

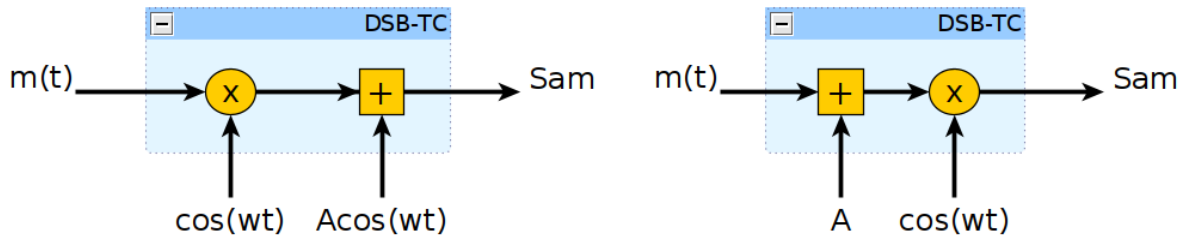
Модулятор

Double Sideband Transmitted Carrier (AM-DSB-TC)

Для запуска скриптов необходимо корневую папку репозитория сделать рабочей папкой **Matlab!**

1. Введение

Амплитудная модуляция с передаваемой несущей осуществляется с помощью умножения информационного сообщения на гармонический сигнал, который называют несущей (carrier). После этого к сигналу добавляется несущая с заданной амплитудой. Другой вариант - это добавить к сигналу константу и результат суммы умножить на несущую. Обе схемы модулятора представлены ниже:



С помощью формул данный модулятор можно записать в следующем виде:

$$s_{am}(t) = m(t) \cdot \cos(2\pi f_c t) + A_c \cos(2\pi f_c t) = (m(t) + A_c) \cdot \cos(2\pi f_c t), \quad (1)$$

где $m(t)$ - информационное сообщение, A_c - амплитуда несущей, f_c - частота несущей, $s_{am}(t)$ - модулированный сигнал.

Рассмотрим, что происходит в частотной области. Будем считать, что спектр информационного сообщения равен

$$m(t) \Longleftrightarrow M(f).$$

Несущая представляет из себя гармонический сигнал, спектр которого состоит из двух дельта-функций:

$$A_c \cos(2\pi f_c t) \Longleftrightarrow \frac{A_c}{2} [\delta(f + f_c) + \delta(f - f_c)].$$

Произведение сигналов во временной области соответствует свертке их спектров в частотной области. Свертка с дельта-функцией соответствует переносу спектра по частоте. Таким образом, модулированный сигнал будет состоять из двух копий спектров информационного сообщения, сдвинутых на частоты f_c и $-f_c$. Также к этому спектру следует добавить спектр несущей, который состоит из двух дельта функций:

$$m(t) \cdot \cos(2\pi f_c t) + A_c \cos(2\pi f_c t) \iff \frac{1}{2} M(f + f_c) + \frac{1}{2} M(f - f_c) + \frac{A_c}{2} [\delta(f + f_c) + \delta(f - f_c)]. \quad (2)$$

Далее рассмотрим несколько частных случаев: однотоновую модуляцию, модуляцию несколькими тонами и модуляцию аудиосигналом.

2. Однотоновая модуляция

В случае однотоновой модуляции информационное сообщение представляет из себя гармонический сигнал вида:

$$m(t) = A_m \cos(2\pi f_m t),$$

где A_m - амплитуда тона, f_m - частота в герцах.

Подставим этот гармонический сингнал в формулу (1) и получим:

$$s_{am}(t) = A_m \cos(2\pi f_m t) \cdot \cos(2\pi f_c t) + A_c \cos(2\pi f_c t).$$

Это выражение можно преобразовать к более удобному виду:

$$s_{am}(t) = (A_m \cos(2\pi f_m t) + A_c) \cdot \cos(2\pi f_c t) = A_c \cdot (1 + M \cos(2\pi f_m t)) \cdot \cos(2\pi f_c t), \quad (3)$$

где $M = A_m/A_c$ - индекс (глубина) модуляции.

