МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное
учреждение высшего образования
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

В. Н. Бондарев

ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ В SCILAB

Методические указания
к лабораторным работам по дисциплине
«Технологии обработки мультимедиа данных»
для студентов дневной и заочной форм обучения направления
09.04.02 «Информационные системы и технологии»

Севастополь
СевГУ
2025

УДК 004.932(076.5)
ББК 32.973-018.2я73
Б811

Р е ц е н з е н т ы :
В. С. Чернега – канд. техн. наук, доцент кафедры
«Информационные системы»
А. А. Брюховецкий – канд. техн. наук, доцент кафедры
«Информационные технологии и компьютерные системы»

Ответственный за выпуск:
И. П. Шумейко – канд. физ.-мат. наук, доцент,
заведующий кафедрой «Информационные системы»

Бондарев, В. Н.

Б811 Цифровая обработка сигналов в Scilab : методические указания к лабораторным работам по дисциплине «Технологии обработки мультимедиа данных» для студентов дневной и заочной форм обучения направления 09.04.02 «Информационные системы и технологии» / В. Н. Бондарев ; Севастопольский государственный университет. — Севастополь : СевГУ, 2025. – 79 с. – Текст : электронный.

Целью методических указаний является оказание помощи студентам в изучении процессов обработки одномерных и двумерных цифровых сигналов различной модальности в среде моделирования Scilab. Излагаются теоретические и практические сведения, необходимые для выполнения лабораторных работ, содержатся варианты заданий, программа исследований и требования к содержанию отчета.

УДК 004.932(076.5)
ББК 32.973-018.2я73

Рассмотрены и утверждены на заседании кафедры «Информационные системы», протокол № 8 от 29 апреля 2025 г.

Рассмотрены и рекомендованы к изданию на заседании Учёного Совета Института информационных технологий, протокол № 5 от 20 мая 2025 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Дисциплина «Технологии обработки мультимедиа данных» (ТОМД) является дисциплиной, формируемой участниками образовательных отношений.

В ходе изучения дисциплины используются знания, полученные студентами при освоении следующих дисциплин бакалавриата: высшая математика, теория вероятностей и математическая статистика, информатика, технологии программирования.

Целью дисциплины является формирование профессиональных компетенций в области цифровой обработки мультимедиа сигналов с целью их дальнейшего использования мультимодальными нейронными сетями.

Задачи дисциплины:

- расширение и углубление знаний студентов компьютерных направлений подготовки в области теории дискретных сигналов и систем;
- расширение знаний в области цифрового спектрального анализа;
- обучение технологиям обработки речи и аудио сигналов;
- обучение технологиям обработки изображений и видеосигналов;
- ознакомление со стандартами сжатия речи, аудио, изображений и видео.

Лабораторный практикум по дисциплине включает восемь лабораторных работ, для выполнения которых отводится 36 академических часов.

Лабораторные работы выполняются в среде численного моделирования Scilab, распространяемой по лицензии свободного программного обеспечения с открытым исходным кодом (GNU GPL 2).

Порядок выполнения лабораторных работ. В лабораторных работах студент решает заданную индивидуальным вариантом задачу. Варианты заданий приведены к каждой лабораторной работе. Номер варианта вычисляется как остаток от деления номера студента в списке группы на 15. Например, номер студента в списке группы 17. Тогда остаток от деления 17 на 15 будет равен 2. Следовательно, студент выбирает вариант 2.

Лабораторная работа выполняется в два этапа. На первом этапе — этапе самостоятельной подготовки — студент должен выполнить следующее:

– проработать по конспекту и рекомендованной литературе, приведенной в конце настоящих методических указаний, основные теоретические положения лабораторной работы и ответить на контрольные вопросы;

– разработать алгоритм решения задачи, руководствуясь рекомендациями, приведенными в методических указаниях;

– написать программу на языке пакета Scilab и отладить ее;

– оформить результаты в виде заготовки отчета по лабораторной работе.

На втором этапе студент должен продемонстрировать выполнение работы преподавателю и защитить отчет по лабораторной работе.

Лабораторная работа № 1

ИССЛЕДОВАНИЕ ФУНКЦИЙ ПАКЕТА SCILAB ДЛЯ ОБРАБОТКИ И ВИЗУАЛИЗАЦИИ ДАННЫХ

1.1 Цель работы

Изучение среды численного моделирования Scilab и ее базовых функций генерации, обработки и отображения сигналов, приобретение практических навыков моделирования в среде Scilab.

1.2 Краткие теоретические сведения

Scilab – это кроссплатформенный свободно распространяемый математический пакет, обладающий сходным с Matlab синтаксисом встроенного языка. Пакет программ Scilab предназначен для выполнения математических вычислений и численного моделирования широкого спектра информационных и управляющих систем.

Пакет можно скачать по адресу https://www.scilab.org/download/scilab-2024.1.0. После установки пакета Scilab для получения справочных сведений о его возможностях и командах необходимо выбрать пункт меню «Справка» или нажать кнопку F1. Откроется окно «Справочная система» (рисунок 1.1), в котором можно выполнять поиск необходимых сведений как с использованием разделов содержания справки, так и по ключевым словам.

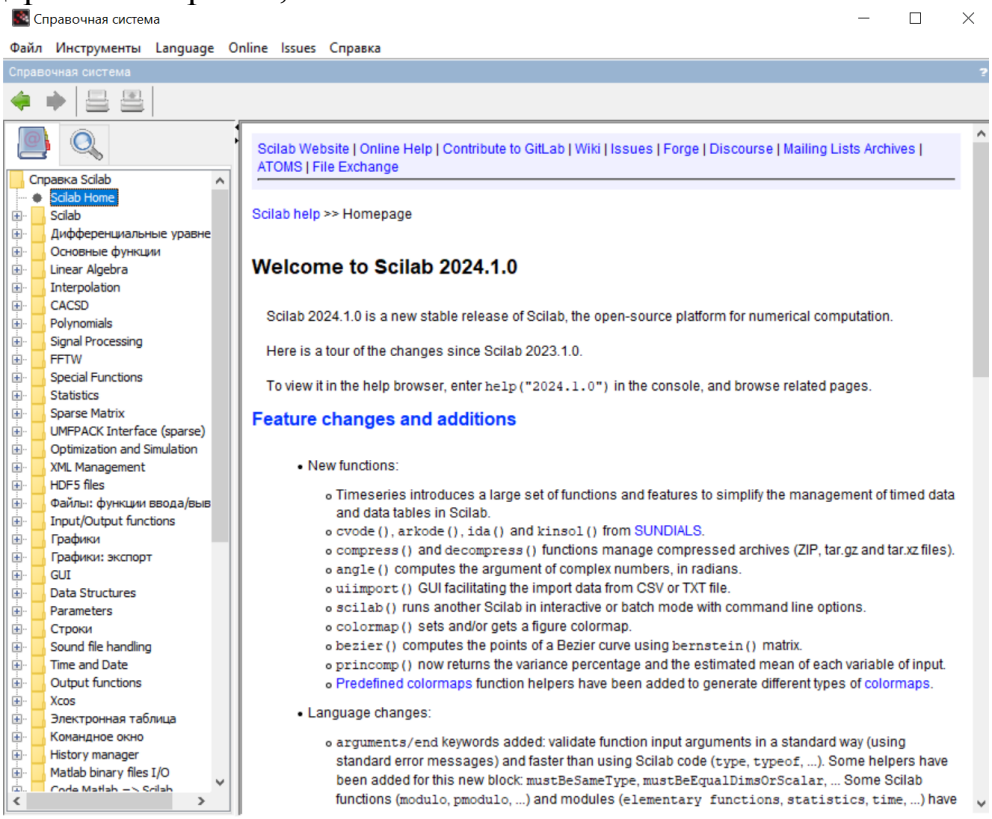

Рисунок 1.1 – Справочная система Scilab

Пакет Scilab позволяет выполнять команды в интерактивном режиме. Порядок ввода команд аналогичен пакету Матлаб. При этом используются следующие специфические обозначения:

// - начало однострочного комментария
% - начальный символ имени **системных переменных**, например:
%i - мнимая единица ($\sqrt{-1}$);
%pi - число π = 3.141592653589793;
%e - исло e = 2.7182818;
%inf - машинный символ бесконечности (∞);
%NaN - неопределенный результат (0/0,∞/∞ и т. п.);
%eps _условный ноль %eps=2.220E-16;
%t, %f – логические константы True и False.

Системные переменные используются в качестве констант в выражениях и не могут быть изменены пользователем. Ниже представлен пример ввода выражения, содержащего системные переменные и встроенные функции, и вычисления его значения в командном окне Scilab:

```
--> y= sin(%pi/4)+%e^2
 y  =
   8.0961629
```

Scilab имеет расширенный перечень встроенных основных функций (см. п. меню «Справка»—> Основные функции). Клавиша Tab может использоваться для автозавершения ввода имени функции или переменной (рисунок 1.2).

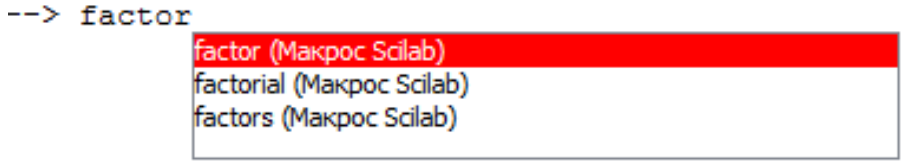

Рисунок 1.2 – Пример автозавершения имени функции

Ввод в программу исходных данных с клавиатуры выполняется вызовом функции **x = input(message [, "string"])**, где **message** – строка, которая отображается на экране перед выполнением ввода, необязательный параметр **"string"** указывается при вводе строки, x – вещественное число или строка (если указан параметр **"string"**).

Для вывода на экран результатов вычислений может использоваться функция **disp**. Если требуется вывести комбинацию строковых значений и чисел, то требуется предварительно преобразовать числа в строки с помощью функции **string:**

```
--> t=19
--> disp("Средняя температура "+string(t)+" градусов")
 Средняя температура 19 градусов
```

Для форматированного вывода на экран значений переменных может использоваться функция, аналогичная функции **printf** языка Си. Например:

```
--> printf("Результат: exp(3) = %f",exp(3))
Результат: exp(3) = 20.085537
```

Ввод-вывод данных из файлов и в файлы выполняется в Scilab с использованием функций mopen, mfscanf, mfprintf, которые полностью аналогичны функциям языка Си: **fopen, fscanf, fprintf.**

При программировании в Scilab необходимо различать понятия **сценарий** и **функция, определяемая пользователем.**

Сценарием называют любую программу Scilab. Любой сценарий состоит из последовательности **инструкций** (команд), которые описывают конкретные действия с объектами Scilab. Сценарии создаются в окне текстового редактора SciNotes, который может быть вызван из пункта меню «Инструменты». Сценарии сохраняются в файлах с расширением *.sce.

Функция, определяемая пользователем, также является сценарием, но характеризуется наличием имени, благодаря которому к ней можно обратиться из любого места сценария. Имеется два способа определения функций: function и deff. Определение функции с использованием конструкции function аналогично большинству языков программирования. Например:

```
function z=f(x,y)
        z=1/(exp(-y)+exp(-x))
endfunction
```

Обычно такое определение функции сохраняется в файле ***.sci** на диске (например, **f.sci**). Чтобы использовать функцию, необходимо загрузить её из файла в оперативную память. Для этого необходимо выполнить команду **exec.** Её единственным аргументом является строка с указанием положения сохраненного файла в файловой системе. Например, если файл **f.sci** был сохранен в домашнем каталоге пользователя, то команда загрузки будет выглядеть следующим образом: **exec('~/f.sci')**. После этого функция, описанная в файле, может быть вызвана в текущей сессии, например, **f(1,2)**.

Определение функции с использованием оператора **deff** реализуется в соответствии с шаблоном:

```
deff('[имя1,...,имяN]=имя_функции(переменная_1,...,переменная_M)',
        'имя1=выражение1;...;имяN=выражениеN'),
```

где список **[имя1,...,имяN]** представляет имена выходных значений функции, **выражение1,…, выражениеN** – правила вычисления этих значений. Листинг ниже демонстрирует определение функции **z=f(x,y)** с помощью оператора **deff** и использование вызова **f** для вычисления значения функции:

```
--> deff('z=f(x,y)', 'z=1/(exp(-y)+exp(-x))');
```

```
--> a=1; b=2; c=f(a,b)
 c  =
  1.9872232
```

Scilab хранит установки, функции и переменные в рабочей области, называемой Workspace. Переменные рабочей области отображаются в окне *обозревателя переменных*. Для очистки рабочей области от всех пользовательских переменных используется команда **clear.**

Система Scilab предназначена для выполнения математических вычислений. Обозначение математических операций и управляющих конструкций (**if, for, while** и др.) такое же, как и в системе Matlab [4]. Рассмотрим примеры некоторых операций с одномерными (векторы) и двумерными массивами (матрицы).

Вектор-строка задается своими элементами, которые записываются внутри квадратных скобок и разделяются пробелами или запятыми. Для задания вектора-столбца его элементы следует разделять точкой с запятой (**;**). Например:

```
--> xvec=[1 2 3] // вектор-строка
 xvec  =
  1.   2.   3.
--> yvec=[1;2;3] // вектор-столбец
 yvec  =
  1.
  2.
  3.
```

Векторы удобно создавать с помощью оператора “ **:** ”. Например, если требуется создать вектор из нечетных чисел от 1 до 10, то можно применить оператор:

```
--> x=1:2:10  // Хнач:Шаг:Хкон
 x  =
  1.   3.   5.   7.   9.
```

Переменную заданную как массив можно использовать в арифметических выражениях в качестве аргумента стандартных математических функций. Результатом выполнения таких операторов являются массивы.

Для определения длины вектора используется функция **length**. Для доступа к i-му элементу вектора применяется команда **x(i)**, а чтобы получить элементы вектора со 2-го по 4-й, используется индексация сегментом 2:4

```
--> x(2:4)
 ans  =
  3.   5.   7.
```

Матрица представляется в виде набора векторов-строк, разделяемых точкой с запятой:

```
--> m=[1 2 3;4 5 6]
 m  =
   1.   2.   3.
   4.   5.   6.
```

Размер матрицы определяется с помощью функции **size**

```
--> size(m)
 ans  =
   2.   3.
```

Для извлечения строки или столбца матрицы используют оператор "**:**", например:

```
--> m(2,:)      #возвращает 2-ю строку матрицы
 ans  =
   4.   5.   6.
--> m(:,3)     #возвращает 3-й столбец матрицы
ans  =
   3.
   6.
```

Матрицы и векторы можно формировать, составляя их из ранее заданных матриц и векторов:

```
--> v1=[1 2 3]; v2=[4 5 6]; v3=[7 8 9];
--> //Горизонтальная конкатенация векторов–строк:
--> V=[v1 v2 v3]
  V = 1 2 3 4 5 6 7 8 9
-->//Вертикальная конкатенация векторов–строк,
-->//результат матрица:
--> V=[v1; v2; v3]
  V =
  1 2 3
  4 5 6
  7 8 9
-->//Горизонтальная конкатенация матриц:
--> M=[V V V]
  M =
  1 2 3 1 2 3 1 2 3
  4 5 6 4 5 6 4 5 6
  7 8 9 7 8 9 7 8 9
-->//Вертикальная конкатенация матриц:
--> M=[V;V]
  M =
  1 2 3
  4 5 6
  7 8 9
  1 2 3
  4 5 6
  7 8 9
```

Для работы с матрицами в Scilab применяются матричные операции: “+” — сложение; “–“ — вычитание; “ ‘ “— транспонирование; “ * “— матричное умножение; “^”— возведение в степень; “\”— левое “деление”; “/ “— правое “деление”; “.* ” — поэлементное умножение матриц; “.^”—_ поэлементное возведение в степень; “.\”— поэлементное левое деление; “./”— поэлементное правое деление. Эти операции аналогичны операциям Matlab [4].

Примеры некоторых действий над матрицами:

```
-->A=[1 2 0;-1 3 1;4 -2 5];
-->B=[-1 0 1;2 1 1;3 -1 -1];
-->//Вычислить (A^T+B)^2 - 2A(0.5B^T-A)
-->(A'+B)^2-2*A*(1/2*B'-A)
ans =
  10. 8. 24.
  11. 20. 35.
  63. - 30. 68.
--> //Решить матричные уравнения A·X=B и X·A=B.
-->A=[3 2;4 3];
-->B=[-1 7;3 5];
-->//Решение матричного уравнения AX=B: решение X=A^-1 B
-->X=A\B
X =
  - 9. 11.
  13. - 13.
-->//Решение матричного уравнения XA=B: решение X=B A^-1
-->X=B/A
X =
  - 31. 23.
  - 11. 9.
-->X*A-B // проверка
ans =
  0. 0.
  0. 0.
```

Для упрощения работы с матрицами и векторами (массивами) в Scilab существуют специальные функции: **ones(m,n)** – создает матрицу из единиц; **zeros(m,n)** – создает матрицу из нулей; **eye(m,n)** - создает единичную матрицу. Функции для вычисления различных числовых характеристик массивов:

sum – сумма элементов массива;

prod – произведение элементов массива;

max, min – максимальное и минимальное значение массива;

mean, median – среднее и медианное значение массива;

det – определитель квадратной матрицы;

rank – ранг матрицы;

norm – норма квадратной матрицы;

cond – число обусловленности матрицы (произведение нормы исходной матрицы на норму обратной матрицы);

spec – возвращает *собственные значения* и *собственные векторы* квадратной матрицы;

inv – вычисляет обратную матрицу;

pinv – вычисляет псевдообратную матрицу;

linsolv – решает систему линейных алгебраических уравнений;

svd – выполняет сингулярное разложение матрицы и др.

Построение двумерного графика в отдельном графическом окне в координатах x,y осуществляется по команде **plot(x,y)**. Команда позволяет также задавать цвет и стиль изображения точек на графике, например команда **plot(1,2,"*r")** отобразит звездочку красным цветом в позиции с координатами (1,2). Для отображения графика функции вида *y=f(x)* удобно значения *x* задавать в виде вектора, который формируется с помощью вызова функции **x=linspace(a,b,n)** , где параметры a и b задают интервал определения функции, а n – число точек разбиения этого интервала. На рисунке 1.3 приведен пример программы, определяющей функции *f(x)* и *g(x)* и отображающей их графики в одной системе координат, соответственно, красным и зеленым цветом с помощью вызова функции **plot**:

```
function y=f(x)
        y=(x^2+2*x)*exp(-x/2)
endfunction
function y=g(x)
        y=cos(2*x)*exp(-x/2)
endfunction
x=linspace(-2,10,100);
plot(x,f,"r",x,g,"g")
xgrid //отображение сетки
// отображение надписей
xtitle('Функции f(x) и g(x)', 'Позиция','Ускорение')
//легенда: 4 – правый нижний угол, %t - рамка
legend('f(x)','g(x)',4,%t)
```

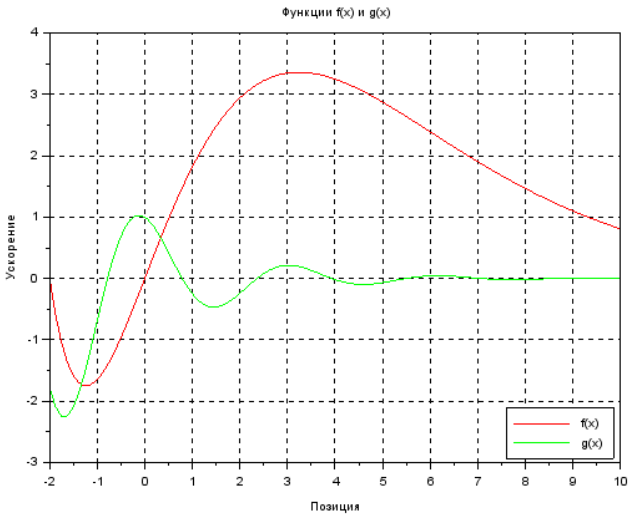

Рисунок 1.3 – Построение двумерных графиков с помощью plot

Функция **subplot(m,n,p)** разбивает графическое окно на **m** окон по вертикали и **n** окон по горизонтали, текущим окном становится окно с номером **p**. Применяется для отображения нескольких графиков – каждый в своем подокне.

Команда **figure(i)** делает активным i-ое окно. Для очистки текущего графического окна применяется функция **clf**. Функция **xdel(i)** закрывает графические окна, где **i** – вектор номеров окон.

Scilab позволяет отображать поверхности и кривые в 3-х мерном пространстве с помощью функции **surf**. Эта функция имеет три параметра: **x, y** и **z**. Векторы **x** и **y**, размерами m и n, определяют координатные точки по осям *Ox* и *Oy*. Матрица **z**, размером nxm, хранит высоты в координатных узлах. Для отображения поверхности, определяемой функцией **z=f(x,y)**, необходимо предварительно определить **z** во всех координатных узлах с помощью вызова функции **feval(x,y,f)**.

Функция **feval** возвращает транспонированную матрицу, размером mxn. Поэтому при вызове функции **surf** матрицу **z** требуется транспонировать. Удобнее 3-d поверхности строить с использованием функций **plot3d** и **plot3d**1. Сравнительный пример построения 3-D поверхности с помощью указанных функций представлен на рисунке 1.4.

```
function z=f(x, y)
  z=3*x^2+y^2;
endfunction
x=linspace(-1,1,50);
y=linspace(-2,2,100);
z=feval(x,y,f);
clf
figure(1)
subplot(1,3,1)
surf(x,y,z')
subplot(1,3,2)
plot3d(x,y,z)
subplot(1,3,3)
plot3d1(x,y,z)
```

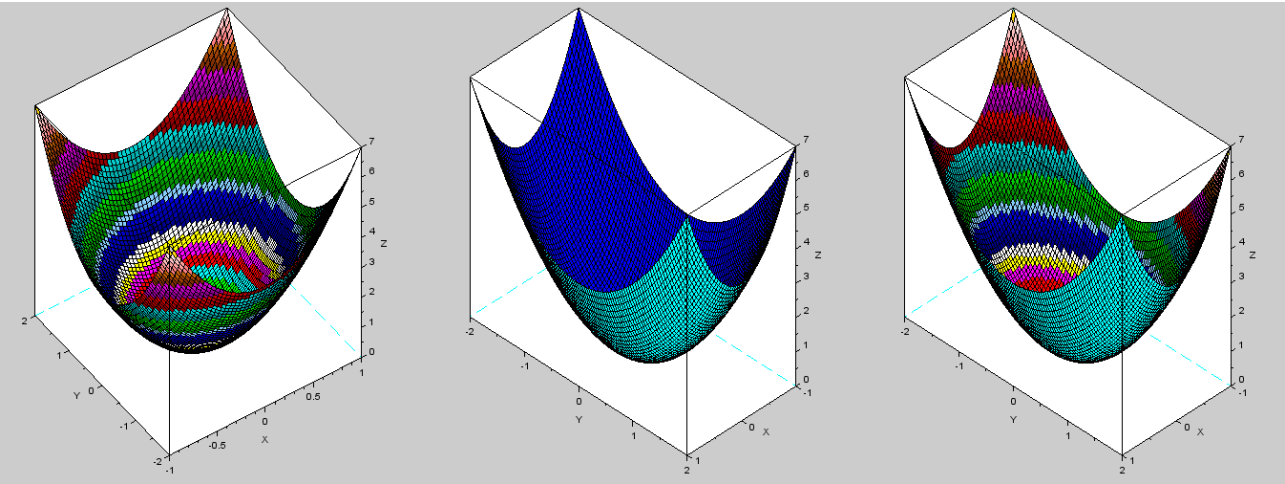

Рисунок 1.4 – 3-D поверхности, построенные функциями surf, plot3d, plot3d1

В Scilab имеются и другие функции для построения 2-х и 3-х мерных графиков, например: **plot2d, mesh, contour, plot3d2, plot3d3, param3d** и др.

