Министерство образования и науки российской федерации

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

РАЗЛОЖЕНИЕ ПЕРИОДИЧЕСКИХ ФУНКЦИЙ В РЯД ФУРЬЕ

Методические указания

по выполнению лабораторной работы № 4

по курсу «Математические основы обработки сигналов»

Томск 2012

1. Цель работы

Целью лабораторной работы является углубление знаний по основам теории систем и получение навыков расчета параметров системы на основе ее передаточной функции и их проверка полученных выводов с помощью системы MATLAB.

2. Теоритический минимум

2.1. Ряд Фурье

Всякая периодическая функция времени *x(t)*, которая в пределах периода ее изменения T удовлетворяет условиям Дирихле (см. ниже), может быть представлена в виде разложения по тригонометрическим функциям Фурье.

$$x(t) = \frac{a_0}{2} + \sum_{n=1}^{N} a_n \cdot cos\ n\omega_1 t\ + b_n \cdot \sin n\omega_1 t,$$

где ω_1 – частота основной гармоники (т.е. гармоники, получаемой при $n = 1$), N – число гармонических компонент (имеется ввиду усеченный ряд Фурье).

Коэффициенты *a* и *b* рассчитываются по формулам

$$a_0 = \frac{2}{T}\int_{-T/2}^{T/2} x(t)dt,$$

$$a_n = \frac{2}{T}\int_{-T/2}^{T/2} x(t) \cdot \cos n\omega_1 t\, dt,$$

$$b_n = \frac{2}{T}\int_{-T/2}^{T/2} x(t) \cdot \sin n\omega_1 t\, dt.$$

Кроме того ряд Фурье может быть переписан следующим образом:

$$x(t) = \frac{A_0}{2} + \sum_{n=1}^{N} A_n \cdot cos\ (n\omega_1 t\ +\ \varphi_n),$$

Где A_n – амплитуды гармонических составляющих ряда, φ_n – начальные фазы гармоник. Амплитуда и фаза определяются через коэффициенты разложения.

$$A_n = \sqrt{a_n^2 + b_n^2},$$

$$\varphi_n = arctg\ \frac{b_n}{a_n}.$$

Условия Дирихле формулируются следующим образом.

Функция x(t) на интервале [-T/2; T/2]:

1) должна иметь конечное число максимумов и минимумов;
2) может иметь разрывы непрерывности первого рода при некоторых значениях аргумента $t = t_i$, число которых должно быть конечно;
3) должна иметь конечные (равные или неравные между собой) предельные значения $x(-T/2 + 0)$ и $x(T/2 - 0)$.

2.2. Создание функций в MATLAB

В процессе выполнения работы потребуется неоднократно создавать сигналы, заданные по одному и тому же закону. Для того, чтобы не вводить каждый раз один и тот же код, можно объединить часто используемые действия в *функцию*. *Функция* в программировании – это фрагмент кода, объединенный под общим именем.

В MATLAB для создания функции требуется создать отдельный файл с названием, совпадающим с именем функции, например *signal.m*. В созданном файле записывается «скелет» будущей функции.

```
function s = signal(t, f1, method)
end
```

Здесь:

`function` – ключевое слово, дающее системе понять, что далее следует определение функции;

`signal` имя функции. По этому имени будет происходить вызов из внешней программы;

`s` – формальная переменная (т.е. переменная, используемая для обозначения переменной; при вызове функции она будет заменена на реальную) для обозначения результата выполнения функции, в нашем случае для сохранения в нее сигнала;

`t, f1, method` – параметры, поступающие на вход функции; они также являются формальными переменными и при вызове могут быть заменены любыми другими переменными соответствующего типа;

`t` – вектор времени;

`f1` – частота периодического сигнала или базовая частота для полигармонического сигнала;

`method` – строковая переменная, используемая для указания требуемого типа функции; в данной работе переменная method будет принимать значения `'sin'` для полигармонического сигнала, `'square'` – для прямоугольного сигнала (меандра), `'saw'` – для пилообразного сигнала и `'piece'` – для кусочно-непрерывного сигнала.

Создание различных сигналов на основании значения переменной method реализуем с помощью оператора выбора `switch … case` (см. подробное описание в MATLAB с помощью команды `help switch`). В итоге получаем заготовок функции.

```
function s = signal(t, f1, method)
    switch method
        case 'sine'
           [код для создания полигармонического сигнала с
           базовой частотой f1]
        case 'square'
           s = square(2*pi*f0*t);
        case 'saw'
           [код для создания пилообразного сигнала с частотой
          f1]
        case 'piecewise'
           [код для создания кусочно-непрерывного сигнала с
           частотой f1]
    end
end
```

2.2. Расчет интегралов в MATLAB

Для расчета коэффициентов ряда Фурье также понадобится рассчитывать значения определенных интегралов. В MATLAB это можно сделать с помощью функции `trapz(y,x)`. Данная функция по методу трапеций рассчитывает значение определенного интеграла функции, заданной в виде массива ее значений `y` в точках числовой оси, заданных вектором независимой переменной `x`. При этом пределы интегрирования определяются начальным и конечным значением вектора `x`.

