МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНА МЕТАЛУРГІЙНА АКАДЕМІЯ УКРАЇНИ

РОБОЧА ПРОГРАМА,
методичні вказівки та індивідуальні завдання до вивчення дисципліни «Цифрова обробка експериментальних даних» для студентів спеціальності 122-комп'ютерні науки

Дніпро НМетАУ 2019

УДК 681.3.06+519.68

Методичні вказівки до виконання лабораторних робіт з дисципліні «Цифрова обробка експериментальних даних». Для студентів напряму 0501 01 – «Комп'ютерні науки» / Укл.: О.І. Михальов, В.В. Гнатушенко, В.В. Гнатушенко. Під ред. О.І. Михальова. – Дніпро: НМетАУ, 2019. – 76 с.

Методичні вказівки є практичною частиною комплексу навчально-методичних матеріалів з дисципліні «Цифрова обробка експериментальних даних», в якій наведені теоретичні матеріали та розглянуті приклади по застосуванню середовища MATLAB для цифрової обробки даних. Розглянуті загальні принципи і математичні моделі перетворення сигналів при цифровій обробці, базові алгоритми цифрової фільтрації сигналів, методи синтезу, програмної реалізації і моделювання цифрових фільтрів, оцінки і забезпечення їх точності. Також розглянуті методи спектрального аналізу сигналів на основі дискретного перетворення Фур'є та математичного апарату вейвлетів, алгоритми швидкого перетворення Фур'є, методи формування і обробки дискретних сигналів, а також реалізації систем цифрової обробки сигналів на основі апаратних і апаратно-програмних засобів.

Методичні вказівки призначені для студентів напряму підготовки 122 – «Комп'ютерні науки», а також для слухачів курсів підвищення кваліфікації, студентів і аспірантів інших спеціальностей.

Укладачі: О.І. Михальов, доктор технічних наук, професор,
Вікторія В. Гнатушенко, доктор технічних наук, доцент,
Володимир В. Гнатушенко, доктор технічних наук, професор,

Відповідальний за випуск: О.І. Михальов, д-р техн. наук, проф.
Рецензент: В.І. Корсун, доктор технічних наук, професор

Друкується за авторською редакцією.
Затверджено на засіданні кафедри інформаційних технологій і систем, протокол № 9 від 06.03.2019.

Підписано до друку 10.05.2019. Формат 60x84 1/16. Папір типогр.
Друк різограф. Облік.-вид. арк 4,75. Умов. друк. арк. 3,40.
Тираж 100 пр. Замовл. № 19/12.

ДНВП «Системні технології»

СОДЕРЖАНИЕ

Лабораторная работа №1
ФОРМИРОВАНИЕ И ИССЛЕДОВАНИЕ ТИПОВЫХ СИГНАЛОВ

Цель: Изучить основные возможности пакета MATLAB для формирования типовых сигналов, используя SIGNAL PROCESSING TOOLBOX & SIMULINK.

1.1 Формирование типовых сигналов

RECTPULS Прямоугольный импульс

Синтаксис: y = rectpuls(t)

y = rectpuls(t, w)

Описание:

Функция y = rectpuls(t) формирует прямоугольный импульс единичной амплитуды для заданной в векторе t последовательности отсчетов времени. Генерируется импульс с шириной 1, центрированный относительно t = 0. В векторе y формируется часть импульса, соответствующая последовательности отсчетов, заданной в векторе t.

Функция y = rectpuls(t, w) формирует импульс ширины w.

Сопутствующие функции: CHIRP, COS, DIRIC, GAUSPULS, PULSTRAN, SAWTOOTH, SIN, SINC, SQUARE, TRIPULS.

SQUARE Последовательность прямоугольных импульсов

Синтаксис: y = square(t)

y = square(t, duty)

Описание:

Функция y=square(t) формирует последовательность прямоугольных импульсов с периодом 2π, для заданной в векторе t последовательности отсчетов времени. Генерируемая последовательность отличается от синусоиды с периодом 2π только тем, что представляет собой прямоугольные импульсы с амплитудой ±1.

Функция y= square(t, duty) формирует последовательность прямоугольных импульсов с заданной продолжительностью положительной полуволны, которая определяется параметром duty, в процентах от периода.

Сопутствующие функции: CHIRP, COS, DIRIC, GAUSPULS, PULSTRAN, REC7PULS, SAWTOOTH, SIN, SINC, TRIPULS.

TRIPULS Треугольный импульс

Синтаксис:

y = tripuls(t)

y = tripuls(t, w)

y = tripuls(t, w, s)

Описание:

Функция y = tripuls(t) формирует симметричный треугольный импульс единичной амплитуды для заданной в векторе t последовательности отсчетов времени. Генерируется импульс с шириной 1, центрированный относительно t=0. В векторе y формируется часть импульса, соответствующая последовательности отсчетов, заданной в векторе t.

Функция y = tripuls(t, w) формирует треугольный импульс ширины w.

Функция y=tripuls(t, w, s) формирует треугольный импульс, наклон которого определяется параметром s, где -1 < s <1. Для симметричного импульса s = 0.

Сопутствующие функции: CHIRP, COS, DIRIC, GAUSPULS, PULSTRAN, RECTPULS, SAWTOOTH, SIN, SINC, SQUARE.

SAWTOOTH Генератор пилообразных и треугольных сигналов

Синтаксис:

x = sawtooth(t)

x = sawtooth(t, 'width')

Описание:

Функция x = sawtooth(t) формирует пилообразный сигнал с амплитудой ±1 и периодом 2π. Последовательность значений аргумента задается в векторе t. При t = 2π x = - 1 .

Функция x = sawtooth(t, width) формирует модифицированный пилообразный сигнал. Параметр width задается в диапазоне от 0 до 1 и определяет часть периода, в которой возрастает сигнал. Сигнал возрастает от -1 до 1 на интервале от 0 до 2π*width, а затем убывает от 1 до -1 на интервале от 2π*width до 2π. Если width=0.5, то формируется симметричная волна. Функция sawtooth(t,l) эквивалентна функции sawtooth(t).

Диагностика:

Если параметр width не является скаляром, выводится сообщение:
Requires WIDTH parameter to be a scalar (Параметр width должен быть скаляром).

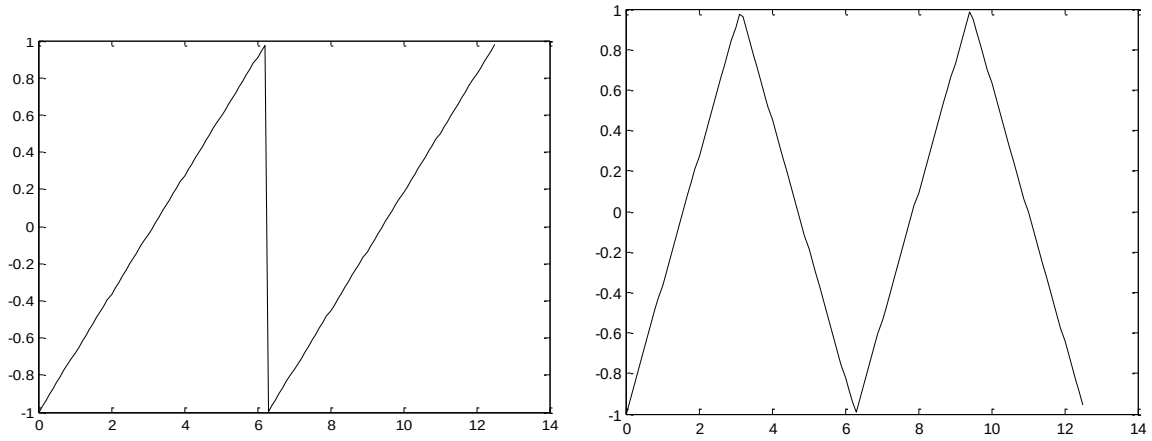
Пример:

Сгенерируем и выведем на экран треугольную последовательность, заданную в диапазоне от 0 до 4π с параметром width = 1 (рисунок 1.1, а):

```
t = (0:0.1 :4*pi);
x = sawtooth(t, 1);
plot(t, x)
```

и с параметром width = 0.5 (рис. 1.1,б):

```
t = (0 : 0.1 : 4*pi);
x = sawtooth(t, 0.5);
plot(t, x)
```



а) width = 1 б) width = 0.5

Рисунок 1.1

Сопутствующие функции: CHIRP, COS, DIRIC, GAUSPULS, PULSTRAN, RECTPULS, SIN, SINC, SQUARE, TRIPULS.

CHIRP Косинусоида с переменной частотой

Синтаксис:

```
y = chirp(t, f0, t1,f1)
y =  chirp(t, f0, t1, f1, 'method')
y =  chirp(t, f0, t1, f1, 'method', phi)
```

Описание:

Функция y = chirp(t, f0, tl, fl) формирует выборку из косинусоидального сигнала с линейно меняющейся частотой для моментов времени, определенных в векторе t; f0 - мгновенная частота в момент времени 0, fl - мгновенная частота в момент времени t1; f0 и f1 задаются в герцах. По умолчанию f0= 0, tl=1, fl=100.

Функция y = chirp(t, f0, tl, fl, 'method') позволяет задать закон изменения частоты, определяющийся значением параметра 'method'. Если параметр method = linear, то закон изменения частоты определяется выражением $f_j(t) = f_0 + \beta t$, где $\beta = (f_1 - f_0)/t_1$; если параметр method = quadratic, то закон изменения частоты определяется выражением $f_j(t) = f_0 + \beta t^2$, где $\beta = (f_1 - f_0)/t_1$; если параметр method = logarithmic, то закон изменения частоты определяется выражением $f_j(t) = f_0 + 10^{\beta t}$, где $\beta = [\log_{10}(f_1 - f_0)]/t_1$. В этом случае fl должно быть больше f0.

Функция y = chirp(t, f0, tl, fl, 'method', phi) позволяет задать начальную фазу в параметре phi (град.). По умолчанию phi = 0.

Функция y = chirp(t, f0, tl, fl, 'method') позволяет задать закон изменения частоты, определяющийся значением параметра 'method'. Принимаемые по умолчанию значения подставляются в том случае, если переменная отсутствует или задано пустое значение. Звук такого сигнала напоминает визг – откуда и его английское название.

Пример:

Построим спектр сигнала с линейно изменяющейся частотой (рисунок 1.2).

```
t = 0 : 0.001 : 2; %Время наблюдения 2 с, частота дискретизации 1кГц.
y = chirp(t, 0,1, 150); % Начальное значение частоты сигнала 0 Гц при t = 0, конечное -150 Гц при t=1.
specgram (y, 256, 1e3, 256, 250)
```

Сопутствующие функции: COS, DIRIC, GAUSPULS, PULSTRAN, RECTPULS, SIN, SINC, SQUARE, TRIPULS.

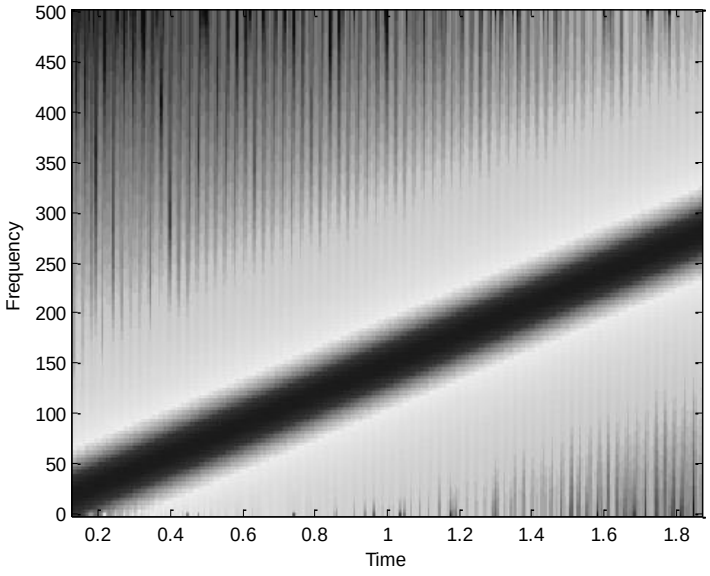


Рисунок 1.2

GAUSPULS Синусоида, модулированная функцией Гаусса

Синтаксис:

yi = gauspuls(t, fc, bw)

yi = gauspuls(t, fc, bw, bwr)

[yi, yq] = gauspuls(...)

[yi, yq, ye] = gauspuls(...)

tc = gauspuls('cutoff, fc, bw, bwr, tpe)

Описание:

Группа функций gauspuls формирует синусоидальный сигнал, модулированный функцией Гаусса.

Функция yi = gauspuls(t, fc, bw) возвращает последовательность отсчетов сигнала, вычисленных в моменты времени, заданные в векторе t; fc определяет частоту синусоиды, а bw - ширину полосы частот сигнала. По умолчанию fc = 1000 Гц, а bw = 0.5.

Функция yi = gauspuls(t, fc, bw, bwr) возвращает последовательность отпетов сигнала, имеющего следующие параметры: амплитуда - 1, ширина полосы частот - 100bw, границы полосы частот определяются уровнем атухания bwr дБ по отношению к нормализованной амплитуде сигнала. Параметр bwr должен быть отрицательным. По умолчанию bwr = -6 дБ. Функции [yi, yq] = gauspuls(...) возвращают два вектора: yi и yq. Вектор yi содержит отсчеты исходного сигнала, а вектор yq - отсчеты сигнала, синусоида в котором сдвинута по фазе на 90°.

Функции [yi, yq, ye] = gauspuls(...) дополнительно возвращают огибающую сигнала.

Функция tc = gauspuls('cutoff', fc, bw, bwr, tpe) вычисляет время отсечения tc (большее или равное 0); tc соответствует моменту времени, в который амплитуда огибающей сигнала снижается до tpe дБ. Величина параметра tpe должна быть отрицательной, поскольку он определяет относительный уровень сигнала, меньший, чем пиковое значение (равное 1). По умолчанию tpe = - 60 дБ.

Пример:

Преобразуем и выведем на экран синусоиду, модулированную функцией Гаусса со следующими параметрами:

- частота синусоиды - 50 кГц;
- ширина полосы-60%;
- частота дискретизации - 1 мГц.

Сигнал заканчивается, когда амплитуда огибающей снизится до 40 дБ.

```
tc = gauspuls('cutoff', 50e3, 0.6, [ ], -40);
t = -tc: 1e-6: tc;
i = gauspuls(t, 50e3, 0.6);
plot(t, i);
```

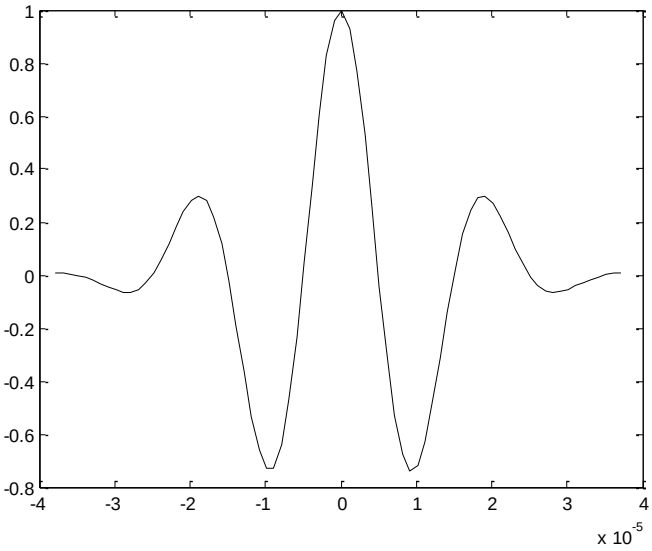


Рисунок 1.3

Сопутствующие функции: CHRP, COS, DIRIC, PULSTRAN, RECTPULS, SAWTOOTH, SIN, SINC, SQUARE, TRIPULS.

