МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное общеобразовательное учреждение высшего образования

«Ижевский государственный технический университет имени М.Т. Калашникова»

Кафедра «Программное обеспечение»

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Теория цифровой обработки сигналов»

на тему «Спектр сигнала»

Выполнил:

студент группы Б18-191-2 Р.А. Гумметов

Принял: И.О. Архипов

Ижевск 2020

**Часть 1. Вычисление спектра прямоугольного импульса**

1.1. Напишем функцию, которая возвращает массивы сигнала и меток времени

1.2. Сгенерируем прямоугольный импульс:

* частота опроса 10000 Гц;
* длительность 1024 отсчета;
* длительность импульса 10 мс.

1.3. Вычислим спектр

1.4. Отобразим сигнал и его спектр

Результат

На рис.1 представлен результат работы программы

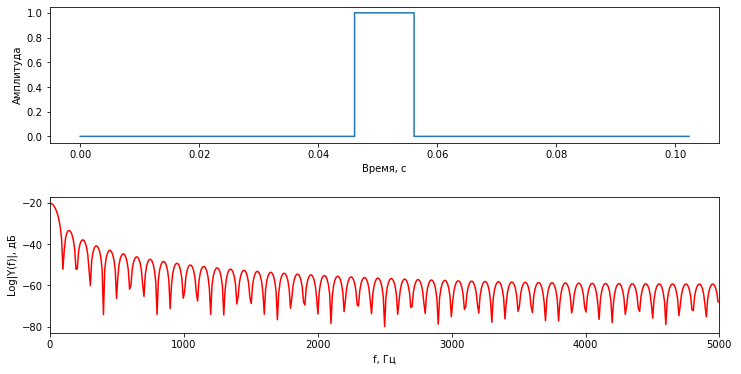


Рис. 1

Код программы для части 1

def imp\_generator(fd, N, d\_imp):

dt = 1.0/fd #Период опроса, с

d=N/fd #Длительность сигнала, с

n\_imp=int(d\_imp\*fd) #Длительность импульса, отсчеты

begin\_imp=round(fd\*d/2-fd\*d\_imp/2) #Начало импульса, отс.

end\_imp=begin\_imp+n\_imp #Конец импульса, отс.

t = np.arange(0.0, d, dt) #Массив меток времени, с

signal\_imp = np.zeros(len(t)) # Инициализация сигнала нулями

signal\_imp[int(begin\_imp):int(end\_imp)] = 1.0 # Создание импульса

return signal\_imp, t

fd=10000 #Частота опроса, Гц

N=1024 #Длительность сигнала, отс

d\_imp=0.01 #Длительность импульса, с

signal\_imp1, t1=imp\_generator(fd, N, d\_imp)

k = np.arange(N) # Номера отсчетов спектра

T = float(N)/fd

frq = k/T # Привязка отсчетов спектра к частотам (полный диапазон частот)

# Вставьте свой код для вычисления и нормализации комплексного спектра

Ycomplex = fft(signal\_imp1)/N

# Вставьте свой код для вычисления спектра амплитуд

Y = np.abs(Ycomplex)

# Вставьте свой код для вычисления спектра амплитуд в логарифмическом масштабе

Ylog = 20\*np.log10(Y+0.0001)

f, ax = plt.subplots(2, 1,figsize=(12, 6))

ax[0].step(t1, signal\_imp1)

ax[0].set\_xlabel('Время, с')

ax[0].set\_ylabel('Амплитуда')

# Отобразить спектр амплитуд (половину частотного диапазона)

ax[1].plot(frq[0:int(N/2)],Ylog[0:int(N/2)],'r') #plotting the log spectrum

ax[1].set\_xlabel('f, Гц')

ax[1].set\_ylabel('Log|Y(f)|, дБ')

ax[1].set\_xlim([0,frq[int(N/2)]])

plt.subplots\_adjust(wspace=0.12, hspace=0.4)

## Часть 2. Вычисление спектра прямоугольного импульса

2.1. Используя функцию-генератор, создайте три прямоугольных импульса (в разных массивах):

* частоты опроса 10000Гц;
* длительности сигналов 1024 отсчета;
* длительности импульсов 4 мс., 2 мс., 1 мс.

