МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)
Кафедра механики, мехатроники и робототехники

УТВЕРЖДАЮ
Проректор по учебной работе
_____О.Г. Локтионова
«10» 02 2022 г.

СЕНСОРНЫЕ СИСТЕМЫ И МЕТОДЫ ОБРАБОТКИ СИГНАЛОВ

Методические указания по выполнению лабораторных работ
для студентов направления 15.03.06 Мехатроника и робототехника

Курск 2022

УДК 62.83

Составители: А.В. Мальчиков

Рецензент
Кандидат технических наук, доцент Е.Н. Политов

Сенсорные системы и методы обработки сигналов: методические указания по выполнения лабораторных работ по дисциплине «Сенсорные системы и методы обработки сигналов» для студентов направления подготовки 15.03.06 Мехатроника и робототехника / Юго-Зап. гос. ун-т; сост.: А.В. Мальчиков. Курск, 2022. 54 с.

Содержатся сведения по вопросам проектирования сенсорных устройств и систем мехатроники и робототехники. Приводятся примеры выполнения лабораторных работ, краткие теоретические положения и контрольные вопросы для защиты.

Предназначены для студентов направления подготовки 15.03.06 «Мехатроника и робототехника всех форм обучения.

Текст печатается в авторской редакции

Подписано в печать 10.02.2022 . Формат 60x84 1\16
Усл.печ.л. 3,0 .Уч.изд.л. 2,8 .Тираж 50 экз. Заказ 735. Бесплатно.
Юго-Западный государственный университет.
305040, г.Курск, ул.50 лет Октября, 94.

Содержание

Лабораторная работа №1. Применение преобразования Фурье дляобработки сигнала акселерометра

Цель работы: *изучение методов реализации преобразования Фурье средствами математического пакета MathCAD на примере задачи обработки сигнала акселерометра*

Аппаратные средства: *математический пакет MathCAD.*

Краткие теоретические сведения

Задача получения и обработки информации с датчиков решается в любой системе с обратной связью. Особенности решения задачи получения информации с датчиков зачастую зависят от типа используемых датчиков и структуры системы. При этом для обработки информации с датчиков могут быть использованы некоторые общие подходы, одним из которых является использование преобразования Фурье для получения информации о частотном спектре сигнала.

Получение информации о частотном спектре сигнала важно для решения ряда задач. Примером такой задачи является фильтрация сигнала, необходимая при произведении измерений в системах, подверженных вибрациям. Так при использовании акселерометра, установленного на борту летательного аппарата важно отделить высокочастотные колебания, вызванными источниками вибрации, находящимися на борту летательного аппарата, колебания, вызываемые взаимодействием корпуса с воздушными потоками, и управляемые корпуса летательного аппарата перемещения (например, крен, тангаж и рыскание). Это особенно актуально для небольших беспилотных летательных аппаратов, где качественная виброизоляция датчиков не всегда возможна.

В рамках данной работы подразумевается знакомство с принципом работы акселерометра. Мы будем рассматривать работу с датчиком в рамках цифровой системы (информация, получаемая с датчика, имеет дискретный характер).

Преобразование Фурье для функции $f(x)$ может быть записано следующим образом:

$$f(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt$$

В рамках данной работы нас интересует возможность применения преобразования Фурье для получения спектра сигнала, полученного с акселерометра. Если мы представим сигнал с датчика в виде некоторой функции $y = F(t)$, то мы можем использовать преобразование Фурье для получения функции частотного спектра в виде некоторой функции $Y = \hat{f}(\omega)$. При этом важно отметить, что функция $y = F(t)$ является дискретной, что оказывает влияние на характер полученной в результате преобразования функции $Y = \hat{f}(\omega)$.

Методика выполнения лабораторной работы

Зададимся некоторой полигармонической функцией $F(t)$, имитирующей сигнал с акселерометра. Запишем аналитическую форму преобразования Фурье (функция $f(\omega)$ в коде ниже).

```
F(t) := 0.1·cos(4·t) + 0.7·sin(6·t) + 0.6·sin(t) + 0.85·sin(12·t) + 0.9·sin(8·t)

f(ω) := ∫_0^100 F(t)·e^(−2·π·t·i·ω) dt

w := 0.01, 0.02.. 10
```

Покажем график функции, имитирующей сигнал с акселерометра (см. рис. 1.1).

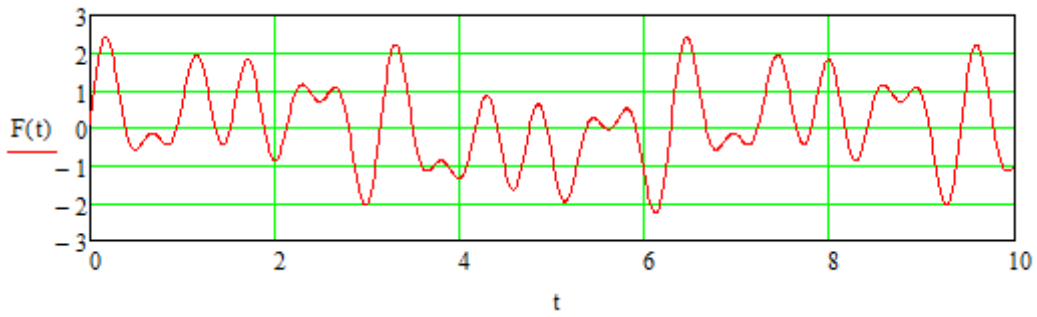


Рис. 1.1 Исходный сигнал с датчика

Используя записанную выше функцию $f(\omega)$ построим спектрограмму сигнала.

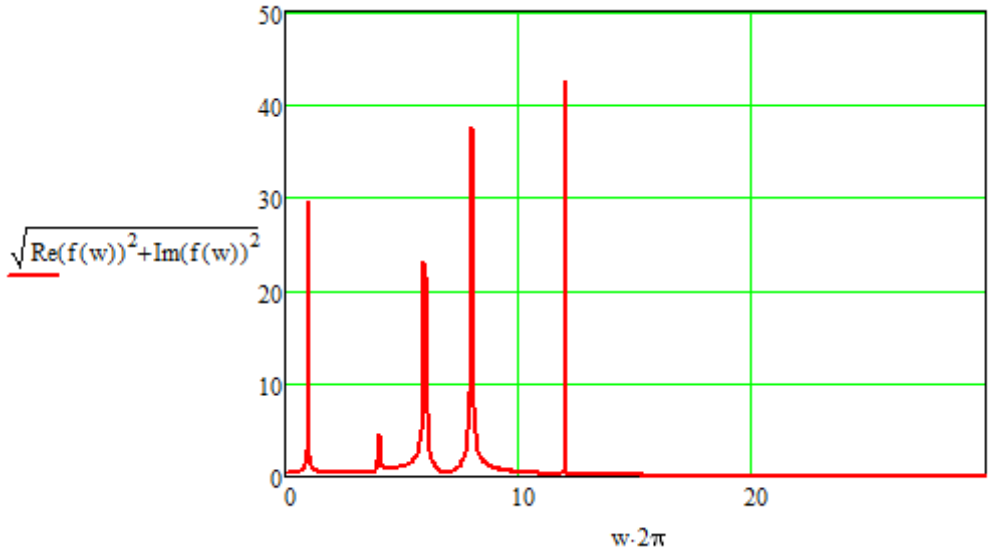


Рис. 1.2 Спектрограмма сигнала с датчика

По оси абсцисс отложены угловые частоты колебаний (в радианах в секунду) для перехода к частотам (в Гц) необходимо убрать масштабирующий коэффициент 2π

Отметим, что функция $f(\omega)$ производит интегрирование на участке от 0 до 100 секунд вместо промежутка (-∞, +∞), который необходимо использовать согласно определению преобразования Фурье. Очевидно, что интегрирование численными методами на промежутке (-∞, +∞) не является рациональным. Вместо этого производим интегрирование на всем рассматриваемом временном промежутке.

Также отметим, что для построения графика мы использовали только мнимую часть полученного результата.

Рассмотрим другой вариант получения спектрограммы. Запишем следующую функцию:

$$f(Time, n) := \frac{1}{Time} \cdot \int_0^{Time} F(t) \cdot \cos\left(\frac{n \cdot \pi \cdot t}{Time}\right) dt$$

$$w := 0, 0.5 .. 1200$$

Отметим, что функция записана таким образом, чтобы участок, на котором будет производиться интегрирование, может

быть задан при вызове функции. Построим спектр используя данную функцию:

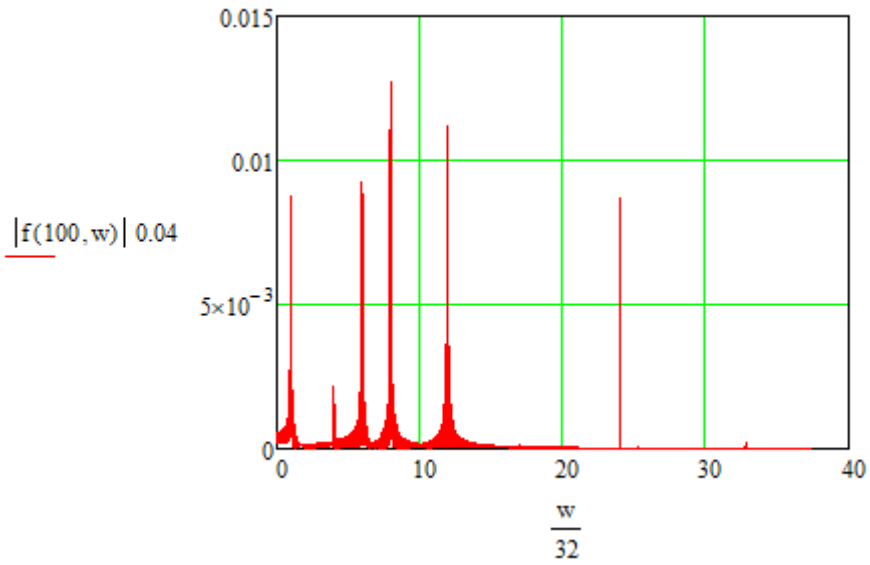


Рис. 1.3 Спектрограмма сигнала с датчика

Отметим, что по оси абсцисс отложены угловые частоты колебаний (в радианах в секунду). Для перехода к такому виду был использован масштабирующий коэффициент. Значение масштабирующего коэффициента меняется, в том числе в зависимости от размеров участка, на котором производится интегрирование.

Обратим внимание, что все описанные выше методы использовали встроенные функции математического пакета MathCAD для реализации преобразования Фурье. Кроме того, в рассмотренных случаях мы работали с непрерывными функциями, а не с дискретным сигналом. Очевидно, что при реализации преобразования Фурье, например, средствами бортового вычислителя беспилотного транспортного средства, робота или другого автономного мобильного объекта, мы не имеем возможности использовать встроенные средства пакета MathCAD. Таким образом нам необходимо рассмотреть программную реализацию преобразования Фурье.

Перед тем, как перейти реализации преобразования Фурье, нам необходимо перевести функцию $F(t)$, имитирующую сигнал с датчика в некоторый дискретный массив данных. Используем код, показанный ниже, для дискретизации функции $F(t)$ с частотой 10

кГц. Используем встроенную в MathCAD функцию fft( ) для получения спектрограммы. Функция fft( ) реализует алгоритм Быстрого преобразования Фурье (англ. Fast Fourier Transform; название функции является сокращением этого термина). Особенностью алгоритма является низкая вычислительная сложность – $0(N \cdot \log(N))$.

$\Delta := 0.00001$

$SinFunc(Count) := \left| \begin{array}{l} \text{for } i \in 0..Count \\ \quad A_i \leftarrow F(i \cdot \Delta) \\ A \end{array} \right.$

$Counter := 2^{21}$

$TestSin := SinFunc(Counter - 1)$

$Four := fft(TestSin)$

$i := 0, 1..100 \qquad length(Four) = 1.049 \times 10^6$

Переменная Counter определяет величину массива данных, сформированного после дискретизации функции $F(t)$. Также эта переменная определяет величину рассматриваемого участка функции. Покажем полученную спектрограмму в виде зависимости.

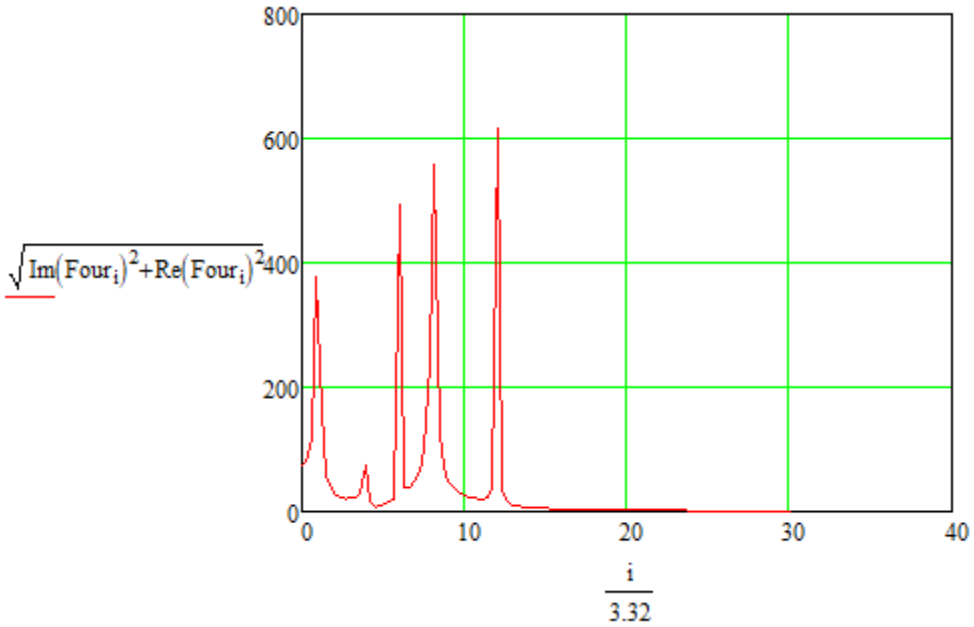
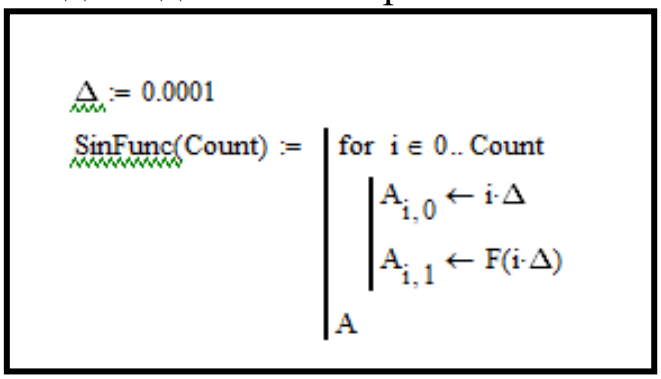


Рис. 1.4 Спектрограмма, полученная с помощью алгоритма быстрого преобразования Фурье

Отметим, что частота дискретизации сигнала и величина рассматриваемого отрезка времени (участка, на котором будет производиться численное интегрирование) существенно влияют на объем вычислений, необходимых для выполнения численного преобразования Фурье. Для снижения вычислительной нагрузки используем дискретизацию с частотой 10 кГц и сгенерируем массив с 2^{19} строк. В данной реализации алгоритма дискретизации каждая строка полученного массива содержит информацию как о значении функции $F(t)$ в данный момент времени, так и о величине t для которой было произведено данное измерение.



Таким образом, структура данных, хранящихся в массиве, имеет следующий вид.

Табл. 1.1 Структура данных массива, полученного в результате дискретизации функции $F(t)$

Номер строки	Данные 1	Данные 2
1	$F(t_0)$	t_0
2	$F(t_1)$	t_1
3	$F(t_2)$	t_2
…	…	…
n	$F(t_n)$	t_n

Данная структура данных удобна в случае, если дискретизация производится с переменным шагом (переменной частотой). Если дискретизация производится с постоянным шагом, можно ограничиться хранением значений функции $F(t)$ в разные моменты времени.

