

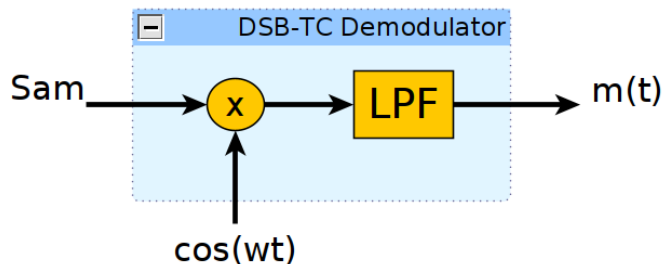
Демодуляция. Часть 1.

Single Sideband Suppressed Carrier (AM-SSB-SC)

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

1. Когерентная демодуляция

Рассмотрим когерентный способ демодуляции однополосного AM-сигнала с подавленной несущей. Для этого принятый сигнал нужно смешать (перемножить) с колебанием, которое является копией несущей. Затем результат необходимо пропустить через фильтр нижних частот (ФНЧ). Схема когерентного демодулятора представлена ниже:



Для примера рассмотрим USB сигнал. Результаты для LSB получаются таким же образом и отличаются только знаком перед квадратурной частью принимаемого однополосного сигнала. В аналитическом виде описанные выше преобразования можно представить следующим образом. Пусть принятый однополосный сигнал имеет вид:

$$s_{am}(t) = m(t) \cdot \cos(2\pi f_c t) - m_H(t) \cdot \sin(2\pi f_c t),$$

где $m(t)$ - информационное сообщение, $m_H(t)$ - преобразование Гильберта от сообщения, f_c - частота несущей.

После перемножения с гармоническим сигналом, частота и фаза которого совпадают с несущей, получим:

$$\begin{aligned} s_{mix}(t) &= [m(t) \cdot \cos(2\pi f_c t) - m_H(t) \cdot \sin(2\pi f_c t)] \cdot \cos(2\pi f_c t) = \\ &= m(t) \cdot \cos(2\pi f_c t) \cdot \cos(2\pi f_c t) - m_H(t) \cdot \sin(2\pi f_c t) \cdot \cos(2\pi f_c t). \end{aligned}$$

Вспомнив правило произведения косинусов и синусов, можем записать результат умножения в виде:

$$s_{mix}(t) = \frac{1}{2} m(t) + \frac{1}{2} m(t) \cdot \cos(2\pi \cdot 2f_c \cdot t) - \frac{1}{2} m_H(t) \cdot \sin(2\pi \cdot 2f_c \cdot t).$$

Таким образом, сигнал на выходе смесителя является суммой информационного сообщения, а также сообщения и его преобразования Гильберта на удвоенной частоте несущей $2f_c$. После пропускания данной суммы через ФНЧ, слагаемые на удвоенной частоте исчезнут, и мы получим:

$$s_{\text{LPF}}(t) = \frac{1}{2} m(t).$$

То есть, восстановленный сигнал является масштабированной копией исходного сообщения.

Сложность при таком методе демодуляции заключается в том, что колебание в смесителе должно точно совпадать по частоте и фазе с несущей. Очевидно, что на практике генераторы на передающей и приемной сторонах не могут быть абсолютно идентичными. Между ними всегда есть некоторая расстройка. Рассмотрим влияние такого рассогласования на качество демодуляции. Пусть рассогласование по частоте и фазе между колебаниями в смесителе и несущей равны Δf и $\Delta \phi$ соответственно. В этом случае на выходе смесителя получим:

$$\begin{aligned} s_{\text{mix}}(t) &= [m(t) \cdot \cos(2\pi f_c t) - m_H(t) \cdot \sin(2\pi f_c t)] \cdot \cos(2\pi(f_c + \Delta f)t + \Delta \phi) = \\ &= \frac{1}{2} m(t) \cdot \cos(2\pi \Delta f t + \Delta \phi) - \frac{1}{2} m_H(t) \cdot \sin(2\pi \Delta f t + \Delta \phi) + \text{слагаемые с удвоенной частотой.} \end{aligned}$$