2.2. Вычислите спектры сигналов

2.3. Измерение параметров спектра импульсов

Результат

На рис.2 представлен результат работы программы

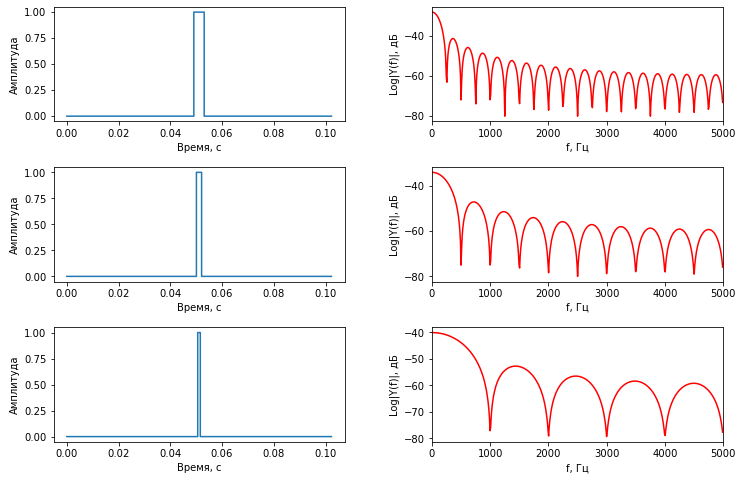


Рис. 2

Код программы для части 2

#Отредактируйте код для выполнения задания

fd = 10000

N = 1024

d\_imp1 = 0.004

d\_imp2 = 0.002

d\_imp3 = 0.001

signal\_imp1, t1 = imp\_generator(fd, N, d\_imp1)

signal\_imp2, t2 = imp\_generator(fd, N, d\_imp2)

signal\_imp3, t3 = imp\_generator(fd, N, d\_imp3)

Ycomplex1 = fft(signal\_imp1)/N

Ycomplex2 = fft(signal\_imp2)/N

Ycomplex3 = fft(signal\_imp3)/N

Y1 = np.abs(Ycomplex1)

Y2 = np.abs(Ycomplex2)

Y3 = np.abs(Ycomplex3)

k = np.arange(N) # Номера отсчетов спектра

T = float(N)/fd

frq = k/T # Привязка отсчетов спектра к частотам (полный диапазон частот)

# Вставьте свой код для вычисления и нормализации комплексного спектра

Ylog1 = 20\*np.log10(Y1+0.0001)

Ylog2 = 20\*np.log10(Y2+0.0001)

Ylog3 = 20\*np.log10(Y3+0.0001)

f, ax = plt.subplots(3, 2,figsize=(12, 8))

ax[0][0].step(t1, signal\_imp1)

ax[0][0].set\_xlabel('Время, с')

ax[0][0].set\_ylabel('Амплитуда')

ax[0][1].plot(frq[0:int(N/2)],Ylog1[0:int(N/2)],'r') #plotting the log spectrum

ax[0][1].set\_xlabel('f, Гц')

ax[0][1].set\_ylabel('Log|Y(f)|, дБ')

ax[0][1].set\_xlim([0,frq[int(N/2)]])

ax[1][0].step(t1, signal\_imp2)

ax[1][0].set\_xlabel('Время, с')

ax[1][0].set\_ylabel('Амплитуда')

ax[1][1].plot(frq[0:int(N/2)],Ylog2[0:int(N/2)],'r') #plotting the log spectrum

ax[1][1].set\_xlabel('f, Гц')

ax[1][1].set\_ylabel('Log|Y(f)|, дБ')

ax[1][1].set\_xlim([0,frq[int(N/2)]])

ax[2][0].step(t1, signal\_imp3)

ax[2][0].set\_xlabel('Время, с')

ax[2][0].set\_ylabel('Амплитуда')

ax[2][1].plot(frq[0:int(N/2)],Ylog3[0:int(N/2)],'r') #plotting the log spectrum

ax[2][1].set\_xlabel('f, Гц')

ax[2][1].set\_ylabel('Log|Y(f)|, дБ')

ax[2][1].set\_xlim([0,frq[int(N/2)]])

plt.subplots\_adjust(wspace=0.3, hspace=0.4)

#### Задание

1. Программным способом выполните измерение уровня боковых лепестков и ширины главного лепестка в спектрах сгенерированных ранее прямоугольных импульсов. Рекомендуется оформить код в виде функции.
2. В отчете оформите результаты измерений в виде таблицы
3. Проанализируйте зависимость ширины главного лепестка спактра импульса от длительности импульса. Постарайтесь сделать вывод.