Ниже представлен скрипт для получения графика сигнала после однотоновой модуляции:

```
clc; clear; close all;

FrameSize = 1011;           % размер обрабатываемой за один раз пачки данных
Fs = 500e3;                 % тактовая частота (Hz)
Ac = 1;                     % амплитуда несущей
M = 0.4;                    % индекс модуляции

% генератор гармонических сигналов
% частота тона: 1 kHz
% частота несущей: 50 kHz
MessageAndCarrierGen = dsp.SineWave(...
    'SampleRate', Fs, ...
    'SamplesPerFrame', FrameSize, ...
    'Frequency', [50e3 1e3], ...
    'Amplitude', [Ac Ac*M] ...
);

% объект для отрисовки графиков
Plotter = dsp.ArrayPlot(...
    'PlotType', 'Line', ...
    'YLimits', [-2.3, 2.3], ...
    'SampleIncrement', 1/Fs, ...
    'XLabel', 'Time', ...
    'YLabel', 'Amplitude', ...
);
```

```

'ChannelNames', {'AM Signal', 'Message', 'AM Signal Envelope'} ...
);

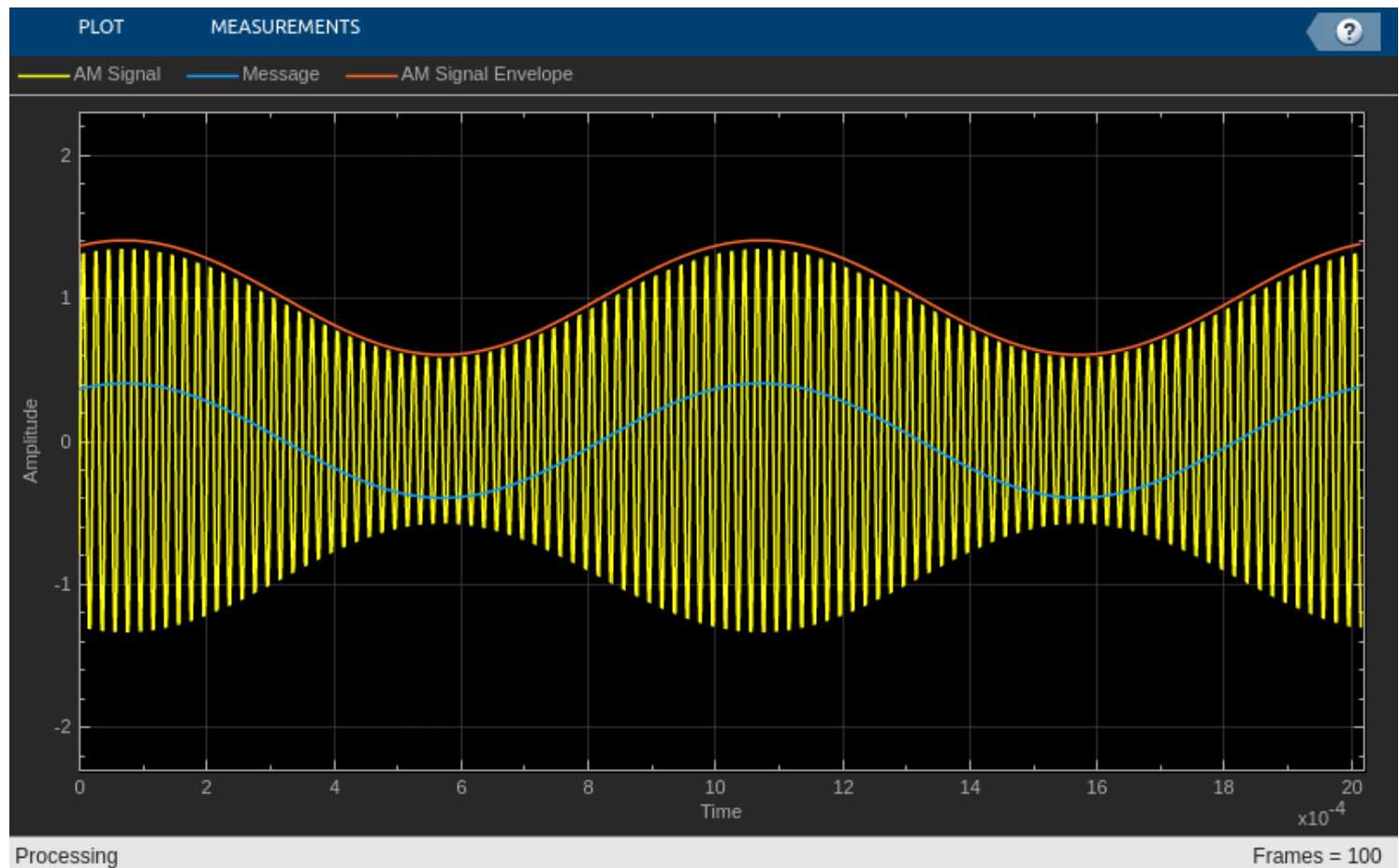
% запуск симуляции
for i = 1:100
    % формирование тонального сигнала и несущей
    SineWaves = MessageAndCarrierGen();
    Carrier = SineWaves(:,1);
    Message = SineWaves(:,2);

    % амплитудная модуляция
    AmSignal = Message.*Carrier + Carrier;

    % вывод результатов на график
    Plotter([AmSignal, Message, Message + Ac])

    % задержка в 0.1 секунды для лучшей визуализации
    pause(0.1)
end

```



Можно увидеть, что при $M \leq 1$ огибающая несущей (красная линия) соответствует форме информационного сообщения. Таким образом, информационное сообщение можно восстанавливать с помощью детектора огибающей. При $M > 1$ огибающая перестает повторять

форму сообщения. Такой режим называется перемодуляцией (overmodulation). Такой сигнал можно принимать только когерентным способом.

Теперь рассмотрим частотную область. Применив в формуле (3) правило произведения косинусов, преобразуем результат к следующему виду:

$$s_{\text{am}}(t) = \frac{A_c M}{2} [\cos(2\pi(f_m + f_c)t) + \cos(2\pi(f_m - f_c)t)] + A_c \cos(2\pi f_c t) .$$

Таким образом, результат однотоновой модуляции равен сумме двух гармонических сигналов, частоты которых равны $f_m + f_c$ и $f_m - f_c$ Гц. В частотной области это соответствует спектру:

$$s_{\text{am}}(t) \Longleftrightarrow \frac{A_c M}{4} [\delta(f + f_c + f_m) + \delta(f + f_c - f_m) + \delta(f - f_c + f_m) + \delta(f - f_c - f_m)] + \frac{A_c}{2} [\delta(f + f_c) + \delta(f - f_c)].$$

Этот результат можно получить, если учесть формулу (2) и, что спектр информационного сигнала равен:

