

Таганрогский радиотехнический университет

Кафедра РТС

MatLab

Пакет DSP

Таганрог

2005

СОДЕРЖАНИЕ

5. Цифровая обработка сигналов (пакет Signal Processing Toolbox)	3
5.1. <i>Формирование типовых процессов</i>	5
5.1.1. Формирование отдельных импульсных процессов.....	5
5.1.2. Формирование колебаний	7
5.2. <i>Общие средства фильтрации. Формирование случайных процессов</i>	11
5.2.1. Общие основы линейной фильтрации	11
5.2.2. Формирование случайных процессов.....	16
5.3. <i>Спектральный и статистический анализ</i>	18
5.3.1 Основы спектрального (частотного) и статистического анализа процессов	18
5.3.2. Примеры спектрального анализа	22
5.3.3- Статистический анализ	28
5.4. <i>Проектирование фильтров</i>	30
5.4.1. Формы представления фильтров и их преобразования.....	30
5.4.2. Разработка аналоговых фильтров	33
5.4.3. Проектирование цифровых БИХ-фильтров.....	37
5.4.4. Проектирование КИХ-фильтров	40
5.5. <i>Графические и интерактивные средства</i>	47
5.5.1. Графические средства пакета Signal	47
5.5.2. Интерактивная оболочка SFTool.....	56

5. ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ (PAKET SIGNAL PROCESSING TOOLBOX)

Цифровая обработка сигналов традиционно включает в себя создание средств численного преобразования массива заданного (измеренного в дискретные моменты времени) процесса изменения некоторой непрерывной физической величины с целью извлечения из него полезной информации о другой физической величине, содержащейся в измеренном сигнале.

Общая схема образования измеряемого сигнала и процесса его преобразования в целях получения информации о величине, которая должна быть измерена, представлена на рис. 5.1.

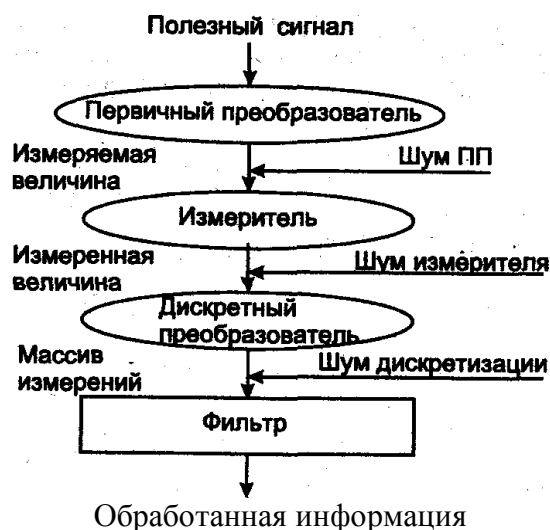


Рис. 5.1

Физическая величина, несущая в себе необходимую информацию, редко имеет такую физическую форму, в которой она может быть измерена непосредственно. Обычно она представляет лишь некоторую составляющую (сторону, часть, черту) некоторой другой физической величины, которая доступна для непосредственного измерения. Связь между этими двумя величинами обозначим введением звена, которое назовем первичным преобразователем (ПП). Обычно закон преобразования известен заранее, иначе восстановить информационную составляющую в дальнейшем было бы невозможно. Первичный преобразователь вносит зависимость Сигнала, который может быть измерен, от некоторых других физических величин. Вследствие этого выходная его величина содержит, кроме полезной информационной составляющей, другие, вредные составляющие или черты, искажающие полезную информацию. И хотя зависимость выхода ПП от этих других величин также известна, однако вследствие возможного неконтролируемого изменения последних со временем, часто трудно спрогнозировать их влияние на искажение полезной составляющей. Назовем вносимую ПП вредную составляющую шумом ПП.

Пусть образованная таким образом непосредственно измеряемая величина измеряется некоторым измерителем. Любой реальный измеритель вносит собственные искажения в измеряемую величину и дополнительные зависимости от некоторых других физических величин, не являющихся объектом измерения. Назовем эти искажения шумами измерителя. Не ограничивая общности, будем полагать, что выходной величиной измерителя является электрический сигнал (измеренная величина), который можно в дальнейшем довольно просто преобразовывать электрическими устройствами.

Для осуществления цифровой обработки измеренная величина должна быть преобразована в дискретную форму при помощи специального устройства, которое содержит экстраполятор и аналого-цифровой преобразователь (АЦП). Первый производит фиксацию отдельного текущего значения измеренной величины в отдельные моменты времени через определенный постоянный промежуток времени, называемый дискретом времени. Второй переводит это значение в цифровую форму, которая позволяет в дальнейшем осуществлять преобразования с помощью цифровых ЭВМ. Хотя оба устройства могут вносить при таких преобразованиях собственные искажения в выходной (дискретный) сигнал, однако ими обычно пренебрегают, так как в большинстве случаев эти дополнительные искажения значительно меньше шумов ПП и измерителя.

Для того чтобы на основе имеющегося дискретизированного сигнала получить полезный сигнал, нужно рассчитать и создать устройство (программу для ЭВМ), которое осуществляло бы такие преобразования входного дискретного, во времени сигнала, чтобы на его выходе искажения, внесенные шумами ПП и измерителя были минимизированы в некотором смысле. Это устройство называют фильтром.

Обычно создание (проектирование) фильтра является задачей неопределенной, которая конкретизируется лишь на основе предварительно полученных знаний о закономерности образования измеряемой величины (модели ПП), о модели образования измеренной величины из измеряемой (модели измерителя), о характеристиках изменения во времени вредных физических величин, влияющих на образование измеряемой и измеренной величин, и закономерностей их влияния на искажение полезной информации.

Так как модели ПП и измерителя могут быть весьма разнообразными, традиционно задачу фильтрации решают только для некоторых наиболее распространенных на практике, чаще всего, для линейных моделей.

В общем случае процесс создания фильтра включает следующие этапы:

- на основе априорной информации о моделях ПП и измерителя и о характеристиках шумов, а также о задачах, которые должен решать фильтр, выбирается некоторый тип фильтра из известных, с разработанной теорией проектирования;
- на основе конкретных числовых данных рассчитываются числовые характеристики выбранного типа фильтра (создается конкретный фильтр);
- проверяется эффективность выполнения разработанным фильтром поставленной перед ним задачи; для этого необходимо симитировать на ЭВМ дискретный сигнал, содержащий полезную (информационную) составляющую с наложенными на нее| предусмотренными шумами ПП и измерителя, пропустить его через построенный фильтр и сравнить полученный на выходе сигнал с известной (в данном случае) полезной его составляющей; разность между ними будет характеризовать погрешности измерения на выходе фильтра;
- так как в реальных условиях некоторые характеристики шумов могут отличаться от принятых при проектировании (создании фильтра), будут полезны испытания эффективности работы фильтра в условиях более приближенных к реальным, нежели принятые при проектировании.

Пакет Signal Processing Toolbox (в дальнейшем сокращенно Signal) предназначен для осуществления операций по трем последним из указанных этапов. Он позволяет проектировать (рассчитывать конкретные числовые характеристики) цифровые и аналоговые фильтры по требуемым амплитудно - и фазо - частотным их характеристикам, формировать последовательности типовых временных сигналов и обрабатывать их спроектированными фильтрами. В пакет входят процедуры, осуществляющие преобразования Фурье, Гильберта, а также статистический анализ. Пакет позволяет рассчитывать корреляционные функции, спектральную плотность мощности сигнала, оценивать параметры фильтров по измеренным отсчетам входной и выходной 1 последовательностей.

5.1. Формирование типовых процессов

5.1.1. Формирование отдельных импульсных процессов

В пакете Signal предусмотрено несколько процедур, образующих последовательности данных, представляющие некоторые одиночные импульсные процессы типовых форм. Процедура **rectpuls** обеспечивает формирование одиночного импульса прямоугольной формы. Обращение вида:

$$y = \text{rectpuls}(t, w)$$

позволяет образовать вектор y значений сигнала такого импульса единичной амплитуды, шириной w , центрированного относительно $t=0$ по заданному вектору t моментов времени. Если ширина импульса w не указана, ее значение по умолчанию принимается равным единице. На рис. 5.2 приведен результат образования процесса, состоящего из трех последовательных прямоугольных импульсов разной высоты и ширины, по такой последовательности команд:

```

» t = 0 : 0.01 : 10;
» y = 0.75*rectpuls(t-3, 2)+0.5*rectpuls(t-8, 0.4)...
+1.35*rectpuls(t-5, 0.8);
» plot(t,y), grid, set(gca,' FontName ', ' Arial Cyr ', ' FontSize ',16)
» title(' Пример применения процедуры RECTPULS ')
» xlabel(' Время ( с ) ')
» ylabel(' Выходной процесс Y(t) ')

```

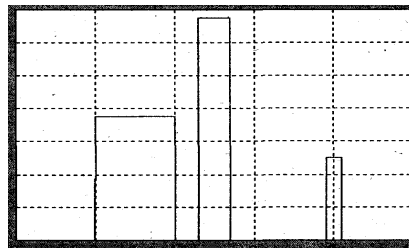


Рис. 5.2

Формирование импульса треугольной формы единичной амплитуды можно осуществить при помощи процедуры **tripuls**, обращение к которой имеет вид:

$$y = \text{tripuls}(t, w, s)$$

Аргументы y , t и w имеют тот же смысл. Аргумент s ($-1 < s < 1$) определяет наклон треугольника. Если $s=0$ или не указан, треугольный импульс имеет симметричную форму. Приведем пример (результат представлен на рис. 5.3):

```

» t = 0 : 0.01 : 10;
» y = 0.75*tripuls(t-1, 0.5)+0.5*tripuls(t-5, 0.5, -1)...
+1.35*tripuls(t-3, 0.8, 1);
» plot (t, y), grid

```

Для формирования импульса, являющегося синусоидой, модулированной функцией Гаусса, используется процедура **gauspuls**. Если обратиться к ней по форме:

$$y = \text{gauspuls}(t, fc, bw),$$

то она создает вектор значений указанного сигнала с единичной амплитудой, с синусоидой, изменяющейся с частотой f_c Гц, и с шириной bw полосы частот сигнала.

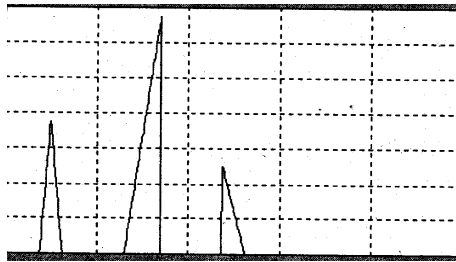


Рис. 5.3

В случае, когда последние два аргумента не указаны, они по умолчанию приобретают значения 1000 Гц и 0,5 соответственно. Приведем пример создания одиночного гауссового импульса (результат приведен на рис. 5.4):

```
» t = 0 : 0.01 : 10;
» y = 0.75*gauspuls(t-3, 1,0.5);
» plot(t, y), grid
```

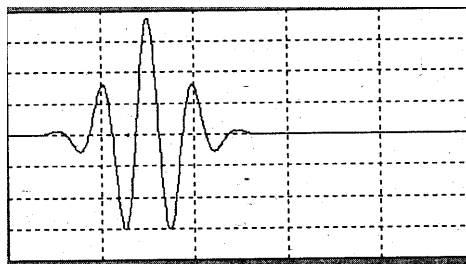


Рис. 5.4

Наконец, рассмотрим процедуру **sinc**, формирующую вектор значений функции $\text{sinc}(t)$, которая определяется формулами:

$$\text{sinc}(t) = \begin{cases} 1 & t = 0 \\ \frac{\sin(\pi \cdot t)}{\pi \cdot t} & t \neq 0 \end{cases}$$

Эта функция является обратным преобразованием Фурье прямоугольного импульса шириной 2π и высотой 1:

$$\text{sinc}(t) = \frac{1}{2\pi} \int_{-\pi}^{\pi} e^{j\omega t} d\omega$$

Приведем пример ее применения:

```
» t = 0 : 0.01 : 50;
» yl = 0.7*sinc(pi*(t-25)/5.);
» plot(t,yl), grid
```

Результат изображен на рис. 5.5.

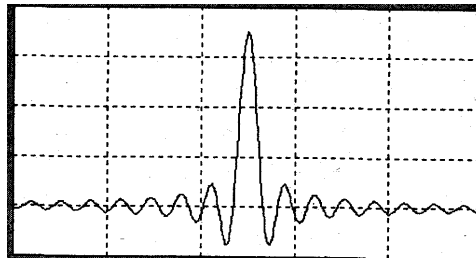


Рис. 5.5

5.1.2. Формирование колебаний

Формирование колебаний, состоящих из конечного числа гармонических составляющих (т.е. так называемых полигармонических колебаний), можно осуществить при помощи обычных процедур **sin(x)** и **cos(x)**. Рассмотрим пример (рис. 5.6):

```
» t = 0 : 0.01 : 50;
» yl = 0.7*sin(pi*t/5);
» plot(t,yl), grid
```

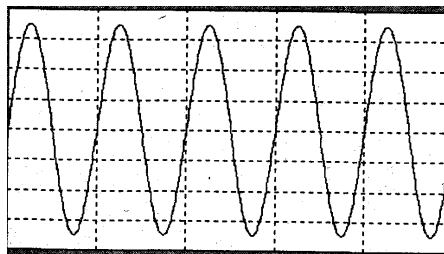


Рис. 5.6

Процесс, являющийся последовательностью прямоугольных импульсов с периодом 2π для заданной в векторе t последовательности отсчетов времени, генерируется при помощи процедуры **square**. Обращение к ней происходит по форме:

```
y = square(t,duty)
```

где аргумент **duty** определяет длительность положительной полуволны в процентах от периода волны. Например (результат приведен на рис. 5.7):

```
» y=0.7*square(pi*t/5, 40);
» plot(t,y), grid
```

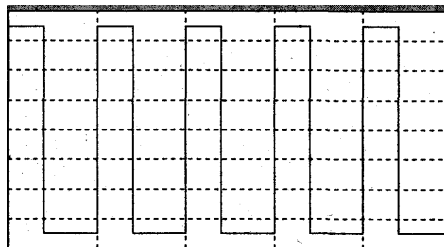


Рис. 5.7

Аналогично, генерирование пилообразных и треугольных колебаний можно осуществлять процедурой **sawtooth**. Если обратиться к ней так:

```
y = sawtooth(t,width),
```

то в векторе y формируются значения сигнала, представляющего собой пилообразные волны с периодом 2π в моменты времени, которые задаются вектором t . При этом

параметр **width** определяет часть периода, в которой сигнал увеличивается. Ниже приведен пример применения этой процедуры:

```
» y = 0.7*sawtooth(pi*t/5, 0.5);
» plot(t,y), grid
```

В результате получаем процесс, изображенный на рис. 5.8.

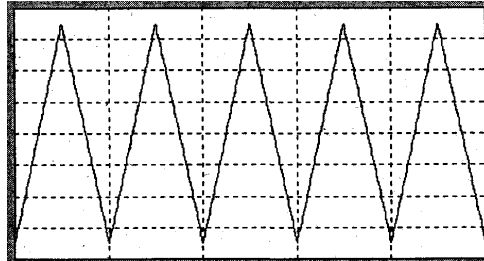


Рис. 5.8

Процедура **pulstran** позволяет формировать колебания, являющиеся последовательностью либо прямоугольных, либо треугольных, либо гауссовых импульсов. Обращение к ней имеет вид:

```
y = pulstran(t,d,'func',p1,p2,...)
```

Здесь **d** определяет вектор значений тех моментов времени, где должны быть центры соответствующих импульсов; параметр **func** определяет форму импульсов и может иметь одно из следующих значений: **rectpuls** (для прямоугольного импульса), **tripuls** (для треугольного импульса) и **gauspuls** (для гауссового импульса); параметры **p1**, **p2**, ... определяют необходимые параметры импульса в соответствии с формой обращения к процедуре, определяющей этот импульс.

Ниже приведены три примера применения процедуры **pulstran** для разных форм импульсов - составляющих.

- Для последовательности треугольных импульсов:

```
» t = 0 : 0.01 : 50
» d = [0 : 50/5 : 50] ;
» y = 0.7*pulstran(t, d,'tripuls',5);
» plot(t,y), grid
```

Результат представлен на рис. 5.9.

- Для последовательности прямоугольных импульсов:

```
» t = 0 : 0.01 : 50
» d = [0 : -50/5 : 50] ;
» y = 0.75*pulstran(t, d,'rectpuls',3);
» plot(t,y), grid
```

Результат приведен на рис. 5.10.

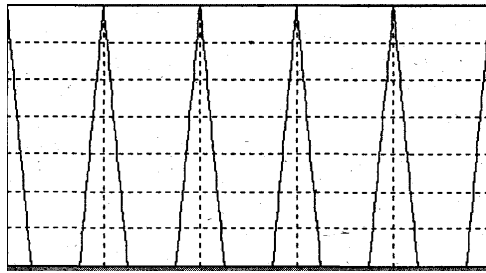


Рис. 5.9

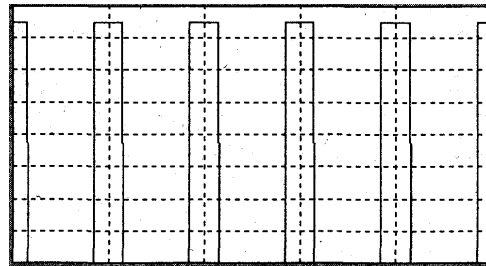


Рис. 5.10

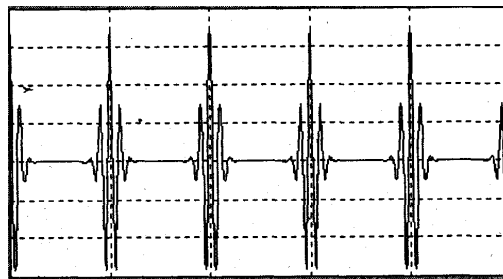


Рис. 5.11

- Для последовательности гауссовых импульсов:

```

» t = 0 : 0.01 : 50
» d = [0 : 50/5 : 50]';
» y = 0.7*pulstran(t, d,'gauspuls',1,0.5);
» plot(t,y), grid

```

Результат приведен на рис. 5.11.

Рассмотрим теперь процедуру **chirp**, формирующую косинусоиду, частота изменения которой линейно изменяется со временем. Общая форма обращения к этой процедуре такова:

$$y = \text{chirp}(t, F0, t1, F1)$$

где $F0$ – значение частоты в герцах при $t=0$, $t1$ – некоторое заданное значение момента времени; $F1$ – значение частоты (в герцах) изменения косинусоиды в момент времени $t1$. Если три последних аргумента не указаны, то по умолчанию им придаются такие значения: $F0=0$, $t1=1$, $F1=100$. Пример:

```

» t = 0 : 0.001 : 1;
» y = 0.75*chirp(t) ;
» plot(t,y), grid

```

Результат показан на рис. 5.12.

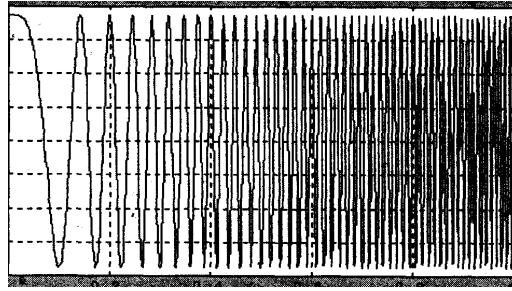


Рис. 5.12

Еще одна процедура – **diric** – формирует массив значений так называемой функции Дирихле, определяемой соотношениями:

$$\text{diric}(t) = \begin{cases} -1^{k(n-1)} & t = 2\pi \cdot k, \quad k = 0, \pm 1, \pm 2, \dots \\ \frac{\sin(nt/2)}{n \cdot \sin(t/2)} & \text{при других } t \end{cases}$$

Функция Дирихле является периодической. При нечетных n период равен 2π , при четных – 4π . Максимальное значение ее равно 1, минимальное – -1. Параметр n должен быть целым положительным числом. Обращение к функции имеет вид:

$$y = \text{diric}(t, n)$$

Ниже приведены операторы, которые иллюстрируют использование процедуры **diric** и выводят график функции Дирихле:

```
» t = 0 : 0.01 : 50;
» yl = 0.7*diric(pi*t/5, 3);
» plot(t, yl), grid
```

Результаты для $n = 3, 4$ и 5 представлены на рис. 5.13 – 5.15.

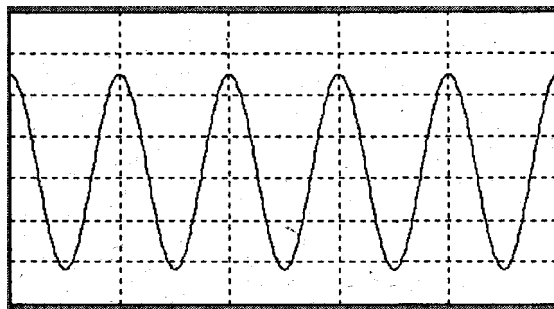


Рис. 5.13

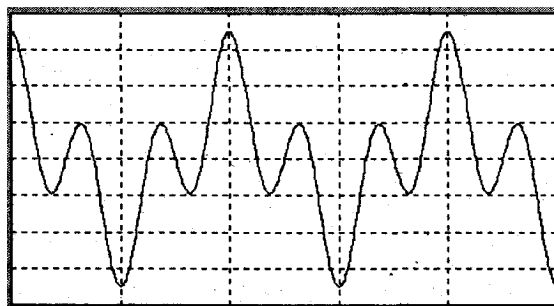


Рис: 5.14

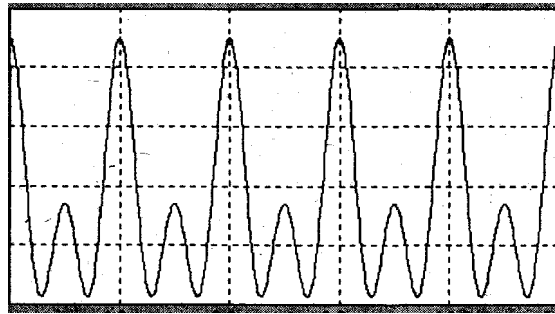


Рис. 5.15

5.2. Общие средства фильтрации. Формирование случайных процессов

5.2.1. Общие основы линейной фильтрации

Рассмотрим основы линейной фильтрации на примере линейного стационарного фильтра, который в непрерывном времени описывается дифференциальным уравнением второго порядка:

$$\ddot{y} + 2\xi \cdot \omega_0 \cdot \dot{y} + \omega_0^2 \cdot y = A \cdot x \quad (5.1)$$

где x – заданный процесс, подаваемый на вход этого фильтра второго порядка; y – процесс, получаемый на выходе фильтра; ω_0 – частота собственных колебаний фильтра, а ξ – относительный коэффициент затухания этого фильтра.