Результат

На рис.3 представлен результат работы программы

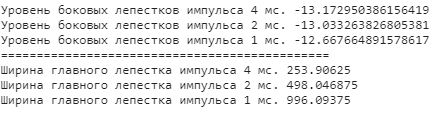


Рис. 3

|  |  |  |
| --- | --- | --- |
| Длительность окна, мс | Уровень боковых лепестков, дБ | Ширина главного лепестка, Гц |
| 1 | -12,668 | 996,094 |
| 2 | -13,033 | 498,047 |
| 4 | -13,173 | 253,906 |

Чем меньше длительность импульса, тем больше ширина главного лепестка спектра импульса.

Код программы

#Напишите код для измерения характеристик спектра импульсов

def fun(Ylog):

i = 0

while Ylog[i] > Ylog[i + 1]:

i = i + 1

m = frq[i] - frq[0]

da = max(Ylog[i : (int)(N/2)]) - Ylog[0]

return m, da

m1, da1 = fun(Ylog1)

m2, da2 = fun(Ylog2)

m3, da3 = fun(Ylog3)

print ('Уровень боковых лепестков импульса 4 мс.',da1)

print ('Уровень боковых лепестков импульса 2 мс.',da2)

print ('Уровень боковых лепестков импульса 1 мс.',da3)

print ('==============================================')

print ('Ширина главного лепестка импульса 4 мс.',m1)

print ('Ширина главного лепестка импульса 2 мс.',m2)

print ('Ширина главного лепестка импульса 1 мс.',m3)

# Часть 3. Спектр гармонического сигнала

3.1. Сгенерируйте сигнал, состоящий из двух синусоидальных гармоник с параметрами:

* чатота дискретизации 10000 Гц;
* длительность сигнала 1024 отсчета;
* амплитуды гармоник 1.0 и 0.7;
* частоты гармоник 300 Гц и 500 Гц.

Результат

На рис.4 представлен результат работы программы

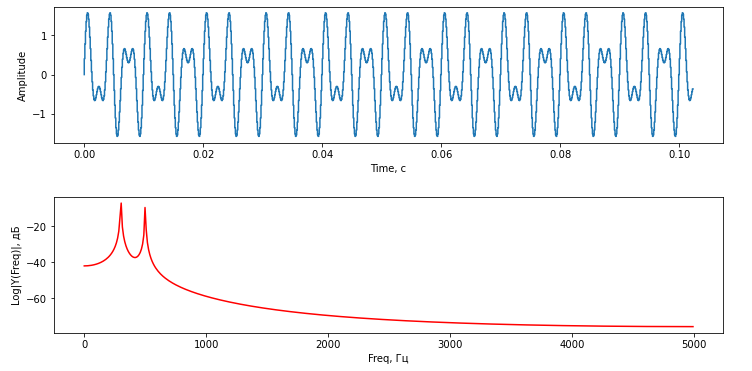


Рис. 4

Код программы для части 3

# Введите код для выполнения задания

fd=10000

dt = 1.0/fd

d = 0.1024

A1 = 1.0

A2 = 0.7

t = np.arange(0.0, d, dt)

s1 = np.sin(2\*np.pi\*300\*t)

s2 = A2 \* np.sin(2\*np.pi\*500\*t)

y = s1 + s2

N=1024 #Размер БПФ

k = np.arange(N) # Номера отсчетов спектра

T = float(N)/fd

frq = k/T # Привязка отсчетов спектра к частотам (полный диапазон частот)