$$A_m \cos(2\pi f_m t) \Longleftrightarrow \frac{A_m}{2} [\delta(f + f_m) + \delta(f - f_m)].$$

```
clc; clear; close all;

FrameSize = 5000;    % размер обрабатываемой за один раз пачки данных
Fs = 100e3;          % тактовая частота (Hz)
Ac = 1;              % амплитуда несущей
M = 0.7;             % индекс модуляции

% генератор гармонических сигналов
% частота тона: 5 kHz
% частота несущей: 35 kHz
MessageAndCarrierGen = dsp.SineWave(...
    'SampleRate',Fs,...
    'SamplesPerFrame',FrameSize,...
    'Frequency',[35e3 5e3],...
    'Amplitude',[Ac Ac*M] ...
);

% объект для вычисления спектра
SpecEstimator = dsp.SpectrumEstimator(...
    'PowerUnits','dBm', ...
    'FrequencyRange', 'centered', ...
    'SampleRate', Fs ...
);

% объект для отрисовки графиков
Plotter = dsp.ArrayPlot(...
    'PlotType','Line', ...
    'XOffset', -Fs/2, ...
```

```

'YLimits', [-30, 30], ...
'XLabel', 'Frequency (Hz)', ...
'YLabel', 'Amplitude (dBm)', ...
'ChannelNames', {'Message', 'AM Signal'}, ...
'SampleIncrement', Fs/FrameSize ...
);

% запуск симуляции
for i = 1:100
    % формирование тонального сигнала и несущей
    SineWaves = MessageAndCarrierGen();
    Carrier = SineWaves(:,1);
    Message = SineWaves(:,2);

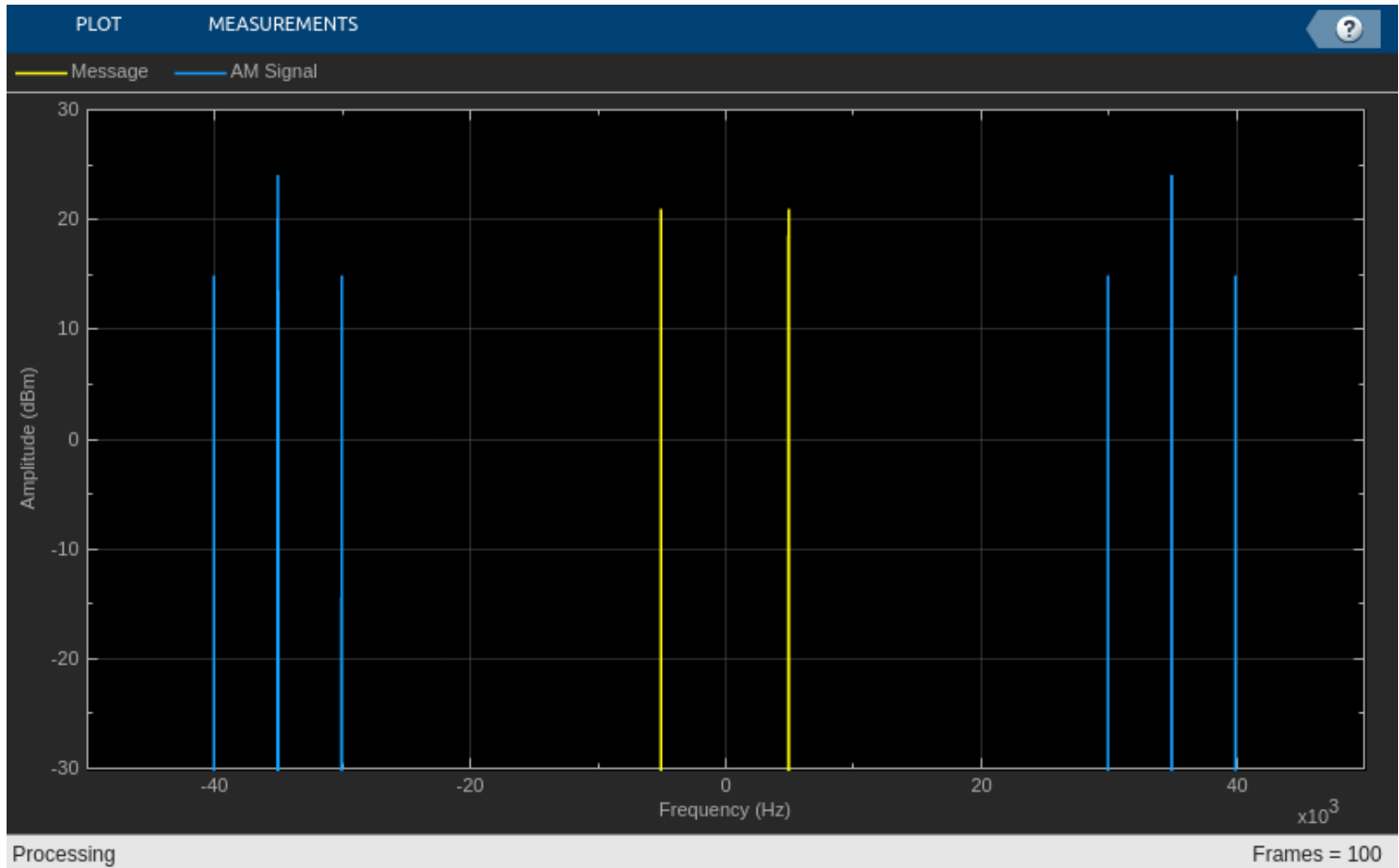
    % амплитудная модуляция
    AmSignal = Message.*Carrier + Carrier;

    % вычисление спектров
    Spectrums = SpecEstimator([Message, AmSignal]);

    % вывод результатов на график
    Plotter(Spectrums)

    % задержка в 0.1 секунды для лучшей визуализации
    pause(0.1)
end

```



На графике представлен спектр информационного (желтый) и модулированного (синий) сигналов. Информационное сообщение гармоническое, поэтому ему соответствуют гармоники на частотах f_m . Спектр модулированного сигнала для однотонового случая состоит из четырех дельта-функций, которые относятся к сообщению и двух дельта-функций, соответствующих несущей.

3. Модуляция несколькими тонами

Разобравшись с однотоновой модуляцией, легко распространить результаты на случай модуляции несколькими тонами. Пусть информационный сигнал представляет из себя сумму гармонических сигналов:

$$m(t) = \sum_{i=1}^N A_i \cos(2\pi f_i t),$$

где A_i - амплитуда i -ой гармоники, f_i - частота i -ой гармоники, N - число гармоник.