Ниже представлен код функции, реализующейпреобразование Фурье и вызов написанной функции. Численное интегрирование реализовано методом прямоугольников.

```
TestSin := SinFunc(2^18)

Fourier(A) := | ωMAX ← 10
              | Δω ← 0.01
              | Count ← ωMAX / Δω
              | CountTime ← length(A^⟨1⟩) − 1
              | for i ∈ 0.. Count
              |     W ← 0
              |     ω ← i·Δω
              |     for j ∈ 0.. CountTime
              |         t1 ← j·Δ
              |         W ← W + Δ·A_{j,1}·e^(−2·π·t1·i·(ω))
              |     B_{i,0} ← ω
              |     B_{i,1} ← √(Im(W)^2 + Re(W)^2)
              | B

Four := Fourier(TestSin)
```

Рассмотрим полученный частотный спектр.

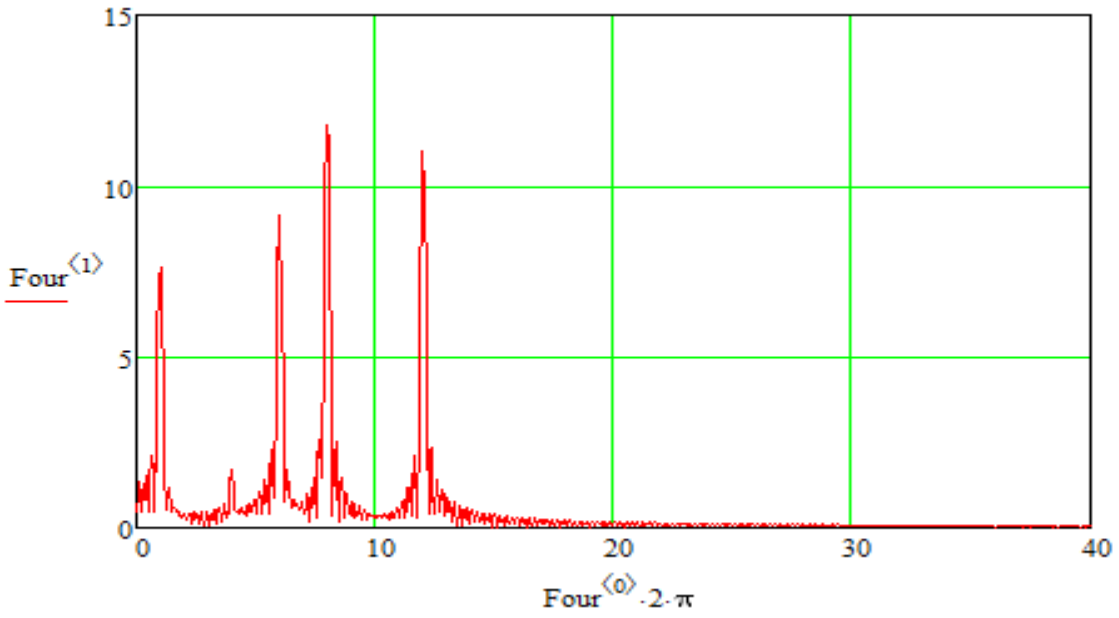
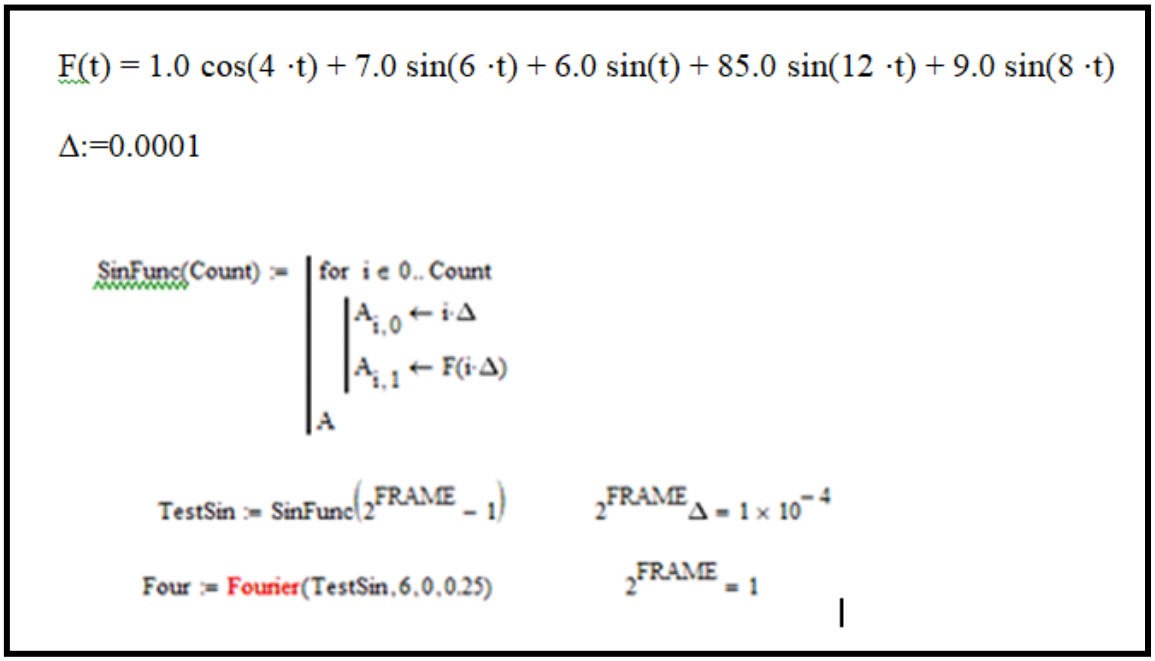


Рис. 1.5 Спектрограмма, полученная с помощью программного преобразования Фурье

При использовании преобразования Фурье для получения частотного спектра дискретного сигнала важно помнить о том, что частота дискретизации и рассматриваемый временной отрезок оказывают значительное влияние на качество полученного результата. Чтобы убедиться в этом проделаем следующий опыт.

Код, представленный ниже, позволит нам оценить характер изменения спектра, полученного преобразованием Фурье, при изменении количества строк массива, имитирующего дискретный сигнал с датчика (это число определяется произведением величины рассматриваемого временного отрезка и частоты дискретизации).



Для того, чтобы использовать встроенный инструмент анимации разместите ниже данного участка кода график, аналогичный показанному на рис. 1.5. После этого зайдите в меню анимации (Инструменты->Анимация->Запись). Настройте параметры анимации, как это показано на рисунке 1.6. После этого выберите область, которую вы хотите снять (рекомендуется выбрать упомянутый выше график) и нажать кнопку «Анимировать».

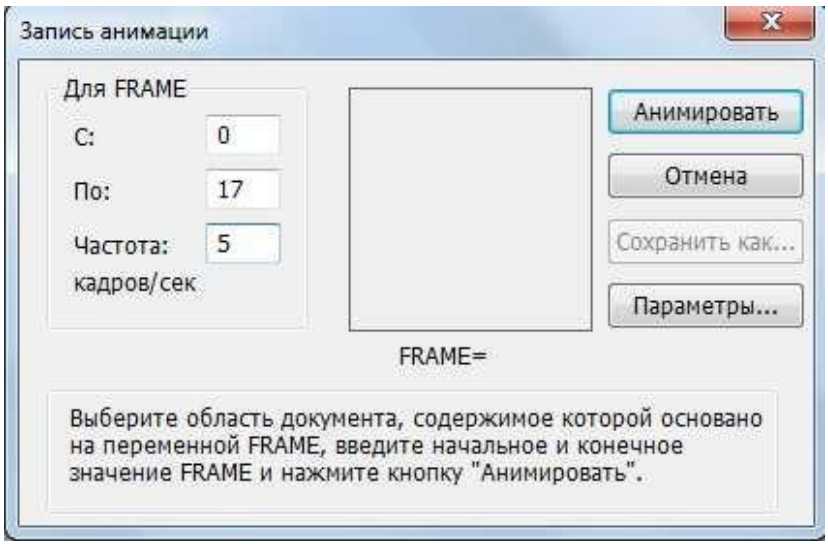
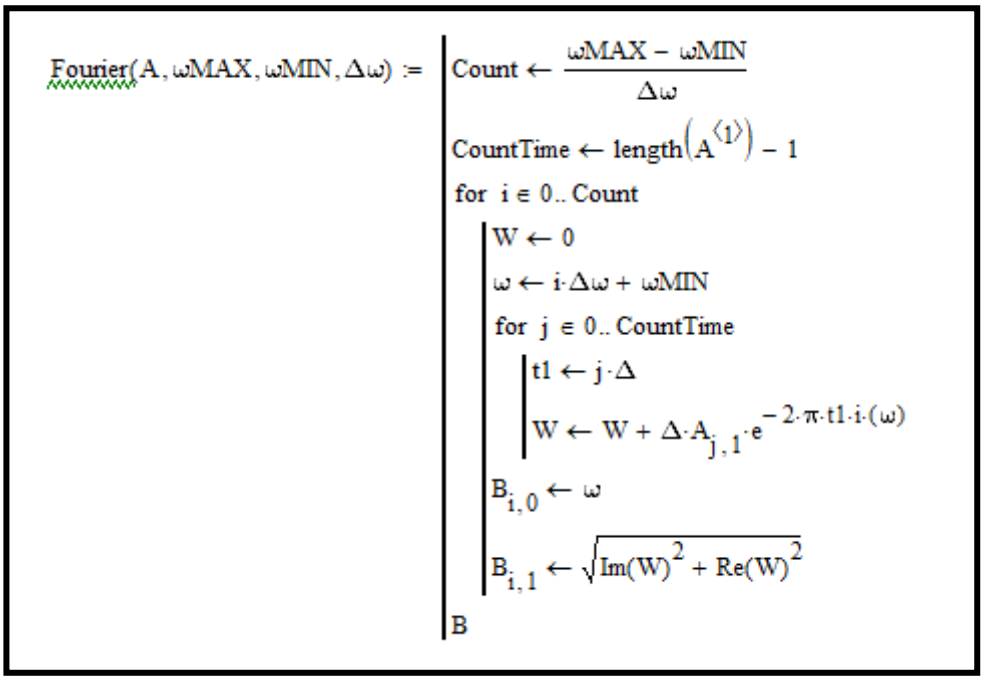


Рис. 1.6 Настройки записи анимации в пакете MathCAD 15

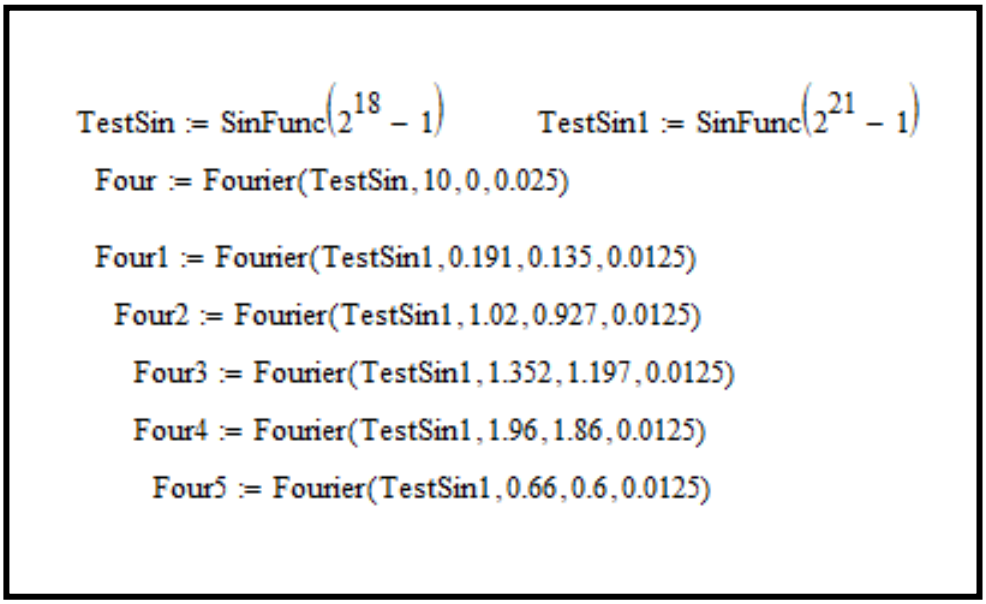
Как уже было отмечено, качество полученного спектра, как и сложность (а как следствие и время) вычислений зависят от количества строк в массиве, имитирующем дискретный сигнал с датчика, и как уже было отмечено, это число определяется частотой дискретизации и рассматриваемым временным промежутком. Для получения более качественного спектра нам необходимо затратить больше времени, что не всегда допустимо, особенно при работе в системах реального времени.

Рассмотрим метод, позволяющий уменьшить временные затраты на построение спектра, сохраняя при этом его высокое качество. Суть метода состоит в том, что мы используем небольшой участок сигнала для предварительного построения спектра. Затем бы строим спектры в узком частотном диапазоне вокруг областей, где на первичном спектре были обнаружены значительные пики. При этом для этих вторичных спектров бы используем значительно больший участок сигнала, что позволяет нам добиться высокого качества. Экономия вычислительных ресурсов достигается за счет того, что высококачественный спектр, требующий большого объема вычислений, строится только для небольшого диапазона частот.

Код функции, реализующей данный метод, представлен ниже.



Фактически данная функция позволяет нам строит частотные спектры для заданного диапазона частот с заданным шагом, что имеет высокую практическую ценность. Ниже показаны примеры вызова написанной функции.



Полученный первичный спектр показан на рисунке 1.7. Вторичные спектры, наложенные на первичный спектр, показаны на рисунке 1.8.

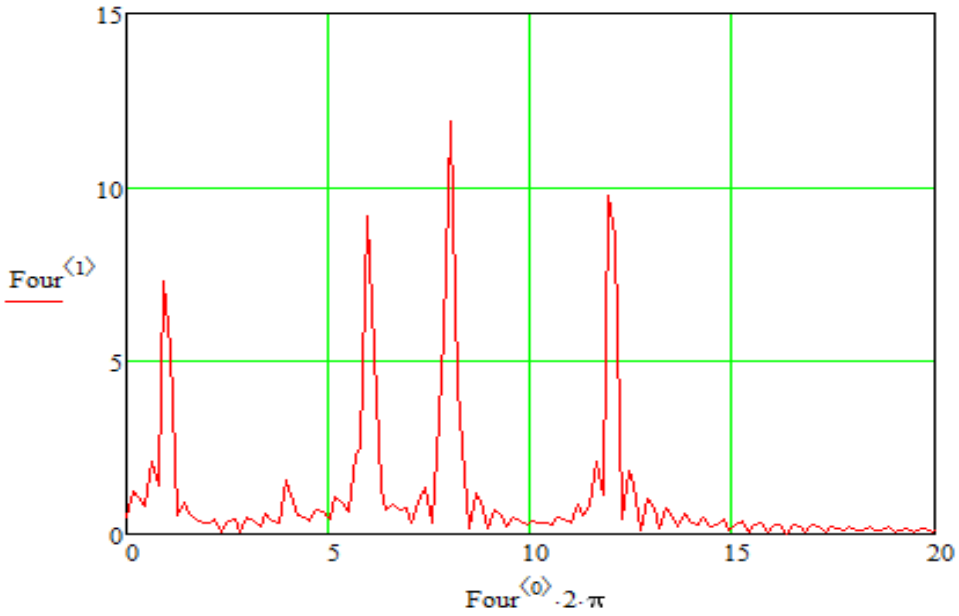


Рис. 1.7 Первичный спектр, полученный с помощью программного преобразования Фурье

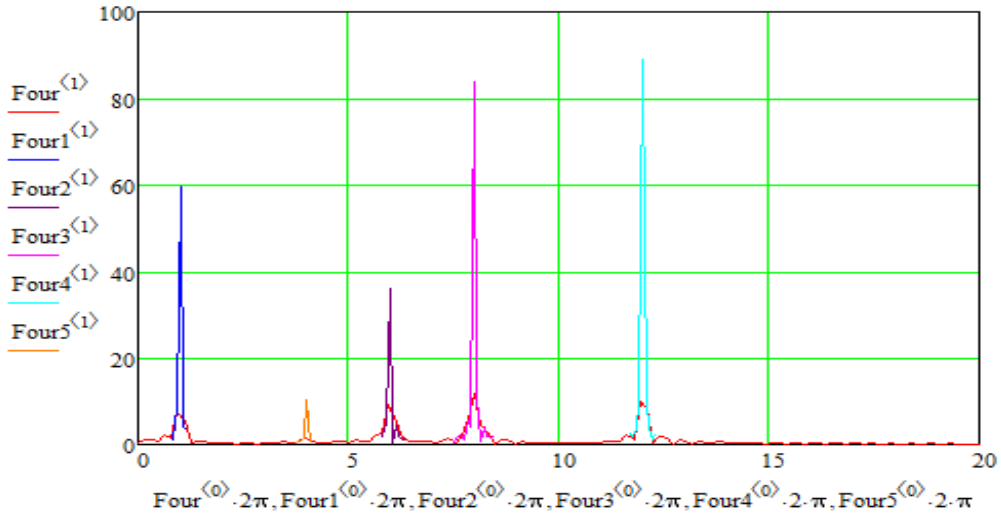


Рис. 1.8 Вторичные спектры, наложенные на первичный спектр

Отметим, что первичный спектр был зашумлен значительно сильнее и имел менее явно выраженные частотные пики, в сравнении со вторичными спектрами.