Ycomplex = fft(y)/N #Вычисление БПФ и нормализация

Y = np.abs(Ycomplex) #Вычислние спектра амплитуд

Ylog1 = 20\*np.log10(Y+0.0001) #Спектр амплитуд в логарифмическом масштабе

f, ax = plt.subplots(2, 1,figsize=(12, 6))

ax[0].step(t, y)

ax[0].set\_xlabel('Time, с')

ax[0].set\_ylabel('Amplitude')

ax[1].plot(frq[0:int(N/2)],Ylog1[0:int(N/2)],'r') #plotting the log spectrum

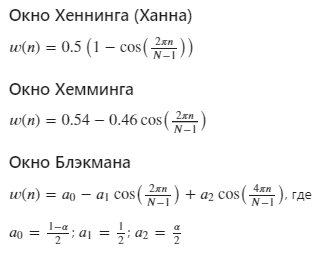
ax[1].set\_xlabel('Freq, Гц')

ax[1].set\_ylabel('Log|Y(Freq)|, дБ')

plt.subplots\_adjust(wspace=0.12, hspace=0.4)

# Часть 4. Оконное преобразование Фурье (кратковременный спектральный анализ)

Минимизировать эффект просачивания можно выбрав окно с меньшей амплитудой боковых лепестков. Предложено ряд функций окна, применение которых позволяет снизить уровень боковых лепестков по сравнению с тем их уровнем, который они имеют в случае прямоугольного окна.



4.1. Исследуем влияние длительности окна на спектр сигнала. Сгенерируйте сигнал, состоящий из двух синусоидальных гармоник с параметрами:

* чатота дискретизации 10000 Гц;
* длительность сигнала 1024 отсчета;
* амплитуды гармоник 1.0 и 0.7;
* частоты гармоник 300 Гц и 500 Гц.

4.2. Домножте сгенерированный сигнал на прямоугольное окно длительностью 10 мс, 40 мс и 80 мс. Вычислите и отобразите спектры полученных сигналов.

Результат

На рис.5 представлен результат работы программы

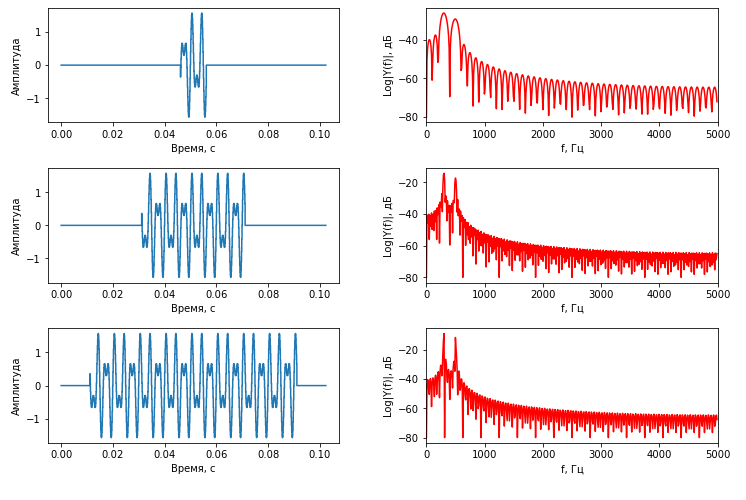


Рис. 5

Код программы

# Введите код для выполнения задания

fd = 10000

N = 1024

A1 = 1.0

A2 = 0.7

# Сгенерируйте сигнал

t1 = np.arange(0.0, d, dt)

s1 = np.sin(2\*np.pi\*300\*t1)

s2 = A2 \* np.sin(2\*np.pi\*500\*t1)

y = s1 + s2

#Сгенерируйте прямоугольные окна

win1, t1=imp\_generator(fd, N, 0.01)

win2, t1=imp\_generator(fd, N, 0.04)

win3, t1=imp\_generator(fd, N, 0.08)

#Домножте сигнал на окна

yw1=y\*win1

yw2=y\*win2

yw3=y\*win3

#Вычислите спектры

k = np.arange(N) # Номера отсчетов спектра

T = float(N)/fd

frq = k/T # Привязка отсчетов спектра к частотам (полный диапазон частот)