Огибающая сигнала будет повторять форму информационного сообщения при условии $A_c + m_{\min} \geq 0$, где m_{\min} - минимальное значение $m(t)$. В случае, если в модуляции участвуют три тона, график сигнала будет иметь вид:

```
clc; clear; close all;
```

```
FrameSize = 1011; % размер обрабатываемой за один раз пачки данных
```

```

Fs = 500e3;           % тактовая частота (Hz)

% генератор информационного сигнала из трех тонов
% частота тонов: 1, 4 и 7 kHz
% амплитуды тонов: 0.1, 0.5 и 0.7
Message = dsp.SineWave(...
    'SampleRate', Fs, ...
    'SamplesPerFrame', FrameSize, ...
    'Frequency', [1e3 4e3 7e3], ...
    'Amplitude', [0.1 0.5 0.7] ...
);

% Минимальное значение сообщения равно -0.1-0.5-0.7 = -1.3
% Чтобы огибающая повторяла форму сообщения, амплитуда несущей должна быть
% больше, чем 1.3
Ac = 1.7;

% генератор несущей
% частота несущей: 100 kHz
Carrier = dsp.SineWave(...
    'SampleRate', Fs, ...
    'SamplesPerFrame', FrameSize, ...
    'Frequency', 100e3, ...
    'Amplitude', 1 ...
);

% объект для отрисовки графиков
Plotter = dsp.ArrayPlot(...
    'PlotType', 'Line', ...
    'YLimits', [-3, 3], ...
    'SampleIncrement', 1/Fs, ...
    'XLabel', 'Time', ...
    'YLabel', 'Amplitude', ...
    'ChannelNames', {'AM Signal', 'Message', 'Envelope'} ...
);

% запуск симуляции
for i = 1:100
    % формирование информационного сигнала
    MessageData = Message();
    MessageData = MessageData(:,1) + MessageData(:,2) + MessageData(:,3);

    % формирование несущей
    CarrierData = Carrier();

    % амплитудная модуляция
    AmSignal = (Ac + MessageData) .* CarrierData;

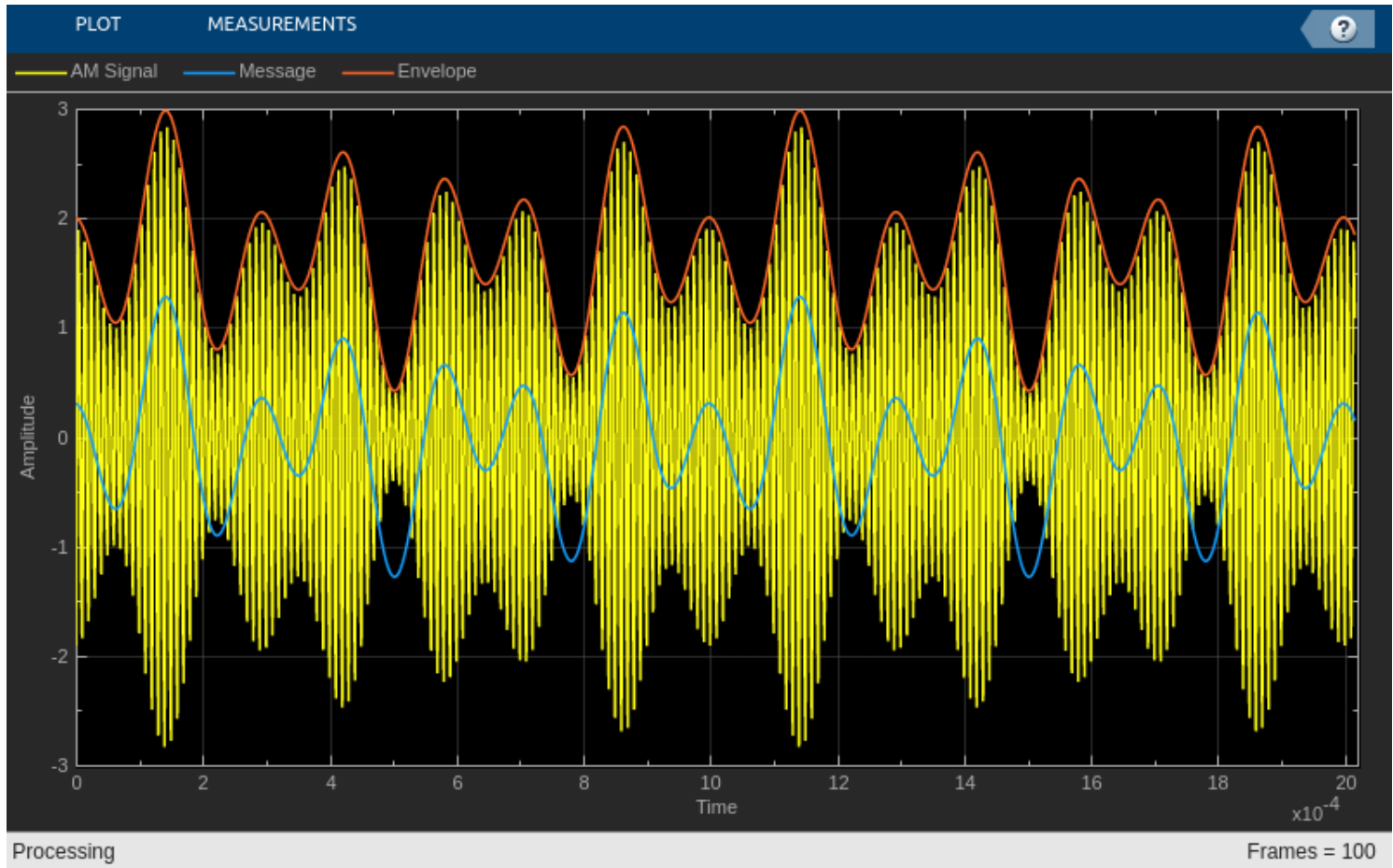
    % вывод результатов на график
    Plotter([AmSignal, MessageData, MessageData + Ac])

```

```
% задержка в 0.1 секунды для лучшей визуализации
```

```
pause(0.1)
```

```
end
```



Рассмотрим, что происходит в частотной области. В случае модуляции несколькими тонами модулированный сигнал примет следующий вид:

$$s_{\text{am}}(t) = A_c \cos(2\pi f_c t) + \sum_{i=1}^N A_i \cos(2\pi f_i t) \cdot \cos(2\pi f_c t) = A_c \cos(2\pi f_c t) + \frac{1}{2} \sum_{i=1}^N A_i [\cos(2\pi(f_i + f_c)t) + \cos(2\pi(f_i - f_c)t)].$$

То есть после модуляции каждому тону будут соответствовать две гармоники на частотах $f_i + f_c$ и $f_i - f_c$. Можно сказать по другому: каждый отдельный тон информационного сигнала будет сдвинут по частоте на f_c . Также в сигнале будет присутствовать тон соответствующий несущей. Построим спектры сигналов для случая модуляции тремя тонами.

```
clc; clear; close all;
```

```
FrameSize = 5000; % размер обрабатываемой за один раз пачки данных
```

```
Fs = 200e3; % тактовая частота (Hz)
```

```
% генератор информационного сигнала из трех тонов
```

```
% частота тонов: 2, 4 и 7 kHz
```



```

% амплитуды тонов: 0.1, 0.5 и 0.7
Message = dsp.SineWave(...
    'SampleRate', Fs,...
    'SamplesPerFrame', FrameSize,...
    'Frequency', [2e3 4e3 7e3],...
    'Amplitude', [0.1 0.5 0.7]);

% Минимальное значение сообщения равно -0.1-0.5-0.7 = -1.3
% Чтобы огибающая повторяла форму сообщения, амплитуда несущей должна быть
% больше, чем 1.3
Ac = 1.7;

% генератор несущей
% частота несущей: 50 kHz
Carrier = dsp.SineWave(...
    'SampleRate', Fs,...
    'SamplesPerFrame', FrameSize,...
    'Frequency', 50e3,...
    'Amplitude', Ac ...
);

% объект для вычисления спектра
SpecEstimator = dsp.SpectrumEstimator(...
    'PowerUnits', 'dBm',...
    'FrequencyRange', 'centered',...
    'SampleRate', Fs ...
);

% объект для отрисовки графиков
Plotter = dsp.ArrayPlot(...
    'PlotType', 'Line', ...
    'XOffset', -Fs/2, ...
    'YLimits', [-20, 35], ...
    'XLabel', 'Frequency (Hz)', ...
    'YLabel', 'Amplitude (dBm)', ...
    'ChannelNames', {'Message', 'AM Signal'}, ...
    'SampleIncrement', Fs/FrameSize ...
);

% запуск симуляции
for i = 1:100
    % формирование информационного сигнала
    MessageData = Message();
    MessageData = MessageData(:,1) + MessageData(:,2) + MessageData(:,3);

    % формирование несущей
    CarrierData = Carrier();

    % амплитудная модуляция
    AmSignal = (Ac + MessageData) .* CarrierData;