SINC **Функция sinc**

Синтаксис: y = sinc(x)

Описание:

Функция y = sinc(x) формирует последовательность значений функции sinc

$$\mathrm{sinc}(x) = \begin{cases} 1, t = 0 \\ \dfrac{\sin(\pi t)}{\pi t}, t \neq 0, \end{cases}$$

которая представляет собой обратное преобразование Фурье прямоугольного импульса ширины 2π и высоты 1:

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

Последовательность отсчетов аргумента задается в массиве x. Последовательность значений функции sinc формируется в векторе y. Количество элементов вектора y равно количеству элементов вектора x.

Известно, что любая функция g(t), частотный спектр которой ограничен диапазоном $\omega \in [-\pi, \pi]$, может быть представлена в виде суммы бесконечного, но счетного количества функций sinc. Из этого следует, что такая функция может быть полностью восстановлена по последовательности ее отсчетов, в общем случае бесконечной:

$$g(t) = \sum_{n=-\infty}^{\infty} g(n)\sin c(t-n)$$

Пример:

Выполним идеальную интерполяцию функции по заданной последовательности отсчетов (рисунок 1.4). За пределами заданного интервала функция равна 0. Частота дискретизации равна частоте Найквиста.

```
t = (1:10)';
x = randn(size(t));
ts = linspace(-5,15, 600)';
y = sinc(ts(:, ones(size(t))) -t(:, ones(size(ts))))'*x;
plot(t, x, 'o', ts, y)
```

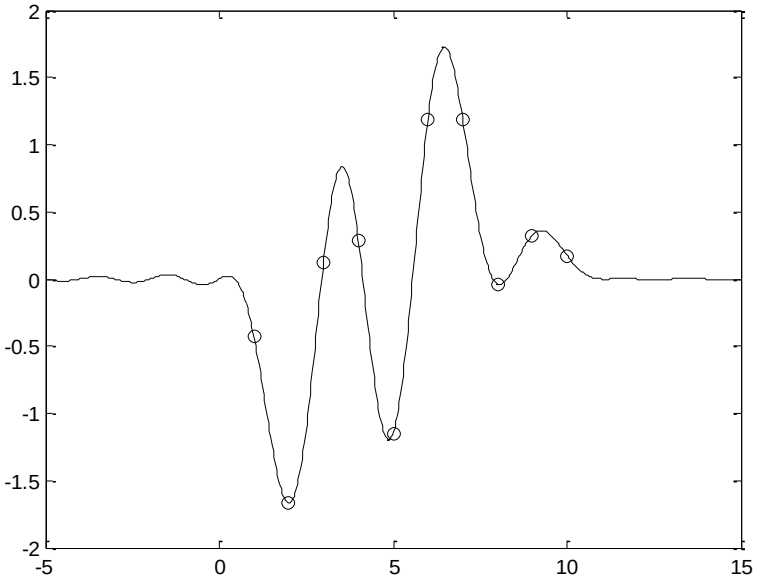


Рисунок 1.4

Сопутствующие функции: CHIRP, COS, DIRIC, GAUSPULS, PULSTRAN, RECTPULS, SAWTOOTH, SIN, SQUARE, TRIPULS.

PULSTRAN **Генератор импульсов**

Синтаксис:

y = pulstran(t, d, 'func')

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

y = pulstran(t, d, p, Fs)

y = pulstran(t, d, p)

Описание:

Группа функций pulstran генерирует последовательность импульсов. Форма импульса может быть задана функцией или определена набором отсчетов.

Функция y = pulstran(t, d, 'func') позволяет в параметре 'func' задать форму импульса. Параметр 'func' принимает одно из следующих значений:

- gauspuls - синусоида, модулированная кривой Гаусса;
- rectpuls - прямоугольный импульс;
- tripuls - треугольный импульс.

Выходной сигнал y рассчитывается для значений аргумента, заданных в векторе t, по формуле y = func(t - d(1)) + func(t - d(2)) + ... Число импульсов в заданном диапазоне значений аргумента равно length(d).

Функция y = pulstran(t, d, 'func', pi, p2,...) позволяет задавать дополнительные параметры обращения к 'func', например func(t - d(l), p1, p2,...).

Функция y = pulstran(t, d, p, Fs) позволяет определять импульс последовательностью отсчетов, заданных в векторе p. Частота дискретизации задается параметром Fs.

При использовании функции y = pulstran(t, d, p) частота дискретизации полагается равной 1 Гц.

Примеры:

1. Сформируем и выведем на экран пилообразный сигнал d частотой повторения импульсов 3 Гц и длиной импульса - 0,1 с (рисунок 1.5). Продолжительность сигнала составит 1с при частоте дискретизации 1кГц.

```
t = 0: 1/1e3: 1;
d =0:1/3:1;
y = pulstran(t, d, 'tripuls', 0.1, -1);
plot(t, y)
```

2. Сформируем и выведем на экран последовательность гауссовых импульсов, имеющих частоту несущей 10 кГц и ширину полосы 50%

(рисунок 1.6). Частота повторения импульсов составляет 1 кГц, частота дискретизации -50 кГц, длина импульса - 10 мс. Коэффициент затухания амплитуды импульсов равен 0.8.

```
t = 0: 1/50e3: 10e-3;
d = [0 : 1/1 e3 : 10e-3; 0.8 .^(0 : 10)]';
y = pulstran(t, d, 'gauspuls', 10e3, 0.5);
plot(t,y);
```

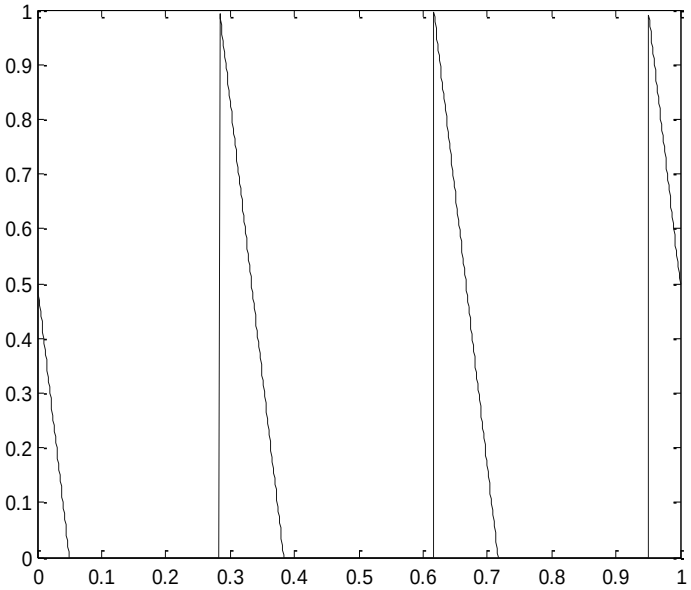


Рисунок 1.5

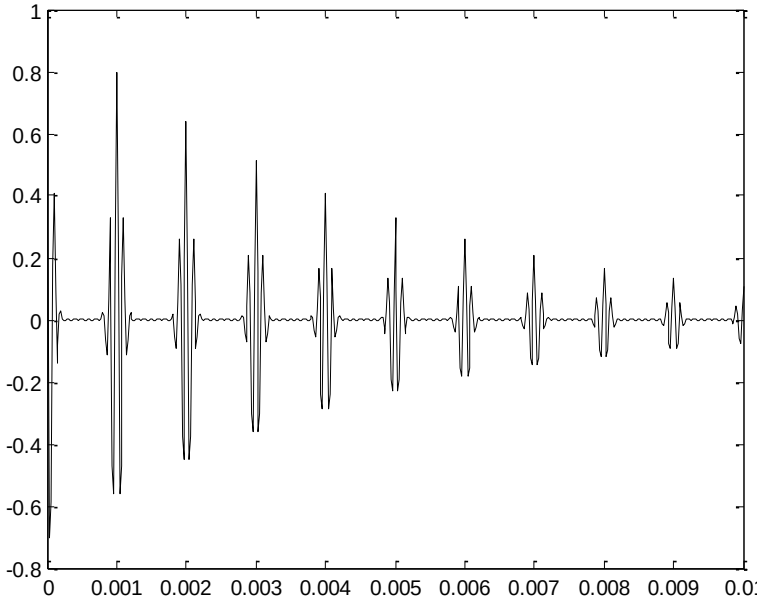


Рисунок 1.6

3. Сформируем последовательность 10 окон Хемминга (рисунок 1.7).

```
p = hamming(32);
```

```
t = 0 : 320; d = (0 : 9)'*32;
y = pulstran(t, d, p );
plot (t, y);
```

Сопутствующие функции: CHRP, COS, DIRIC, GAUSPULS, RECTPULS, SAWTOOTH, SIN, SINC, SQUARE, TRIPULS.

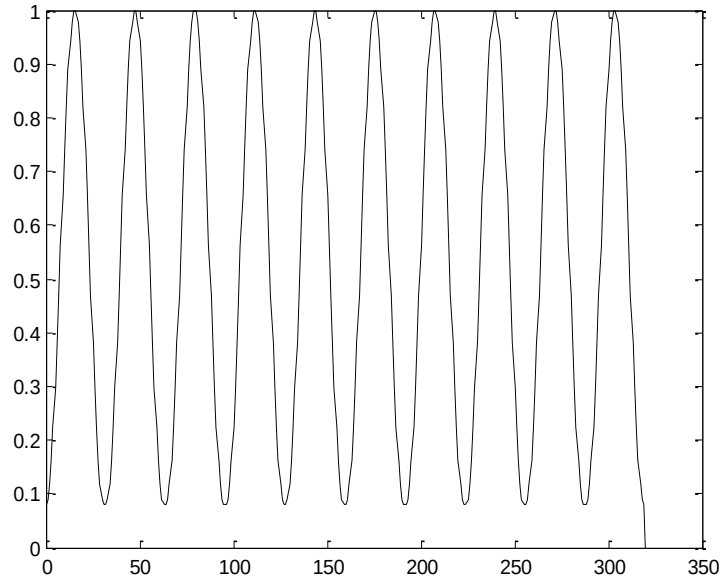


Рисунок 1.7

1.2 Применение пакета *SUMULINK* для формирования и исследования сигналов.

Теоретическая часть данного вопроса подробно описана в:

Михальов О.І., Гнатушенко В.В., Гнатушенко В.В. Системи штучного інтелекту: Методичні вказівки до виконання лабораторних робіт. Част.2. — Д.: НМетАУ, 2005. – 55 с.

ЗАДАНИЕ

1. Изучить вопросы формирования типовых сигналов при работе в командной строке.

2. Проработать самостоятельно вопросы формирования и исследования всех рассмотренных типовых сигналов в пакете ***SUMULINK.***

ЛИТЕРАТУРА

1. Гольденберг Л.М., Матюшкин Б.Д., Поляк М.Н. Цифровая обработка сигналов: Учебное пособие для вузов. – М.: Радио и связь, 1990. – 256 с.
2. Лазарев Ю.Ф. MATLAB 5.x. – К.: Издательская группа BHV, 2000. – 384 с.
3. Дьяконов В. П. MATLAB 6/6.1/6.5 + Simulink 4/5. Обработка сигналов и изображений. М.: Солон-Пресс. — 2004.
4. Черных И.В. S1MULINK. Среда создания инженерных приложений. М.: ДИАЛОГ-МИФИ. - 2003.
5. Дьяконов В.П. Simulink 4. Специальный справочник. СПб.: ПИТЕР. — 2002.
6. Медведев В.С, Потемкин В.Г. Control System Toolbox. MATLAB 5 для студентов. М.: ДИАЛОГ-МИФИ. - 2004.

Лабораторная работа № 2
БЫСТРОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ

Цель: Изучить методы спектрального анализа сигналов на основе дискретного преобразования Фурье, рассмотреть алгоритмы быстрого преобразования Фурье и реализовать в системе MATLAB.

2.1 Получение спектра сигнала

Спектром временной зависимости (функции) *x(t)* называется совокупность ее гармонических составляющих (гармоник), образующих ряд Фурье. Спектральный анализ периодических функций заключается в нахождении коэффициентов a_k, b_k ряда Фурье

$$x(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty}(a_k \cos 2\pi k f_1 t + b_k \sin 2\pi k f_1 t), \qquad (2.1)$$

где f_1-частота повторения (или частота первой гармоники), *k*-номер гармоники. Коэффициенты ряда Фурье определяются выражениями

$$a_k = \frac{2}{T}\int_0^T y(t)\cos 2\pi k f_1 t dt, \qquad (2.2)$$

$$b_k = \frac{2}{T}\int_0^T y(t)\sin 2\pi k f_1 t dt, \qquad (2.3)$$

где $T=1/f_1$-период повторения периодической функции *x(t).*

Для описания аналоговых и дискретных сигналов в частотной области используется аппарат преобразования Фурье. Спектром $X_a(j\omega)$ аналогового сигнала $x_a(t)$ называют прямое преобразование Фурье

$$X_a(j\omega) = \int_0^{\infty} x_a(t) e^{-j\omega t} dt. \qquad (2.4)$$

В свою очередь, согласно обратному преобразованию Фурье

$$x_a(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X_a(j\omega) e^{j\omega t} d\omega. \qquad (2.5)$$

Пара преобразований для решетчатой функции (дискретной последовательности) $x(nT)$ имеет вид:

$$X(e^{j\omega T}) = \Phi\{x(nT)\} = \sum_{n=0}^{\infty} x(nT) e^{-j\omega nT}; \qquad (2.6)$$

$$x(nT) = \Phi^{-1}\{X(e^{j\omega T})\} = \frac{T}{2\pi}\int_{\frac{-\pi}{T}}^{\frac{\pi}{T}} X(e^{j\omega nT})d\omega, \qquad (2.7)$$

где $X(e^{j\omega T})$ называют спектром дискретного сигнала.