После ФНЧ слагаемые на удвоенной частоте исчезнут:

$$s_{\text{LPF}}(t) = \frac{1}{2} m(t) \cdot \cos(2\pi \Delta f t + \Delta \phi) - \frac{1}{2} m_H(t) \cdot \sin(2\pi \Delta f t + \Delta \phi).$$

Таким образом, как и для случая DSB сигнала, восстановленное сообщения будет масштабировано на множитель $\cos(2\pi \Delta f t + \Delta \phi)$, амплитуда которого изменятся с частотой, равной расстройке по частоте. По мимо этого после фильтрации останется еще одно слагаемое, равное $\frac{1}{2} m_H(t) \cdot \sin(2\pi \Delta f t + \Delta \phi)$. Оно также будет вносить искажения в восстановленное сообщение.

Ниже представлен скрипт, который демонстрирует когерентный метод демодуляции для аудиосообщения на несущей частоте 60кГц. С помощью переменных FreqOffset и PhaseOffset можно задать расстройку по частоте и фазе и на слух оценить искажения сигнала. Переменная ModulationMethod задает вид модуляции (USB или LSB).

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

SignalFrameSize = 10000; % количество отсчетов Ам-сигнала, получаемых за один раз
FramesNumber = 500;      % число обрабатываемых пачек данных
AudioAmp = 3;             % коэффициент усиления аудиосигнала
RateRatio = 10;           % коэффициент увеличения частоты дискретизации
Fc = 60e3;                % частота несущей

FreqOffset = 500;         % расстройка по частоте (Hz)
PhaseOffset = 90 * pi/180; % расстройка по фазе (градусы)

% выбор метода модуляции
ModulationMethod = "USB";

% объект для считывания отсчетов аудиофайла
if (ModulationMethod == "USB")
    WaveFileName = 'wav/Audio_USB_SC.wav';
else
    WaveFileName = 'wav/Audio_LSB_SC.wav';
end
```

```

AudioReader = dsp.AudioFileReader(...
    WaveFileName, ...
    'SamplesPerFrame', SignalFrameSize...
);

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

% генератор несущей
Carrier = dsp.SineWave(...
    'SampleRate', SignalFs,...
    'SamplesPerFrame', SignalFrameSize,...
    'Frequency', Fc + FreqOffset,...
    'PhaseOffset', pi/2 + PhaseOffset...
);

% расчет коэффициентов фильтра нижних частот
H = Audio_Lowpass_FIR_Coeff();

% создание объекта для фильтрации
LowpassFIR = dsp.FIRFilter(H.Numerator);

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

% 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', {'AM Signal', 'Mixed Signal', 'Filtered Signal'}, ...
    'SampleIncrement', SignalFs/SignalFrameSize ...
);

Message = [];

% запуск симуляции
for i = 1:FramesNumber
    % считывание отсчетов АМ-сигнала и выделение синфазного канала
    AmSignal = AudioReader();
    AmSignal = AmSignal(:,1);

```

```

% смешивание AM-сигнала и несущей
MixedSignal = AmSignal .* Carrier();

% фильтрация сигнала
BasebandSignal = LowpassFIR(MixedSignal);

% понижение частоты дискретизации
Message = [Message; AudioAmp * DownSampler(BasebandSignal)];

% вычисление спектров
SpectrumData = SpecEstimator([AmSignal MixedSignal BasebandSignal]);

% вывод результатов на график
Plotter(SpectrumData)

% задержка в 0.1 секунды для лучшей визуализации
pause(0.01)
end

```



Выше представлены преобразования в частотной области. Желтым цветом обозначен спектр принимаемого однополосного сигнала. Синим - спектр на выходе смесителя. Можно увидеть, что он состоит из информационного сообщения и спектральных составляющих на удвоенной частоте несущей. После фильтрации остается только информационное сообщение (красный цвет).