Передаточная функция фильтра, очевидно, имеет вид:

$$W(s) = \frac{y(s)}{x(s)} = \frac{A}{s^2 + 2\xi \cdot \omega_0 \cdot s + \omega_0^2} \quad (5.2)$$

Для контроля и графического представления передаточной функции любого линейного динамического звена, удобно использовать процедуру **freqs**. В общем случае обращение к ней имеет вид:

$$h = \text{freqs}(b,a,w)$$

При этом процедура создает вектор h комплексных значений частотной характеристики $W(j\omega)$ по передаточной функции $W(s)$ звена, заданной векторами коэффициентов ее числителя (b) и знаменателя (a), а также по заданному вектору w частоты ω . Если аргумент w не указан, процедура автоматически выбирает 200 отсчетов частоты, для которых вычисляется частотная характеристика.

Если не указана выходная величина, т.е. обращение имеет вид:

$$\text{freqs}(b,a,w)$$

процедура выводит в текущее графическое окно два графика – АЧХ и ФЧХ.

Приведем пример. Пусть для передаточной функции (5.2) выбраны такие значения параметров:

$$A = 1; \xi = 0.05; T_0 = 2\pi/\omega_0 = 1$$

Вычислим значения коэффициентов числителя и знаменателя и выведем графики АЧХ и ФЧХ:

```

» T0 = 1; dz = 0.05;
» om0=2*pi/T0; A = 1;
» al(1) = 1; al(2) = 2*dz*om0; al(3)= om0^2; bl(1) = A;
» freqs (b1,a1)

```

Результат показан на рис. 5.16.

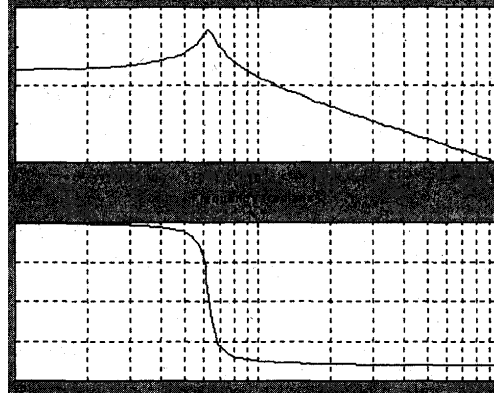


Рис. 5.16

Допустим, заданный процесс $x(t)$ представлен в виде отдельных его значений в дискретные моменты времени, которые разделены одинаковыми промежутками T_s времени (дискретом времени). Обозначим через $x(k)$ значение процесса в момент времени $t = k \cdot T_s$, где k – номер измерения с начала процесса.

Запишем уравнение (5.1) через конечные разности процессов x и y , учитывая, что конечно-разностным эквивалентом производной \dot{y} является конечная разность:

$$\frac{\Delta y(k)}{T_s} = \frac{y(k) - y(k-1)}{T_s}$$

а эквивалентом производной второго порядка \ddot{y} является конечная разность второго порядка:

$$\frac{\Delta^2 y(k)}{T_s^2} = \frac{\Delta y(k) - \Delta y(k-1)}{T_s^2} = \frac{y(k) - 2y(k-1) + y(k-2)}{T_s^2}$$

Тогда разностное уравнение:

$$(1 + 2\xi\omega_0 \cdot T_s + \omega_0^2 \cdot T_s^2) \cdot y(k) - 2(1 - \xi\omega_0 \cdot T_s) \cdot y(k-1) + y(k-2) = A \cdot T_s^2 \cdot x(k) \quad (5.3)$$

является дискретным аналогом дифференциального уравнения (5.1).

Применяя к полученному уравнению Z-преобразование, получим:

$$y(z) \cdot [a_0 + a_1 \cdot z^{-1} + a_2 \cdot z^{-2}] = A \cdot T_s^2 \cdot x(z), \text{ где} \quad (5.4)$$

$$a_0 = 1 + 2\xi\omega_0 \cdot T_s + \omega_0^2 \cdot T_s^2;$$

$$a_1 = -2(1 + \xi\omega_0 \cdot T_s); \quad (5.5)$$

$$a_2 = 1$$

Дискретная передаточная функция фильтра определяется из уравнения (5.4):

$$G(z) = \frac{y(z)}{x(z)} = \frac{A \cdot T_s^2}{a_0 + a_1 \cdot z^{-1} + a_2 \cdot z^{-2}} \quad (5.5)$$

Таким образом, цифровым аналогом ранее введенного колебательного звена является цифровой фильтр с коэффициентами числителя и знаменателя, рассчитанными по формулам (5.4) и (5.5):

```

» T0 = 1; dz = 0.05; Ts = 0.01;
» om0 = 2*pi/T0; A = 1; oms = om0*Ts;
» a(1) = 1+2*dz*oms+oms^2;
» a(2) = -2*(1+dz*oms) ;
» a(3) = 1;
» b(1) = A*Ts*Ts;

```

Чтобы получить частотную дискретную характеристику $G(t^{j\omega})$ по дискретной передаточной функции $G(z)$, заданной векторами значений ее числителя b и знаменателя a , удобно использовать процедуру **freqz**, обращение к которой аналогично обращению к процедуре **freqs**:

`freqz(b,a)`

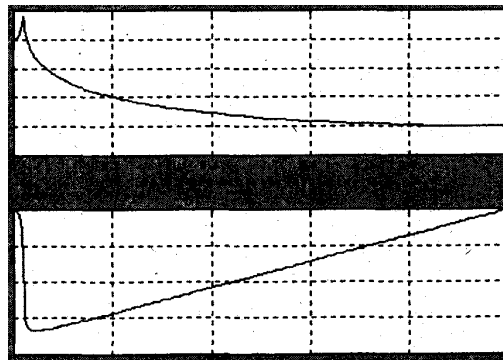


Рис. 5.17

В системе MatLAB фильтрация, т.е. преобразование заданного сигнала с помощью линейного фильтра, описываемого дискретной передаточной функцией вида:

$$G(z) = \frac{y(z)}{x(z)} = \frac{b_0 + b_1 \cdot z^{-1} + \dots + b_m \cdot z^{-m}}{a_0 + a_1 \cdot z^{-1} + \dots + a_n \cdot z^{-n}} \quad (5.7)$$

осуществляется процедурой **filter** следующим образом:

`y = filter(b,a,x)`

где x – заданный вектор значений входного сигнала; y – вектор значений выходного сигнала фильтра, получаемого вследствие фильтрации; b – вектор коэффициентов числителя дискретной передаточной функции (5.7) фильтра; a – вектор коэффициентов знаменателя этой функции.

В качестве примера рассмотрим такую задачу. Пусть требуется получить достаточно верную информацию о некотором полезном сигнале, имеющем синусоидальную форму с известным периодом $T1=1$ с и амплитудой $A1=0.75$. Сформируем этот сигнал как вектор его значений в дискретные моменты времени с дискретом $Ts=0.001$ с:

```

» Ts=0.001;
» t=0 : Ts : 20;
» A1=0.75; T1=1;
» Yp=A1*sin(2*pi*t/T1) ;
» plot(t(10002:end),Yp(10002:end)),grid,...
set(gca, 'FontName','ArialCyr','FontSize',16)
» title("Полезный" процесс');
» xlabel('Время (с)') ;
» ylabel('Yp(t) ')

```

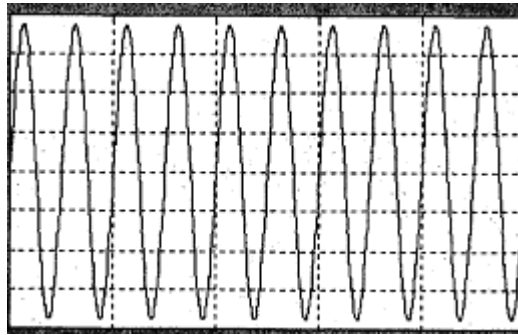


Рис. 5.18

Допустим, что вследствие прохождения через ПП (первичный преобразователь) к полезному сигналу добавился шум ПП виде более высокочастотной синусоиды с периодом $T_2=0.2$ с и амплитудой $A_2=5$, а в результате измерения к нему еще добавился белый гауссовый шум измерителя с интенсивностью $A_{ш}=5$. В результате созданся такой измеренный сигнал $x(t)$ (рис. 5.19):

```

» T2 = 0.2; A2 = 10; eps = pi/4;
» Ash = 5;
» x = A1.*sin(2*pi*t./T1)+A2.*sin(2*pi.*t./T2+eps)+...
Ash*randn (-1, length (t) ) ;
» plot(t(10002:end),x(10002:end)), grid, set(gca,'FontName',...
'ArialCyr','FontSize',16),
» title('Входной процесс ');
» xlabel('Время (с)');
» ylabel('X(t) ')

```

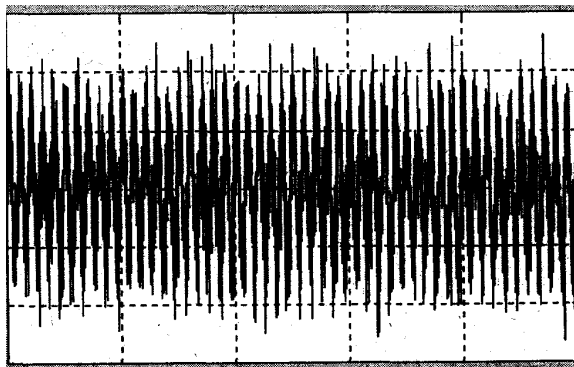


Рис. 5.19

Требуется так обработать измеренные данные x , чтобы восстановить по ним полезный процесс как можно точнее.

Так как частота полезного сигнала заранее известна, восстановление его можно осуществить при помощи резонансного фильтра отмеченного выше вида. При этом

необходимо создать такой фильтр, чтобы период его собственных колебаний T_f был равен периоду колебаний полезного сигнала ($T_f = T_1$). Чтобы после прохождения через такой фильтр амплитуда восстановленного сигнала совпадала с амплитудой полезного сигнала, нужно входной сигнал фильтра домножить на постоянную величину $2\xi\omega_0^2$ (поскольку при резонансе амплитуда выходного сигнала "уменьшается" именно во столько раз по сравнению с амплитудой входного сигнала). Сформируем фильтр, описанный выше:

```

» Tl = 1; Tf = Tl; dz = 0.05;
» om0 = 2*pi/Tf; A = 1; oms = om0*Ts;
» a(1) = 1+2*dz*oms+oms^2;
» a(2) = -2*(1+dz*oms) ;
» a(3) = 1;
» b(1) = fi*T's*Ts;

```

и пропустим сформированный процесс через него:

```

» y = filter(b,a,x)
» plot(t(10002:end),y(10002:end),t(10002:end), Yp(10002:end)),grid

```

В результате получим восстановленный процесс, изображенный на рис. 5.20. Для сравнения на этом же графике изображен восстанавливаемый процесс. Как видим, созданный фильтр достаточно хорошо восстанавливает полезный сигнал (сравните с рис. 5.19).

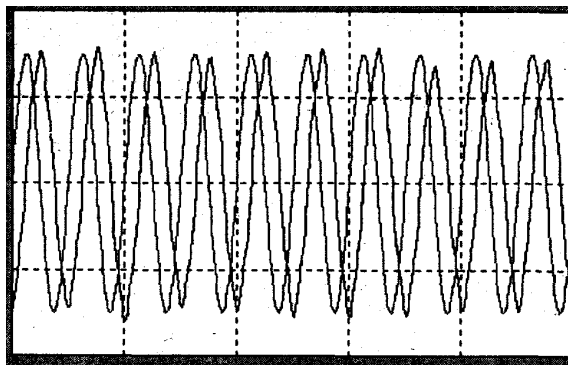


Рис. 5.20

Однако более точному восстановлению препятствуют два обстоятельства:

1. Восстановленный процесс устанавливается на выходе фильтра только спустя некоторое время вследствие нулевых начальных условий самого фильтра как динамического звена. Это продемонстрировано ниже, на рис. 5.21.

```

» y = filter (b,a,x)
» plot(t,y,t,Yp),grid

```

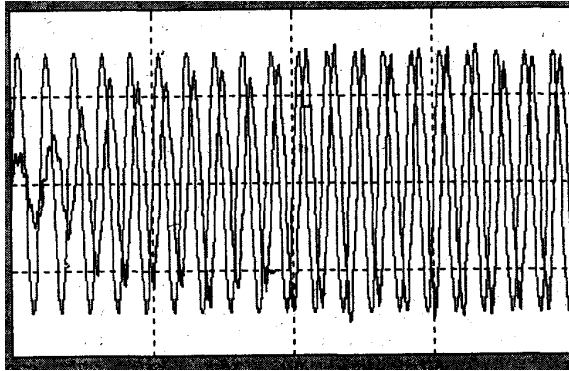


Рис. 5.21

2. В установившемся режиме наблюдается значительный сдвиг ($\pi/2$) фаз между восстанавливаемым и восстановленным процессами; это понятно, так как при резонансе сдвиг фаз между входным и выходным процессами достигает именно такой величины. Чтобы избежать фазовых искажений полезного сигнала при его восстановлении, можно воспользоваться процедурой двойной фильтрации – **filtfilt**. Обращение к ней имеет такую же форму, что и к процедуре **filter**. В отличие от последней процедура **filtfilt** осуществляет обработку вектора x в два приема: сначала в прямом, а затем в обратном направлении. Результат применения этой процедуры в рассматриваемом случае приведен на рис. 5.22.

```
» y = filtfilt(b,a,x)
» plot(t,y,t,Yp),grid
```

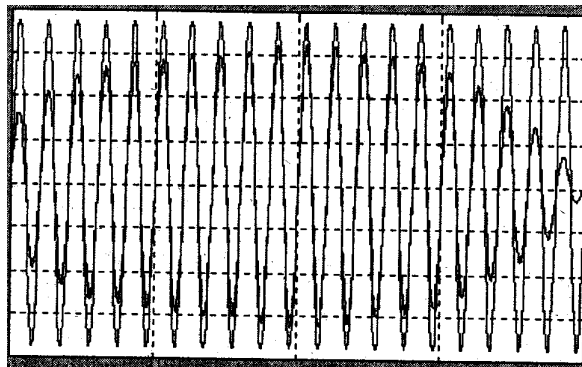


Рис. 5.22

5.2.2. Формирование случайных процессов

В соответствии с теорией сформировать случайный процесс с заданной корреляционной функцией можно, если сначала сформировать случайный процесс, являющийся нормально (по гауссовому закону) распределенным белым шумом, а затем "пропустить" его через некоторое динамическое звено (формирующий фильтр). На выходе получается нормально распределенный случайный процесс с корреляционной функцией, вид которой определяется типом формирующего фильтра как динамического звена.

Белый гауссов шум в MatLAB образуется при помощи процедуры **randn**. Для этого достаточно задать дискрет времени T_s , образовать с этим шагом массив (вектор) t моментов времени в нужном диапазоне, а затем сформировать по указанной процедуре вектор-столбец длиной, равной длине вектора t , например:

```
» Ts = 0.01;
» t = 0 : Ts : 20;
» xl = randn(1,length(t));
```

Построим график полученного процесса:


```
» plot(t,xl),grid
```

Процесс $x_1(t)$ с дискретом времени $T_s=0.01$ с представлен на рис. 5.23.

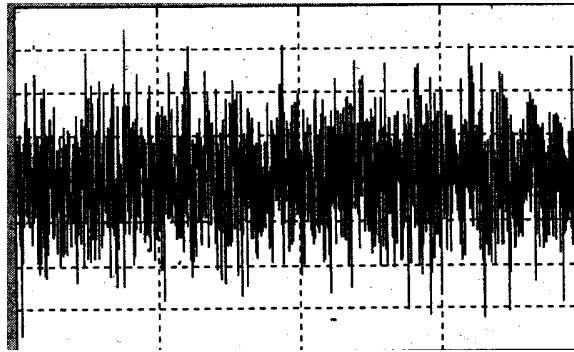


Рис. 5.23

Для другого значения дискрета времени ($T_s=0.001$ с), повторяя аналогичные операции, получим процесс $x_2(t)$ (рис. 5.24).

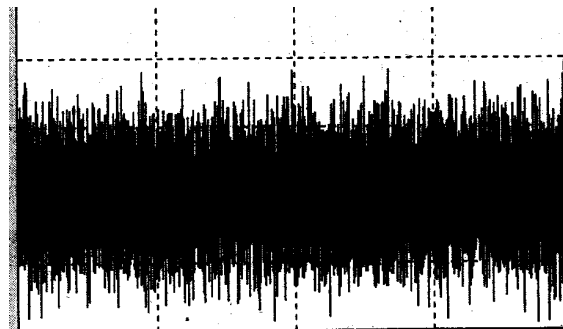


Рис. 5.24

Создадим дискретный фильтр второго порядка с частотой собственных колебаний $\omega_0 = 2\pi \cdot \text{rad/s} = 1\text{Гц}$ и относительным коэффициентом затухания $\xi = 0,05$ по формулам (5.5) коэффициентов:

```
» om0 = 2*pi; dz = 0.05; A = 1; oms = om0*Ts;
» a(1) = 1+2*dz*oms+oms^2;
» a(2) = -2*(1+dz*oms) ;
» a(3) = 1;
» b(1) = A*2*dz*oms^2;
```

Пропустим образованный процесс $x_1(t)$ через созданный фильтр:

```
» y1 = filter(b,a,xl);
```

Построим график процесса $y_1(t)$ на выходе фильтра:

```
» plot(t,y1),grid
```

Результат представлен на рис. 5.25.

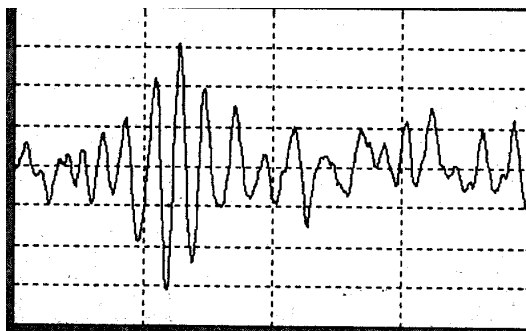


Рис. 5.25

Аналогичные операции произведем с процессом $x_2(t)$. В результате получим процесс $y_2(t)$, приведенный на рис. 5.26.

```

» Ts = 0.001;
» om0 = 2*pi; dz = 0.05; A = 1; oms = om0*Ts;
» a(1) = 1+2*dz*oms+oms^2;
» a(2) = -2*(1+dz*oms) ;
» a(3) = 1;
» b(1) = A*2*dz*oms^2;
» yl = filter(b,a,x2); t = 0 : Ts : 20;
» plot(t,yl),grid

```

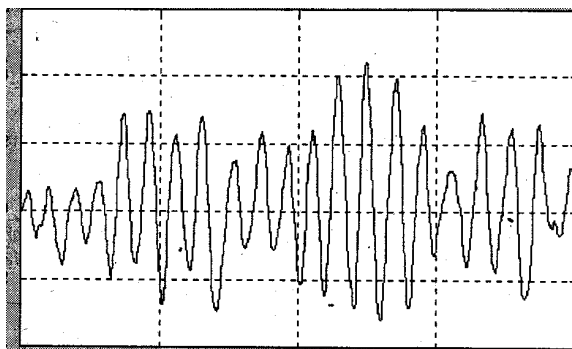


Рис. 5.26

Как видим, на выходе формирующего фильтра действительно образуется случайный колебательный процесс с преобладающей частотой 1 Гц.

5.3. Спектральный и статистический анализ

5.3.1 Основы спектрального (частотного) и статистического анализа процессов

Основная задача спектрального анализа сигналов – выявление гармонического спектра этих сигналов, т.е. определение частот гармонических составляющих сигнала (выявление частотного спектра), амплитуд этих гармонических составляющих (амплитудного спектра) и их начальных фаз (фазового спектра).

В основе спектрального анализа лежит теория Фурье о возможности разложения любого периодического процесса с периодом $T = \frac{2\pi}{\omega} = \frac{1}{f}$ (где ω – круговая частота

периодического процесса, а f – его частота в герцах) в бесконечную, но счетную сумму отдельных гармонических составляющих.