2.2 Дискретное преобразование Фурье

Пусть $x(nT)$ – периодическая последовательность с периодом NT (период – N отсчетов) т.е. $x(nT)= x(nT+mNT)$, m – целое. Дискретным преобразованием Фурье (ДПФ) называют пару взаимно-однозначных преобразований:

$$X(k) = X(k\Omega) = \sum_{n=0}^{N-1} x(nT)e^{-jkn\Omega T}, \text{ k=0, 1, 2, …, N-1;} \qquad (2.8)$$

$$x(n) = x(nT) = \frac{1}{N}\sum_{k=0}^{N-1} X(k\Omega)e^{jkn\Omega T}, \text{ n=0, 1, 2, …, N-1;} \qquad (2.9)$$

где $\Omega = \frac{2\pi}{NT}$ - основная частота преобразования (бин ДПФ), причем (2.8) определяет прямое ДПФ, а (2.9) – обратное ДПФ.

Дискретное преобразование Фурье $X(k)$, как и сама последовательность $x(n)$, является периодической функцией по аргументу k с периодом N. ДПФ может быть использовано и для представления последовательности $x(nT)$ конечной длины N, определенной при $n=0, 1, 2, ..., N-1$ и равной нулю вне интервала [0, N-1]. Действительно такую последовательность можно рассматривать как один период соответствующей периодической последовательности и использовать преобразования (2.8), (2.9).

2.3 Быстрое преобразование Фурье (БПФ)

Дискретное преобразование Фурье $X(k)$ конечной последовательности $X(nT)$, n=0, 1, 2, …, N-1 определяется согласно (2.8), (2.9):

$$X(k) = \sum_{n=0}^{N-1} x(nT)W_N^{nk}, \; k\text{=0, 1, 2, …, } N\text{-1;} \qquad (2.10)$$

$$x(nT) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)W_N^{-nk}, \text{ n=0, 1, 2, …, } N\text{-1,} \qquad (2.11)$$

где $W_N = e^{-j\frac{2\pi}{N}}$, причем W_N является периодической последовательностью с периодом N, так как $W_N^{(nk+mN)} = W_N^{nk}$, m=0, ±1, ±2, ….

Непосредственное вычисление ДПФ (2.10) при комплексных значениях *x(nT)* требует для каждого значения *k* (*N*-1) умножений и (*N*-1) сложений комплексных чисел или 4(*N*-1) умножений и (2*N*-2) сложений действительных чисел, а для всех *N* значений *k*=0, 1, 2, …, *N*-1 требуется примерно N^2 умножений и N^2 сложений комплексных чисел. Таким образом, для больших значений *N* (порядка нескольких сотен или тысяч) прямое вычисление ДПФ требует выполнения весьма большого числа арифметических операций умножения и сложения, что затрудняет реализацию вычисления в реальном масштабе времени процессов и спектров.

Быстрым преобразованием Фурье называют набор алгоритмов, реализация которых приводит к существенному уменьшению вычислительной сложности ДПФ. Исходная идея алгоритмов состоит в том, что *N*-точечная последовательность разбивается на две более короткие, например на две (*N*/2)-точечных последовательности, вычисляются ДПФ для этих более коротких последовательностей и из этих ДПФ конструируется ДПФ исходной последовательности. Для двух (*N*/2)-точечных последовательностей требуется примерно $(N/2)^2*2=N^2/2$ умножений комплексных чисел, т.е. число умножений (а так же сложений) уменьшается примерно в 2 раза. Аналогично вместо вычисления ДПФ (*N*/2)- точечной последовательности можно вычислить ДПФ для двух (*N*/4)-точечных последовательностей и таким образом вновь сократить требуемое количество операций умножения и сложения. Если $N=2^{\nu}$, $\nu>0$ и целое, то процесс уменьшения размера ДПФ может быть продолжен до тех пор, пока не останутся только 2-точечные ДПФ. При этом общее число этапов вычисления ДПФ будет равно $\nu=\log_2 N$ раз, а число требуемых операций для вычисления *N*-точечной ДПФ будет порядка $N\nu$, т.е. уменьшается примерно в $N/\log_2 N$ раз. Так при *N*=1000 для прямого вычисления ДПФ требуется примерно $N^2=10^6$ операций комплексных умножений и сложений, а при использовании алгоритмов БПФ таких операций требуется всего порядка 10^4, т.е. объем вычислений сокращается примерно на два порядка.

Существуют различные алгоритмы БПФ, например, широко распространены алгоритм с прореживанием по времени, в котором требуется перестановка отсчетов входной последовательности *x(nT)* и алгоритм с прореживанием по частоте, в котором требуется перестановка отсчетов выходной последовательности X(k). В обоих алгоритмах БПФ требуется примерно $N\log_2 N$ операций (комплексных умножений) и оба алгоритма могут быть реализованы по способу с замещением, используя только один массив ячеек памяти.

2.3.1 Прямое одномерное дискретное БПФ — fft

Алгоритм БПФ реализован функцией y=fft(x[,n]). С ее помощью по известному вектору сигнала *x(n)* вычисляется вектор

$$X(k+1) = \sum_{n=0}^{N-1} x(n+1)e^{-j(2\pi)/N}, \qquad (2.12)$$

где N = length(x) — длина вектора исходных данных.

Если N есть степень числа 2, то используется высокоэффективный алгоритм БПФ для вещественных или комплексных данных. Время вычислений для комплексных данных примерно на 40-50% больше, чем для действительных. Если N является простым числом, выполняется алгоритм дискретного преобразования Фурье — ДПФ — по приведенной выше формуле. Если $N<n$, то недостающие элементы массива x дополняются нулями.

Прямое БПФ переводит представление сигнала из временной области в частотную. Это иллюстрирует приведенный ниже пример:

```
t=(0:1/99:1);                          % Вектор времени
x=sin(2*pi*10*t)+.5*sin(2*pi*30*t);  % Вектор сигнала
y=fft(x); % Вектор ДПФ сигнала
m=abs( y); p=unwrap(angle(y));        % Векторы амплитуд и фаз
f=(0:length(y)-l)*99/length(y);        % Вектор частот
plot(f,m); title( 'Magnitude');          % График АЧХ
set(gca,'XTick',[ 10 30 70 90]);
figure; plot(f,p*180/pi); title('Phase');     %График ФЧХ
set(gca,'XTick',[ 10 30 70 90]);
```

Этот пример показывает задание вектора временной зависимости сигнала, представленного суммой двух синусоид с частотами 10 и 30 Гц с амплитудами 1 и 0,5 соответственно. Затем производится дискретное БПФ и строится график АЧХ (рисунок 2.1).

Завершается пример построением графика фазочастотной характеристики спектра (ФЧХ), представленной на рис. 2.2. Обратите внимание на весьма полезную технику указания характерных частот спектра.

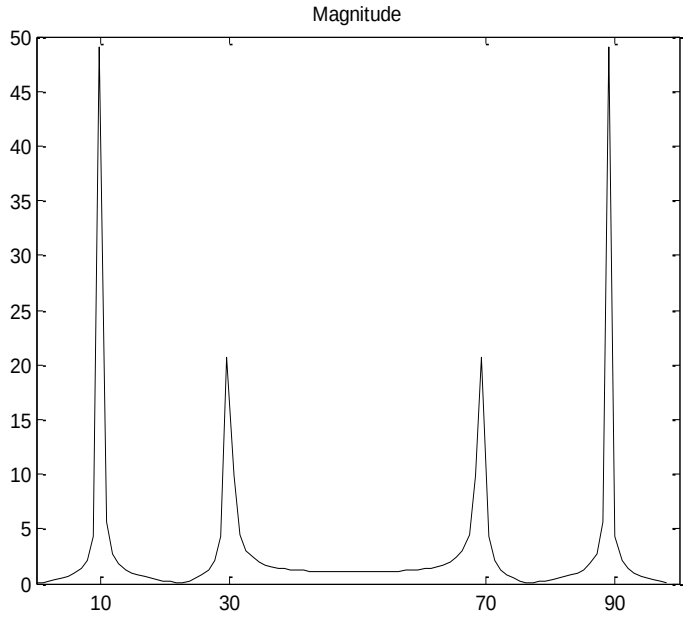


Рисунок 2.1 АЧХ спектра

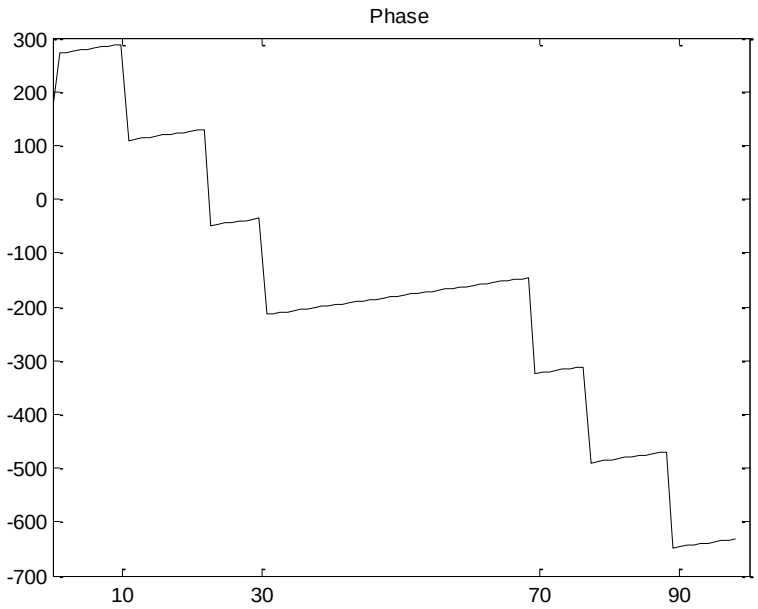


Рисунок 2.2 ФЧХ спектра

2.3.2 Перегруппировка выходного массива преобразования Фурье fftshift

При выполнении прямого БПФ спектральные компоненты, близкие к нулевой частоте, группируются по краям спектрограммы — рис. 1, например. Функция y=fftshift(x) обеспечивает перегруппировку элементов выходного массива преобразования Фурье таким образом, что эти компоненты оказываются в центре графика. Это иллюстрирует следующий пример:

```
t=(0:1/99:1);              % Вектор времени
x=sin(2*pi*10*t)+.5*sin(2*pi*30*t);    % Вектор сигнала
y=fftshift(fft(x));        %ДПФ сигнала с перегруппировкой
m=abs(y);
p=unwrap(angle(y));        % Векторы амплитуд и фаз
f=(0:1ength(y)-l)*99/length(y);    % Вектор частот
plot(f,m); title('Magnitude');          % График АЧХ
```

Рисунок 2.3 показывает АЧХ спектра для этого примера. Сравнив рисунок 2.1 с рисунком 2.3, нетрудно заметить, что нулевая частота здесь соответствует центру графика.

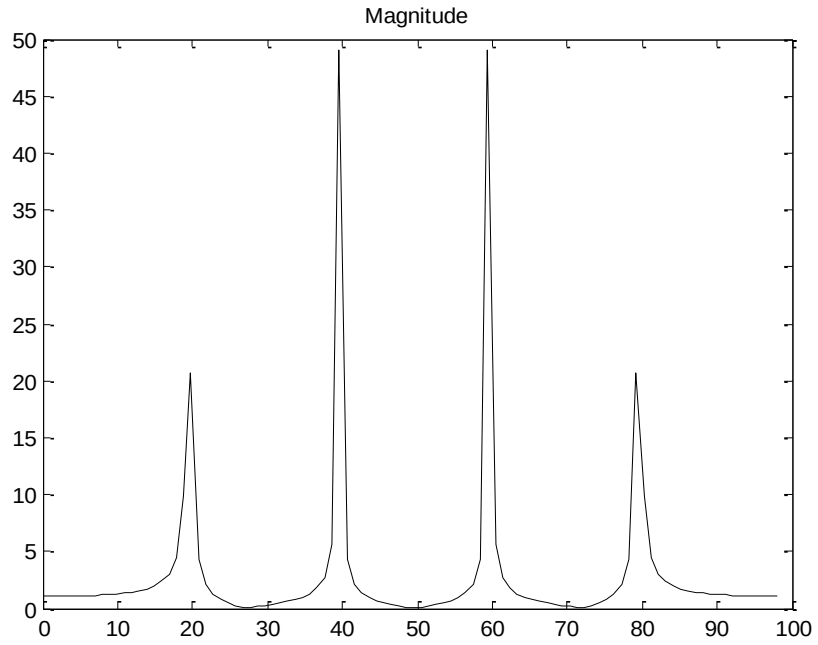


Рисунок 2.3 АЧХ спектра с перегруппировкой массива частот

2.3.3 Обратное одномерное дискретное БПФ –ifft

Обратное дискретное преобразование Фурье реализуется формулой:

$$X(t) = \frac{1}{N}\sum_{k=0}^{N-1} X(k+1)e^{-j2\pi/N}, \text{где} N = lenght(x) \qquad (2.13)$$

Все сказанное об алгоритме БПФ относится и к обратному преобразованию, которое реализуется функцией y=ifft(x[.n]). Для проверки функций fft и i fft можно использовать следующий пример:

```
x=[1 2 3 4];
```

```
X=fft(x)
X =
   10.000      -2.0000+2.0000 i       -2.0000    -2.0000-2.0000 i
x=ifft(X)
x=
1 2 3 4
```

Здесь исходный вектор подвергается вначале прямому, а затем обратному преобразованию Фурье. Как и следовало ожидать, после цепочки таких преобразований вновь получается исходный вектор.

Замечание

Полученный результат, однако, возможен только для простых, чисто тестовых и демонстрационных примеров. При векторах большого размера это уже не совсем так, а в случае введения ограничений на число гармоник, полученных при прямом БПФ, искажения синтезированного сигнала могут быть весьма заметными.

2.3.4 Матрица дискретного преобразования Фурье — dftmtx

Функция A=dftmtx(n) возвращает матрицу дискретного преобразования Фурье размера n x n, такую, что матричное выражение y=A*x задает прямое дискретное преобразование Фурье. Следующий пример демонстрирует применение функции dftmtx:

```
>>x=[1 2 3 4]; y1=fft(x);
>>n=length ( x ) ; y2=x*dftmtx ( n )
y1=
10.0000   -2.0000+2.0000i    -2.0000      -2.0000-2.0000i
y2=
 10.0000   -2.0000+2.0000i    -2.0000-0.0000i  -2.0000-2.0000i
>>norm(y1-y2)
ans = 2. 3019e-015
```

Здесь для вектора x выполнено прямое дискретное БПФ вначале с помощью функции fft, а затем с применением функции dftmtx. Результаты совпадают с точностью до машинной погрешности. Величина погрешности оценивается нормой разности векторов y1 и y2, полученных в этом примере.

Матрица Ai=conj(dftrntx(n))/n, соответственно, обеспечивает обратное дискретное преобразование Фурье.

Замечание.

При всей привлекательности использования функции dftmtx, для вычисления ДПФ эффективнее использовать функцию fft. В качестве примера, иллюстрирующего падение точности вычисления при использовании функции dftmtx на больших выборках, в предыдущей программе рассмотрите x=1:256 .

ЗАДАНИЕ

1 Провести прямое и обратное БПФ (проиллюстрировать графиками АЧХ и ФЧХ) следующих функций:

1.1 Для всех сигналов из лабораторной работы №1.