Scilab позволяет выполнять статистическую обработку данных в массивах. Функция **mean(M)** вычисляет среднее всех элементов матрицы **M**, **mean(M,'r')** – вычисляет среднее по строкам, а **mean(M,'c')**- среднее по столбцам. Функция **stdev(M)** вычисляет стандартное отклонение, а функция **variance(M)** – дисперсию всех элементов массива **M**. Специальная функция **tabul(M)** вычисляет частоты появления значений в массиве **M**. Для генерации псевдослучайных чисел могут использоваться функции **grand** и **rand**. Функция **rand(n1,n2,…nn[, p])** - формирует многомерную матрицу *случайных чисел*, необязательный параметр **p** - это стро-

ковая переменная, с помощью которой можно задать тип распределения случайной величины (**'uniform'** - равномерное , **'normal**'- гауссовское). На рисунке 1.5 изображен листинг программы для генерации матрицы размером 10x10 из нормально распределенных случайных чисел с нулевым средним и единичной дисперсией, а также построения гистограммы распределения.

```
--> x=rand(10,10,'normal');
--> mean(x)
 ans  =
  -0.0218062
--> variance(x)
 ans  =
   1.1118125
--> histplot(12,x);
```

.

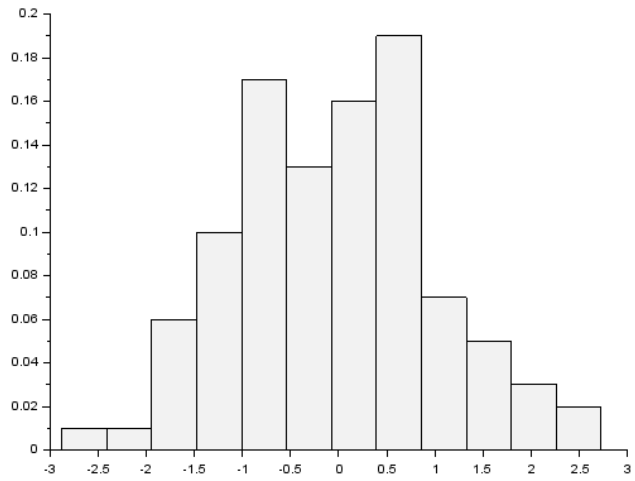

Рисунок 1.5 – Пример использования функций rand и histplot

В функции **grand** доступны улучшенные генераторы случайных чисел в том смысле, что они имеют более длинный период и более лучшие статистические свойства. Кроме этого, функция **grand** обладает большими функциональными возможностями, в частности, позволяет генерировать случайные числа с различными законами распределения. Ниже приведены некоторые примеры вызова функции **grand**:

```
x=grand(400,500,'nor',0,1); //нормальный распределение
x=grand(1,1000,'unf',-1,1); // равномерное распред., числа в диапазоне [-1,1]
x=grand(1,1000,'exp',3);     // экспоненциальное распределение, лямбда=3
x=grand(1,5000,'chi',10);    // распред. хи квадрат с 10-ю степенями свободы
x=grand(1,5000,'poi',2);     // распределение Пуассона с мат. ожиданием 2
```

1.3 Варианты заданий

1.3.1. Ознакомиться с основными командами и функциями Scilab, а также правилами построения сценариев [3].
1.3.2. Выбрать вариант задания, взяв остаток от деления номера студента в списке группы на 15. Выполнить задания 1.3.3-1.3.7.
1.3.3. Определить функцию $f(x)$ и вычислить N её значений на заданном отрезке. Вывести значения аргумента и значения функции. Построить с необходимыми надписями и координатной сеткой график функции $f(x)$, заданной ниже (номер функции соответствует варианту).

1. Функция $f(x) = \frac{\sin x \cos x}{x+\cos x}$, N=15, на отрезке [0, 2π].

2. Функция $f(x) = \ln(2x-1)\sqrt{e^{-x}+4e^{x}}$, N=10, на отрезке [0.7, 4].

3. Функция $f(x) = (\sin x - 1)\sqrt{e^{-\cos x} + tg\ x\ e^{x}}$, N=20, на отрезке [0.05, 1].

4. Функция $f(x) = x \sin x + \frac{e^{-x}-e^{x}}{e^{-x}+e^{x}}$, N=30, на отрезке [0, 1].

5. Функция $f(x) = x\,(\sin x + \cos x)\frac{x-ctg\ 2x}{1+sin^2\ x}$, N=10, на отрезке [0, π/2].

6. Функция $f(x) = \frac{x}{1+\sqrt[3]{x+1}}$., N=30, на отрезке [10,100].

7. Функция $f(x) = \frac{sh\ x+ch\ x}{th\ x+5}$, N=20, на отрезке [-3,3].

8. Функция $f(x) = x^3\,|\sin x + \cos x\,| + x - tg\ 2x$, N=10, на отрезке [0, π/2].

9. Функция $f(x) = \log_2(\sin x + 1)\sqrt{e^{-\cos x}}$, N=30, на отрезке [0, π/2].

10. Функция $f(x) = \frac{\sqrt[3]{x^2+1}}{\sqrt{|x|+0.5}}\frac{x}{1+sin^2\ x}$, N=10, на отрезке [-π/2, π/2].

11. Функция $f(x) = \frac{sin^2(2x+\frac{\pi}{2})}{\sqrt{\frac{|x|+1}{4}}} + \frac{xln\ x}{1+x^2}$, N=20, на отрезке [-π/2, π/2].

12. Функция $f(x) = \frac{\sqrt{(3x+1)^3}}{x+2} + \sin^2(\frac{2x-1}{x+1})$, N=15, на отрезке [0, 1].

13. Функция $f(x) = \sqrt{\log_2(\sin x + 1)} + \sqrt{1+x^{3.5}}$, N=30, на отрезке [0, π/2].

14. Функция $f(x) = 4cos^2\left(\frac{x+\frac{\pi}{3}}{x+2}\right) - \ln(\frac{e^{-x}-e^{x}}{e^{-x}+e^{x}})$, N=20, на отрезке [0, 1].

15. Функция $f(x) = \ln\left(\sqrt{\frac{2x-1}{x+1}}\right) + \sin(\sqrt{2+x})$, N=10, на отрезке [0.7, 4].

1.3.4. Если возможно, то вычислить матрицу обратную D:

1. $D = 2(A^2 + B)(2B - A)$, где

$$A = \begin{pmatrix} 2 & 3 & -1 \\ 4 & 5 & 2 \\ -1 & 0 & 7 \end{pmatrix}, \qquad B = \begin{pmatrix} -1 & 0 & 5 \\ 0 & 1 & 3 \\ 2 & -2 & 4 \end{pmatrix}$$

2. $D = 3A - (A + 2B)B^2$, где

$$A = \begin{pmatrix} 4 & 5 & -2 \\ 3 & -1 & 0 \\ 4 & 2 & 7 \end{pmatrix}, \qquad B = \begin{pmatrix} 2 & 1 & -1 \\ 0 & 1 & 3 \\ 5 & 7 & 3 \end{pmatrix}$$

3. $D = 3A^2 - (A + 2B)B$, где

$$A = \begin{pmatrix} 4 & 5 & -2 \\ 3 & -1 & 0 \\ 4 & 2 & 7 \end{pmatrix}, \qquad B = \begin{pmatrix} 2 & 1 & -1 \\ 0 & 1 & 3 \\ 5 & 7 & 3 \end{pmatrix}$$

4. $D = (A - B^2 2)(2A + B^3)$, где

$$A = \begin{pmatrix} 5 & 2 & 0 \\ 10 & 4 & 1 \\ 7 & 3 & 2 \end{pmatrix}, \qquad B = \begin{pmatrix} 3 & 6 & -1 \\ -1 & -2 & 0 \\ 2 & 1 & 3 \end{pmatrix}$$

5. $D = 2(A - B)(A^2 + B)$, где

$$A = \begin{pmatrix} 5 & 1 & 7 \\ -10 & -2 & 1 \\ 0 & 1 & 2 \end{pmatrix}, \qquad B = \begin{pmatrix} 2 & 4 & 1 \\ 3 & 1 & 0 \\ 7 & 2 & 1 \end{pmatrix}$$

6. $D = (A - B)^2 A + 2B$, где

$$A = \begin{pmatrix} 5 & -1 & 3 \\ 0 & 2 & -1 \\ -2 & -1 & 0 \end{pmatrix}, \qquad B = \begin{pmatrix} 3 & 7 & -2 \\ 1 & 1 & -2 \\ 0 & 1 & 3 \end{pmatrix}$$

7. $D = (A^2 - B^2)(A + B^2)$, где

$$A = \begin{pmatrix} 7 & 2 & 0 \\ -7 & -2 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} 0 & 2 & 3 \\ 1 & 0 & -2 \\ 3 & 1 & 1 \end{pmatrix}$$

8. $D = 2(A - B)(A^2 + B)$, где

$$A = \begin{pmatrix} 5 & 1 & 7 \\ -10 & -2 & 1 \\ 0 & 1 & 2 \end{pmatrix}, \qquad B = \begin{pmatrix} 2 & 4 & 1 \\ 3 & 1 & 0 \\ 7 & 2 & 1 \end{pmatrix}$$

9. $D = 2A - (A^2 + B)B$, где

$$A = \begin{pmatrix} 1 & 4 & 2 \\ 2 & 1 & -2 \\ 0 & 1 & -1 \end{pmatrix}, \qquad B = \begin{pmatrix} 4 & 6 & -2 \\ 4 & 10 & 1 \\ 2 & 4 & -5 \end{pmatrix}$$

10. $D = 2(A - 0,5B) + A^3B$, где

$$A = \begin{pmatrix} 5 & 3 & -1 \\ 2 & 0 & 4 \\ 3 & 5 & -1 \end{pmatrix}, \qquad B = \begin{pmatrix} 1 & 4 & 16 \\ -3 & -2 & 0 \\ 5 & 7 & 2 \end{pmatrix}$$

11. $D = (A - B)A^2 + 3B$, где

$$A = \begin{pmatrix} 3 & 2 & -5 \\ 4 & 2 & 0 \\ 1 & 1 & 2 \end{pmatrix}, \qquad B = \begin{pmatrix} -1 & 2 & 4 \\ 0 & 3 & 2 \\ -1 & -3 & 4 \end{pmatrix}$$

12. $D = 3(A^2 + B^2) - 2AB$, где

$$A = \begin{pmatrix} 4 & 2 & 1 \\ 3 & -2 & 0 \\ 0 & -1 & 2 \end{pmatrix}, \qquad B = \begin{pmatrix} 2 & 0 & 2 \\ 5 & -7 & -2 \\ 1 & 0 & -1 \end{pmatrix}$$

13. $D = 2A^3 + 3B(AB - 2A)$, где

$$A = \begin{pmatrix} 1 & -1 & 0 \\ 2 & 0 & -1 \\ 1 & 1 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} 5 & 3 & 1 \\ -1 & 2 & 0 \\ -3 & 0 & 0 \end{pmatrix}$$

14. $D = A(A^2 - B) - 2(B + A)B$, где

$$A = \begin{pmatrix} 2 & 3 & 1 \\ -1 & 2 & 4 \\ 5 & 3 & 0 \end{pmatrix}, \qquad B = \begin{pmatrix} 2 & 7 & 13 \\ -1 & 0 & 5 \\ 5 & 13 & 21 \end{pmatrix}$$

15. $D = (2A - B)(3A + B) - 2A^2B$, где

$$A = \begin{pmatrix} 1 & 0 & 3 \\ -2 & 0 & 1 \\ -1 & 3 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} 7 & 5 & 2 \\ 0 & 1 & 2 \\ -3 & -1 & -1 \end{pmatrix}$$

1.3.5. Вычислить 3 значения функции на заданном отрезке. Вывести значения аргумента и значения функции на экран. Построить 3-D график функции (номер функции соответствует варианту):

1. Функция $f(x,y)=\frac{\sin x\cos y}{x+\cos x}$, на отрезке x∈ [0, π],y∈[0, π]
2. Функция $f(x,y)=x\sin xy+\frac{e^{-x}-e^{y}}{e^{-y}+e^{x}}$, на отрезке x∈ [0, π],y∈[0, 1]
3. Функция $f(x,y)=\left(\frac{e^{-x}-e^{y}}{2}\right)^2+\left(\frac{e^{-y}+e^{x}}{2}\right)^2$, на отрезке x,y∈[-3, 3]
4. Функция $f(x,y)=4xy-\cos^2(\frac{e^{-y}-e^{x}}{e^{-x}+e^{y}})$, на отрезке x,y∈[0, 1]
5. Функция $f(x,y)=e^{\frac{y+5}{x^2}}-\sqrt{x+y}$, на отрезке x,y∈[0, 3]
6. Функция $f(x,y)=\mathrm{tg}\left(\sqrt{\frac{2x-y}{x+y+1}}\right)+\sqrt{y+x}$, на отрезке x,y∈ [0.7, 4]
7. Функция $f(x,y)=\frac{x+2y}{1+\sqrt[3]{xy+1}}$., на отрезке x,y∈[10,100]
8. Функция $f(x,y)=\frac{sin^2(2xy+\frac{\pi}{2})}{\sqrt{\frac{|xy|+1}{4}}}$, на отрезке x,y∈[-π/2, π/2]
9. Функция $f(x,y)=\log_2(\sin x+\sin y+1)\sqrt{e^{-\cos x}}$, на отрезке x,y∈[0, π/2]
10. Функция $f(x,y)=|\sin x+\cos y|+x+y$, на отрезке x,y∈[0, π/2]
11. Функция $f(x,y)=\frac{sh\,x+ch\,y}{th\,xy+5}$, на отрезке x,y∈[-3,3]
12. Функция $f(x,y)=4cos^2\left(\frac{x+y+\frac{\pi}{3}}{xy+2}\right)$, на отрезке x,y∈[0, 1]
13. Функция $f(x,y)=x\,(\sin y+\cos x)\frac{y}{1+sin^2\,y}$, на отрезке x,y∈[0, π/2]
14. Функция $f(x,y)=5x^2-2y^2+xy$, на отрезке x,y∈[0, 3]
15. Функция $f(x,y)=(\sin x^2+\cos y^2)^y$, на отрезке x,y∈[-1, 1]

1.3.6. Сгенерировать матрицу из (*N*+10)x(100**N*) одинаково распределенных случайных чисел, где *N* – номер студента в списке группы. Построить гистограмму распределения этих чисел. Вычислить среднее, дисперсию и стандартное отклонение, медиану. Варианты распределений случайных чисел:

1. Равновероятное распределение на интервале (0,1).
2. Нормальное распределение с мат. ожиданием 1 и дисперсией 2.
3. Равновероятное распределение на интервале (-5,5).
4. Нормальное распределение с мат.ожиданием 5 и дисперсией 0.25.
5. Экспоненциальное распределение с параметром лямбда=5.
6. Экспоненциальное распределение с параметром лямбда=10.

7. Распределение Пирсона хи-квадрат с 5-ю степенями свободы.
8. Распределение Пирсона хи-квадрат с 20-ю степенями свободы.
9. Распределение Пирсона хи-квадрат с 1, 10, 30-ю степенями свободы.
10. Распределение Пирсона хи-квадрат с 1, 5, 20-ю степенями свободы.
11. Распределение Пуассона с мат. ожижанием 2.
12. Распределение Пуассона с мат. ожижанием 10.
13. Нормальное распределение с мат. ожиданием -5 и дисперсией 5.
14. Равновероятное распределение на интервале (-3,10).
15. Экспоненциальное распределение с параметром лямбда=1.

1.3.7. Напишите на языке пакета Scilab программу, генерирующую синусоидальные колебания с частотой **Frec**, продискретизированные с шагом **1/Fsample, 1/(2*Fsample), 1/(0.5*Fsample)** (значения **Frec** и **Fsample** определяются вариантом из таблицы 1.1), сохраняющую полученные результаты, соответственно, в векторах **x1, x2** и **x3 (**время реализации сигнала равно двум периодам колебания**)** и отображающую в одном графическом окне график **x1(t),** а другом — **x2(t)** и **x3(t),** а также сохраняющую значения **x1, x2** и **x3** во внешнем файле.

Таблица 1.1 — Варианты заданий

№ варианта	M	N	Frec	Fsample
1	10	5	15	120
2	6	12	12	96
3	5	17	7	60
4	10	11	30	240
5	15	4	37	300
6	7	11	22	160
7	9	5	36	300
8	10	8	33	260
9	5	7	10	80
10	8	12	12	92
11	7	14	17	132
12	16	4	20	160
13	8	11	12	100
14	5	7	9	68
15	12	8	30	240

1.4 Методические указания по выполнению работы

1.4.1. При выполнении задания 1.3.3 рекомендуется воспользоваться функцией **plot2d.** Синтаксис функции (параметры в квадратных скобках не обязательны):

plot2d([logflag,][x,],y[,style[,strf[,leg[,rect[,nax]]]]])
plot2d([logflag,][x,],y,<opt_args>)

Здесь **x** – действительная матрица или вектор; если **x** отсутствует, то предполагается, что **x** – вектор и его элементы изменяются от 1:n, где n длина y;

y – действительная матрица или вектор;

<opt_args> - последовательность ключей **key1=value1, key2=value2,...,** где **key1, key2,...** могут иметь значения:

logflag - устанавливает масштаб вдоль осей (линейный или логарифмический), возможные значения: **"nn", "nl", "ln"** и **"ll";**

style - устанавливает стиль для каждой кривой, целочисленное положительное или отрицательное значение;

strf - управляет отображением заголовка, задается в виде последовательности из трех цифр **"xyz"** (по умолчанию **strf = "081"**).

Значения этих цифр и остальных ключей можно посмотреть в справке к функции. Ниже приведен пример простого сценария для построения графиков 3-х синусоидальных функций с использованием **plot2d**:

```
clf;
x=[0:0.1:2*%pi]';
plot2d(x,[sin(x) sin(2*x) sin(3*x)])
```

1.4.2. В ходе выполнения задания 1.3.4. используйте встроенные функции для работы с матрицами.

1.4.3. Для построения 3-D графиков (задание 1.3.5) рекомендуется изучить возможности пункта меню «Инструменты» графического окна, в частности, вращения изображения и увеличения области, а также возможностей контекстного меню, которое появляется при нажатии правой кнопки мышки. Используйте эти инструменты для более детального анализа свойств функции и редактирования изображения и надписей.

1.4.4. Для построения гистограмм в соответствии с заданием 1.3.6 воспользуйтесь расширенными возможностями функции histplot, приведенными в примере:

```
d=rand(1,10000,'normal');
clf(1); figure(1); histplot(20,d)
xgrid
clf(2); figure(2); histplot(20,d,normalization=%f)
clf(3); figure(3); histplot(20,d,leg='rand(1,10000,''normal'')',style=5)
clf(4); figure(4); histplot(20,d,leg='rand(1,10000,''normal'')',style=16,
rect=[-3,0,3,0.5]);
```

1.4.5. При выполнении задания 1.3.7 обратите внимание на то, что Scilab не является символьным пакетом, и поэтому не в состоянии генерировать анало-

говые (непрерывные) сигналы (например, команда x=sin(t) не приведет к генерации гармонического сигнала). Вместо этого аналоговые сигналы могут быть представлены отдельными отсчетами:

```
fSample=1000;
frec=400;
t=[0:1/fSample:0.5];   //т.е. t=[0 0.001 0.002 0.003 ... 0.05]
x=sin(2*pi*frec*t);
plot(t,x)
```

Для записи значений в файл откройте текстовый файл с помощью функции mopen в режиме записи и запишите туда значения сигналов, используя функцию mfprintf. Проверьте результаты записи значений в файл, просмотрев его в любом текстовом редакторе.

1.5 Содержание отчета

1.5.1. Цель работы.
1.5.2. Вариант задания, описание используемых формул.
1.5.3. Листинги программ с комментариями.
1.5.4. Результаты вычислений и графики функций.
1.5.5. Выводы по результатам исследований.

1.6 Контрольные вопросы

1.6.1. Дайте общую характеристику системы моделирования Scilab и приведите основные отличия от системы Matlab.
1.6.2. Какие типы данных, используются в Scilab и каким образом задаются константы в программах для Scilab?
1.6.3. Как в Scilab определяются пользовательские функции?
1.6.4. Приведите пример задания вектора-строки, вектора столбца и матрицы.
1.6.5. Поясните, как осуществляются операции конкатенации матриц и как они реализуются инструкциями Scilab?
1.6.6. Как математически записываются комплексные числа и какие операции существуют в Scilab для работы с такими числами?
1.6.7. Какие основные статистические функции имеются в Scilab для обработки матриц?
1.6.8. Каким образом можно сохранить данные в файле или ввести их в Scilab из файла?
1.6.9. Какие основные команды Scilab используются для построения двумерных графиков?
1.6.10. Как сделать, чтобы вывод нового графика не стирал предыдущий график?
1.6.11. Как отобразить на графике координатную сетку, подписать рисунок и названия осей координат?

1.6.12. Как построить в Scilab 3-D поверхность?

1.6.13. Напишите сценарий генерирования случайной последовательности с нормальным законом распределения и построения гистограммы распределения.

1.6.14. Как можно разделить графическое окно на отдельные подокна?

1.6.15. Как можно разделить переменные на равномерные интервалы? Приведите пример.

1.6.16. Напишите сценарий генерирования смеси гармонического сигнала с шумом и графического отображения этого процесса.

1.6.17. Напишите сценарий генерации случайных чисел с экспоненциальным распределением и построения гистограммы такого распределения.

Лабораторная работа № 2
СИНТЕЗ ПЕРИОДИЧЕСКИХ СИГНАЛОВ НА ОСНОВЕ РЯДОВ ФУРЬЕ

2.1. Цель работы

Исследование представления периодических сигналов в виде тригонометрического ряда Фурье, синтез сигналов различной формы, анализ погрешностей аппроксимации.

2.2. Краткие теоретические сведения

Математический аппарат, с помощью которого решают задачу синтеза сигналов, основывается на использовании обобщенного ряда Фурье [1]:

$$x(t) \approx \sum_k c_k \phi_k(t), \qquad (2.1)$$

где $x(t)$ — аппроксимируемый сигнал; $\{\varphi_k(t)\}$ — ортогональная система базисных функций; c_k — коэффициенты разложения. Формула (2.1) позволяет аппроксимировать сигнал $x(t)$ в виде суммы базисных функций с весами, равными коэффициентам c_k.

Система базисных функций $\{\varphi_k(t)\}$ называется ортогональной на отрезке $[a, b]$, если скалярное произведение любых 2-х функций (φ_k, φ_m) удовлетворяет условию:

$$(\varphi_k, \varphi_m) = \int_a^b \varphi_k(t)\varphi_m(t)dt = \begin{cases} 0, & k \neq m \\ \|\varphi_k\|^2, & k = m \end{cases},$$

где квадрат нормы функции $\varphi_k(t)$: $\|\varphi_k\|^2 = \int_a^b \varphi_k{}^2(t)dt$.

Коэффициенты $c_к$ обобщенного ряда Фурье могут быть найдены из выражения:

$$c_k = \frac{\int_a^b x(t)\varphi_k(t)dt}{\int_a^b \varphi_k{}^2(t)dt} = \frac{(x, \varphi_k)}{\|\varphi_k\|^2} .$$

На практике приходится ограничивать ряд Фурье конечным числом членов N. Это приводит к ошибке аппроксимации:

$$\varepsilon(t) = x(t) - \sum_{k=0}^{k} c_k \varphi_k(t) .$$

Обычно рассматривают норму ошибки:

$$\|\varepsilon\| = \left\{ \int_a^b \varepsilon^2(t)dt \right\}^{1/2} .$$

Аппроксимация сигнала $x(t)$ рядом Фурье при фиксированном числе слагаемых обеспечивает наилучшее приближение в смысле минимума нормы ошибки.

Если ортогональная система является полной, то увеличением количества членов в сумме (2.1) можно сделать норму ошибки сколь угодно малой. Относительную среднеквадратическую погрешность аппроксимации можно определить из выражения:

$$\delta = \frac{\|\varepsilon\|}{\|x(t)\|}, \quad \text{где} \quad \|x(t)\| = \left\{ \int_a^b x^2(t)dt \right\}^{1/2} .$$

Существуют различные ортогональные системы базисных функций. Так, в качестве базисных функций можно использовать тригонометрические функции кратных аргументов, полиномы Чебышева, Лежандра, Эрмита, функции Бесселя, Лагерра, Хаара, Уолша и др [1].