Напомним некоторые положения спектрального анализа. Прежде всего, любой периодический процесс с периодом T может быть представлен в виде так называемого комплексного ряда Фурье:

$$x(t) = \sum_{m=-\infty}^{+\infty} X^*(m) \cdot e^{j(2\pi \cdot m \cdot f) \cdot t} = \sum_{m=-\infty}^{+\infty} X^*(m) \cdot e^{j(m \cdot \omega) \cdot t} \quad (5.8)$$

причем комплексные числа $X^*(m)$, которые называют комплексными амплитудами гармонических составляющих, вычисляются по формулам:

$$X^*(m) = \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) \cdot e^{-j(2\pi \cdot m \cdot f) \cdot t} dt = \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) \cdot e^{-j(m \cdot \omega) \cdot t} dt \quad (5.9)$$

Таким образом, частотный спектр периодического колебания состоит из частот, кратных основной (базовой) частоте f , т.е. частот:

$$fm = m \cdot f (m = 0, 1, 2, \dots) \quad (5.10)$$

Действительные и мнимые части комплексных амплитуд $X^*(m)$ образуют соответственно действительный и мнимый спектры периодического колебания. Если комплексную амплитуду (5.9) представить в экспоненциальной форме:

$$X^*(m) = \frac{a_m}{2} \cdot e^{j \cdot \varphi_m} \quad (5.11)$$

то величина a_m будет представлять собой амплитуду гармонической составляющей с частотой $f_m = m \cdot f$, а φ_m – начальную фазу этой гармоники, имеющей форму косинусоиды, т.е. исходный процесс можно также записать в виде:

$$x(t) = a_0 + \sum_{m=-\infty}^{+\infty} a_m \cdot \cos(2\pi \cdot m \cdot f \cdot t + \varphi_m) \quad (5.12)$$

который, собственно, и называют рядом Фурье.

Для действительных процессов справедливы следующие соотношения:

$$\operatorname{Re}\{X(-m)\} = \operatorname{Re}\{X(m)\}; \operatorname{Im}\{X(-m)\} = -\operatorname{Im}\{X(m)\} \quad (5.13)$$

т.е. действительная часть спектра является четной функцией частоты, а мнимая часть спектра – нечетной функцией частоты.

Разложения (5.12) и (5.8) позволяют рассматривать совокупность комплексных амплитуд (5.9) как изображение периодического процесса в частотной области. Желание распространить такой подход на произвольные процессы, в том числе и непериодические, привели к необходимости ввода понятия Фурье - изображения в соответствии со следующим выражением:

$$X(f) = \int_{-\infty}^{+\infty} x(t) \cdot e^{-j(2\pi \cdot f) \cdot t} dt \quad (5.14)$$

Этот интеграл, несмотря на его внешнее сходство с выражением (5.9) для комплексных коэффициентов ряда Фурье, довольно существенно отличается от них.

Во-первых, в то время как физическая размерность комплексной амплитуды совпадает с размерностью самой физической величины $x(t)$, размерность Фурье - изображения равна размерности $x(t)$, умноженной на размерность времени.

Во-вторых, интеграл (5.14) существует (является сходящимся к конечной величине) только для так называемых "двусторонне затухающих" процессов (т.е. таких, которые уменьшаются до нуля как при $t \rightarrow +\infty$, так и при $t \rightarrow -\infty$). Иначе говоря, его нельзя применять к так называемым "стационарным" колебаниям.

Обратное преобразование Фурье-изображения в исходный процесс $x(t)$ в этом случае определяется интегралом:

$$x(t) = \int_{-\infty}^{+\infty} X(f) \cdot e^{-j(2\pi \cdot f) \cdot t} df \quad (5.15)$$

который представляет собой некоторый аналог комплексного ряда Фурье (5.1).

Указанное серьезное противоречие несколько сглаживается при численных расчетах, так как в этом случае можно иметь дело только с процессами ограниченной длительности, причем сам процесс в заданном диапазоне времени должен быть задан своими значениями в ограниченном числе точек.

В этом случае интегрирование заменяется суммированием, и вместо вычисления интеграла (5.14) ограничиваются вычислением суммы:

$$X[(k-1) \cdot \Delta f] = \Delta t \cdot \sum_{m=-\infty}^n x[(m-1) \cdot \Delta t] \cdot e^{-j2\pi \cdot (k-1) \cdot (m-1) \cdot \Delta f \cdot \Delta t} \quad (5.16)$$

Тут по сравнению с интегралом (5.14) осуществлены такие замены:

- непрерывный интеграл приближенно заменен ограниченной суммой площадей прямоугольников, одна из сторон которых равна дискрету по времени Δt , с которым представлены значения процесса, а вторая – мгновенному значению процесса в соответствующий момент времени;
- непрерывное время t заменено дискретными его значениями $(m-1) \cdot \Delta t$, где m – номер, точки от начала процесса;
- непрерывные значения частоты f заменены дискретными ее значениями $(k-1) \cdot \Delta f$, где k – номер значения частоты, а дискрет частоты равен $\Delta f = 1/T$, где, в свою очередь, T – промежуток времени, на котором задан процесс;
- дифференциал dt заменен ограниченным приращением времени Δt .

Если обозначить дискрет времени Δt через T_s , ввести обозначения:

$$x(m) = x[(m-1) \cdot \Delta t]; X(k) = X[(k-1) \cdot \Delta f]$$

а также учесть то, что число точек, в которых задан процесс, равно:

$$n = \frac{T}{\Delta t} = \frac{T}{T_s} = \frac{T}{\Delta f \cdot \Delta t} \quad (5.17)$$

то соотношение (5.15) можно представить в более удобной форме:

$$X(k) = T_s \cdot \sum_{m=1}^n x(m) \cdot e^{-j(2\pi/n) \cdot (k-1) \cdot (m-1)} \quad (5.18)$$

Как было отмечено в разд. 1.4.5 (формулы (2) и (3)), процедуры MatLAB **fft** и **ifft** осуществляют вычисления в соответствии с формулами:

$$y(k) = \sum_{m=1}^n x(m) \cdot e^{-j \cdot 2\pi \cdot (m-1) \cdot (k-1) / n} \quad (5.19)$$

$$x(m) = \frac{1}{n} \cdot \sum_{k=1}^n y(k) \cdot e^{j \cdot 2\pi \cdot (m-1) \cdot (k-1) / n} \quad (5.20)$$

соответственно. Сравнивая (5.18) с (5.19), можно сделать вывод, что процедура **fft** находит дискретное Фурье-изображение заданного дискретного во времени процесса $x(t)$, поделенное на дискрет времени:

$$y(k) = \frac{X(k)}{T_s} \quad (5.21)$$

Осуществляя аналогичную операцию дискретизации соотношения (5.9) для комплексной амплитуды $X^*(k)$, получим:

$$\begin{aligned}
X^*(k) &= \frac{Ts}{T} \cdot \sum_{m=-\infty}^{+\infty} x(m) \cdot e^{-j(2\pi/n) \cdot (k-1) \cdot (m-1)} = \\
&= \frac{1}{n} \cdot \sum_{m=-\infty}^{+\infty} x(m) \cdot e^{-j(2\pi/n) \cdot (k-1) \cdot (m-1)} = \frac{y(k)}{n}
\end{aligned} \tag{5.22}$$

Из этого следует, что комплексный спектр разложения стационарного процесса равен поделенному на число измерений результату применения процедуры **fft** к заданному вектору измеренного процесса.

Если же принять во внимание, что для большинства стационарных колебательных процессов именно частотный, амплитудный и фазовый спектры не зависят от длительности T конкретной реализации и выбранного дискрета времени T_s , то надо также сделать вывод, что для спектрального анализа стационарных процессов наиболее целесообразно применять процедуру **fft**, результат которой делить затем на число точек измерений.

Перейдем к определению спектральной плотности мощности (СПМ), или, сокращенно, спектральной плотности (СП). Это понятие в теории определяется как Фурье - изображение так называемой корреляционной функции $R_{12}(\tau)$ и применяется, в основном, для двух одновременно протекающих стационарных процессов $x_1(t)$ и $x_2(t)$. Взаимная корреляционная функция (ВКФ) двух таких процессов определяется соотношением:

$$R_{12}(\tau) = \lim_{T \rightarrow \infty} \frac{1}{T} \cdot \int_{-\frac{T}{2}}^{\frac{T}{2}} x_1(t) \cdot x_2(t + \tau) \cdot dt \tag{5.23}$$

т.е. ВКФ является средним во времени значением произведения первой функции на сдвинутую относительно нее на время задержки τ вторую функцию.

Итак, взаимная спектральная плотность (ВСП) двух стационарных процессов может быть определена следующим образом:

$$S_{12}(f) = \int_{-\infty}^{+\infty} R_{12}(\tau) \cdot e^{-j(2\pi \cdot f) \cdot \tau} d\tau \tag{5.24}$$

При числовых расчетах, когда оба процесса $x_1(t)$ и $x_2(t)$ заданы на определенном ограниченном промежутке T времени своими значениями в некоторых точках, разделенных дискретом времени T_s , формулу (5.23) можно трансформировать в такую:

$$R_{12}(l) = \frac{1}{n-1} \sum_{m=1}^{n-1} x_1(m) \cdot x_2(m+l-1), \quad (l=1,2,\dots,n/2) \tag{5.25}$$

или в несколько более простое соотношение:

$$R_{12}(l) = \frac{2}{n} \sum_{m=1}^{n/2} x_1(m) \cdot x_2(m+l-1), \quad (l=1,2,\dots,n/2) \tag{5.26}$$

а вместо (5.24) использовать

$$S_{12}(k) = Ts \cdot \sum_{l=1}^{n/2} R_{12}(l) \cdot e^{-j \cdot (2\pi/n) \cdot (k-1) \cdot (l-1)}, \quad (k=1,2,\dots,n/2) \tag{5.27}$$

Если теперь подставить выражение (5.26) в (5.27) и изменить в нем порядок суммирования, то можно прийти к такому соотношению между ВСП и результатами преобразований процедурой **fft** заданных измеренных значений процессов:

$$S_{12}(k) = Ts \cdot \left\{ \frac{2}{n} y_2(k) \right\} \cdot \bar{y}_1(k), \quad (k=1,2,\dots,n/2) \tag{5.28}$$

где черта сверху означает комплексное сопряжение соответствующей величины.

С учетом (5.21) и (5.22) выражение (5.28) можно представить также в виде:

$$S_{12}(k) = X_2^*(k) \cdot \bar{X}_1(k) \tag{5.29}$$

Из этого следует, что взаимная спектральная плотность двух процессов при любом значении частоты равна произведению значения комплексного спектра второго процесса на комплексно - сопряженное значение Фурье - изображения первого процесса на той же частоте.

Формулы (5.21), (5.22) и (5.28) являются основой для вычислений в системе **MatLAB** соответственно Фурье - изображения процесса, его комплексного спектра и взаимной спектральной плотности двух процессов.

5.3.2. Примеры спектрального анализа

Чтобы применить процедуру **fft** как преобразование процесса, представленного во временной области, в его представление в частотной области, необходимо, как было отмечено в разделе 1.4.5, сделать следующее:

- по заданному значению дискрета времени T_s рассчитать величину F_{\max} диапазона, частот (в герцах) по формуле:

$$F_{\max} = 1/T_s \quad (5.30)$$

- по заданной длительности процесса T рассчитать дискрет частоты df по формуле:

$$df = 1/T \quad (5.31)$$

- по вычисленным данным сформировать вектор значений частот, в которых будет вычислено Фурье-изображение. Последнее проще (но не наиболее правильно) сделать таким образом:

$$f1 = 0 : df : F_{\max} \quad (5.32)$$

В результате применения процедуры **fft** будет получено представление процесса в частотной области. Обратная процедура **ifft**, если ее применить к результатам первого преобразования, дает возможность восстановить исходный процесс во временной области. Однако процедура **fft** не дает непосредственно Фурье - изображения процесса. Чтобы получить Фурье - изображение, необходимо выполнить следующие действия (разд. 1.4.5):

- к результатам действия процедуры **fft** применить процедуру **fftshift**, которая переставляет местами первую и вторую половины полученного вектора;
- перестроить вектор частот по алгоритму:

$$f = -F_{\max}/2 : df : F_{\max}/2 \quad (5.33)$$

Приведем примеры.

Фурье-изображение прямоугольного импульса

Сформируем процесс, состоящий из одиночного прямоугольного импульса. Зададим дискрет времени $T_s = 0.01$ с, длительность процесса $T = 100$ с, амплитуду импульса $A = 0.75$ и его ширину $w = 0.5$ с:

```
» Ts = 0.01; T = 100; A = 0.75; w = 0.5;
» t = 0 : Ts : T;
» y = A*rectpuls(t, w) ;
» plot(t(1:100), y(1:100) )
```

Результат показан на рис. 5.27.

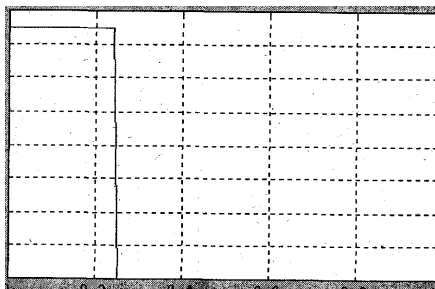


Рис: 5.27

Применим к вектору y процедуру **fft** и построим график зависимости модуля результата от частоты. Графики в частотной области удобнее выводить при помощи процедуры **stem** (рис. 5.28):

```
» x = fft(y);
» df = 1/T; Fmax = 1/Ts;
» f = 0 : df : Fmax;
» a = abs(x) ;
» stem(f,a), grid
```

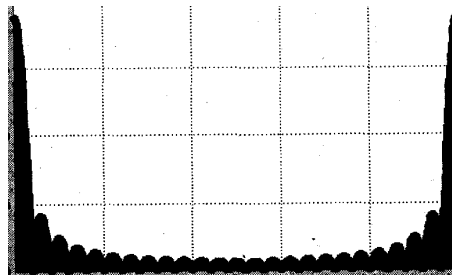


Рис. 5.28

Теперь построим график модуля Фурье-изображения процесса:

```
» xp = fftshift(x) ;
» fl = -Fmax/2 : df : Fmax/2;
» a = abs(xp) ;
» stem(fl,a), grid
```

Получим результат, приведенный на рис. 5.29.

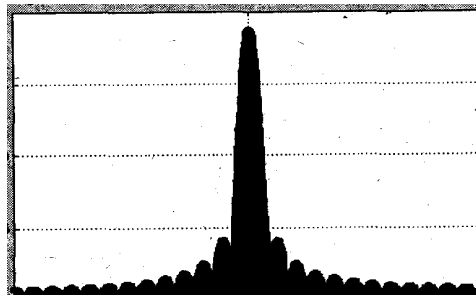


Рис. 5.29

В заключение построим графики действительной и мнимой частей Фурье - изображения прямоугольного импульса:

```
» dch = real(xp); mch = imag(xp);
» plot(fl, dch, fl, mch), grid
```

Полученные графики представлены на рис. 5.30.

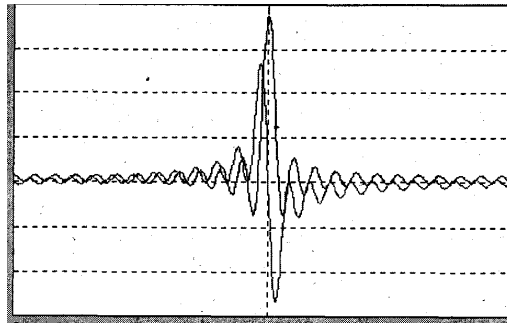


Рис. 5.30

Фурье-изображение полигармонического процесса

Рассмотрим пример трехчастотных гармонических колебаний – с частотой $1/\pi$, 1 и 3 Гц и амплитудами соответственно 0.6, 0.3 и 0.7:

$$y(t) = 0.6 \cdot \cos(2t) + 0.3 \cdot \sin(2\pi \cdot t) + 0.7 \cdot \cos(6\pi \cdot t + \pi/4)$$

Найдем Фурье-изображение этого процесса и выведем графики самого процесса, модуля его Фурье-изображения, а также действительную и мнимую части:

```

» Ts = 0.01; T = 100; ;
» t = 0 : Ts : T;
» Y = 0.6*cos(2*t)+0.3*sinh(2*pi*t)+0.7*cos(6*pi*t+pi/4);
» plot(t,Y), grid

```

График процесса показан на рис. 5.31.

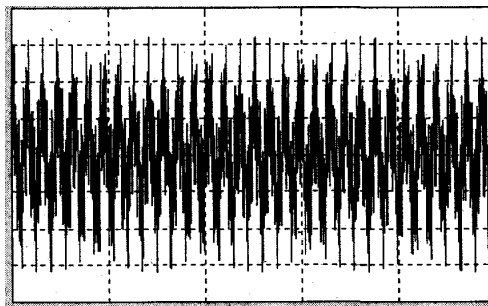


Рис. 5.31

Находим модуль Фурье-изображения этого процесса:

```

» df = 1/T; Fmax = 1/Ts; dovg=length(t);
» f = -Fmax/2 : df : Fmax/2;
» X = fft(Y) ; Xp = fftshift(X) ;
» A = abs (Xp) ;
» sl = dovg/2 - 400; s2 = dovg/2 + 400;
» stem(f(sl:s2),A(sl:s2)), grid,

```

Результат представлен на рис. 5.32.

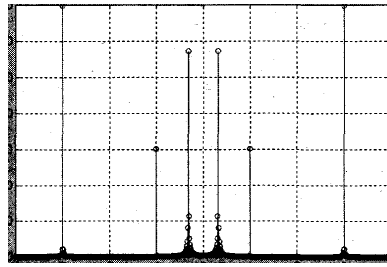


Рис. 5.32

Если изменить дискрет времени на $T_s=0.02$, получим результат, изображенный на рис. 5.33.

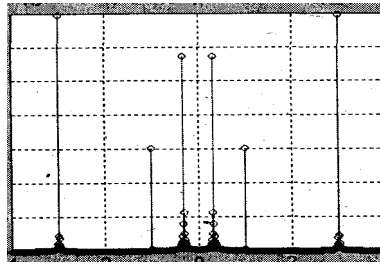


Рис. 5.33

Как видно, результат Фурье-преобразования в значительной степени зависит от величины дискрета времени и мало что говорит об амплитудах гармонических составляющих. Это обусловлено различием между определениями Фурье-изображения, и комплексного спектра. Поэтому для незатухающих (установившихся, стационарных) колебаний любого вида намного удобнее находить не Фурье-изображение, а его величину, деленную на число точек в реализации. В предыдущей части программы это эквивалентно замене оператора $X = \text{fft}(Y)$ на $X = \text{fft}(Y)/\text{dovg}$, где dovg – длина вектора t . В результате получается комплексный спектр (рис. 5.34), полностью соответствующий коэффициентам комплексного ряда Фурье.

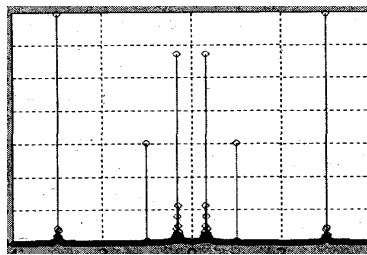


Рис. 5.34

Выделим действительную и мнимую части комплексного спектра:

```

» dch = real(Xp); mch = imag(Xp);
» sl = dovg/2 - 400; s2 = dovg/2 + 400;
» subplot(2,1,1)
» plot(f(sl:s2), dch(sl:s2)) , grid,
» subplot(2,1,2)
» plot(f(sl:s2), mch(sl:s2)), grid

```

По полученным графикам (рис. 5.35) можно судить не только о частотах и амплитудах, но и о начальных фазах отдельных гармонических составляющих.

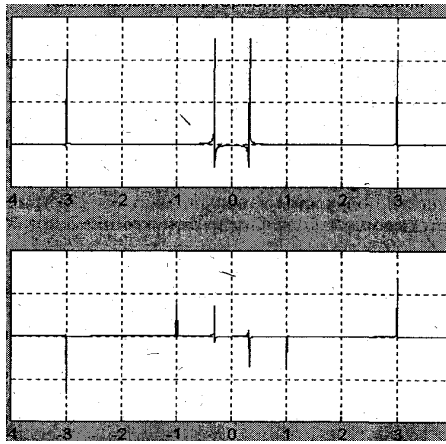


Рис. 5.35

Фурье-изображение случайного процесса

В заключение рассмотрим Фурье-преобразование случайного стационарного процесса, сформированного ранее (рис. 5.25). Сначала сформируем процесс в виде белого гауссового шума (рис. 5.36) с шагом во времени 0.01 и длительностью 100 с:

```

» Ts=0.01'; T = 100; % Задание параметров процесса
» t=0 : Ts : T;
» xl = randn(I,length(t)); % Формирование белого шума
» % Построение графика белого шума
» plot(t,xl), grid

```

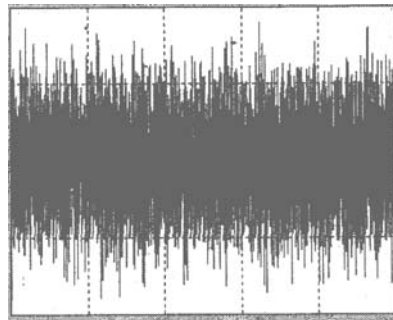


Рис. 5.36

Теперь создадим формирующий фильтр, "пропустим" через него белый шум (рис. 5.37):

```

» % Расчет параметров формирующего фильтра
» om0 = 2*pi; dz = 0.05; A = 1; oms = om0*Ts;
» a(1) = 1+2*dz*oms + oms^2;
» a(2) = -2*(1+dz*oms) ;
» a(3) = 1;
» b(1) = A*2*dz*oms^2;
» % Формирование "профильтрованного" процесса
» yl = filter(b,a,xl) ;
» % Построение графика процесса
» plot(t,yl),grid

```

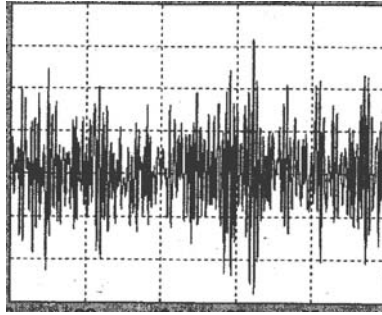


Рис. 5.37

Вычислим Фурье-изображение (ФИ) для процесса-шума с учетом замечания, сделанного для установившихся процессов, и построим на рис. 5.38 графики модуля ФИ и спектральной плотности мощности (СПМ):

```

» % Формирование массива частот
» df = 1/T; Fmax = 1/Ts;
» f = - Fmax/2 : df : Fmax/2;
» dovg = length(f);
» % Расчет скорректированных массивов Фурье-изображений
» Ful = fft(xl)/dovg; Fu2 = fft(yl)/dovg;
» Fulp = fftshift(Ful); Fu2p = fftshift(Fu2) ;
» % Формирование массивов модулей ФИ
» A1 = abs(Fulp); A2 = abs(Fu2p);
» % Вычисление Спектральных Плотностей Мощности
» S1 = Fulp.*conj(Fulp)*dovg; S2 = Fu2p.*conj(Fu2p)*dovg;
» % Вывод графиков белого шума
» subplot(2,1,1) ;
» stem(f,A1),grid,
» set(gca,'FontName','Arial Cyr','FontSize',10)
» title('Модуль ФИ гауссового белого шума');
» subplot(2,1,2) ;
» stem(f,S1),grid,
» set(gca,'FontName','Arial Cyr','FontSize',10)
» title('Спектральная плотность мощности');
» xlabel('Частота (Гц)');

```

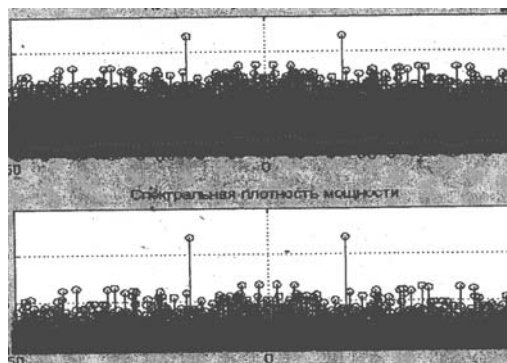


Рис. 5.38

Из рис. 5.38 видно, что спектральная плотность практически одинакова по величине во всем диапазоне частот, чем и обусловлено название процесса – белый шум, Аналогичную процедуру выполним и для "профильтрованного" процесса (рис. 5.39):

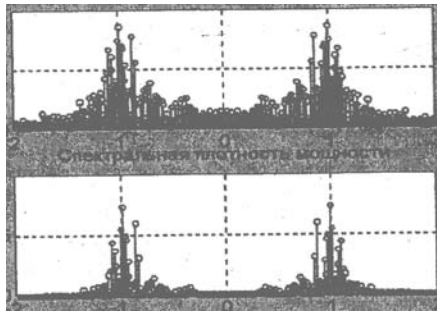


Рис. 5.39

```
% Вывод графиков профильтрованного процесса
cl = fix(dovg/2)-200, c2 = fix ('dovg/2) +200, length (f)
subplot(2,1,1);
stem(f(cl:c2),A2(cl:c2)).grid
subplot(2,1,2);
stem(f(cl:c2),S2(cl:c2)),grid
```

Проводя эти вычисления еще раз с новой длительностью процесса $T = 20$ с (рис. 5.40), можно наглядно убедиться, что величины ФИ и СПМ при этом практически не изменяются.

