Модулятор

Single Sideband Suppressed Carrier

(AM-SSB-SC)

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

1. Введение

Одним из недостатков методов обычной амплитудной модуляции является удвоение требуемой полосы передачи по сравнению с полосой информационного сообщения. Это удвоение излишне, так как спектр АМ-сигнала симметричен относительно частоты несущей, поэтому половина спектра уже несет в себе всю информацию о передаваемом сообщении. Это можно увидеть на спектраграммах, представленных в DSB TC Transmitter и DSB SC Transmitter. Амплитудная модуляция, которая не приводит к увеличению полосы сигнала, называется однополосной модуляцией (SSB - Sibgle Side Band). Обычная модуляция по этой причине иногда называется модуляция с удвоенной полосой (DSB - Double Side Band). Рассмотрим устройство модулятора формирующего однополосный сигнал.

2. Однотональная модуляция с использованием полосового фильтра

Начнем с простейшего случая однотональной модуляции, при котором информационное сообщение представляет из себя гармонический сигнал вида:

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

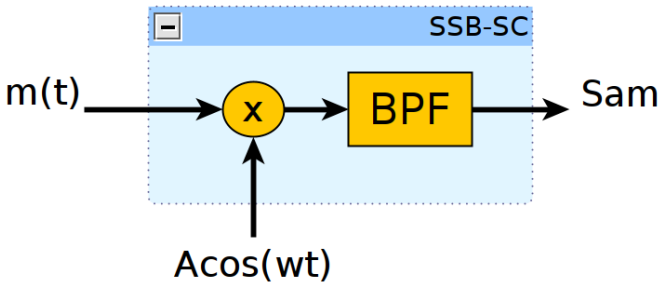
где A_m - амплитуда тона, f_m - частота в герцах.

Если мы просто смешаем это сообщение с несущей, то получим две гармоники на несущей частоте, хотя само сообщение состоит всего из одного тона. Это можно увидеть, если умножить сообщение на несущую и затем воспользоваться формулой произведения косинусов:

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

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

Один из простейший способов формирования однополосного сигнала заключается в использовании полосового фильтра для удаления одной половины спектра. Схема передающего устройства имеет вид:



Если удаляется верхняя половина, то сигнал называется Lower Side Band (LSB). В случае однотональной модуляции LSB сигнал будет иметь вид:

$$s_{\mathrm{LSB}}(t) = \frac{A_m A_c}{2}\cos(2\pi(f_m - f_c)t).$$

Если удаляется нижняя половина спектра, то сигнал называется Upper Side Band (USB). В случае однотональной модуляции USB сигнал будет иметь вид:

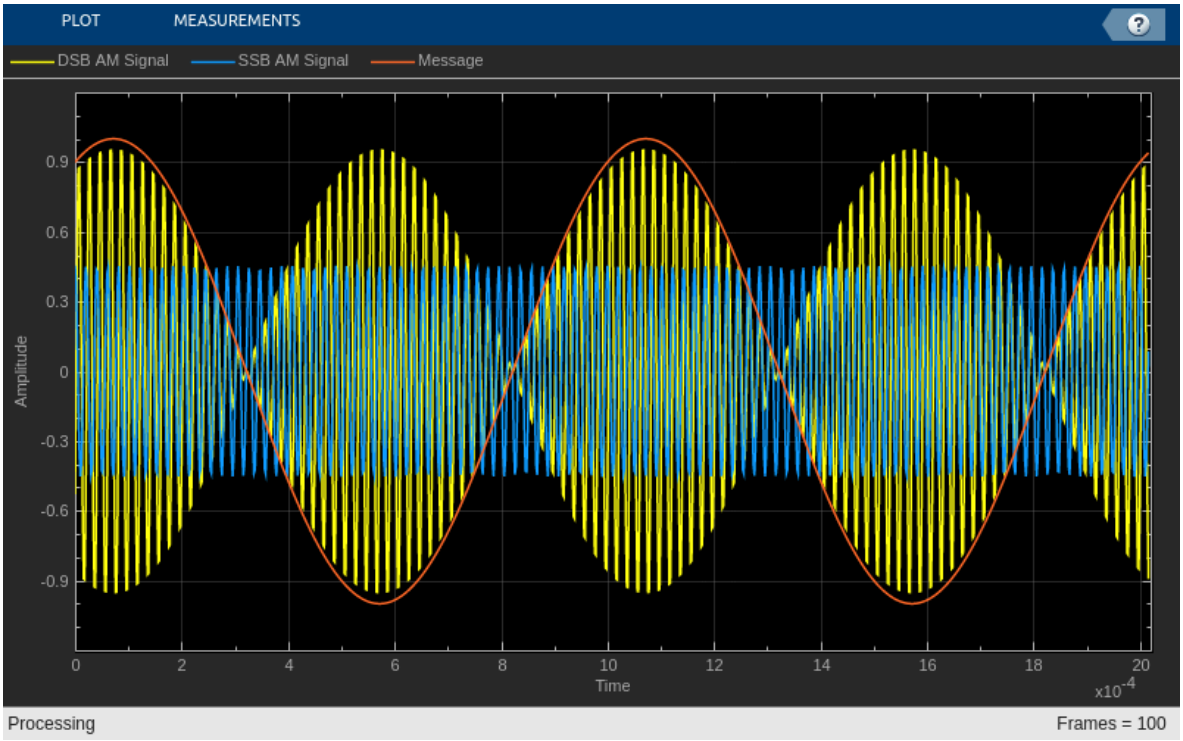
$$s_{\mathrm{USB}}(t) = \frac{A_m A_c}{2}\cos(2\pi(f_m + f_c)t).$$

Также заметим, что в модулированном сигнале в явном виде отсутстует гармоника на частоте несущей, поэтому такой сигнал также называют сигнал с подавленной несущей (Single Side Band Suppressed Carrier).