Задание на выполнение лабораторной работы

Задание на выполнение лабораторной работы состоит в построении спектра сигнала, используя все методы, описанные в

данной работе (включая интегрирование встроенными средствами математического пакета, численное интегрирование, быстрое преобразование Фурье и применение метода построения вторичных спектров в узких частотных диапазонах).

Функцию, имитирующую сигнал с датчика, следует выбирать из табл. 1.2 в соответствии с номером студента в списке группы.

Табл. 1.2 Задания на выполнение лабораторной работы

№	Задание
1	$F(t) = 0.5\sin(2 \cdot t) + 0.3\sin(12 \cdot t) + 0.1\sin(0.5 \cdot t)$
2	$F(t) = 0.2\sin(7 \cdot t) + 0.1\cos(6 \cdot t) + 0.3\cos(3 \cdot t)$
3	$F(t) = 0.35\sin(10 \cdot t) + 0.15\sin(8 \cdot t) + 0.35\sin(6 \cdot t)$
4	$F(t) = 0.15\sin(2 \cdot t) + 0.45\sin(8 \cdot t) + 0.85\sin(12 \cdot t) + 1.05\sin(0.5 \cdot t)$
5	$F(t) = 0.25\sin(13 \cdot t) + 0.35\sin(12 \cdot t) + 0.75\sin(5 \cdot t) + 1.05\sin(4 \cdot t)$
6	$F(t) = 0.45\sin(2 \cdot t) + 0.65\sin(0.5 \cdot t) + 0.05\sin(18 \cdot t) + 0.5\sin(9 \cdot t)$
7	$F(t) = 0.85\sin(7 \cdot t) + 0.9\sin(4 \cdot t) + 0.1\sin(10 \cdot t) + 0.2\sin(12 \cdot t)$
8	$F(t) = 0.6\cos(14 \cdot t) + 0.95\sin(2 \cdot t) + 0.5\sin(t) + 0.7\sin(3 \cdot t) + 0.8\sin(12 \cdot t)$
9	$F(t) = 0.1\cos(4 \cdot t) + 0.7\sin(6 \cdot t) + 0.6\sin(t) + 0.85\sin(12 \cdot t) + 0.9\sin(8 \cdot t)$
10	$F(t) = 0.5\cos(12 \cdot t) + 0.55\sin(17 \cdot t) + 0.65\sin(3 \cdot t) +$ $+ 0.3\sin(15 \cdot t) + 0.45\sin(t)$

Лабораторная работа №2. Оконное преобразование Фурье. Преобразование Габора

Цель работы: *изучить методику использования оконного преобразования Фурье и преобразования Габора для обработки сигналов*

Аппаратные средства: *математический пакет MathCAD.*

Краткие теоретические сведения

Преобразование Фурье позволяет перевести сигнал из временного домена в частотный домен, что может быть использовано для проведения анализа частотного спектра сигнала. В динамических системах зачастую важным оказывается время появления той или иной гармоники, входящей в состав сигнала. Примером такой задачи является детектирование момента возникновения и/или прекращения вибраций корпуса устройства. При этом преобразование Фурье не позволяет оценить изменение спектра сигнала во времени. Для оценки изменения спектра сигнала во времени могут быть использованы методы оконного преобразования Фурье. В англоязычной литературе для обозначения оконного преобразования Фурье используется термин «short-time Fourier transform», термин «windowed Fourier transform» используется значительно реже.

В общем виде выражение, описывающее оконное преобразование Фурье для функции $f(x)$, выглядит следующим образом:

$$\hat{f}(\omega,\tau) = \int_{-\infty}^{\infty} f(t)\cdot\bar{\omega}(t-\tau)\cdot e^{-it\omega}dt$$

где $\bar{\omega}(t-\tau)$ – так называемая оконная функция, $f(t)$– исходный сигнал, а $\hat{f}(\omega,\tau)$ – частотный спектр участка сигнала, попавшего в «окно». Величина τ определяет, для какого момента времени мы строим спектрограмму.

Обратим внимание на то, что существенным отличием оконного преобразования Фурье от обычного преобразования Фурье является наличие функции $\bar{\omega}(t-\tau)$. Задача этой функции состоит в том, чтобы отсечь сигнал, находящийся вне пределов некоторого окна. Существует множество различных функций, используемых в роли оконной функции. Зачастую используются

функции, имеющие область значений в диапазоне от 0 до 1, что позволяет избежать усиления или изменения знака сигнала. В рамках данной работы рассмотрим два варианта реализации оконной функции – прямоугольное окно и гауссову функцию. Оконное преобразование Фурье с использованием гауссовой функции в качестве окна называется преобразованием Габора (названо в честь венгерского физика Денеша Габора, лауреата нобелевской премии по физике 1971 года).

Функция прямоугольного окна выглядит следующим образом:

$$\omega(t) = \begin{cases} 1 \text{ если } t \in [t-a; t+a] \\ 0 \text{ если } t \notin [t-a; t+a] \end{cases}$$

После наложения данной функции весь сигнал, находящийся вне пределов окна, обратится в ноль. Данная функция относительно проста в реализации, что является одним из её основных преимуществ. При этом использование данной оконной функции может внести нежелательные искажения в спектрограмму, связанные с резкими границами окна. При использовании данной оконной функции выражение может быть переписано следующим образом:

$$\hat{f}(\omega, \tau) = \int_{\tau-a}^{\tau+a} f(t) \cdot e^{-it\omega} dt$$

Гауссова функция может быть представлена в следующем виде:

$$\bar{\omega}(t) = \frac{1}{\sigma * \sqrt{2\pi}} \cdot e^{-\frac{t-t_0}{2\sigma^2}}$$

На практике при использовании гауссовой функции предполагается, что существует некоторый диапазон $[t - b; t + b]$ за пределами которого значение этой функции пренебрежимо мало. Таким образом, становится возможным записать выражение для функции следующим образом:

$$\hat{f}(\omega, \tau) = \int_{\tau-b}^{\tau+b} f(t) \frac{1}{\sigma * \sqrt{2\pi}} \cdot e^{\frac{t-\tau}{2\sigma^2}} \cdot e^{-it\omega} dt$$

Так можно заметить, полученная функция является некоторой комбинацией гауссовой и прямоугольной оконных функций.

Методика выполнения лабораторной работы

Зададимся некоторыми полигармоническими функциями $F(t)$ и $F_1(t)$, а также зададим частоту дискретизации Δ.

$$k := 1$$

$$F1(t) := 0.03 \sin(15 \cdot k \cdot t)$$

$$F(t) := 0.02 \sin(2 \cdot k\, t) + 0.01 \sin(10 \cdot k \cdot t) + 0.015 \sin(25 \cdot k \cdot t)$$

$$\Delta := \frac{0.0001}{k}$$

Ниже приведен код, накладывающий гармонический сигнал $F_1(t)$ на сигнал $F(t)$ на промежутке $(T_1; T_2)$.

```
SinFunc(Time, T1, T2, Δ) := | Count ← Time / Δ
                            | for i ∈ 0.. Count
                            |     t ← i · Δ
                            |     A[i,0] ← t
                            |     A[i,1] ← F(t)
                            |     A[i,1] ← A[i,1] + F1(t)  if (t > T1) ∧ (t < T2)
                            | A
```

Ниже показан пример вызова написанной функции:

```
Signal := SinFunc(50, 5, 20, Δ)
```

Полученный в результате вызова функции сигнал показан на рис.2.1

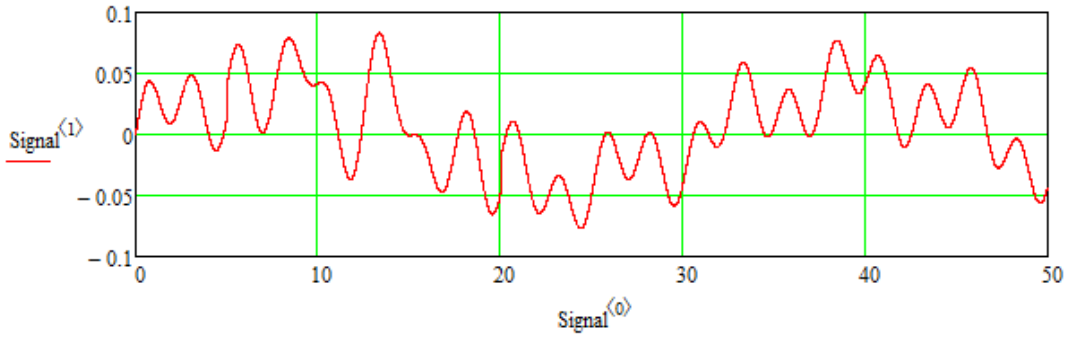


Рис. 2.1 Гармонический сигнал $F(t)$ с наложенным на него сигналом $F_1(t)$ на промежутке $(T_1; T_2)$

Рассмотрим другой способ сложения гармонических сигналов

на заданном временном диапазоне. Для этого мы можем записать следующее выражение

$$F2(t,t0,\alpha) := F(t) + 1.5 \cdot F1(t) \cdot e^{-\pi\alpha\,(t-t0)^2}$$

$$t := 0, 0.001 .. 40$$

Этот метод использует гауссову функцию, в результате чего результирующий сигнал не имеет резких границ в области, где появляется (и исчезает) добавленная гармоника. Результат сложения сигналов данным методом показан на рис. 2.2.

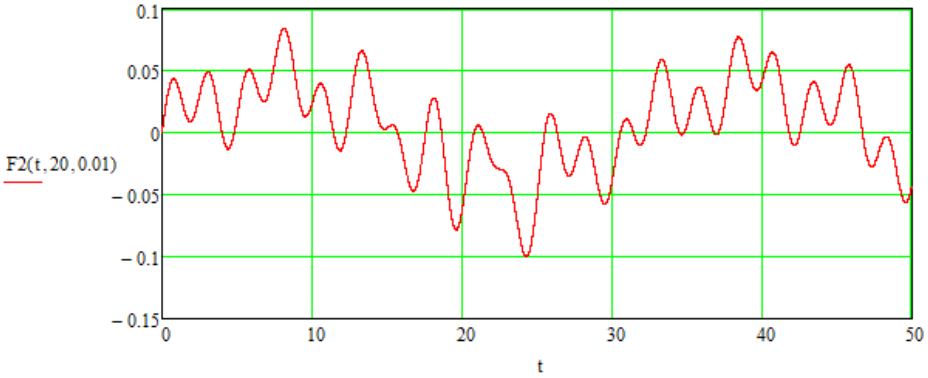


Рис. 2.2 Гармонический сигнал $F(t)$ с наложенным на него сигналом $F_1(t)$ умноженным на гауссову функцию

Запишем аналитическую функцию, реализующую преобразование Фурье для заданного диапазона частот. Подробнее данная функция и особенности её использования были рассмотрены в лабораторной работе «Применение преобразования Фурье для обработки сигнала акселерометра».

$$\begin{aligned}
&\text{Furier}(A, \omega MAX, \omega MIN, \Delta\omega, \Delta) := \\
&\quad \text{Count} \leftarrow \frac{(\omega MAX - \omega MIN)}{\Delta\omega} \\
&\quad \text{CountTime} \leftarrow \text{length}\left(A^{\langle 1 \rangle}\right) - 1 \\
&\quad \text{for } i \in 0..\text{Count} \\
&\quad\quad W \leftarrow 0 \\
&\quad\quad \omega \leftarrow i \cdot \Delta\omega + \omega MIN \\
&\quad\quad \text{for } j \in 0..\text{CountTime} \\
&\quad\quad\quad t1 \leftarrow j \cdot \Delta \\
&\quad\quad\quad W \leftarrow W + \Delta \cdot A_{j,1} \cdot e^{-2\pi \cdot t1 \cdot i \cdot (\omega)} \\
&\quad\quad B_{i,0} \leftarrow \omega \\
&\quad\quad B_{i,1} \leftarrow \sqrt{\text{Im}(W)^2 + \text{Re}(W)^2} \\
&\quad B
\end{aligned}$$

Ниже показан пример вызова данной функции

$$Res := Fourier(Signal, 6 \cdot k, 0, 0.02 \cdot k, \Delta)$$

Результат работы функции представлен на рисунке 3.

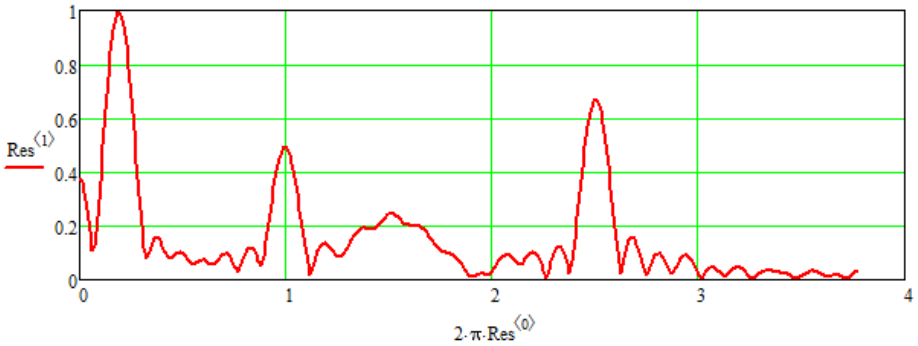


Рис. 2.3 Частотный спектр сигнала

Следует обратить внимание, что хотя мы и получили информацию о частотном спектре сигнала, в результате преобразования мы потеряли информацию о характере изменения сигнала. Так, анализируя полученный график, мы можем сделать вывод о наличии в спектре гармонической составляющей с круговой частотой 15 радиан в секунду, при этом мы не можем оценить момент появления этой составляющей в спектре.