```

```
% ВЫЧИСЛЕНИЕ СПЕКТРОВ
```

```
SpectrumData = SpecEstimator([MessageData, AmSignal]);
```

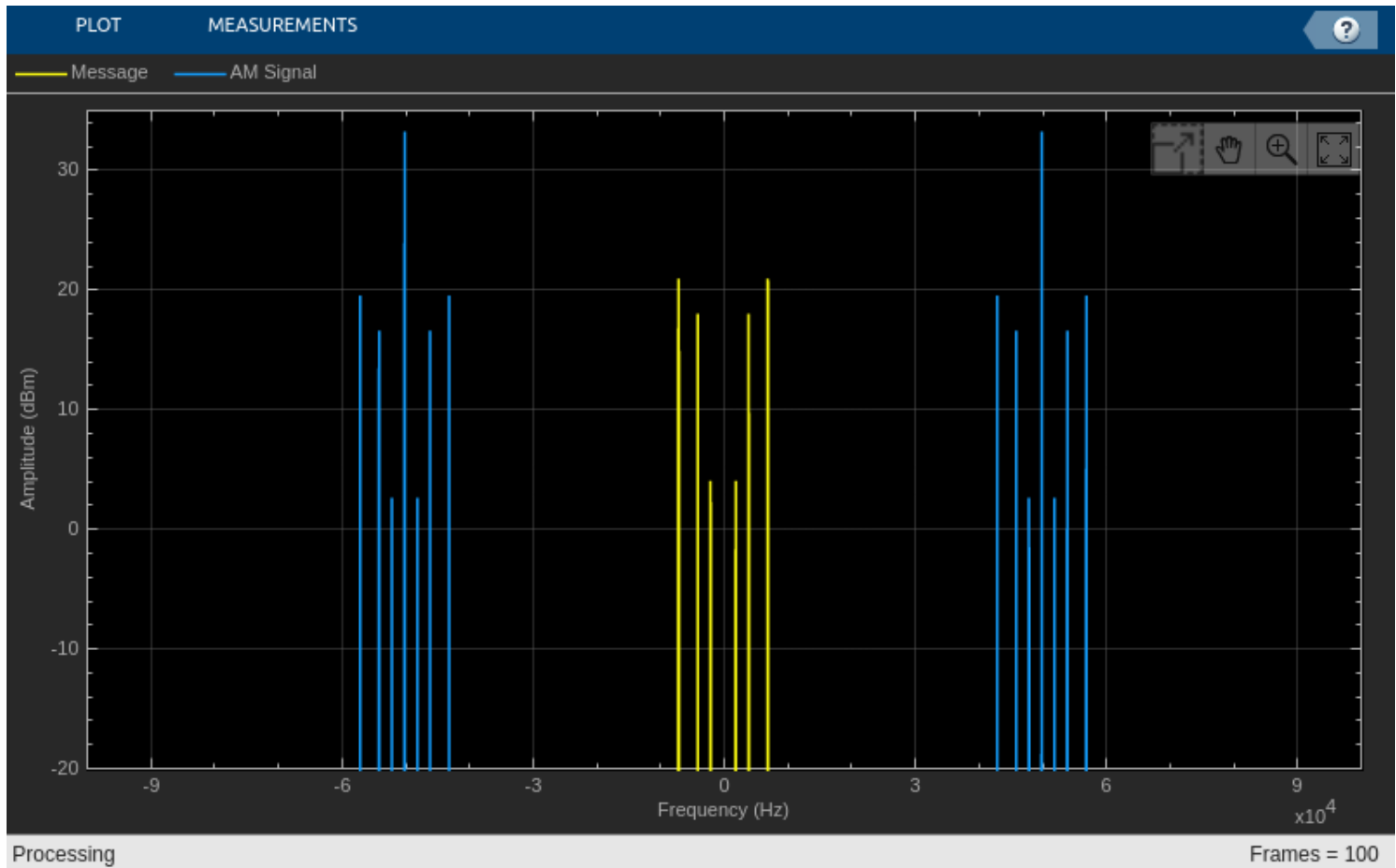
```
% ВЫВОД РЕЗУЛЬТАТОВ НА ГРАФИК
```

```
Plotter(SpectrumData)
```

```
% ЗАДЕРЖКА В 0.1 СЕКУНДЫ ДЛЯ ЛУЧШЕЙ ВИЗУАЛИЗАЦИИ
```

```
pause(0.1)
```

```
end
```



Можно увидеть, что спектр модулированного сигнала (синий) соответствует спектру информационного сообщения (желтый), сдвинутому на частоту несущей. Также в спектре сигнала присутствует тон на несущей частоте.