Для сигналов вида:

- *Двухсторонний экспоненциальный импульс*: $x(t)=A*exp(-a*|t|)$, *где A и a – первая и последняя цифра в номере зачетки (например, A=5, a=0.3).*
- *Гауссов импульс:* $x(t)=A*exp(-a^2*t^2)$.
- Сигнал: $x(t)=A*sinc(x)$.
- *Зашумленный гармонический сигнал*:

$$x(t)=A_1*cos(\omega_1 t+\varphi_1)+ A_2*cos(\omega_2 t+\varphi_2)=+a*randn(t) ,$$

используя программный фрагмент, приведенный в Help для функции fft.

2 Исследовать, что происходит с АЧХ и ФЧХ при зашумлении Гауссова и экспоненциального импульсов.

3 Для всех указанных сигналов вычислить матрицу ДПФ.

4 Произвести анализ спектров различных видов сигналов:

1) синусоидальных колебаний;
2) прямоугольных колебаний со смещением;
3) прямоугольных колебаний без смещения;
4) белого шума.

Для этого в програмном пакете SimuLink необходимо построить схему, представленную на рисунке 2.4 , где для получения синусоидальных колебаний на сумматор (Sum) подается сигнал (с амплитудой *A*=1 и частотой *F*=0.1 Гц (2*pi*0.1)) с генератора синусоидальных сигналов (SineWave) при нулевом значении выходного сигнала блока постоянной составляющей (Constant).

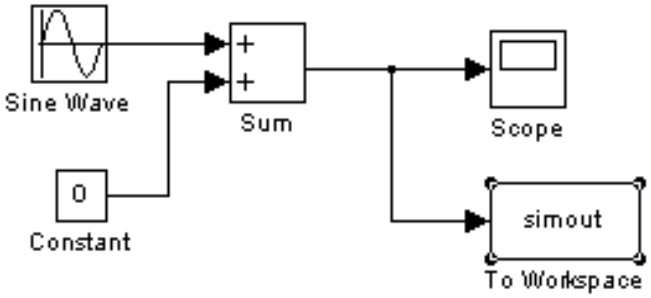


Рисунок 2.4

Прямоугольные колебания со смещением получаются путем суммирования выходных сигналов блоков PulseGenerator с амплитудой сигнала A=2 и Constant с выходным значением A= -1. Для получения прямоугольных колебаний без смещения амплитуда выходного сигнала блока PulseGenerator должна составлять A=1, при отсутствие сигнала на выходе блока Constant. В свойствах блоков Scope (просмотр сигнала) и ToWorkspase (вывод в рабочую область) необходимо установить период дискретизации (Sample Time) $T_д$=0,1 с. При завершении настроек системы нужно промоделировать сигнал (кнопка ▶). Полученный сигнал импортируется в среду SPTool, где в качестве источника необходимо выделить simout, и установить частоту дискретизации $F_д$=10 Гц. Когда сигнал импортирован в окне SpektrWiever можно увидеть его спектр, для этого надо выбрать в качестве метода получения спектра FFT (быстрое преобразование Фурье) и линейную шкалу графика. В ходе лабораторной работы необходимо получить спектры всех четырех видов сигналов при различных частотах дискретизации и сравнить их.

ЛИТЕРАТУРА

1. Лазарев Ю.Ф. MATLAB 5.x. – К.: Издательская группа BHV, 2000. – 384 с.
2. Дьяконов В. П. MATLAB 6/6.1/6.5 + Simulink 4/5. Основы применения. Полное руководство пользователя. — М.: Солон-Р. — 2002.
3. Дьяконов В. П. MATLAB 6/6.1/6.5 + Simulink 4/5 в математике и моделировании. Полное руководство пользователя. — М.: Солон-Р. — 2002.
4. Дьяконов В. П. MATLAB 6/6.1/6.5 + Simulink 4/5. Обработка сигналов и изображений. Полное руководство пользователя. — М.: Солон-Р. — 2004.
5. Потемкин В. Г. Система инженерных и научных расчетов MATLAB 5.x. В 2-х томах. - М.: ДИАЛОГ-МИФИ. - 1999.
6. Гультяев А. Визуальное моделирование в среде MATLAB: учебный курс. — СПб.: ПИТЕР. - 2001.

Лабораторная работа №3
СВЕРТКА И ДЕКОНВОЛЮЦИЯ

Цель: Изучить основные свойства свертки и деконволюции и реализующие их функции системы MATLAB.

Основными базовыми функциями обработки данных являются свертка и деконволюция. Они применяются не только при проектировании фильтров, но и в геофизике, медицине, неразрушающем контроле, криптографии, обработке экспериментальных данных в науке и технике, обработке изображений и т. д.

3.1 Свертка одномерных сигналов — conv

Пусть имеется две последовательности, представленные векторами a и b. С*верткой* называют одномерный массив, вычисляемый так:

$$c(n+1) = \sum_{k=0}^{N-1} a(k+1)b(n-k) \qquad (3.1)$$

При записи этого выражения учтено, что нумерация индексов всех массивов (даже массивов Java) в MATLAB идет с 1. Свертка реализуется функцией conv(*a*,*b*).

Пример 1:

a=[2 4 6]; *b*=[3 5 7]; *c*=conv(*a*,*b*)

c=

6 22 52 58 42

Операцию свертки часто используют для вычисления сигнала y на выходе линейной системы по сигналу x на входе при известной импульсной характеристике системы h. (рисунок 3.1):

$$y(k) = \sum_{m=-\infty}^{k} x(m)h(k-m) \qquad (3.2)$$

3.2 Операция, обратная свертке — deconv

Операцию, обратную свертке, выполняет функция [q,r]=deconv(b,a). Данная функция позволяет вычислить импульсную характеристику линейной системы. Если y=conv(x,h), то q = y и r = 0.

Пример 2:

a=[2 4 6]; *b*=[3 5 7]; *c=conv(a.b)*

```
 c =
6   22   52   58   42
 [q,r]=deconv(c,a)
q=
357
r =
00000
```

4.1 Свертка двумерная и многомерная — conv2 и convn

Для осуществления свертки двумерных массивов (матриц) A и B с размерами *ma × na и mb × nb* служит функция C=conv2(A,B[, 'shape']). Параметр 'shape' в ней может иметь следующие значения:

'full' — полноразмерная свертка (принята по умолчанию);

'same' — центральная часть свертки, определяемая размером массива A;

'valid' —центральная часть свертки с размером [ma-mb+1, na—nb+1] при size(A) > size(B).

Выполнение двумерной свертки эффективно при size(A) > size(B). Эта операция используется при обработке изображений. Свертка многомерная реализуется функцией convn с записью, подобной рассмотренной для функции conv2. Она используется с многомерными массивами.

ЗАДАНИЕ

1. Самостоятельно проработать вопрос реализации свертки как матричного умножения для данных примеров 1,2, применив функцию *convmtx(x,n).* Сравнить полученные результаты с результатами примеров 1 и 2.

2. Пусть задан входной сигнал (отсчеты) *x(k)=[1,5,8,10,4,-3,-7,2,5]* и импульсно-переходная характеристика линейной системы *h(k)=[3,4,6,8,10,5,3,2,1,2,3].* Найти выходной сигнал *y.*

3. Решить задачу идентификации линейной системы при известных входных и выходных сигналах *x, y* пункта 2.

4. Используя данные пункта 2 найти спектр свертки сигналов. Построить АЧХ и ФЧХ.

5. Пусть заданы сигналы *s(k)=[1, 3, 5, 6, 3, 2, -1, 2, 4]* и *x(k) (данные из п.2).* Найти произведение этих сигналов и спектр этого произведения. Построить АЧХ и ФЧХ.

6. Сгенерировать произвольные двумерные матрицы **A** и **B.** Найти двумерную свертку. Исследовать при этом влияние параметра *'shape'*.

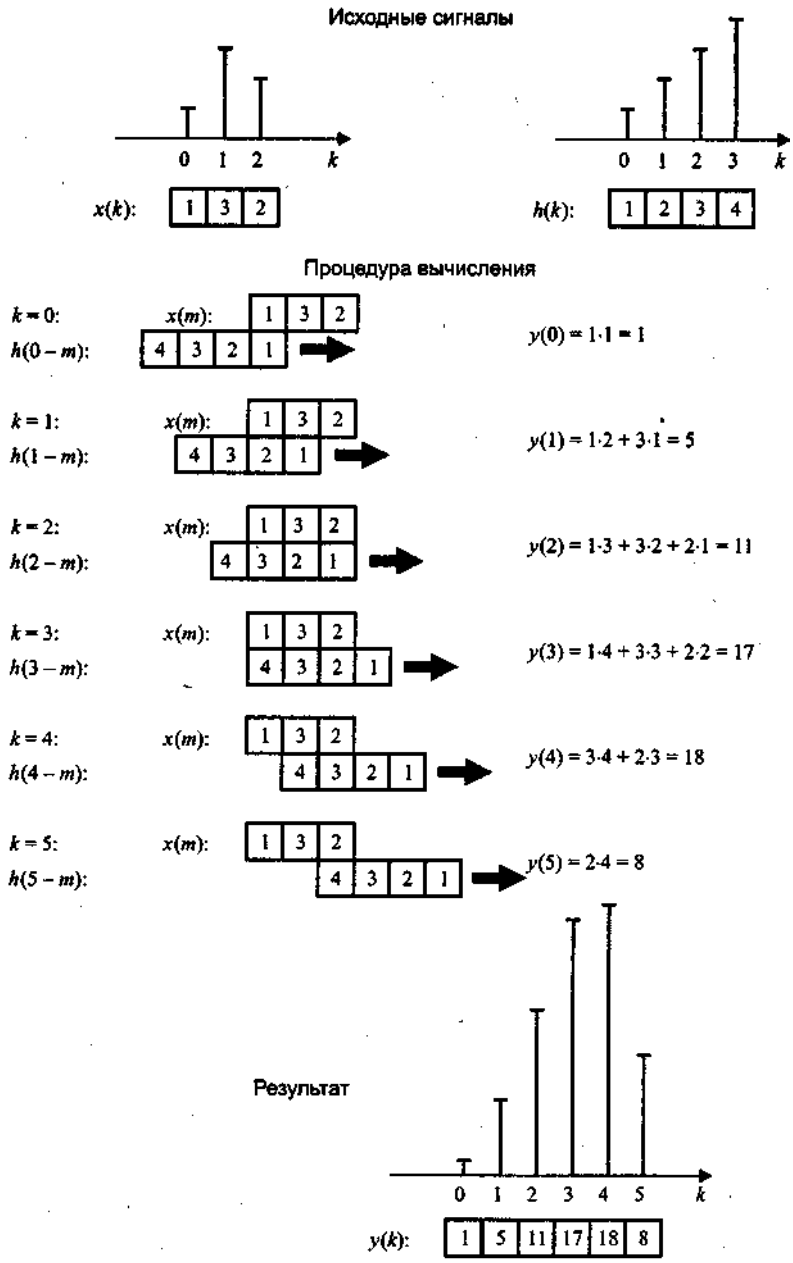


Рисунок 3.1 Вычисление дискретной свертки

ЛИТЕРАТУРА

1. Чен К., Джиблин П., Ирвинг А. MATLAB в математических исследованиях. Мир, 2001.

2. Потемкин В. Введение в MATLAB. Диалог-МИФИ. 2000.

Лабораторная работа №4

ИССЛЕДОВАНИЕ ХАРАКТЕРИСТИК АНАЛОГОВЫХ И ЦИФРОВЫХ ФИЛЬТРОВ

Цель: Изучить частотные характеристики аналоговых и дискретных фильтров; аналитически исследовать указанные фильтры в пакете Control System Toolbox; «экспериментально» исследовать аналоговые и дискретные фильтры в пакете Simulink.

4.1 Теоретические основы

Фильтр — это устройство или программа, которая обеспечивает частотно зависимые преобразования входного сигнала. Для фильтра низких частот устройства или программы должны обеспечивать отсутствие амплитудных искажений входного сигнала в области частот от 0 до некоторой заданной ω_c, и эффективное подавление частот, которые превышают граничную частоту ω_c.

Аналоговый фильтр может быть представлен непрерывной передаточной функцией:

$$W(s) = \frac{Y(s)}{X(s)} = \frac{M(s)}{N(s)}, \qquad (4.1)$$

где, *Y(s), X(s)* – изображение по Лапласу выходного и входного сигналов фильтра, соответственно; *M(s), N(s)* — полинома числителя и знаменателя передаточной функции.

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

$$A(\omega) = 20 \times \lg\{\left|W(j\omega\right|^{-1}\} = 10\lg L(\omega^2),$$

фазовую характеристику $Q(\omega)$

$$Q(\omega) = \arg(W(j\omega))$$

характеристику групповой задержки τ

$$\tau = \frac{dQ(\omega)}{d\omega}.$$

Функцию $L(s^2) = \left|W(s) \times W(-s)^{-1}\right|$ называют функцией затухания.

Идеальный фильтр низких частот (ФНЧ) пропускает только низкочастотные составляющие. Его характеристика затухания имеет вид, приведенный на рис.4.1.

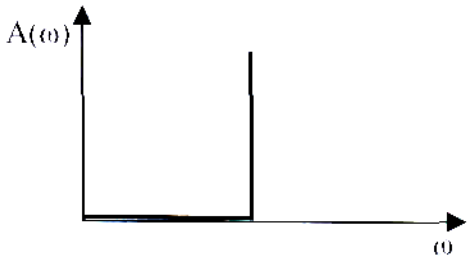


Рисунок 4.1 Частотные характеристики затухания идеального ФНЧ

Диапазон частот от 0 до ω_C называется полосой пропускания, остальной частотный диапазон — полосой задержания. Граница между этими полосами ω_C называется частотой среза. В реальных фильтрах переход от частоты пропускания к частоте задерживания происходит плавно (рисунок 4.2).

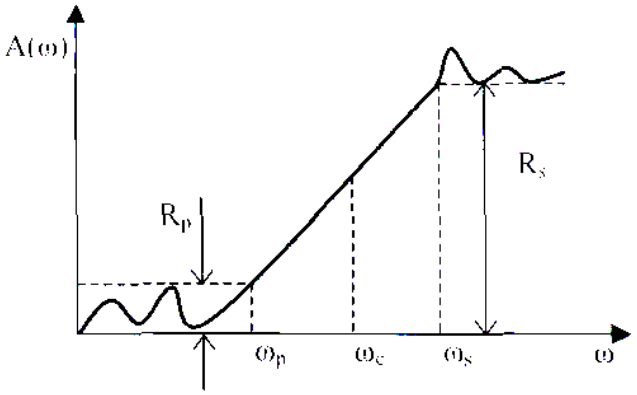


Рисунок 4.2 Частотные характеристики затухания реального ФНЧ

На рисунке 4.2 показаны контрольные точки ФНЧ, которые используются при его проектировании:

W_P – граничная частота пропускания;

W_S – граничная частота задержки;

R_P – максимально допустимое подавление сигнала в полосе пропускания, дБ;

R_S – минимально допустимое подавление сигнала в полосе задерживания, дБ.