Рассмотрим пример реализации оконного преобразования Фурье с использованием прямоугольной оконной функции.

$$\begin{aligned}
&FurierWindow(A, \omega MAX, \omega MIN, \Delta\omega, T1, T2, \Delta) := \\
&\quad Count \leftarrow \frac{(\omega MAX - \omega MIN)}{\Delta\omega} \\
&\quad \text{for } i \in 0..Count \\
&\quad\quad W \leftarrow 0 \\
&\quad\quad \omega \leftarrow i \cdot \Delta\omega + \omega MIN \\
&\quad\quad \text{for } j \in \frac{T1}{\Delta}..\frac{T2}{\Delta} \\
&\quad\quad\quad t1 \leftarrow j \cdot \Delta \\
&\quad\quad\quad W \leftarrow W + \Delta \cdot A_{j,1} \cdot e^{-2\pi \cdot t1 \cdot i \cdot (\omega)} \\
&\quad\quad B_{i,0} \leftarrow \omega \\
&\quad\quad B_{i,1} \leftarrow \sqrt{Im(W)^2 + Re(W)^2} \\
&\quad B
\end{aligned}$$

Ниже показан пример вызова написанной функции:

```
Time := 0
Res := FurierWindow(Signal, 6 · k, 0, 0.02 · k, 0 + Time, 50 + Time, Δ)
```

Результат оконного преобразования для промежутка от 0 до 5 секунд показан на рисунке 2.4:

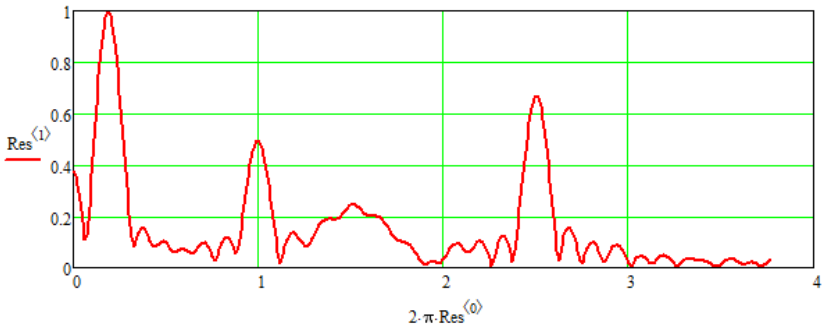


Рис. 2.4 Спектрограмма, полученная с помощью оконного преобразования Фурье с прямоугольной оконной функцией на промежутке от 0 до 5с

Запишем выражение для выполнения преобразования Габора:

$$G(\omega,t,\alpha) := \int_{t-2}^{t+2} e^{-\pi\alpha(\tau-t)^2} \cdot e^{-2\cdot\pi\cdot\tau\cdot i\cdot(\omega)} \cdot F2(\tau,25,0.01)\,d\tau$$

$$gFT(\omega) := \sqrt{Im(G(\omega,2,0.01))^2 + Re(G(\omega,2,0.01))^2}$$

$$\omega := 0, 0.02 .. 6$$

Представленный выше код использует встроенные средства пакета Mathcad для реализации численного интегрирования. Результат применения данной функции показан на рис. 2.5.

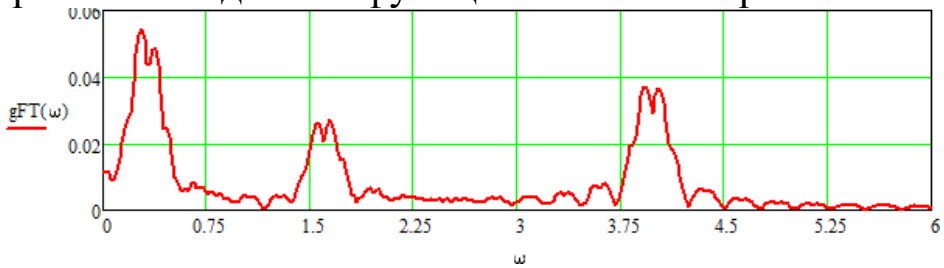
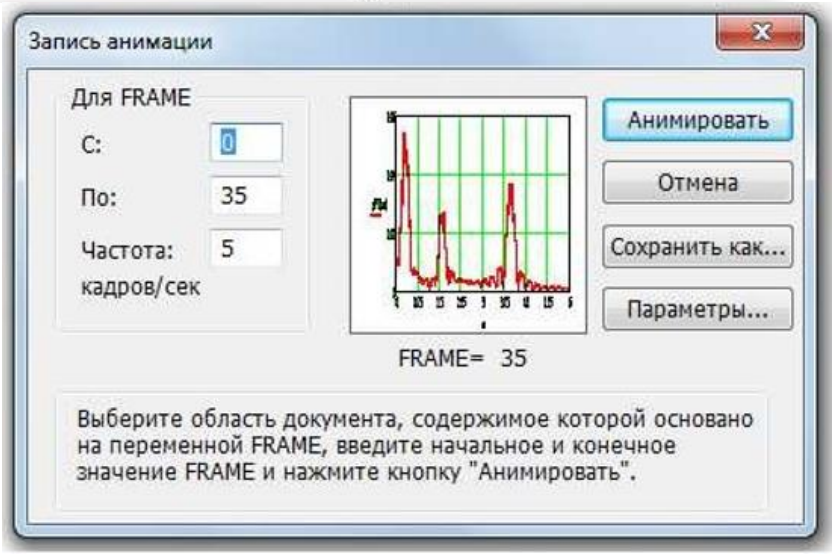


Рис. 2.5 Спектрограмма, полученная с помощью преобразования Габора для момента времени 2с

Одним из эффективных способов визуализации результата применения преобразования Габора является анимация изменения спектра по мере перемещения оконной функции. Ниже представлен код и настройки встроенной функции «Запись анимации» среды Mathcad.

$$FRAME1 := 0$$

$$gFT(\omega) := \sqrt{Im(G(\omega, FRAME1 + 2, 0.01))^2 + Re(G(\omega, FRAME1 + 2, 0.01))^2}$$



Рассмотрим реализация преобразования Габора программными методами. Для реализации метода численного интегрирования будем использовать метод прямоугольников.

$$Gabor(A, \omega MAX, \omega MIN, \Delta\omega, T, \alpha, \Delta) :=$$

$$Count \leftarrow \frac{\omega MAX - \omega MIN}{\Delta\omega}$$

for $i \in 0..Count$

$W \leftarrow 0$

$\omega \leftarrow i \cdot \Delta\omega + \omega MIN$

for $j \in \frac{T-2}{\Delta}..\frac{T+2}{\Delta}$

$t \leftarrow j \cdot \Delta$

$W \leftarrow W + \Delta \cdot A_{j,1} \cdot e^{-2\cdot\pi\cdot t\cdot i\cdot\omega} \cdot e^{-\pi\alpha(t-T)^2}$

$B_{i,0} \leftarrow \omega$

$B_{i,1} \leftarrow \sqrt{Im(W)^2 + Re(W)^2}$

B

Ниже показан пример вызова написанной функции и полученные результаты:

```
Res := Gabor(Signal,6,0,0.05,23,0.01,Δ)
```

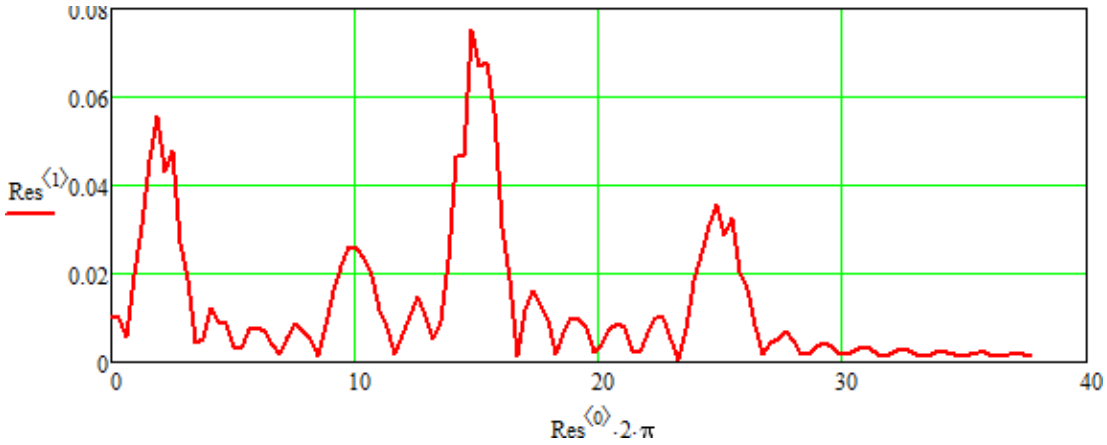


Рис. 2.6 Спектрограмма, полученная в результате преобразования Габора для момента времени 23с

Обратив внимание, что с помощью описанных методов может быть построена диаграмма для любого участка сигнала, при этом от ширины окна зависит как точность определения составляющих.

Задание на выполнение лабораторной работы

Задание на выполнение лабораторной работы состоит в нахождении времени появления и времени исчезновения гармоники сигнала используя один из описанных в работе методов оконного преобразования Фурье.

Метод оконного преобразования Фурье и исходные данные следует выбирать из таблицы 1 в соответствии с номером студента в списке группы.

Табл. 2.1 Задания на выполнение лабораторной работы

№	Задание
1	Оконное преобразование Фурье с прямоугольным окном, $T_1 = 3$с, $T_2 = 15$ с, $k = 2$
2	Преобразование Габора, $T_1 = 23$ с, $T_2 = 26$ с, $k = 3$
3	Преобразование Габора, $T_1 = 4$ с, $T_2 = 25$ с, $k = 0.5$
4	Оконное преобразование Фурье с прямоугольным окном, $T_1 = 1$с, $T_2 = 20$ с, $k = 0.2$
5	Преобразование Габора, $T_1 = 12$ с, $T_2 = 13$ с, $k = 5$
6	Преобразование Габора, $T_1 = 31$ с, $T_2 = 33$ с, $k = 12$
7	Оконное преобразование Фурье с прямоугольным окном, $T_1 = 5$с, $T_2 = 20$ с, $k = 0.1$
8	Преобразование Габора, $T_1 = 5$ с, $T_2 = 8$ с, $k = 20$
9	Преобразование Габора, $T_1 = 7$ с, $T_2 = 13$ с, $k = 6$
10	Оконное преобразование Фурье с прямоугольным окном, $T_1 = 0$с, $T_2 = 12$ с, $k = 3$

Лабораторная работа №3. Применение фильтра Гаусса для восстановления зашумленного сигнала

Цель работы: *изучение методов применения фильтра Гаусса для восстановления зашумленного сигнала.*

Аппаратные средства: *программный комплекс MATLAB, среда моделирования Simulink.*

Краткие теоретические сведения

Современные системы автоматического управления во многих случаях полагаются на информацию, поступающую с датчиков, для выработки управляющего сигнала. Точность информации, поступающей с датчика оказывает существенное влияние на качество работы системы автоматического управления. При этом на практике сигналы с датчиков зачастую оказываются «зашумлены». Под «зашумленным» сигналом подразумевают сигнал, представляющий собой смесь содержащего в себе полезную информацию сигнала и шума, сигнала не несущего полезной информационной нагрузки. Использование зашумленного сигнала в цепи обратной связи системы управления может вызвать ухудшение работы этой системы.

Для решения проблемы зашумленности сигнала рассмотрим метод фильтрации сигнала, использующий преобразование Фурье и фильтр Гаусса. Суть метода заключается в том, что над зашумленным сигналом производится преобразование Фурье для перехода к частотному домену, где на сигнал накладывается фильтр Гаусса. Подразумевается, что мы знаем какая область частот нас интересует. Над полученным сигналом производится обратное преобразование Фурье. В результате описанных действий получаем сигнал во временном домене, очищенный от шума.

Напомним, что преобразование Фурье описывается следующим образом:

$$f(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt$$

Применение фильтра Гаусса равносильно умножению исходного сигнала на функцию Гаусса, описываемую следующим выражением:

$$\omega(t) = \frac{1}{\sigma * \sqrt{2\pi}} e^{-\frac{t-t_0}{2\sigma^2}}$$

Данный метод подходит для борьбы с белым шумом. При наличии других видов шумов следует проанализировать применимость и эффективность метода перед его применением.

Методика выполнения лабораторной работы

Работа выполняется в среде MATLAB. До начала работы с кодом рекомендуется выбрать рабочую папку и создать новый .m file в этой папке. Сначала зададим функцию, имитирующую не зашумленный сигнал с датчика:

```
N = 10000;
dt = 0.001;
for i=1:N,
 A(i) = 0.85*sin(7*i*dt) + 0.9*sin( 4*i*dt) + 0.1*sin(10*i*dt) + 0.2*sin(12*i*dt);
end
plot(A);
```

Исполнение данного участка кода выведет на экран заданную функцию.

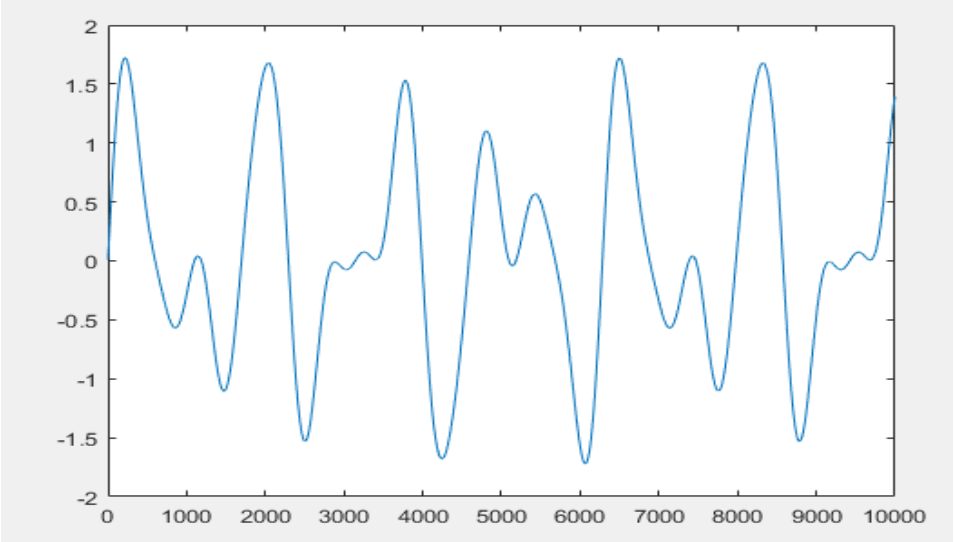


Рис. 3.1 Исходный сигнал

Добавим к сигналу имитацию белого шума. Для этого прибавим к каждому элементу вектора A случайную величину в диапазоне от 0 до 2:

```
N = 10000;
dt = 0.001;
for i=1:N,
  A(i) = 0.85*sin(7*i*dt) + 0.9*sin( 4*i*dt) + 0.1*sin(10*i*dt) + 0.2*sin(12*i*dt);
end
for i=1:N,
 A(i) = A(i) + 4.5*randn;
end
plot(A);
```

Исполнение данного участка кода выведет на экран отображение зашумленного сигнала:

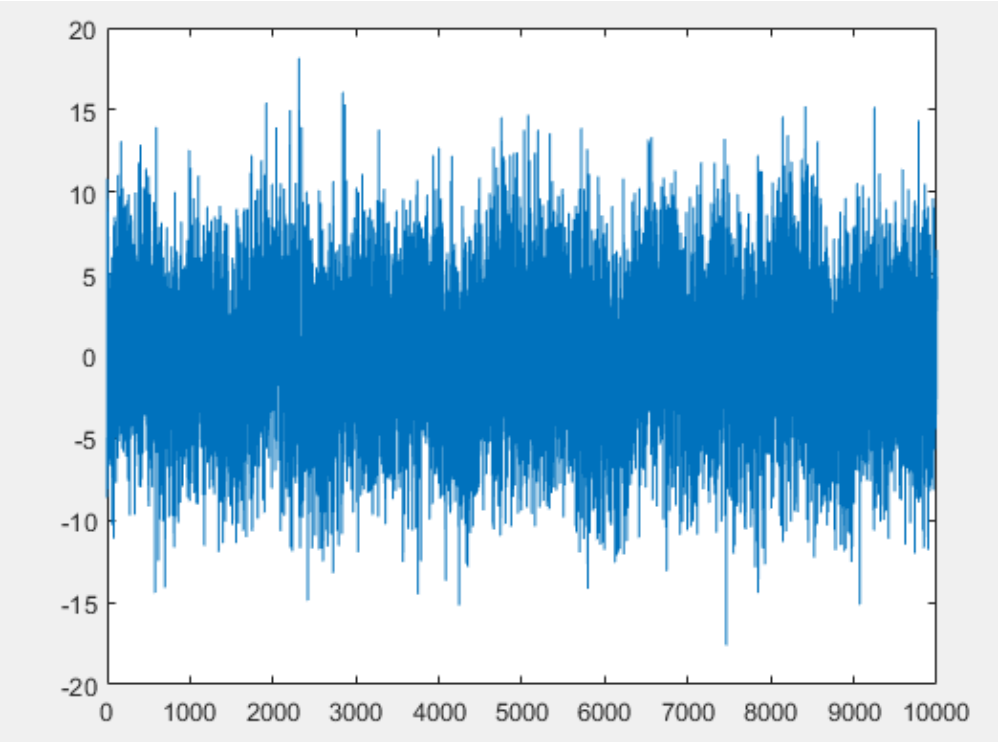


Рис. 3.2 Зашумленный сигнал

Используем быстрое преобразование Фурье для перехода в частотный домен. Воспользуемся встроенными функциями пакета MATLAB – fft() и fftshift().

```
N = 10000;
dt = 0.001;
for i=1:N,
 A(i) = 0.85*sin(7*i*dt) + 0.9*sin( 4*i*dt) + 0.1*sin(10*i*dt) + 0.2*sin(12*i*dt);
end
for i=1:N,
 A(i) = A(i) + 4.5*randn;
end
F = fft(A);
F = fftshift(F);
plot(abs(F));
```