```

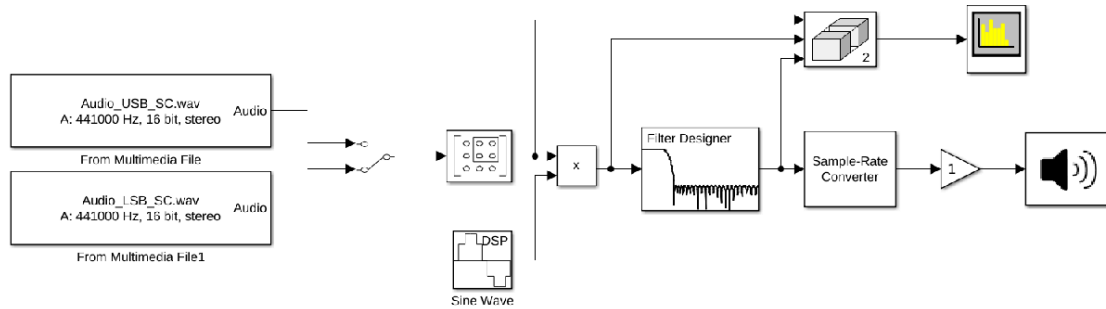
% проигрывание полученного сообщения
sound(Message, AudioFs);

```

На слух можно оценить искажения аудиосигнала из-за рассогласования по фазе и частоте. В отличие от DSB сигнала при расстройке по фазе сигнал слышится также хорошо, и он не искажен. Это связано с тем, что теперь

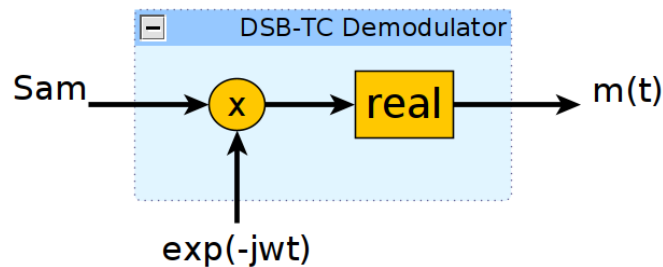
в сигнале после фильтрации присутствует слагаемое $\frac{1}{2} m_H(t) \cdot \sin(2\pi \Delta f t + \Delta \phi)$. При нулевой расстройке по фазе оно равно нулю, и возрастает при увеличении расстройки. То есть, при уменьшении первого слагаемого с сообщением, слагаемое с его преобразованием Гильберта растет, и наоборот. Сигналы $m(t)$ и $m_H(t)$ имеют одинаковые амплитудные спектры и различаются лишь фазой. Так как человеческий слух реагирует только на амплитудный спектр, для нас эти сигналы кажутся одинаковыми. Расстройка по частоте приводит к ухудшению качества и появлению свистящих звуков и биений.

В файле *SSB_SC_Receiver_Part_1_1.slx* представлена Simulink модель когерентного демодулятора, который ранее был реализован в виде скрипта. С помощью переключателя можно выбирать между USB и LSB сигналами.



2. Когерентная демодуляция при квадратурном приеме

В случае, когда прием однополосного сигнала осуществляется квадратурным способом (например почти во всех SDR-приемниках), структуру когерентного демодулятора можно упростить:



То есть, принятый однополосный сигнал нужно умножить на мнимую экспоненту, а потом просто выделить действительную часть. Далее для примера рассмотрим случай USB сигнала. В аналитическом виде описанные выше преобразования можно представить следующим образом. Пусть принятый однополосный сигнал имеет вид:

$$s_{am}(t) = m_A(t) \cdot e^{2\pi f_c t},$$

где, как и ранее, $m_A(t) = m(t) + j \cdot m_H(t)$ - аналитический сигнал, $m(t)$ - информационное сообщение, $m_H(t)$ - преобразование Гильберта от сообщения, f_c - частота несущей.

После умножения на мнимую экспоненту, частота и фаза которой совпадает с несущей, получим:

$$s_{\text{mix}}(t) = m_A(t) \cdot e^{2\pi f_c t} \cdot e^{-2\pi f_c t} = m_A(t) = m(t) + j \cdot m_H(t).$$

Выделение действительной части приведет к точному восстановлению информационного сообщения:

$$\text{Re}\{s_{\text{mix}}(t)\} = m(t).$$

Теперь рассмотрим, что произойдет при наличии расстройки по фазе и частоте. Сигнал на выходе смесителя можно представить в виде:

$$s_{\text{mix}}(t) = m_A(t) \cdot e^{2\pi f_c t} \cdot e^{-2\pi (f_c + \Delta f)t - \Delta\phi} = m_A(t) \cdot e^{-2\pi \Delta f t - \Delta\phi}.$$

После вычисления действительной части получим:

$$\text{Re}\{s_{\text{mix}}(t)\} = m(t) \cdot \cos(2\pi \Delta f t + \Delta\phi) - m_H(t) \cdot \sin(2\pi \Delta f t + \Delta\phi).$$

Данный результат совпадает с формулой, полученной в предыдущем разделе. Соответственно наличие расстройки по частоте и фазе будет приводить к аналогичным искажениям.

Ниже представлен скрипт для реализации когерентной демодуляции в случае квадратурного приема. С помощью переменных FreqOffset и PhaseOffset можно задать расстройку по частоте и фазе и на слух оценить искажения сигнала. Переменная ModulationMethod задает вид модуляции (USB или LSB).