Ниже представлен скрипт для получения временной диаграммы одноплосного сигнала для случая однотональной модуляции. С помощью переменной ModulationMethod можно выбрать, полосовой фильтр, который сформирует USB или LSB сигнал:

```
clc; clear; close all;
addpath('matlab/SSB_SC');

FrameSize = 1011;   % размер обрабатываемой за один раз пачки данных
Fs = 500e3;         % тактовая частота (Hz)

% выбор метода модуляции
ModulationMethod = "LSB";

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

% расчет коэффициентов и создание полосового фильтра
if (ModulationMethod == "USB")
    H = Transmitter_USB_FIR_Coeff_1();
```

```
else
    H = Transmitter_LSB_FIR_Coeff_1();
end
BandpassFIR = dsp.FIRFilter(H.Numerator);

% объект для отрисовки графиков
Plotter = dsp.ArrayPlot(...
    'PlotType', 'Line', ...
    'YLimits', [-1.2, 1.2], ...
    'SampleIncrement', 1/Fs, ...
    'XLabel', 'Time', ...
    'YLabel', 'Amplitude', ...
    'ChannelNames', {'DSB AM Signal','SSB AM Signal','Message'} ...
    );

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

    % амплитудная модуляция
    DSBAmSignal = Message.*Carrier;

    % фильтрация для удаления второй полосы
    SSBAmSignal = BandpassFIR(DSBAmSignal);

    % вывод результатов на график
    Plotter([DSBAmSignal, SSBAmSignal, Message])

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



На временной диаграмме можно видеть гармонический сигнал, соответствующй информационному сообщению (красный). Желтым обозначен результат умножения сообщения на несущую. Этот сигнал соответствует обычной DSB модуляции. Синим цветом представлен сигнал после фильтрации полосовым фильтром. Так как модуляция однотональная, этот сигнал соответсвует синусоиде, частота которой равна $f_c - f_m$ для LSB или $f_c + f_m$ для USB.

Теперь рассмотрим, как эти преобразования выглядят в частотной области. Как упоминалось ранее, результат однотональной DSB модуляции равен сумме двух гармонических сигналов, частоты которых равны $f_m + f_c$ и $f_c - f_m$ Гц:

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

В частотной области это соответствует спектру:

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

Этот результат можно получить, если вспомнить, что спектр гармонического сигнала состоит из двух дельта-функций, и произведение сигналов во временной области соответствует их свертке в частотной.

Если использовать полосовой фильтр для удаления гармонки на частоте $f_m + f_c$, то мы получим LSB сигнал, спектр которого равен:

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

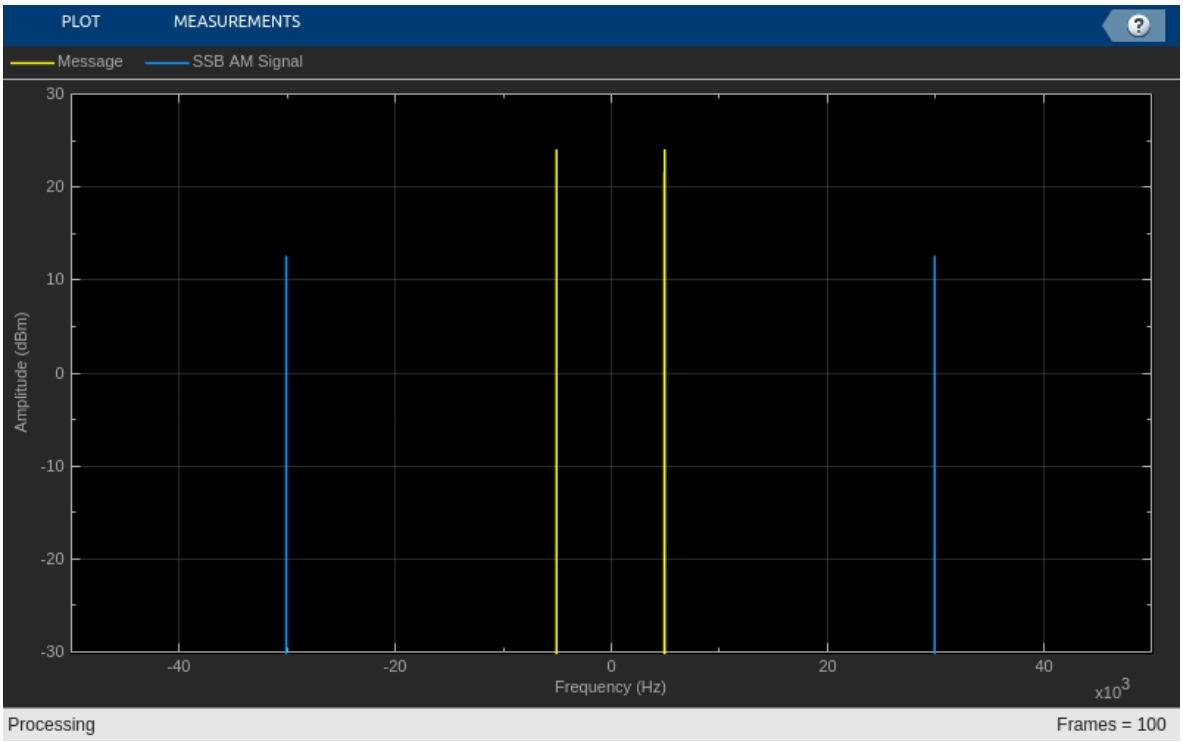
Аналогичные результаты получаем для USB сигнала:

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

Ниже представлен скрипт для получения спектра одноплосного сигнала для случая однотональной модуляции. С помощью переменной ModulationMethod можно выбрать, полосовой фильтр, который сформирует USB или LSB сигнал:

```
clc; clear; close all;
addpath('matlab/SSB_SC');

FrameSize = 5000;   % размер обрабатываемой за один раз пачки данных
Fs = 100e3;         % тактовая частота (Hz)

% выбор метода модуляции
ModulationMethod = "LSB";

% генератор гармонических сигналов
% частота тона: 5 kHz
% частота несущей: 35 kHz
MessageAndCarrierGen = dsp.SineWave(...
    'SampleRate',Fs,...
    'SamplesPerFrame',FrameSize,...
    'Frequency',[35e3 5e3],...
    'Amplitude',[1 1] ...
    );

% расчет коэффициентов и создание полосового фильтра
if (ModulationMethod == "USB")
    H = Transmitter_USB_FIR_Coeff_2();
else
    H = Transmitter_LSB_FIR_Coeff_2();
end
BandpassFIR = dsp.FIRFilter(H.Numerator);

% 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', 'SSB 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
    DSBAmSignal = Message.*Carrier;

    % фильтрация для удаления второй полосы
    SSBAmSignal = BandpassFIR(DSBAmSignal);

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

    % вывод результатов на график
    Plotter(Spectrums)

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



На графике представлен спектр информационного сообщения (желтый) и спектр однополосного сигнала (синий). Можно увидеть, что модулированный сигнал состоит из двух дельта-функций, что подтвеждает предыдущие рассуждения.