Выполнение данного участка кода выведет на экран спектрограмму зашумленного сигнала:

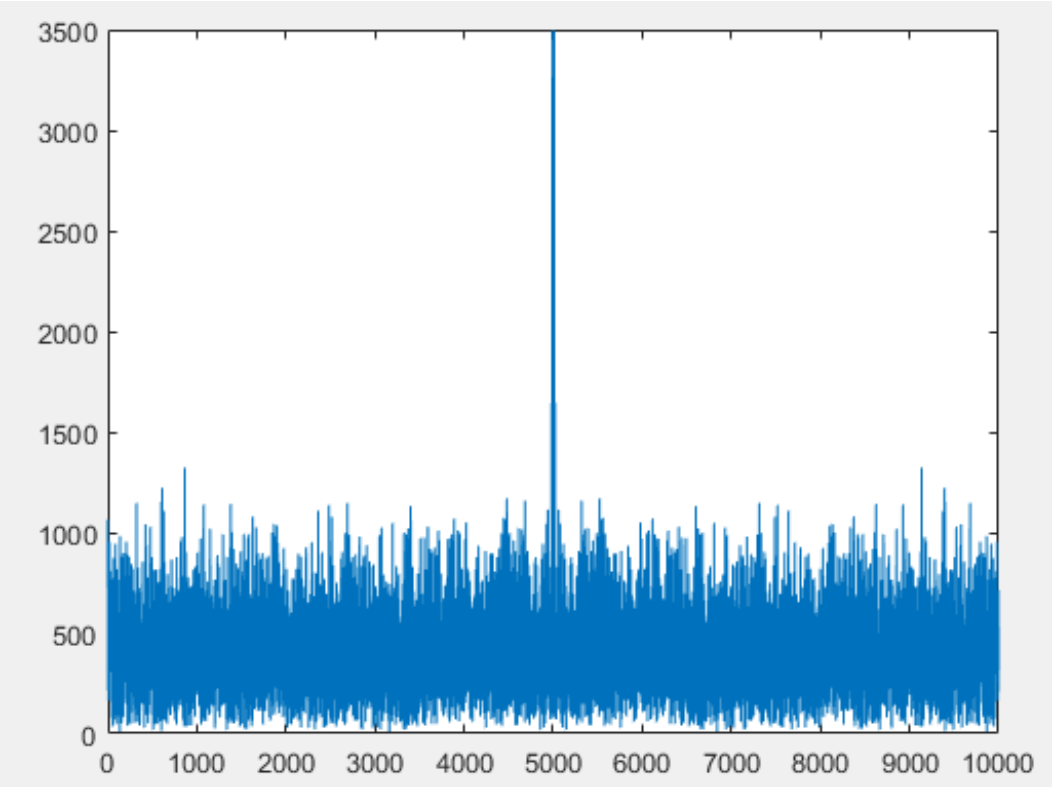


Рис. 3.3 Спектрограмма зашумленного сигнала

Обратим внимание, что на спектрограмме присутствует один пик, соответствующий частоте полезного сигнала. Остальные частоты представлены равномерно, что характерно для белого шума. Представленный ниже код генерирует фильтр Гаусса (Функцию Гаусса представленную в виде вектора той же размерности что и вектор, содержащий спектр зашумленного сигнала)

```
N = 10000;
dt = 0.001;
for i=1:N,
 A(i) = 0.85*sin(7*i*dt) + 0.9*sin( 4*i*dt) + 0.1*sin(10*i*dt) + 0.2*sin(12*i*dt);
end
for i=1:N,
 A(i) = A(i) + 4.5*randn;
end
F = fft(A);
F = fftshift(F);
GWP = 0.0001; %Это параметр определяет ширину гауссовой %функции
Fsize = size(F,2);
for i=1:Fsize,
 W(i) = exp(-GWP*(i - (Fsize/2)).^2);
end
plot(W);
```

Выполнение данного участка кода выведет на экран Гауссову функцию:

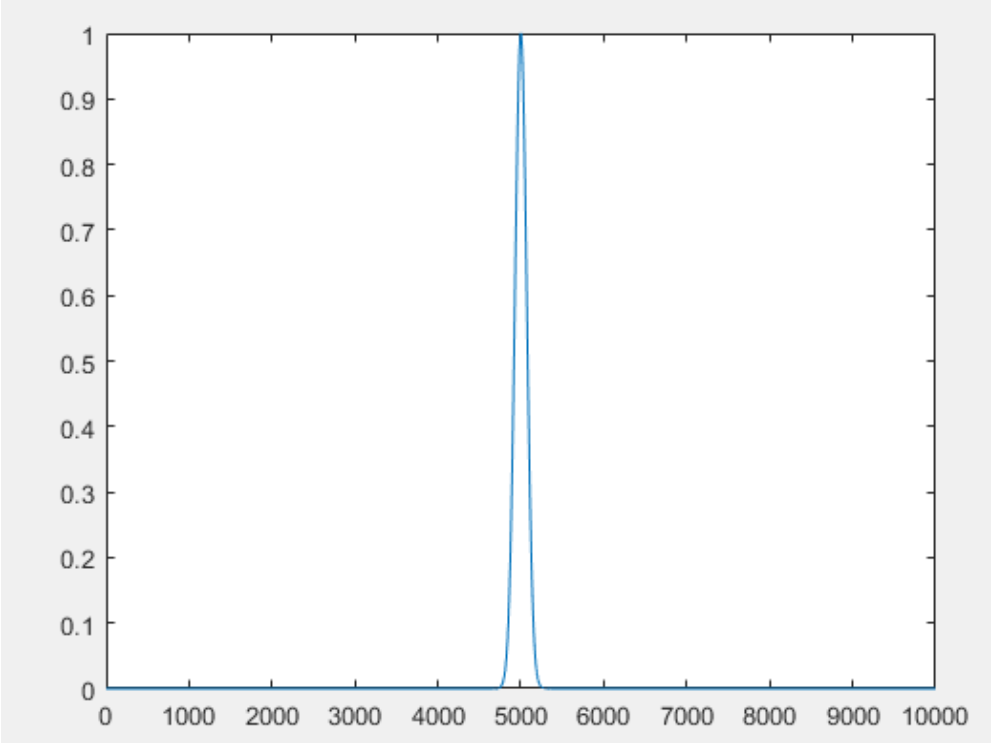


Рис. 3.4 Функция Гаусса

Умножим спектр зашумленного сигнала на полученную Гауссову функцию и применим к результату обратное преобразование Фурье:

```
N = 9*1024;
dt = 0.005;
B = 1:N;
A = 1:N;
for j=1:N,
 B(j) =  0.85*sin(7*j*dt) + 0.9*sin( 4*j*dt) + 0.1*sin(10*j*dt) + 0.2*sin(12*j*dt);
end
for j=1:N,
 A(j) = B(j) + 4.5*randn;
end
F = fft(A);
F = fftshift(F);
GWP = 0.01;
Fsize = size(F,2);
for j=1:Fsize,
 W(j) = exp(-GWP*(j - (Fsize/2)).^2) + 1*j*exp(-GWP*(j -(Fsize/2)).^2);
end
F_filt = F.*W;
Res = ifftshift(F_filt);
Res = ifft(Res);
Scale = abs( exp(-GWP*(0).^2) + 1i*exp(-GWP*(0).^2) );
Summ = 0;
for j=1:size(Res,2),


 Summ = Summ + abs(Res(j));
end
Summ = Summ / size(Res,2);
for j=1:size(Res,2),
 Res(j) = abs(Res(j)) + Summ*(1/Scale - 1);
end
subplot (2,3,1), plot(B);
subplot (2,3,2), plot(A);
subplot (2,3,3), plot(Res);
subplot (2,3,4), plot(abs(W));
subplot (2,3,5), plot(abs(F));
```

Выполнение данной программы выведет на экран пять графиков – исходный сигнал, зашумленный сигнал, очищенный сигнал, Гауссову функцию, спектрограмму зашумленного сигнала.

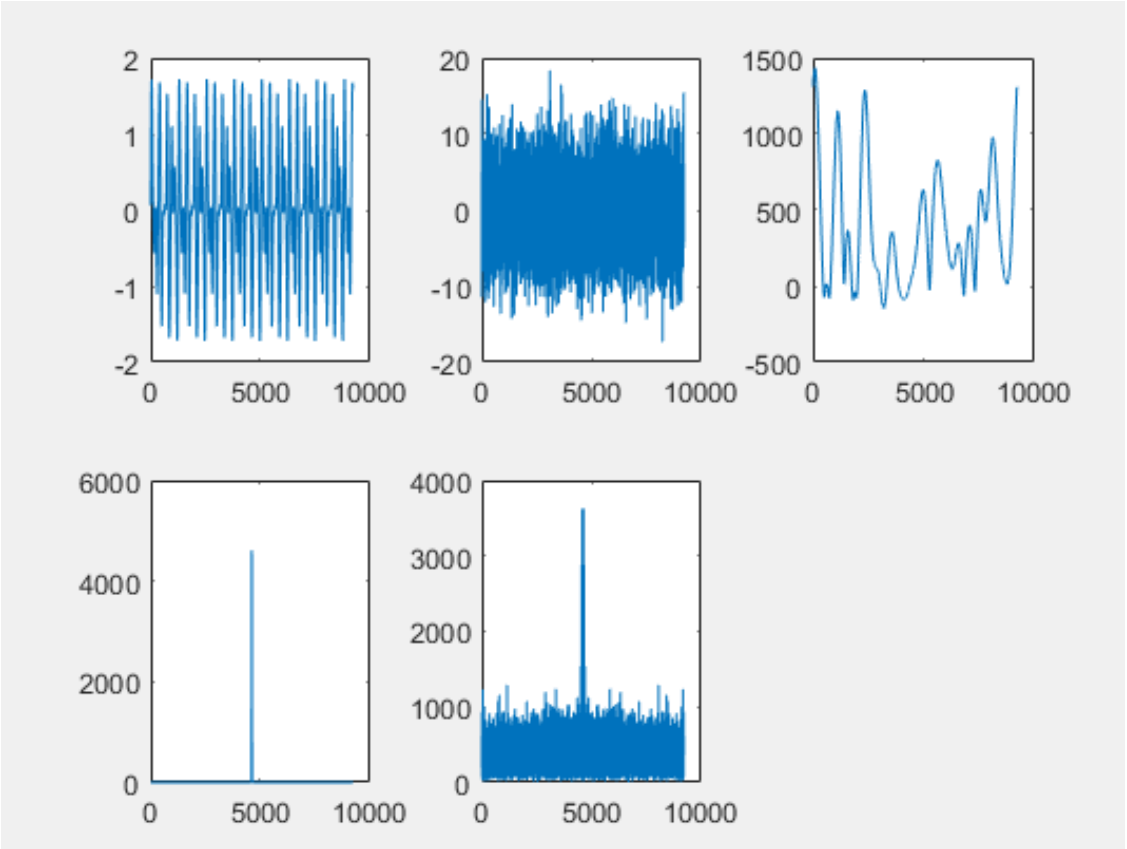


Рис. 3.5 Справа на лево: исходный сигнал, зашумленный сигнал, очищенный сигнал, Гауссова функция, спектрограмма зашумленного сигнала.

Как не сложно заметить, очищенный от шума сигнал не полностью совпадает с исходным сигналом. Это характерно для сигналов, полученных очисткой от белого шума с помощью наложения Гауссовой функции в частотном домене.

Задание на выполнение лабораторной работы

Задание на выполнение лабораторной работы состоит фильтрации гармонического сигнала от белого шума. Вид сигнала (функция F(t)) и амплитуду накладываемого белого шума η следует выбирать из таблицы 3.1 в соответствии с номером студента в списке группы.

Таблица 1 Задания на выполнение лабораторной работы

№	Задание
1	$F(t) = 0.35\sin(10 \cdot t) + 0.15\sin(8 \cdot t) + 0.35\sin(6 \cdot t),\ \eta = 3$
2	$F(t) = 0.15\sin(2 \cdot t) + 0.45\sin(8 \cdot t) + 0.85\sin(12 \cdot t) + 1.05\sin(0.5 \cdot t),$ $\eta = 5$
3	$F(t) = 0.25\sin(13 \cdot t) + 0.35\sin(12 \cdot t) + 0.75\sin(5 \cdot t) + 1.05\sin(4 \cdot t),$ $\eta = 3$
4	$F(t) = 0.45\sin(2 \cdot t) + 0.65\sin(0.5 \cdot t) + 0.05\sin(18 \cdot t) + 0.5\sin(9 \cdot t),$ $\eta = 1.5$
5	$F(t) = 0.85\sin(7 \cdot t) + 0.9\sin(4 \cdot t) + 0.1\sin(10 \cdot t) + 0.2\sin(12 \cdot t),$ $\eta = 4.5$
6	$F(t) = 0.6\cos(14 \cdot t) + 0.95\sin(2 \cdot t) + 0.5\sin(t) + 0.7\sin(3 \cdot t) +$ $+ 0.8\sin(12 \cdot t),$ $\eta = 5$
7	$F(t) = 0.1\cos(4 \cdot t) + 0.7\sin(6 \cdot t) + 0.6\sin(t) + 0.85\sin(12 \cdot t) +$ $+ 0.9\sin(8 \cdot t),$ $\eta = 3$
8	$F(t) = 0.5\cos(12 \cdot t) + 0.55\sin(17 \cdot t) + 0.65\sin(3 \cdot t) + 0.3\sin(15 \cdot t) +$ $+ 0.45\sin(t),$ $\eta = 4.5$
9	$F(t) = 0.5\sin(2 \cdot t) + 0.3\sin(12 \cdot t) + 0.1\sin(0.5 \cdot t),\ \eta = 1.5$
10	$F(t) = 0.2\sin(7 \cdot t) + 0.1\cos(6 \cdot t) + 0.3\cos(3 \cdot t),\ \eta = 3$

Лабораторная работа №4. Применение фильтра Гаусса для очистки изображения от белого шума

Цель работы: *изучение метода применения фильтра Гаусса дляочистки изображения от белого шума.*

Аппаратные средства: *программный комплекс MATLAB, средамоделирования Simulink.*

Краткие теоретические сведения

При обработке изображения зачастую ставится цель получения некоторой специфической информации об изображении. Например, это может быть информация о наличии на изображении какого-то конкретного объекта или определение позиции этого объекта. Для того, чтобы выделить объект на изображении удобно подвергнуть это изображение предварительной обработке. Достаточно часто такая обработка включает выделение контуров, имеющихся на обрабатываемом изображении.

Перед тем, как произвести выделение контуров, изображение, как правило, обрабатывают операторами, «подчеркивающими» контуры. Примерами таких операторов могут служить операторы Робертса, Собела и Лапласа. При этом качество полученного в результате преобразования изображения сильно зависит от уровня шума на исходном изображении. Так при наличии белого шума на изображении может привести к тому, что применение оператора Робертса приведет усилению высокочастотных составляющих этого шума и вместо «подчеркивания» контуров будем наблюдать «подчеркивания» шума.

Для очистки изображения от белого шума можно использовать преобразование Фурье и фильтрацию функцией Гаусса. В рамках данной работы рассмотрим очистку изображения от белого шума используя встроенные функции пакета MATLAB для обработки изображений, перехода к частотному домену, восстановления изображения по его спектру.