4. Модуляция звуковым сигналом

Рассмотрим, как будет выглядеть спектр сигнала после модуляции, если информационное сообщение является аудиосигналом. В файле `Audio_Source.wav` записано звуковое сообщение с частотой дискретизации 44.1 kHz. Частота несущей будет равна 60 kHz. Чтобы избежать наложения спектров, выберем итоговую частоту дискретизации модулированного сигнала в 5 раз больше частоты аудиосигнала, то есть 5×44.1 kHz. Это потребует провести интерполяцию для увеличения частоты дискретизации информационного сообщения.

```

clc; clear; close all;

AudioFrameSize = 1000; % количество отсчетов аудиофайла, получаемых за один раз
FramesNumber = 100; % число обрабатываемых пачек данных
RateRatio = 5; % коэффициент увеличения частоты дискретизации
Ac = 2; % амплитуда несущей

% объект для считывания отсчетов аудиофайла
AudioReader = dsp.AudioFileReader(...
    'wav/Audio_Source.wav', ...
    'SamplesPerFrame', AudioFrameSize...
);

% дополнительные расчеты
AudioFs = AudioReader.SampleRate; % получаем частоту дискретизации аудиосо
SignalFs = AudioFs * RateRatio; % частота дискретизации модулированного
SignalFrameSize = AudioFrameSize * RateRatio; % количество отсчетов Ам-сигнала, получа

% генератор несущей с частотой 60 kHz
Carrier = dsp.SineWave(...
    'SampleRate', SignalFs,...
    'SamplesPerFrame', SignalFrameSize,...
    'Frequency', 60e3,...
    'Amplitude', Ac);

% интерполятор
Upsampler = dsp.SampleRateConverter(...
    'Bandwidth', 40e3, ...
    'InputSampleRate', AudioFs, ...
    'OutputSampleRate', SignalFs ...
);

% объект для вычисления спектра
SpecEstimator = dsp.SpectrumEstimator(...
    'PowerUnits', 'dBm', ...
    'FrequencyRange', 'centered', ...
    'SampleRate', SignalFs);

% объект для отрисовки графиков
Plotter = dsp.ArrayPlot(...
    'PlotType', 'Line', ...
    'XOffset', -SignalFs/2, ...
    'YLimits', [-90, 35], ...
    'XLabel', 'Frequency (Hz)', ...
    'YLabel', 'Amplitude (dBm)', ...
    'ChannelNames', {'Message', 'AM Signal'}, ...
    'SampleIncrement', SignalFs/SignalFrameSize ...
);

% запуск симуляции

```

```

for i = 1:FramesNumber
    % считывание отсчетов аудиосообщения и выделение одного канала из
    % стерео сигнала
    AudioData = AudioReader();
    AudioData = AudioData(:,1);

    % увеличение частоты дискретизации аудиосообщения
    MessageData = Upsampler(AudioData);

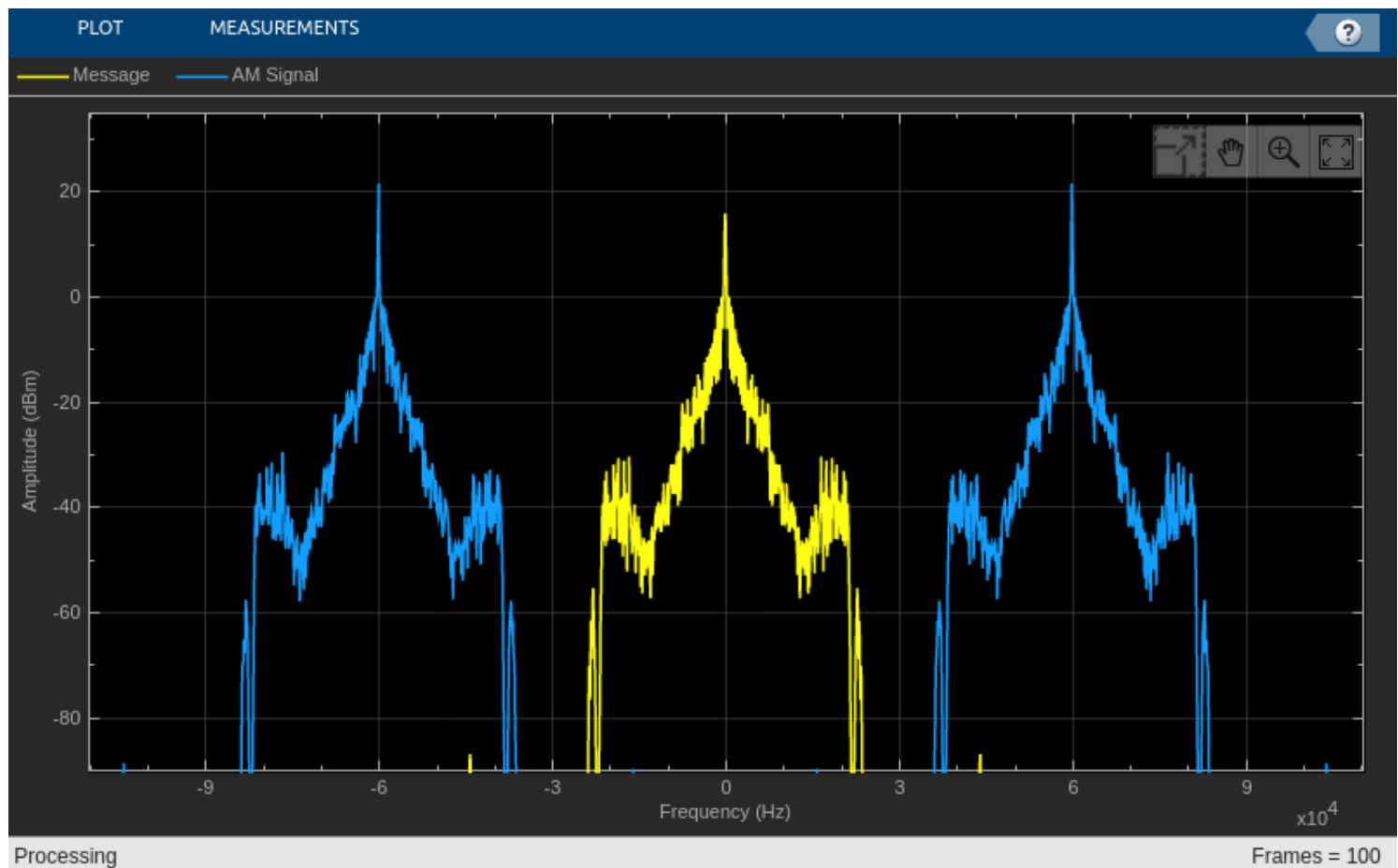
    % амплитудная модуляция
    AmSignal = (Ac + MessageData) .* Carrier();

    % вычисление спектров
    SpectrumData = SpecEstimator([MessageData, AmSignal]);

    % вывод результатов на график
    Plotter(SpectrumData)

    % задержка в 0.1 секунды для лучшей визуализации
    pause(0.1)
end

```



Можно увидеть, что спектр информационного сообщения перенесен на частоту несущей.