```
clc; clear; close all;

SignalFrameSize = 10000; % количество отсчетов Ам-сигнала, получаемых за один раз
FramesNumber = 500;      % число обрабатываемых пачек данных
AudioAmp = 1;             % коэффициент усиления аудиосигнала
RateRatio = 10;           % коэффициент увеличения частоты дискретизации
Fc = 60e3;                % частота несущей

FreqOffset = 0;           % расстройка по частоте (Hz)
PhaseOffset = 0 * pi/180; % расстройка по фазе (градусы)

% выбор метода модуляции
ModulationMethod = "LSB";

% объект для считывания отсчетов аудиофайла
if (ModulationMethod == "USB")
    WaveFileName = 'wav/Audio_USB_SC.wav';
else
    WaveFileName = 'wav/Audio_LSB_SC.wav';
end
AudioReader = dsp.AudioFileReader(...
    WaveFileName, ...
    'SamplesPerFrame', SignalFrameSize...
);

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

% генератор несущей
```

```

Carrier = dsp.SinWave(...
    'SampleRate', SignalFs,...
    'SamplesPerFrame', SignalFrameSize,...
    'Frequency', [Fc + FreqOffset, Fc + FreqOffset],...
    'PhaseOffset', [pi/2 + PhaseOffset, PhaseOffset]...
);

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

% 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', {'AM Signal', 'Carrier', 'Mixed Signal'}, ...
    'SampleIncrement', SignalFs/SignalFrameSize ...
);

Message = [];

% запуск симуляции
for i = 1:FramesNumber
    % считывание отсчетов AM-сигнала и формирование комплексного сигнала
    AmSignal = AudioReader();
    AmSignal = AmSignal(:,1) + 1j*AmSignal(:,2);

    % получение несущей
    CarrierWave = Carrier();
    CarrierWave = CarrierWave(:,1) - 1j*CarrierWave(:,2);

    % смешивание AM-сигнала и несущей
    MixedSignal = AmSignal .* CarrierWave;
    MixedSignal = real(MixedSignal);

    % понижение частоты дискретизации
    Message = [Message; AudioAmp*DownSampler(MixedSignal)];

    % вычисление спектров
    SpectrumData = SpecEstimator([AmSignal CarrierWave MixedSignal]);