Методика выполнения лабораторной работы

Загрузим изображение из файла используя встроенную

функцию MATLAB – imread(). Функция имеет два параметра – имя файла с изображением, находящегося в рабочей папке и его расширение. Функция возвращает переменную в которую было загружено (трёхмерный массив). Для того, чтобы отобразить загруженный файл используем функцию imshow(). Параметром этой функции является переменная, в которую было загружено изображение.

```
Picture=imread('Image','jpeg');
imshow(Picture);
```

Результат работы приведённого участка коды показан на рисунке 4.1.

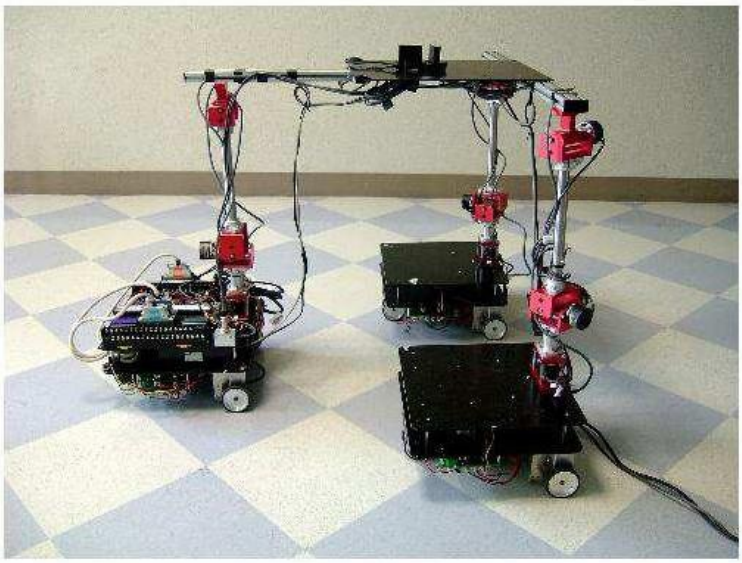


Рис. 4.1 Исходное изображение

Преобразуем исходное цветное изображение в черно-белое. Для этого используем встроенную функцию rgb2gray(). Данная функция преобразует трёхмерный массив, хранящий исходное цветное изображение в двухмерный, в серых тонах.

```
Picture=imread('Image','jpeg');
GreyPicture = rgb2gray(Picture);
imshow(GreyPicture);
```

Результат работы приведенного кода показан на рис. 4.2.

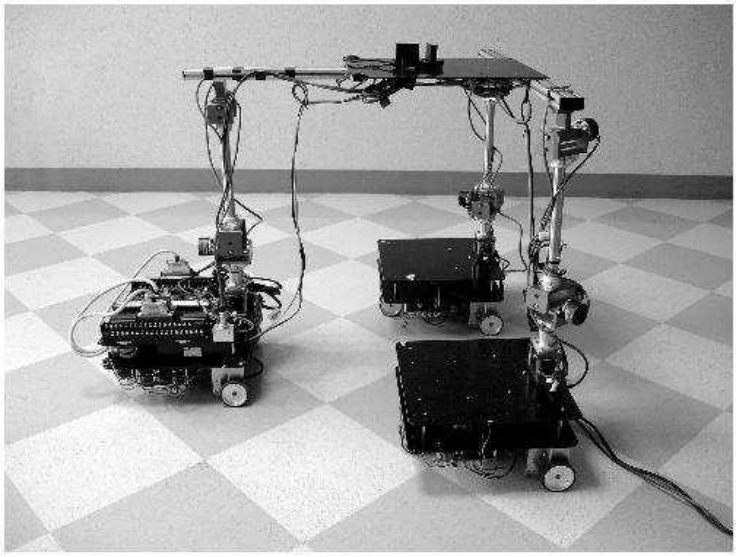


Рис. 4.2 Черно-белое изображение

Для удобства работы с изображением нам необходимо изменить формат хранения данных. Для этого воспользуемся функцией double(). По умолчанию изображение оказывается записано в формате, когда каждому пикселю соответствуют три байта (один байт на каждый цветовой канал RGB формата). После преобразования к серым тонам каждому пикселю соответствует 1 байт, хранящий информацию о яркости пикселя. Таким образом значение яркости пикселя изменяется в диапазоне от 0 до 255. Выполняемое с помощью функции double() преобразование позволяет перейти к формату, где каждому пикселю соответствует

2 байта, хранящие информацию о яркости пикселя, при этом значение яркости может изменяться в диапазоне от 0 до 65025.

```
Picture=imread('Image','jpeg');
GreyPicture = rgb2gray(Picture);
GreyPicture = double(GreyPicture); %меняем формат хранения данных

pcolor(GreyPicture), shading interp;
```

Обратим внимание на то, что для вывода изображения в новом формате требуется использовать другую функция отображения – pcolor(). Параметр shading interp определяет внешний вид поверхности.

Результат работы приведённого выше кода показан на рисунке

4.3.

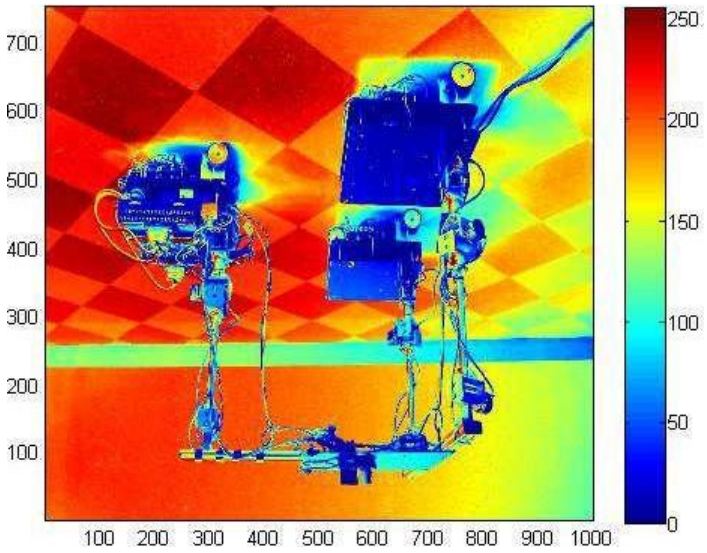


Рис. 4.3 Черно-белое изображение в формате double

Добавим на изображение белый шум. Сделать это можно с помощью кода, показанного ниже.

```
Picture=imread('Image','jpeg');
GreyPicture = rgb2gray(Picture);
GreyPicture = double(GreyPicture); %меняем формат хранения данных

for i=1:size(GreyPicture, 1),
for j=1:size(GreyPicture, 2),
GreyPicture(i,j) = GreyPicture(i,j)+100*rand(1);
end
end

pcolor(GreyPicture), shading interp;
```

Рассмотрим методику преобразования изображения в частотный домен. Для этого воспользуемся встроенными функциями fft2() и fftshift().

```
Picture=imread('Image','jpeg');
GreyPicture = rgb2gray(Picture);
GreyPicture = double(GreyPicture); %меняем формат хранения данных

F = fftshift(fft2(GreyPicture));
F = abs(F);
pcolor(log(F)), shading interp;
```

Результатом работы кода становится представление спектрограммы изображения в виде двухмерного массива.

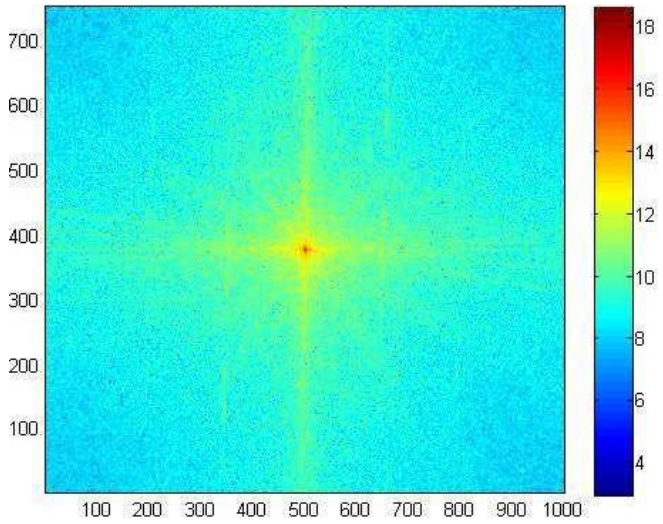


Рис. 4.4 Спектрограмма изображения

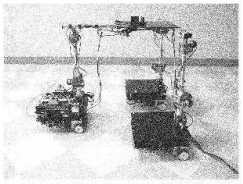
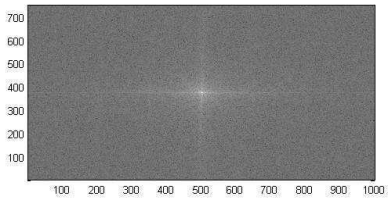
Приведем код, реализующий все рассмотренные выше операции. После преобразования изображения в частотный домен наложим на изображение фильтр использующий функцию Гаусса. После наложения фильтра очищенный спектр преобразуется функцией ifft2 совершающей обратную операцию преобразования Фурье для двухмерных частотных спектров.

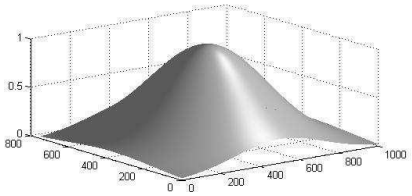
```
Picture=imread('Image','jpeg');
GreyPicture = rgb2gray(Picture);
GreyPicture = double(GreyPicture); %меняем формат хранения данных

for i=1:size(GreyPicture, 1),
for j=1:size(GreyPicture, 2),
```

```
GreyPicture(i,j) = GreyPicture(i,j)+ 100*rand(1);
end
end

F = fft2(GreyPicture);
F = fftshift(F);

xMax = size(F, 1);
yMax = size(F, 2);

GWP = 0.0001; %этот параметр определяет «ширину» Гауссовой
%функции

for i=1:xMax,
for j=1:yMax,
W(i,j) = exp(-GWP*(i - (xMax/2)).^2 - GWP*(j - (yMax/2)).^2);
end
end

F_filt = F.*W;
F_filt = ifftshift(F_filt);
F_filt = ifft2(F_filt);
FileteredImage = uint8(F_filt);

subplot(2,2,1), imshow(uint8(GreyPicture));
subplot(2,2,2), pcolor(log(abs(F))), shading interp;
subplot(2,2,3), surfl(W), shading interp, colormap(hot);
subplot(2,2,4), imshow(FileteredImage);
```

Обратим внимание, что для того, чтобы перейти к исходному изображению необходимо изменить формат представления изображения с помощью функции uint8().

Результат работы кода представлен на рисунках 4.5-4.10.



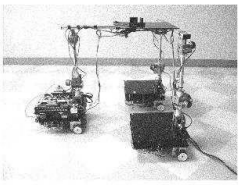
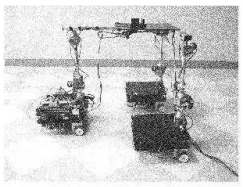
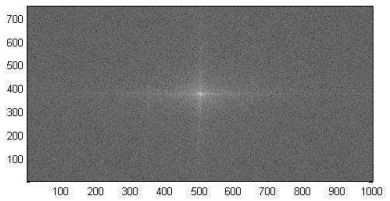
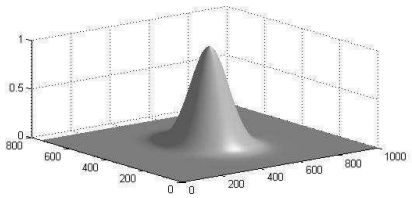


Рис. 4.5 Результат работы кода при GWP = 0.00001, белый шум в диапазоне от 0 до 100



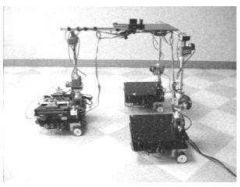
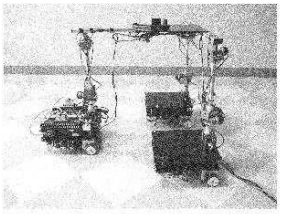
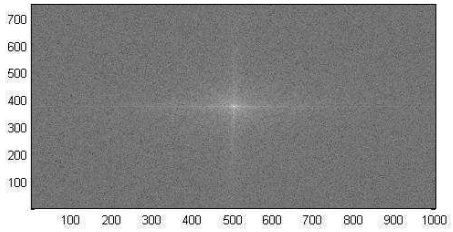
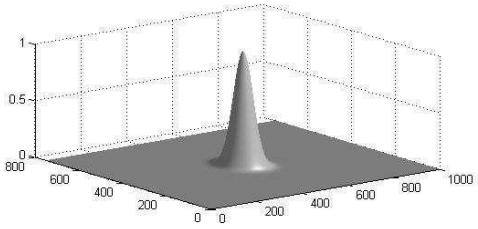


Рис. 4.6 Результат работы кода при GWP = 0.0001, белый шум в диапазоне от 0 до 100

Сравнивая результаты, полученные при различных значениях переменной GWP можно сделать следующие наблюдения. Как это очевидно из приведенного кода, при уменьшении значения переменной уменьшается «ширина» фильтрующей функции, как следствие большее количество высоких частот оказывается отфильтрованными. В результате при использовании более «узкого» фильтра удается в большей степени очистить изображение от шумов, что заметно, в том числе визуально. При этом полученное изображение оказывается размытым.



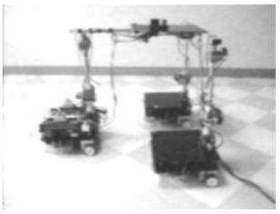
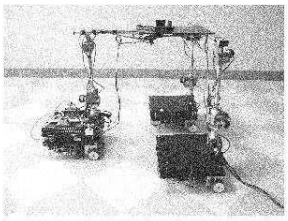
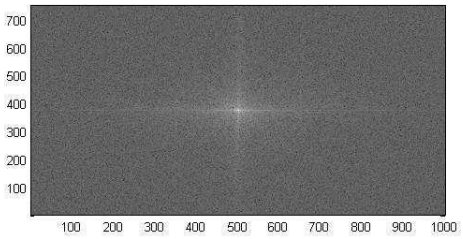
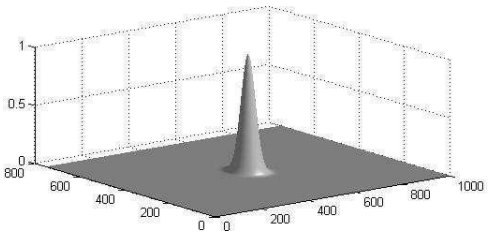


Рис. 4.7 Результат работы кода при GWP = 0.0005, белый шум в диапазоне от 0 до 100



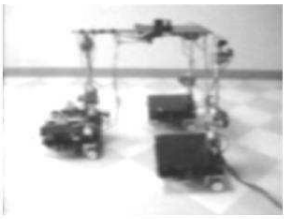


Рис. 4.8 Результат работы кода при GWP = 0.001, белый шум в диапазоне от 0 до 100

Тенденция, описанная выше проявляется и при дальнейшем снижении значения переменной GWP.