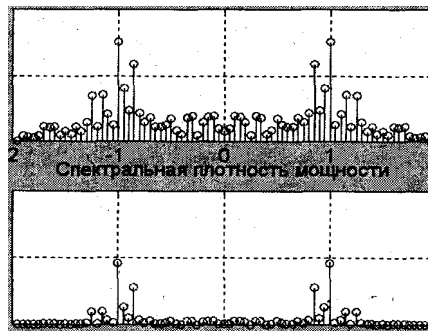


Рис. 5.40

5.3.3- Статистический анализ

К задачам статистического анализа процессов относится определение некоторых статистических характеристик процессов, а именно: корреляционных характеристик, спектральных плотностей мощности и т.п.

В предыдущем разделе уже были определены СП случайного процесса на основе установленной связи СП с Фурье-изображением. Однако в Signal Processing Toolbox предусмотрена специальная процедура **psd**, позволяющая сразу находить СП сигнала. Обращение к ней имеет вид:

```
[S,f] = psd(x,nfft,Fmax)
```

где x – вектор заданных значений процесса, $nfft$ – число элементов вектора, которые обрабатываются процедурой **fft**, $Fmax=1/Ts$ – значение частоты дискретизации сигнала, S – вектор значений СП сигнала, f – вектор значений частот, которым соответствуют найденные значения СП. В общем случае длина последних двух векторов равна $nfft/2$. Приведем пример применения процедуры **psd** для нахождения СП предыдущего случайного процесса:

```
» [C,f] = psd(y,dovg,Fmax);
» subplot
» stem(f(1:200),C(1:200)).grid,
» set(gca,'FontName','Arial Cyr','FontSize',16)
» title('Спектральная плотность мощности');
```

```
»xlabel('Частота (Гц)');
```

В результате получим картину, представленную на рис. 5.41.

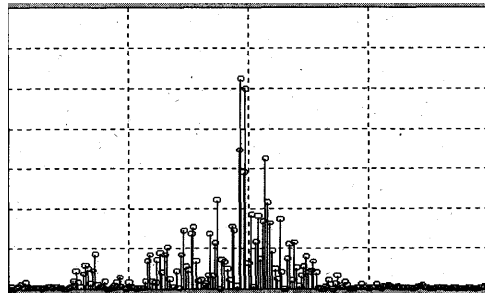


Рис. 5.41

Если ту же процедуру вызвать без указания выходных величин, то результатом ее выполнения станет выведение графика зависимости СП от частоты.

Например, обращение:

```
psd(yl<dovg,Fmax)
```

приведет к построению в графическом окне (фигуре) графика, как на рис. 5.42. При этом значения СП будут откладываться в логарифмическом масштабе в децибелах.

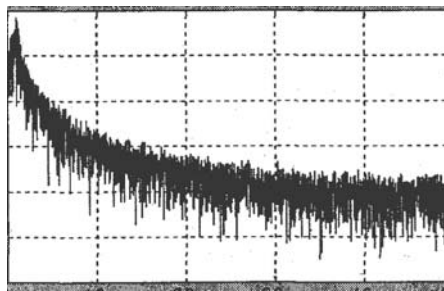


Рис. 5.42

Группа функций **xcorr** вычисляет оценку взаимной корреляционной функции (ВКФ) двух последовательностей x и y . Обращение $c = \text{xcorr}(x,y)$ вычисляет и выдает вектор c длины $2N-1$ значений ВКФ векторов x и y длины N . Также позволяет вычислить АКФ (автокорреляционную функцию) последовательности, заданной в векторе x . Вычислим АКФ для случайного процесса, сформированного ранее:

```
» R = xcorr(yl);
» tau = -10+Ts : Ts : 10;
» lt = length(tau);
» slr = round(length(R)/2)-lt/2;
» s2r = round(length(R)/2)+lt/2-l;
» plot(tau,R(slr:s2r)),grid
```

На рис. 5.43 представлен результат применения процедуры **xcorr**.

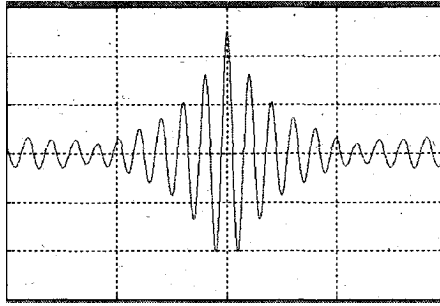


Рис. 5.43

5.4. Проектирование фильтров

5.4.1. Формы представления фильтров и их преобразования

Фильтр как звено системы автоматического управления может быть представлен в нескольких эквивалентных формах, каждая из которых полностью описывает его:

- В форме рациональной передаточной функции (**tf** - представление); если звено является непрерывным (аналоговым), то оно описывается непрерывной передаточной функцией:

$$W(s) = \frac{b(s)}{a(s)} = \frac{b(1) \cdot s^m + b(2) \cdot s^{m-1} + \dots + b(m+1)}{a(1) \cdot s^n + a(2) \cdot s^{n-1} + \dots + a(n+1)} \quad (5.34)$$

а в случае дискретного фильтра последний может быть представлен дискретной передаточной функцией вида:

$$W(z) = \frac{b(z)}{a(z)} = \frac{b(1) + b(2) \cdot z^{-1} + \dots + b(m+1) \cdot z^{-m}}{a(1) + a(2) \cdot z^{-1} + \dots + a(n+1) \cdot z^{-n}} \quad (5.35)$$

В обоих случаях для задания звена достаточно задать два вектора: **b** – вектор коэффициентов числителя и **a** – знаменателя передаточной функции.

- В виде разложения передаточной функции на простые дроби;
в случае простых корней такое разложение имеет вид (для дискретной передаточной функции):

$$\frac{b(z)}{a(z)} = \frac{r(1)}{1 - p(1) \cdot z^{-1}} + \dots + \frac{r(n)}{1 - p(n) \cdot z^{-1}} + k(1) + k(2) \cdot z^{-1} + \dots + k(m-n+1) \cdot z^{-(m-n)} \quad (5.36)$$

- В этой форме звено описывается тремя векторами: вектором-столбцом **r** вычетов передаточной функции, вектором-столбцом **p** полюсов и вектором-строкой **k** коэффициентов целой части дробно-рациональной функции.

- В каскадной форме (**sos**-представление), когда передаточная функция звена представлена в виде произведения передаточных функций не выше второго порядка:

$$H(z) = \prod_{k=1}^L H_k(z) = \prod_{k=1}^L \frac{b_{0k} + b_{1k} \cdot z^{-1} + b_{2k} \cdot z^{-2}}{a_{0k} + a_{1k} \cdot z^{-1} + a_{2k} \cdot z^{-2}} \quad (5.37)$$

Параметры каскадного представления задаются в виде матрицы **sos**, содержащей вещественные коэффициенты:

$$sos = \begin{bmatrix} b_{01} & b_{11} & b_{21} & a_{01} & a_{11} & a_{21} \\ b_{02} & b_{12} & b_{22} & a_{02} & a_{12} & a_{22} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ b_{0L} & b_{1L} & b_{2L} & a_{0L} & a_{1L} & a_{2L} \end{bmatrix} \quad (5.38)$$

- В пространстве состояний (**ss**-представление), т.е. с помощью уравнений звена в форме:

$$\begin{aligned} \dot{x} &= A \cdot x + B \cdot u \\ y &= C \cdot x + D \cdot u \end{aligned} \quad (5.39)$$

В этой форме звено задается совокупностью четырех матриц A, B, C и D;

- Путем задания векторов z нулей передаточной функции, p – ее полюсов и k – коэффициента передачи звена (**zp** -представление):

$$W(s) = k \cdot \frac{[s - z(1)] \cdot [s - z(2)] \cdot \dots \cdot [s - z(m)]}{[s - p(1)] \cdot [s - p(2)] \cdot \dots \cdot [s - p(n)]} \quad (5.40)$$

• Решетчатое **latc** -представление; в этом случае решетчатый фильтр задается векторами k коэффициентов знаменателя решетчатого дискретного фильтра и v – коэффициентов его числителя; коэффициенты k решетчатого представления некоторого полинома с коэффициентами, представленными вектором a, определяются по этому вектору с помощью рекурсивного алгоритма Левинсона. Пакет Signal предоставляет пользователю ряд процедур, позволяющих преобразовать звено (фильтр) из одной формы в другую.

Процедуры преобразования, к tf-форме

1. Процедура **zp2tf** осуществляет вычисления векторов коэффициентов числителя (b) и знаменателя (a) передаточной функции в форме (5.34) по известным векторам z ее нулей, p – ее полюсов и k – коэффициенту усиления, звена. Обращение к процедуре имеет вид:

$$[b,a] = zp2tf(z, p, k)$$

В общем случае многомерного звена величина z является матрицей, число столбцов которой должно быть равно числу выходов. Вектор-столбец k содержит коэффициенты усиления по всем выходам звена. В векторе a выдаются вычисленные коэффициенты знаменателя, а матрица b содержит коэффициенты числителей. При этом каждая строка матрицы соответствует коэффициентам числителя для отдельной выходной величины.

2. Процедура **ss2tf** преобразовывает описание звена (системы) из пространства состояний в форму передаточной функции. Обращение к ней вида:

$$[b,a] = ss2tf(A, B, C, D, iu)$$

позволяет найти коэффициенты числителей (b) и знаменателя (a) передаточных функций системы по всем выходным величинам и по входу с номером iu, если заданы матрицы A, B, C, D описания системы в виде (5.39).

3. Процедура **sos2tf** позволяет найти передаточную функцию звена по заданным параметрам каскадной формы. Для этого надо обратиться к этой процедуре таким образом:

$$[b,a] = sos2tf(sos)$$

где sos – заданная матрица каскадной формы (5.38).

4. С помощью процедуры **latc2tf** можно вычислить коэффициенты числителя и знаменателя передаточной функции (5.35) по коэффициентам знаменателя и числителя решетчатого фильтра. При этом обращение к ней должно иметь один из видов:

$$\begin{aligned} [b,a] &= \text{latc2tf}(k,V); \\ [b,a] &= \text{latc2tf}(k,'iir') ; \\ b &= \text{latc2tf}(k,'fir'); \\ b &= \text{latc2tf}(k) \end{aligned}$$

Первый вид используется, если заданы коэффициенты решетчатого представления и числителя v , и знаменателя k БИХ-фильтра (фильтра с бесконечной импульсной характеристикой). Второй вид применяется, если решетчатый БИХ-фильтр имеет только полюсы. Третий и четвертый виды служат для вычисления коэффициентов передаточной функции решетчатого КИХ-фильтра (с конечной импульсной характеристикой).

5. Нахождение коэффициентов передаточной функции по коэффициентам разложения ее на простые дроби (5.36) осуществляется путем использования функций **residue** и **residuez**. Первая применяется для непрерывной передаточной функции вида (5.34), вторая – для дискретной передаточной функции (5.35). При обращении вида:

$$\begin{aligned} [b,a] &= \text{residue}(r,p,k); \\ [b,a] &= \text{residuez}(r,p,k) \end{aligned}$$

вычисляются коэффициенты числителя и знаменателя передаточной функции по заданным векторам ее разложения – вычетов r , полюсов p и коэффициентам целой части k . С помощью тех же процедур осуществляется разложение заданной передаточной функции на простые дроби.

При этом обращение к ним должно быть таким:

$$\begin{aligned} [r,p,k] &= \text{residue}(b,a); \\ [r,p,k] &= \text{residuez}(b,a). \end{aligned}$$

Процедуры, перехода от tf-формы к другим формам

1. Вычисление нулей, полюсов и коэффициентов усиления звена заданной передаточной функцией можно осуществить, применяя процедуру **tf2zp** в такой форме:

$$[z,p,k] = \text{tf2zp}(b,a)$$

При этом вектор z будет содержать значения нулей передаточной функции с коэффициентами числителя (b) и знаменателя (a) вектор p – значения полюсов, а k будет равен коэффициенту усиления звена.

2. Нахождение матриц A , B , C и D , описывающих звено (заданной передаточной функцией в виде совокупности дифференциальных уравнений в форме Коши (5.39), осуществимо с помощью процедуры **tf2ss**. Если обратиться к ней в форме:

$$[A,B,C,D] = \text{tf2ss}(b,a)$$

где b и a – соответственно векторы коэффициентов числителя и знаменателя передаточной функции, то в результате получим искомые матрицы в указанном порядке.

3. Вычисление коэффициентов решетчатого фильтра по заданной дискретной передаточной функции можно осуществить при помощи процедуры **tf2latc**, обращаясь к ней следующим образом:

$$\begin{aligned} [k,v] &= \text{tf2latc}(b,a); \\ [k,v] &= \text{tf2latc}(1,a); \\ k &= \text{tf2latc}(1,a); \\ k &= \text{tf2latc}(b) \end{aligned}$$

Первое обращение позволяет вычислить коэффициенты знаменателя и v – числителя решетчатого БИХ-фильтра (модель авторегрессии скользящего среднего). Обращение во второй форме дает возможность определить вектор коэффициентов знаменателя k и скалярный коэффициент v , когда БИХ-фильтр имеет только полюсы (не имеет нулей). В третьей форме определяются только коэффициенты знаменателя решетчатого фильтра.

Наконец, четвертая форма предназначена для нахождения вектора k коэффициентов решетчатого КИХ-фильтра (задаваемого только вектором коэффициентов числителя передаточной функции).

Другие преобразования

1. Вычисление коэффициентов решетчатого представления по коэффициентам полинома можно осуществить, используя функцию **poly2rc**. Обращение:

$$k = \text{poly2rc}(a)$$

позволяет найти коэффициенты решетчатого представления k по коэффициентам a заданного полинома. Вектор a должен содержать только вещественные элементы, и должно выполняться условие $a \neq 0$. Размер вектора k на единицу меньше размера вектора a коэффициентов полинома.

По коэффициентам решетчатого представления очень просто определить, находятся ли все полюсы внутри единичного круга. Для этого достаточно проверить, что все элементы вектора k по абсолютной величине не превышают единицы. Обратная задача вычисления коэффициентов полинома по коэффициентам решетчатого представления решается путем применения функции **rc2poly**:

$$a = \text{rc2poly}(k)$$

2. Процедура **sos2ss** определяет матрицы (5.39) A, B, C и D , описывающие звено в пространстве состояний, по заданной матрице SOS каскадной формы (5.38):

$$[A, B, C, D] = \text{sos2ss}(SOS)$$

Элементы матрицы SOS должны быть вещественными. Обратный переход осуществляется при помощи функции **ss2sos**:

$$(A, B, C, D) = \text{ss2sos}(SOS)$$

3. Функция **sos2zp** дает возможность определить (5.40) векторы z нулей, p – полюсов и коэффициент усиления k звена, заданного каскадной формой передаточной функции (т.е. матрицей SOS (5.38)):

$$[z, p, k] = \text{sos2zp}(SOS)$$

Обратный переход осуществляется при помощи функции **zp2sos**:

$$SOS = \text{zp2sos}(z, p, k)$$

4. Нахождение нулей z , полюсов p и коэффициента усиления k звена по его описанию в пространстве состояний можно произвести путем обращения к процедуре **ss2zp** по форме:

$$[z, p, k] = \text{ss2zp}(A, B, C, D, iu)$$

где iu – номер входа, по которому ищется передаточная функция.

Обратное преобразование осуществляется процедурой **zp2ss**:

$$[A, B, C, D] = \text{zp2ss}(z, p, k)$$

5.4.2. Разработка аналоговых фильтров

Цель разработки фильтров заключается в обеспечении частотно - зависимого изменения заданной последовательности данных (сигнала). В простейшем случае разработки фильтра низких частот целью является построение такого звена, которое обеспечило бы отсутствие амплитудных искажений входного сигнала в области частот от 0 до некоторой заданной и эффективное подавление гармонических компонент с более высокими частотами.

Аналоговый фильтр может быть представлен непрерывной передаточной функцией:

$$W(s) = \frac{Y(s)}{X(s)} = \frac{M(s)}{N(s)} \quad (5.41)$$

где $Y(s)$ и $X(s)$ – изображения по Лапласу соответственно выходного и входного сигналов фильтра, а $M(s)$ и $N(s)$ – полиномы от s соответственно в числителе и знаменателе передаточной функции.

В качестве основных характеристик фильтра обычно принимают так называемую характеристику затухания $A(\omega)$, которая является величиной, обратной модулю частотной передаточной функции, и измеряется в децибелах:

$$A(\omega) = 20 \cdot \lg \{ |W(j\omega)|^{-1} \} = 10 \cdot \lg L(\omega^2) \quad (5.42)$$

фазовую характеристику $\vartheta(\omega)$:

$$\vartheta(\omega) = \arg(H(j\omega)) \quad (5.43)$$

и характеристику групповой задержки τ :

$$\tau = \frac{d\vartheta(\omega)}{d\omega} \quad (5.44)$$

Функцию

$$L(s^2) = [H(s) \cdot H(-s)]^{-1} \quad (5.45)$$

называют функцией затухания.

Нетрудно понять, что если z_i являются нулями передаточной функции $W(s)$, а p_i – ее полюсами, то нулями функции затухания будут $\pm p_i$ а полюсами $\pm z_i$.

Идеальный фильтр -низких частот (ФНЧ) пропускает только низкочастотные составляющие. Его характеристика затухания имеет вид, показанный на рис. 5.44 а.

Диапазон частот от 0 до ω_c называется полосой пропускания, остальной частотный диапазон – полосой задерживания. Граница между этими полосами (ω_c) называется частотой среза. Аналогично, идеальные фильтры высоких частот (ФВЧ), полосовой и режекторный можно определить как фильтры, имеющие характеристики затухания, показанные на рис. 5.44 б, в и г.

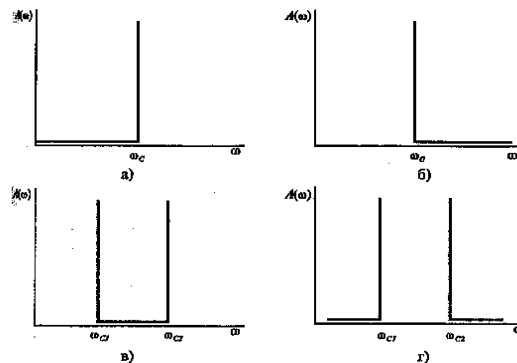


Рис. 5.44

Реальный ФНЧ отличается от идеального тем, что:

- затухание в полосе пропускания не равно нулю (децибел);
- затухание в полосе задерживания не равно бесконечности;
- переход от полосы пропускания к полосе задерживания происходит постепенно (не скачкообразно).

Возможный вид характеристики затухания реального фильтра приведен на рис. 5.45 а. На нем обозначено:

- ω_p – граничная частота полосы пропускания;
- ω_s – граничная частота полосы задерживания;
- R_p – максимальное подавление в полосе пропускания;
- R_s – минимальное подавление в полосе задерживания.

Частота среза в этом случае является условной границей между полосами пропускания и задерживания, которая определяется либо по уровню подавления в 3 дБ, либо как

$\sqrt{\omega_p \cdot \omega_s}$ в эллиптических фильтрах.