3. Однотональная модуляция с использованием преобразования Гильберта

Применение полосовых фильтров для удаление части спектра сопряжено с рядом сложностей. Во-первых, в зависимости от спектра информационного сообщения может потребоваться фильтр с очень узкой переходной

полосой. Такой фильтр будет иметь много коэффициентов, что увеличивает вычислительные затраты на обработку сигнала. Во-вторых, фильтрация выполняется на последнем этапе формирования сигнала на повышенной частоте дискретизации. В-третьих, при изменении частоты несущей коэффициенты фильтра также необходимо будет пересчитывать.

В связи с представленными выше недостатками, как правило, на практике однополосный сигнал формируется другим способом, основанным на преобразовании Гильберта. Рассмотрим из-за чего происходит удвоение полосы сигнала при обычной модуляции. Любой действительный сигнал имеет симметричный спектр относительно нулевой частоты. Как правило, спектр информационного сообщения располагается около нулевой частоты, поэтому такой сигнал называют Base Band. При умножении сообщения на гармонический сигнал его спектр переносится на частоту несущей. На эту частоту переносятся компоненты спектра сообщения как из положительной, так и из отрицательной области частот. Ширина спектра Base Band сигнала на положительных и отрицательных частотах совпадает, поэтому после переноса на несущую ширина спектра модулированного сигнала удваивается.

Удвоение спектра можно избежать, если удалить компоненты спектра сообщения на отрицательных частотах. Так как при этом спектра станет несимметричным, то сообщение будет комплексным. Такой сигнал называют аналитическим. Аналитический сигнал можно получить с помощью преобразования Гильберта, которое представляет из себя всепропускающий фильтр, изменяющий только фазу сигнала. Преобразование Гильберта формирует мнимую часть аналитического сигнала. Таким образом, можем записать:

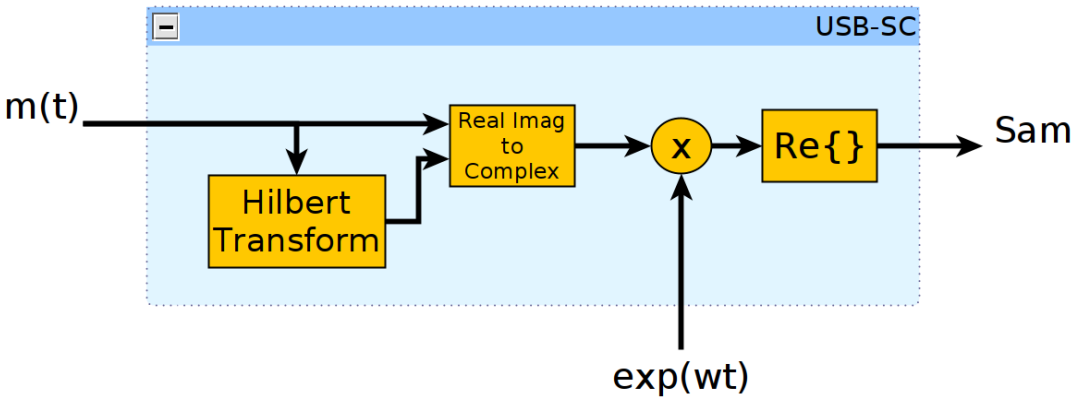
$$m_A(t) = m(t) + j \cdot m_H(t),$$

где $m(t)$ - исходное информационное сообщение, $m_H(t) = H\{m(t)\}$ - преобразование Гильберта от сообщения, $m_A(t)$ - аналитический сигнал с односторонним спектром.

Аналитический сигнал далее можно перенести на несущую частоту с помощью умножения на комплексную экспоненту. В результате сигнал останется комплексным. Чтобы сделать его вещественным, оставим только его действительную часть. В результате получим USB сигнал, так как на несущую будут перенесены только положительные частоты. В виде формул данные преобразования можно записать так:

$$s_{\text{USB}}(t) = \text{Real}\left\{m_A(t) \cdot e^{j \cdot 2\pi \cdot f_c \cdot t}\right\}.$$

Структурная схема передатчика предсталена ниже:



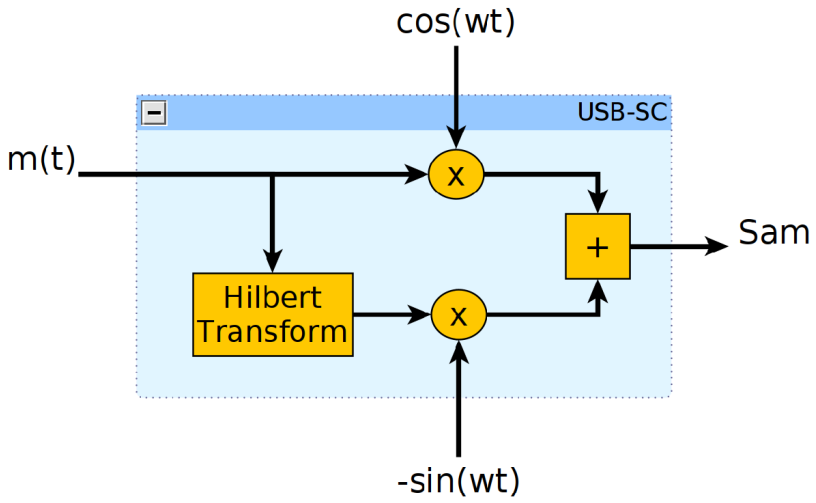
Если представить комплексную экспонету и аналитический сигнал как сумму действительной и мнимой части, то выражение для $s_{\text{USB}}(t)$ можно переписать в виде:

$$s_{\text{USB}}(t) = \text{Real}\{(m(t) + j \cdot m_H(t),) \cdot (\cos(2\pi \cdot f_c \cdot t) + j \cdot \sin(2\pi \cdot f_c \cdot t))\} =$$

$$= \text{Real}\{(m(t) \cdot \cos(2\pi \cdot f_c \cdot t) - m_H(t) \cdot \sin(2\pi \cdot f_c \cdot t)) + j \cdot (m_H(t) \cdot \cos(2\pi \cdot f_c \cdot t) + m(t) \cdot \sin(2\pi \cdot f_c \cdot t))\} =$$

$$= m(t) \cdot \cos(2\pi \cdot f_c \cdot t) - m_H(t) \cdot \sin(2\pi \cdot f_c \cdot t).$$