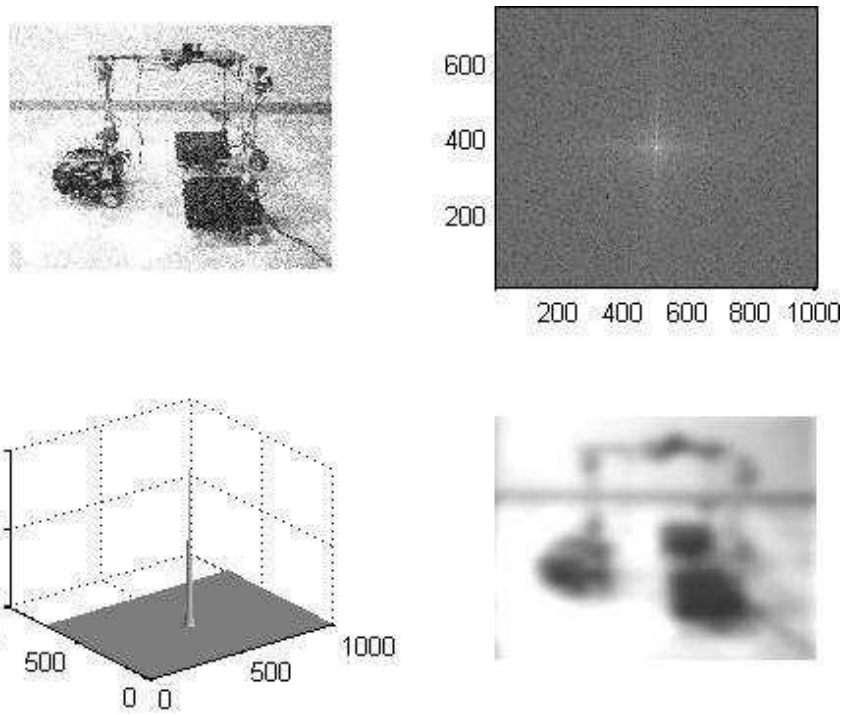


Рис. 4.9 Результат работы кода при GWP = 0.01, белый шум в диапазоне от 0 до 100

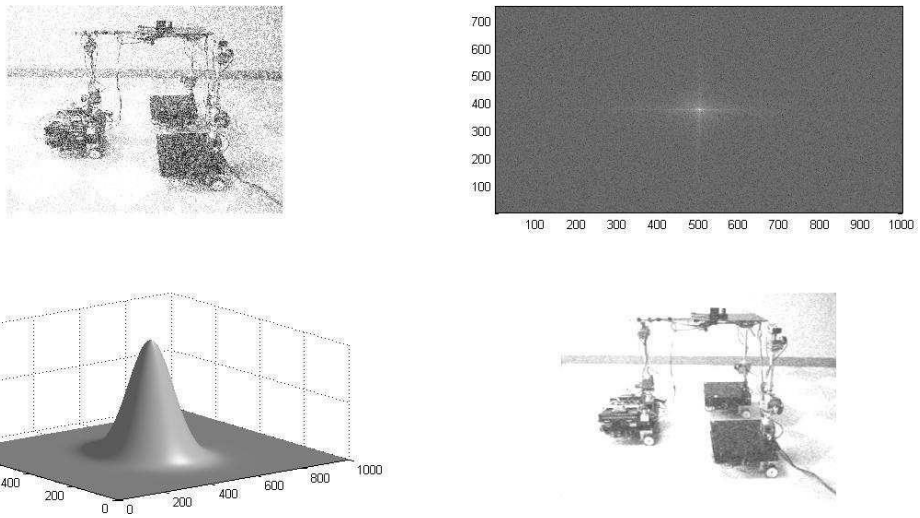


Рис. 4.10 Результат работы кода при GWP = 0.0001, белый шум в диапазоне от 0 до 200

Повысить уровень зашумленности можно увеличив амплитуду случайного сигнала, наложенного на изображение (см. рис. 4.10). Как видно из рис.4.10 полученный после фильтрации результат хуже, чем результат, полученный при тех же параметрах фильтра но меньшей амплитуде случайного сигнала (см. рис. 4.6).

Рассмотрим случай, когда помеха задана синусоидальной зависимостью:

$$f_{i,j} = g_{i,j} + A_x \sin(\omega_x i) + A_y \sin(\omega_y i)$$

где $f_{i,j}$ – i-й, j-й элемент массива, хранящего зашумленное изображение (i-й, j-й пиксель зашумленного изображения), $g_{i,j}$– соответствующий элемент массива, хранящего исходное изображение (соответствующий пиксель исходного изображения), A_x , A_y – константы, определяющие амплитуду помехи, ω_x, ω_y– константы, определяющие частоту помехи.

Приведем пример кода, реализующего наложение на изображение подобной помехи:

```
Ax = 100;
Ay = 100;
wx = 0.1;
wy = 0.1;
for i=1:size(GreyPicture, 1),
for j=1:size(GreyPicture, 2),
GreyPicture(i,j) = GreyPicture(i,j)+Ax*sin(i*wx)+Ay*sin(j*wy);
end
end
```

Примеры изображений, полученных таким образом, приведены на рис. 4.11.

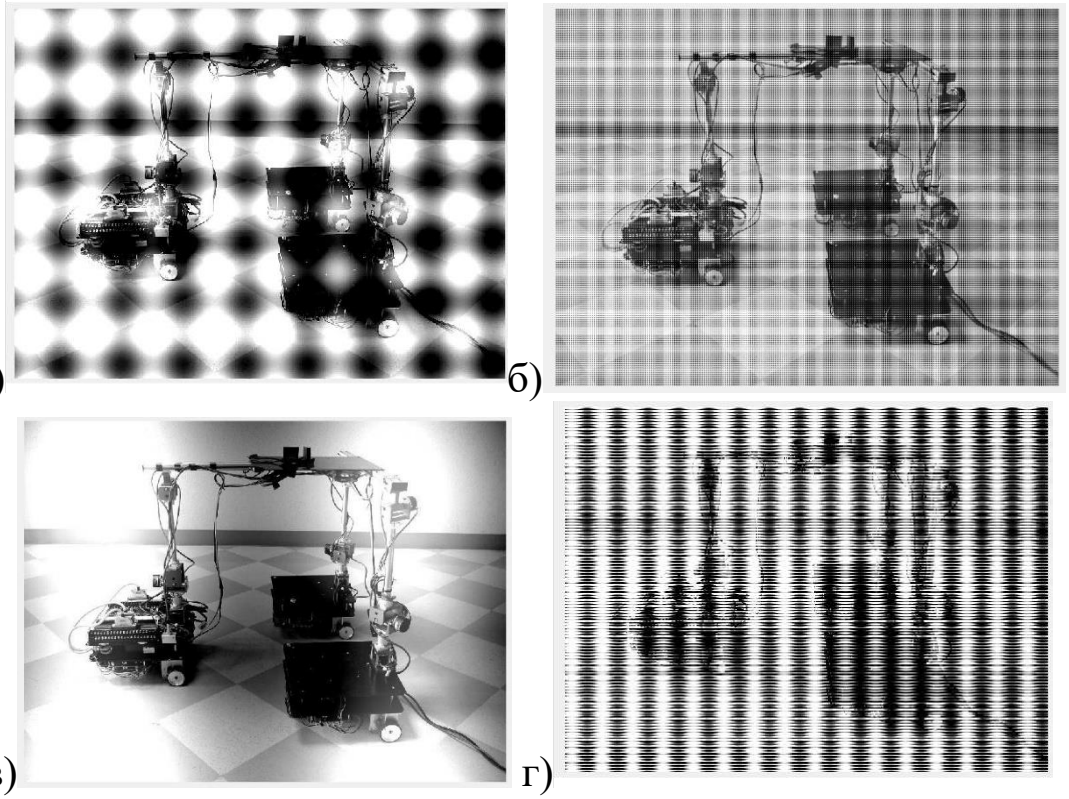


Рис. 4.11 Примеры изображений, зашумленных согласно приведенному ранее выражению

Помеха, заданная синусоидальными зависимостями, может также иметь следующую форму:

$$f_{i,j} = g_{i,j} + A\sin(\omega_x i + \omega_y i),$$

где A – константа, определяющая амплитуду помехи. Приведем пример кода, реализующего наложение на изображение подобной помехи:

```
A = 100;
wx = 0.1;
wy = 0.1;
for i=1:size(GreyPicture, 1),
for j=1:size(GreyPicture, 2),
GreyPicture(i,j) = GreyPicture(i,j)+A*sin(i*wx+j*wy);
end
end
```

Примеры изображений, полученных таким образом, приведены на рисунке 12.

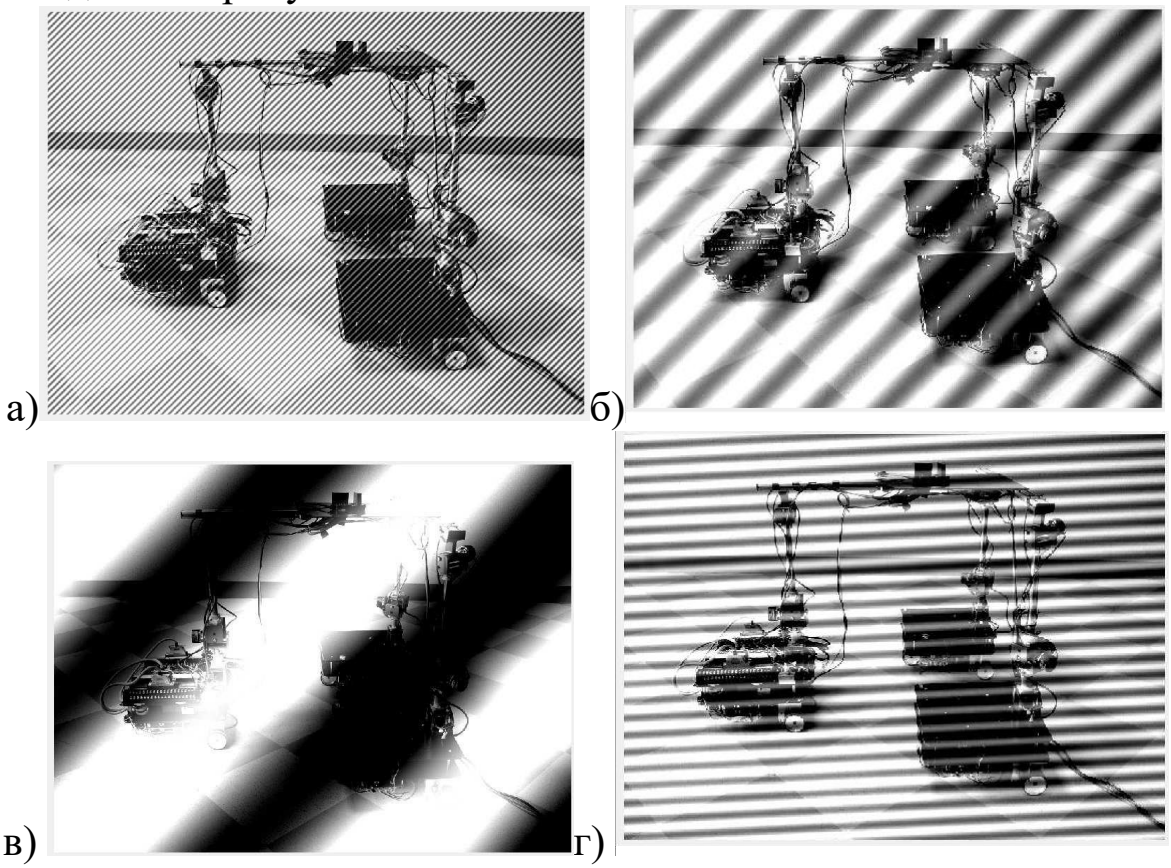


Рис. 4.12 Примеры изображений м помехами заданными синусоидальными зависимостями

Продемонстрируем, что фильтр Гаусса может быть применен для очистки изображения от помехи, заданной синусоидальной

зависимостью. Рассмотрим изображение, ан которое наложена помеха согласно выражению (2), причем A=500, $\omega_x = 4$, $\omega_y = 4$.

Результат обработки изображения фильтром Гаусса приведен на рисунке 4.13.

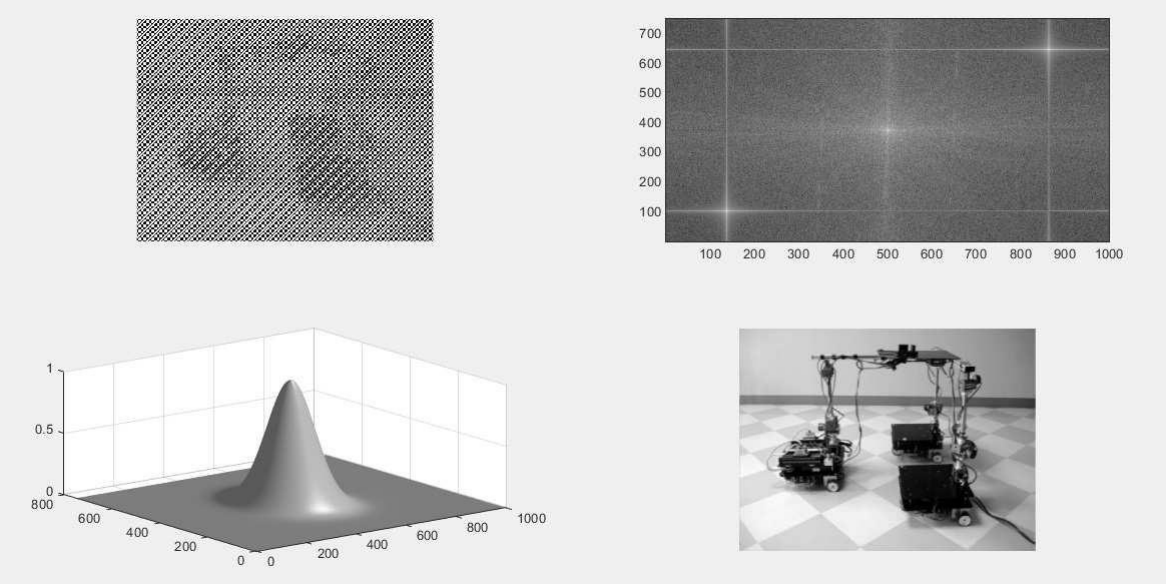


Рис. 4.13 Результат обработки зашумленного изображения фильтром Гаусса (помеха задана синусоидальной зависимостью, GWP = 0.0001)

Обратим внимание на то, что на спектре сигнала отчетливо видна составляющая, внесенная помехой.

Задание на выполнение лабораторной работы

Задание на выполнение лабораторной работы состоит в очистке изображения от белого шума. Использовать изображение форматов *.jpg, *.png, *.bmp. Использовать несколько вариантов ширины фильтра. Наложить белый шум в диапазоне от 0 до N (значение N взять из таблицы). Наложить на изображение дополнительное возмущение синусоидальной формы, форму функции взять из таблицы.

Табл. 4.1 Задания на выполнение лабораторной работы

№	Задание
1	N = 100, $F(t)=20\sin(12\cdot i+15\cdot j)$, $F(t)=200\sin(10\cdot i+5\cdot j)$
2	N = 250, $F(t)=30\sin(2\cdot i+3\cdot j)$, $F(t)=300\sin(12\cdot i+2\cdot j)$
3	N = 350, $F(t)=15\sin(12\cdot i+12\cdot j)$, $F(t)=150\sin(10\cdot i+1\cdot j)$
4	N = 500, $F(t)=22\sin(3\cdot i+2\cdot j)$, $F(t)=220\sin(10\cdot i+5\cdot j)$
5	N = 280, $F(t)=70\sin(13\cdot i+3\cdot j)$, $F(t)=700\sin(10\cdot i+5\cdot j)$
6	N = 120, $F(t)=25\sin(2\cdot i+2\cdot j)$, $F(t)=250\sin(10\cdot i+5\cdot j)$
7	N = 320, $F(t)=32\sin(3\cdot i+3\cdot j)$, $F(t)=320\sin(15\cdot i+12\cdot j)$
8	N = 400, $F(t)=25\sin(21\cdot i+21\cdot j)$, $F(t)=250\sin(21\cdot i+21\cdot j)$
9	N = 50, $F(t)=28\sin(15\cdot i+5\cdot j)$, $F(t)=280\sin(5\cdot i+5\cdot j)$
10	N = 220, $F(t)=16\sin(6\cdot i+7\cdot j)$, $F(t)=160\sin(4\cdot i+3\cdot j)$

Лабораторная работа №5. Применение разложения в тригонометрический ряд Фурьедля очистки сигнала от шума

Цель работы: *изучение методов реализации разложения в тригонометрический ряд Фурье периодических и не периодических сигналов средствами математического пакета MathCAD, а также использования полученного разложения для очистки сигнала от шума.*

Аппаратные средства: *математический пакет MathCAD.*

Краткие теоретические сведения

Периодическая функция может быть представлена своим разложением в тригонометрический ряд Фурье:

$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty}[a_n \cos(n \cdot t) + b_n \sin(n \cdot t)],$$

где a_0, a_n, b_n– коэффициенты ряда Фурье. Для их вычисления можно воспользоваться следующими формулами:

$$a_0 = \frac{1}{\pi}\int_{-\pi}^{\pi} f(t)dt,$$

$$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(t)\cos(n \cdot t)\,dt,$$

$$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(t)\sin(n \cdot t)\,dt$$

Более общая форма записи данного разложения имеет следующий вид:

$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty}[a_n \cos\left(\frac{n*\pi*t}{L}\right) + b_n \sin(\frac{n*\pi*t}{L})],$$

Для вычисления коэффициентов ряда следует воспользоваться следующими формулами:

$$a_0 = \frac{1}{L}\int_{-L}^{L} f(t)dt,$$

$$a_n = \frac{1}{L}\int_{-L}^{L} f(t)\cos\frac{n\cdot\pi\cdot t}{L}\,dt,$$

$$b_n = \frac{1}{L}\int_{-L}^{L} f(t)\sin\frac{n\cdot\pi\cdot t}{L}dt$$

Методика выполнения лабораторной работы

Зададимся некоторой полигармонической функцией *f(t)*:

$$f(t) = \sum_{i=1}^{n}[A_i \sin(\omega_i t) + B_i \cos(\omega_i t)]$$

В Mathcad это реализуется, например, следующим кодом:

```
F(t) := 0.25 sin(13·t) + 0.35 sin(12·t) + 0.75 sin(5·t) + 11.05 sin(1.2·t)
```

Покажем график функции f(t) (см. рис. 5.1).