Если в качестве базисных функций использовать тригонометрические функции кратных аргументов, то периодическую функцию $x(t)$ с периодом T можно представить в виде ряда Фурье [1, 5]:

$$x(t) = a_0 / 2 + \sum_{k=1}^{\infty} (a_k \cos k\omega_1 t + b_k \sin k\omega_1 t) . \qquad (2.2)$$

Коэффициенты этого ряда определяются по формулам:

$$a_0 = \frac{2}{T} \int_{-T/2}^{T/2} x(t)dt \ , \ a_k = \frac{2}{T} \int_{-T/2}^{T/2} x(t)\cos k\omega_1 t dt \ , \ b_k = \frac{2}{T} \int_{-T/2}^{T/2} x(t)\sin k\omega_1 t dt$$

Амплитуда и фаза k-й гармоники разложения (2.2) равна: 2

$$A_k = \sqrt{a_k^{\,2} + b_k} \ , \varphi_k = arctg(b_k / a_k) \ .$$

Совокупности коэффициентов $A_к$ и $\varphi_к$ называют амплитудным и фазовым частотным спектром периодического сигнала $x(t)$.

Ряд (2.1) приводится к форме (2. 2), если положить

$$c_k = A_k e^{j\varphi_k}$$

и в качестве базисных использовать комплексные экспоненциальные функции.

2.3 Варианты заданий

Варианты заданий представлены в таблице 2.1

Таблица 2.1 — Варианты заданий

№ варианта	$T_и$	T	Вид сигнала
1	1	4	рис. 2.1
2	1	4	рис. 2.2

3	1	4	рис. 2.3
4	1	4	рис. 2.4
5	1	2	рис. 2.1
6	1	2	рис. 2.2
7	1	2	рис. 2.3
8	1	2	рис. 2.4
9	1	8	рис. 2.1
10	1	8	рис. 2.2
11	1	8	рис. 2.3
12	1	8	рис. 2.4
13	1	1	рис. 2.1
14	1	1	рис. 2.2
15	1	3	рис. 2.3
16	1	3	рис. 2.4
17	1	3	рис. 2.1
18	1	3	рис. 2.2
19	1	5	рис. 2.3
20	1	5	рис. 2.4
21	1	5	рис. 2.1
22	1	5	рис. 2.2
23	1	6	рис. 2.3
24	1	6	рис. 2.4
25	1	6	рис. 2.1
26	1	6	рис. 2.2
27	1	6	рис. 2.3
28	1	7	рис. 2.4
29	1	7	рис. 2.1
30	1	7	рис. 2.2

2.4. Порядок выполнения лабораторной работы

2.4.1. Рассчитать по формулам (2.2) значения первых десяти коэффициентов разложения в ряд Фурье по системе тригонометрических функций для следующих сигналов, изображенных на рис.2.1 —2.4.

2.4.2. Написать программу на языке пакета Scilab, позволяющую:

- отображать рассчитанный амплитудный спектр заданного сигнала;
- определять относительную среднеквадратическую погрешность аппроксимации;
- синтезировать заданный сигнал по рассчитанным значениям коэффициентов и отображать как заданный исходный сигнал, так и синтезированный.

2.5. Содержание отчета

Отчет должен содержать следующие разделы:

- цель работы;
- постановка задачи (вариант задания);
- расчет коэффициентов разложения заданного сигнала в ряд Фурье;
- амплитудный спектр заданного сигнала;
- расчет погрешности аппроксимации;
- графики исходного и синтезированного сигнала;
- текст программы с комментариями;
- выводы.

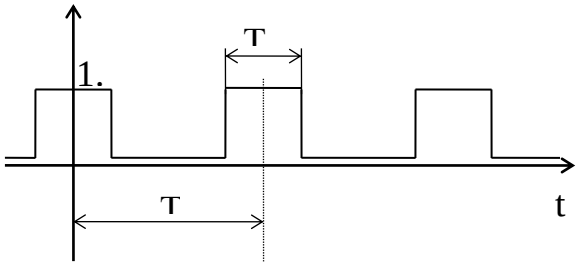

Рисунок 2.1 —Прямоугольные импульсы

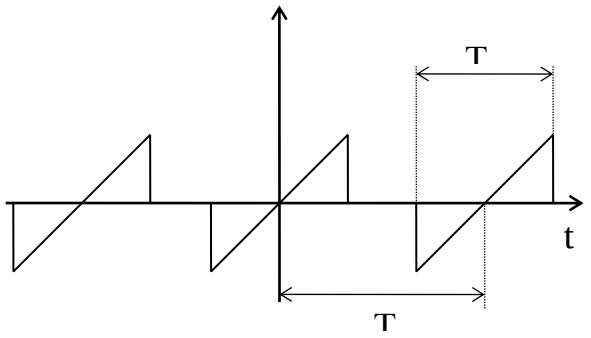

Рисунок 2.2 — Пилообразные импульсы

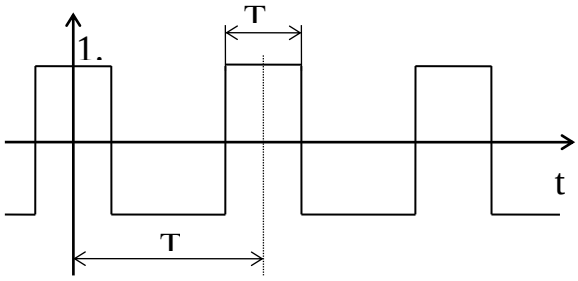

Рисунок 2.3 — Разнополярные прямоугольные импульсы

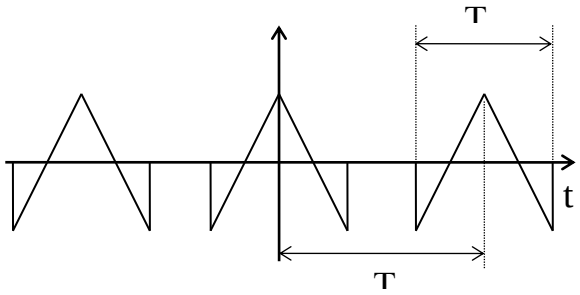

Рисунок 2.4 — Треугольные импульсы

2.6. Контрольные вопросы

2.6.1. Какая система функций называется системой базисных ортогональных функций?
2.6.2. Как определяется норма сигнала, и какой физический смысл имеет это понятие?
2.6.3. Запишите формулу обобщенного ряда Фурье?
2.6.4. Как определяются коэффициенты обобщенного ряда Фурье?
2.6.5. Как определяется погрешность аппроксимации сигналов при конечном числе ортогональных составляющих ряда Фурье?
2.6.6. Каким свойством обладают спектры периодических сигналов?
2.6.7. Как влияет изменение длительности импульса и периода повторения на спектр периодической последовательности прямоугольных видеоимпульсов?
2.6.6. Как отразится на спектре периодического сигнала изменение положения начала отсчета времени?

2.6.7. Как показать, что система функций $\{\cos k\Omega_1 t, \sin k\Omega_1 t\}$ является полной и ортогональной?
2.6.8. Как вычисляются коэффициенты ряда Фурье при разложении периодического сигнала по системе тригонометрических функций (по системе мультипликативно-ортогональных функций, по системе полиномиальных функций Чебышева, по системе функций Уолша)?

Лабораторная работа № 3
ГАРМОНИЧЕСКИЙ АНАЛИЗ СИГНАЛОВ НА ОСНОВЕ ДИСКРЕТНОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ

3.1 Цель работы

Исследование свойств дискретного преобразования Фурье (ДПФ) при гармоническом анализе детерминированных сигналов, исследование «размывания» спектра и «наложения» частот.

3.2 Краткие теоретические сведения

Рассмотрим непрерывный сигнал *x(t)* на конечном интервале времени $(0,T)$, который представлен своими отсчетами $x[0]$, $x[1]$, ..., $x[N-1]$, взятыми в моменты времени $t=nT$, где T— шаг дискретизации. Требуется по отсчетам данных выявить спектральный состав рассматриваемого сигнала.

Для решения рассматриваемой задачи может быть применено дискретное преобразование Фурье (ДПФ). Пара преобразований ДПФ задается выражениями [1, 5]:

$$X[k]=\frac{1}{N}\sum_{n=0}^{N-1}x[n]\exp(-j2\pi kn/N) \qquad (3.1)$$

и

$$x[n]=\sum_{k=0}^{N-1}X[k]\exp(j2\pi kn/N) \quad , \qquad (3.2)$$

где $x[n]$ — исходная N-точечная последовательность данных; $X[k]$ — результирующая N-точечная последовательность спектральных отсчетов.

Отметим следующие свойства ДПФ [1]:

- число спектральных отсчетов $X[k]$, вычисляемых по формуле (3.1), равно числу N отсчетов исходного временного ряда;
- спектральный отсчет $X[0]$ является средним значением всех отсчетов $x[n]$*:*

$$X[0]=\frac{1}{N}\sum_{n=0}^{N-1}x[n] \quad ;$$

- если отсчеты $x[n]$ — вещественные числа, то спектральные отсчеты $X[k]$, номера которых располагаются симметрично относительно отсчета с номером $N/2$, образуют комплексно-сопряженные пары. Следовательно, для анализа амплитудного спектра можно использовать только первую половину отсчетов $X[k]$.

Одной из особенностей спектра дискретных сигналов является возможность появления ложных спектральных составляющих. На рис. 3.1 показано соотношение спектров исходного непрерывного сигнала и его дискретного аналога. Спектр дискретного сигнала является периодическим с периодом, равным частоте дискретизации $f_д$. Если частота дискретизации меньше верхней граничной

частоты спектра сигнала, то в результате наложения повторяющихся спектров образуются ложные спектральные составляющие [1, 5].

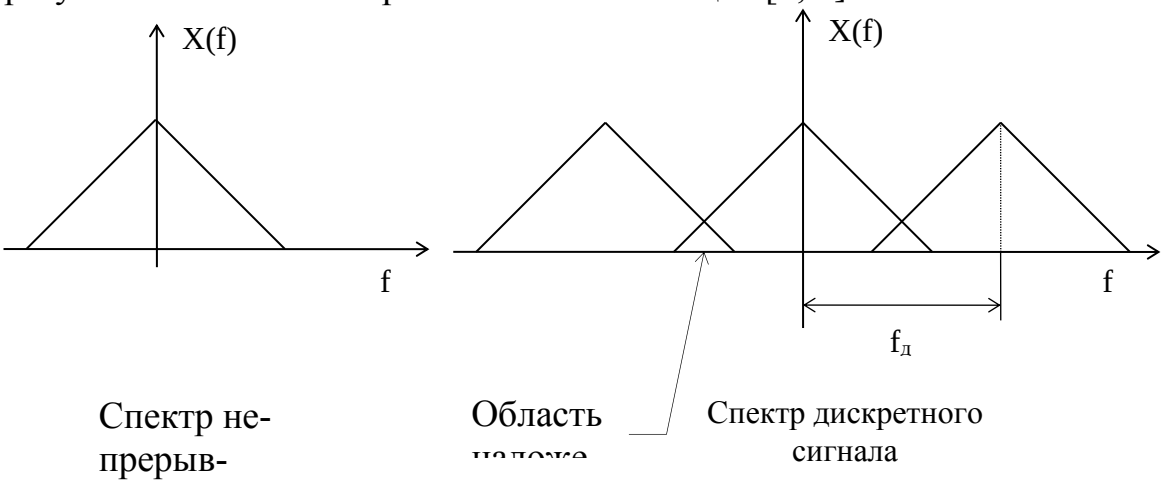

Рисунок 3.1 — Спектры непрерывного и дискретного сигналов

Иная особенность ДПФ состоит в «размывании» спектра [1]. Ограничение временного ряда $x[n]$ конечным числом значений равносильно умножению его на временное окно прямоугольной формы.

$$w[n] = \begin{cases} 1, & n \le N-1 \\ 0, & n > 0 \end{cases} .$$

Поэтому последовательность наблюдаемых данных $x[n]$ из N отсчетов можно описать как произведение бесконечной последовательности $x_0[n]$ на $w[n]$:

$$x[n] = x_0[n]\, w[n] .$$

ДПФ последовательности $x[n]$, выраженной через ДПФ последовательности $x_0[n]$ и временного окна $w[n]$, равно свертке этих преобразований:

$$X(\omega) = X_0(\omega) * W(\omega)$$

где

$$W(\omega) = T_0 e^{-j\omega T_0 \frac{(N-1)}{2}} \frac{\sin(N\omega T_0/2)}{\sin(\omega T_0/2)}$$

Функция $W(\omega)$, называемая ядром Дирихле, представляет собой ДПФ прямоугольного окна $w[n]$. Преобразование Фурье наблюдаемой конечной последовательности $x[n]$ является искаженной версией преобразования Фурье бесконечной последовательности $x_0[n]$. Влияние прямоугольного окна на спектр синусоиды с частотой f_0 =35Гц иллюстрирует рис.3.2.

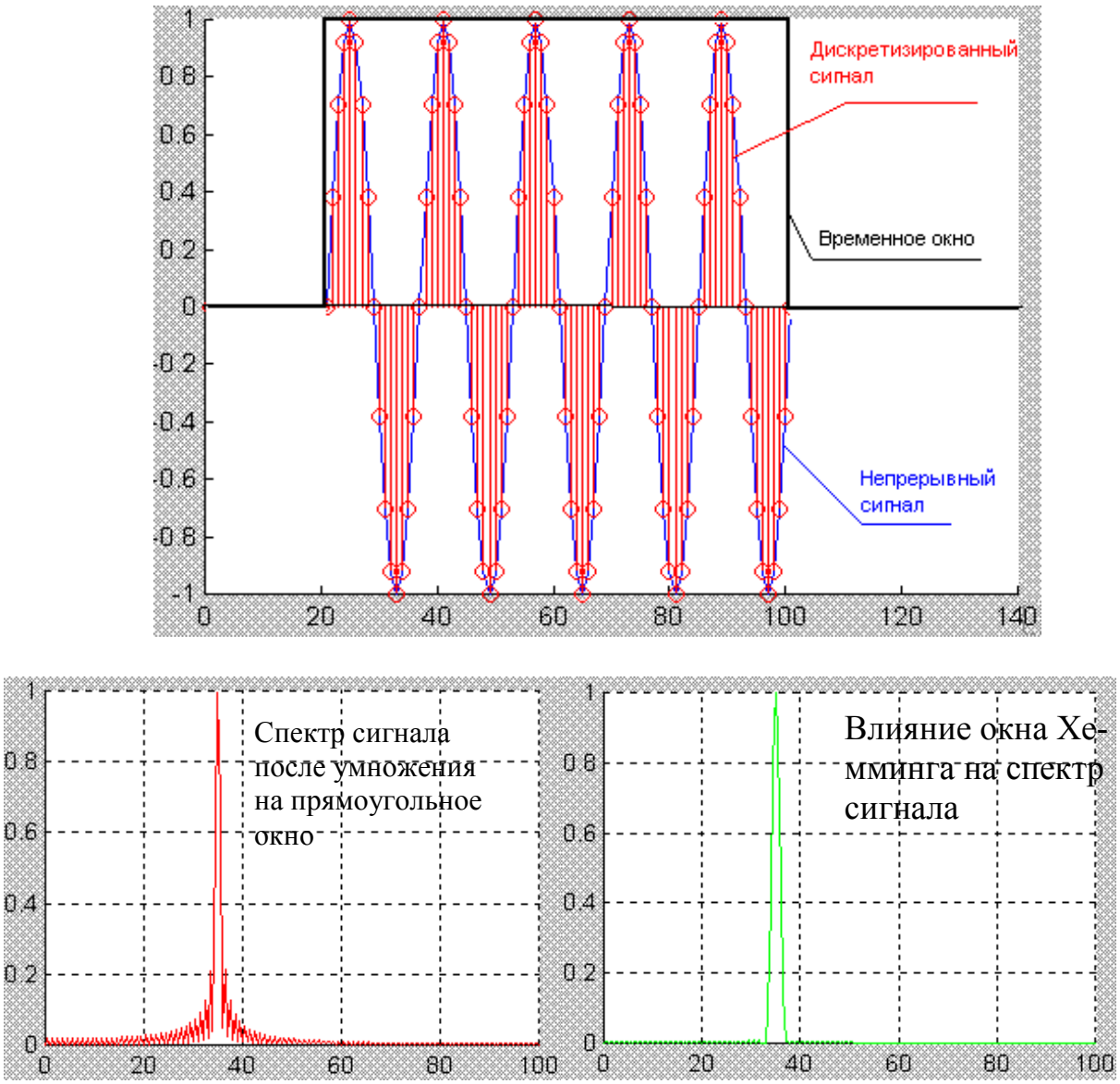

Рисунок 3.2 — Влияние окон на спектр дискретной синусоиды

Из рисунка 3.2 видно, что предполагаемый в точке f_0 теоретический спектральный пик ДПФ бесконечной последовательности $x_0[n]$ расширяется за счет свертки со спектром прямоугольного окна, т.е. происходит «утечка» энергии в боковые лепестки. Боковые лепестки могут искажать амплитуды соседних спектральных пиков сложных сигналов.

Для устранения этого влияния необходимо локализовать распределение энергии сигнала у центральной частоты f_0. Обычно это достигается умножением временного ряда на соответствующим образом подобранное временное окно. Вместе с тем необходимо учитывать, что при «подавлении» энергии боковых лепестков возможна потеря отдельных спектральных составляющих из-за расширения основного лепестка.

Ниже приведены определения некоторых наиболее часто используемых функций *w***[***n***]** окон [1], *-N/2<=n<=N/2-1***:**

- треугольное:

$$w[n] = 1 - \frac{|n|}{N} \quad ;$$

- косинус-квадрат (окно Ханна):

$$w[n] = \cos^2(\frac{\pi n}{N}) = \frac{1}{2}\left(1 - \cos\frac{2\pi n}{N}\right) \ ;$$

- приподнятый-косинус (окно Хэмминга):

$$w[n] = \alpha + (1-\alpha)\cos\frac{2\pi n}{N} \ , \quad \text{обычно } \alpha=0.54;$$

- окно Блэкмана:

$$w[n] = 0.42 - 0.50\cos\left(\frac{2\pi n}{N}\right) + 0.08\cos\left(\frac{4\pi n}{N}\right);$$

- окно Кайзера:

$$w[n] = \frac{I_0\left[\pi\alpha\sqrt{1-\left(\frac{n}{N/2}\right)^2}\right]}{I_0(\pi\alpha)};$$

где $I_0(x)$ — модифицированная функция Бесселя первого рода нулевого порядка, α — константа (обычно α=3,0). При увеличении параметра α уровень боковых лепестков оконной функции уменьшается.

- окно Чебышева определяется в частотной области (задается ширина основного лепестка и уровень боковых лепестков) и вычисляется через обратное дискретное преобразование Фурье.

Вычисление ДПФ требует выполнения N^2 операций сложения с умножением. С целью сокращения количества операций ДПФ вычисляют с помощью алгоритма быстрого преобразования Фурье (БПФ). Вычисление БПФ требует всего $2N \log_2 N$ операций. Например, при $N=2^{10}$ объем вычислений БПФ сокращается примерно в 50 раз по сравнению с ДПФ. Наиболее широкое распространение алгоритм БПФ получил при вычислении дискретных последовательностей, длина которых кратна двум. Вместе с тем разработаны и другие методы, основанные на преобразованиях числовых последовательностей иной длины. Вычислить прямое и обратное ДПФ с помощью алгоритма БПФ в Scilab можно, воспользовавшись функциями **fft** и **ifft.**

3.3 Варианты заданий

Варианты заданий приведены в таблице 3.1.

Таблица 3.1— Варианты заданий

№ варианта	Вид окна	f1	f2	f3	fsample1	fsample2
1	Хэмминг	10	14	30	120	50
2	Ханн	8	12	24	96	40

3	Кайзер	5	7	15	60	25
4	Чебышев	20	28	60	240	100
5	Треугольное	25	35	75	300	140
6	Хэмминг	15	22	45	160	80
7	Ханн	25	35	75	300	140
8	Кайзер	20	28	60	240	100
9	Чебышев	5	7	15	60	25
10	Треугольное	8	12	24	96	40
11	Хэмминг	10	14	30	120	50
12	Ханн	16	24	40	160	80
13	Кайзер	8	11	24	56	40
14	Чебышев	5	7	15	60	25
15	Треугольное	20	28	60	240	100
16	Хэмминг	25	35	75	300	140
17	Ханн	15	22	45	160	80
18	Кайзер	10	14	30	120	50
19	Чебышев	5	7	15	60	25
20	Треугольное	20	28	60	240	100
21	Хэмминг	25	35	75	300	140
22	Ханн	15	22	45	160	80
23	Кайзер	25	35	75	300	140
24	Чебышев	20	28	60	240	100
25	Треугольное	10	14	30	120	50
26	Хэмминг	8	12	24	96	40
27	Ханн	5	7	15	60	25
28	Кайзер	20	28	60	240	100
29	Чебышев	25	35	75	300	140
30	Треугольное	15	22	45	160	80

3.4. Порядок выполнения лабораторной работы

3.4.1. Сформировать, используя пакет Scilab, гармонические сигналы **a, b, c**:

```
secs=5;
t=(0:1/fsample:secs);
a=sin(2*%pi*f1*t);
b=sin(2*%pi*f2*t);
c=sin(2*%pi*f3*t);
```

Здесь **f1, f2, f3** — частоты определяемые вариантом задания; **fsample** — частота дискретизации; **secs** — длительность реализации сигнала. Варианты заданий см. в таблице 3.1.

3.4.2. Построить амплитудные спектры сигналов **a, b** и **c,** используя функцию **fft.** Например:

xa==1/N*abs(fft(a));

Для отображения графиков спектров воспользоваться функцией **plot**. Ось частот должна иметь значения от 0 до **fsample-(fsample/N)**, где **N** — количество отсчетов сигнала.

3.4.3. Сформировать сигнал, являющийся суммой сигналов **a, b, c:**

d=a(1:N)+0.75*b(1:N)+0.5*c(1:N);

и построить его амплитудный спектр:

AmpSpec=1/N*abs(fft(d));

Для того, чтобы наблюдать "размытый" спектр дискретного сигнала необходимо уменьшить шаг дискретизации по частоте. Для этого при формировании сигнала d необходимо заполнить нулями N отсчетов перед началом реализации d и N отсчетов после окончания реализации**:**

```
z=zeros(1,N);
// дополнение сигнала d нулями
dn=[z d z];
```

3.4.4. Сформировать сигнал **wd**, являющийся произведением сигнала d и заданного временного окно **w**. Например:

```
w=window('hn', N);
wd=w .*d;
```

Здесь **window('hn', N)** – функция Scilab, вычисляющая окно типа **'hn'** (Ханна) длины **N.** Типы окон:

- **'re'**: прямоугольное;
- **'tr'**: треугольное;
- **'hm'**: Хэмминга;
- **'hn'**: Ханна;
- **'kr'**: Кайзера, в этом случае должен быть указан дополнительный аргумент wpar;
- **'ch'**: Чебышева, в этом случае должен быть указан дополнительный аргумент wpar в виде вектора из 2-х элементов.

Необязательный параметр wpar для окон Кайзера и Чебышева:

- **'kr': wpar** должен быть строго положительным числом;

- **'ch': wpar** должен быть двухэлементным вектором **[уровень боковых лепестков, ширина основного лепестка]**, где **уровень боковых лепестков >0** и **0< ширина основного лепестка <.5**. Только одно из этих значений должно быть задано, второе должно быть равно -1. Например: **[0.001 -1]** или **[-1, 0.1].**

3.4.5. Построить графики сигнала **wd** и его амплитудного спектра.

3.4.6. Сравнить реализации и спектры сигналов **d** и **wd**. Сформулировать выводы.

3.4.7. Построить графики прямоугольного окна и окна заданного по варианту задания во временной и частотных областях. Описать свойства окна.