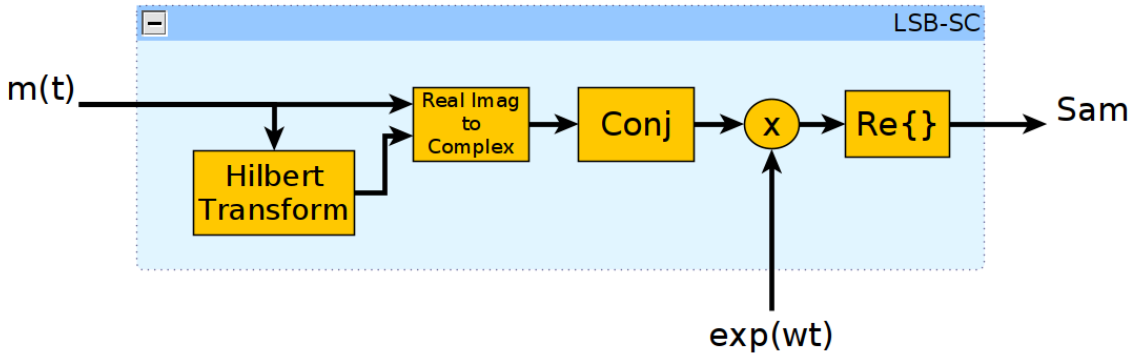
Таким образом, схему приемника можно также представить в виде:



Если провести комплексное сопряжение аналитического сигнала, то получим сигнал, который содержит только отрицательные частоты. После переноса на несущую частоту и выделения действительной части получим LSB сигнал. То есть, имеем:

$$s_{\text{LSB}}(t) = \text{Real}\left\{m_A^*(t) \cdot e^{j \cdot 2\pi \cdot f_c \cdot t}\right\}.$$

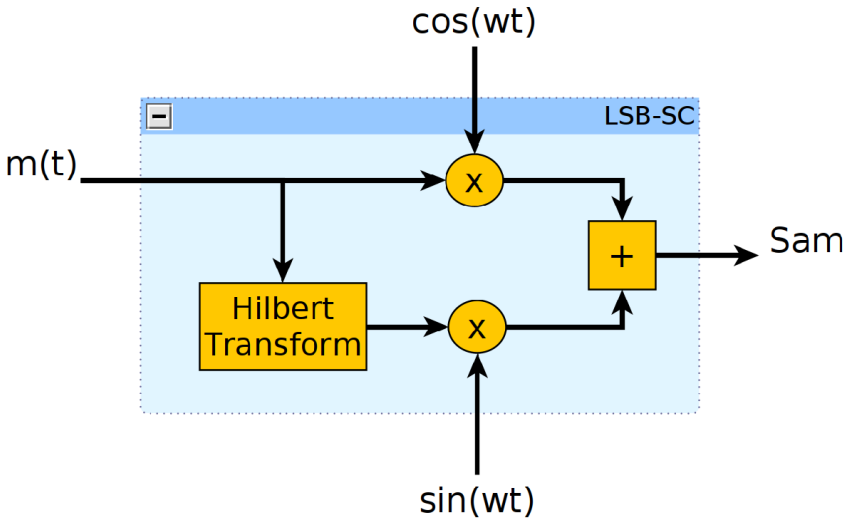
Структурная схема передатчика предсталена ниже:



Если представить комплексную экспонету и аналитический сигнал как сумму действительной и мнимой части, то выражение для $s_{\text{LSB}}(t)$ можно переписать в виде:

$$s_{\text{LSB}}(t) = m(t) \cdot \cos(2\pi \cdot f_c \cdot t) + m_H(t) \cdot \sin(2\pi \cdot f_c \cdot t).$$

Таким образом, схему приемника также можно представить в виде:



Рассмотрим данный метод формирования однополосного сигнала для случая однотональной 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.

Преобразование Гильберта от косинуса равно синусу той же частоты. Значит можем записать:

$$m_H(t) = A_m \sin(2\pi f_m t).$$

Таким образом, аналитичечкий сигнал можно представить в виде:

$$m_A(t) = m(t) + j \cdot m_H(t) = A_m \cos(2\pi f_m t) + j \cdot A_m \sin(2\pi f_m t) = A_m \cdot e^{j \cdot 2\pi \cdot f_m \cdot t}.$$

После переноса на несущую частоту с помощью умножения на комплексную экспоненту и взятия действительной части получим:

$$s_{\text{USB}}(t) = \text{Real}\left\{A_m \cdot e^{j \cdot 2\pi \cdot f_m \cdot t} \cdot e^{j \cdot 2\pi \cdot f_c \cdot t}\right\} = \text{Real}\left\{A_m \cdot e^{j \cdot 2\pi \cdot (f_m + f_c) \cdot t}\right\} = A_m \cos(2\pi (f_c + f_m) t).$$

Как и ожидалось, USB сигнал для однотонального случая состоит из единственного тона на частоте $f_c + f_m$.

Чтобы получить LSB сигнал, необходиму дополнительно произвести комплексное сопряжение аналитического сигнала. В связи с этим можем записать:

$$m_A^*(t) = A_m \cdot e^{-j \cdot 2\pi \cdot f_m \cdot t}.$$

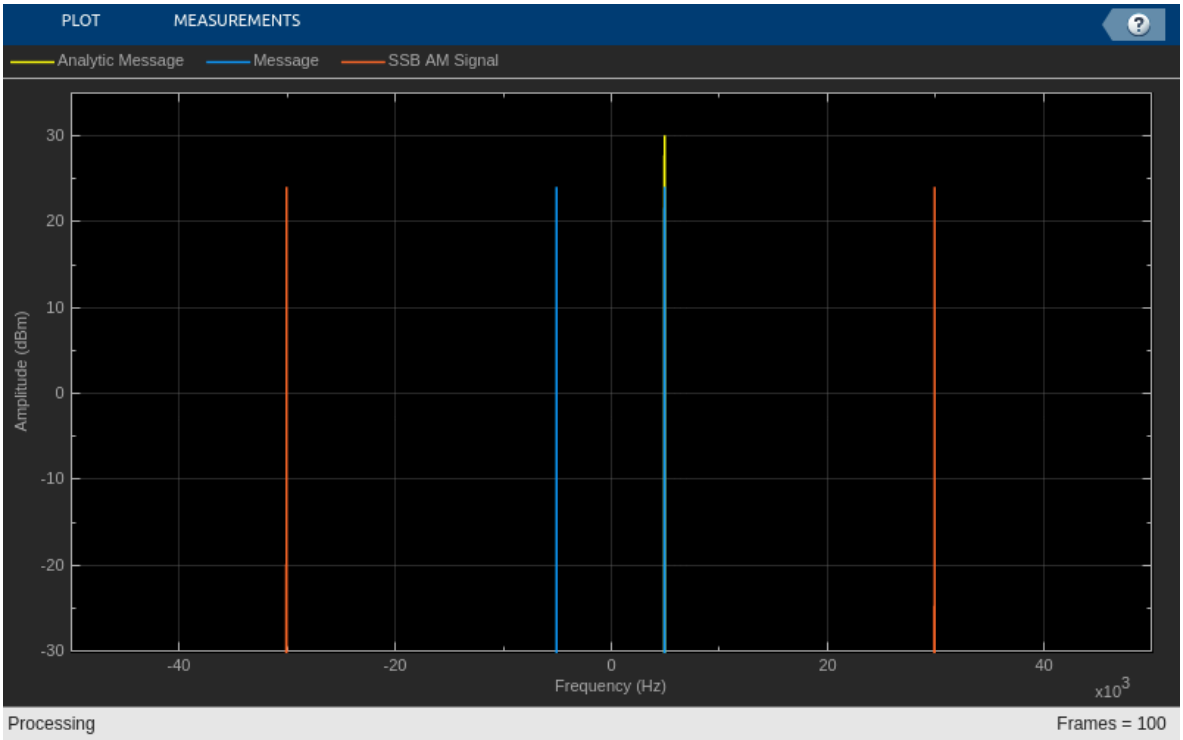
$$s_{\text{LSB}}(t) = \text{Real}\left\{A_m \cdot e^{-j \cdot 2\pi \cdot f_m \cdot t} \cdot e^{j \cdot 2\pi \cdot f_c \cdot t}\right\} = \text{Real}\left\{A_m \cdot e^{j \cdot 2\pi \cdot (f_c - f_m) \cdot t}\right\} = A_m \cos(2\pi (f_c - f_m) t).$$