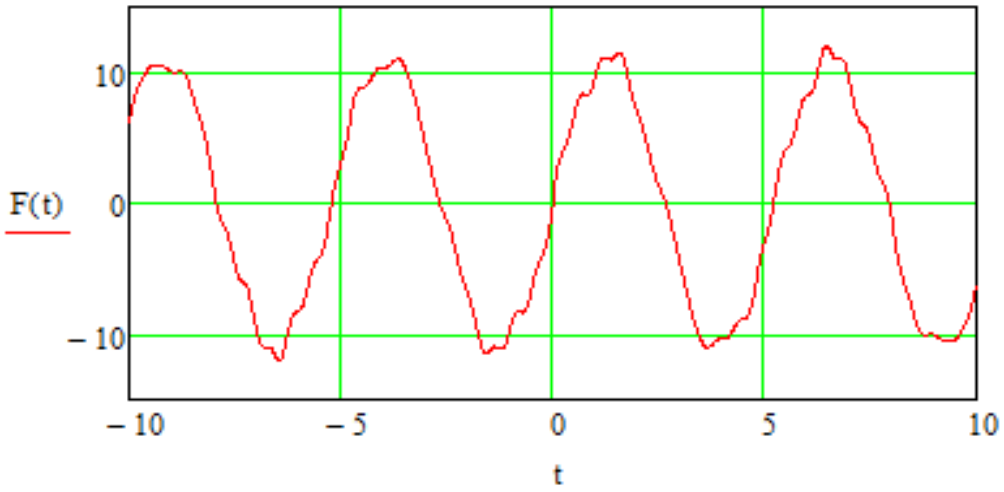


Рис. 5.1. Зависимость f(t)

Для реализации разложения в ряд Фурье воспользуемся аналитической формой выражений для коэффициентов ряда:

$$a0 := \frac{1}{\pi} \cdot \int_{-\pi}^{\pi} F(t)\, dt$$

$$GetA(n) := \frac{1}{\pi} \cdot \int_{-\pi}^{\pi} F(t) \cdot \cos(n \cdot t)\, dt$$

$$GetB(n) := \frac{1}{\pi} \cdot \int_{-\pi}^{\pi} F(t) \cdot \sin(n \cdot t)\, dt$$

Если требуется получить большое количество членов тригонометрического ряда удобно использовать вызов записанных выше функций в цикле:

Number := 300

a :=
for i ∈ 1..Number
  $Out_i \leftarrow GetA(i)$
Out

b :=
for i ∈ 1..Number
  $Out_i \leftarrow GetB(i)$
Out

Аппроксимация тригонометрическим рядом F(t) исходной функции имеет следующий вид:

$$f(t) := \frac{a0}{2} + \sum_{n=1}^{Number} \left(a_n \cdot \cos(n \cdot t) + b_n \cdot \sin(n \cdot t) \right)$$

Покажем на одном рисунке исходную функцию и её аппроксимацию.

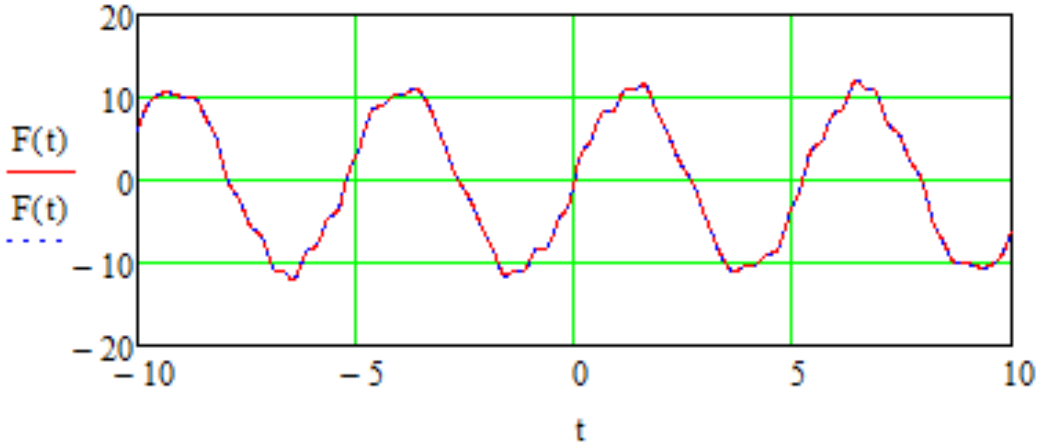


Рис. 5.2. Зависимость F(t) и f(t)

Можем видеть, что зависимости совпали. Набор коэффициентов тригонометрического ряда можно рассматривать как спектр функции. Обработаем этот спектр пороговой обработкой, реализованной следующим кодом:

```
threshold := 20

A := | for i ∈ 1..Number
     |     Out_i ← a_i  if ((i < threshold))
     |     Out_i ← 0  if ((i > threshold))
     | Out

B := | for i ∈ 1..Number
     |     Out_i ← b_i  if ((i < threshold))
     |     Out_i ← 0  if ((i > threshold))
     | Out
```

Построим функцию, используя полученные в результате обработки значения спектра:

$$Ffiltered(t) := \frac{a0}{2} + \sum_{n=1}^{Number} \left(A_n \cdot \cos(n \cdot t) + B_n \cdot \sin(n \cdot t)\right)$$

Исходная и полученная зависимость показана на рисунке 5.3

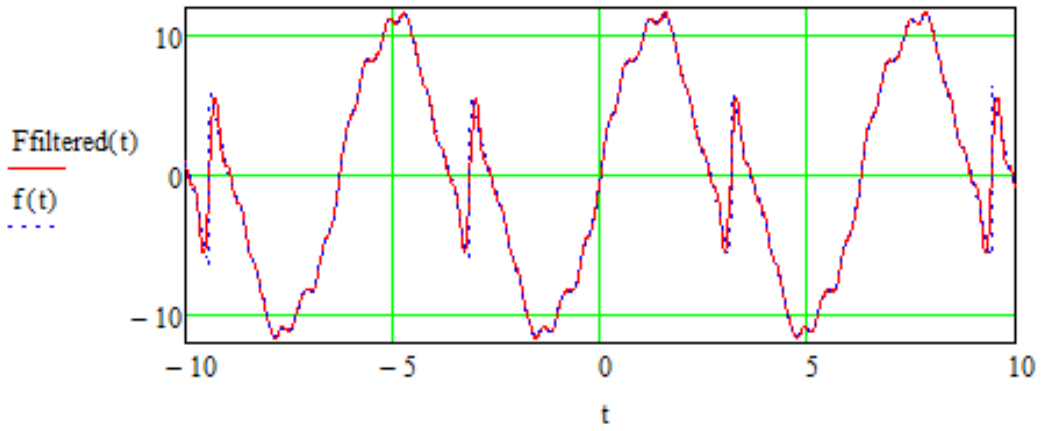


Рис. 5.3. Зависимость $F_{filtered}(t)$ и $f(t)$

Рассмотрим случай, когда функция f(t) непериодическая (см. рисунок 5.4):

$$f(t) = \sum_{i=1}^{n} [A_i \sin(\omega_i t) + B_i \cos(\omega_i t)] + \sum_{i=1}^{m} (a_i t^i)$$

Функцию указанного вида можно получить, например, используя следующий код:

$$f(t) := 10.15 \cdot \sin(2 \cdot t) + 0.45 \sin(8 \cdot t) + 0.85 \sin(12 \cdot t) + 1.05 \sin(10.5 \cdot t) + 5 \cdot t^3$$

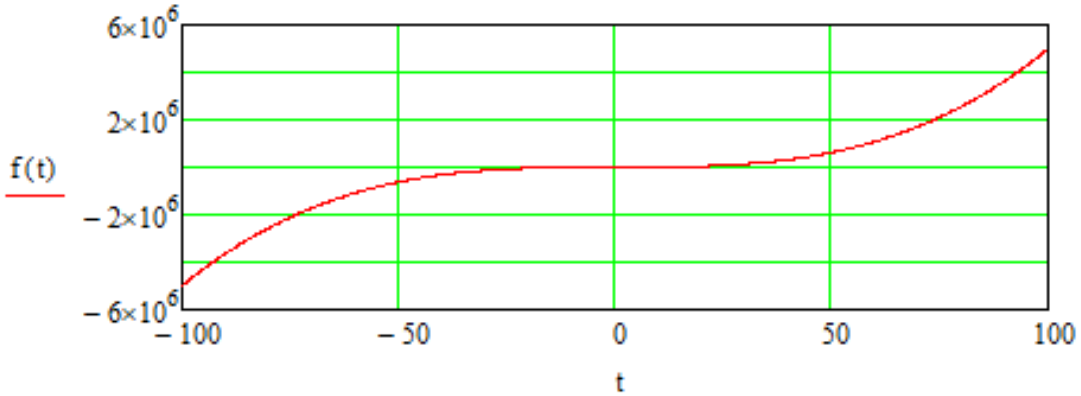


Рис. 5.4. Непериодическая зависимость

Используя написанный ранее код, произведем аппроксимацию заданной непериодической функции тригонометрическим рядом. Результат показан на рисунке 5.5.

После очистки спектра и восстановления сигнала получим следующую зависимость:

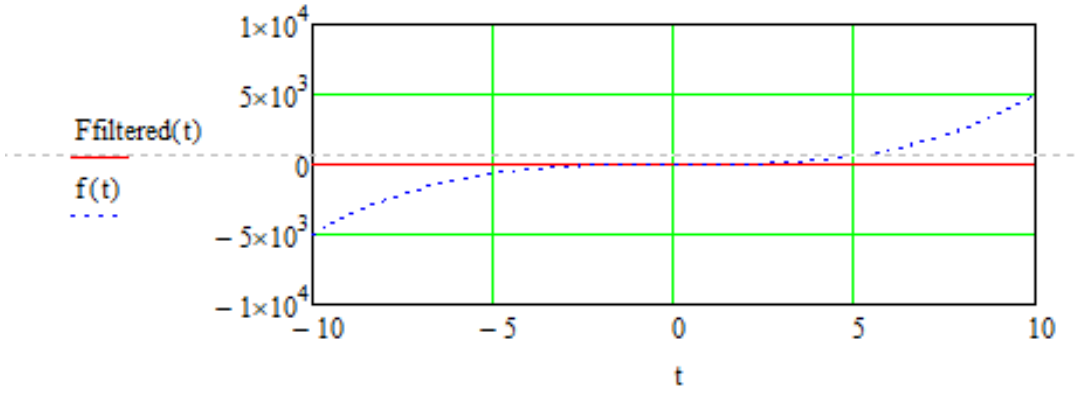


Рис. 5.5. Зависимость $F_{filtered}(t)$ и $f(t)$

Используем следующий код, позволяющий задавать размер период аппроксимации:

$$L := 2\pi$$

$$a0 := \frac{1}{L}\cdot\int_{-L}^{L} f(t)\,dt$$

$$GetA(n) := \frac{1}{L}\cdot\int_{-L}^{L} f(t)\cdot\cos\left(\frac{n\cdot\pi\cdot t}{L}\right)dt$$

$$GetB(n) := \frac{1}{L}\cdot\int_{-L}^{L} f(t)\cdot\sin\left(\frac{n\cdot\pi\cdot t}{L}\right)dt$$

$$Number := 300$$

$$a := \left|\begin{array}{l}\text{for } i \in 1..Number\\ \quad Out_i \leftarrow GetA(i)\\ Out\end{array}\right. \qquad b := \left|\begin{array}{l}\text{for } i \in 1..Number\\ \quad Out_i \leftarrow GetB(i)\\ Out\end{array}\right.$$

$$F(t) := \frac{a0}{2} + \sum_{n=1}^{Number}\left(a_n\cdot\cos\left(\frac{n\cdot\pi\cdot t}{L}\right) + b_n\cdot\sin\left(\frac{n\cdot\pi\cdot t}{L}\right)\right)$$

Период аппроксимации определяет отрезок времени, на котором исходная функция будет аппроксимирована точно. Он определяется переменной L.

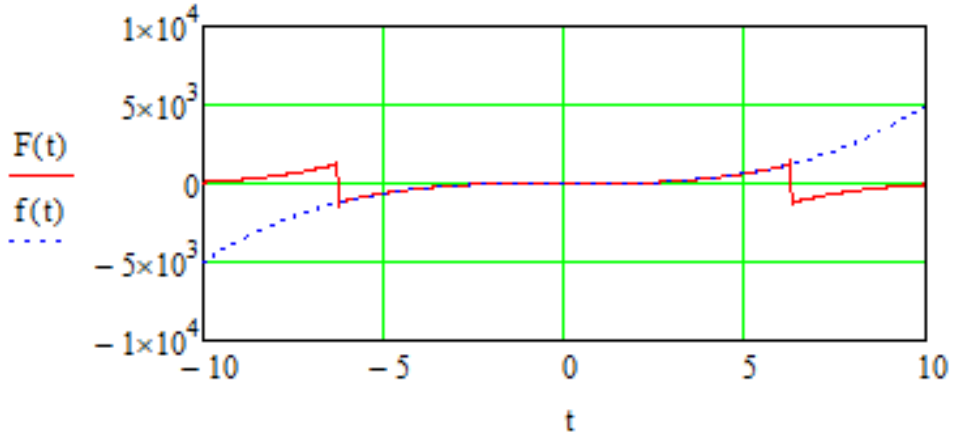


Рис. 5.6. Зависимости $F(t)$ и $f(t)$

Библиографический список

1. С.А. Воротников Информационные устройства робототехнических систем / Издательство МГТУ им. Н. Э. Баумана, 384 стр., 2005 г.

2. М. Я. Кельберт, Что такое преобразование Фурье?, Матем. просв., сер. 3, 4, МЦНМО, М., с 188–202, 2000 г.

3. В.А. Ильин, Математический анализ / В.А. Садовничий, Б.Х. Сендов // Главная редакция физико-математической литературы издательства "Наука", 720 стр., 1979 г.

4. И.И. Привалов Ряды Фурье / Либроком, 168 стр., 2011 г.

5. Вернер Вольфганг Рогозинский, Годфри Гарольд Харди Ряды Фурье / Либроком 152 стр., 2009 г.

6. Яцун, С.Ф. Информационные устройства и системы в мехатронике [Текст]: учебное пособие / С.Ф. Яцун, П.А. Безмен // Курск: Юго-Зап. гос. ун-т. – 2013. – 240 с.

7. Яцун, С.Ф. Датчики и обработка сигналов в мехатронике [Текст]: учебное пособие / С.Ф. Яцун, П.А. Безмен // Курск: Юго-Зап. гос. ун-т. – 2014. – 238 с.