Командой MATLAB buttord определяется порядок n и частота среза W_c аналогового фильтра Баттерворта по исходным данным этой команды, заданной частотой пропускания и задерживания (W_P, W_S) и максимальным допустимым

подавлением сигналов в полосе пропускания и задерживания *(R_P,R_S)*. Причем, если команда buttord имеет параметр s

$$[_n,W_c]=buttord(W_P,W_s,R_P,R_{s,}{}^{/}s^{/}),$$

то W_P, W_s должны быть заданы в радианах в секунду, значения частоты W_c. Также будет получено в радианах за секунду, а *n* будет определять степень характеристического управления.

Если команда buttord записана без параметра *s*

$$[n,\ W_c\ ] = buttord(W_p,W_s\ ,R_P,R_S),$$

то частоты W_p и W_s задаются в относительных единицах ω/ω_B, изменяющихся от 0 до 1 (где 1 соответствует частоте ω_B), определяемой выражением:

$$\omega_B = \frac{\pi}{T_П}, \qquad (4.2)$$

где $T_П$ - период дискретности.

Если определяется дискретный фильтр, соответствующий данному аналоговому фильтру, то ω_B определяет частоту, при которой наблюдаются максимальное различие между амплитудными значениями аналогового и дискретного фильтрами и по этому расхождению можно оценить правильность выбора $T_П$. В то же время целесообразно помнить, что ω_B — это частота, выше которой частотными характеристиками аналогового сигнала можно пренебречь.

В пакете MATLAB Control System Toolbox аналоговые и цифровые фильтры могут быть представлены:

- в виде уравнений пространства состояний - форма ss;
- через передаточные функции - форма tf;
- через нули и полюса - форма zpk.

Переход от непрерывного представления уравнений к их дискретному представлению определяется командой *c2d,* а переход к непрерывному представлению, если задан его дискретный аналог, определяется командой *d2c.*

Приведенные выше теоретические положения рассмотрены на конкретных примерах.

```
%Программа 1
Wr=1; %Граничная частота пропускания.
Ws=10;%Граничная частота задержки.
```

```
Rp=6;%Максимально допустимое подавление в полосе
пропускания(дБ).
Rs=20;%Минимально допустимое подавление в полосе
задерживания(дБ)
[n,Wn]=buttord(Wr,Ws,Rp,Rs,'s')  %Определение параметров
%аналогового фильтра Баттерворта.
[z,p,k]=buttap(n);%Определение нулей и полюсов фильтра
Баттерворта.
[b,a]=zp2tf(z,p,k)%Определение коэффициентов фильтра
Баттерворта.      h=tf([b],[a])%Определение передаточной
функции
%аналогового фильтра Баттерворта.
%Проектирование фильтра Баттерворта с более жесткими
требованиями: в полосе пропускания уменьшено максимально
допустимое подавление
Wr=1 ;
Ws=10;
Rp=2 ;
Rs=20;
[n2,Wn2]=buttord(Wr,Ws,Rp,Rs,'s')
[z,p,k]=buttap(n2);
[b2,a2]=zp2tf(z,p,k)
h2=tf([b2],[a2])%Определение     передаточной     функции
%аналогового фильтра Баттерворта.
%Проектирование фильтра Баттерворта с более жесткими
требованиями: в полосе Задерживания увеличено максимально
допустимое подавление
Wr=1;
Ws=10;
Rp=2;
Rs=52;
[n3, Wn2] =buttord (Wr,Ws,Rp,Rs,'s')
[z,p,k]=buttap(n3);
[b3,a3]=zp2tf(z,p,k)
h3=tf([b3],[a3])   %Определение   передаточной   функции
аналогового
%фильтра Баттерворта.
figure(1)%Построение ЛЧХ для трех фильтров Баттерворта.
bode(h,h2,h3),grid on
%Проектирование дискретных фильтров Баттерворта при t=0,2.
t=0.2;       %Интервал дискретности.
hd=c2d(h,t)  %Передаточная функция дискретного Фильтра,
%соответствующая   аналоговому   фильтру   h.
h2d=c2d(h2,t)%Передаточная            функция
дискретного Фильтра,
%соответствующая       аналоговому       фильтру       h2.
h3d=c2d(h3,t)%Передаточная функция дискретного Фильтра,
%соответствующая аналоговому фильтру h3.
```

```
figure(2)        %ЛАЧ и ЛФЧ характеристики
фильтров h и hd. bode(h,hd),grid on
figure(3)     %ЛАЧ и ЛФЧ характеристики фильтров
h2 и h2d. bode(h2,h2d),grid on
figure(4)     %ЛАЧ и ЛФЧ характеристики фильтров
h3 и h3d. bode(h3,h3d),grid on
%Проектирование дискретных фильтров Баттерворта при
t=0,05.
t=0.05;             %Интервал дискретности.
hd=c2d(h,t)     %Определение параметров дискретных фильтров
h2d=c2d(h2,t)   %для спроектированных непрерывных фильтров
h3d=c2d(h3,t)       %при новых интервалах дискретности.
figure(5)           %Построение ЛАЧ и ЛФЧ характеристик
bode(h,hd),grid  on     %аналоговых  и  дискретных  (при
измененном
figure(б)           %интервале дискретности)фильтров.
bode(h2,h2d),grid on
figure(7)
bode(h3,h3d),grid on
```

В этой программе последовательно исследуется три фильтра Баттерворта, имеющие одинаковые частоты пропускания W_r и задерживания W_s, но разные величины допустимого подавления сигнала в полосе пропускания R_p и полосе задерживания R_s. Параметры W_r, W_s, R_p, R_s определяют частоту среза W_n и порядок фильтров Баттерворта n. Причем, чем ближе частотные характеристики реального фильтра приближаются к частотным характеристикам идеального фильтра, тем больше величина n. Таким образом, порядок фильтра будет увеличиваться при увеличении протяженности горизонтального участка в полосе пропускания (уменьшением R_p), при увеличении требования к подавлению сигнала в полосе задерживания (увеличению R_s) или при уменьшении частотного диапазона между W_p и W_s. Связь между параметрами W_r, W_s, R_p и R_s (требования к фильтру) и порядком фильтра иллюстрируется примерами Программы 1.

В первом варианте допустимого уменьшения амплитуды в конце полосы пропускания в два раза, что определяет величину R_p

$$20\log10(1/2) \approx -6 \text{ } дБ.$$

Во втором варианте ФНЧ в полосе пропускания имеет более плоскую характеристику и допустимое уменьшение амплитуды на частоте среза составляет 20%

$$20\log10(1/0{,}8) = -1{,}93 \approx -2 дБ.$$

В третьем варианте проектируемого фильтра повышены требования к подавлению сигнала в полосе задержки (сигнал уменьшается в 400 раз)

$$201ogl0(1/400) \approx 52.$$

С повышением фильтрующих свойств увеличивается порядок фильтра, что иллюстрируется на рисунке 4.1.

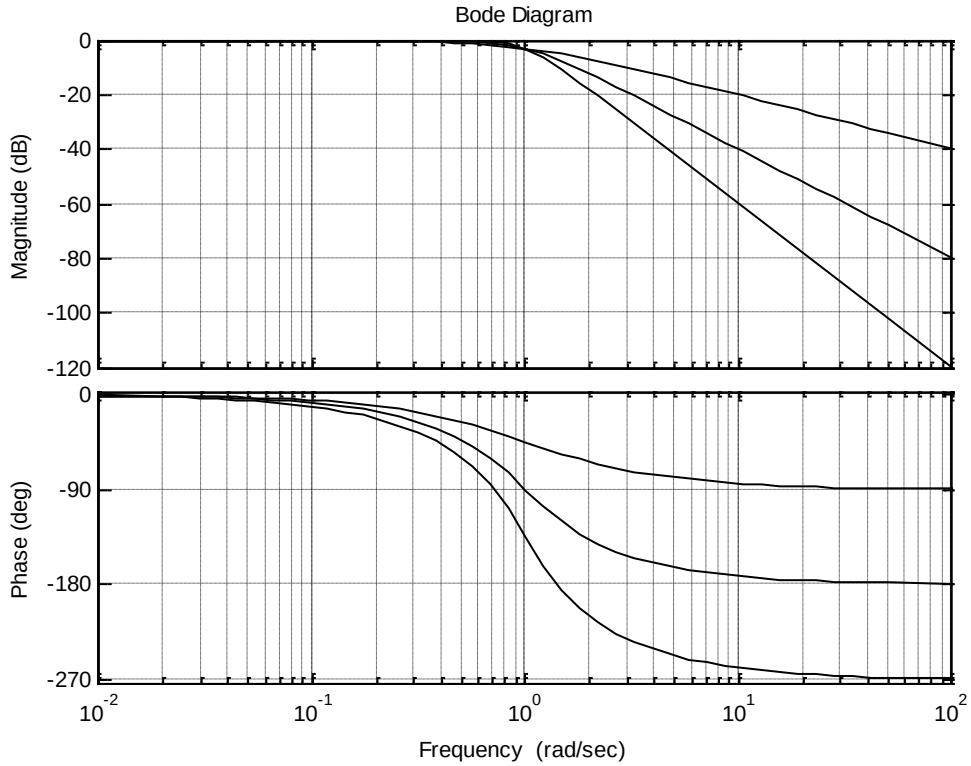


Рисунок 4.1 ЛАЧХ фильтров Баттерворта (сверху вниз - фильтр первого порядка (-20 дБ/дек); второго порядка (-40 дБ/дек); третьего порядка (-60 дБ/дек))

Во второй части Программы 1 командой *c2d* определяются Z-передаточные функции фильтров для фильтров первого, второго и третьего порядков при разных интервалах дискретности. Для $T_П$ = 0,2с имеем три передаточные функции

$$W_1(z) = \frac{0.1813}{z - 0.8187},$$

$$W_2(z) = \frac{0.1818z + 0.01654}{z^2 - 1.7192 + 0.7536}, \qquad (4.3)$$

$$W_3 = \frac{0.0012052^2 + 0.0043582 + 0.0009868}{z^3 - 2.6012^2 + 2.2782 - 0.6703},$$

Z-передаточные функции для $T_П$ = 0,05с представлены ниже

$$W_1(z) = \frac{0.04877}{z - 0.9512},$$
$$W_2(z) = \frac{0.001221z + 0.001192}{z^2 - 1.9292 + 0.9317}, \qquad (4.4)$$
$$W_3 = \frac{2.032 \times 10^{-5} \times z^2 + 7.926 \times 10^{-5} \times z + 1.933 \times 10^{-5}}{z^3 - 2.9z^2 + 2.805z - 0.9048}$$

В Программе 1 имеется команда *bode* для построения логарифмических характеристик непрерывных и дискретных систем (рис.4.2). Результаты выполнения этой команды представлены на рисунке 4.2.

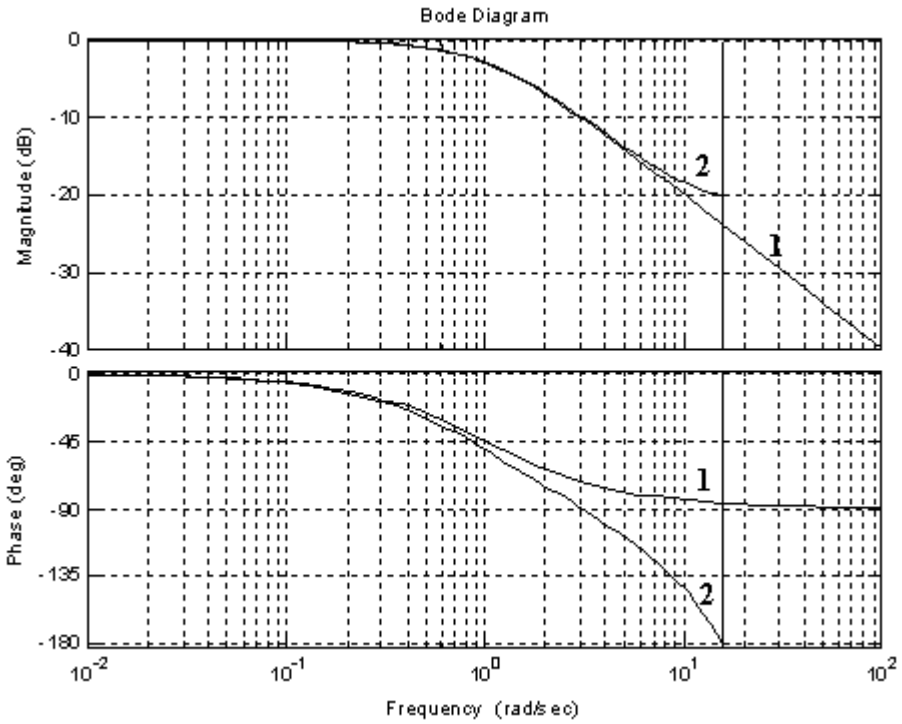


Рисунок 4.2 Логарифмические характеристики непрерывного и дискретного ФНЧ первого порядка для $T_П$ = 0,2с (1 - непрерывный фильтр; 2 - дискретный фильтр)

Из-за периодичности частотные характеристики дискретного фильтра рассчитываются от нуля до частоты $\omega_R = \pi / T_П$=3,14/0,2 = 15,7 *р/с*. Из рис.4.2 следует, что между амплитудными и фазовыми характеристиками непрерывного и дискретного фильтров на частоте ω_B наблюдаются значительные различия: для амплитудных характеристик различия достигают 6 дБ, для фазовых - 90°. В некоторых случаях величины указанных различий недопустимы. К полученным результатам следует относиться с осторожностью

еще и потому, что различие между ЛАЧХ непрерывного и дискретного фильтра проявляется при недостаточном ослаблении выходного сигнала, которое составляет (-20) дБ. Рекомендуемые ослабления выходного сигнала, при котором переход к дискретному представлении практически не вносит ошибки, составляет -(30-60) дБ. Уменьшение $T_П$ позволит приблизить характеристики дискретного фильтра к аналоговому в области высоких частот.

На рис.4.3 и рис. 4.4 приведены графики непрерывных и дискретных ФНЧ при увеличении требований к частотным характеристикам фильтров (уменьшились искажения в полосе пропускания и увеличилось подавление частот в полосе задерживания). Результаты Программы 1 показывают, что увеличился порядок фильтра и при этом:

- частота ω_B не изменилась;
- уменьшилась амплитудная ошибка на частоте ω_B;
- уменьшилась фазовая ошибка.

Причем, уменьшение амплитудных и фазовых ошибок наблюдается при значительных ослаблениях выходного сигнала -(50-70) дБ.