То есть, LSB сигнал для однотонального случая состоит из единственного тона на частоте $f_c - f_m$.

Ниже представлен скрипт для получения спектра одноплосного сигнала с помощью преобразования Гильберта для случая однотональной модуляции. С помощью переменной ModulationMethod можно выбрать метод модуляции: USB или LSB:

```
clc; clear; close all;

FrameSize = 5000;   % размер обрабатываемой за один раз пачки данных
Fs = 100e3;         % тактовая частота (Hz)

% выбор метода модуляции
ModulationMethod = "LSB";

% генератор однотонального сигнала
% частота тона: 5 kHz
Message = dsp.SineWave(...
    'SampleRate',Fs,...
    'SamplesPerFrame',FrameSize,...
    'Frequency', 5e3,...
    'Amplitude', 1 ...
    );

% генератор несущей
% частота несущей: 35 kHz
Carrier = dsp.SineWave(...
    'SampleRate', Fs,...
    'SamplesPerFrame', FrameSize,...
    'Frequency', 35e3,...
    'ComplexOutput', true, ...
    'Amplitude', 1);

% формирует аналитический сигнал
HilbertTranform = dsp.AnalyticSignal(...
    'FilterOrder', 100 ...
    );

% 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, 35], ...
    'XLabel', 'Frequency (Hz)', ...
    'YLabel', 'Amplitude (dBm)', ...
    'ChannelNames', {'Analytic Message', 'Message', 'SSB AM Signal'}, ...
    'SampleIncrement', Fs/FrameSize ...
    );
```

```
% запуск симуляции
for i = 1:100
    % формирование тонального сигнала
    MessageData = Message();

    % формирование аналитического сигнала
    AnalyticData = HilbertTranform(MessageData);

    % формирование несущей
    CarrierWave = Carrier();

    % однополосная модуляция
    if (ModulationMethod == "USB")
        SSBAmSignal = real(AnalyticData.*CarrierWave);
    else
        SSBAmSignal = real(conj(AnalyticData).*CarrierWave);
    end

    % вычисление спектров
    Spectrums = SpecEstimator([AnalyticData, MessageData, SSBAmSignal]);

    % вывод результатов на график
    Plotter(Spectrums)

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



На графике представлен спектр информационного сигнала (синий) и спектр аналитического сигнала (желтый). В зависимости от вида модуляции (USB или LSB) аналитеческий сигнал будет располагаться только на положительных или только на отрицательных частотах. Также можно увидеть, что модулированный сигнал состоит

из двух дельта-функций, что совпадает с предыдущими результатами, полученными с помощью полосового фильтра.

4. Модуляция несколькими тонами

Разобравшись с однотональной модуляцией, легко распространить результаты на случай модуляции несколькими тонами. Пусть информационное сообщение представляет из себя сумму гармонических сигналов:

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

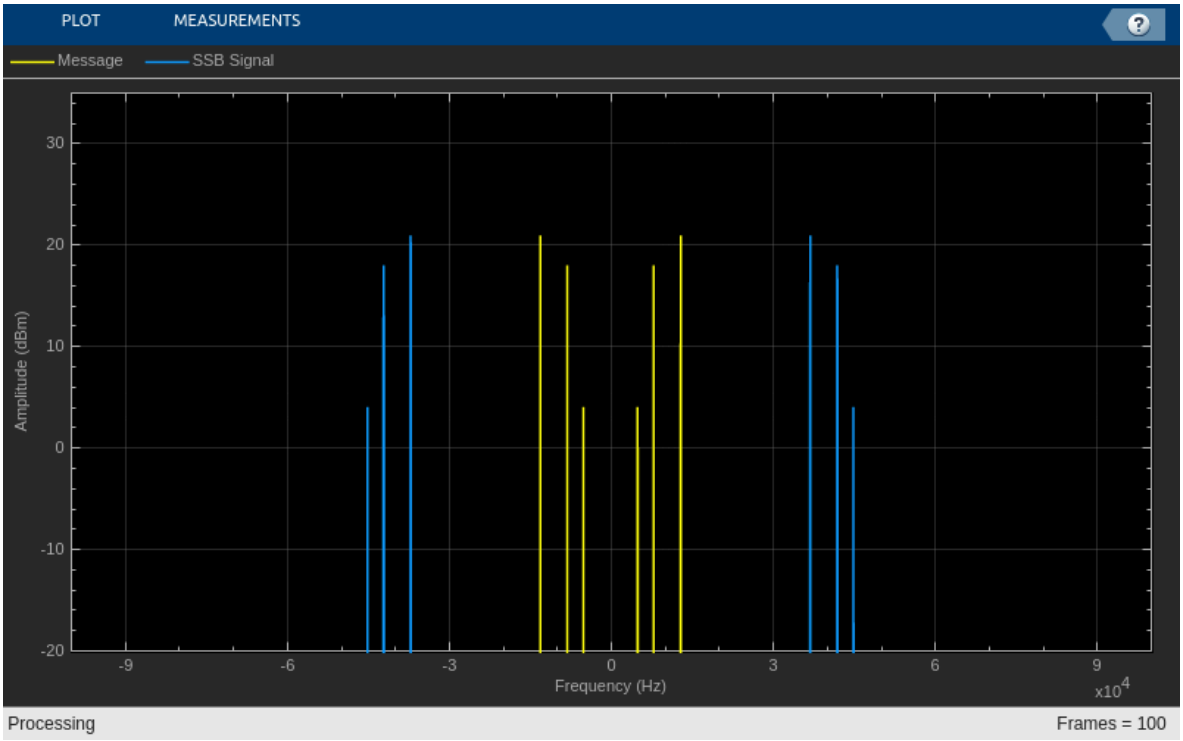
Рассмотрим, что происходит в частотной области. В случае обычной DSB модуляции сигнал на несущей частоте примет следующий вид:

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

То есть, после модуляции каждому тону будут соответстовать две гармоники на частотах $f_i + f_c$ и $f_i - f_c$. Как упоминалось ранее, наличие двух тонов приводит к увеличению спетра в два раза. С помощью предсталенных выше схем SSB передатчиков можно удалить верхниюю или нижнюю половину спектра. 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)

% выбор метода модуляции
ModulationMethod = "LSB";

% генератор информационного сигнала из трех тонов
% частота тонов: 5, 8 и 13 kHz
% амплитуды тонов: 0.1, 0.5 и 0.7
Message = dsp.SineWave(...
    'SampleRate', Fs,...
    'SamplesPerFrame', FrameSize,...
    'Frequency', [5e3 8e3 13e3],...
    'Amplitude', [0.1 0.5 0.7]);

% генератор несущей
% частота несущей: 35 kHz
Carrier = dsp.SineWave(...
    'SampleRate', Fs,...
    'SamplesPerFrame', FrameSize,...
    'Frequency', 50e3,...
    'ComplexOutput', true, ...
    'Amplitude', 1);

% формирует аналитический сигнал
HilbertTranform = dsp.AnalyticSignal(...
    'FilterOrder', 100 ...
```

```
    );

% 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', 'SSB Signal'}, ...
    'SampleIncrement', Fs/FrameSize ...
    );

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

    % формирование аналитического сигнала