Для построения частотной характеристики окна воспользуйтесь следующей функцией Scilab:

[W,fr] = frmag(w, n) ,

где **w** – вектор отсчетов окна, **n** – число точек, в которых вычисляется частотная характеристика окна, **W** – вектор значений АЧХ окна, **fr** – вектор нормированных частот, в которых получены значения АЧХ. Вычисленные значения АЧХ отобразите на графике в логарифмическом масштабе в дБ (**20log10(W)**).

Для построения АЧХ также можно использовать функцию fft(w). В этом случае для повышения разрешения по частоте при построении АЧХ окна необходимо будет дополнить отсчеты окна нулями.

3.4.8. Повторить пункты 3.4.1-3.4.7 для иной частоты дискретизации в соответствии с вариантом задания. Объяснить результаты.

3.5 Содержание отчета

Цель работы, постановка задачи и вариант задания, краткие теоретические сведения, графики реализаций и амплитудных спектров сигналов a, b, c, d, dw для различных частот дискретизации, временные и спектральные графики окон в соответствии с вариантом, текст программы с комментариями, выводы.

3.6 Контрольные вопросы

3.6.1. Как записываются прямое и обратное дискретное преобразования Фурье?
3.6.2. Назовите и объясните свойства ДПФ?
3.6.3. Объясните «механизм» наложения частот?
3.6.4. Объясните причину «размывания» спектральных составляющих при выполнении ДПФ?
3.6.5. Почему весовые окна позволяют уменьшить «утечку» энергии?
3.6.6. Назовите и охарактеризуйте основные параметры весовых окон?
3.6.7. Приведите определения и свойства окон Ханна и Хемминга?
3.6.8. В чем преимущества применения БПФ?
3.6.9. Сформулируйте основную идею алгоритма БПФ?
3.6.10. Выполните вывод формул БПФ.
3.6.11. Объясните операцию БПФ «бабочка».
3.6.12. Нарисуйте граф 8-точечного БПФ.
3.6.13. Объясните принцип бит-инверсного упорядочения при вычислении БПФ.
3.6.14. Как в Scilab можно вычислить прямое и обратное БПФ?

Лабораторная работа № 4
ЛИНЕЙНЫЕ ДИСКРЕТНЫЕ СИСТЕМЫ И ЦИФРОВЫЕ ФИЛЬТРЫ

4.1 Цель работы

Изучение основных понятий линейных дискретных систем и исследование методов реализации цифровых фильтров во временной и частотной областях.

4.2 Краткие теоретические сведения

Выходная последовательность $y(nT)$ линейной дискретной системы может быть вычислена с помощью свертки входной последовательности $x(nT)$ с импульсной характеристикой линейной дискретной системы $h(nT)$ [1]:

$$y(nT) = \sum_{k=-\infty}^{\infty} h(kT)x(nT - kT) \quad . \tag{4.1}$$

или

$$y(nT) = h(nT)*x(nT) \, , \tag{4.2}$$

где T – период дискретизации.

Импульсная характеристика (ИХ) дискретной линейной системы определяется как отклик системы на входное воздействие

$$x(nT) = \begin{cases} 1, \, при & n = 0 \\ 0, \, при & n \neq 0 \end{cases} .$$

В цифровой обработке сигналов рассматриваются устойчивые и физически реализуемые линейные дискретные системы. Система является устойчивой, если

$$\sum_{k=-\infty}^{\infty} | h(nT) | < \infty$$

Система является физически реализуемой, если $h(nT)=0$ при $n<0$.

В общем случае для линейной дискретной системы последовательности $x[n]=x(nT)$ и $y[n]=y(nT)$ связаны между собой разностным уравнением [1, 5]:

$$y[n] = \sum_{k=0}^{N-1} b[k]x[n-k] - \sum_{k=1}^{M-1} a[k]y[n-k] \, , \tag{4.3}$$

где $a[k]$ и $b[k]$ — постоянные коэффициенты, определяющие свойства системы.

Выражение (4.3) представляет алгоритм функционирования цифрового фильтра. В зависимости от вида импульсной характеристики цифровые фильтры принято делить на два класса: КИХ-фильтры (фильтры с конечной импульсной характеристикой) и БИХ-фильтры (фильтры с бесконечной импульсной характеристикой). Отметим, что формула (4.3) определяет БИХ-фильтр. Одна из возможных структурных схем реализации БИХ-фильтра изображена рис.4.1. Фильтр, функционирующий на основе (4.3), также называют рекурсивным фильтром.

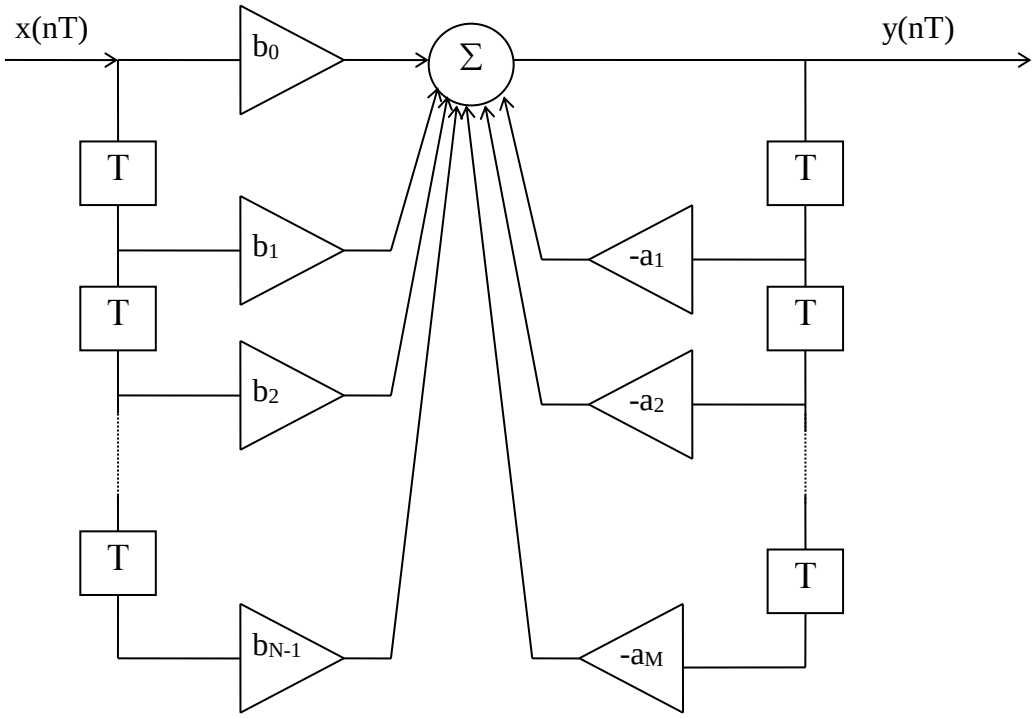

Рисунок 4.1 — Структурная схема БИХ-фильтра

Если все коэффициенты $a[k]$ в уравнении (4.3) равны нулю, то

$$y[n] = \sum_{k=0}^{N} b[k]x[n-k]. \qquad (4.4)$$

Фильтр (4.4) имеет конечную длительность импульсной характеристики, равную NT. В этом случае $b[k]=h[k]$, т. е. коэффициенты КИХ-фильтра представляют отсчеты его импульсной характеристики. Выражение (4.4) определяет нерекурсивный фильтр.

Передаточной функцией $H(z)$ называют отношение z-образов выходного $Y(z)$ и входного $X(z)$ сигналов фильтра при нулевых начальных условиях: $H(z)=Y(z)/X(z)$ [1, 5]. Для рекурсивного и нерекурсивного фильтров передаточные функции соответственно равны:

$$H(z) = \frac{\sum_{k=0}^{N-1} b[k]z^{-k}}{1+\sum_{k=1}^{M-1} a[k]z^{-k}},$$

$$H(z) = \sum_{k=1}^{N-1} b[k]z^{-k}.$$

Коэффициенты фильтров являются коэффициентами соответствующих передаточных функций. Комплексные частотные характеристики представляют

функции, получаемые в результате подстановки $z=exp(j\omega T)$ в передаточные функции. Модуль комплексной частотной характеристики
$A(\omega) = |H(exp(j\omega T))|$ называется амплитудно-частотной характеристикой (АЧХ). Аргумент комплексной частотной характеристики $\varphi(\omega) = arg[H(\exp(j\omega T))]$ называется фазочастотной характеристикой (ФЧХ).

Частотные характеристики ЦФ с вещественными коэффициентами обладают следующими свойствами:

1) все частотные характеристики представляют собой периодические функции частоты с периодом, равным частоте дискретизации $\omega_д=2\pi/T$;

2) АЧХ представляет собой четную функцию частоты, а ФЧХ — нечетную функцию.

Так как частотные характеристики линейных дискретных систем являются периодическими, то требования к ним задают на интервале частот, равном половине периода $[0, \pi/T]$. С целью сопоставимости частотных характеристик различных фильтров выполняют нормировку частоты одним из двух способов. Либо используют нормированную частоту $w = \omega T$, тогда
$w_д = \omega_д T = 2\pi$ и требования к частотным характеристикам задают на интервале $[0, \pi]$. Либо полагают, что нормированная частота $w = \omega T/2\pi$, тогда $w_д=\omega_д T/2\pi=1$ и требования к частотным характеристикам задают в интервале [0, 0.5]. При этом изменяются аргументы в обозначениях частотных характеристик $H(e^{jw})$, $A(w)$, $\varphi(w)$.

Импульсная и частотная характеристики дискретных фильтров связаны парой дискретных преобразования Фурье:

$$h[n] = F^{-1}\{H(e^{j\omega kT})\},$$

$$H(e^{j\omega kT}) = F\{h[n]\}.$$

Исходя из этого, возможна реализация фильтров не только во временной области, в соответствии с (4.1), но и в частотной. При этом свертка (4.1) может быть реализована путем вычисления обратного преобразования Фурье от произведения преобразований Фурье импульсной характеристики фильтра $h(nT)$ и входного сигнала $x(nT)$, рассматриваемых на конечном интервале времени NT.

Пакет Scilab содержит ряд функций для исследования линейных дискретных систем и цифровых фильтров.

y = filter(B, A, x) — фильтрация данных из вектора **x** с помощью фильтра, функционирующего на основе выражения (4.3). Коэффициенты фильтра хранятся в векторах **B**, **A.**

[H,w]=frmag(B,A,N) — вычисление нормированной амплитудно-частотной характеристики фильтра с коэффициентами, которые хранятся в векторах **B**, **A.** Результат **H** — вектор значений АЧХ, **w** — вектор частот, **N** — длина векторов **H** и **w.**

В общем случае для реализации быстрой фильтрации в частотной области, необходимо умножить БПФ от входной последовательности $x[n]$ на БПФ импульсной характеристики фильтра $h[n]$ и вычислить ОБПФ произведения:

y=ifft(fft(x).*fft(h)) .

Если **h** – импульсная характеристика фильтра, заданного векторами коэффициентов **B** и **A,** то вычисленный вектор **y** будет соответствовать результату, возвращаемому вызовом **y = filter(B, A, x)** с точностью до переходных процессов.

4.3 Варианты заданий

Заданы передаточные функции линейных дискретных систем вида:

$$H(z) = \frac{b_0 + b_1 \cdot z^{-1} + b_2 \cdot z^{-2}}{a_0 + a_1 \cdot z^{-1} + a_2 \cdot z^{-2}}$$

В таблице 4.1 приведены коэффициенты передаточной функции для каждого варианта задания.

Таблица 4.1— Варианты заданий

Вариант	a0	a1	a2	b0	b1	b2
1	1	-1.5610	0.6414	0.0201	0.0402	0.0201
2	1	-1.1430	0.4128	0.0675	0.1349	0.0675
3	1	-0.7478	0.2722	0.1311	0.2622	0.1311
4	1	-0.3695	0.1958	0.2066	0.4131	0.2066
5	1	0	0.1716	0.2929	0.5858	0.2929
6	1	-1.1430	0.4128	0.6389	-1.2779	0.6389
7	1	-1.5610	0.6414	0.8006	-1.6012	0.8006
8	1	-0.7478	0.2722	0.5050	-1.0100	0.5050
9	1	-0.3695	0.1958	0.3913	-0.7827	0.3913
10	1	0	0.1716	0.2929	-0.5858	0.2929
11	1	-0.1842	0.1776	0.3404	-0.6809	0.3404
12	1	-0.5570	0.2270	0.4460	-0.8920	0.4460
13	1	-0.9428	0.3333	0.5690	-1.1381	0.5690
14	1	-1.3490	0.5140	0.7157	-1.4315	0.7157
15	1	-1.7786	0.8008	0.8949	-1.7897	0.8949
16	1	-1.7786	0.8008	0.0055	0.0111	0.0055
17	1	-1.3490	0.5140	0.0413	0.0825	0.0413
18	1	-0.9428	0.3333	0.0976	0.1953	0.0976
19	1	-0.5570	0.2270	0.1675	0.1675	0.3350
20	1	-0.1842	0.1776	0.2483	0.4967	0.2483
21	1	-1.5610	0.6414	0.0201	0.0402	0.0201
22	1	-1.1430	0.4128	0.0675	0.1349	0.0675
23	1	-0.7478	0.2722	0.1311	0.2622	0.1311
24	1	-0.3695	0.1958	0.2066	0.4131	0.2066
25	1	0	0.1716	0.2929	0.5858	0.2929
26	1	-1.1430	0.4128	0.6389	-1.2779	0.6389
27	1	-1.5610	0.6414	0.8006	-1.6012	0.8006

28	1	-0.7478	0.2722	0.5050	-1.0100	0.5050
29	1	-0.3695	0.1958	0.3913	-0.7827	0.3913
30	1	0	0.1716	0.2929	-0.5858	0.2929

4.4 Порядок выполнения лабораторной работы

4.4.1. Построить частотную характеристику фильтра с помощью функции **frmag**.

4.4.2. Построить импульсную характеристику фильтра, как реакцию фильтра на единичное входное воздействие.

4.4.3. Сформировать сигнал $x(nT)$ в виде суммы 3-х гармонических косинусоидальных сигналов. Частоты косинусов ω_1, ω_2, ω_3 выбрать таким образом, чтобы фильтр пропускал гармоники ω_1 и ω_2, и подавлял ω_3.

4.4.4. Построить спектр сигнала, сформированного в соответствии с п.4.4.3.

4.4.5. Выполнить фильтрацию сигнала $x(nT)$ во временной области.

4.4.6. Выполнить фильтрацию сигнала $x(nT)$ в частотной области.

4.4.7. Построить графики выходных процессов в соответствии с п.4.4.5. и п.4.4.6 в одной системе координат.

4.4.8. Построить спектр выходного сигнала фильтра.

4.4.9. Сравнить результаты фильтрации, полученные двумя способами.

4.5 Содержание отчета

Цель работы, постановка задачи, краткие теоретические сведения, текст программы с комментариями, частотная характеристика фильтра, импульсная характеристика фильтра, входной процесс $x(nT)$, графики выходных процессов при фильтрации во временной и частотной областях, выводы.

4.6 Контрольные вопросы

4.6.1. Запишите соотношение, устанавливающее взаимосвязь входной и выходной последовательности линейной дискретной системы.

4.6.2. Что понимают под импульсной характеристикой?

4.6.3. Что такое передаточная функция линейной дискретной системы?

4.6.6. Что такое АЧХ и ФЧХ?

4.6.7. Сформулируйте свойства комплексной частотной характеристики?

4.6.8. Сформулируйте понятия устойчивости и физической реализуемости дискретной линейной системы?

4.6.9. Определите на языке Scilab функцию, вычисляющую выходной сигнал рекурсивного и нерекурсивного фильтра.

4.6.10. Объясните возможность реализации фильтрации в частотной области?

4.6.11. каким образом выполняют нормирование частоты при построении ачх дискретных линейных систем?
4.6.12. Как взаимосвязаны импульсная и частотная характеристики?

Лабораторная работа № 5
СИСТЕМЫ С МНОГОЧАСТОТНОЙ ДИСКРЕТИЗАЦИЕЙ

5.1 Цель работы

Исследование свойств линейных дискретных систем с повышением и понижением частоты дискретизации.

5.2 Краткие теоретические сведения

Восходящей дискретной системой (ВДС) называется система, частота дискретизации сигнала на выходе которой, выше частоты дискретизации входного сигнала. *Нисходящей дискретной системой* (НДС) называется система, частота дискретизации на выходе которой, ниже частоты дискретизации входного сигнала [1].

Каждая ВДС содержит элемент, увеличивающий частоту дискретизации — *интерполятор*, находящийся на входе системы, и *дискретный фильтр*, выполняющий последующую обработку сигнала с выходной частотой дискретизации. Каждая НДС состоит из *дискретного фильтра*, выполняющего предварительную обработку входного сигнала с входной частотой дискретизации, и элемента, уменьшающего частоту дискретизации — *дециматора*, находящегося на выходе системы.

Интерполятор, увеличивает частоту дискретизации в L раз (L—целое) и представляет элемент, преобразующий входной дискретный сигнал $x(nT')$ с периодом дискретизации T', в выходной дискретный сигнал $y(nT)$ с периодом дискретизации $T=T'/L$. Уравнение функционирования интерполятора может быть записано в виде:

$$y(nT)=\begin{cases} x\left(\frac{n}{L}\cdot T'\right) & при \quad n=0,L,\,2L... \\ 0 & при \quad других \quad n \end{cases} \qquad (5.1)$$

Таким образом, сигнал $y(nT)$ получается из $x(nT')$ путем ввода между соседними отсчетами $x(nT')$ дополнительных нулевых отсчетов. На рис.5.1. изображены: интерполятор, входная и выходная последовательности отсчетов при L=3. Позиции нулевых отсчеты выходной последовательности обозначены условно.

Спектр дискретного сигнала $x(nT')$ периодичен по оси частот с частотой дискретизации $\omega'_д=2\pi/T'$. Модуль спектра $|X(e^{j\omega T'})|$ сигнала $x(nT')$ при $T'=2T$. изображен на рис.5.2. в виде условного треугольного графика.

Спектр выходного сигнала интерполятора $y(nT)$ будет иметь тот же вид, что и спектр его входного сигнала (рис.5.2). Действительно, поскольку последовательность $y(nT)$ отлична от нуля только при n=0,L,2L,... , то в выражении для спектра сигнала $y(nT)$

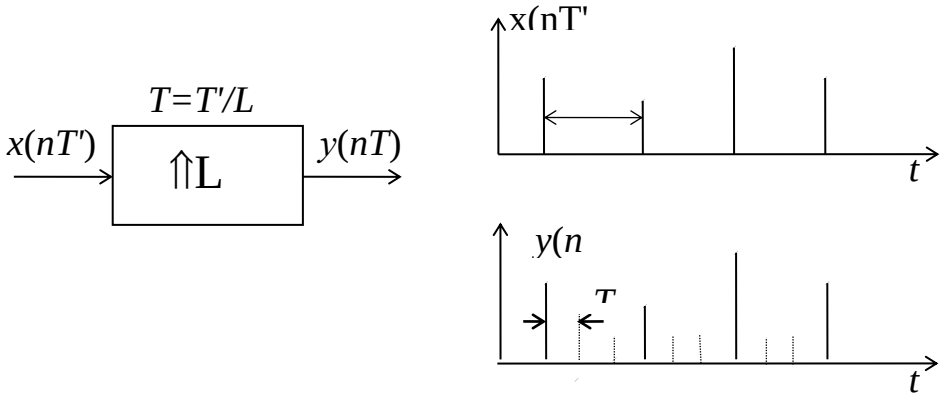

Рисунок 5.1 — Интерполятор, его входная и выходная последовательности

$$Y(e^{j\omega T}) = \sum_{n=0}^{\infty} y(nT)e^{-j\omega nT} \tag{5.2}$$

можно заменить n на nL. Тогда

$$Y(e^{j\omega T}) = \sum_{n=0}^{\infty} y(nLT)e^{-j\omega nLT} = \sum_{n=0}^{\infty} x(nT')e^{-j\omega nT'} = X\left(e^{j\omega T'}\right). \tag{5.3}$$

Таким образом, спектры входного и выходного сигналов интерполятора совпадают, т.е. спектр сигнала $y(nT)$ будет периодически повторяться со "старой" частотой дискретизации $\omega'_д=2\pi/T'$, которая соответствует частоте дискретизации входного сигнала интерполятора.

Если предположить, что некоторый сигнал $y^*(nT)$ получается путем непосредственной дискретизации исходного аналогового сигнала x(t) с шагом дискретизации T, то спектр $Y^*(e^{j\omega T})$ должен повторяться с частотой $\omega_д=2\pi/T$ (рис. 5.2).

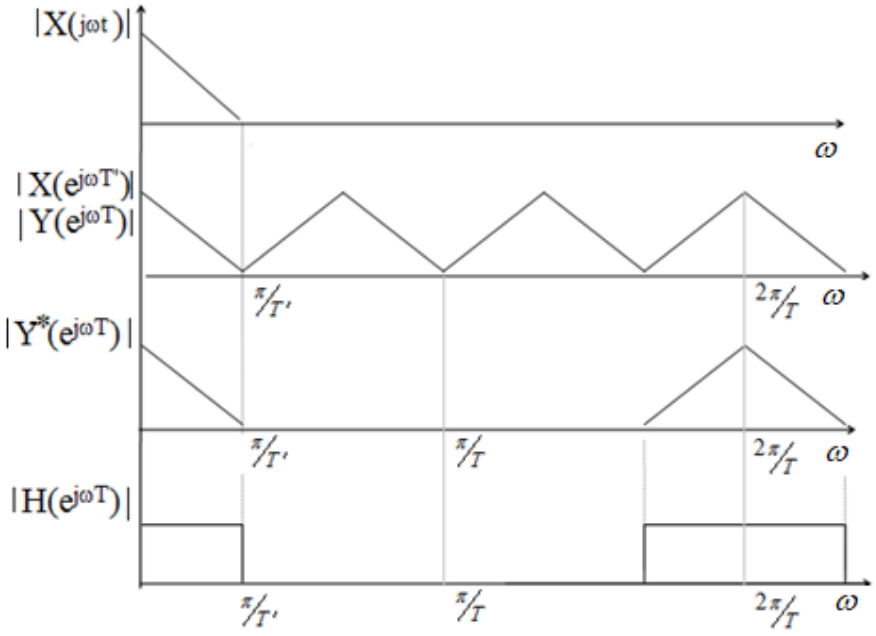

Рисунок 5.2 — Спектры сигналов $x(nT')$ и $y(nT)$

Из сравнения графиков спектров видно, что спектр $|Y(e^{j\omega T})|$ сигнала на выходе интерполятора отличается от требуемого спектра $|Y^*(e^{j\omega T})|$ наличием "лишних" частотных составляющих. Требуемый сигнал $y^*(nT)$ может быть получен из $y(nT)$ путем исключения этих составляющих. Для этого на выходе интерполятора устанавливают ФНЧ. Идеализированная АЧХ такого фильтра представлена на рис. 5.2 (нижний график). АЧХ фильтра на выходе интерполятора должна удовлетворять требованиям [1]:

$$\left|H\left(e^{j\omega T}\right)\right| = \begin{cases} L & при \quad |\omega| < \pi/T, \\ 0 & при \quad \pi/T, < |\omega| < \pi/T \end{cases} . \tag{5.4}$$

Дециматор понижает частоту дискретизации в M раз. В этом случае из входного дискретного сигнала, описываемого решетчатой функцией $x(nT)$ с периодом дискретизации T, берется каждый M-ый отсчет: $y(nT')$=y(nMT)=x(nMT). Спектр выходного сигнала дециматора будет повторяться со сниженной в M раз частотой дискретизации.

Уменьшение частоты дискретизации сигнала $x(nT)$, основной спектр которого занимает полосу частот от 0 до π/T, приводит к наложению спектров (рис. 5.3).