Типичные характеристики затухания реальных фильтров высоких частот, полосовых и режекторных представлены на рис. 5.45 б, в и г соответственно.

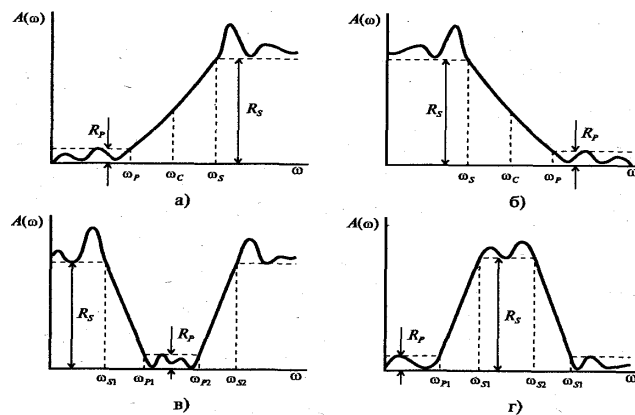


Рис. 5.45

Аппроксимацией фильтра называют реализуемую передаточную функцию, у которой график характеристики затухания $A(\omega)$ как функция от частоты приближается к одной из идеальных характеристик на рис. 5.44. Такая передаточная функция характеризует устойчивое физически реализуемое звено и должна удовлетворять следующим условиям:

- она должна быть рациональной функцией от s с вещественными коэффициентами;
- ее полюсы должны лежать в левой полуплоскости s -плоскости;
- степень полинома числителя должна быть меньше или равной степени полинома знаменателя.

В пакете SIGNAL предусмотрен ряд процедур, осуществляющих расчет аналоговых аппроксимаций фильтров низких частот.

Группа процедур **buttord**, **cheblord**, **cheb2ord**, **ellipord** используется для определения минимального порядка и частоты среза аналогового или цифрового фильтра по заданным характеристикам фильтра:

- W_p – граничной частоте пропускания;
- W_s – граничной частоте задерживания;
- R_p – максимально допустимому подавлению в полосе пропускания, дБ; -
- R_s – минимально допустимому подавлению в полосе задерживания, дБ.

Все эти процедуры предназначены для вычисления соответствующей аппроксимации ФНЧ, ФВЧ, полосовых и режекторных фильтров минимального порядка.

Функция **buttord** определяет порядок n и частоту среза W_n для аппроксимации в виде аналогового фильтра Баттерворта при обращении вида:

$$[n, W_n] = \text{buttord}(W_p, W_s, R_p, R_s, 's')$$

При этом значения частот W_p , W_s должны быть заданы в радианах в секунду, значение частоты среза W_n также получается в радианах в секунду. Для ФВЧ величина W_p должна превышать W_s . Для полосовых и режекторных фильтров W_p и W_s должны быть двухэлементными векторами, определяющими граничные частоты полос, причем первой должна стоять меньшая частота. В этом случае параметр W_n , вычисляемый процедурой, представляет собой двухэлементный вектор-строку.

Аналогично используются процедуры **cheblord**, **cheb2ord**, **ellipord**, которые определяют порядок аналоговых фильтров Чебышева 1-го и 2-го типов и эллиптического фильтра соответственно.

Вторая группа процедур позволяет определить векторы z нулей, p – полюсов и коэффициент усиления k основных аппроксимаций линейных фильтров заданного порядка n . К таким процедурам относятся **besselap**, **buttap**, **cheblap**, **cheb2ap** и **ellipap**, создающие фильтр низких частот (ФНЧ) с частотой среза, равной 1 радиан в секунду. Процедура **besselap** путем обращения к ней:

$$[z,p,k] = \text{besselap}(n)$$

вычисляет нули и полюсы аналогового фильтра Бесселя, процедура **buttapp** при помощи аналогичного обращения создает аналоговый фильтр Баттерворта.

Аналоговый прототип фильтра Чебышева нижних частот 1-го типа, имеющий пульсации в полосе пропускания не более R_p дБ, создается процедурой **cheblap** таким образом:

$$[z,p,k] = \text{cheblap}(n,R_p)$$

ФНЧ Чебышева 2-го типа, имеющий величину подавления в полосе задерживания не менее R_S дБ, создается с использованием процедуры **cheb2ap** так:

$$[z,p,k] = \text{cheb2ap}(n,R_S).$$

Процедура **ellipap** путем обращения к ней:

$$[z,p,k] = \text{ellipap}(n,R_p,R_S)$$

дает возможность найти нули и полюсы эллиптического ФНЧ, обеспечивающего пульсации в полосе пропускания не более R_p дБ и подавление в полосе задерживания не менее R_S дБ.

Третью группу образуют процедуры, позволяющие пересчитать параметры рассчитанного ФНЧ известной аппроксимации с частотой среза, равной 1, в ФНЧ, - ФВЧ, полосовой или режекторный фильтр с заданной частотой среза. Эта группа состоит из процедур **lp2lp**, **lp2hp**, **lp2bp** и **lp2bs**. Первая образует фильтр нижних частот, вторая – ФВЧ, третья – полосовой фильтр и четвертая – режекторный фильтр. Обращение к процедуре **lp2lp** может иметь две формы:

$$[bt,at] = \text{lp2lp}(b,a,W_0)$$

$$[At,Bt,Ct,Dt] = \text{lp2lp}(A,B,C,D,W_0)$$

Первая форма применяется при задании исходного ФНЧ (с частотой среза 1 рад/с) в виде коэффициентов числителя (a) и знаменателя (b). W_0 в этом случае означает желаемую частоту среза получаемого ФНЧ. Результатом являются вычисленные значения векторов коэффициентов числителя – at и знаменателя – bt полученного ФНЧ.

Вторая форма применяется при задании исходного ФНЧ в пространстве состояний.

Результат получается также в форме матриц пространства состояний.

Применение процедуры **lp2hp** формирования ФВЧ полностью аналогично.

Процедура **lp2bp** формирования полосового фильтра тоже имеет два вида вызова:

$$[bt,at] = \text{lp2bp}(b,a,W_0,W_w)$$

$$[At,Bt,Ct,Dt] = \text{lp2bp}(A,B,C,D,W_0,W_w)$$

Здесь параметр W_0 – это центральная частота полосы пропускания, а W_w означает ширину полосы пропускания. В остальном особенности использования и смысл обозначений сохраняются прежними.

Использование функции **lp2bs** проектирования режекторного типа полностью аналогично, за исключением того, что параметры W_0 и W_w в этом случае имеют смысл центра полосы задерживания и ее ширины.

Четвертую группу образуют процедуры полной разработки фильтров указанных аппроксимаций по заданным порядку и значению частоты среза W_c . В нее входят процедуры **besself**, **butter**, **cheby1**, **cheby2** и **ellip**. Общим для них всех является следующее.

- С их помощью можно проектировать ФНЧ, ФВЧ, полосовые и режекторные фильтры соответствующей аппроксимации; если параметр W_c является скаляром и флажок **high** после него не указан, то проектируется ФНЧ с частотой среза W_c ; если же указанный флажок в обращении есть, то в результате проектируется ФВЧ; если параметр W_c задай как вектор из двух величин, то результатом вычислений являются параметры полосового фильтра с полосой пропускания $W_1 \leq \omega \leq W_2$, где W_1 первый элемент этого вектора, а W_2 – второй элемент; наконец, если дополнительно к этому в конце списка входных параметров указан флажок **stop**, то рассчитываются параметры режекторного фильтра с полосой задерживания, указанной элементами вектора W_c .

- Результаты расчета фильтра могут иметь три формы в зависимости от того, какое количество параметров указано при обращении к процедуре в качестве выходных, например:

```
[b,a] = besself(n,Wc,'ftype')
[z,p,k] = besself(n,Wc,'ftype')
[A,B,C,D] = besself(n,Wc,'ftype')
```

Если указано два выходных параметра, то им будут присвоены значения коэффициентов числителя и знаменателя передаточной функции фильтра; при указании трех параметров на выходе, они примут значения векторов нулей, полюсов и коэффициента усиления фильтра; если же выходов указано четыре, то ими становятся значения матриц пространства состояний проектируемого фильтра;

- Почти все они могут применяться для проектирования как аналоговых, так и цифровых фильтров; чтобы с их помощью создать аналоговый фильтр, необходимо в число входных параметров процедуры последним включить специальный флажок (**s**); исключение составляет фильтр Бесселя, аналога которому в цифровой форме не существует.

5.4.3. Проектирование цифровых БИХ-фильтров

Конечной задачей проектирования линейного цифрового фильтра будем считать расчет значений элементов векторов **b** числителя и **a** знаменателя его дискретной передаточной функции $G(z)$, записанной в виде (5.7):

$$G(z) = \frac{y(z)}{x(z)} = \frac{b_0 + b_1 \cdot z^{-1} + \dots + b_m \cdot z^{-m}}{a_0 + a_1 \cdot z^{-1} + \dots + a_n \cdot z^{-n}}$$

Если эти два вектора известны, осуществление самой фильтрации, как было сказано ранее, происходит путем применения процедуры **filter**, в которой аргументами выступают эти векторы.

Напомним, что представленная дискретная передаточная функция описывает в сжатой форме такое конечно-разностное уравнение фильтра:

$$\begin{aligned} a_0 \cdot y(k) + a_1 \cdot y(k-1) + a_2 \cdot y(k-2) + \dots + a_n \cdot y(k-n) = \\ = b_0 \cdot x(k) + b_1 \cdot x(k-1) + \dots + b_m \cdot x(k-m) \end{aligned} \quad (5.46)$$

Если $n = 0$, фильтр называют нерекурсивным, а число m – порядком фильтра. Такой фильтр имеет конечную импульсную характеристику, поэтому его также называют КИХ - фильтром.

В случае $n > 0$ фильтр называется рекурсивным. Порядком фильтра при этом называют наибольшее из чисел m и n . В этом случае импульсная характеристика фильтра является бесконечной, и его называют также БИХ - фильтром. БИХ - фильтры представляют собой некоторые аналоги динамических звеньев.

Одним из средств проектирования БИХ - фильтров, предусмотренных в пакете Signal, является разработка соответствующего аналогового прототипа, т.е. нахождение передаточной функции по Лапласу непрерывного фильтра, и последующий переход к цифровому фильтру путем нахождения цифрового аналога непрерывного звена.

Последнее можно осуществить с помощью билинейного преобразования s -плоскости в z -плоскость. Билинейное преобразование выполняется в соответствии с выражением:

$$H(z) = H(s) \Big|_{s = 2f_s \frac{z-1}{z+1}} \quad (5.47)$$

где f_s – частота дискретизации сигнала. При этом ось $j\omega$ преобразуется в единичную окружность на z - плоскости.

В пакете Signal билинейное преобразование осуществляется с помощью процедуры **bilinear**, к которой можно обратиться тремя способами:

```
[bd,ad] = bilinear(b,a,Fs,Fp);
[zd,pd,kd] = bilinear (z,p,k,Fs,Fp) ;
```

$$[Ad, Bd, Cd, Dd] = \text{bilinear}(A, B, C, D, Fs, Fp)$$

Все они преобразуют параметры, характеризующие аналоговый прототип фильтра, в аналогичные параметры, описывающие дискретный БИХ-фильтр. Вид и количество входных параметров определяют вид и число выходных. Параметр F_s задает частоту дискретизации в герцах. Параметр F_p не обязателен. Он определяет частоту в герцах, для которой значения АЧХ до и после выполнения преобразования должны совпадать, т.е. задает так называемые предискажения.

Обращение в первой форме позволяет определить коэффициенты полиномов числителя и знаменателя дискретной передаточной функции фильтра вида (5.35) по заданным коэффициентам полиномов числителя и знаменателя непрерывной передаточной функции вида (5.34). Обращение во второй форме дает возможность вычислить нули, полюсы и коэффициент усиления дискретного фильтра по заданным аналогичным параметрам аналогового прототипа. И наконец, третья форма определяет матрицы дискретного пространства состояний фильтра по известным матрицам непрерывного пространства состояний.

Второй способ построения цифрового фильтра по его Анало - говому прототипу заключается в таком преобразовании параметров аналогового фильтра в параметры дискретного фильтра, при котором импульсная характеристика последнего совпадала бы с импульсной характеристикой аналогового фильтра в точках через дискрет по времени.

Это в MatLAB осуществляется применением процедуры **impinvar**:

```
[bz,az] =impinvar(b,a,Fs)
[b,a] =maxflat(nb,na,Wc)'
[b,a] =maxflat(nb,'sym',Wc)
[b,a,b1,b2] =maxflat(nb,na,Wc)
[b,a] =maxflat(nb,na,Wc,'design_flag')
```

Здесь b и a – заданные векторы коэффициентов числителя и знаменателя передаточной функции аналогового прототипа фильтра, bz и az – вычисляемые коэффициенты числителя и знаменателя дискретной передаточной функции дискретного фильтра, F_s – заданная частота дискретизации сигнала в герцах. Если параметр F_s при обращении не указан, то по умолчанию он принимается равным 1 Гц.

Третий способ формирования дискретных фильтров – использование ранее рассмотренных процедур формирования фильтров **butter**, **cheby1**, **cheby2** и **ellip**. Если при обращении к этим процедурам не указывать в конце списка входных- параметров флажка s , то результатом работы этих процедур будут параметры именно цифровых фильтров.

Основное отличие применения этих функций для разработки цифровых фильтров заключается в другом представлении задаваемых частот в векторе Wc . Все частоты должны задаваться по отношению к так называемой частоте Найквиста. Частотой Найквиста называют половину частоты дискретизации сигнала.

Так как диапазон частот изменения дискретного сигнала всегда меньше частоты дискретизации, то все частотные характеристики дискретных фильтров определяются только внутри диапазона от 0 до частоты - Найквиста. Поэтому все задаваемые в векторе Wc граничные частоты должны быть меньше единицы.

Существуют еще две процедуры расчета БИХ-фильтров.

Процедура **maxflat** производит расчет обобщенного цифрового фильтра Баттерворта.

Формы обращения к ней таковы:

Первое обращение позволяет вычислить коэффициенты b – числителя и a – знаменателя дискретной передаточной функции $H(z)$ цифрового ФНЧ Баттерворта с частотой среза Wc , порядок числителя которой равен nb , а знаменателя – na .

При обращении второго- вида вычисляются коэффициенты цифрового симметричного КИХ-фильтра Баттерворта. В этом - случае na принимается равным 0. Параметр nb должен быть четным.

Если обратиться к процедуре так, как указано в третьем обращении, указать в качестве выходных четыре величины, то дополнительные параметры *b1* и *b2* дадут коэффициенты двух полиномов, произведение которых является полиномом числителя *b* искомой дискретной передаточной функции, причем все нули полинома *b1* равны -1, а полином *b2* содержит все остальные нули полинома *b*.

Добавление в список входных параметров процедуры параметра '**design_flag**' позволяет изменять характер выводимой на экран информации. Если значение этого параметра равно **trace**, на экране отображаются параметры, используемые в процессе проектирования:

```
» nb = 10; na = 2 ; w = 0.5 ;
» [b,a,b,b2] = maxflat(nb,na,w,'trace')
```

Table:

L	M	N	wo_min/pi	wo_max/pi
10.0000	0	2.0000	0	0.2394
9.0000	1.0000	2.0000	0.2394	0.3259
8.0000	2.0000	2.0000	0.3259	0.3991
7.0000	3.0000	2.0000	0.3991	0.4669
6.0000	4.0000	2.0000	0.4669	0.5331
5.0000	5.0000	2.0000	0.5331	0.6009
4.0000	6.0000	2.0000	0.6009	0.6741
3.0000	7.0000	2.0000	0.6741	0.7606
2.0000	8.0000	2.0000	0.7606	1.0000

b =

Columns 1 through 7

0.0414 0.2263 0.4932 0.5252 0.2494 0.0079 -0.0266

Columns 8 through 11

0.0008 0.0023 -0.0004 -0.0000

a =

1.0000 - 0 0.5195

b1 =

1 6 15 20 15 6 1

b2 =

0.0414 -0.0221 0.0049 -0.0004 -0.0000

Если же этому параметру задать значение **plots**, то на экран будут выведены графики амплитудной характеристики, группового времени замедления, а также графическое изображение нулей и полюсов:

```
» [b,a,b1,b2] = maxflat(nb,na,w,'plots ')
```

b =

Columns 1 through 7

```

0.0414 0.2263 0.4932 0.5252 0.2494 0.0079 -0.0266
Columns 8 through 11
0.0008 0.0023 -0.0004 -0.0000
a = 1.0000 0 0.5195
b1 = 1 6 15 20 15 6 1
b2 = -0.0221 0.0049 -0.0004 -0.0000

```

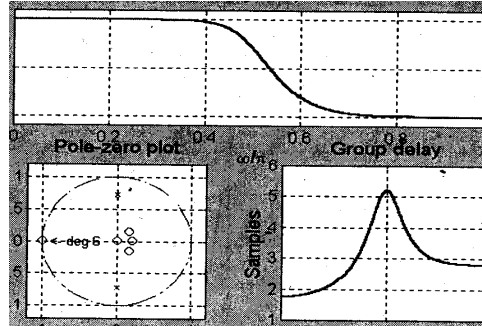


Рис. 5.46

Расчет БИХ-фильтра по заданной амплитудно-частотной характеристике производится процедурой **yulewalk**. Если набрать в командном окне MatLAB строку:

```
» [b,a] = yulewalk(n,f,m)
```

то будут вычислены коэффициенты **b** числителя и **a** знаменателя дискретной передаточной функции БИХ-фильтра порядка **n**, АЧХ которого задана векторами **f** (частоты в нормированных значениях) и **m** – соответствующих значений отношений амплитуд выхода и входа. Первый элемент вектора **f** должен быть равен 0, а последний – 1. Все остальные элементы должны быть расположены в неубывающем порядке. Частоты, при которых происходит скачок АЧХ, указываются два раза с разными значениями соответствующих им амплитуд.

Приведем пример расчета ФНЧ 8-го порядка и построим желаемую АЧХ и АЧХ полученного фильтра (рис. 5.47).

```

» f = [0 0.5 0.5 1];
» m = [1 100];
» [b,a] = yulewalk(8,f,m);
» [h,w] = freqz(b,a,128);
» plot(f,m,w/pi,abs(h))
» grid, title('Пример использования процедуры YULEWALK')
» xlabel('Нормализованная частота')
» ylabel('А Ч Х')

```



Рис. 5.47

5.4.4. Проектирование КИХ-фильтров

В отличие от БИХ-фильтров, которые характеризуются двумя векторами – коэффициентов **b** числителя и **a** знаменателя своей дискретной передаточной функции,

КИХ-фильтры описываются только одним вектором b . Знаменатель их дискретной передаточной функции тождественно равен 1.

Группа функций **firl** предназначена для расчета коэффициентов b цифрового КИХ-фильтра с линейной фазой методом взвешивания с использованием окна. Общий вид обращения к этой процедуре:

```
b = firl(n,Wn, 'ftype', window)
```

Процедура вычисляет вектор $n+1$ коэффициентов b КИХ-фильтра с нормализованной частотой среза Wn .

Параметр 'ftype' задает желаемый тип фильтра (ФНЧ, ФВЧ, полосовой или режекторный). Он может отсутствовать (и тогда по умолчанию рассчитываются параметры ФНЧ с частотой среза Wn , если последняя задана как скаляр, или полосового фильтра с полосой пропускания от $W1$ до $W2$, если параметр Wn задан в виде вектора из двух элементов [$W1$ $W2$]) или принимать одно из четырех значений: high, stop, DC-1 и DC-0. В первом случае синтезируется ФВЧ с частотой среза Wn . Во втором – режекторный фильтр (при этом Wn должен быть вектором из двух элементов, значения которых определяют границы полосы задерживания по отношению к частоте Найквиста). В третьем случае рассчитываются параметры многополосного фильтра, первая полоса которого является полосой пропускания, а в четвертом – тоже многополосный фильтр, первая полоса которого является полосой задерживания.

При расчете режекторных фильтров и ФВЧ порядок фильтра следует назначать четным числом.

Параметр window позволяет задавать отсчеты окна в векторе-столбце window длины $n+1$. Если этот параметр не указан, то, по умолчанию, будет использовано окно Хемминга.

Для вычисления окон различного типа в MatLAB предусмотрены функции:

bartlett(n) Создает вектор-столбец из n элементов окна Бартлетта

blackman(n) Создает вектор-столбец из n элементов окна Блэкмана

boxcar(n) Создает вектор-столбец из n элементов прямоугольного окна

chebwin(n,r) Создает вектор-столбец из n элементов окна Чебышева, где r – желаемый уровень допустимых пульсаций в полосе задерживания в децибелах

bamming(n) Создает вектор-столбец из n элементов окна Хэмминга

hanning(n) Создает вектор-столбец из n элементов окна Хэннинга

kaizer(n,beta) Создает вектор-столбец из n элементов окна Кайзера, где параметр β определяет затухание боковых лепестков преобразования Фурье-окна

triang(n) Создает вектор-столбец из n элементов треугольного окна

Приведем пример. Произведем расчет полосового КИХ-фильтра 24-го порядка с полосой пропускания $0,35 \leq \omega / \omega_N \leq 0,65$:

```
» b = firl(48,[0.35 0.65]);
```

```
» freqz(b,1,512)
```

Результат показан на рис. 5.48.

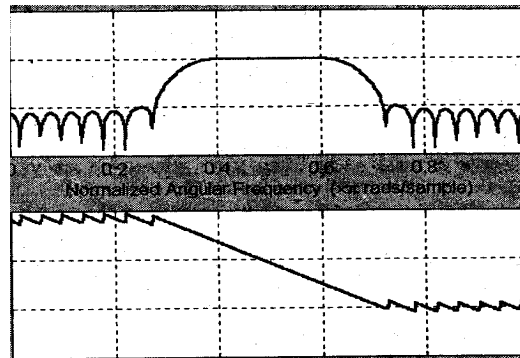


Рис. 5.48

Группа процедур **fir2** служит для расчета коэффициентов цифрового КИХ-фильтра с произвольной амплитудно - частотной характеристикой, задаваемой векторами **f** частот и **m** – соответствующих желаемых значений АЧХ. Общий вид обращения к процедуре таков:

```
b = fir2(n,f,m,npt,lap>window)
```

Вектор **f** должен содержать значения нормализованной частоты в неубывающем порядке от 0 до 1. Вектор **m** должен быть той же длины, что и вектор **f**, и содержать желаемые значения АЧХ на соответствующих частотах.