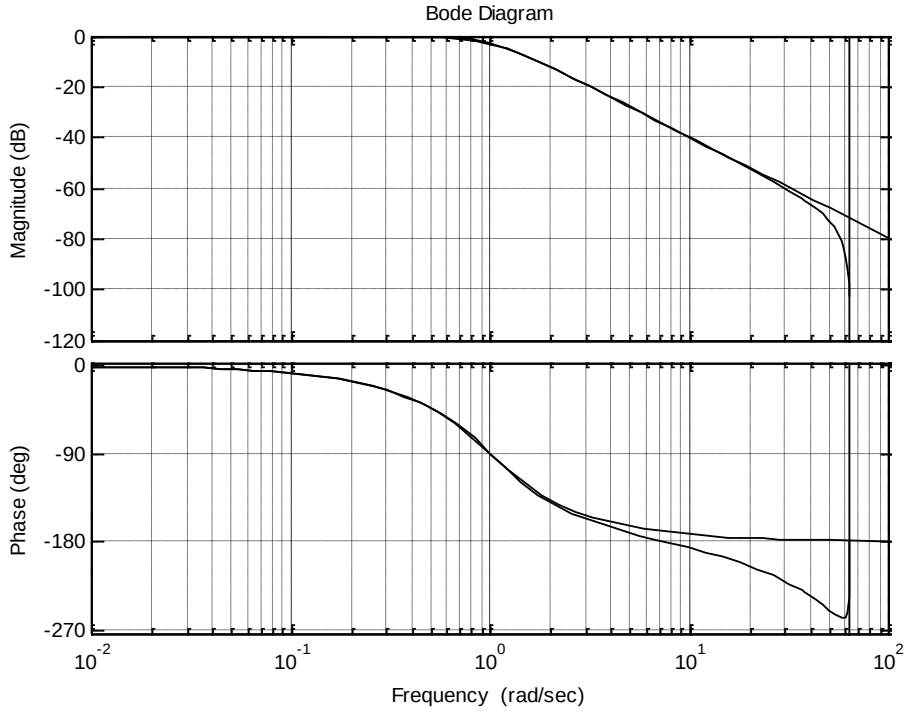


Рисунок 4.3 Логарифмические характеристики непрерывною и дискретного ФНЧ второго порядка для $T_П$ = 0,2с (сверху - непрерывный фильтр; снизу - дискретный фильтр)

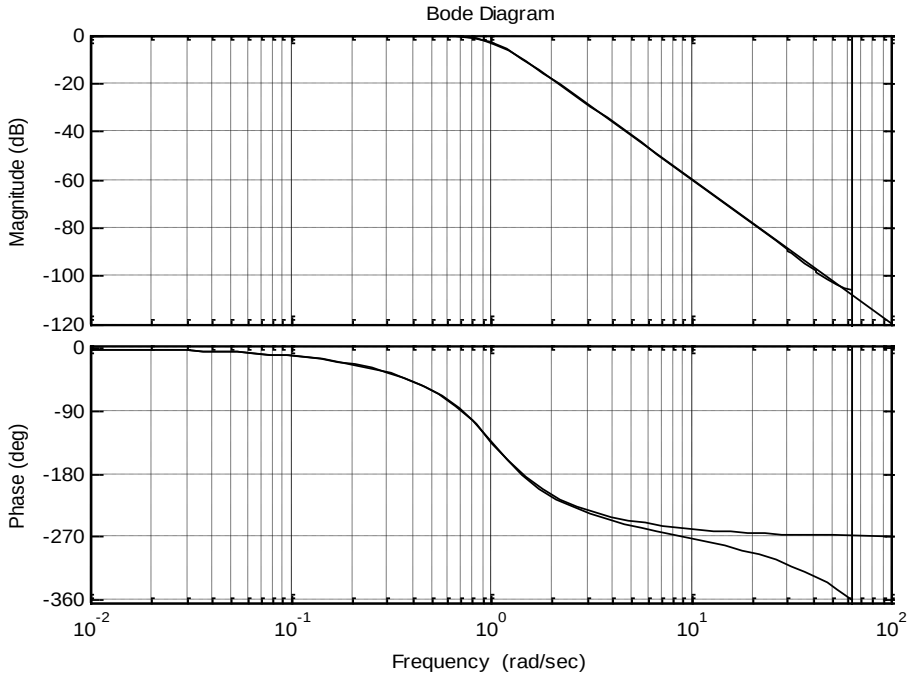


Рисунок 4.4 Логарифмические характеристики непрерывного и дискретного ФНЧ третьего порядка для $T_П = 0{,}2c$ (сверху - непрерывный фильтр; снизу - дискретный фильтр)

На рисунках 4.5, 4.6 и 4.7 приведены графики непрерывных и дискретных систем ФНЧ при $T_П$ = 0,05с.

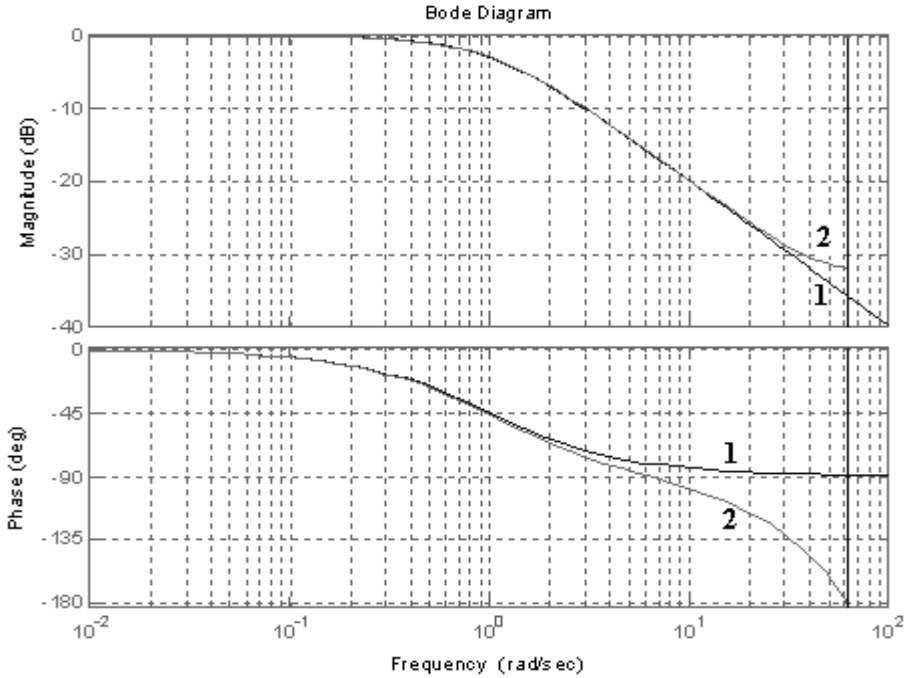


Рисунок 4.5 Логарифмические характеристики непрерывного и дискретного Ф11Ч первого порядка для $T_П$ = 0,05с (1 - непрерывный фильтр; 2 - дискретный фильтр)

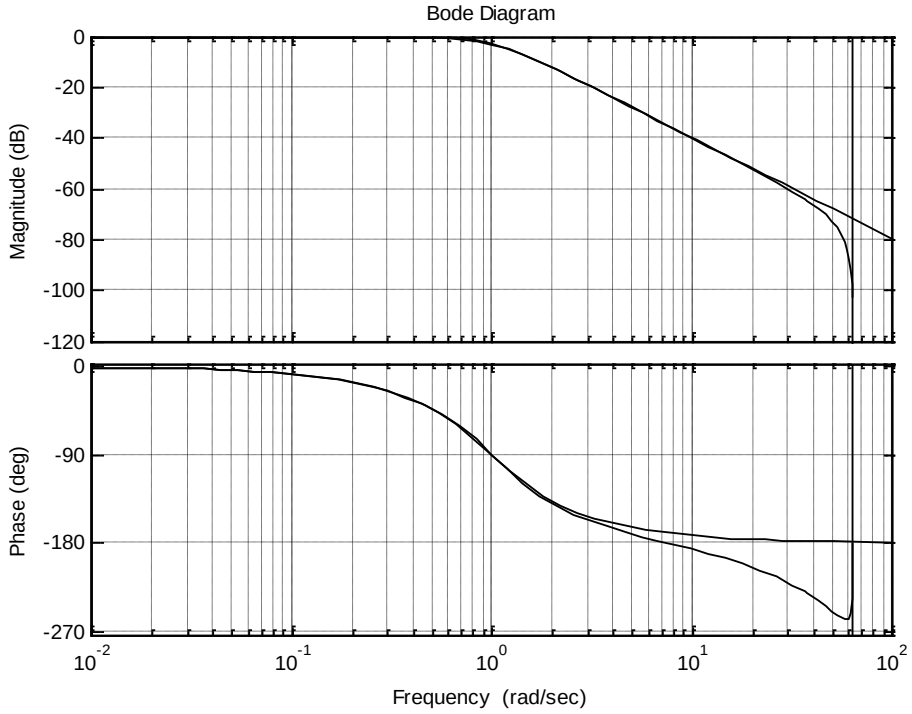


Рисунок 4.6 Логарифмические характеристики непрерывного и дискретного ФНЧ второго порядка для $T_П = 0{,}05$с (1сверху- непрерывный фильтр; снизу - дискретный фильтр)

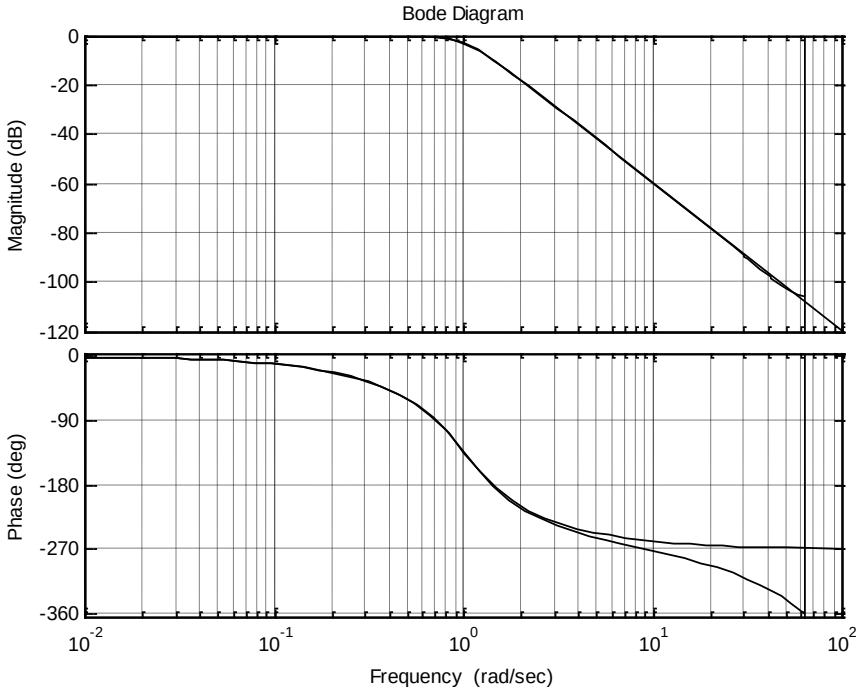


Рисунок 4.7 Логарифмические характеристики непрерывного и дискретного ФНЧ третьего порядка для $T_П = 0{,}05$с (сверху - непрерывный фильтр; снизу - дискретный фильтр)

Анализ графиков показывает, что уменьшение $T_П$ приводит к увеличению границы частотного диапазона ω_B=3,14/0,05 = 62,8 *p/c* и приближению частотных характеристик дискретных фильтров к частотным характеристикам аналоговых фильтров. Причем, наблюдаем отличия частотных характеристик происходит при значительных ослаблениях выходного сигнала (-35-110) дБ.

По полиномам числителя и знаменателя передаточной функции ФНЧ команда *lp21p* определяет передаточные функции ФНЧ с новыми частотами среза:

[b1,a1]=1$_P$21p(b,a,W$_0$)

где **b, a** - коэффициенты исходного фильтра низких частот;

W$_0$ - желаемая частота среза проектируемого фильтра;

b1, a1 - коэффициенты спроектированного фильтра с новыми частотами среза.

В Программе 2 приведены примеры определения передаточных функций ФНЧ (фильтра Баттерворта) для трех частот среза. Командой *buttap* определяют нули, пояса и коэффициент усиления фильтра Баттерворта заданного порядка п. Команда zp2tf преобразует математическую модель ФНЧ: от информации заданной нулями, полосами и коэффициентом усиления, к информации представленной передаточными функциями.

```
%Программа 2
wr=0.1; %Частота пропускания в относительных единицах.
ws=0.6;    %Частота Задерживания в относительных единицах.
rp=6;%Допустимое подавление в полосе пропускания.
rs=40;      %Допустимое подавление в полосе задерживания.
[n1,wc1]=buttord(wr,ws,rp,rs)   %Определение параметров
фильтра
%Баттерворта.
 [z1,p1,k1]=buttap(n1); %Определение фильтра Баттерворта  в
 форме ZPK [b1,a1]=zp2tf(z1,p1,k1)%Определение полинома
 числителя и знаменателя
                      %фильтра Баттерворта.
hl=tf([b1],[a1])       %Определение фильтра Баттерворта
в форме TF.
wl=10; %Частота среза проектируемого фильтра Баттерворта.
[b10,a10]=lp2lp(b1,a1,w1)%Параметры фильтра Баттерворта
на частоте w1.
w2=100;       %Частота среза проектируемого фильтра
Баттерворта.
[b100,a100]=lp2lp(b1,a1,w2)   %Параметры фильтра
Баттерворта
%с частотой среза w2.
```

```
q1=tf([b10],[a10])    %Передаточная функция фильтра
Баттерворта
                      %с частотой среза w1
q2=tf([b100],[a100])   %Передаточная функция фильтра
Баттерворта
                      %с частотой среза w2.
figure(1)     %Логарифмические характеристики проектируемых
фильтров
bode(h1,'g',q1,'k',q2,'r'),grid on  %при разных  частотах
среза
%(w=l;w=10;w=100)
```