    AnalyticData = HilbertTranform(MessageData);

    % формирование несущей
    CarrierWave = Carrier();

    % однополосная модуляция
    if (ModulationMethod == "USB")
        SSBAmSignal = real(AnalyticData.*CarrierWave);
    else
        SSBAmSignal = real(conj(AnalyticData).*CarrierWave);
    end

    % вычисление спектров
    Spectrums = SpecEstimator([MessageData, SSBAmSignal]);

    % вывод результатов на график
    Plotter(Spectrums)

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



Можно увидеть, что спектр модулированного сигнала (синий) в зависимости от вида модуляции соответсвует верхней или нижней половине спектра информационного сообщения (желтый).

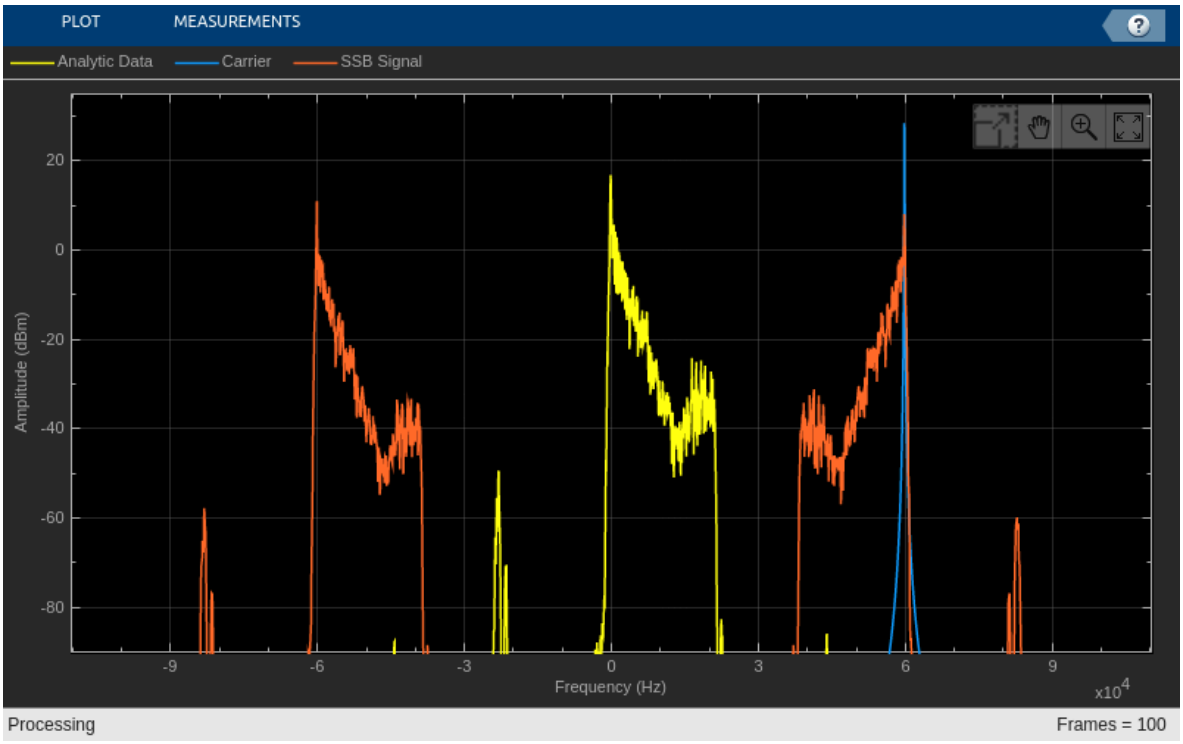
5. Модуляция звуковым сигналом

Рассмотрим, как будет выглядеть спектр сигнала после однополосной 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

% выбор метода модуляции
ModulationMethod = "LSB";

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

% дополнительные расчеты
AudioFs = AudioReader.SampleRate;               % получаем частоту дискретизации аудиосообщения
SignalFs = AudioFs * RateRatio;                 % частота дискретизации модулированного сигнала
```

```
SignalFrameSize = AudioFrameSize * RateRatio;   % количество отсчетов Ам-сигнала, получаемых за один раз

% генератор несущей с частотой 60 kHz
Carrier = dsp.SineWave(...
    'SampleRate', SignalFs,...
    'SamplesPerFrame', SignalFrameSize,...
    'Frequency', 60e3,...
    'ComplexOutput', true, ...
    'Amplitude', 1);

% объект вычисления преобразования Гильберта
% формирует аналитический сигнал
HilbertTranform = dsp.AnalyticSignal(...
    'FilterOrder', 100 ...
    );

% 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', {'Analytic Data', 'Carrier', 'SSB 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);

    % формирование аналитического сигнала
    AnalyticData = HilbertTranform(AudioData);

    % увеличение частоты дискретизации аудиосообщения
    UpsampledData = Upsampler(AnalyticData);

    % формирование несущей
```

```
    CarrierWave = Carrier();

    % однополосная модуляция
    if (ModulationMethod == "USB")
        SSBAmSignal = real(UpsampledData.*CarrierWave);
    else
        SSBAmSignal = real(conj(UpsampledData).*CarrierWave);
    end

    % вычисление спектров
    SpectrumData = SpecEstimator([UpsampledData, CarrierWave, SSBAmSignal]);

    % вывод результатов на график
    Plotter(SpectrumData)

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



Можно увидеть, что с помощью преобразования Гильберта одна из половин спектра удаляется не полностью. На спектре SSB сигнала (красный) видны остатки подавленной половины. Эти оставшиеся компоненты спектра могут влиять на соседние каналы связи. В нашем примере, амплитуда подавленых спектральных компонентов отличается от сигнала почти на 40-60 дБ. В зависимости от требований к передатчику этого может быть достаточно, а может быть и нет.

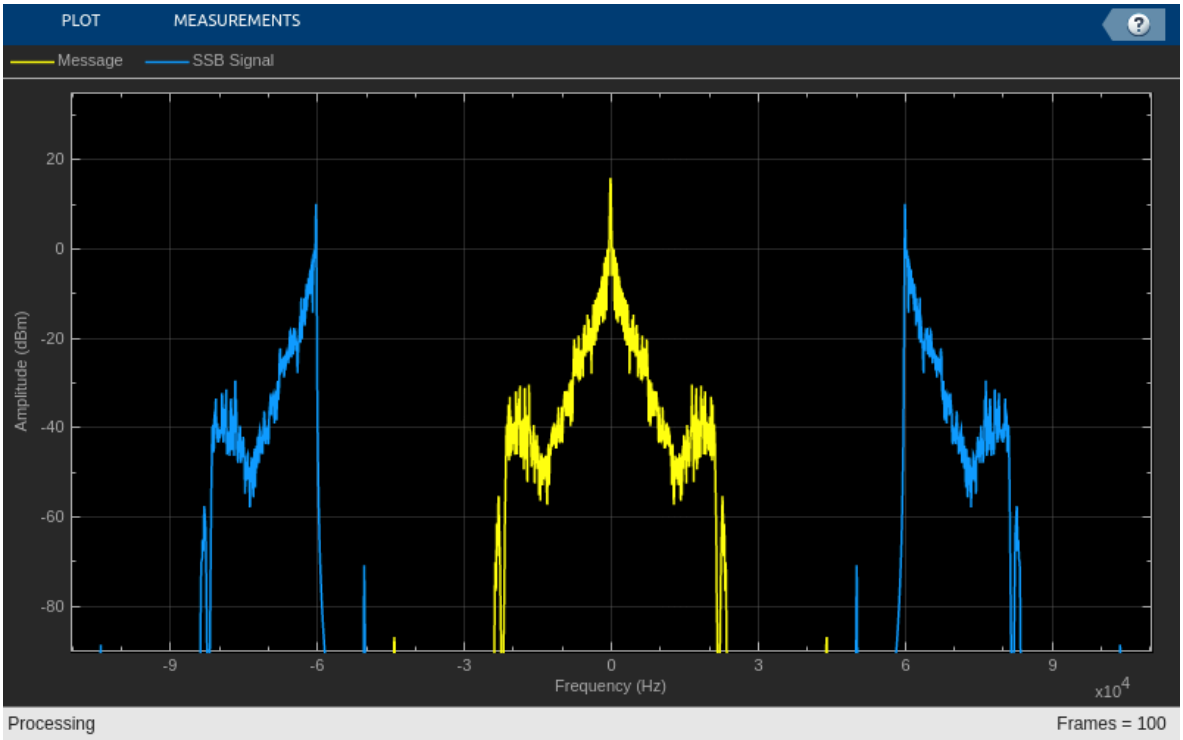
6. Готовые решения Matlab

В среде Matlab есть готовая функция для выполения однополосной амплитудной модуляции: ssbmod(). Проверим ее работу. Функция принимает на вход информационный сигнал, частоту несущей, частоту дискретизации и

начальную фазу несущей. Если в функцию передается строка "upper", то выполнятеся USB модуляция, иначе LSB. 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

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

% выбор метода модуляции
ModulationMethod = "USB";

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

% дополнительные расчеты
AudioFs = AudioReader.SampleRate;               % получаем частоту дискретизации аудиосообщения
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', 'SSB 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
    if (ModulationMethod == "USB")
        AmSignal = ssbmod(MessageData, CarrierFreq, SignalFs, CarrierPhase, 'upper');
    else
        AmSignal = ssbmod(MessageData, CarrierFreq, SignalFs, CarrierPhase);
    end
    % вычисление новой фазы нусущей
    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
```