Для лучшего понимания приведем разъяснения метода трапеций (рис. 1).

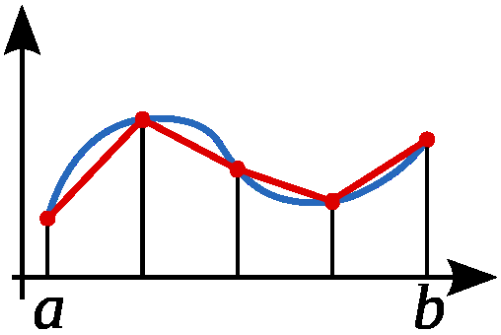

Рис. 1. Аппроксимация площади исходной функции трапециями

Как известно, значение определенного интеграла равно площади подынтегральной функции. В случае цифрового сигнала эта площадь может быть аппроксимирована суммой площадей трапеций, образованных точками самой функции и числовой оси, как показано на рисунке 1. Соответственно, чтобы рассчитать таким методом интеграл, требуется, прежде всего, задать массив отсчетов времени (времени интегрирования), т.к. по сути именно массив времени задает нижний и верхний пределы интегрирования, а также шаг интегрирования.

```
tint = -5:0.001:5;  %в данном случае нижний предел – -5,
                    %верхний – 5, шаг – 0.001
```

Затем требуется в каждый момент времени задать значение функции.

```
y = signal(t,f1,method);
```

Теперь остается лишь применить операцию интегрирования.

```
I = trapz(tint, y);
```

3. Программа работы

В данной работе требуется

1. Создать функцию для задания различных сигналов.
2. Написать файл основной программы для разложения сигнала в ряд Фурье, т.е. расчета коэффициентов разложения.
3. Написать код для восстановления сигнала посредством расчета суммы ряда.

4. Построить графики для сравнения исходного и восстановленного сигнала, а также амплитудный и фазовый спектр (рис. 2).
5. Провести анализ качества разложения при различном количестве гармоник в разложении
6. По результатам работы написать вывод, где отразить полученные данные и знания

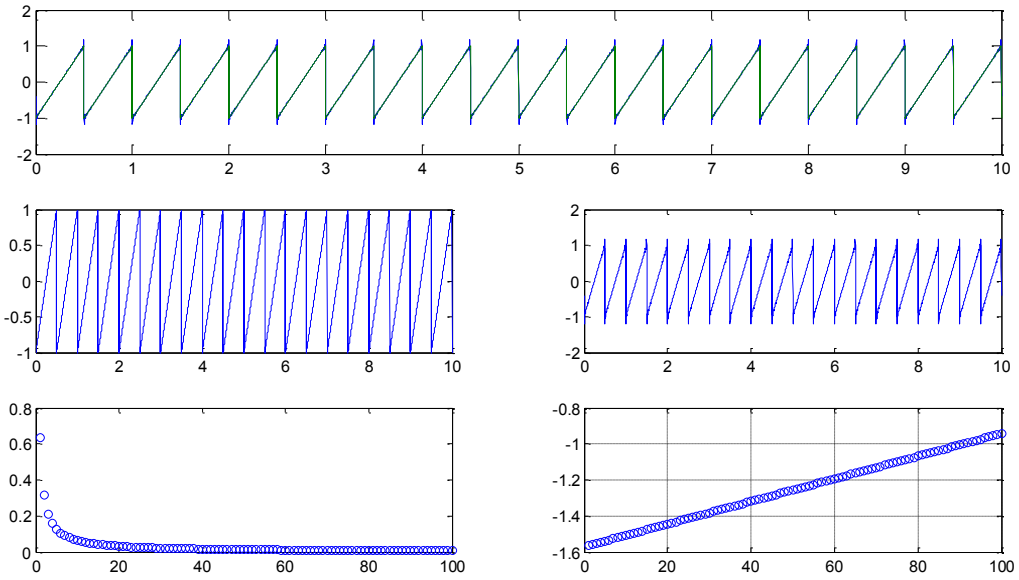

Рис. 2. Пример построения графиков

```
clear; clc;
type = {тип_сигнала};
N = {число_гармоник};
f0 = {базовая_частота};

w0 = {расчет_круговой_частоты};
T = {расчет_периода};
tint = {время_интегрирования};
y = {расчет_сигнала_на_времени_интегрирования};

a0 = {расчет_a0};
for n = 1:N
    a(n) = {расчет_an};
    b(n) = {расчет_bn};
end

t = {время_существования_сигнала};
s = {исходный_сигнал};
x = 0;
for n = 1:N
    x = x + {сумма_косинусной_и_синусной_составляющей};
end
x = x + {постоянная составляющая};
```

```
A = {расчет_амплитуд_гармоник};
phi = {расчет_фаз_гармоник};

{построение графиков};
```