Результаты выполнения Программы 2 приведены на рисунке 4.8

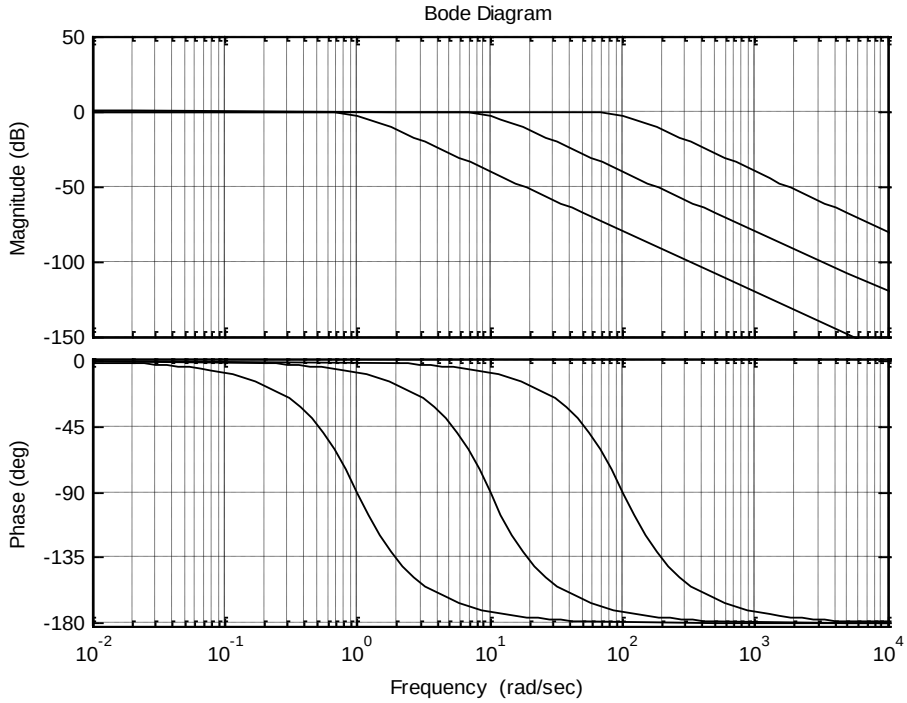
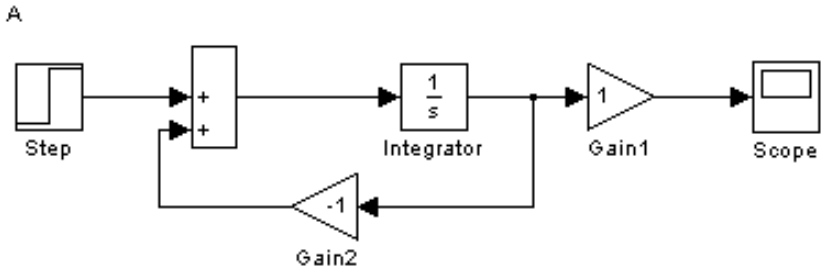
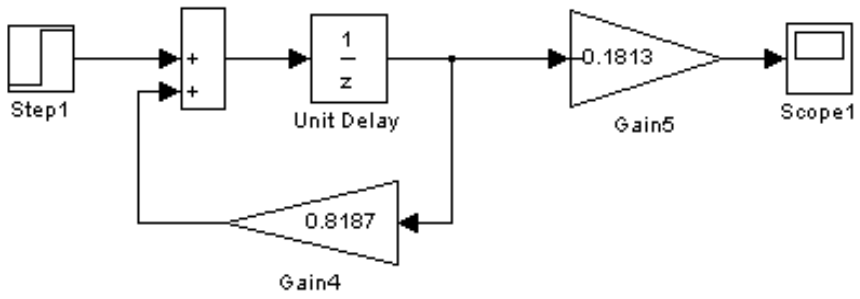


Рисунок 4.8 Логарифмические характеристики непрерывного ФНЧ второго порядка для разных частот среза (слева направо - частота среза 1 р/с; частота среза 10 р/с; частота среза 100 р/с.)

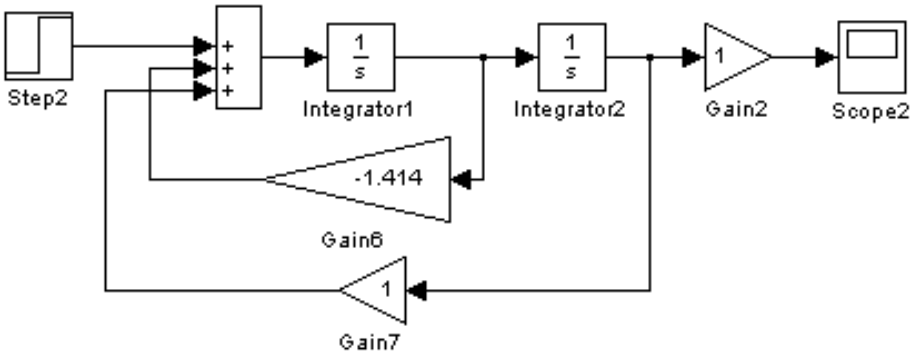
По передаточным функциям, определённым в программе 2, на рисунке 4.9 в пакете Simulik представлены структурные схемы непрерывных и дискретных фильтров нижних частот.



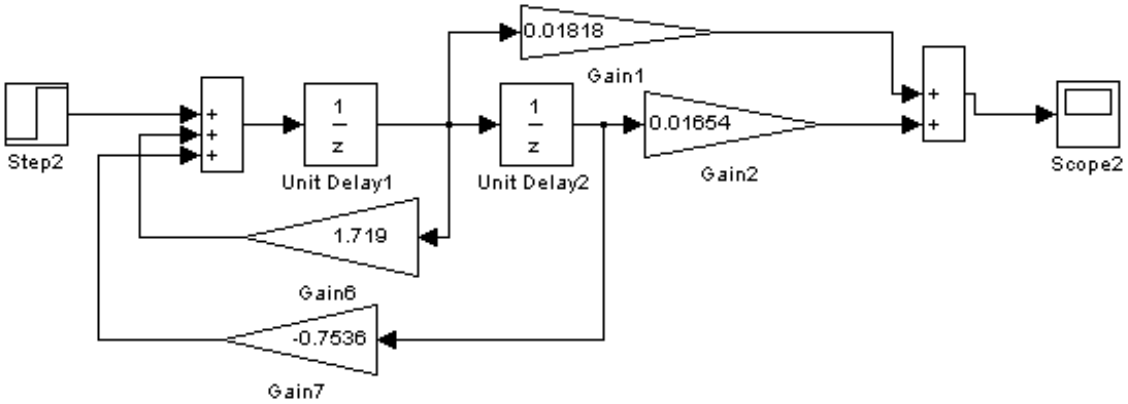
B


Step1
1
z
Unit Delay
0.1813
Gain5
Scope1
0.8187
Gain4

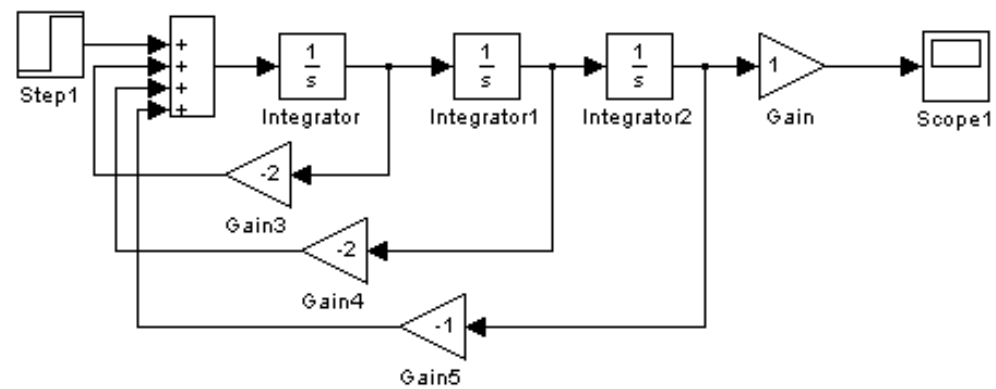
C


Step2
1
s
Integrator1
1
s
Integrator2
1
Gain2
Scope2
-1.414
Gain6
1
Gain7

D


0.01818
Gain1
Step2
1
z
Unit Delay1
1
z
Unit Delay2
0.01654
Gain2
Scope2
1.719
Gain6
-0.7536
Gain7

E


Step1
1
s
Integrator
1
s
Integrator1
1
s
Integrator2
1
Gain
Scope1
-2
Gain3
-2
Gain4
-1
Gain5

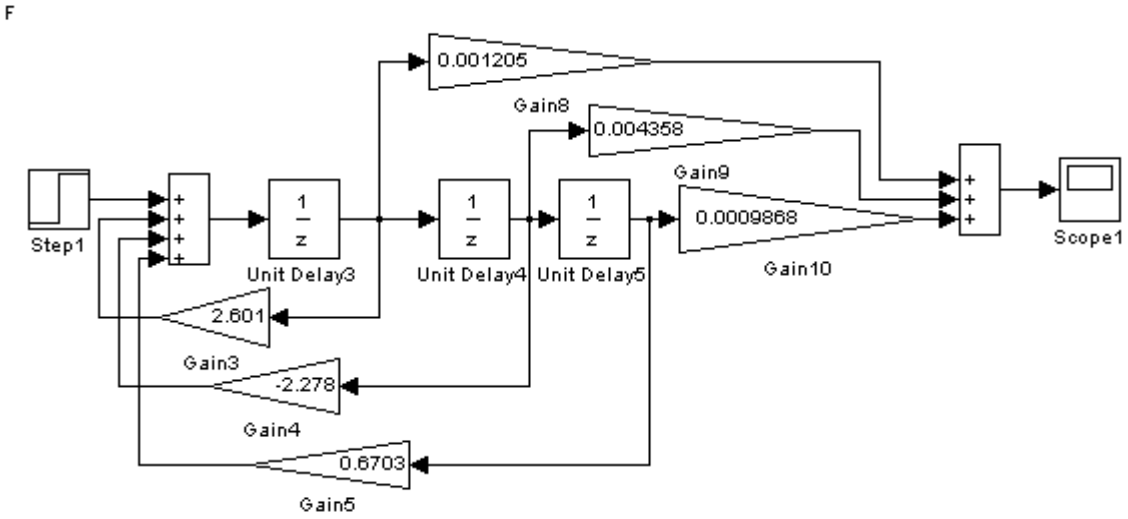


Рисунок 4.9 Структурные схемы ФНЧ (A, C, E - непрерывная передаточная функция в форме tf фильтров I, II, III порядков соответственно; B, D, F - реализация их Z-передаточной функции на регистрах сдвига)

ЗАДАНИЕ

1. Исследовать характеристики фильтров, реализовав программы 1 и 2.

2. По заданным преподавателем параметрам составить программу расчета дискретных фильтров Баттерворта.

№	Вихід. сигнал	Тип фільтра	Прототип	Частотні характеристики Bpass, Bstop, кГц	Apass, дБ	Astop, дБ	Реаліза-ція
1	Speak3	ФНЧ	Батерворта	7.0, 8.0	0.1	15	Каскадна
2	Speak4	ФНЧ	Батерворта	5.0, 6.0	0.1	15	Паралельн
3	Speak5	ФВЧ	Батерворта	8.0, 7.0	0.1	15	Каскадна
4	Speak3	ФВЧ	Батерворта	8.0, 7.0	0.1	15	Паралельн
5	Speak3	ФНЧ	Батерворта	7.2, 8.3	0.1	15	Каскадна
6	Speak2	ФНЧ	Батерворта	5.2, 6.0	0.1	15	Паралельн
7	Speak1	ФВЧ	Батерворта	8.0, 7.2	0.1	15	Каскадна
8	Speak3	ФВЧ	Батерворта	8.0, 7.2	0.1	15	Паралельн
9	Speak2	ФНЧ	Батерворта	7.0, 8.0	0.1	15	Каскадна
10	Speak1	ФНЧ	Батерворта	5.0, 6.0	0.1	15	Паралельн

3. Выполнить программу и построить логарифмические характеристики дискретных фильтров.

4. По передаточным функциям дискретных фильтров в пакете Simulink составить структурные схемы.

5. В пакете Simulink промоделировать структурные схемы дискретных

фильтров.

ЛИТЕРАТУРА

1. Гольденберг Л.М., Матюшкин Б.Д., Поляк М.Н. Цифровая обработка сигналов: Учебное пособие для вузов. – М.: Радио и связь, 1990. – 256 с.
2. Лазарев Ю.Ф. MATLAB 5.x. – К.: Издательская группа BHV, 2000. – 384 с.
3. Дьяконов В. П. MATLAB 6/6.1/6.5 + Simulink 4/5. Обработка сигналов и изображений. М.: Солон-Пресс. — 2004.
4. Черных И.В. S1MULINK. Среда создания инженерных приложений. М.: ДИАЛОГ-МИФИ. - 2003.

Лабораторная работа № 5
РАСЧЕТ ЦИФРОВЫХ ФИЛЬТРОВ В СРЕДЕ MATLAB

Цель: Изучить возможные способы расчета цифровых фильтров в среде MATLAB.

В среде MATLAB цифровые фильтры можно рассчитать по меньшей мере тремя способами:

1) в командном окне;
2) с помощью пакета **sptool**;
3) с помощью пакета **fdatool**.

5.1 Расчет цифровых фильтров в командном окне

5.1.1 Расчет коэффициентов a_k нерекурсивного фильтра с помощью функции fir1.

Функция **fir1** реализует вычисления по методу обратного преобразования Фурье с использованием окон:

a=fir1(n,Wn,'ftype',window,'normalization')

Здесь:

n – порядок фильтра – целое четное число (кол-во коэффициентов фильтра равно n+1);

Wn – относительная.частота среза (по отношению к частоте Найквиста, равной половине частоты дискретизации Fd) – число в диапазоне (0,1); является вектором из двух чисел, если фильтр полосовой или режекторный;

'ftype' – тип фильтра (если отсутствует или **'low'**– ФНЧ; **'high'** – ФВЧ; **'bandpass'** – полосовой; **'stop'** – режекторный;

window – вектор-столбец из **n+1** элементов (по умолчанию применяется окно Хэмминга **hamming(n+1)**);

'normalization' – нормировка АЧХ и ИПХ (по умолчанию значение **'scale'** – единичное значение АЧХ в центре полосы пропускания; **'noscale'** – нормировка не производится).

Пример:

```
window=rectwin(7)        % синтез прамоугольного окна из 7 отсчетов
a=fir1(6,0.5,window)     % расчет коэфф-в КИХ-фильтра с нормализацией
```

Результат:

a=[-0.1148 0.0000 0.3443 0.5409 0.3443 0.0000 -0.1148]

Сравнивая эти результаты с рассчитанными вручную коэффициентами, нетрудно видеть разницу. Например, ручные расчеты дают $a_0 = 0.5$, тогда как в MATLAB мы получили $a_0 = 0.5409$. Естественно предположить, что причиной тому проводимая по умолчанию нормировка ИПХ. Проверяем это предположение, задавая в программе значение **'noscale'** для параметра нормализации:

```
window=rectwin(7)          % синтез прамоугольного окна из 7 отсчетов
a=fir1(6,0.5,window,'noscale')   % расчет коэфф-в КИХ-фильтра без
нормализации
```

Результат:

a=[-0.1061 0.0000 0.3183 0.5000 0.3183 0.0000 -0.1061]

5.1.2 Расчет коэффициентов a_k и b_k рекурсивного фильтра

Команда генерирования коэффициентов a_k и b_k рекурсивного фильтра Баттерворта:

[a,b] = butter(n,Wn,'ftype')

Смысл обозначений тот же, что и для нерекурсивного фильтра.

Если **Wn** является двухэлементным вектором, т.е. **Wn = [w1 w2],** функция **butter** возвращает коэффициенты полосового фильтр порядка **2n** с полосой пропускания (или задерживания – для режекторных фильтров) **w1 < w < w2.**

5.1.3 Построение графиков АЧХ и ФЧХ

Осуществляется командой

freqz(a,b)

или

freqz(a,b,N) % N – число отсчетов АЧХ и ФЧХ

Графики АЧХ и ФЧХ строятся в диапазоне 0 – 1 нормированных (по частоте Найквиста Fd/2) значений частоты частот (рисунок 5.1).