5. Готовые решения Matlab

В среде Matlab есть готовая функция для выполнения амплитудной модуляции: `ammod()`. Проверим ее работу. Функция принимает на вход информационный сигнал, частоту несущей, частоту дискретизации, начальную фазу несущей и амплитуду несущей. Основная часть скрипта совпадает со скриптом из предыдущего раздела.

```
clc; clear; close all;

AudioFrameSize = 1000; % количество отсчетов аудиофайла, получаемых за один раз
FramesNumber = 100;    % число обрабатываемых пачек данных
RateRatio = 5;         % коэффициент увеличения частоты дискретизации

Ac = 2;                % амплитуда несущей
CarrierPhase = 0;      % начальная фаза несущей
CarrierFreq = 60e3;     % частота несущей

% объект для считывания отсчетов аудиофайла
AudioReader = dsp.AudioFileReader(...
    'wav/Audio_Source.wav', ...
    'SamplesPerFrame', AudioFrameSize...
);

% дополнительные расчеты
AudioFs = AudioReader.SampleRate; % получаем частоту дискретизации аудиосигнала
SignalFs = AudioFs * RateRatio;   % частота дискретизации модулированного сигнала
SignalFrameSize = AudioFrameSize * RateRatio; % количество отсчетов Ам-сигнала, получаемых за один раз

% интерполятор
Upsampler = dsp.SampleRateConverter(...
    'Bandwidth', 40e3, ...
    'InputSampleRate', AudioFs, ...
    'OutputSampleRate', SignalFs ...
);

% объект для вычисления спектра
SpecEstimator = dsp.SpectrumEstimator(...
    'PowerUnits', 'dBm', ...
    'FrequencyRange', 'centered', ...
    'SampleRate', SignalFs);

% объект для отрисовки графиков
Plotter = dsp.ArrayPlot(...
    'PlotType', 'Line', ...
    'XOffset', -SignalFs/2, ...
    'YLimits', [-90, 35], ...
    'XLabel', 'Frequency (Hz)', ...
    'YLabel', 'Amplitude (dBm)', ...
    'ChannelNames', {'Message', 'AM Signal'}, ...
    'SampleIncrement', SignalFs/SignalFrameSize ...
);
```

```

);

% запуск симуляции
for i = 1:FramesNumber
    % считывание отсчетов аудиосообщения и выделение одного канала из
    % стерео сигнала
    AudioData = AudioReader();
    AudioData = AudioData(:,1);

    % увеличение частоты дискретизации аудиосообщения
    MessageData = Upsampler(AudioData);

    % амплитудная модуляция
    AmSignal = ammod(MessageData, CarrierFreq, SignalFs, CarrierPhase, Ac);

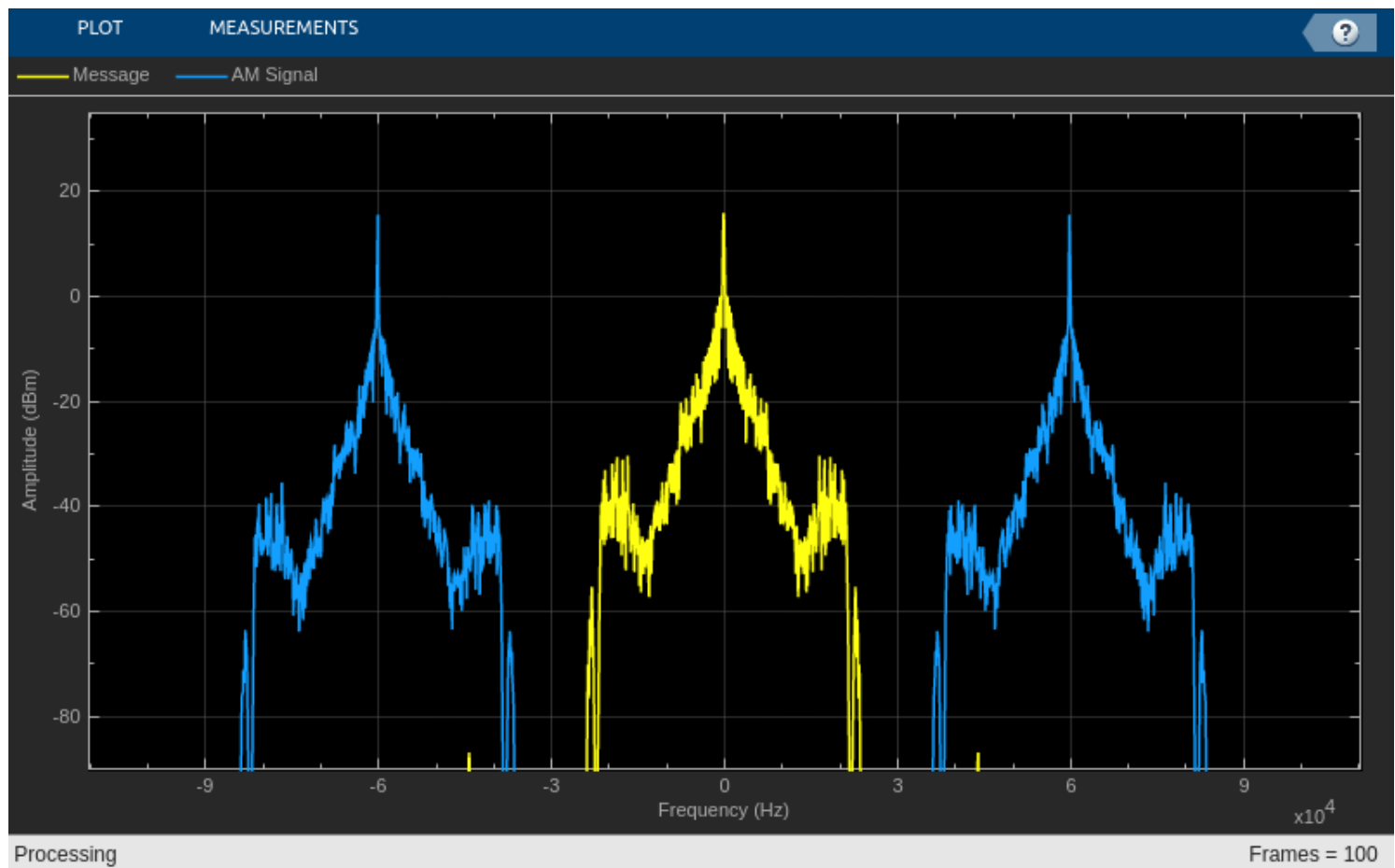
    % вычисление новой фазы несущей
    CarrierPhase = CarrierPhase + 2*pi*CarrierFreq/SignalFs*SignalFrameSize;

    % вычисление спектров
    SpectrumData = SpecEstimator([MessageData, AmSignal]);

    % вывод результатов на график
    Plotter(SpectrumData)

    % задержка в 0.1 секунды для лучшей визуализации
    pause(0.1)
end

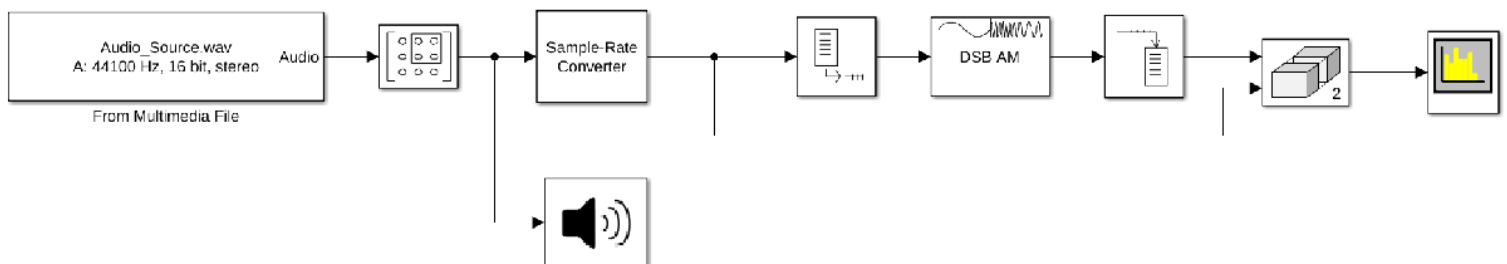
```