Ycomplex1 = fft(yw1)/N #Вычисление БПФ и нормализация

Y1 = np.abs(Ycomplex1) #Вычислние спектра амплитуд

Ylog1 = 20\*np.log10(Y1+0.0001) #Спектр амплитуд в логарифмическом масштабе

Ycomplex2 = fft(yw2)/N #Вычисление БПФ и нормализация

Y2 = np.abs(Ycomplex2) #Вычислние спектра амплитуд

Ylog2 = 20\*np.log10(Y2+0.0001) #Спектр амплитуд в логарифмическом масштабе

Ycomplex3 = fft(yw3)/N #Вычисление БПФ и нормализация

Y3 = np.abs(Ycomplex3) #Вычислние спектра амплитуд

Ylog3 = 20\*np.log10(Y3+0.0001) #Спектр амплитуд в логарифмическом масштабе

#Отобразите сигналы и спектры

f, ax = plt.subplots(3, 2,figsize=(12, 8))

ax[0][0].step(t1, yw1)

ax[0][0].set\_xlabel('Время, с')

ax[0][0].set\_ylabel('Амплитуда')

ax[0][1].plot(frq[0:int(N/2)],Ylog1[0:int(N/2)],'r') #plotting the log spectrum

ax[0][1].set\_xlabel('f, Гц')

ax[0][1].set\_ylabel('Log|Y(f)|, дБ')

ax[0][1].set\_xlim([0,frq[int(N/2)]])

ax[1][0].step(t1, yw2)

ax[1][0].set\_xlabel('Время, с')

ax[1][0].set\_ylabel('Амплитуда')

ax[1][1].plot(frq[0:int(N/2)],Ylog2[0:int(N/2)],'r') #plotting the log spectrum

ax[1][1].set\_xlabel('f, Гц')

ax[1][1].set\_ylabel('Log|Y(f)|, дБ')

ax[1][1].set\_xlim([0,frq[int(N/2)]])

ax[2][0].step(t1, yw3)

ax[2][0].set\_xlabel('Время, с')

ax[2][0].set\_ylabel('Амплитуда')

ax[2][1].plot(frq[0:int(N/2)],Ylog3[0:int(N/2)],'r') #plotting the log spectrum

ax[2][1].set\_xlabel('f, Гц')

ax[2][1].set\_ylabel('Log|Y(f)|, дБ')

ax[2][1].set\_xlim([0,frq[int(N/2)]])

plt.subplots\_adjust(wspace=0.3, hspace=0.4)

4.3. Исследуем влияние формы окна на спектр сигнала. Для эксперимента будем использовать сигнал из предыдущего опыта (см. п. 4.1.).

4.4. Домножайте сигнал на окна Хемминга, Ханна, Блэкмана длительностью 40 мс. Вычислите и отобразите спектры полученных сигналов.

Результат

На рис.6 представлен результат работы программы

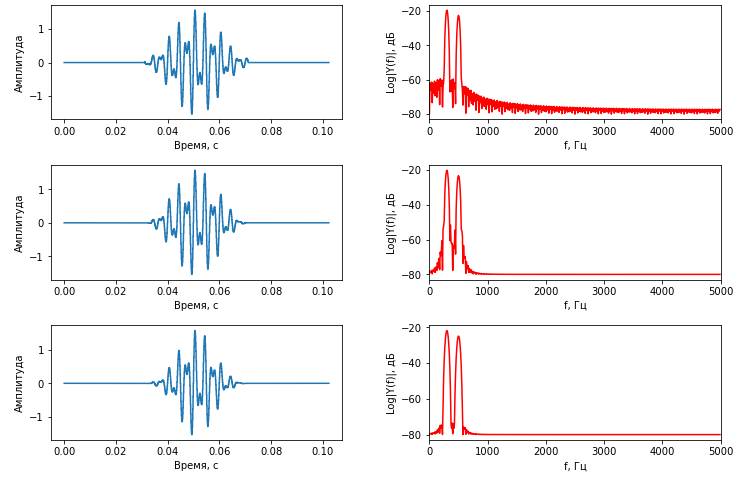


Рис. 6

Код программы

# Введите код для выполнения задания

#Сгенерируйте окна и выполните взвешивание сигнала

M=int(0.04\*fd) #Длительность окна в отсчетах

win=np.zeros(N)

win[int(N/2-M/2):int(N/2+M/2)]=np.hamming(M)

yw1=y\*win

win[int(N/2-M/2):int(N/2+M/2)]=np.hanning(M)

yw2=y\*win

win[int(N/2-M/2):int(N/2+M/2)]=np.blackman(M)

yw3=y\*win

#Вычислите спектры

Ycomplex1 = fft(yw1)/N #Вычисление БПФ и нормализация

Y1 = np.abs(Ycomplex1) #Вычислние спектра амплитуд

Ylog1 = 20\*np.log10(Y1+0.0001)