Параметр **npt** позволяет задать число точек, по которым выполняется интерполяция АЧХ.

Параметр **lap** определяет размер (число точек) области около точек скачкообразного изменения АЧХ, в которой выполняется сглаживание. Если эти параметры не указаны, то по умолчанию принимается **npt = 512** и **lap = 25**.

Рассчитаем двухполосный фильтр 30-го порядка:

```
» f = [0 0.2 0.2 0.6 0.6 0.8 0.8 1];
» m = [1 1000.50.500];
» b = fir2 (30,f,m) ;
» [h,w] = freqzfb, 1,512) ;
» plot(f,m,w/pi,abs(h))
» grid
» title('АЧХ КИХ-фильтра (процедура FIR2)')
» xlabel('Нормализованная частота')
» ylabel ('АЧХ')
```

На рис. 5.49 приведены желаемая АЧХ и АЧХ, полученная в результате расчета.

Следующая процедура – **fircis** – также рассчитывает многополосный фильтр, но в несколько другой форме – путем задания кусочно - постоянной желаемой АЧХ.

Формат обращения к ней имеет вид:

```
b = fircis(n,f,amp,up,lo,'design_flag')
```

Здесь **f**, как и ранее, вектор значений нормализованных частот (от 0 до 1), определяющих границы полос фильтра. Вектор **amp** определяет кусочно - постоянную требуемую АЧХ фильтра, количество его элементов равно числу полос фильтра и, следовательно, на 1 меньше числа элементов вектора **f**. Векторы **up** и **lo** определяют соответственно верхние и нижние допустимые отклонения АЧХ спроектированного фильтра от желаемой АЧХ для каждой из полос. Размер их совпадает с размером вектора **amp**.

Параметр **'design_flag'** может принимать три значения:

- **trase** – для обеспечения вывода результатов в виде текстовой таблицы;
- **plots** – для графического отображения АЧХ, групповой задержки, нулей и полюсов;
- **both** – для отображения результатов как в текстовой, так и в графической форме.

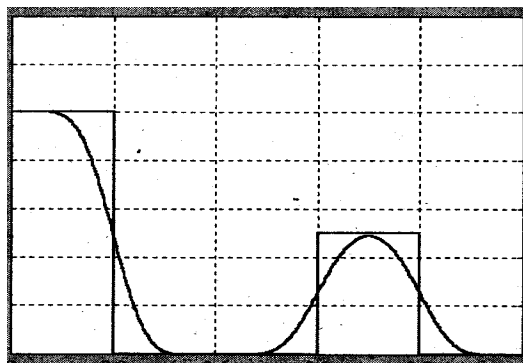


Рис. 5.49

Приведем пример разработки прежнего двухполосного фильтра:

```
» n = 30;
» f = [0 0.2 0.6 0.8 1];
» amp = [10 0.5 0];
```

```

» up = [1.02 0.02 0.51 0.02];
» lo = [0.98 -0.02 0.49 -0.02 ];
» b = fircls(n,f,aig>,up,lo', 'both')

```

Результат приведен ниже и на рис. 5.50.

Bound Violation = 0.0755112846369

Bound Violation = 0.-0116144793011

Bound Violation = 0.0004154355279

Bound Violation = 0.0000905996658

Bound Violation = 0.0000214272508

Bound Violation = 0.0000009624286

Bound Violation = 0.0000002393147

Bound Violation = 0.0000000596813

Bound Violation = 0.0000000146532

Bound Violation = 0.0000000036610

b =

Columns 1 through 7

```

-0.0001 -0.0031 0.0226 0.0101 0.0060 0.0011 -0.0105 Columns 8 through 14
-0.0231 -0.0626 0.0090 -0.0001 -0.0145 0.1775 0.1194 Columns 15 through 21
0.1272 0.3023 0.1272 0.1194 0.1775 -0.0145 -0.0001 Columns .22 through 28
0.0090 -0.0626 -0.0231 -0.0105 0.0011 0.0060 0.0101 Columns 29 through 31
0.226 -0.0031 -0.0001

```

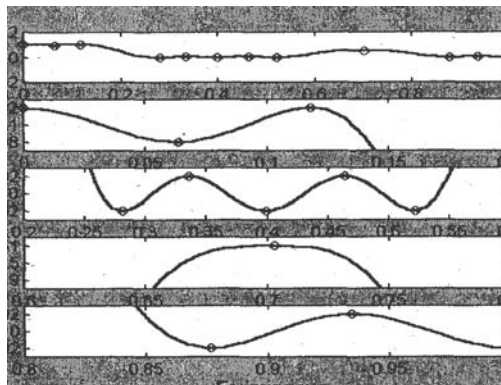


Рис. 5.50

Для сравнения с результатами работы процедуры **fir2** построим график полученной АЧХ, аналогичный приведенному на рис. 5.49:

```

» [h,w] = freqz(b,1,512);
» plot (w/pi,abs (h) )
» grid
» title('АЧХ КИХ - фильтра (процедура FIRCLS)')
» xlabel('Нормализованная частота')
» ylabel('А Ч X')

```

Результат представлен на рис. 5.51.

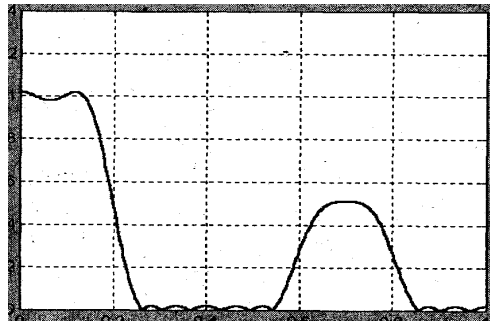


Рис. 5.51

Процедура **fircls1** предназначена для расчета параметров ФНЧ и ФВЧ с КИХ методом наименьших квадратов с учетом допусков на отклонения АЧХ. Предусмотрены следующие виды обращения к этой процедуре:

```

b = fircls1(n, Wo, dp, ds)
b = fircls1(n, Wo, dp, ds, 'high')
b = fircls1(n, Wo, dp, ds, Wt)
b = fircls1(n, Wo, dp, ds, Wt, 'high')
b = fircls1(n, Wo, dp, ds, Wp, Ws, k)
b = fircls1(n, Wo, dp, ds, Wp, Ws, k, 'high')
b = fircls1(n, Wo, dp, ds, ..., 'design_flag')

```

Параметр W_o представляет собой нормализованную частоту среза; dp определяет максимально допустимое отклонение АЧХ рассчитанного фильтра от 1 в полосе пропускания, а ds – максимальное отклонение АЧХ рассчитанного фильтра от 0 в полосе задерживания.

Наличие флажка 'high' определяет, что рассчитываются параметры ФВЧ. Если этот флажок отсутствует, рассчитывается ФНЧ.

С помощью параметра W_t задается частота W_t , выше которой при $W_t > W_o$ или ниже которой при $W_t < W_o$ гарантируется выполнение требований к АЧХ синтезируемого фильтра.

Параметры W_p , W_s и k позволяют соответственно задать граничную частоту пропускания, граничную частоту задерживания и отношение ошибки в полосе пропускания к ошибке в полосе задерживания.

Флаг 'design_flag' имеет тот же смысл и принимает те же значения, что и у предыдущей процедуры.

Группа процедур **remez** осуществляет расчет коэффициентов цифрового КИХ - фильтра с линейной ФЧХ по алгоритму Паркса - МакКлелла, в котором использован обменный алгоритм Ремеза и метод аппроксимации Чебышева. При этом минимизируется максимальное отклонение АЧХ спроектированного фильтра от желаемой АЧХ. Приведем наиболее полный вид обращения к процедуре:

```

b = remez (n, f, a, W, 'ftype')

```

Вектор f должен состоять из последовательных, в возрастающем порядке записанных пар нормализованных (от 0 до 1) частот, определяющих соответственно нижнюю и верхнюю границы диапазона полосы пропускания или задерживания. Вектор a должен содержать желаемые значения АЧХ на частотах, определяемых соответствующими элементами вектора f . Желаемая АЧХ в полосе частот от $f(k)$ до $f(k+1)$ при нечетном k представляет собой отрезок : прямой от точки $f(k)$, $a(k)$ до точки $f(k+1)$, $a(k+1)$. В диапазонах от, $f(k)$ до $f(k+1)$ при четном k значение желаемой АЧХ не определено (а значит, при проектировании фильтра АЧХ в этих диапазонах ' может принимать любое значение). Следует заметить, что значение $f(1)$ всегда должно быть равным 0. Кроме того, необходимо, чтобы, векторы f

и a были одинаковой длины, причем общее количество элементов каждого вектора должно быть четным числом.

Вектор W задает значения коэффициентов веса каждой из полос, АЧХ, заданных парами частот вектора f . Эти коэффициенты используются при аппроксимации АЧХ и определяют достигаемое при аппроксимации соотношение между реальным и желаемым значением АЧХ в каждом из диапазонов. Число элементов вектора; W равно половине числа элементов вектора f . |

Флаг 'ftype' может принимать одно из двух значений:

- **hilbert** – в этом случае процедура проектирует фильтры с нечетной симметрией и линейной фазой;
- **differentiator** – синтезируется фильтр с использованием специальных методов взвешивания; при этом для ошибок задаются веса, пропорциональные $1/f$; поэтому ошибки аппроксимации на низких частотах меньше, чем на высоких; для дифференциаторов, АЧХ которых, пропорциональна частоте, минимизируется максимальная относительная ошибка.

Ниже приводится пример проектирования полосового фильтра 17-го порядка:

```

» f = [0.0.3 0.4 0.6 .0.7 1];
» a = [0 0 1 1 0 0];
» b = remez(17,f,a);
» [h, w] = freqz(b, 1, 512);
» plot (f,a,w/pi,abs (h) ) , grid
» title('АЧХ КИХ-фильтра (процедура REMEZ)')
» xlabel('Нормализованная частота')
» ylabel('А Ч X')

```

Результат приведен на рис. 5.52.

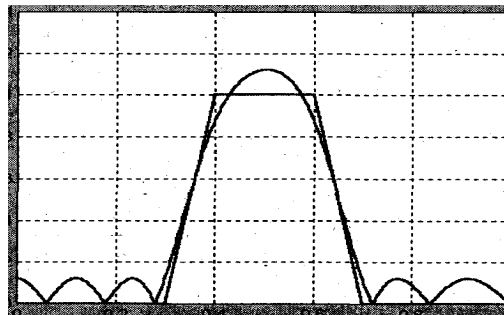


Рис. 5.52

Особенностью следующей процедуры, **cremez**, является то, что исходные данные по желаемой форме АЧХ фильтра задаются в виде функции, условно обозначенной **fresp**. Формы обращения к этой процедуре приведены ниже:

```

b = cremez(n,f,'fresp')
b = cremez(n,f,'fresp',w)
b = cremez (n f ,{'fresp',p1,p2,...},w)
b = cremez(n, f, a, w)
b = cremez(...,'sym')
b = cremez(...,'debug')
b = cremez(...,'skip_stage2')
[b,delta,opt] = cremez(...)

```

Параметры n , f имеют тот же смысл и такие же требования, к их представлению, как при применении процедуры **remez**. В отличие от последней, вектор значений желаемой АЧХ, соответствующих заданным значениям вектора f , определяется путем обращения к функции **fresp**.

Функция **fresp** может принимать одно из следующих значений:

- **lowpass**, **highpass**, **bandpass**, **bandstop** (ФНЧ, ФВЧ, полосовой и режекторный фильтры) – при этом рассчитываются параметры указанного типа фильтра; если к функции **fresp** не указаны дополнительные параметры (первые два вида обращений к процедуре), то групповое время задержки (ГВЗ) принимается равным $n/2$; в случае же обращения к процедуре в третьей форме, где в качестве дополнительного параметра функции **fresp** указан один – d , $\text{ГВЗ} = n/2 + d$;
- **multiband** (многополосный фильтр) – синтезируется фильтр, заданный вектором a желаемой АЧХ при значениях частот, определенных вектором f ; при этом вектор a указывается в качестве первого дополнительного параметра к функции **multiband** (третья форма обращения); если кроме этого вектора не указаны другие дополнительные параметры, то ГВЗ принимается равным $n/2$, если же указан еще один дополнительный, параметр – d , то $\text{ГВЗ} = n/2 + d$;
- **differentiator** (дифференциатор) – эта функция позволяет рассчитывать коэффициенты дифференцирующего фильтра с линейной фазой; при обращении к этой функции в качестве дополнительного параметра необходимо указать частоту дискретизации F_s ; по умолчанию $F_s = 1$;
- **hilbfilt** (фильтр Гильберта) – в этом случае находятся коэффициенты фильтра Гильберта с линейной фазой. Четвертая форма обращения к процедуре эквивалентна следующей:

$$b = \text{cremez}(n, f, ('multiband', a), w)$$

Параметр **sym** позволяет задать тип симметрии импульсной характеристики (ИХ) фильтра. Он может принимать следующие значения:

- **none** – в этом случае ИХ может быть произвольной; это значение параметра используется по умолчанию, если при определении желаемой АЧХ задаются отрицательные значения частот;
- **even** – АЧХ должна быть вещественной с четным типом симметрии; такое значение параметра используется по умолчанию при проектировании ФНЧ, ФВЧ, полосовых и режекторных фильтров;
- **odd** – АЧХ должна быть вещественной с нечетным типом симметрии; такое значение по умолчанию используется при проектировании фильтров Гильберта и дифференциаторов;
- **real** – АЧХ должна иметь сопряженный тип симметрии.

Использование флага **skip_stage2** (седьмой вид обращения к процедуре) позволяет не выполнять второй этап алгоритма оптимизации, который рассчитывает коэффициенты фильтра в тех случаях, когда этого нельзя сделать с помощью алгоритма Ремеза.

Исключение второго этапа сокращает время расчетов, но может повлечь снижение точности. По умолчанию выполняются оба этапа оптимизации. Параметр **debug** (см. шестой тип вызова процедуры) определяет вид выводимых на экран результатов расчета фильтра и может принимать следующие значения: **trace**, **plots**, **both** и **off**. По умолчанию используется **off** (т.е. на экран не выводится информация).

Использование дополнительного выходного параметра **delta** (см. восьмой вид обращения к процедуре) дает возможность применять в дальнейших операциях значение максимальной амплитуды пульсаций АЧХ.

Выходной параметр **opt** содержит набор дополнительных характеристик:

- **opt.grid** – вектор отсчетов частоты, использованных при оптимизации;

- `opt.H` – вектор значений АЧХ, соответствующих значениям элементов в векторе `opt.grid`;
- `opt.error` – вектор значений ошибок на частотах вектора `opt.grid`;
- `opt.fextr` – вектор, содержащий частоты с экстремальными ошибками АЧХ.

На рис. 5.53 изображен результат применения процедуры `cremez` для расчета параметров полосового КИХ-фильтра 30-го порядка.

```
» b = cremez(30,[0 0.5 0.6 0.8 0.9 1],'bandpass');
» freqz(b,1,512)
```

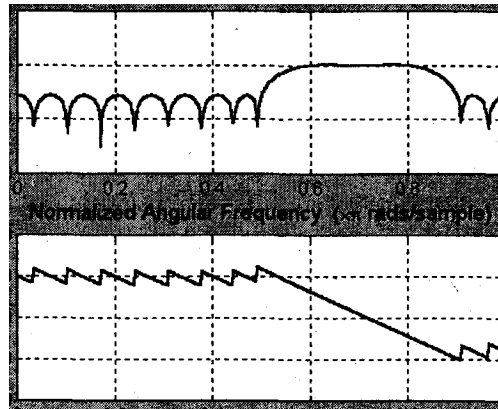


Рис. 5.53

5.5. Графические и интерактивные средства

5.5.1. Графические средства пакета *Signal*

Некоторые графические средства пакета *Signal* уже упоминались ранее. К ним относятся, прежде всего, процедуры **freqs** и **freqz**, применение которых без выходных параметров приводит к построению в графическом окне (фигуре) графиков АЧХ и ФЧХ аналогового звена по заданным векторам коэффициентов числителя и знаменателя передаточной функции по Лапласу (для первой из них) либо цифрового фильтра (звена) по коэффициентам его дискретной передаточной функции (для второй процедуры). Напомним, что общая форма вызова этих функций при выведении графиков такова:

`freqs(b,a,n)` ИЛИ `freqz(b,a)`

При этом `b` и `a` представляют собой векторы коэффициентов числителя и знаменателя передаточной функции, а `n` задает число отсчетов в строящихся АЧХ и ФЧХ.

Пример применения функции **freqs** приведен на рис. 5.16, а функции **freqz** – на рис. 5.17. Из графиков видно, что:

- первая процедура строит АЧХ в логарифмическом масштабе, а вторая – в децибелах;
- частоты в первом случае откладываются в радианах в секунду и в логарифмическом масштабе, а во втором – в виде отношения к частоте Найквиста, в равномерном масштабе и в диапазоне от 0 до 1;
- форма оформления графиков достаточно жесткая и не предусматривает возможности изменения размеров графиков надписей по осям и вывода заголовка. Некоторые процедуры расчета фильтров, такие как **fircls**, **fircls1**, **cremez** и **maxflat** предусматривают выведение соответствующих графических изображений некоторых

параметров спроектированного фильтра, если в качестве последнего входного параметра при обращении к процедуре указан флаг 'plots'.

Так, функция **maxflat** в этом случае выводит три графические зависимости:

- АЧХ в пределах до частоты Найквиста в равномерном масштабе;
- карту расположения нулей и полюсов в комплексной Z-плоскости;
- частотный график групповой задержки фильтра.

Например:

```
» [b,a,b1,b2] = maxflat (10, 2, 0.5, 'plots')
```

приводит к появлению в графическом окне изображения, показанного на рис. 5.54.

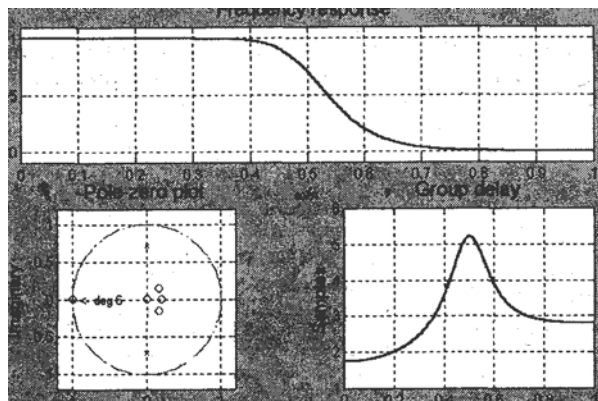


Рис. 5.54

При вызове функции **fircis** с этим флагом на график выводятся фрагменты АЧХ с максимальными отклонениями от требуемой АЧХ (рис. 5.55):

```
» n = 30;
» f = (0 0.2 0.6 0.8 1);
» amp = [1 0 0.5 0];
» up = [1.02 0.02 0.51 0.02];
» lo = [0.98 -0.02 0.49 -0.02];
» fircis(n,f,amp,up,lo,'plots')
```

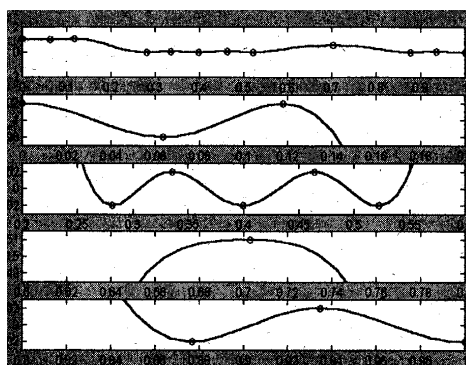


Рис. 5.55

Аналогичные графики строятся и при вызове функции **fircls1**. Отличие в том, что теперь графики не снабжены никаким текстом (рис. 5.56):

```
» fircls1(n,0.5,0.01,0.01,'plots')
```

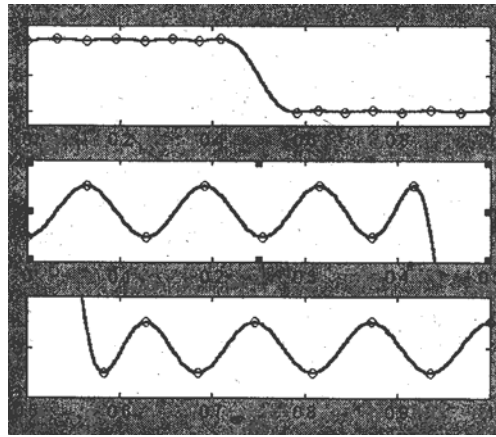



Рис. 5.56

Процедура **cremez** при таком обращении выводит следующие графики (в одном графическом окне): АЧХ, ФЧХ, зависимость погрешности по амплитуде от частоты и зависимость погрешности по фазе от частоты. Это продемонстрировано на рис. 5.57:

```
» cremez(30,[0 0.5 0.6 0.8 0.9 1],'bandpass','plots •')
```

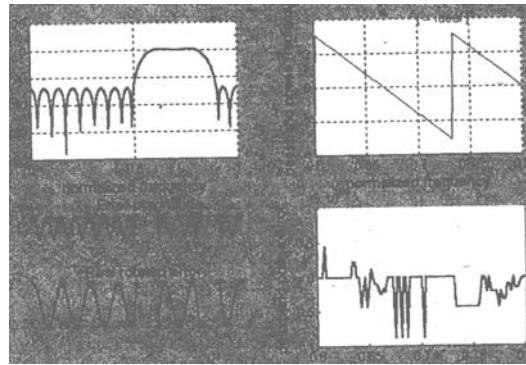


Рис. 5.57

В пакете **Signal** имеются еще три важные для инженера графические процедуры: **grpdelay**, **impz** и **zplane**. Первая строит график группового времени задержания (ГВЗ) от частоты, вторая – импульсную характеристику заданного фильтра, а третья отображает на комплексной Z-плоскости положение нулей и полюсов фильтра.