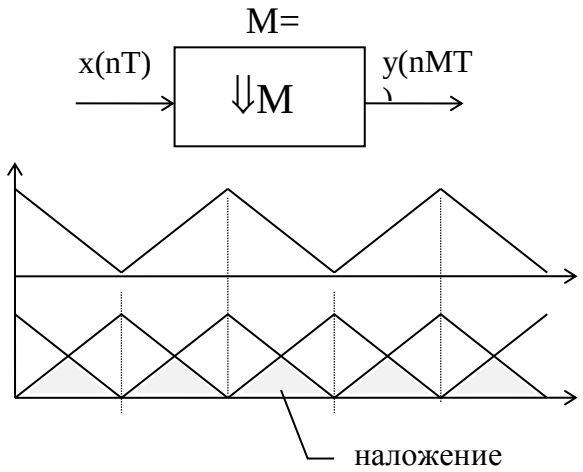

Рисунок 5.3— Децимация и наложение спектров

Для того, чтобы уменьшение частоты дискретизации не приводило к наложению спектров, спектр входного сигнала $x(nT)$ дециматора не должен содержать гармонических составляющих на частотах выше $\pi/(MT)$. Следовательно, на входе дециматора должен быть установлен фильтр нижних частот с АЧХ:

$$\left|H\left(e^{j\omega T}\right)\right| = \begin{cases} 1 & при \quad |\omega| < \pi/MT \\ 0 & при \quad |\omega| > \pi/MT \end{cases} . \tag{5.5}$$

Система, в которой увеличение (уменьшение) частоты дискретизации производится за один прием (однократно), называется соответственно простейшей

восходящей линейной системой (ПВДС) и простейшей нисходящей линейной системой (ПНДС).

ПВДС представлена на рис. 5.4. Входной сигнал $x(nT')$=x(nLT) с интервалом дискретизации $T'=LT$ поступает на интерполятор, который увеличивает частоту дискретизации в L раз. Выходной сигнал $y(nT)$ ПВДС получается в результате обработки сигнала $x^*(nT)$ на выходе интерполятора дискретным ФНЧ с передаточной функцией $H(z)$, АЧХ которого соответствует (5.4).

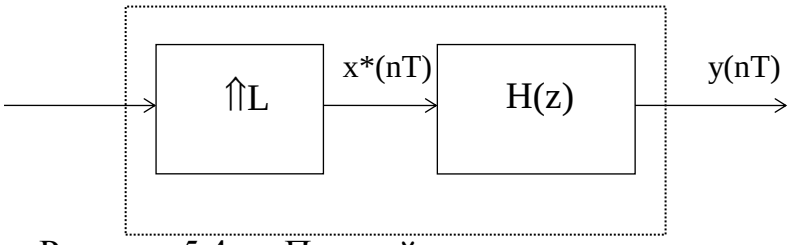

Рисунок 5.4 — Простейшая восходящая дискретная система

ПНДС изображена на рис.5.5. Входной сигнал $x(nT)$ с периодом дискретизации T обрабатывается дискретным ФНЧ с передаточной функцией $H(z)$, АЧХ которого соответствует (5.5). На выходе фильтра установлен дециматор, уменьшающий частоту дискретизации в M раз.

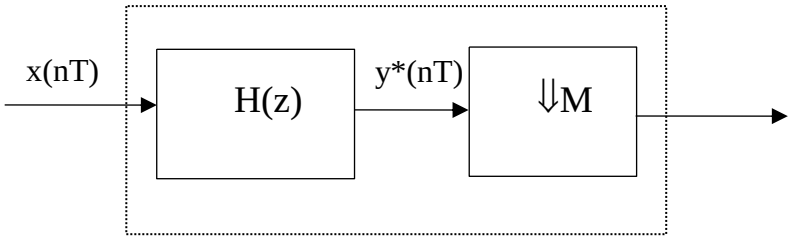

Рисунок 5.5 — Простейшая нисходящая дискретная система

Возможно последовательное соединение ПВДС и ПНДС (рис. 5.6). В этом случае выполняется изменение частоты дискретизации в L/M раз, т.е. возможно дробное преобразование частоты [1]. В этом случае частота среза фильтра нижних частот H(z) равна π/T', если $L > M$, и равна π/T , если $L < M$.

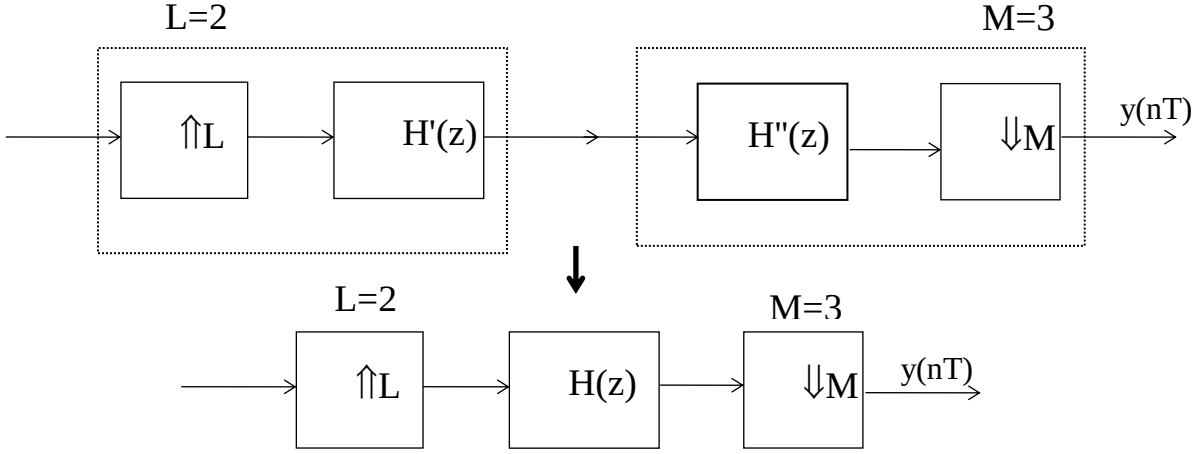

Рисунок 5.6 — Дробное преобразование частоты дискретизации

В системах с изменением частоты дискретизации преимущественное применение находят нерекурсивные фильтры с конечной импульсной характеристикой (КИХ), так как они, благодаря линейности ФЧХ, позволяют сохранять форму сигнала.

В Scilab имеется функция **wfir** для проектирования КИХ фильтров с линейной ФЧХ методом оконных функций [1]. Формат вызова функции:

[h, Hw,fr]=wfir(ftype,forder,cfreq,wtype,fpar),

где **ftype** – строка, задающая тип фильтра: **'lp'** - ФНЧ,**'hp'** – ФВЧ, **'bp'** – ПФ ,**'sb'** – заграждающий фильтр; **forder** – порядок фильтра (положительное целое, нечетное для **'hp'** или **'sb'**; **cfreq** – вектор частот среза фильтра (0<**cfreq(1),cfreq(2)**<.5), для **'lp'** или **'hp'** задается только **cfreq(1)**; **wtype** – тип используемого окна: **'re'** – прямоугольное,**'tr**'–треугольное, **'hm'** – Хэмминга, '**hn'**–Ханна, **'kr'** – Кайзера, **'ch'** – Чебышева); **fpar** – вектор параметров окна, для окна Кайзера **fpar(1)>0** и **fpar(2)=0**, для окна Чебышева **fpar(1)>0**, **fpar(2)<0** или **fpar(1)<0** и **0<fpar(2)<.5**; **h** – отсчеты импульсной (весовой) функции фильтра; **Hw** – отсчеты АЧХ фильтра на сетке частот **fr.** Пример вызова функции для проектирования ФНЧ с использованием окна Хэмминга, с нормированной частотой среза 0.2 и порядком, равным 33 (рис. 5.7):

```
[h,Hw,fr]=wfir("lp",33,[.2 0],"hm",[0 0])
figure()
subplot(1,2,1)
plot(h)
xtitle('Импульсная характеристика')
subplot(1,2,2)
plot(fr, Hw)
xtitle('АЧХ')
```

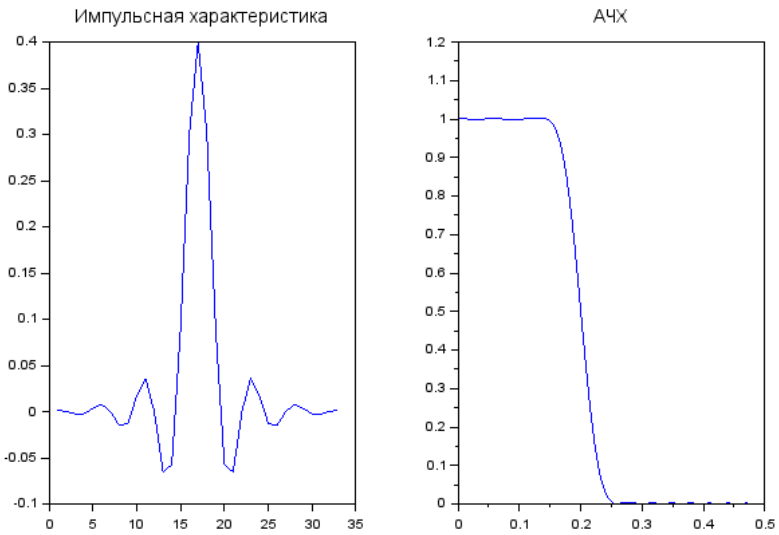

Рис. 5.7 – Импульсная и частотные характеристики ФНЧ

5.3. Варианты заданий

Варианты заданий приведены в таблице 5.1.

Таблица 5.1 — Варианты заданий

Вариант	f1,Гц	f2,Гц	$f_д$ ',Гц	$f_д$ '',Гц	$f_д$ ''',Гц
1	750	1050	8000	2000	40000
2	1000	1550	9000	3000	36000
3	800	2550	10000	5000	30000
4	900	1150	11000	2200	22000
5	850	1550	12000	3000	60000
6	1200	1675	13000	3250	65000
7	2350	3550	14000	7000	42000
8	750	1550	15000	3000	30000
9	1500	2050	16000	4000	80000
10	900	2050	8000	4000	16000
11	1700	2300	9000	4500	27000
12	560	1050	10000	2000	20000
13	870	1425	11000	2750	55000
14	1350	2050	12000	4000	48000
15	2700	3300	13000	6500	39000
16	950	1800	14000	3500	70000
17	2600	3800	15000	7500	45000
18	1200	1650	16000	3200	32000
19	500	8050	8000	1600	16000
20	600	1175	9000	2250	45000
21	850	1300	10000	2500	50000
22	2000	2800	11000	5500	33000
23	750	1250	12000	2400	24000

24	1100	1350	13000	2600	26000
25	1200	1450	14000	2800	28000
26	1900	2550	15000	5000	30000
27	3850	4050	16000	8000	48000
28	760	950	9000	1800	18000
29	880	1050	10000	2000	20000
30	1350	1425	11000	2750	55000

5.4 Порядок выполнения лабораторной работы

5.4.1 Исследовать ПНДС:

– сформировать двухтональный дискретный сигнал $x(t)=\sin(2\pi f_1 t)+0.75\sin(2\pi f_2 t)$ на интервале длительностью 1 секунда и с частотой дискретизации $f_д'$, построить график отсчетов сигнала $x(t)$;

– используя окно Хемминга построить спектр сигнала $x(t)$ с конечным значением по оси частот $f_д'$;

– для выходной частоты дискретизации $f_д''$ определить коэффициент децимации $f_д'/ f_д''$;

– синтезировать ФНЧ ПНДС с помощью функции **wfir** пакета Scilab, рекомендуется использовать порядок фильтра **forder=51** ;

– построить графики импульсной и амплитудно-частотной характеристик ФНЧ, конечное значение на оси частот АЧХ должно быть равно $f_д'/2$;

– реализовать ПНДС в соответствии с рис. 5.5, фильтрацию сигнала выполнить с использованием функции **filter** пакета Scilab, для децимации выходного сигнала фильтра использовать следующий фрагмент кода:

```
y_star=filter(h,1,x);                  // фильтрация сигнала x
Ny=floor(N|M);
i=(1:1:Ny);                            // новая индексация
y=y_star(i*M);                         // децимация,
```

где **M** — коэффициент децимации; **N** — размерность вектора **x**, хранящего отсчеты сигнала $x(t)$;

– построить график выходного сигнала **y** ПНДС;

– используя окно Хемминга построить спектр сигнала **y**, конечное значение на оси частот должно быть равно $f_д''/2$;

– выполнить анализ полученных результатов и сделать выводы.

5.4.2. Исследовать ПВДС:

– сформировать двухтональный дискретный сигнал на интервале 0,1 секунды: $x(t)=\sin(2\pi f_1 t)+0.75\sin(2\pi f_2 t)$ с частотой дискретизации $f_д'$;

– используя окно Хемминга построить спектр сигнала $x(t)$, конечное значение по оси частот должно быть равно $f_д'$;

– определить коэффициент интерполяции $L= f_д'''/ f_д'$;

– выполнить интерполяцию сигнала $x(t)$ с использованием кода:

```
x_star=zeros(1,N*L);
i=(1:1:N);
x_star(i*L)=x(i);  // создание копии x, дополненной нулями,
```

где **L** — коэффициент интерполяции; **N** — размерности вектора **x**, хранящего отсчеты сигнала $x(t)$.

– используя окно Хемминга построить спектр сигнала **x_star**, конечное значение по оси частот должно быть равно $f_д'''$;

– синтезировать ФНЧ ПВДС с помощью функции **wfir** пакета Scilab. порядок фильтра — 101.

– построить импульсную и амплитудно-частотную характеристики синтезированного фильтра, конечное значение на оси частот должно быть равно равно $f_д'''$;

– реализовать ПВДС, выполнив фильтрацию вектора **x_star**:

```
y=L*filter(h,1, x_star);
```

– построить реализацию и спектр сигнала **y**;

– выполнить анализ полученных результатов и сделать выводы.

5.5 Содержание отчета

Цель работы, постановка задачи, краткие теоретические сведения, текст программы с комментариями, временная реализация и спектр двухтонального дискретного сигнала, частотная и импульсная характеристики ФНЧ ПНДС, временная реализация и спектр сигнала после фильтрации ФНЧ ПНДС, временная реализация и спектр сигнала после децимации ПНДС, временная реализация и спектр сигнала после интерполяции ПВДС, частотная и импульсная характеристики ФНЧ ПВДС, временная реализация и спектр сигнала после фильтрации ФНЧ ПВДС, выводы по работе.

5.6. Контрольные вопросы

5.6.1. Что понимают под ВДС?

5.6.2. Что понимают под НДС?

5.6.3. Сформулируйте алгоритм функционирования интерполятора?

5.6.4. Как соотносятся спектры входного и выходного сигналов интерполятора?

5.6.5. Какие требования предъявляются к ФНЧ ПВДС?

5.6.6. Сформулируйте алгоритм функционирования дециматора?

5.6.7. Как соотносятся спектры входного и выходного сигналов децимато-ра?

5.6.8. Какие требования предъявляются к ФНЧ ПНДС?

5.6.9. Каким образом выполняется дробное преобразование частоты?

5.6.10. Какие требования предъявляются к фильтру при дробном преобразовании частоты?

Лабораторная работа № 6
КРАТКОВРЕМЕННЫЙ СПЕКТРАЛЬНЫЙ АНАЛИЗ И ГОМОМОРФНАЯ ОБРАБОТКА РЕЧЕВЫХ СИГНАЛОВ

6.1 Цель работы

Исследование спектральных методов обработки речевых сигналов. Определение параметров речевых сигналов на основе частотного и кепстрального анализа.

6.2 Краткие теоретические сведения

Голосовой аппарат человека представляет собой акустическую систему, состоящую из ротового и носового каналов, возбуждаемую квазипериодическими колебаниями голосовых связок и турбулентным шумом [1].

Звуки в этой системе образуются тремя способами. Вокализованные (звонкие) звуки — путем возбуждения голосового тракта квазипериодическими импульсами воздуха, создаваемыми легкими и вибрациями голосовых связок. Фрикативные звуки образуются проталкиванием воздуха через сужения в определенных областях голосового тракта, в результате чего возникает турбуленция, которая является источником шума, возбуждающего голосовой тракт. Взрывные звуки образуются путем создания избыточного давления при смыкании губ с последующим их быстрым размыканием. Все эти источники создают широкополосное возбуждение голосового тракта, который в свою очередь действует как линейный фильтр с изменяющимися во времени параметрами.

На рис. 6.1 приведена цифровая модель формирования речевого сигнала.

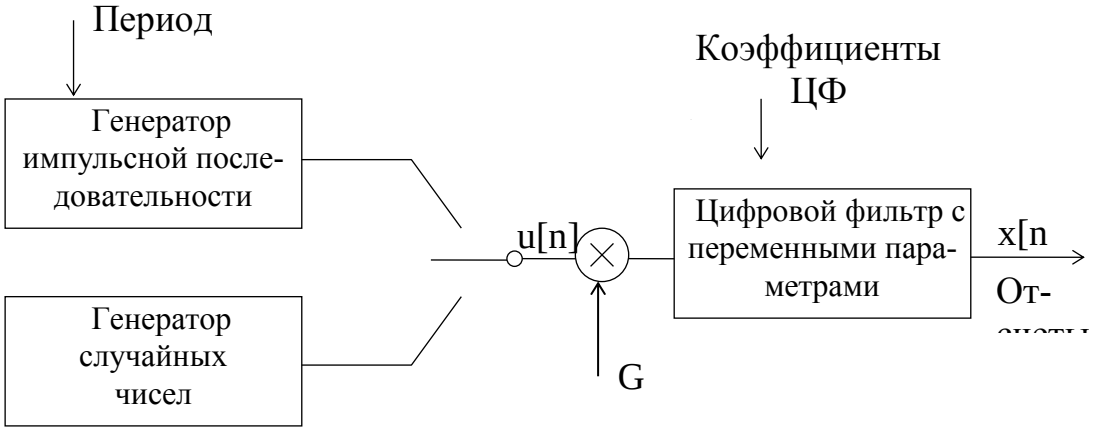

Рисунок 6.1— Цифровая модель формирования речевого сигнала

Предполагается, что в этой модели дискретные отсчеты речевого сигнала формируются на выходе цифрового фильтра (ЦФ) с переменными параметрами, который аппроксимирует передаточную функцию голосового тракта. На временном интервале порядка 10-30 мс характеристики ЦФ можно считать неизменными. На каждом таком интервале ЦФ может быть задан совокупностью своих коэффициентов. В случае вокализованной речи ЦФ возбуждается генератором

квазипериодической импульсной последовательности, расстояние между соседними импульсами которого соответствует периоду основного тона. На интервалах невокализованной речи ЦФ возбуждается генератором случайных чисел, который вырабатывает шумовой сигнал с равномерной спектральной плотностью. В обоих случаях сигнал, поступающий на вход ЦФ, масштабируется с коэффициентов *G*.

В лабораторной работе рассматриваются кратковременный спектральный анализ и гомоморфная обработка речи как наиболее широко используемые способы обработки речевых сигналов.

В основу кратковременного спектрального анализа положено фундаментальное допущение о том, что на коротких интервалах времени речевой сигнал является стационарным. Кратковременный спектральный анализ речи используется в кодировщиках речи, спектрографах, системах распознавания речи. Для вычисления кратковременного спектра речи применяется алгоритм быстрого преобразования Фурье (БПФ).

Кратковременное дискретное преобразование Фурье определяется следующим образом [1]

$$X_l(j\omega) = T_0 \sum_{n=0}^{N-1} x_l[n] e^{-j\omega n T_0} \quad , \tag{6.1}$$

где T_0 – период дискретизации, $x_l[n]$ – отрезок речи, взвешенный окном $w[n]$, длиной N отсчетов:

$$x_l[n] = w[n]x[n] \quad . \tag{6.2}$$

Частотное разрешение спектрального анализа обратно пропорционально длине окна N, что иллюстрируется на рис. 6.2. В качестве функции окна здесь использовано окно Хемминга.

При частоте дискретизации f_s=22050Гц и значении N=512 частотное разрешение спектра равно f_s/N=43 Гц (узкополосный спектр, рис.6.2). Заметим, что такое разрешение позволяет выявить каждую гармонику частоты основного тона в отдельности. Если для построения спектра использовать N=64, то разрешающая способность по частоте снижается и отдельные гармоники становятся не различимыми (широкополосный спектр, рис. 6.2.). Это можно использовать для оценки огибающей спектра.

При узкополосном кратковременном спектральном анализе сигнал возбуждения голосового тракта $u[n]$ (см. рис. 6.1) проявляется в виде узких пиков в спектре вокализованных речевых сигналов на частотах, кратных частоте основного тона. Этот факт положен в основу алгоритма вычисления частоты основного тона путем произведения гармоник спектра [1]:

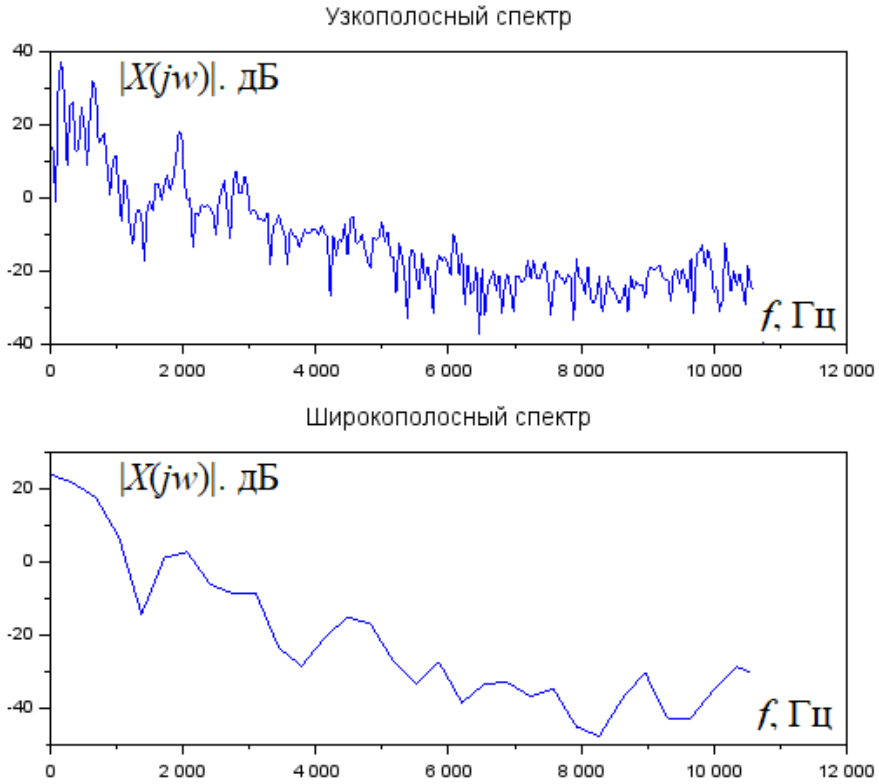

Рисунок 6.2— Узкополосный и широкополосный спектры вокализованного речевого сигнала

$$F_l(\omega) = \prod_{k=1}^{K} \left| X_l(\omega k) \right|^2 \qquad (6.3)$$

Взяв логарифм, получим сумму логарифмов гармоник спектра:

$$\overline{F}_l(\omega) = 2 \cdot \sum_{k=1}^{K} \log \left| X_l(\omega \cdot k) \right| \ .$$

Видно, что $\bar{F}_l(\omega)$ представляет собой сумму из K сжатых по частоте составляющих $\log|X_n(\omega k)|$. Введение функции $F_l(\omega)$ мотивируется тем, что в вокализованной речи сжатие частотной шкалы в целое число раз должно привести к усилению гармоники на частоте основного тона. Было обнаружено, что этот метод устойчив к шумам, поскольку вклад шумов в $X_l(\omega)$ не имеет коррелированной структуры. По этой же причине невокализованная речь не даст пика в $F_l(\omega)$. По положению максимального значения в $F_l(\omega)$ можно определить мгновенную частоту основного тона вокализованного участка речи.

При изучении речи одним из полезных инструментов является спектрограмма. Спектрограмма представляет изображение распределения гармонических составляющих речевого сигнала в виде функции частоты и времени. Иными словами, спектрограмма – это функция $|X_l(\omega)|$ в координатах частота-время-интенсивность. Пример спектрограммы изображен на рис. 6.3.