    % вывод результатов на график

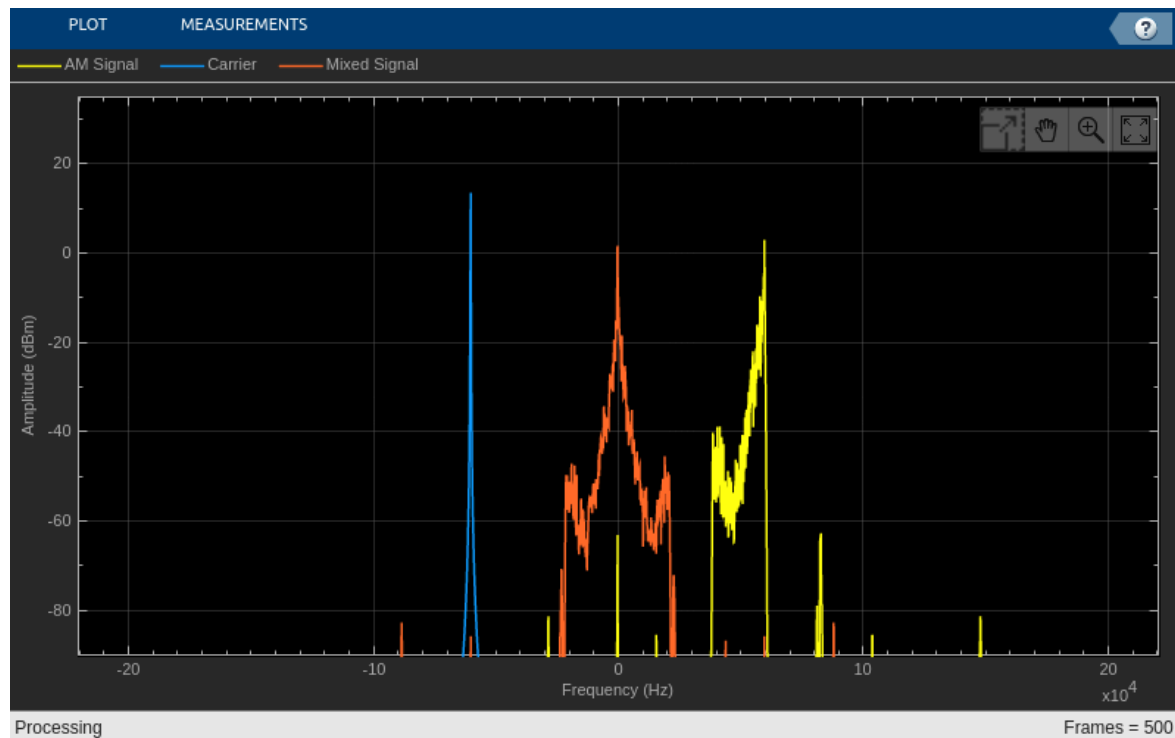
```

```
Plotter(SpectrumData)
```

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

```
pause(0.01)
```

```
end
```

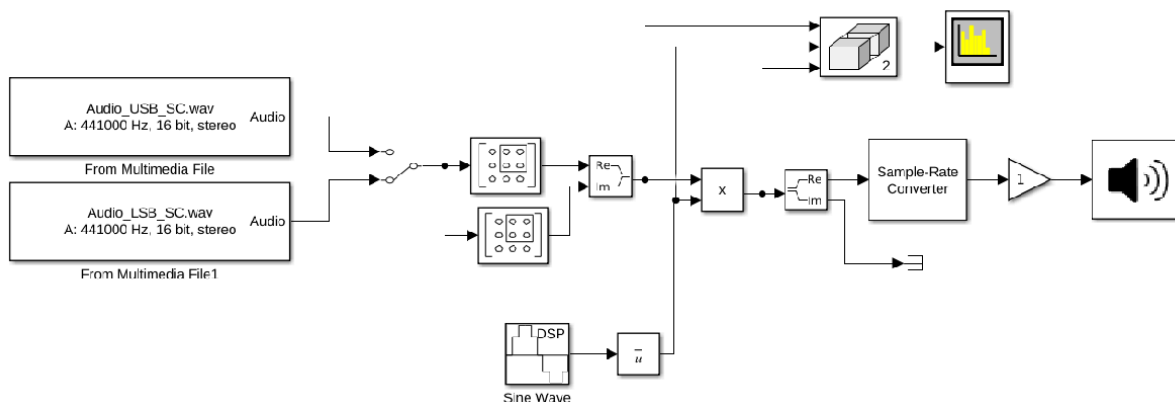


Выше представлены преобразования в частотной области. Желтым цветом обозначен спектр принимаемого однополосного сигнала. Так как прием квадратурный, спектр сигнала комплексный и расположен только в положительной области частот. Синим цветом представлена мнимая экспонента, а красным - сигнал на выходе смесителя.