#Ylog1 =

#Ylog2 =

Ycomplex2 = fft(yw2)/N #Вычисление БПФ и нормализация

Y2 = np.abs(Ycomplex2) #Вычислние спектра амплитуд

Ylog2 = 20\*np.log10(Y2+0.0001) #Спектр амплитуд в логарифмическом масштабе

Ycomplex3 = fft(yw3)/N #Вычисление БПФ и нормализация

Y3 = np.abs(Ycomplex3) #Вычислние спектра амплитуд

Ylog3 = 20\*np.log10(Y3+0.0001) #Спектр амплитуд в логарифмическом масштабе

#Ylog3 =

#Отобразите сигналы и спектры

f, ax = plt.subplots(3, 2,figsize=(12, 8))

ax[0][0].step(t1, yw1)

ax[0][0].set\_xlabel('Время, с')

ax[0][0].set\_ylabel('Амплитуда')

ax[0][1].plot(frq[0:int(N/2)],Ylog1[0:int(N/2)],'r') #plotting the log spectrum

ax[0][1].set\_xlabel('f, Гц')

ax[0][1].set\_ylabel('Log|Y(f)|, дБ')

ax[0][1].set\_xlim([0,frq[int(N/2)]])

ax[1][0].step(t1, yw2)

ax[1][0].set\_xlabel('Время, с')

ax[1][0].set\_ylabel('Амплитуда')

ax[1][1].plot(frq[0:int(N/2)],Ylog2[0:int(N/2)],'r') #plotting the log spectrum

ax[1][1].set\_xlabel('f, Гц')

ax[1][1].set\_ylabel('Log|Y(f)|, дБ')

ax[1][1].set\_xlim([0,frq[int(N/2)]])

ax[2][0].step(t1, yw3)

ax[2][0].set\_xlabel('Время, с')

ax[2][0].set\_ylabel('Амплитуда')

ax[2][1].plot(frq[0:int(N/2)],Ylog3[0:int(N/2)],'r') #plotting the log spectrum

ax[2][1].set\_xlabel('f, Гц')

ax[2][1].set\_ylabel('Log|Y(f)|, дБ')

ax[2][1].set\_xlim([0,frq[int(N/2)]])

plt.subplots\_adjust(wspace=0.3, hspace=0.4)

Чем короче интервал вычисления спектра, тем больше влияют спектры прямоугольных сигналов окон и больше эффект просачивания.

# Часть 5. Исследование спектра временных окон разной формы

5.1. Сгенерируйте импульсы (прямоугольный, Хемминга, Ханна, Блэкмана):

* частота дискретизации 10000 Гц;
* длительность сигнала 1024 отсчета;
* длительность импульсов 10 мс.

5.2. Вычислите и отобразите спектры сгенерированных импульсов.

5.3. Проведите измерения уровня боковых лепестков и ширины главного лепестка полученных спектров. В отчете результаты измерений оформите в виде таблицы

5.4. Сравните результаты измерений в таблице с результатами анализа спектров в п. 4.4. и сделайте вывод о влиянии уровня боковых лепестков на величину эффекта просачивания