Рассмотрим в качестве примера применение этих процедур к БИХ-фильтру, созданному процедурой **maxflat**:

```
» [b,a] = maxflat(10,2,0.5) ;
» grpdelay(b,a,128)
```

Результат применения функции **grpdelay** приведен на рис. 5.58. Применяя процедуру **impz** к тому же фильтру:

```
» impz(b,a)
```

получим график импульсной дискретной характеристики фильтра, изображенный на рис. 5.59.

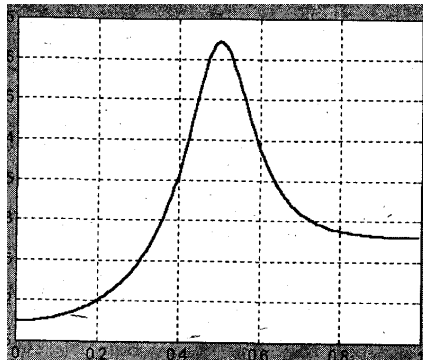


Рис. 5.58

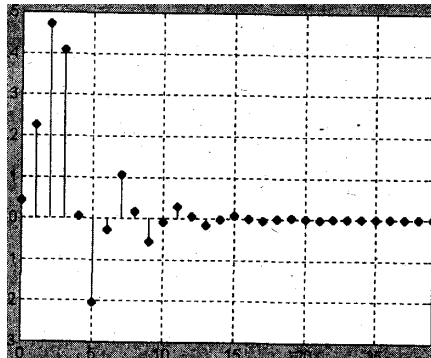


Рис. 5.59

Использование процедуры `zplane` для этого фильтра:

» `zplane(b, a)`

приводит к построению графика, показанного на рис. 5.60.

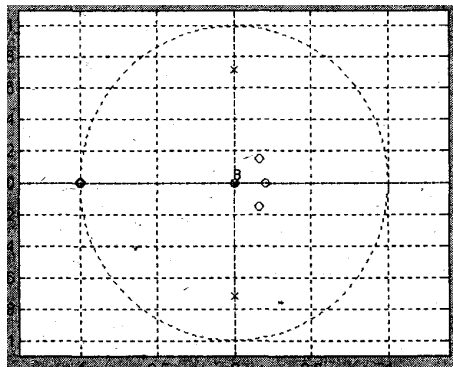


Рис. 5.60

Рассмотрим применение некоторых графических функций на примере двух коррелированных случайных процессов. Для этого вначале сформируем эти процессы:

```
Ts = 0.01; T = 100; % Задание параметров процесса t = 0 : Ts : T;
X1 = randn(1,length(t)); % Формирование белого шума
% Расчет параметров формирующего фильтра
om0 = 2*pi; d2 = 0.05; A = 1; oms = om0*Ts;
a(1) = 1+2*dz,*oms+oms^2;
a(2) = -2*(1+dz*oms) ;
a(3) = 1;
b(1) = A*oms^2;
```

```

% Формирование "профильтрованного" процесса
y1 = filter(b,a,x1) ;
% Построение графика процесса
subplot(3,1,1)
plot(t,y1), grid, set(gca,'FontName','Arial
Cyr','FontSize',8)
title('Процесс на выходе фильтра (T0 = 1; dz=0.05,
Ts=0.01)') ;
ylabel('Y1(t)')
% Расчет параметров первого звена
om0 = 2*pi*0.20; dz = 0.05; A = 1; oms = om0*Tc;
a1(1) = 1+2*dz*oms+oms^2;
a1(2) = -2*(1+dz*oms) ;
a1(3) = 1;
b1(1) = A*oms^2;
% Формирование "первого" процесса
x = filter(b1,a1,y1);
% Построение графика первого процесса
subplot(3,1,2)
plot(t,x),grid, set(gca,'FontName','Arial Cyr','FontSize' 8)
title('Первый случайный процесс (T0 = 5; dz = 0.05, Ts = 0.01)');
xlabel('Время (с)');
ylabel('X(t)')
% Расчет параметров второго звена
om0 = 2*pi*0.5; dz = 0.05; A = 1; oms = om0*Tc;
a2(1) = 1+2*dz*oms+oms^2;
a2(2) = -2*(1+dz*oms) ;
a2(3) = 1;
b2(1) = A*oms^2;
% Формирование второго процесса
y = filter(b2,a2,y1) ;
% Построение графика второго процесса
subplot(3,1,3)
plot(t,y),grid, set(gca,'FontName','Arial Cyr','FontSize', 8 )
title ('Второй случайный процесс (T0 = 2; dz = 0.05, Ts = 0.01)')
xlabel('Время (с)');
ylabel('Y(t)')

```

Графики порождающего процесса и двух процессов, производных от него, приведены на рис. 5.61.

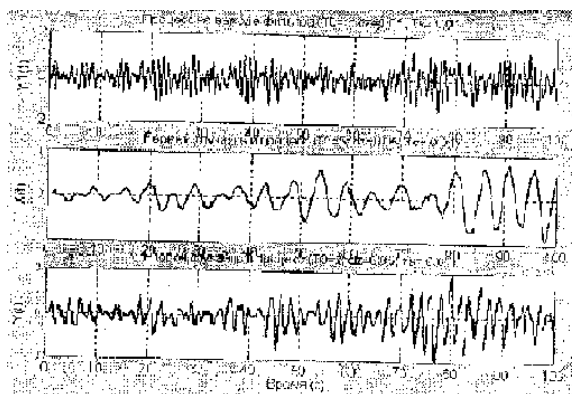


Рис.5.61

Представление графика длинного процесса в виде совокупно нескольких фрагментов меньшей длины по аргументу можно осуществить при помощи процедуры **strips** путем такого обращения к ней:

```
strips(x,sd,Fs,scale)
```

где x – вектор значений выводимой на график функции, sd параметр, задающий в секундах длину одного фрагмента аргументу, Fs – значение частоты дискретизации, $scale$ – маси по вертикальной оси.

В качестве примера выведем график порождающего случайного процесса, разбивая его на отдельные фрагменты по 20 секунд и задавая диапазон изменения значения функции в каждом фрагменте от -2 до 2 :

```
» strips(yl,20,100,2)
» grid; title ('Применение процедуры STRIPS для вывода Yl(t)') ;
» xlabel('Время, с')
```

На рис. 5.62 представлен результат.

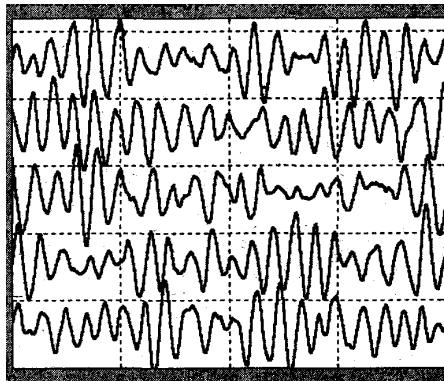


Рис. 5.62

Теперь познакомимся с графическими процедурами статистической обработки процессов. Ранее (разд. 5.3) была описана функция **psd**, которая, если не указывать выходных параметров, выводит в графическое окно график спектральной плотности мощности (рис. 5.42). Аналогичный график зависимости модуля взаимной спектральной плотности двух сигналов от частоты строит процедура **csd**, если обратиться к ней таким образом:

```
csd(x,y,nfft,Fs)
```

Здесь x и y – заданные последовательности отсчетов двух сигналов, **nfft** – число отсчетов, по которым вычисляется взаимная спектральная плотность, Fs – частота дискретизации этих сигналов.

Применим функцию **psd** к случайному сигналу $X(t)$, а процедуру **csd** – для нахождения взаимной спектральной плотности сигналов $X(t)$ и $Y(t)$. Результаты приведены соответственно на рис. 5.63 и 5.64.

```
» psd(x,10000,100); title ('Применение процедуры PSD... к процессу X(t)');
» ylabel('Спектральная плотность'); xlabel('Частота, Гц')
```

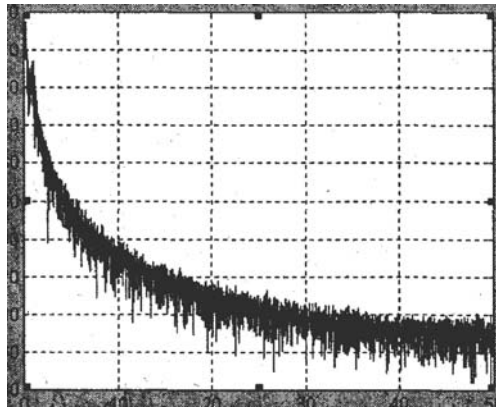


Рис. 5.63

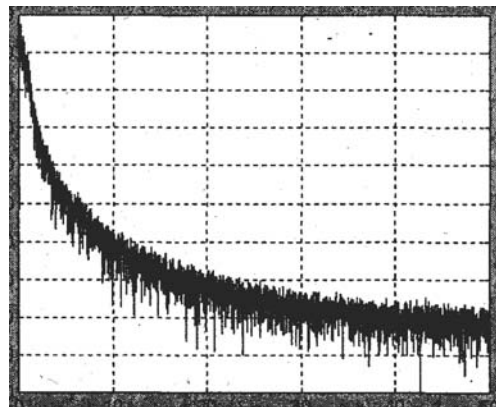


Рис. 5. 64

```

» csd(x, y, 10000,100) ; title'' Применение процедуры CSD. . .
    к процессам X(t) и Y(t)';
» ylabel('Взаимная СП'); xlabel('Частота, Гц ')

```

Процедура **cohere** при обращении:

```
cohere(x,y,nfft,Fs)
```

вычисляет и выводит график от частоты квадрата модуля функции когерентности сигналов $X(t)$ и $Y(t)$, вычисленного по $nfft$ точкам, заданным с частотой дискретизации F_s . Применяя эту процедуру к сформированным случайным процессам, получим картину, представленную на рис. 5.65:

```
» cohere (x, y, 10000, 100)
```

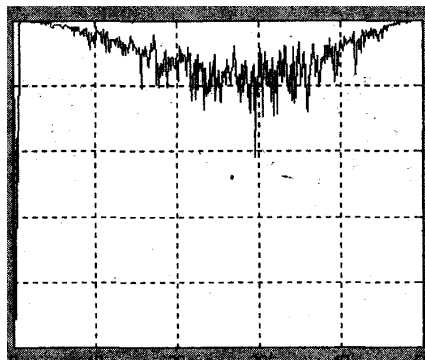


Рис. 5.65

Ознакомимся с процедурой **spectrum**, которая выполняет спектральный анализ двух процессов $X(t)$ и $Y(t)$. Обращение:

$$P = \text{spectrum}(x,y)$$

приводит к вычислению матрицы P , состоящей из восьми столбцов

$$P = [P_{xx}, P_{yy}, P_{xy}, T_{xy}, S_{xy}, P_{xxc}, P_{yyc}, P_{xyc}]$$

где P_{xx} – вектор-столбец, содержащий оценку СПМ процесса X ; P_{yy} вектор-столбец, содержащий оценку СПМ процесса Y ; P_{xy} – вектс взаимной спектральной плотности процессов X и Y ; T_{xy} – комплексная передаточная функция: $T_{xy} = P_{xy}/P_{xx}$; S_{xy} – функция когерентности, $S_{xy} = ((\text{abs}(P_{xy}))^2)/(P_{xx} * P_{yy})$; P_{xxc} , P_{yyc} , P_{xyc} – векторы, содержащие доверительные интервалы для оценок P_{xx} , P_{yy} и P_{xy} .

Если эту функцию вызвать без выходных параметров:

$$\text{spectrum}(x,y)$$

то результатом ее работы будет поочередный вывод в одно графическое окно таких графиков:

1. Зависимости СПМ первого сигнала от нормализованной частоты (рис. 5.66); на графике появляются три кривые – кривая» оценки усредненного значения СПМ на фиксированной частоте| и две кривые с добавлением и вычитанием доверительного интервала на этой частоте.

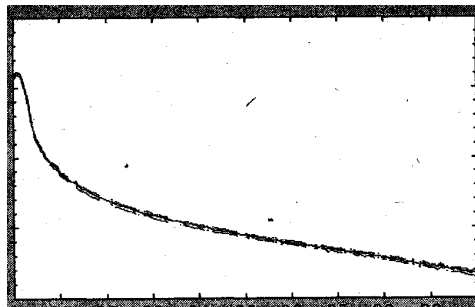


Рис. 5.66

2. После нажатия клавиши [Enter] прежние кривые исчезнут и на том же поле появятся три аналогичные кривые (рис. 5.67) второго процесса $Y(t)$.

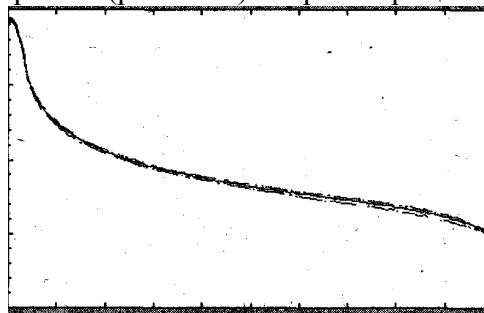


Рис. 5.67

3. Следующее нажатие [Enter] приведет (рис. 5.68) к появлению кривой зависимости модуля передаточной функции взаимной спектральной плотности указанных процессов от частоты.

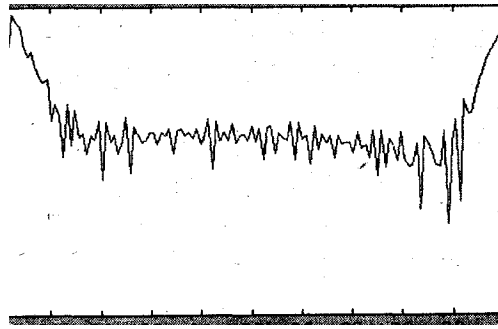


Рис. 5.68

4. Еще одно нажатие [Enter] приводит к появлению графика зависимости аргумента передаточной функции ВСП от частоты (рис. 5.69).

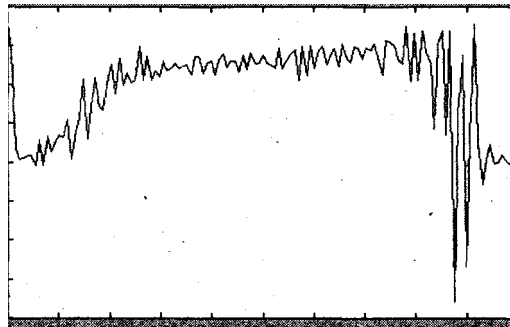


Рис. 5.69

5. Последнее нажатие [Enter] вызовет появление в поле графика функции когерентности (рис. 5.70).

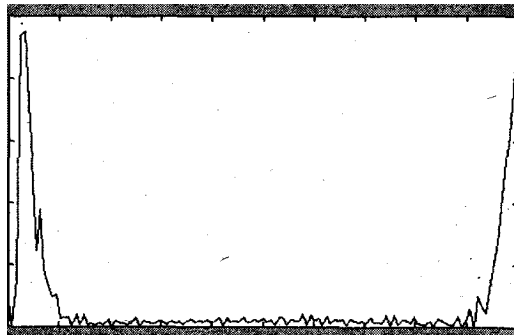


Рис. 5.70

Для построения спектрограммы процесса в MatLAB предусмотрена процедура **specgram**.

Примечание. Спектрограммой называется зависимость амплитуды вычисленного в окне ДПФ (дискретного преобразования Фурье) от момента времени, определяющего положение этого окна.

Для примера применим эту процедуру к сформированному выше процессу $X(t)$:

» `specgram(x,10000,100)`

В результате получаем в графическом окне картину, изображенную на рис. 5.71.



Рис. 5.71

Общий вид обращения к процедуре `specgram` напоминает к процедуре `psd`:

`specgram(x,nfft,Fs)`

где x – вектор процесса, спектрограмма которого вычисляется, **nfft** – количество точек этого процесса, участвующих в вычислениях и F_s – частота дискретизации процесса. Наконец, графическое представление имеет и процедура **tfe**, которая оценивает параметры и строит график АЧХ передаточной функции звена, на вход которого подан процесс, представленный первым вектором в обращении к процедуре, а на выходе получен процесс, представленный вторым вектором. В целом, чтобы получить график АЧХ, нужно обратиться к процедуре таким образом:

`tfe(x,y,nfft,Fs)`

где x – вектор значений Входного процесса, y – вектор выходного процесса, **nfft** – количество обрабатываемых точек (элементов указанных векторов), F_s – частота дискретизации.

Применяя процедуру к ранее сформированным процессам $X(t)$ и $Y(t)$:

» `tfe(x,y,10000,100)`

получим график, изображенный на рис. 5.72.

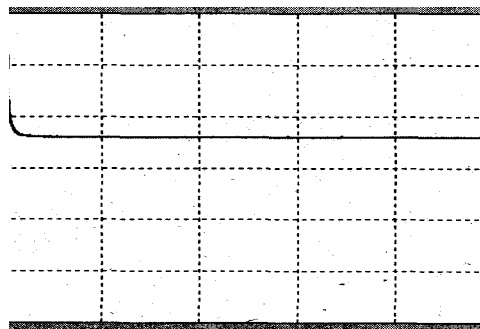


Рис. 5.72

5.5.2. Интерактивная оболочка *SFTool*

Процедура `SPTool` активизирует графическую интерактивную оболочку пакета `Signal`, включающую:

- средство поиска и просмотра сигналов – `Signal Browser`;
- проектировщик фильтров – `Filter Designer`;
- средство просмотра характеристик фильтров – `Filter Viewer`;
- средство просмотра спектра – `Spectrum Viewer`.

Оболочка активизируется путем ввода в командном окне `MatLAB` команды **sptool**. В результате на экране появляется окно, представленное на рис. 5.73.

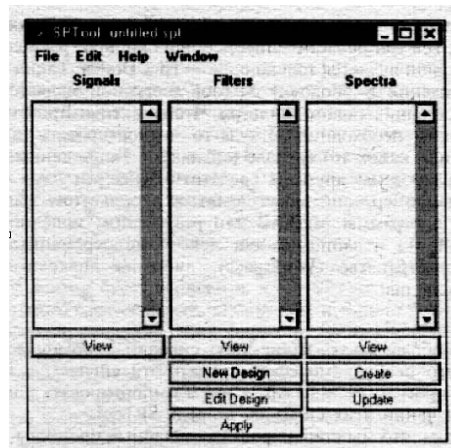


Рис. 5.73

Как видим, окно SPTool состоит из трех областей – **Signals** (Сигналы), **Faters** (Фильтры) и **Spectra** (Спектры), под каждой из которых имеются кнопки, указывающие на то, что можно сделать с объектами, расположенными в этих областях. Так, под областью **Signals** находится лишь кнопка **View**. Это означает, что объекты (сигналы), имена которых расположены в этой области, могут быть только просмотрены. Под областью **Filters** находятся четыре кнопки, которые указывают на то, что объекты (фильтры), имена которых размещаются внутри него, могут быть:

- созданы (кнопка New Design);
- отредактированы (кнопка Edit Design);
- просмотрены (кнопка View);
- применены к одному или нескольким объектам, выделенным в области Signals (Apply).

Аналогично, с объектами области Spectra (спектрами) можно производить такие действия:

- создавать (кнопка Create);
- просматривать (кнопка View);
- обновлять, т.е. создавать заново под тем же именем (кнопка Update).

Внутри областей обычно размещаются имена (идентификаторы) соответствующих переменных или процедур, входящих в открытый в SPTool файл с расширением .spt (имя этого файла указывается в заголовке окна SPTool).

При первом обращении в заголовке окна находится имя **untitled.spt**, все три области – пустые, а из кнопок, расположенных ниже, активной является только одна – **New Design**. Таким образом, после вхождения в оболочку **SPTool** доступной является только операция создания нового фильтра. Чтобы активизировать остальные кнопки, необходимо откуда-то импортировать данные о каком-то (или каких-то) сигнале (сигналах). Такие данные должны быть сформированы другими средствами, нежели сама оболочка-**SPTool**, (например, они могут являться результатом выполнения некоторой программы MatLAB или результатом моделирования в среде SimuLink) и записаны как некоторые переменные либо в рабочем пространстве (Workspace), либо на диске в файле с расширением .mat.

Импорт сигналов

Чтобы обрабатывать какие-либо сигналы с помощью **SPTool**, прежде всего необходимо сформировать эти сигналы с помощью некоторой программы MatLAB, а затем импортировать полученные векторы значений этих сигналов в среду **SPTool**.

Допустим, что мы сгенерировали случайные процессы $X(t)$, $Y(t)$; и $Yl(t)$ в соответствии с программой, приведенной в разд. 5.5.1. В результате в рабочем пространстве MatLAB появились векторы: x , y и $y1$, каждый из которых содержит по 10000 элементов.

Импортируем их в среду **SPTool**.

Войдя в среду **SPTool** (рис. 5.73), выберем из меню **File** (Файл) команду **Import** (Импорт). После этого откроется окно **Import to SPTool**, представленное на рис. 5.75.

