f]$  является комплексно-сопряжённым ему и описывается модулем  $Mag X[f]$  и фазой  $Phase X[f]$ » следует читать «спектр  $X^*[f]$  является комплексно-сопряжённым ему и описывается модулем  $Mag X[f]$  и фазой  $-Phase X[f]$ ». В этом же абзаце, вместо «комплексно-сопряжённый сигнал  $X^*[f]$  будет включать действительную  $Re X[f]$  и мнимую  $Im X[f]$  части» следует читать «комплексно-сопряжённый сигнал  $X^*[f]$  будет включать действительную  $Re X[f]$  и мнимую  $-Im X[f]$  части».

11. Стр. 561, подрисуночная подпись. Вместо **Рис. 28.1** следует читать **Рис. 28.2**.