В пакете Scilab имеется специальная функция **mapsound(x)** для построения спектрограмм. Функция **mapsound(x)** разрезает входной сигнал **x** на последовательные фрагменты одинаковой длительности **Dt**. Затем для каждого фрагмента

вычисляется дискретное преобразование Фурье (ДПФ). Абсолютные значения амплитуд спектральных составляющих для фрагмента с номером **i** отображаются в момент времени **i*Dt**. Функция может быть вызвана в расширенном формате:

mapsound(x, Dt, freqRange, fs, Colormap),

где **x** — вектор отсчетов речевого сигнала; **Dt** — шаг дискретизации времени при построении спектрограммы, соответствует временному интервалу анализа спектра; **freqRange** — вектор **[fmin, fmax]**, задающий диапазон частот спектральных отсчетов, если задано скалярное значение, то специфицируется только **fmax**; **fs** — частота дискретизации, по умолчанию **fs**=22050Гц; **Colormap** — цветовая палитра, например, **cool, parula** и др. Если **Dt** не указан, **mapsound(…)** вычисляет его, чтобы иметь (почти) одинаковое количество частотных и временных интервалов с максимально возможным (временным, частотным) разрешением.

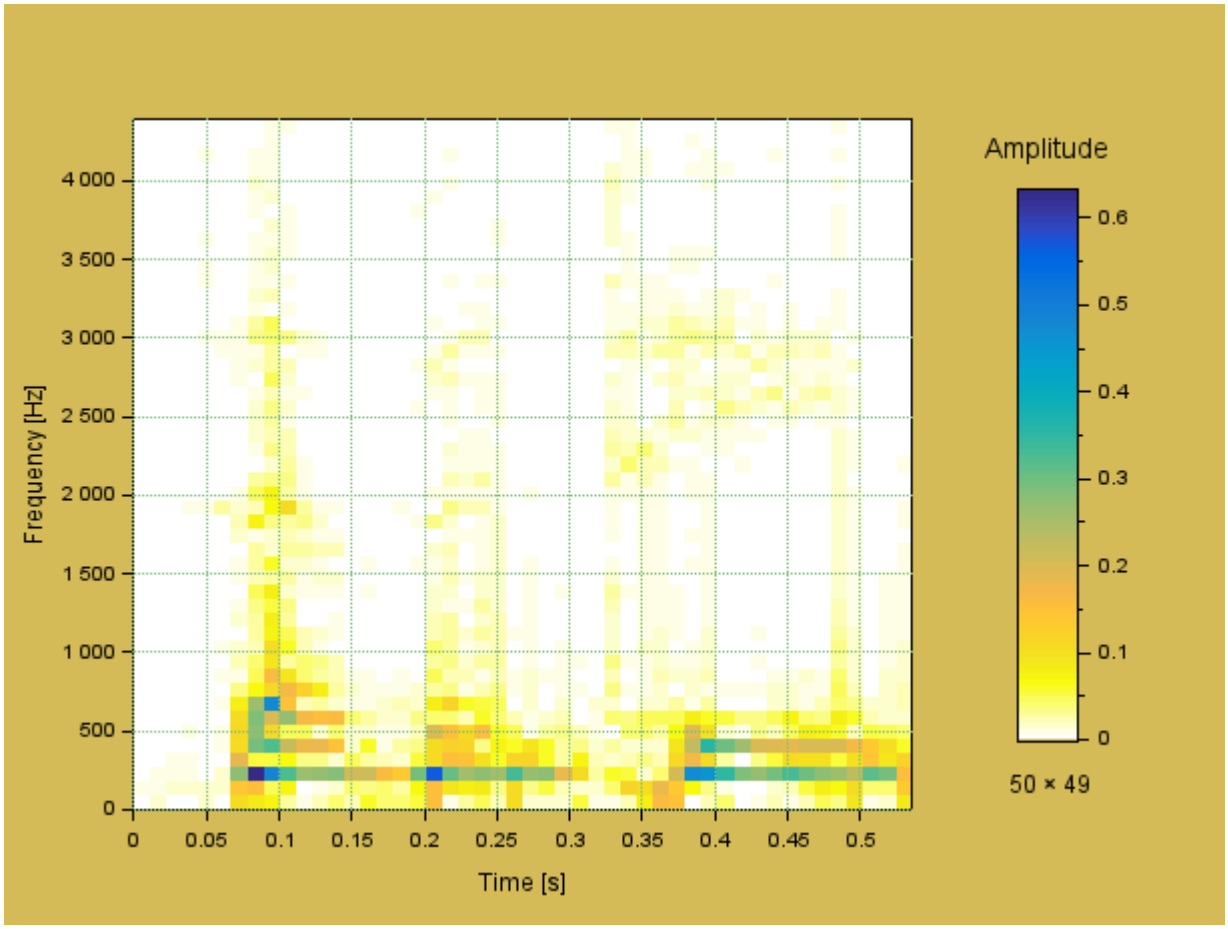

Рисунок 6.3 — Спектрограмма речи

В соответствии с рис.6.1 речевой сигнал $x[n]$ является сверткой функции возбуждения $u[n]$ (случайного шума либо квазипериодической последовательности импульсов) и импульсной характеристики голосового тракта (цифрового фильтра). Гомоморфный анализ речи позволяет разделить эти компоненты. Схема гомоморфного анализа речи изображена на рис. 6.4.

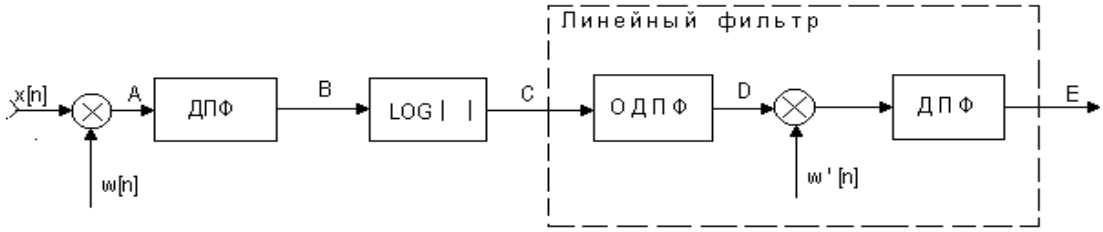

Рисунок 6.4 — Схема гомоморфного анализа речи

ДПФ от речевого сигнала $x[n]$ (точка B на рис. 6.4) будет представлять собой произведение Фурье преобразований функции возбуждения и импульсной характеристики цифрового фильтра. Вычислив логарифм этого произведения, получим (точка C на рис. 6.4) сумму логарифмов спектра функции возбуждения и частотной характеристики голосового тракта. Поскольку обратное дискретное преобразование Фурье (ОДПФ) является линейной операцией, сигнал в точке D, называемый *кепстром,* равен результату сложения кепстров функции возбуждения и голосового тракта. Результаты указанных преобразований применительно к вокализованной речи представлены на рис. 6.5.

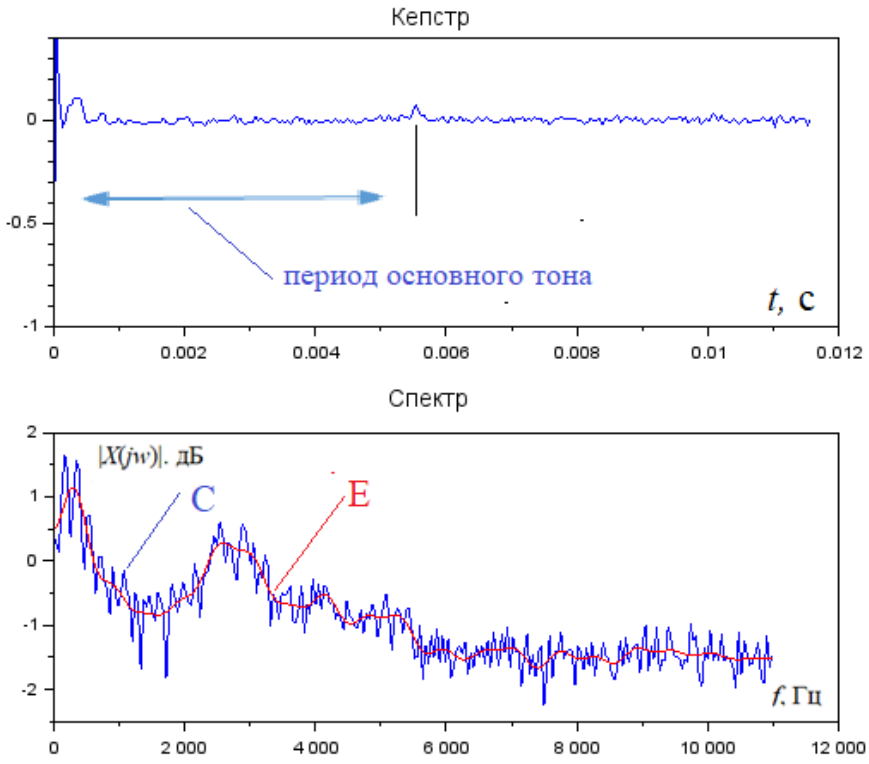

Рисунок 6.5 — Кепстр и спектр вокализованной речи

Пульсирующая кривая, помеченная буквой C на нижнем графике, представляет логарифм кратковременного спектра. Она содержит медленноменяющуюся составляющую, обусловленную передаточными свойствами голосового тракта и быстроменяющуюся периодическую составляющую, обусловленную периодическим сигналом возбуждения. Медленноменяющаяся составляющая

формирует в кепстре (точка D) область малых времен, а быстроменяющаяся составляющая проявляется в виде небольшого пика в кепстре, возникающего через интервал времени, равный периоду основного тона речевого сигнала. Важно отметить, что кепстр является аддитивной комбинацией, в которой составляющие голосового тракта и сигнала возбуждения практически не перекрываются. Таким образом, кепстр весьма удобен для определения периода основного тона вокализованной речи и для определения характера — вокализованный или невокализованный — данного образца речи.

Амплитудно-частотная характеристика голосового тракта, т.е. огибающая спектра, может быть получена посредством линейной фильтрации прологарифмированных значений спектра, в результате которой будут подавлены его быстроменяющиеся составляющие. Один из возможных способов такой фильтрации состоит перемножении кепстральных значений на подходящее временное окно, пропускающее лишь составляющие из области малых времен с последующим вычислением ДПФ кепстра. Кривая, полученная в результате такой обработки, обозначена буквой E (см. рис. 6.5).

6.3 Варианты заданий

Варианты заданий приведены в таблице 6.1

Таблица 6.1 — Варианты заданий

Номер варианта	Файл 1	Файл 2
1	ta0	ta50
2	ta1	ta40
3	ta2	ta30
4	ta3	ta20
5	ta4	ta10
6	ta5	ta19
7	ta6	ta18
8	ta7	ta17
9	ta8	ta16
10	ta9	ta15
11	ta10	ta14
12	ta11	ta13
13	ta12	ta20
14	ta13	ta11
15	ta14	ta10
16	ta15	ta9
17	ta16	ta8
18	ta17	ta7
19	ta18	ta6
20	ta19	ta5
21	ta20	ta3
22	ta30	ta2
23	ta40	ta1
24	ta50	ta0
25	ta0	ta5

26	ta1	ta10
27	ta2	ta20
28	ta3	ta30
29	ta4	ta40
30	ta5	ta50

6.4 Порядок выполнения лабораторной работы

6.4.1. Выполнить анализ речи с помощью кратковременного спектрального анализа:

- с помощью функции **wavread(…)** ввести данные из 2-х заданных файлов, соответствующих варианту задания, и сохранить их в векторах **x1** и **x2;**
- построить графики временных реализаций сигналов **x1** и **x2;**
- путем просмотра реализаций сигналов **x1** и **x2** выбрать по одному вокализованному и невокализованному участку для каждого из сигналов **x1** и **x2;**
- выполнить узкополосный и широкополосный спектральный анализ вокализованных участков каждого из сигналов **x1** и **x2**, построить графики соответствующих реализаций и логарифмов модулей их спектров;
- по результатам спектрального анализа определить частоту основного тона, пользуясь (6.3) (K=5). Построить графики $F_l(\omega)$ для вокализованных и невокализованных участков сигналов **x1** и **x2;**
- построить спектрограммы речевых сигналов **x1** и **x2.**

6.4.2. Выполнить гомоморфный анализ речи:

- для вокализованных участков речи **x1** и **x2** выполнить их обработку по схеме, представленной на рис. 6.4;
- представить на графиках процессы в точках C, D, E;
- определить по кепстру частоту основного тона.

6.5 Содержание отчета

Цель работы, постановка задачи (вариант задания), краткие теоретические сведения с расчетными формулами, тексты программ обработки с комментариями, графики сигналов **x1** и **x2**, узкополосный и широкополосный спектры вокализованных и невокализованных участков речи, графики $F_l(\omega)$, значение частоты основного тона, спектрограммы сигналов **x1** и **x2**, графики процессов в точках C,D,E в соответствии со схемой гомоморфной обработки, значения частоты основного тона, определенное по кепстру, выводы по работе.

6.6 Контрольные вопросы

6.6.1. Как формируются вокализованные звуки в соответствии с цифровой моделью формирования речевого сигнала?

6.6.2. Как формируются фрикативные звуки в соответствии с цифровой моделью формирования речевого сигнала?

6.6.3. Опишите цифровую модель формирования речи?
6.6.4. Объясните суть кратковременного спектрального анализа?
6.6.5. Что такое узкополосный и широкополосный спектральный анализ?
6.6.6. Как по кратковременному спектру определить частоту основного тона?
6.6.7. Основные этапы гомоморфной обработки речи? Каким образом гомоморфная обработка позволяет разделить функцию возбуждения и импульсную характеристику голосового тракта?
6.6.8. Что такое кепстр, и как определить частоту основного тона по кепстру?
6.6.9 Объясните на графиках, что понимают под быстроменяющейся и медленноменяющейся составляющими спектра речи?
6.6.10. Объясните на графиках, что понимают под областью малых и больших времен кепстра?
6.6.11. Чему на графике спектра соответствует область малых (больших) времен кепстра?
6.6.12. Каким должно быть временное окно, что сохранить область малых времен кепстра?

Лабораторная работа № 7
ЛИНЕЙНЫЙ ПРЕДИКТИВНЫЙ АНАЛИЗ РЕЧИ

7.1 Цель работы

Исследование метода линейного предсказания речи, определение коэффициентов линейного предсказания, анализ свойств ошибки предсказания речи, определение значений формантных частот речи.

7.2 Краткие теоретические сведения

Способ анализа речи на основе линейного предсказания базируется на использовании цифровой модели речи, изображенной на рис. 6.1. Предположим, что цифровой фильтр на рис. 6.1 является рекурсивным фильтром с передаточной функцией вида [1]

$$H(z) = \frac{G}{1 + \sum_{k=1}^{p} a[k] \cdot z^{-k}}\ , \qquad (7.1)$$

где G, $a[k]$ – коэффициенты фильтра, p – порядок фильтра. Корни полинома в знаменателе (7.1.) называют полюсами. АЧХ фильтра имеет пики на частотах, соответствующих полюсам. Поэтому фильтр (7.1) при надлежащем выборе коэффициентов может успешно моделировать формантные частоты речи. Модель речи, основанную на использовании фильтра (7.1), также называют авторегрессионной (АР) моделью.

Передаточной функции (7.1) соответствует следующее выражение, определяющее отчеты речи на выходе фильтра $x[n]$

$$x[n] = -\sum_{k=1}^{p} a[k] \cdot x[n-k] + G \cdot u[n].\ , \qquad (7.2)$$

где $u[n]$ – входной сигнал фильтра. Отметим, что значения входного сигнала фильтра (функции возбуждения фильтра) $u[n]$ неизвестны. В случае вокализованной речи входной сигнал фильтра представляется в виде серии импульсов, которая формируется с помощью генератора импульсной последовательности, а для невокализованной речи — белым шумом, формируемым генератором случайных чисел.

В ходе анализа речи нам известны предыдущие значения речевых отсчетов $x[n]$, $x[n\text{-}1]$, $x[n\text{-}2]$ и т.д. Поэтому текущий отсчет речи $x[n]$ можно предсказать по предыдущим отсчетам речи, используя формулу линейного предсказания

$$\tilde{x}[n] = -\sum_{k=1}^{p} \hat{a}[k] x[n-k]. \qquad (7.3)$$

Коэффициенты $\hat{a}[k]$ называют коэффициентами линейного предсказания (ЛПК). Ошибка предсказания будет равна

$$e[n] = x[n] - \tilde{x}[n] = x[n] + \sum_{k=1}^{p} \hat{a}[k]\, x[n-k]. \qquad (7.4)$$

Один из способов определения ЛПК основан на минимизации квадрата ошибки предсказания (7.4):

$$E_l = \sum_{n=0}^{N-1} e^2[n] = \sum_{n=0}^{N-1} \left[x_l[n] + \sum_{k=1}^{p} \hat{a}[k] x_l[n-k] \right]^2, \tag{7.5}$$

где $x_l[n]$ — стационарные участки речи. Для нахождения $\hat{a}[k]$, обеспечивающих минимум (7.5), необходимо решить систему уравнений

$$\frac{\partial E}{\partial \hat{a}[k]} = 0 \qquad ,1 \le k \le p \tag{7.6}$$

.

Выполнив дифференцирование, получим систему нормальных уравнений:

$$\sum_{n=0}^{N-1} x_l[n-i] x_l[n] = \sum_{k=1}^{p} \hat{a}[k] \sum_{n=0}^{N-1} x_l[n-k] x_l[n-i], 1 \le i \le p. \tag{7.7}$$

Обозначив

$$r[i] = \sum_{n=0}^{N-i-1} x_l[n] x_l[n+i], \tag{7.8}$$

перепишем систему (7.7) в компактной форме:

$$r[i] = \sum_{k=1}^{p} \hat{a}[k] \cdot r[i-k] \qquad ,1 \le i \le p, \tag{7.9}$$

где r — автокорреляционная функция участка речи $x_l[n]$, который получают путём умножения $x[n]$ на временное окно $w[n]$ длиной в N отсчётов.

Существует несколько процедур решения системы уравнений (7.9). Одна из процедур нахождения коэффициентов линейного предсказания использует рекурсивный алгоритм Левинсона-Дарбина [1].

Коэффициенты линейного предсказания широко применяются при решении следующих задач анализа речи:

1. Оценка спектра речи. Коэффициенты ЛПК позволяют определить амплитудно-частотную характеристику (АЧХ) фильтра:

$$| H(e^{j\omega T_0}) | = | \frac{G}{1 + \sum_{k=1}^{P} \hat{a}[k] e^{-jk\omega T_0}} |. \tag{7.10}$$

АЧХ соответствует медленно меняющейся составляющей кратковременного спектра речевого сигнала. На рис. 7.1 представлены в логарифмическом масштабе: кратковременный спектр речевого сигнала $S(f)$ и АЧХ фильтра $H(f)$, вычисленная с помощью (7.10). Из рисунка 7.1. видно, что АЧХ фильтра представляет огибающую кратковременного спектра речи (p=28).

2. Оценка формантных частот. Частоты пиков на графике АЧХ соответствуют форматным частотам. Эти пики определяются полюсами передаточной функции (7.1).

3. Определение периода основного тона. Отметим, что ошибка предсказания (7.4) с учетом (7.2) будет похожа на функцию возбуждения $u[n]$. Поэтому для вокализованных звуков по $e[n]$ можно определить период основного тона.

4. Синтез речи. Коэффициенты линейного предсказания и информация о функции возбуждения могут быть использованы для синтеза речи по схеме, изображенной на рис. 6.1.

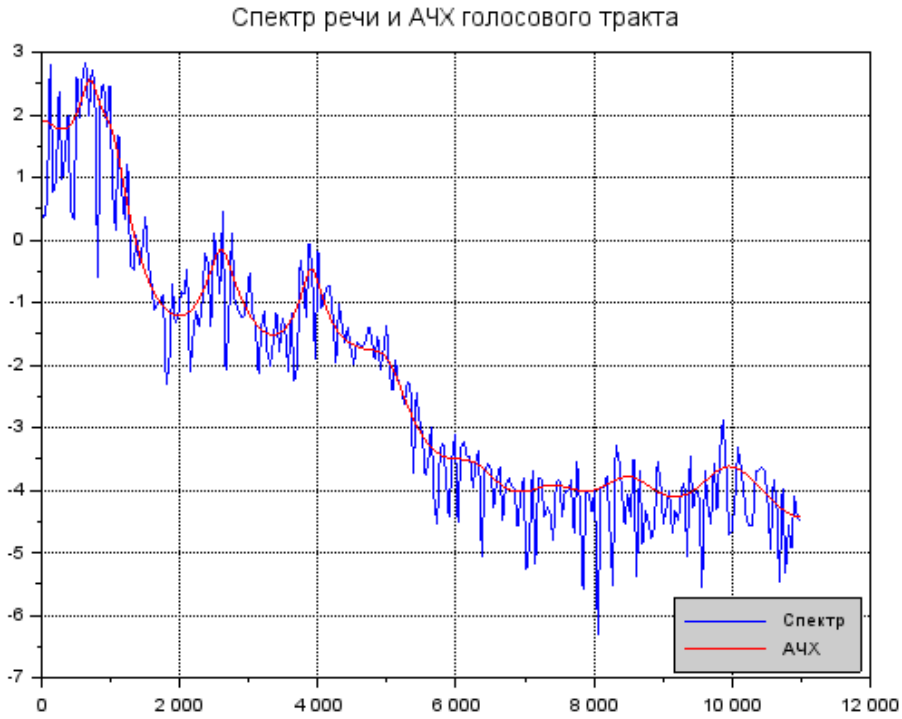

Рисунок 7.1 — Кратковременный спектр и АЧХ голосового тракта

7.3 Варианты заданий

Варианты заданий приведены в табл. 7.1.

Номер варианта	Имя файла
1	яхта
2	пять
3	палка
4	цель
5	пень
6	синего
7	ты
8	облако
9	тётя
10	пуля
11	трюм
12	ива
13	ютить
14	мой
15	трава
16	река
17	дом

7.4 Порядок выполнения лабораторной работы

7.4.1. Загрузить файл, заданный вариантом задания в оперативную память. Понизить частоту дискретизации записанного в файле речевого сигнала в два раза. Прослушать речевой сигнал и построить его график во времени. Выбрать вокализованный участок речевого сигнала, соответствующий гласному ударному звуку. Выполнить кратковременный узкополосный и широкополосный анализ вокализованного участка речи.

7.4.2. Выполнить гомоморфный спектральный анализ этого же участка речи.

7.4.3. Определить ЛПК-коэффициенты для выбранного участка речи. Для этого необходимо сначала вычислить автокорреляционную функцию (7.8) вокализованного участка речи. В пакете Scilab имеется встроенная функция

```
cor = corr(x,nlags)
```

с помощью, которой можно вычислить (7.8). Здесь **x** – вектор, представляющий участок речи, **nlags** – число вычисляемых коэффициентов корреляции, **cor** – возвращаемый вектор значений корреляционной функции.

При наличии значений корреляционной функции коэффициенты линейного предсказания речи могут быть вычислены с помощью функции Левинсона

```
[la, lmse]= levin(p, cor),
```

где **p** – максимальный порядок линейного предсказателя, **la** – список полиномов знаменателя передаточной функции (7.1) со степенями от 1 до **p**, **lmse** – список средних квадратов ошибок предсказателей (7.3) разных порядков. Функция Левинсона рекурсивно решает систему нормальных уравнений (7.9) при разных **p**.

Указанную последовательную обработку речи рекомендуется определить в виде функции **lpc**, вычисляющей коэффициенты линейного предсказания:

```
function [a, mse, cor]=lpc(x, p);
    cor=corr(x,length(x)); // вычисление автокорреляции
    sz = size(cor);
    if sz(1) == 1 then // транспонирование, если cor вектор -строка
       cor = cor';
    end
    [la,lmse]=levin(p,cor); //решение системы нормальных уравнений
    a = la($);  //  возвращаем полином максимальной степени
    mse = lmse($); // возвращаем с.к.о. предсказателя  макс. степени
endfunction
```

Извлечь коэффициенты $A[k]$ из полинома **a** можно с помощью функции **A=coeff(a),** где **A** – вектор коэффициентов линейного предсказания.