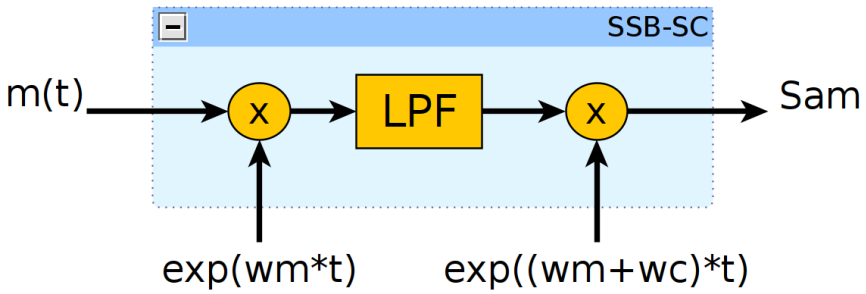
Спектр модулированного сигнала совпадает со спектром из предыдущего пункта.

7. Однотональная модуляция с использованием сдвига спектра и низкочастотного фильтра

Одним из недостатков метода однополосной модуляции с использованием полосового фильтра, является необходимость проводить фильтрацию на повышенной частоте дискретизации. Этого можно избежать, если провести фильтрация до переноса сигнала на частоту несущей. Такой метод однополосной модуляции иногда назавают третьим методом (third method).