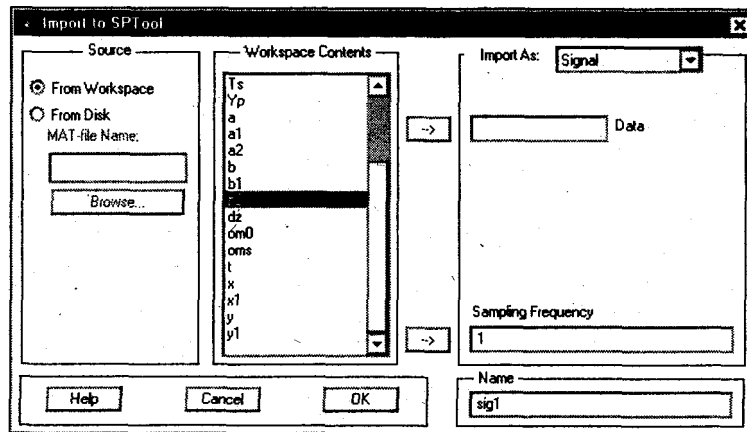


Рис. 5.75

В области **Source** (Источник) этого окна выбран переключатель **From Workspace** (Из рабочего пространства). Поэтому все имена переменных рабочего пространства представлены во второй области – **Workspace Contents** (Содержимое рабочего пространства). Выбрав при помощи мыши необходимую переменную, следует нажать кнопку со стрелкой, указывающей на поле ввода **Data**. После этого в поле ввода **Data** должно появиться имя выбранной переменной.

Затем в поле **Sampling Frequency** (Частота дискретизации) нужно ввести желаемое значение частоты дискретизации. Фактически этим параметром задается временной промежуток T_s между отдельными значениями выбранного, вектора процесса.

В поле ввода **Name** (Имя) необходимо указать имя, под которым введенный вектор будет записан в среде **SPTool**.

После этого следует нажать кнопку **OK**, и импорт сигнала в среду **SPTool** будет произведен. Окно **Import to SPTool** исчезнет, а окно **SPTool** изменит свой вид (рис. 5.77): в области **Signals** появится запись имени вектора сигнала, и кнопка **View** под этой областью станет доступной. Кроме того, станет доступной кнопка **Create** под областью **Spectra**. Это означает, что можно находить спектральные - характеристики импортированного сигнала.

Повторяя эти действия, можно перенести в **SPTool** и другие сигналы (X и Y).

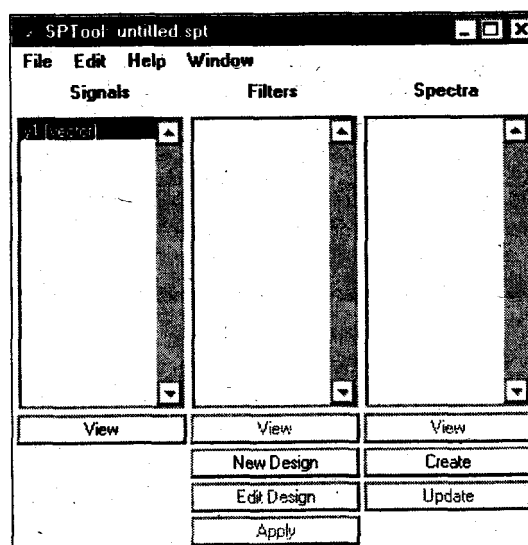


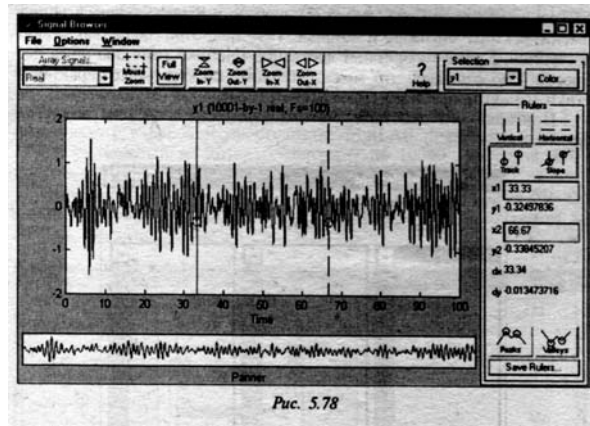
Рис. 5.77

Если векторы процессов записаны в MAT-файл, то для импорта нужно после вызова окна **Import to SPTool** выбрать в я переключатель **From disk**. В результате станут доступными поле ввода MAT-file **Name** и кнопка **Browse** (рис 5.75). Введя имя необходимого MAT-файла или отыскав MAT-файл при помощи кнопки **Browse**, вновь вызываем в окно **SPTool** его содержимое. Последующие действия аналогичны ранее рассмотренным.

Просмотр сигналов

После импорта вектора сигнала можно воспользоваться ствами его просмотра. Для этого достаточно выделить в обла **Signals** нужные сигналы и нажать на кнопку **View** под областью. В результате должно появиться окно **Signal Browser**.

В нашем случае, выбрав сигнал y_1 , получим окно, изображенное на рис. 5.78.



Как видим, центральную часть окна занимает изображение кривых зависимости выделенных процессов от времени. В заголовке графика указаны имена сигналов, изображенных на графике размерность соответствующих векторов и частота дискретизации.

В верхнем правом углу окна расположена область **Selection**, с помощью которой можно изменить цвета кривых, отображаемых в окне графиков.

Справа от графического поля окна (в области **Rulers**) размещаются инструменты, обеспечивающие точный отсчет показаний значений аргумента и процесса в двух точках графика.

Эти точки графика определяются пересечением графика процесса, имя которого стоит в списке **Selection**, с двумя вертикальными линиями розового цвета, расположенными в поле графиков. Изменение положения этих линий по шкале времени выполняется с помощью мыши. Подведите курсор к одной из вертикальных линий, и он примет вид руки. Нажмите левую кнопку мыши и, не отпуская ее, переместите курсор вправо или влево. При этом в области **Rulers** будут отображаться координаты X и Y точек пересечения обеих вертикальных прямых с графиком процесса, а также разности между ними. Координата X соответствует времени, а Y — значению процесса.

На рис. 5.79 отображены все три процесса. Если нажать кнопку **Slope**, в поле графика появится еще одна прямая, соединяющая указанные ранее точки пересечения выбранной кривой с вертикальными линиями, а в области **Rulers** отобразится новое число m , которое представляет собой тангенс угла наклона этой прямой к оси времени.

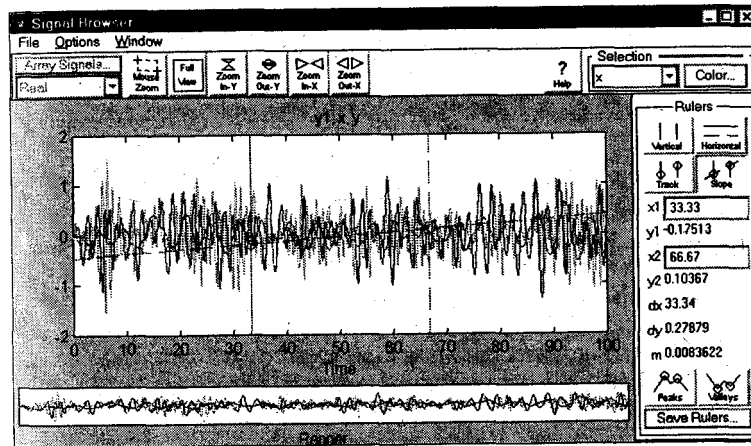


Рис. 5.79

В верхней части окна располагаются средства управления окном и масштабами внутри графического поля.

Создание спектров сигналов

После ввода сигналов в **SPTool** можно найти оценки спектральных свойств этих сигналов. Для этого достаточно в области сигналов окна **SPTool** отметить сигнал, оценку спектральной плотности которого вы хотите получить, и нажать на кнопку **Create** в нижней части окна. После этого на экране появится окно **Spectrum Viewer** (рис. 5.80).

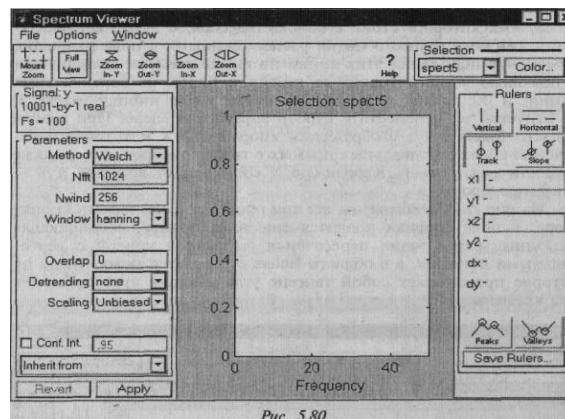


Рис. 5.80

Оно напоминает окно **Signal Browser**. Верхняя и правая части этих окон практически одинаковы. Однако графическая область окна **Spectrum Viewer** является пустой, а слева от нее располагается область, элементы которой позволяют:

- выбрать метод нахождения спектральной характеристики сигнала;
- установить количество обрабатываемых точек сигнала;
- установить количество точек сглаживающего окна;
- выбрать тип окна сглаживания;
- установить размер перекрытия окон;
- установить метод исключения тренда;
- установить метод масштабирования графика.

Метод вычисления спектра выбирается при помощи списка **Method**, который содержит следующие элементы: Burg, FFT, MEM, MTM, MUSIC, Welch, YuleAR. Каждому из них соответствует метод (процедура) вычисления спектра сигнала.

Для проведения вычислений после выбора метода следует нажать кнопку **Apply** ниже левого поля. Например, выделим для обработки процесс Y1, нажмем кнопку **Create** и выберем метод FFT. После нажатия кнопки **Apply** окно **Spectrum Viewer** примет вид, представленный на рис. 5.82.

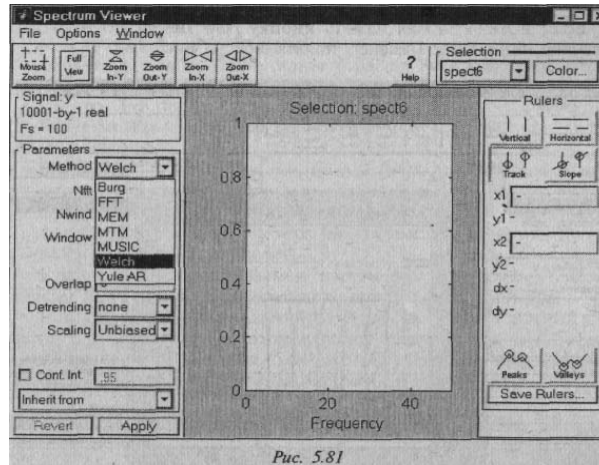


Рис. 5.81

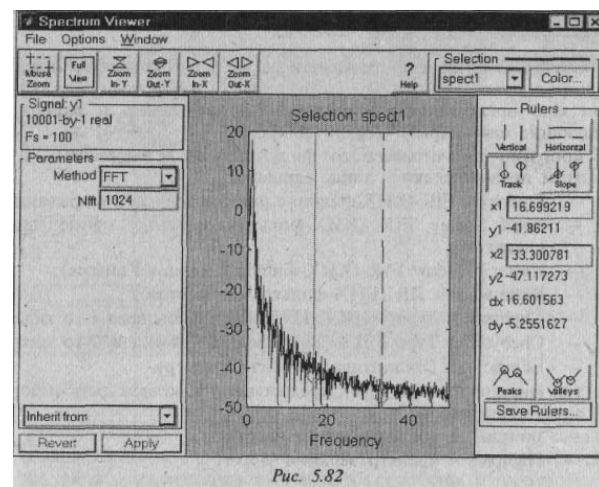


Рис. 5.82

Проектирование фильтра

Если в окне **SPTool** нажать кнопку **New Design**, то на экране появится окно **Filter Designer**, показанное на рис. 5.83.

Это окно позволяет произвести расчет коэффициентов нового фильтра и затем записать эти коэффициенты в объект - фильтр. При этом оно предоставляет возможность устанавливать и изменять следующие параметры будущего фильтра:

- прототип рассчитываемого фильтра (список **Algorithm**); при этом предоставляются такие варианты:
 - Equiripple FIR (КИХ-фильтр с равноотстоящими разрывами);
 - Least Square FIR (КИХ-фильтр по методу наименьших квадратов);
 - Kaizer Window FIR (КИХ-фильтр с окном Кайзера);
 - Butterworth IIR (БИХ-фильтр Баттерворта);
 - Chebyshev Type I IIR (БИХ-фильтр Чебышева 1-го типа);
 - Chebyshev Type 2 IIR (БИХ-фильтр Чебышева 2-го типа);
 - Elliptic IIR (Эллиптический БИХ-фильтр).
- тип фильтра (список **Type**); предоставляется возможность выбс следующих типов:
 - Lowpass – фильтр нижних частот;
 - Highpass – фильтр верхних частот;
 - Bandpass – полосовой фильтр;
 - Bandstop – режекторный фильтр.
- параметры полосы пропускания (область **Passband**); здесь можно установить, например, (для фильтра нижних частот) граничную частоту F_p полосы пропускания и

максимально допустимое значение R_p подавления амплитуд внутри полосы пропускания (в децибелах);

- параметры полосы задерживания (область Stopband); здесь можно установить, например, (для фильтра нижних частот) граничную частоту F_s полосы задерживания и минимально допустимое значение R_s подавления амплитуд внутри полосы задерживания (в децибелах).

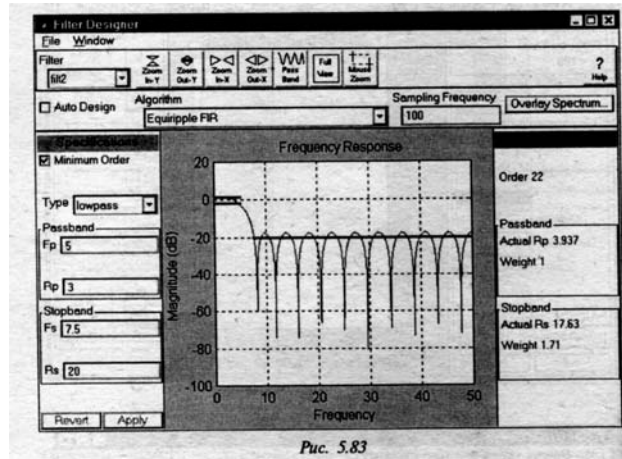


Рис. 5.83

Количество устанавливаемых параметров и их смысл автоматически изменяются при переходе к другому типу фильтра.

Например, установив алгоритм фильтра Баттерворта нижних частот с граничными частотами полос пропускания в 0.5 Гц и задерживания в 0.7 Гц (рис. 5.84) и нажав кнопку **Apply**, получим параметры такого фильтра и запишем их в объект **filt4**.

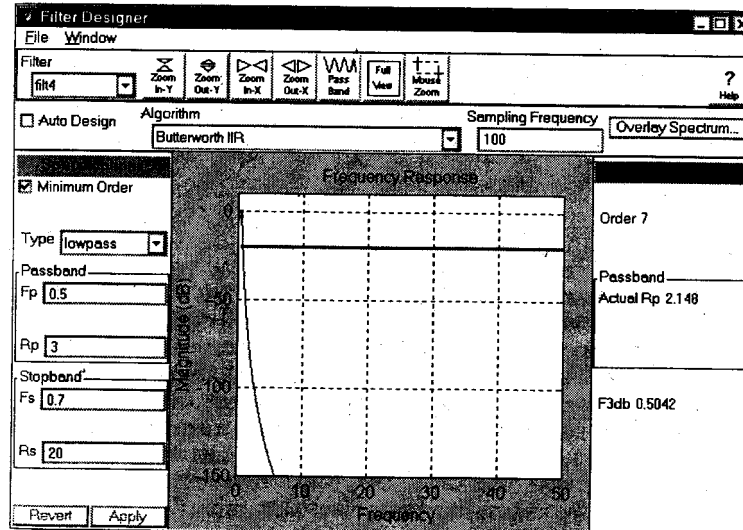


Рис. 5.84

Просмотр свойств фильтра

После создания фильтра можно просмотреть графики различных характеристик спроектированного и записанного фильтра. Для этого достаточно выделить имя фильтра, свойства которого нужно посмотреть, в области **Filters** окна **SPTool**, а затем нажать кнопку **View** под этой областью.

Например, для только что созданного фильтра **filt4** мы получим окно **Filter Viewer**, показанное на рис. 5.85.

Пример вывода всех доступных графиков одновременно доказан на рис. 5.86.

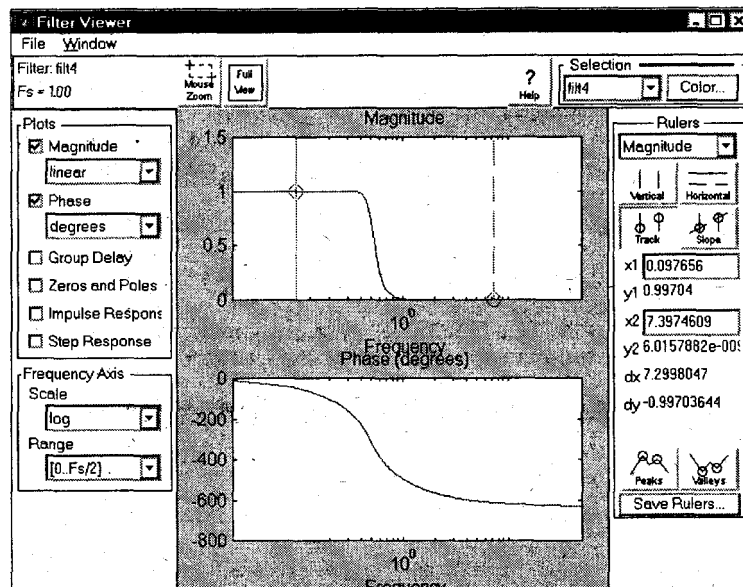


Рис. 5.85

Как видим, в окне выведены графики АЧХ и ФЧХ фильтра. В число средств просмотра фильтров входят (см. левую область окна Filter Viewer):

- возможность вывода на экран одновременно любого сочетания таких графиков: АЧХ, ФЧХ, частотной зависимости группового времени задержки, графического представления расположения нулей и полюсов дискретной передаточной функции в Z-плоскости, графика временного отклика фильтра на импульсное единичное воздействие и графика отклика на ступенчатое единичное воздействие; для этого надо отметить галочкой с помощью мыши нужные виды графиков в области **Plots** (Графики) окна;
- возможность изменить вид шкалы как по оси частот, так и по оси амплитуд, установить диапазон представления графиков по частоте и изменить единицы представления фазового сдвига (области **Plots** и **Frequency Axis**).

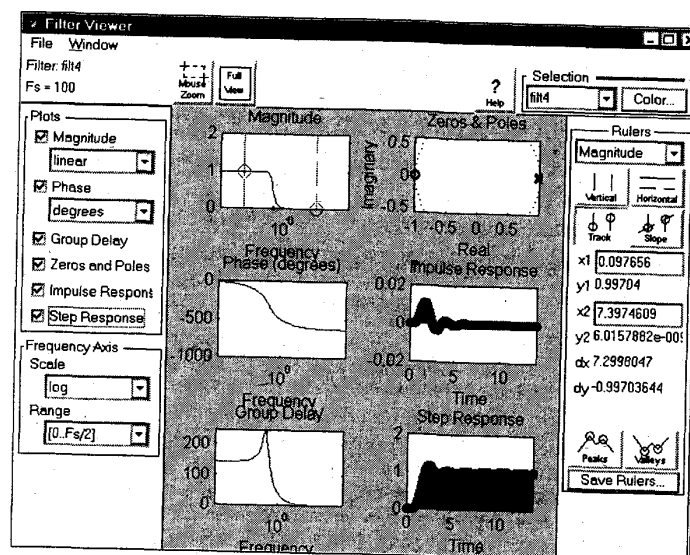


Рис. 5.86

Применение разработанного фильтра для фильтрации

Использовать в среде **SPTool** разработанный фильтр чрезвычайно просто. Для этого в области **Signals** окна **SPTool** нужно выделить имя сигнала, который нужно преобразовать с помощью фильтра, в области **Filters** – имя фильтра, с помощью которого надо преобразовать этот сигнал, и нажать кнопку **Apply**. В результате в первой области

(**Signals**) появится имя нового сигнала, начинающееся с сочетания **sig** с последующим порядковым номером.

Полученный сигнал можно просмотреть, как это было описано ранее, используя команду **View**.

Например, применяя только что разработанный фильтр **filt4** к процессу $Y_l(t)$, получим процесс, изображенный на рис. 5.87.

Спектральные характеристики полученного процесса можно изучить, применяя область **Spectra**, как это было описано.

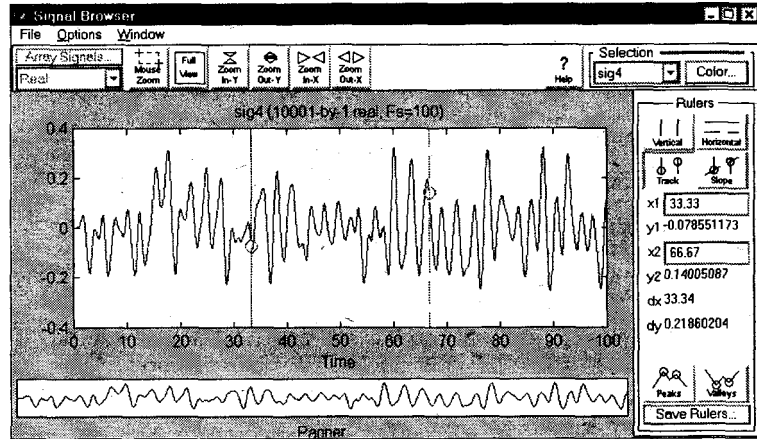


Рис. 5.87

Вторичное использование результатов SPTool

При завершении сеанса работы с **SPTool** система запрашивает, нужно ли записать полученные результаты на диск. В случае положительного ответа она сохраняет все данные в файле с расширением **.spt**. Кроме того, в меню **File** окна **SPTool** предусмотрены команды записи в файл (рис. 5.74) – **Save Session** и **Save Session As**.

При повторном запуске **SPTool** можно воспользоваться результатами такого сохранения, используя команду **Open Session** и выбирая один из записанных SPT-файлов.