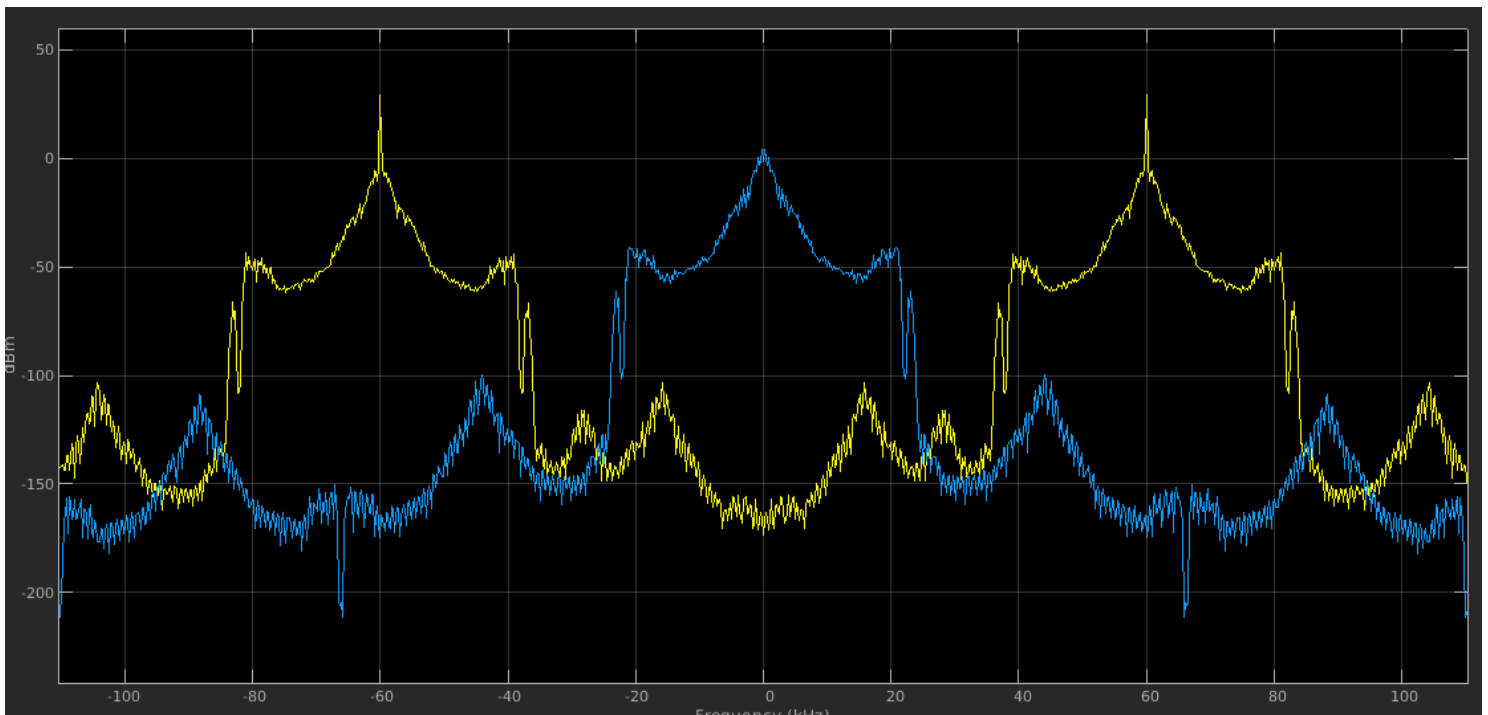


Спектр модулированного сигнала совпадает со спектром из предыдущего пункта.

6. Модуляция DSB-TC в Simulink

В файле DSB_TC_Transmitter.slx представлена Simulink-модель модуляции аудиосообщением. Как и в предыдущих скриптах в файле Audio_Source.wav записано звуковое сообщение с частотой дискретизации 44.1 kHz. Частота несущей равна 60 kHz. Перед модуляцией частота дискретизации увеличивается в 5 раз. Блок DSB AM принимает и выдает данные только по одному отсчету. Поэтому до этого блока данные преобразуются из вектора в отсчеты (Unbuffer), а после него опять собираются в вектор (Buffer). Схема модели и спектры сигналов представлены ниже:





7. Преимущества и недостатки **DSB-SC**

Преимущества:

- простота модулятора;
- при условии $A_c + m_{\min} \geq 0$ можно принимать некогерентным способом;
- несущая передается с сигналом, упрощает когерентный приемник.

Недостатки:

- лишняя мощность затрачивается на передачу несущей;
- полоса модулированного сигнала в 2 раза больше полосы информационного сообщения.

Литература:

1. B. P. Lathi Modern Digital and Analog Communication Systems
2. R. Stewart, K. Barlee, D. Atkinson, L. Crockett Software Defined Radio using MATLAB® & Simulink and the RTL-SDR