```
% проигрывание полученного сообщения
```

```
sound(Message, AudioFs);
```

В файле *SSB_SC_Receiver_Part_1_2.slx* представлена Simulink модель когерентного демодулятора в квадратурном случае. С помощью переключателя можно выбирать между USB и LSB сигналами.



3. Некогерентная демодуляция однополосного сигнала с подавленной несущей

В случае однополосной амплитудной модуляции с подавленной несущей сама несущая не передается, что является преимуществом, так приводит к экономии мощности. Однако, с другой стороны это усложняет прием такого сигнала. Очевидно, что из-за наличия в сигнала слагаемого, равного преобразованию Гильберта от сообщения, огибающая сигнала не будет совпадать с сообщением. Таким образом, некогерентный демодулятор для приема сигнала с подавленной несущей применять нельзя.

Убедимся в этом для случая квадратурного приема. Для получения огибающей комплексного сигнала достаточно вычислить его модуль. Воспользуемся тем же скриптом, который рассматривается в DSB TC Receiver Part 3.

Скрипт представлен ниже:

```
clc; clear; close all;

SignalFrameSize = 10000; % количество отсчетов Ам-сигнала, получаемых за один раз
FramesNumber = 500;      % число обрабатываемых пачек данных
AudioAmp = 1;             % коэффициент усиления аудиосигнала
RateRatio = 10;           % коэффициент увеличения частоты дискретизации

% выбор метода модуляции
ModulationMethod = "LSB";

% объект для считывания отсчетов аудиофайла
if (ModulationMethod == "USB")
    WaveFileName = 'wav/Audio_USB_SC.wav';
else
    WaveFileName = 'wav/Audio_LSB_SC.wav';
end
AudioReader = dsp.AudioFileReader(...
    WaveFileName, ...
    'SamplesPerFrame', SignalFrameSize...
);

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

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

% объект для отрисовки графиков
Plotter = dsp.ArrayPlot(...
    'PlotType', 'Line', ...
    'YLimits', [-0.2, 0.2], ...
    'XLabel', 'Time (sec)', ...
    'YLabel', 'Amplitude (dBm)', ...
    'ChannelNames', {'AM Signal', 'Envelope'}, ...
    'SampleIncrement', 1/SignalFs ...
);
```

```

);

Message = [];

% запуск симуляции
for i = 1:FramesNumber
    % считывание отсчетов АМ-сигнала и формирование комплексного сигнала
    AmSignal = AudioReader();
    AmSignal = AmSignal(:,1) + 1j*AmSignal(:,2);

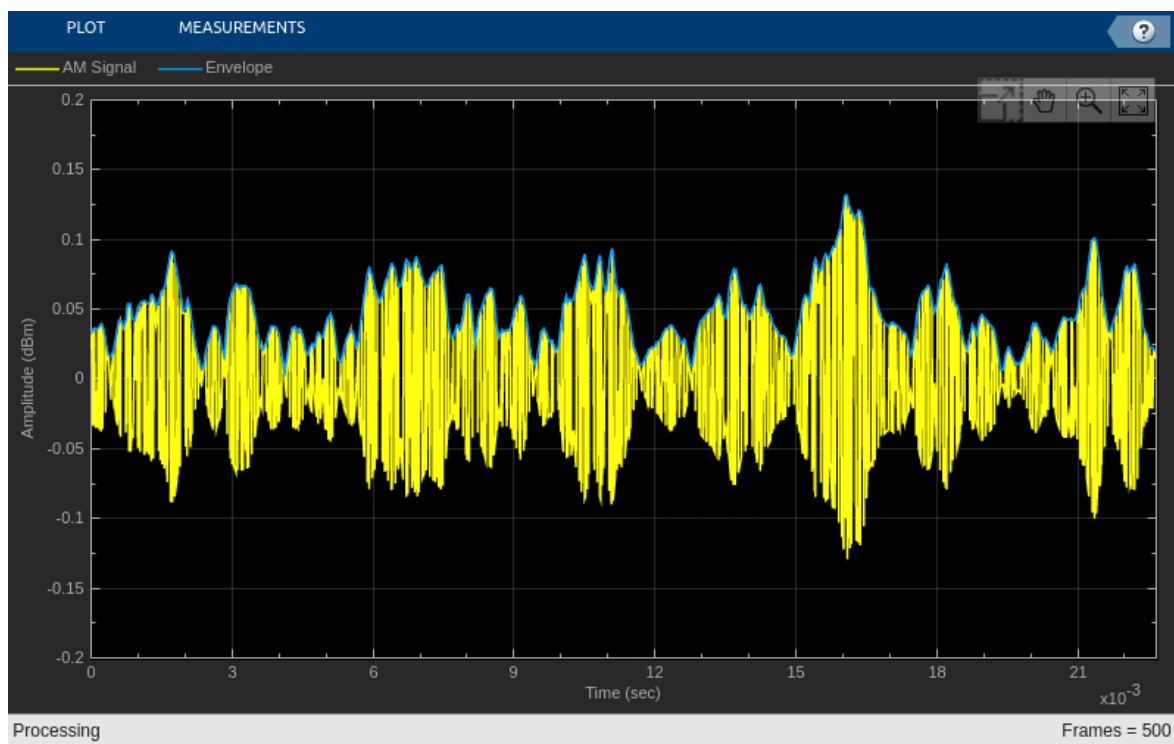
    % вычисление амплитуды сигнала
    AbsAmSignal = abs(AmSignal);