Результат

На рис.7 представлен результат работы программы

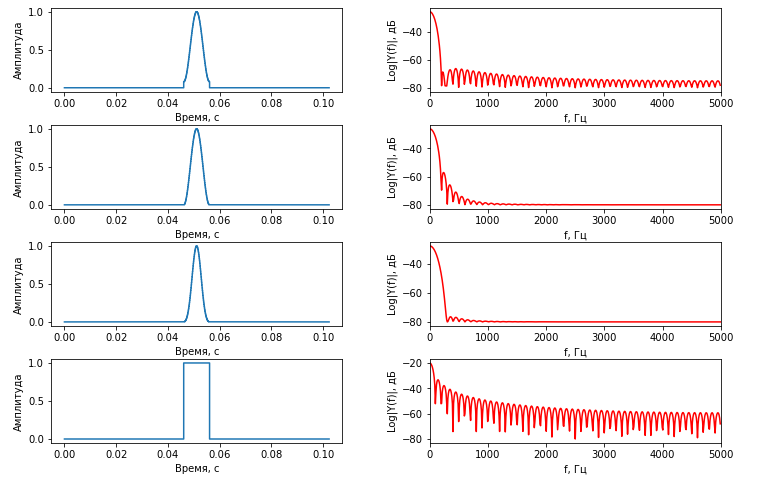


Рис. 7

Код программы для части 5

fd = 10000

N = 1024

M=int(0.01\*fd) #Длительность окна в отсчетах

win=np.zeros(N)

win[int(N/2-M/2):int(N/2+M/2)]=np.hamming(M)

yw1=win

win=np.zeros(N)

win[int(N/2-M/2):int(N/2+M/2)]=np.hanning(M)

yw2=win

win=np.zeros(N)

win[int(N/2-M/2):int(N/2+M/2)]=np.blackman(M)

yw3=win

signal\_imp1, t1=imp\_generator(fd, N, 0.01)

yw4=signal\_imp1

k = np.arange(N) # Номера отсчетов спектра

T = float(N)/fd

frq = k/T # Привязка отсчетов спектра к частотам (полный диапазон частот)

Ylog1 = 20\*np.log10(np.abs(fft(yw1)/N)+ 0.0001) #Вычисление БПФ и нормализация

Ylog2 = 20\*np.log10(np.abs(fft(yw2)/N)+ 0.0001) #Вычисление БПФ и нормализация

Ylog3 = 20\*np.log10(np.abs(fft(yw3)/N)+ 0.0001) #Вычисление БПФ и нормализация

Ylog4 = 20\*np.log10(np.abs(fft(yw4)/N)+ 0.0001) #Вычисление БПФ и нормализация

#Отобразите сигналы и спектры

f, ax = plt.subplots(4, 2,figsize=(12, 8))

ax[0][0].step(t1, yw1)

ax[0][0].set\_xlabel('Время, с')

ax[0][0].set\_ylabel('Амплитуда')

ax[0][1].plot(frq[0:int(N/2)],Ylog1[0:int(N/2)],'r') #plotting the log spectrum

ax[0][1].set\_xlabel('f, Гц')

ax[0][1].set\_ylabel('Log|Y(f)|, дБ')

ax[0][1].set\_xlim([0,frq[int(N/2)]])

ax[1][0].step(t1, yw2)

ax[1][0].set\_xlabel('Время, с')

ax[1][0].set\_ylabel('Амплитуда')

ax[1][1].plot(frq[0:int(N/2)],Ylog2[0:int(N/2)],'r') #plotting the log spectrum

ax[1][1].set\_xlabel('f, Гц')

ax[1][1].set\_ylabel('Log|Y(f)|, дБ')

ax[1][1].set\_xlim([0,frq[int(N/2)]])

ax[2][0].step(t1, yw3)

ax[2][0].set\_xlabel('Время, с')

ax[2][0].set\_ylabel('Амплитуда')

ax[2][1].plot(frq[0:int(N/2)],Ylog3[0:int(N/2)],'r') #plotting the log spectrum

ax[2][1].set\_xlabel('f, Гц')

ax[2][1].set\_ylabel('Log|Y(f)|, дБ')

ax[2][1].set\_xlim([0,frq[int(N/2)]])

ax[3][0].step(t1, yw4)

ax[3][0].set\_xlabel('Время, с')

ax[3][0].set\_ylabel('Амплитуда')

ax[3][1].plot(frq[0:int(N/2)],Ylog4[0:int(N/2)],'r') #plotting the log spectrum

ax[3][1].set\_xlabel('f, Гц')

ax[3][1].set\_ylabel('Log|Y(f)|, дБ')

ax[3][1].set\_xlim([0,frq[int(N/2)]])

plt.subplots\_adjust(wspace=0.3, hspace=0.4)

Чем меньше уровень боковых лепестков, тем меньше эффект просачивания.