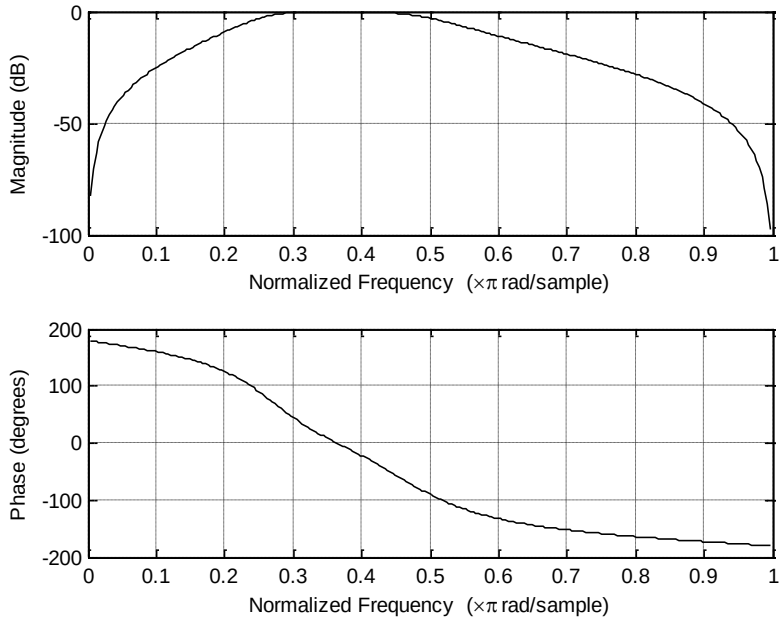


Рисунок 5.1 АЧХ и ФЧХ полосового фильтра Баттерворта 2-го порядка для w1=0.25, w2=0.5

5.1.4 Построение графика ИПХ

Построение графика ИПХ осуществляется парой команд:

[h,t] = impz(a,b);

stem(t,h)

При этом в случае рекурсивного фильтра количество выводимых отсчетов ИПХ выбирается автоматически (рисунок 5.2).

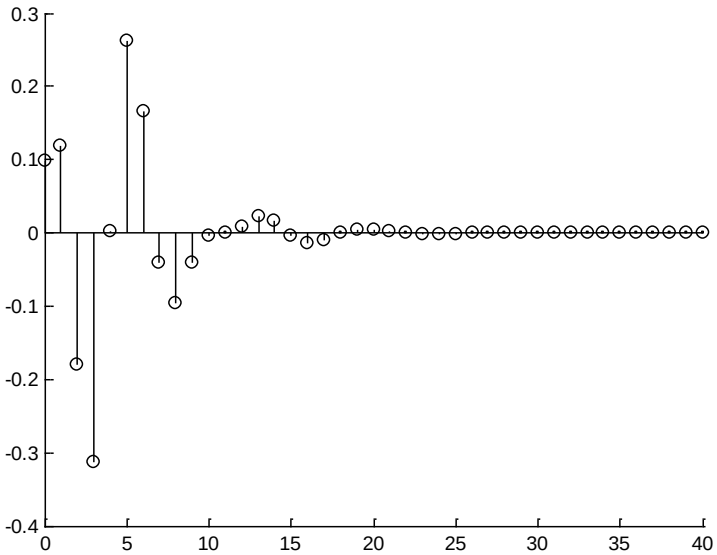


Рисунок 5.2 ИПХ полосового фильтра Баттерворта 2-го порядка для w1=0.25, w2=0.5

Пара команд

```
[h,t] = impz(a,b,N);     % N – число отсчетов ИПХ
stem(t,h)
```

позволяет вывести на график N отсчетов ИПХ.

5.1.5 Моделирование процесса цифровой фильтрации

Такое моделирование производится в 3 этапа:

1) моделирование входного процесса x_n и построение его графика;

2) цифровая фильтрация процесса x_n, в результате которой получаем процесс y_n;

3) построение графика процесса y_n и сопоставление его с графиком процесса x_n.

5.1.5.1. Моделирование входного процесса x_n и построение его графика

Моделирование отрезка входного процесса x_n в виде гармонического сигнала частотой $f0$, длительностью T с, дискретизированного с частотой Fd:

```
t=0:1/Fd:T;
x=sin(2*pi*f0*t);
```

Построение графика процесса x_n:

```
plot(t,x);
```

Моделирование отрезка входного процесса s_n в виде прямоугольных импульсов единичной высоты, следующих с частотой $f0$, длительностью $\tau < \frac{1}{f0}$ с, дискретизированного с частотой Fd:

```
t=0:1/Fd:T;
s=(square(2*pi*f0*t, f0*tau*100)+1)/2;
```

Моделирование отрезка шума r_n в виде (0;1)-нормального белого шума той же длительности T с, дискретизированного с частотой Fd:

```
r=randn(1,length(t));
```

Моделирование отрезка входного процесса в виде аддитивной смеси гармонического сигнала единичной амплитуды и (0;1)-нормального белого шума:

xr=x+r;

5.1.5.2 Два способа фильтрации в командном окне

Два способа:

1) с помощью функции свертки:

y=conv(x,a);

2) с помощью функции дискретной фильтрации:

y=filter(a,b,x)

Особенности фильтрации с помощью функции свертки:

1) можно работать только с КИХ-фильтрами;
2) длина отклика равна сумме длин воздействия и ИПХ минус единица

Особенности фильтрации с помощью функции дискретной фильтрации:

1) можно работать как с КИХ-, так и с БИХ фильтрами;
2) длина отклика равна длине воздействия.

5.2 Расчет фильтра a_k с помощью пакета sptool

5.2.1 Интерактивная оболочка SPTool

Процедура SPTool активизирует графическую интерактивную оболочку пакета Signal, включающую:

- средство поиска и просмотра сигналов – Signal Brouser;
- 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 рисунке 5.3.

Как видим, окно **SPTool** состоит из трех областей – **Signals** (Сигналы), **Filters** (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.

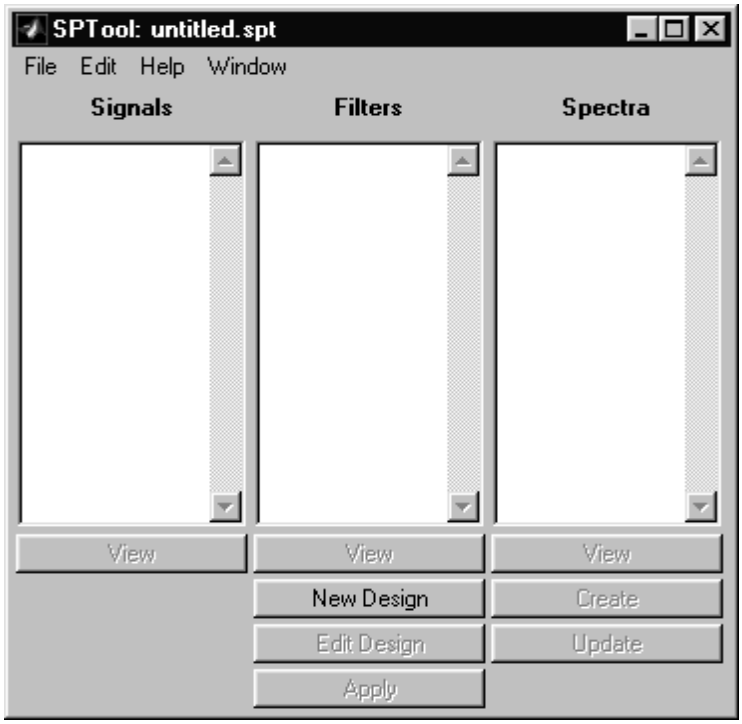


Рисунок 5.3

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 Signal (кнопка 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.

5.2.2 Пример

Для активизации пакета набираем в командном окне команду **sptool.** Затем в появившемся окне в колонке кнопок “Filters” необходимо нажать кнопку “New”.

В появившемся окне «Filter Designer» необходимо:

1) задать частоту дискретизации (задаем 100 Гц);
2) выбрать в позиции **Algoritm** значение **Kaiser Window FIR** (выбираем из 3-х вариантов: **Equiripple FIR**, **Least Square FIR** и **Kaiser Window FIR**);
3) отключить флажок **Minimum Order**;
4) задать **Order=6**;
5) задать **Type=lowpass;**
6) задать **Passband Fp=25**;
7) отключить флажок **Autodesigne**;
8) закрыть окно **Filter Designer**;
9) в окне **sptool** в колонке **Filters** нажать кнопку **View**;
10) в появившемся окне **Filter Viewer** наблюдаем графики АЧХ, ФЧХ, ИПХ (ИПХ наблюдаем после активизации соответствующего флажка).

Примечание 1: мы выбрали в позиции **Algoritm** значение **Kaiser Window FIR**. Кроме данного алгоритма, есть еще два алгоритма: **Equiripple FIR**, **Least**

Square FIR. Из всех этих 3-х алгоритмов только алгоритм **Кайзера** реализует *метод обратного преобразования Фурье* с весовым окном Кайзера. При значении параметра 0 окно Кайзера превращается в обычное прямоугольное окно.

Как показывает эксперимент, рассчитанные таким образом коэффициенты ФНЧ оказываются **ненормированными**, т.е. будут в точности равными вычисленным вручную.

Примечание 2: чтобы узнать значения коэффициентов, нужно:

1) активизировать график ИПХ, щелкнув по нем мышкой;

2) активизировать вертикальные маркеры (кнопкой, расположенной под меню);

3) поместить один из маркеров (всего имеется 2 маркера – 1-й изображается сплошной вертикальной линией. 2-й - пунктирной) напротив нужного отсчета ИПХ;

4) считать значение отсчета ИПХ в специальном окошке.

На рисунке 5.4 показано окно **Filter Viewer** (в режиме предпечатного просмотра) для данного конкретного случая (маркер 1 установлен против максимального отсчета, равного 0.5, а маркер 2 – против смежного отсчета, равного 0.3183).

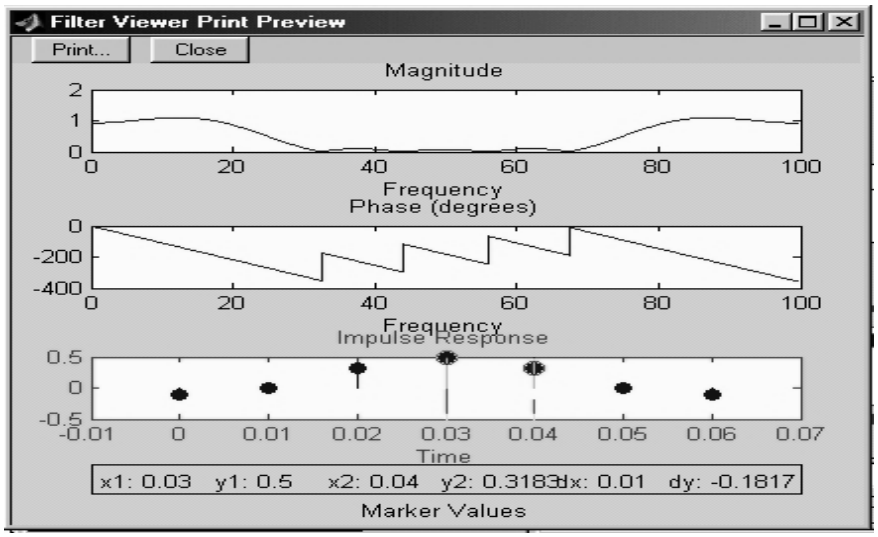


Рисунок 5.4

Пакет **sptool** позволяет моделировать процесс фильтрации с помощью рассчитанного фильтра. Для этого в среду пакета **sptool** (как уже было сказано) нужно импортировать входной сигнал, сгенерированный в рабочем окне программы MATLAB. Фильтрация происходит после нажатия кнопки Apply.

5.3 Расчет фильтра a_k с помощью пакета fdatool

Для активизации пакета нужно в командном окне набрать команду **fdatool.** Затем в появившемся окне в разделе **Designe Filter** задать:

1) Designe Method: FIR=Window;
2) Window Specifications: Window=Rectangular;
3) Filter order: Specify order=6;
4) задать частоту дискретизации (Fs=100 Гц);
5) задать Filter Type=lowpass;
6) задать Passband Fc=25;
7) с помощью кнопок под меню включаем режим просмотра коэффициентов фильтра.

В результате проведенных расчетов убеждаемся, что здесь по умолчанию производится нормирование ИПХ (см. выше **fir1**).

Пакет **fdatool** не позволяет моделировать процесс фильтрации с помощью рассчитанного фильтра. Однако рассчитанные коэффициенты фильтра можно импортировать из среды **fdatool** в среду MATLAB, где их можно использовать для моделирования фильтрации.

5.4 Сопоставление способов расчета

Расчеты в командном окне: достоинство – гибкость управления расчетами; недостаток – сложный синтаксис команд; особенность – коэффициенты фильтра по умолчанию являются нормированными, хотя заданием значения ‘noscale’ параметра ‘normalization’ можно отказаться от нормирования.

Расчеты с помощью пакета sptool: достоинство – не нужно помнить синтаксис команд; недостаток – ограниченный набор окон для метода обратного преобразования Фурье (только окно Кайзера); особенность – вычисляются только ненормированные коэффициенты фильтра.

Расчеты с помощью пакета fdatool: достоинство – не нужно помнить синтаксис команд; особенность – вычисляются только нормированные коэффициенты фильтра.

ЗАДАНИЕ

1. Ознакомиться с демонстрационными примерами системы MATLAB (обязательно: «Фильтрация сигнала» (раздел «Обработка сигналов»), «Адаптивная фильтрация RLS подавитель шума» (раздел «Блок-схемы DSP»).
2. Проработать все примеры, наведенные в данной лабораторной работе.

ЛИТЕРАТУРА

1. Гольденберг Л.М., Матюшкин Б.Д., Поляк М.Н. Цифровая обработка сигналов: Учебное пособие для вузов. – М.: Радио и связь, 1990. – 256 с.
2. Лазарев Ю.Ф. MATLAB 5.x. – К.: Издательская группа BHV, 2000. – 384 с.
3. Дьяконов В. П. MATLAB 6/6.1/6.5 + Simulink 4/5. Обработка сигналов и изображений. М.: Солон-Пресс. — 2004.
4. Ануфриев И. Е. Самоучитель MatLab 5.3/б.х.— СПб.: БХВ-Петербург, 2002. — 736 с.
5. Дьяконов В. П. Matlab 6/6.1/6.5+Simulink 4/5. Основы программирования: Руководство пользователя. — М.: Солон-Пресс, 2002. — 768 с.
6. Мэтьюз Д., Финк К. Численные методы. Использование MATLAB (3-е издание). — СПб.: Вильяме, 2001. — 720 с.
7. Чен К., Джиблин П., Ирвинг А. MATLAB в математических исследованиях. — М.: Мир, 2001. — 346 с.
8. Черных И. В. Simulink: среда создания инженерных приложений.— М.: Диалог-МИФИ, 2004. — 496 с.