Рассмотрим необходимые преобразования. С помощью умножения на комплексную экспоненту нужно сдвинуть сигнал так, чтобы середина спектра будущего однополосного сигнала оказалась на нулевой частоте. Спектр при этом будет несимметричным, а сигнал станет комплексным. После такого переноса с помощью низкочастотного фильтра можно удалить одну половину спектра, а оставшуюся часть перенести на несущую частоту.

Схема приемного устройства при этом будет иметь вид:



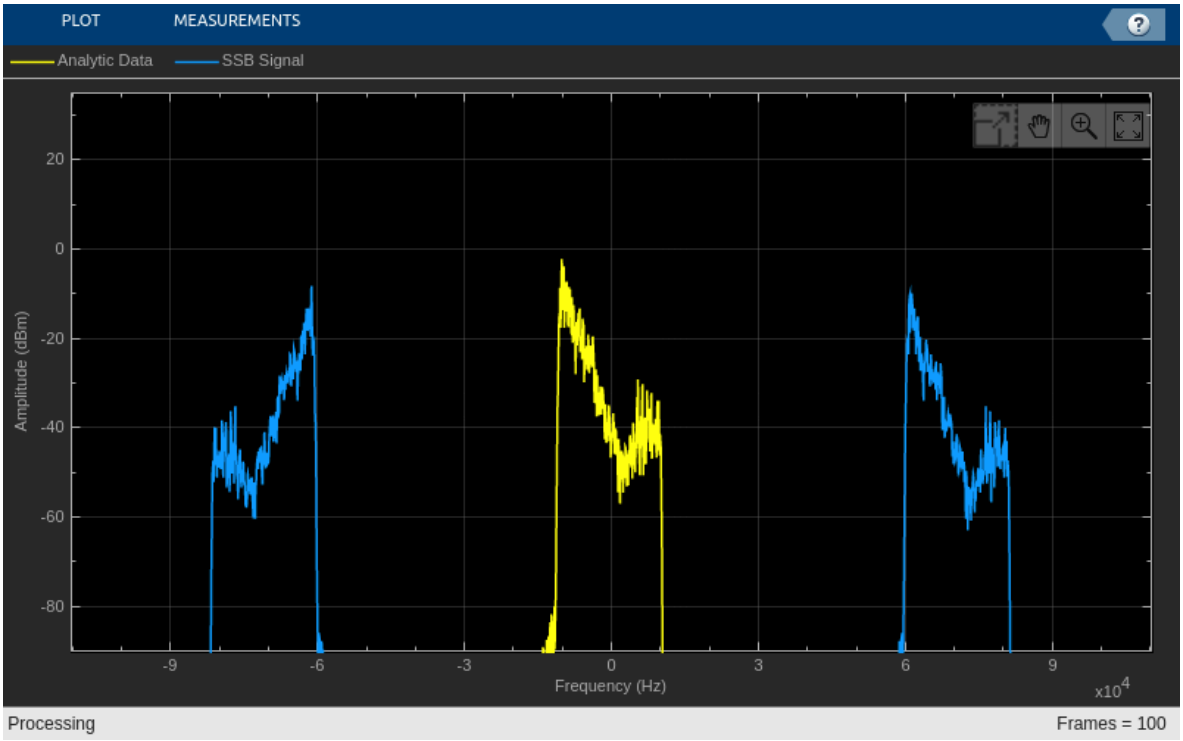
Ниже представлен скрипт для получения спектра однополосного сигнала при модуляции аудиосообщением. С помощью переменной ModulationMethod можно выбрать, какой сигнал будет сформирован: USB или LSB. Считая, что полоса аудиосигнала равна 22 kHz, перед низкочастотной фильтрацией сообщение должно быть сдвинуто на -11 kHz для USB, и на 11 kHz для LSB.

Чтобы сдвинуть сигнал на частоту несущей, после низкочастотной фильтрации его необходимо сдвинуть обратно на 11 kHz. Это можно сделать сразу во время переноса на несущую, умножая сигнал на комплексную экспоненту, частота которой равна сумме частоты несущей и частоты сдвига.

```
clc; clear; close all;
addpath('matlab/SSB_SC');

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

% выбор метода модуляции
ModulationMethod = "USB";

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

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

```
% генератор тона с частотой 11 kHz
BasebandMixer = dsp.SineWave(...
    'SampleRate', AudioFs,...
    'SamplesPerFrame', AudioFrameSize,...
    'Frequency', 11e3,...
    'ComplexOutput', true, ...
    'Amplitude', 1);

% расчет коэффициентов и создание низкочастотного фильтра
H = Transmitter_Baseband_FIR_Coeff();
BasebandFIR = dsp.FIRFilter(H.Numerator);

% генератор несущей с частотой 60 kHz
Carrier_USB = dsp.SineWave(...
    'SampleRate', SignalFs,...
    'SamplesPerFrame', SignalFrameSize,...
    'Frequency', 60e3 + 11e3,...
    'ComplexOutput', true, ...
    'Amplitude', 1);

Carrier_LSB = dsp.SineWave(...
    'SampleRate', SignalFs,...
    'SamplesPerFrame', SignalFrameSize,...
    'Frequency', 60e3 - 11e3,...
    'ComplexOutput', true, ...
    'Amplitude', 1);

% 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', {'Analytic Data', 'SSB 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);

    % формирование тона для переноса сперктра
    MixerWave = BasebandMixer();

    % фильтрация для удаления второй полосы
    if (ModulationMethod == "LSB")
        SSBData = BasebandFIR(AudioData.*MixerWave);
    else
        SSBData = BasebandFIR(AudioData.*conj(MixerWave));
    end

    % увеличение частоты дискретизации аудиосообщения
    UpsampledData = Upsampler(SSBData);

    % формирование несущей
    if (ModulationMethod == "LSB")
        CarrierWave = Carrier_LSB();
    else
        CarrierWave = Carrier_USB();
    end

    % перенос на частоту несущей
    SSBAmSignal = real(UpsampledData.*CarrierWave);

    % вычисление спектров
    SpectrumData = SpecEstimator([UpsampledData, SSBAmSignal]);

    % вывод результатов на график
    Plotter(SpectrumData)

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



Желтым цветом на графике представлен спектр сообщения после низкочастотной фильтрации, а синим - спектр модулированного сигнала. Можно увидеть, что получены те же результаты, что и при использовании полосового фильтра и преобразования Гильберта.

8. Преимущества и недостатки SSB-SC

Преимущества:

- уменьшение вдвое ширины спектра модулированного сигнала;
- модуляция эффективна с точки зрения мощности, так как несущая не передается.

Недостатки:

- возможна только когерентная демодуляция.

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