    % понижение частоты дискретизации
    Message = [Message; AudioAmp*DownSampler(AbsAmSignal)];

    % вывод результатов на график
    Plotter([real(AmSignal) AbsAmSignal])

    % задержка в 0.1 секунды для лучшей визуализации
    pause(0.01)
end

```



Выше представлены временные диаграммы сигналов на входе (желтый цвет, действительная составляющая принятого сигнала) и выходе (синий цвет) демодулятора.

```

% проигрывание полученного сообщения
sound(Message, AudioFs);

```

На слух можно обнаружить явные искажения аудиосообщения, что подтверждает невозможность приема однополосного сигнала с подавленной несущей некогерентным способом.

4. Когерентная демодуляция с помощью обычной фазовой автоподстройки частоты

Рассмотренный в DSB TC Receiver Part 2 способ восстановления несущей с помощью PLL для сигнала с подавленной несущей так же не применим. В принимаемом сигнале отсутствует в явном виде гармонический сигнал соответствующий несущей. Из-за этого фазовая автоподстройка частоты не может войти в режим синхронизации и точно подстроить частоту и фазу управляемого генератора.

Убедимся в этом для случая квадратурного приема. Воспользуемся тем же скриптом, который рассматривается в DSB TC Receiver Part 2. Скрипт представлен ниже:

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

SignalFrameSize = 10000; % количество отсчетов Ам-сигнала, получаемых за один раз
FramesNumber = 500;      % число обрабатываемых пачек данных
AudioAmp = 1;             % коэффициент усиления аудиосигнала
RateRatio = 10;           % коэффициент увеличения частоты дискретизации
Fc = 60e3;                % частота несущей

% расстройка по частоте (Hz)
FreqOffset = 1000;

% выбор метода модуляции
ModulationMethod = "USB";

% объект для считывания отсчетов аудиофайла
if (ModulationMethod == "USB")
    WaveFileName = 'wav/Audio_USB_SC.wav';
else
    WaveFileName = 'wav/Audio_LSB_SC.wav';
end
AudioReader = dsp.AudioFileReader(...
    WaveFileName, ...
    'SamplesPerFrame', SignalFrameSize...