7.4.4. Построить график автокорреляционной функции **cor.** Оценить по графику значение периода основного тона участка речи.

7.4.5. Построить график спектра речи по ЛПК, вычислив значения АЧХ фильтра (7.1) на основе (7.10). Для вычисления и построения графика АЧХ рекомендуется воспользоваться функцией **frmag:**

```
[Hw, fr] = frmag(1, A, Nh);
plot(fr*fs, log(Hw),'g')
```

Здесь **Nh** – число отсчетов АЧХ, **Hw** – значения АЧХ на нормированных частотах **fr**. Значения **fr** нормированы относительно частоты дискретизации **fs**.

7.4.6. Выполнить сравнение графиков спектров, полученных при выполнении п.п. 7.4.1-7.4.5, построив их в едином масштабе в одном графическом окне. Подобрать порядок предсказателя **p** таким образом, чтобы спектр речи, вычисленный на основе ЛПК, хорошо аппроксимировал кратковременный спектр в районе первых 3-4 форматных частот.

7.4.7. Вычислить ошибку $e[n]$, используя (7.4) и построить график $e[n]$. Определить частоту основного тона по графику $e[n]$.

7.4.8. По графику спектра речи, полученном в п. 7.4.4 методом линейного предсказания, определить первые 3-4 формантные частоты. Также определить числовые значения формантных частот, путем нахождения полюсов передаточной функции (7.1). Обычно полюсы образуют комплексно-сопряженные пары, при четном **p** число пар равно **p**/2. Для определения частоты полюса (формантной частоты) необходимо значения полюса представить в комплексной показательной форме, определить аргумент полюса и выполнить его масштабирование с учетом **fs** :

```
// находим корни (полюсы 7.1) полинома a
roots_a=roots(a);
//находим вектор частот формант путем вычисления
// аргументов полюсов и масштабирования с учетом fs
formants_a=atan(imag(roots_a), real(roots_a))/(2*%pi)*fs;
// сортируем вектор частот по возрастанию частот
a_sorted = gsort(abs(formants_a),'g','i');
disp('Формантные частоты, Гц',a_sorted)
```

7.4.9. Сопоставить вычисленные формантные частоты в 7.4.8 со средними частотами формант гласных (таблица 7. 2) и определить обработанную фонему.

7.5 Содержание отчёта

Цель работы, краткие теоретические сведения, тексты программ, реализующие задания с комментариями, графики вокализованных участков речи, графики узкополосного и широкополосного спектров, график спектра, полученного путём гомоморфной обработки, значения ЛПК-коэффициентов, график спектра, полученного методом линейного предсказания, график корреляционной функции участка анализируемой речи, график ошибки предсказания, значения частот основного тона, значения первых 5-ти формантных частот, выводы.

Таблица 7.2. – Средние частоты формант гласных (в Гц)

Гласные	Частота речевых формант		
	1-я форманта	2-я форманта	3-я форманта
У	300	625	2500
О	535	780	2500
А	700	1080	2600
Е	440	1800	2550
И	240	2250	3200
Ы	300	1480	2230

7.6 Контрольные вопросы

7.6.1. Запишите выражение для передаточной функции линейного предсказателя речи, запишите соответствующее выражение для вычисления выходного сигнала предсказателя во временной области.
7.6.2. Как определяется ошибка предсказания?
7.6.3. Выполните вывод системы нормальных уравнений.
7.6.4.Как вычисляется функция автокорреляции?
7.6.5. Каким методом решается система автокорреляционных уравнений (7.9) ?
7.6.6. Как получить оценку спектра на основе ЛПК-коэффициентов?
7.6.7. Как по графику спектра, полученному на основе коэффициентов линейного предсказания, определить формантные частоты?
7.6.8. Какому сигналу на рис. 6.1 соответствует ошибка предсказания?
7.6.9. Как по графику ошибки предсказания определить период основного тона?
7.6.10. Как определить полюсы передаточной функции фильтра (7.1)?
7.6.11. Как определить формантные частоты, если известны полюсы авторегрессионного фильтра?
7.6.12. Приведите примеры применения линейного пре

Лабораторная работа № 8
ЦИФРОВАЯ ФИЛЬТРАЦИЯ ИЗОБРАЖЕНИЙ

8.1 Цель работы

Исследование способов представления изображений в ЭВМ и алгоритмов двумерной обработки сигналов. Исследование способов улучшения изображений с помощью точечных и локальных операторов двумерной цифровой фильтрации.

8.2 Краткие теоретические сведения

8.2.1 Модуль Scilab для обработки изображений и компьютерного зрения (IPCV)

В Scilab могут использоваться различные модули обработки изображений. В настоящей лабораторной работе используется модуль **Image Processing and Computer Vision Toolbox (IPCV)**. Его установку можно выполнить двумя способами: с использованием пунктов меню IDE Scilab или из командного окна Scilab.

В первом случае необходимо в IDE Scilab выбрать пункты меню «Инструменты → Управление модулями Atoms». В открывшемся окне в разделе «Обработка изображений» выбрать модуль «Image Processing and Computer Vision Toolbox» (рисунок 8.1). Для установки модуля нажать кнопку «Установить» (должно быть подключение к интернету). После загрузки и установки модуля перезапустить Scilab.

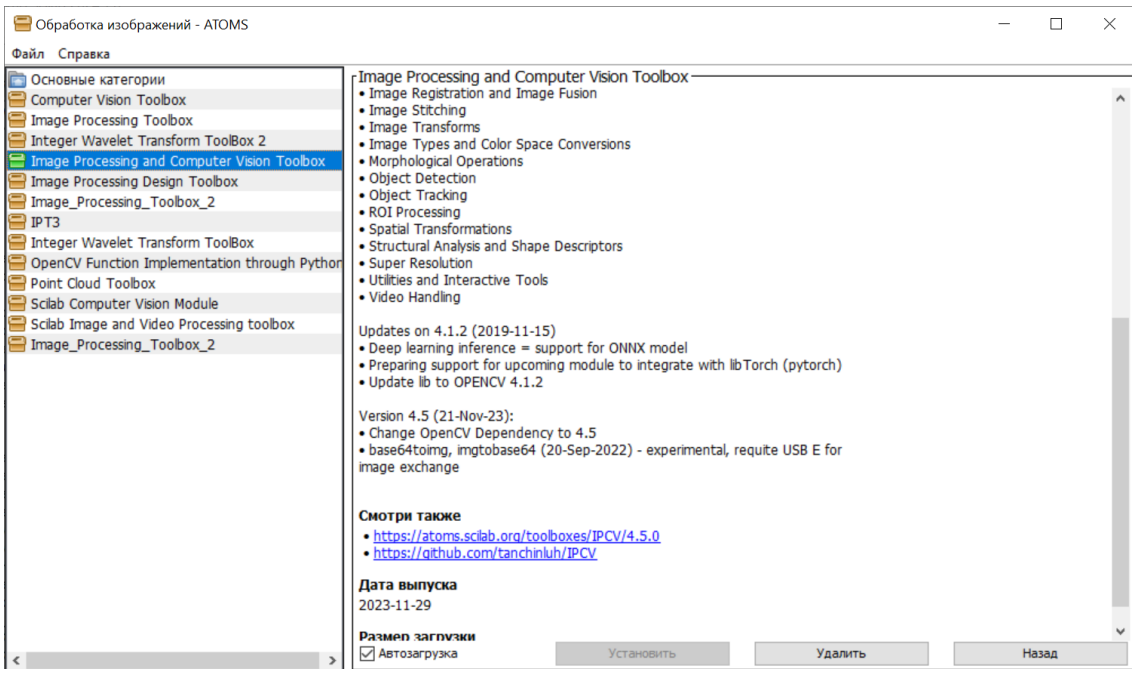

Рисунок 8.1 – Окно выбора модуля Image Processing and Computer Vision Toolbox

При установке модуля из командного окна Scilab необходимо предварительно скачать бинарный архив, подходящий для вашей операционной системы, расположенный по адресу https://atoms.scilab.org/toolboxes/IPCV/4.5.0 . Например, для ОС Windows выберите **IPCV_4.5.0.bin.x64.Windows.** Разархивируйте архив в папку IPCV_4.5.0.bin.x64.Windows и выполните команду:

exec("путь к папке\IPCV_4.5.0.bin.x64.Windows\IPCV\loader.sce")

8.2.2 Представление изображений в IPCV

IPCV, аналогично пакету Matlab, поддерживает четыре формата представления изображений [2, 4]:

- индексированные изображения;
- полутоновые изображения;
- бинарные изображения;
- цветные(RGB) изображения.

Основным типом данных, используемым для представления изображений, является матрица, которая представляет двумерную последовательность чисел $x[m,n]$, кодирующих свойства пикселя изображения с координатами (m,n) [1, 4].

Индексированные изображения используют две матрицы: матрицу цветов и матрицу изображения [4]. Матрица цветов является упорядоченным множеством значений, которые представляют цвета изображения. Каждому пикселю изображений соответствует определенный цвет, задаваемый индексом матрицы цветов. Матрица индексированных изображений состоит из целых чисел, которые являются индексами матрицы цветов. Каждая строка матрицы цветов, выбираемая с помощью индекса, представляет вектор [R, G, B], где R,G,B – вещественные числа, задающие интенсивность соответственно красного, зеленого и синего цветов. Поэтому матрицу цветов называют палитрой. Одно и тоже изображение можно отображать, используя различные палитры **colormap**. Для отображения индексированного изображения **x** с помощью палитры в IPCV можно применить функцию **imshow(x, [colormap])**. Если второй параметр функции **imshow** не задан, то используется палитра по умолчанию.

Полутоновые изображения в IPCV представляются матрицей интенсивностей [4]. Каждый элемент такой матрицы является числом, например, в диапазоне от 0.0 до 1.0. Элемент матрицы интенсивностей соответствует пикселю, отображаемому градациями серого цвета. Нулевому значению интенсивности соответствует черный цвет, а единичному — белый. Полутоновый формат изображений используется для представления черно-белых фотографий, медицинских снимков. Для отображения матрицы интенсивности, которую так же можно рассматривать как двумерную последовательность чисел $x[m,n]$, в IPCV также используют функцию **imshow(x)**, которая умеет отображать изображения различных типов **double (0-1), uint8 (0-255)** и др.

Бинарные изображения представляют частный случай полутоновых изображений, когда используются всего две градации серого цвета. Матрица интенсивностей для бинарного изображения имеет всего два уровня значений

0(черный) и 1(белый). Такая матрица интенсивностей называется логической или булевой. Она может быть получена с помощью вызова функции **im_bw = im2bw(im, thresh)**, где **im** – изображение любого формата, поддерживаемого в IPCV, **tresh** – порог в диапазоне [0,1].

Цветные изображения в большинстве случаев представляют в RGB формате в виде матриц интенсивностей трех цветов: **R** - красный, **G** – зеленый, **B** – синий. Цветные изображения могут быть преобразованы в полутоновые изображения с помощью вызова функции **G=rgb2gray(RGB)**, где **RGB** – цветное изображение размерностью *M*x*N*x3, **G** = 0.299***R** + 0.587***G** + 0.114***B** – матрица интенсивностей серого цвета. В некоторых случаях необходимо выполнять преобразование индексированных изображений в цветные RGB-изображения. Данная операция может быть выполнена с помощью вызова функции **RGB = ind2rgb(x, colormap).** Формат **RGB** позволяет обрабатывать отдельно каждую из матриц интенсивностей **R**, **G** или **B** (например, выполнять фильтрацию): **R=RGB(:,:,1), G=RGB(:,:,2), B=RGB(:,:,3).**

Модуль IPCV содержит и другие функции для преобразования цветовых пространств и типов изображений, например: **rgb2ycbcr, ycbcr2rgb, rgb2hsv, hsv2rgb, rgb2ntsc, ntsc2rgb, rgb2lab, mat2gray, imnorm, im2double, im2int32, im2int16, im2int8, im2uint16, im2uint8** и др.

Загрузить в память заданное изображение можно с помощью вызова функции **im = imread(filename)**, где **filename** – строка с именем файла, содержащего изображение, im – матрица изображения (для RGB изображения – матрица целочисленных значений типа **uint8** размером *M*x*N*x3). Формат изображения определяется по расширению файла. Функция поддерживает работу со следующими расширениями файлов: **jpg, png, tif** и др.

При обработке изображений в большинстве случаев используется система координат, соответствующая индексированию матриц. Например, обращение **x**[2,3] соответствует обращению к свойству (яркости, цвету и др.) третьего пикселя второй строки. Отсчет номеров пикселей начинается с верхнего левого угла изображения.

8.2.3 Основные характеристики изображений

Рассмотрим полутоновое изображение. *Среднее значение* интенсивности изображения определяется выражением [1, 2]

$$m_x = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x[m,n] = \sum_{i=0}^{I-1} P_i u_{i,}$$

где u_i − *i*-й уровень интенсивности (яркости) серого; P_i − частота появления *i*–го уровня интенсивности. Среднее значение позволяет оценить, является ли изображение светлым или темным. Для оценки средней интенсивности изображений в IPCV используется функция **mean2(im)**, где **im** – одноканальное изображение.

Контрастность – разность между максимальным и минимальным уровнями интенсивностей пикселей изображения. Для оценки контрастности изображения

используют стандартное отклонение – квадратный корень из среднего квадрата отклонения интенсивности текущего пикселя от её среднего значения:

$$s_x = \sqrt{\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}(x[m,n]-m_x)^2}.$$

Оценка стандартного отклонения в IPCV может быть выполнена с помощью функции **stdev2(im)**, где **im** – одноканальное изображение.

Для характеристики распределения уровней полутонового изображения применяется **гистограмма** относительных частот появления определенных уровней интенсивности на изображении. Относительная частота интенсивности *i*–го уровня пикселя определяется как

$$P_x(i) = k_i / (M \cdot N), \text{ при } i=0...I-1,$$

где k_i − число пикселей *i*–го уровня интенсивности, *I* – число уровней интенсивности.

В IPCV оценить и отобразить гистограмму распределения уровней интенсивностей пикселей можно с помощью вызова функции **[counts, cells] = imhist(im, bins, [width], [color])**, где: **bins** – число интервалов гистограммы (если не задано, то определяется типом изображения, например для **uint8** и **int8** – 2^8, для **uint16** и **int16** – 2^{16}, для **double** -10); **counts** – число пикселей заданных уровней интенсивности; **cells** – интервалы интенсивностей. Если при вызове **imhist** указано более 2-х аргументов, то гистограмма отображается в графическом окне. В этом случае: **width** – аргумент, передаваемый функции **bar**, обозначает относительную ширину столбца гистограммы на графике (по умолчанию 0.8, т.е. 80%); **color** – аргумент, передаваемый функции **bar**, обозначает цвет столбчатой гистограммы (по умолчанию – ‘**blue’**).

8.2.4 Повышение контрастности изображения

Одним из эффективных методов повышения контрастности является метод ***контрастного растягивания***, при котором заданный диапазон изменения яркости входного изображения **x** линейно растягивается на всю шкалу уровней яркости от 0 до 1 (или 0 - 255) [1]. В IPCV линейное растяжение можно выполнить с помощью функции **imout = imadjust(im,src,dest)**, где **im** – входное изображение, **src** – диапазон интенсивностей [min max] гистограммы входного изображения, **dest** – диапазон интенсивностей [min max] гистограммы выходного изображения **imout**. Здесь значения min и max должны быть в диапазоне от 0 до 1.

Иная возможность повышения контрастности основана на выполнении такого преобразования *T(i)* уровней интенсивностей пикселей исходного изображения, которое обеспечивает эквализацию (выравнивание) гистограммы [2]. Пусть уровни интенсивности изображения находятся в интервале *I*= [0, $i_{\max}$] и пусть $p_x(i)$ – непрерывная плотность распределения этих уровней. Определим преобразование *T(i)* в виде интегрального (кумулятивного) распределения

$$s = T(i) = \int_0^i p_x(w)dw .$$

T(i) отображает интервал *I* в единичный интервал таким образом, что *T(0)*=0 и *T(* $i_{\max}$ *)*=1. При этом плотность распределения уровней яркости ***s*** будет равномерной, т.е. преобразование *T*(*i*) порождает изображение уровни яркости, которого являются равновероятными.

Так как в ЭВМ вычисления выполняются с дискретными величинами, а не с непрерывной плотностью распределения, то эквализация гистограммы выполняется с использованием дискретного варианта преобразования *T(i)* :

$$s_i = i_{\max} T_d(i) = i_{\max} \sum_{w=0}^{i} k_w / (M \cdot N).$$

В IPCV эквализацию гистограммы можно выполнить с помощью функции **imout = imhistequal(im)**.

8.2.5 Линейная фильтрация изображений

Рассмотрим фильтрацию изображений с помощью линейных фильтров с конечной импульсной характеристикой, которая представляется в виде прямоугольного окна размером *M*'x*N*' с весами $h[m', n']$ в позициях (m', n'). Для определения местоположения окна задаются координаты опорной точки *(*M_c *,* N_c*)* внутри окна. В качестве опорной выбирают точку с координатами

$$M_c = Integer\left(\frac{M'+1}{2}\right), \; N_c = Integer\left(\frac{N'+1}{2}\right).$$

Последовательность значений выходного изображения фильтра формируется путем двумерной дискретной свертки входного изображения *x*[*m*,*n*] с функцией окна h[*m'*,*n'*]:

$$y[m,n] = \sum_{m'=1}^{M'} \sum_{n'=1}^{N'} x[m - M_c + m', n - N_c + n'] \cdot h[m', n'] .$$

Оконную функцию часто называют маской. На практике обычно используют квадратные маски небольшого размера, например, 3x3, 5x5 или 7x7. Рассмотрим вычисление свертки изображения с маской *h* размером 3x3:

$$h = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}.$$

Маска накладывается на левый верхний угол изображения и на каждом шаге сдвигается относительно изображения (рисунок 8.2) вправо и вниз. В каждой позиции маски вычисляется сумма произведении значений весов маски и соответствующих отсчетов изображения *x*[*m*,*n*], которые находятся в пределах окна маски:

$$y[m,n] = h_{11}x(m-1,n-1) + h_{12}x(m-1,n) + h_{13}x(m-1,n+1) +$$
$$h_{21}x(m,n-1) + h_{22}x(m,n) + h_{23}x(m,n+1) +$$
$$h_{31}x(m+1,n-1) + h_{32}x(m+1,n) + h_{33}x(m+1,n+1)$$

Полученная сумма произведений назначается опорной точке маски.

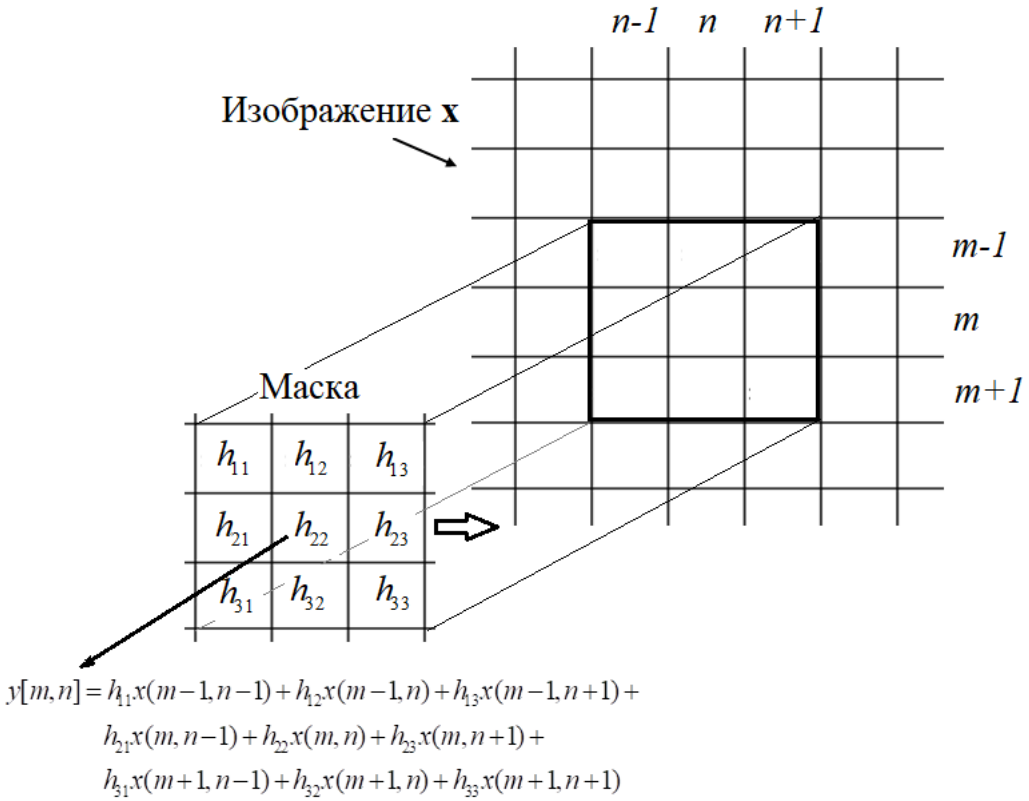

Рисунок 8.2— Вычисление двумерной свертки

Для реализации 2D свертки в IPCV используют функцию **imout=filter2(im,h)**, где: **h** – маска фильтра (импульсная характеристика) типа **double**; **im** – входное изображение типов **int8, uint8, int16, uint16, int32, double**; **imout** – выходное изображение типа **double**. Отсчеты входного изображения, выходящие за границы изображения, полагаются равными ближайшим отсчетам входного изображения. Размерность выходного изображения равна размерности входного изображения. Функция может обрабатывать многоканальные изображения. В этом случае маска **h** применяется отдельно к каждому каналу.

При линейной фильтрации с помощью оконных функций применяют два вида фильтров [1, 2]:

1. Фильтры, у которых сумма коэффициентов оконной функции *SK* больше нуля. Такие фильтры являются усредняющими (фильтрами нижних частот), поэтому их применяют для подавления высокочастотных составляющих изображений. Примеры оконных функций фильтров нижних частот:

$$h1 = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \quad h2 = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, \quad h3 = \frac{1}{16}\begin{bmatrix} 2 & 1 & 2 \\ 1 & 4 & 1 \\ 2 & 1 & 2 \end{bmatrix}.$$

Нормирующий коэффициент этих оконных функций выбирается таким образом, чтобы их применение не вызывало смещение средней яркости обработанного изображения. Отсчеты оконной функцией *h2* аппроксимируют двумерный гауссиан. Соответствующий фильтр называют фильтром Гаусса размером 3x3. Фильтры Гаусса больших размеров могут быть получены прямым вычислением значений двумерной экспоненты $h(x,y) = c\exp(-(x^2+y^2)/\sigma^2)$, где σ – среднеквадратичное отклонение, определяющее ширину фильтра; c – масштабный множитель.

2. Фильтры, у которых сумма коэффициентов оконной функции *SK*=0. Такие фильтры применяются для выделения верхних пространственных частот (обнаружение краёв, выделение контуров и др.).

Иной способ вычисления 2D свертки основан на том, что свёртке 2-х сигналов в пространственной области соответствует произведение их спектров в частотной области. Для реализации этого способа необходимо вычислить двумерные дискретные преобразования Фурье (ДДПФ) от исходного изображения и импульсной характеристики (окна) фильтра, перемножить полученные спектры, а затем выполнить обратное двумерное дискретное преобразование Фурье (ОДДПФ) и получить обработанное изображение. Такой способ линейной фильтрации также называют **быстрой сверткой**, если для вычисления ДДПФ и ОДДПФ применяется алгоритм двумерного БПФ (**fft2**) Ниже приведен фрагмент кода, обеспечивающий вычисление быстрой двумерной свертки с использованием фильтра с окном вида *h*1:

```
im = imread(fullpath(getIPCVpath() + "/images/measure_gray.jpg"));
// определяем размеры входного изображения im
im_size=size(im);
//определяем окно фильтра
h1=[1 1 1; 1 1 1; 1 1 1]./9
// дополняем нулями окно фильтра до размера im и находим его
// частотную характеристику (ЧХ) с помощью  функции fft2pad
Hw=fft2pad(h1,im_size(1),im_size(2));
// находим ДДПФ (спектр) входного изображения
Sp_im = fft2(im2double(im));
// выполняем фильтрацию в частотной области
// путем перемножения спектра входного изображения и ЧХ фильтра
Sp_im_out = Sp_im.*Hw;
// выполняем ОДДПФ и получаем обработанное изображение
im_out = real(ifft(Sp_im_out));
imshow(im_out);
```

8.2.6 Обнаружение перепадов яркости и ребер

Для обнаружения на изображениях однородных областей, ребер, углов, пятен и др. необходимо анализировать профиль интенсивности изображения. Однородные области характеризуются малой вариацией интенсивности. Ребра характеризуются большими изменениями интенсивности в одном направлении и

малыми в ортогональных направлениях. Углы соответствую профилю интенсивности с большими вариациями интенсивности в 2=х ортогональных направлениях. Для обнаружения указанных элементов изображений используют **дифференциальные операторы**. Сумма элементов оконной функции *h* таких операторов равна нулю, что свидетельствует о том, что в зонах изображения с постоянной интенсивностью они дают нулевой отклик. Однако на границах участков изображения с разной интенсивностью отклик таких операторов не равен нулю, что собственно и позволяет им обнаруживать указанные выше элементы [1].

Дифференциальные операторы выполняют оценку градиента (приращения) функции яркости *f*(*x*,*y*) изображения

$$\overrightarrow{\nabla f}(x,y) = \frac{\partial f(x,y)}{\partial x}\vec{i} + \frac{\partial f(x,y)}{\partial y}\vec{j},$$

где *x*,*y* – непрерывные пространственные координаты изображения. Градиент представляет собой вектор длина и угол направленности которого определяются как:

$$S = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}, \quad \theta = arctg(\frac{\partial f / \partial y}{\partial f / \partial x}).$$

На практике при обработке цифровых изображений применяют различные аппроксимации абсолютного значения градиента:

$$S = \sqrt{S_m^2 + S_n^2}, \qquad S = |S_m| + |S_n|, \qquad S = \max\left(|S_m|, |S_n|\right),$$

где S_m и S_n – приращения яркости в вертикальном направлении (по строкам *m*) и горизонтальном направлении (столбцам *n*) изображения **x:**

$$S_m = x[m+1,n] - x[m,n]; \; S_n = x[m,n+1] - x[m,n].$$

Указанные приращения S_m и S_n могут быть определены с помощью окон h_m и h_n, соответственно

$$h_m = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \qquad h_n = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

Имеются различные градиентные операторы для обнаружения перепадов яркости и границ. Известны операторы Превит, Собела, Кирша, Робертса и др. [1]. Например, оконные функции операторов Собела определяются следующим образом:

$$h_m = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \qquad h_n = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}.$$

Иная возможность обнаружения границ, связана с их представлением в виде множества точек перегиба функции яркости. Такие точки могут быть

найдены путем нахождения нулей лапласиана, который представляет собой сумму вторых частных производных функции яркости изображения:

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} = 0.$$

Дискретный вариант лапласиана вычисляется по приближенной формуле

$$S_L(m,n) = \{x[m,n+1] + x[m+1,n] + x[m-1,n] + x[m,n-1]\} - 4x[m,n] \ .$$

Вычисления лапласиана по указанной формуле можно выполнить с помощью маски h_{L1}. Также вычисления лапласиана выполняют с помощью иных масок, например, h_{L2} или h_{L3}:

$$h_{L1} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad h_{L2} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \quad h_{L3} = \begin{bmatrix} 1 & 2 & 1 \\ 2 & -12 & 2 \\ 1 & 2 & 1 \end{bmatrix}.$$

Так как вычисление этого оператора базируется на вторых производных, то он чувствителен к наличию шумов изображения. Поэтому обычно лапласиан комбинируют с усредняющими фильтрами, например, с фильтром Гаусса. Если G – фильтр Гаусса, а L – фильтр Лапласа, то вычисление 2-х сверток $L*(G*x)=G*(L*x)$, очевидно, можно реализовать в виде одной свертки с маской $L*G$ (обычно большего размера). Такой фильтр называют называется LOG-фильтром (Laplacian of the Gaussian). Кроме этого, отклик LOG-фильтра может быть аппроксимирован разностью откликов узкого и широкого фильтров Гаусса. Такой фильтр называют DOG-фильтром (Difference of Gaussians).

Для построения границ в IPCV имеется функция **E = edge(im, method, thresh, direction)**, где: **im** – исходное одноканальное изображение; **method** – вид градиентного оператора, в качестве значений может быть: **'sobel'** (по умолчанию), **'prewitt', 'log', 'fftderiv'** или **'canny'**; **thresh** – устанавливает уровень порога от 0 до 1, по умолчанию 0,2; **direction** – определяет направление градиента, может быть **'horizontal', 'vertical'** или **'both'** (по умолчанию); **E** – черно-белое изображение границ, которое представляется булевой матрицей и имеет тот же размер, что и **im**. Если **thresh<0** , то **E** является изображением градиентов в тонах серого цвета.

Рассмотрим синтаксис вызова функции для каждого поддерживаемого метода:

E=edge(im, 'sobel', thresh, direction) – обнаруживает края в **im** , используя оценку градиента Собела;

E=edge(im, 'prewitt', thresh, direction) – обнаруживает края в **im** , используя оценку градиента Превит;

E=edge(im, 'log', thresh, sigma) – обнаруживает края в **im** , используя лапласиан гауссиана, **sigma** — стандартное отклонение LoG-фильтра размером nxn, где n = ceil(sigma*3)*2+1, по умолчанию равно 2;

E=edge(im, 'fftderiv', thresh, direction, sigma) – обнаруживает края в **im** , используя метод градиента FFT, sigma по умолчанию равно 1.0;

E=edge(im, 'canny', thresh, sigma) – обнаруживает края в **im** , используя метод Канни. Параметры: **sigma** — апертура оператора Канни, которая равна 1, 3, 5 или 7 (по умолчанию sigma =3); **thresh** — вектор, содержащий значения нижнего и верхнего порогов, если **thresh** скаляр, то нижний порог равен 0.4* **thresh**, а верхний — **thresh** (**thresh** не может быть отрицательным, по умолчанию **thresh** =0.2).

Примеры вызова функции:

```
im = imread(fullpath(getIPCVpath() + "/images/baboon.png"));
im = rgb2gray(im);
E = edge(im, 'sobel');
imshow(E);

E = edge(im, 'canny', [0.06, 0.2]);
imshow(E);

E = edge(im, 'prewitt');
imshow(E);
```

8.2.7 Улучшение изображений с помощью нелинейной фильтрации

Линейная фильтрация наиболее часто используется для подавления ограниченного по частоте аддитивного шума. При этом для каждого случая необходимо синтезировать соответствующий фильтр, учитывая частотное распределение шумов [1, 2]. Сглаживание шума обеспечивается низкочастотной фильтрацией. Существенным недостатком такой обработки является размывание границ.

Иной способ подавления шумов на изображении основан на применении медианной фильтрации, в процессе которой сначала производится сортировка по величине пикселов, попавших в окно фильтра, а затем замена значения опорной точки окна величиной интенсивности пиксела, расположенного в середине (медиане) упорядоченного ряда. В связи с этим медианный фильтр не оказывает влияние на изображения со ступенчатым или линейным изменением интенсивности.

Медианная фильтрация является нелинейным методом обработки. Она весьма эффективна для обработки изображений, искаженными помехами типа «соль и перец», т.е., когда на светлом фоне имеются темные точки («перец») и на темном фоне светлые точки «соль»). Медианные фильтры часто применяются итеративно, причем обработка повторяется до тех пор, пока на профильтрованном изображении не прекратятся изменения.

В модуле IPCV медианная фильтрация реализуется функцией **immedian(im,sz)**, где **im** – входное изображение, **sz** – размер окна.

8.2.7 Проектирование двумерных фильтров

Модуль IPCV содержит несколько полезных функций для построения двумерных фильтров.

Функция **mkfftfilter** обеспечивает построение фильтра с заданной амплитудно-частотной характеристикой (АЧХ). Синтаксис вызова функции:

Hw= mkfftfilter(im,name,rc1,rc2),

где im – исходное изображение, name – строка, представляющая название фильтра (может быть ‘binary', 'butterworth1', 'butterworth2’, 'exp’, 'gauss’ или 'trapeze’), rc1 – частота среза фильтра в диапазоне от 0 до1, rc2 – частота задерживания в диапазоне от 0 до 1, H – матрица, представляющая АЧХ фильтра со значениями от 0 до 1, которую затем можно применить для обработки спектра изображения.

Эта функция позволяет построить АЧХ некоторых широко используемых фильтров с целью её дальнейшего применения для обработки спектра изображения. Двумерное преобразование Фурье дает информацию о пространственных частотах, присутствующих в спектре изображения. По спектру можно восстановить исходное изображение. Путем перемножения спектра изображения и АЧХ фильтра можно внести в спектр заданные изменения и затем восстановить изображение с исключенными спектральными составляющими. АЧХ фильтров, возвращаемые функцией **mkfftfilter** , определены на круговой опорной области и могут изменять только амплитуду частотных составляющих спектра (на фазу не действуют).

Функция **mkfftfilter** возвращает АЧХ следующих фильтров нижних частот:

‘binary’ – идеальная прямоугольная АЧХ ФНЧ с частотой среза **rc1**;

‘butterworth1', 'butterworth2’, ‘butterworth3’ – АЧХ ФНЧ Баттерворта первого, второго и третьего порядков с частотой среза **rc1**;

‘exp’, 'gauss’ – ФНЧ с АЧХ в виде экспоненциальной функции соответственно первой и второй степени с частотой среза **rc1**;

'trapeze’ – ФНЧ с АЧХ трапециевидной формы с частотой среза **rc1** и частотой задерживания **rc2.**

Функция **fspecial** обеспечивает построение импульсных характеристик некоторых широко используемых 2D фильтров. Синтаксис вызова функции

h = fspecial(type, op1, op2),

где **type** – строка, задающая тип фильтра (может быть **'sobel', 'prewitt', 'gaussion', 'laplacian', 'log', 'average', 'unsharp', 'motion'**), **op1** – первый параметр фильтра, **op2** – второй параметр фильтра, **h** – возвращаемая импульсная характеристика фильтра, которая имеет тип **double.** Если аргументы не заданы, то **fspecial** использует значения по умолчанию. Рассмотрим синтаксис вызова функции для каждого поддерживаемого типа фильтра:

h = fspecial('sobel') :

Возвращает горизонтальный фильтр Собела 3x3. Если вам нужен вертикальный фильтр Собела используйте транспонирование матрицы **h**;

h = fspecial('prewitt') :

Возвращает горизонтальный фильтр Превит 3x3. Если вам нужен вертикальный фильтр Превит используйте транспонирование матрицы **h**;

h = fspecial('gaussian', hsize, sigma) :

Возвращает гауссовский фильтр нижних частот. Размер возвращаемого фильтра определяется параметром **hsize. hsize** может быть вектором 1x2, который указывает число строк и столбцов **h**. Если **hsize** скаляр, то **h** квадратная матрица, **hsize** по умолчанию равно [3, 3]; значение **sigma** по умолчанию равно 0,5.

h = fspecial('laplacian', alpha) :

Возвращает фильтр Лапласа 3x3. Возвращаемые значения в окне фильтра **h=[alpha, 1-alpha, alpha; 1-alpha, -4, 1-alpha; alpha, 1-alpha, alpha]/(alpha+1).** Значение alpha по умолчанию равно 0,2.

h = fspecial('log', hsize, sigma) :

Возвращает лапласиан гауссовского фильтра. Размер возвращаемого фильтра определяется параметром hsize. hsize может быть вектором 1x2, который указывает число строк и столбцов **h**. Если hsize скаляр, то **h** квадратная матрица, **hsize** по умолчанию равно [5, 5]; значение **sigma** по умолчанию равно 0,5.

h = fspecial('average',hsize) :

Возвращает усредняющий фильтр. Размер возвращаемого фильтра определяется параметром **hsize. hsize** может быть вектором 1x2, который указывает число строк и столбцов **h**. Если **hsize** скаляр, то **h** квадратная матрица, **hsize** по умолчанию равно [3, 3].

h = fspecial('unsharp', alpha) :

Возвращает фильтр улучшения резкости размером 3x3. Значение **alpha** должно быть в диапазоне [0, 1], по умолчанию равно 0,2;

h = fspecial('motion', length, angle) :

Возвращает фильтр размытия движения с заданной длиной **length** и углом **angle.**

Пример использования функции

```
imin = imread(fullpath(getIPCVpath() + "/images/baboon.png"));
// вычисление оконной функции оператора Собела
h = fspecial('sobel');
// двумерная свертка окна с изображением
imout = filter2(imin, h);
imshow(imout);
```

При моделировании процесса обработки изображении с использованием указанных выше фильтров часто используют функцию **imnoise,** с помощью которой добавляют шум к изображению. Синтаксис вызова функции **imn = imnoise(im, type [,parameters])**, где: **im** – входное изображение; **type** – строка, имеющая одно из следующих значений: **'salt & pepper'** – шум типа «соль и перец»; **'speckle'** – мультипликативный шум; **'gaussian'** – гауссовский белый/аддитивный шум; **'localvar'** – дисперсия значений заданных пикселей (гауссовский шум с нулевым средним); **parameters** – последовательность параметров для управления распределением шума в зависимости от выбранного типа, если не указано, используются значения по умолчанию (см. ниже). **imn** – зашумленное изображение, которое имеет

тот же размер и тип, что и входное изображение **im** . По умолчанию считается, что "1" соответствует максимальному значению интенсивности изображения, а "0" - минимальному. Если входное изображение **im** является целочисленным изображением, оно сначала будет преобразовано в **double** с помощью функции **im2double**. Перед возвратом результата изображение будет преобразовано в тот же тип, что и входное изображение. Элементы в выходной матрице **imn**, которые превышают диапазон типа **integer** или **double**, будут усечены.

Особенности вызова функции для разных типов шумов:

imn = imnoise(im,'salt & pepper', d) – добавляет шум типа “соль и перец”, где **d** — плотность шума (вероятность замены исходного пикселя шумом), по умолчанию **d**=0,05;

imn = imnoise(im,'gaussian', m, v) – добавляет гауссовский аддитивный шум со средним значением **m** и дисперсией **v** (по умолчанию: **m**=0 и **v**=0,01);

im = imnoise(im,'localvar',V) – добавляет аддитивный гауссовский шум с нулевым средним значением, где дисперсия пикселя **im**(i,j) равна **V**(i,j).

imn = imnoise(im,'localvar', intensity, V) – добавляет аддитивный гауссовский шум с нулевым средним значением, а локальная дисперсия шума **var** является функцией значений интенсивности изображения **im** и определяется вызовом равна **matrix(interp1(intensity(:),V(:),im(:)), size(im))**;

imn = imnoise(im,'speckle',v) – добавляет мультипликативный шум, используя выражение **imn = im + noise*im**, где шум имеет равномерное распределение со средним значением 0 и дисперсией **v** (по умолчанию: **v**=0.04).

8.3 Варианты заданий

Варианты задании приведены в таблице 8.1.

Таблица 8.1— Варианты заданий

№ варианта	Размер фильтра для **fspecial**	Тип фильтра для **mkfftfilte**r	Дифференциальный оператор	Изображение
1	3x3	binary	sobel log	rice.png football.jpg
2	5x5	butterworth1	prewitt laplacian	pout.tif hands1.jpg
3	7x7	butterworth2	log prewitt	toysflash.png bag.png
4	3x3	butterworth3	unsharp sobel	office_2.jpg cell.tif
5	5x5	exp	sobel log	mandi.tif coins.png
6	7x7	gauss	prewitt laplacian	liftingbody.png eight.tif
7	3x3	trapeze	log prewitt	cameraman.tif gantrycrane.png
8	5x5	binary	unsharp sobel	rice.png football.jpg

9	7x7	butterworth1	sobel log	pout.tif hands1.jpg
10	3x3	butterworth2	prewitt laplacian	toysflash.png bag.png
11	5x5	butterworth3	log prewitt	office_2.jpg cell.tif
12	7x7	exp	unsharp sobel	mandi.tif coins.png
13	3x3	gauss	sobel log	liftingbody.png eight.tif
14	5x5	trapeze	prewitt laplacian	cameraman.tif gantrycrane.png
15	7x7	binary	log prewitt	rice.png football.jpg
16	3x3	butterworth1	unsharp sobel	pout.tif hands1.jpg
17	5x5	butterworth2	Sobel log	toysflash.png bag.png
18	7x7	butterworth3	prewitt laplacian	office_2.jpg cell.tif
19	3x3	exp	log	mandi.tif coins.png
20	5x5	gauss	unsharp	liftingbody.png eight.tif

8.4 Порядок выполнения лабораторной работы

8.4.1. Загрузить заданное вариантом изображение, используя функцию **imread**. Отобразить исходное изображение.

8.4.2. Определить статистические характеристики исходного изображения (среднее, среднеквадратическое отклонение), построить гистограмму распределения интенсивности, выполнить улучшение контрастности методом линейного растяжения и методом эквализации гистограммы. Отобразить изображения с улучшенной контрастностью и их гистограммы.

8.4.3 Вычислить и отобразить 2D спектр исходного изображения и 2D спектр контрастного изображения, полученного путем эквализации гистограммы. При этом для отображения спектров используйте логарифмическое масштабирование (**log10(abs(Sp)+1**, где **Sp** – 2D спектр).

8.4.4. Получить два изображения. Одно из них путем добавления к исходному изображению гауссовского шума, а другое путем добавления мультипликативного шума с помощью функции **imnoise**. Вычислить и отобразить 2D спектры полученных изображений с шумом.

8.4.5. Вычислить с помощью функции **mkfftfilter** АЧХ фильтра, заданного вариантом, отобразить АЧХ. Реализовать быструю свертку с использованием вычисленной АЧХ и выполнить подавление высокочастотных компонент спектра для 2-х изображений с шумами (полученными в п.8.4.4). Отобразить обрабо-

танные изображения. Выполнить подавление низкочастотных компонент спектра для 2-х изображений с шумами (полученными в п.8.4.4), для этого реализовать быструю свертку с использованием АЧХ, равной **1-Hw**, где **Hw** – АЧХ, полученная с помощью функции **mkfftfilter.**

8.4.6. Вычислить импульсные характеристики **h** гауссова (**'gaussian'**) и усредняющего фильтров (**'average'**) с помощью функции **fspecial.** Вычислить и отобразить АЧХ указанных фильтров в 3D, используя фрагмент кода:

```
//дополняем импульсную х-ку h нулями и вычисляем её ДДПФ
// переставляем квадранты спектра местами  и получаем АЧХ Hw
Hw = fftshift(fft2pad(h,256,256));
// визуализируем АЧХ в 3D на сетке 100x100
imsurf(abs(Hw),100);
title('АЧХ фильтра');
```

Выполнить с помощью функции **imfilter** фильтрацию изображения, зашумленного мультипликативным шумом, отдельно гауссовым фильтром и отдельно усредняющим фильтром. Отобразить полученные изображения.

8.4.7. Добавить к исходному изображению шум типа «соль и перец». Отобразить полученное изображение. Обработать изображение помощью медианного фильтра, отобразить результат обработки.

8.4.8. Загрузить и отобразить второе изображение, заданное вариантом. Построить с помощью функции **edge** границы и контуры, имеющиеся на изображении, с использованием 2-х заданных вариантом дифференциальных операторов.

8.5. Содержание отчета

Цель работы, краткие теоретические сведения, текст программы с комментариями, все построенные гистограммы, изображения с улучшенной контрастностью, 2D спектры исходного изображения и контрастированного изображения, 2D спектры зашумленных изображений, АЧХ синтезированных фильтров в частотной области и результаты фильтрации с помощью быстрой свертки, АЧХ фильтров в 3D и соответствующие результаты обработки изображений, результаты медианной фильтрации, окна дифференциальных операторов и результаты обнаружения границ и контуров на заданном изображении.

8.6. Контрольные вопросы

8.6.1. Какие существуют способы представления изображений в IPCV?
8.6.2. Объясните индексное представление изображений.
8.6.3. Объясните полутоновое представление изображений.
8.6.4. Объясните бинарное представление изображений.
8.6.5. Объясните RGB-представление изображений.
8.6.6. Как оценить среднее значение и среднеквадратическое отклонение интенсивности изображения?

8.6.7. Что такое контрастность и как можно её повысить?
8.6.8. Запишите уравнение двумерной свертки и объясните алгоритм её вычисления.
8.6.7. Запишите формулу двумерного дискретного преобразования Фурье. Объясните повторяемость двумерного спектра. Где на 2D спектре располагаются нижние и верхние частоты?
8.6.8. Что такое “быстрая свертка”. Сформулируйте алгоритм быстрой свертки.
8.6.8. Какие функции имеются в IPCV для проектирования фильтров в частотной области? Что они вычисляют?
8.6.9. Какие функции имеются в IPCV для проектирования фильтров в пространственной области? Что они вычисляют?
8.6.10. Объясните алгоритм медианной фильтрации?
8.6.11. Объясните принципы обнаружения перепадов яркости и границ. Что такое дифференциальные операторы? Приведите примеры дифференциальных окон?
8.6.12. Как определяется лапласиан? Приведите примеры окон Лапласа. Как определяется LoG фильтр, DoG фильтр.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Бондарев, В. Н. Цифровая обработка сигналов: методы и средства : учебное пособие для студентов вузов, обучающихся по направлению 0804 "Компьютерные науки" / В. Н. Бондарев, Г. Трестер, В. С. Чернега. - 2-е издание. - Харьков : Конус, 2001. - 397 с. : ил ; 21 см. - Библиография: с. 391-393. - ISBN 966-7636-14-3 (в пер.) - Текст : непосредственный.

2. Гонсалес, Р. Цифровая обработка изображений в среде MATLAB : перерод с английского / Р. Гонсалес, Р. Вудс, С. Эддинс. - Москва : Техносфера, 2006. - 616 с. : ил. + 1 эл. опт. диск (CD-ROM). - (Мир цифровой обработки ; № XI-04). - Загл. на корешке : Мир цифровой обработки; № XI-04. - Библиография: с. 614-615. - ISBN 5-94836092-X. - ISBN 0-13-008519-7. - Текст : непосредственный.

3. Масягин, В. Б. Математическое моделирование и информационные технологии при проектировании в среде Scilab : учебное пособие / В. Б. Масягин, С. Б. Скобелев, А. С. Серков. — Омск : ОмГТУ, 2022. — 138 с. — URL: https://e.lanbook.com/book/343553 (дата обращения: 02.06.2025). — Режим доступа: для авториз. пользователей. — ISBN 978-5-8149-3412-3. — Текст : электронный.

4. Поршнев С. В. MATLAB 7: основы работы и программирования : учеб. пособие для студ. вузов, обуч. по напр. 654600 "Информатика и вычислительная техника" / С. В. Поршнев. — М. : БИНОМ, 2011. — 320 с. — Текст : непосредственный.

5. Сергиенко А. Б. Цифровая обработка сигналов : учебное пособие для студентов высших учебных заведений, обучающихся по направлению подготовки дипломированных специалистов "Информатика и вычислительная техника" / А. Б. Сергиенко. — 2-е изд. — Москва [и др.] : Питер, 2006. — 750 с. — (Учебник для вузов). — ISBN 5-469-00816-9. — Текст : непосредственный.

Бондарев Владимир Николаевич

ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ В SCILAB

Методические указания

В авторской редакции

Изд. № 110/2025. Объем 5 п. л. Усл. печ. л. 4,65. Уч.-изд. л. 4,9.
Издательский центр ФГАОУ ВО «Севастопольский государственный университет»