);

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

% фпч для восстановления несущей
AmPLL = AmComplexPLL( ...
    'SampleFrequency', SignalFs, ...
    'NoiseBandwidth', 100, ...
    'Dampingfactor', 0.7, ...
    'CentralFrequency', Fc + FreqOffset ...
);
```

```

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

% 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', {'AM Signal', 'Carrier', 'Mixed Signal'}, ...
    'SampleIncrement', SignalFs/SignalFrameSize ...
);

Message = [];

% запуск симуляции
for i = 1:FramesNumber
    % считывание отсчетов AM-сигнала и формирование комплексного сигнала
    AmSignal = AudioReader();
    AmSignal = AmSignal(:,1) + 1j*AmSignal(:,2);

    % восстанавливаем несущую
    [Carrier, Offset] = AmPLL(AmSignal);

    % смешивание AM-сигнала и несущей
    MixedSignal = AmSignal .* conj(Carrier);
    MixedSignal = real(MixedSignal);

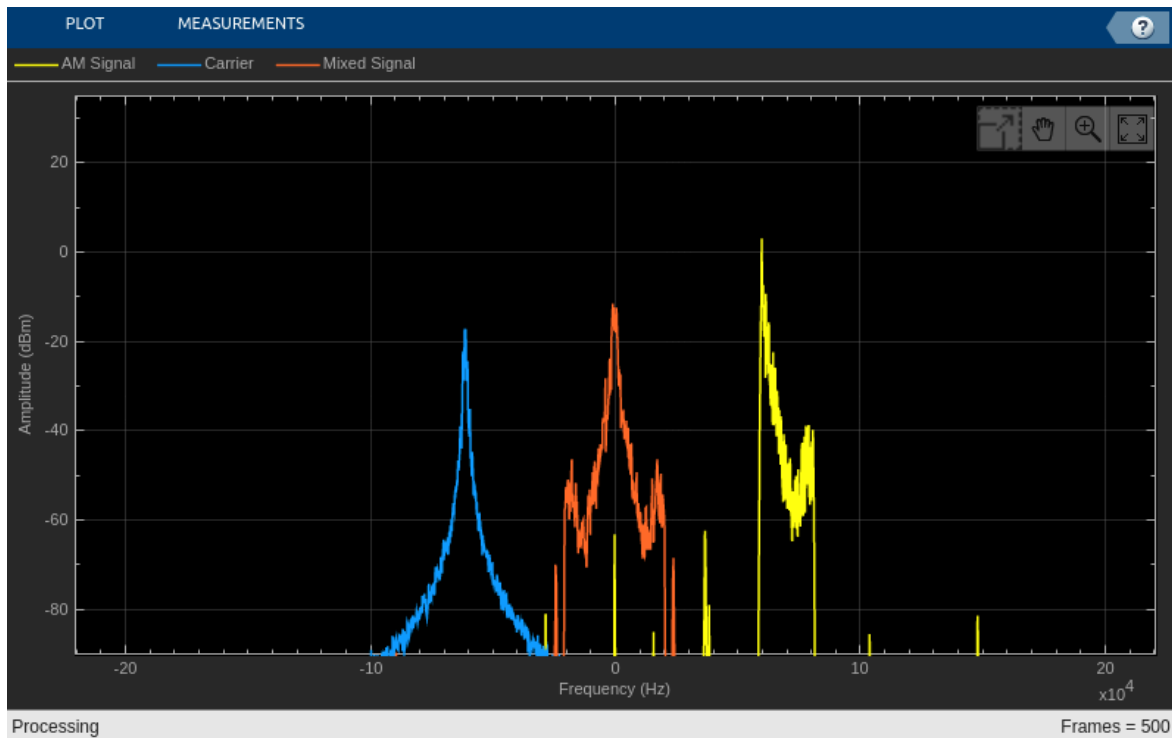
    % понижение частоты дискретизации
    Message = [Message; AudioAmp*DownSampler(MixedSignal)];

    % вычисление спектров
    SpectrumData = SpecEstimator([AmSignal conj(Carrier) MixedSignal]);

    % вывод результатов на график
    Plotter(SpectrumData)

    % задержка в 0.1 секунды для лучшей визуализации
    pause(0.01)
end

```



Выше представлены сигналы в частотной области. Желтым цветом обозначен спектр принимаемого однополосного сигнала. Синим цветом обозначен сигнал на выходе NCO PLL, а красным - сигнал на выходе смесителя. Можно увидеть, что спектр на выходе NCO намного шире, чем на графике, показанном в DSB TC Receiver Part 2. Это свидетельствует о плохой синхронизации, так как сигнал на выходе NCO отличается от гармонического.

```
% проигрывание полученного сообщения
sound(Message, AudioFs);
```

На слух также можно обнаружить искажения аудиосообщения. Таким образом, качественно восстановить несущую, просто подав принятый однополосный сигнал на вход PLL, не получится.

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