Демодуляция. Часть 1.

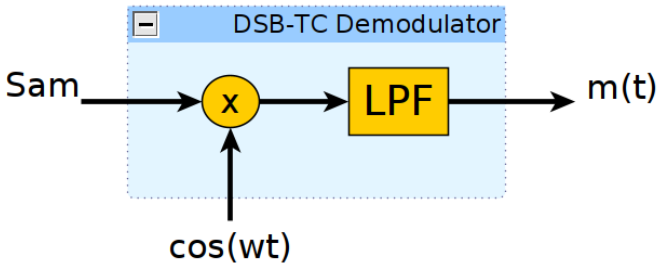
Double Sideband Suppressed Carrier

(AM-DSB-SC)

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

1. Когерентная демодуляция

Рассмотрим когерентный способ демодуляции АМ-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:



В аналитическом виде описанные выше преобразования можно представить следующим образом. Пусть принятый АМ-сигнал имеет вид:

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

где $m(t)$ - информационное сообщение, f_c - частота несущей.

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

$$s_{\text{mix}}(t) = m(t) \cdot \cos(2\pi f_c t) \cdot \cos(2\pi f_c t).$$

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

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

Таким образом, сигнал на выходе смесителя явлется суммой инфомационного сообщения и сигнала на удвоенной частоте несущей $2f_c$. После пропускания данной суммы через ФНЧ, второе слагаемое исчезнет, 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 вляние такого рассогласования на качество демодуляции. Пусть рассогласование по частоте и фазе между колебаниями в смесителе и несущей равны Δf и $\Delta\phi$ соотвветственно. В этом случае на выходе смесителя получим:

$$s_{\text{mix}}(t) = m(t) \cdot \cos(2\pi f_c t) \cdot \cos(2\pi (f_c + \Delta f)t + \Delta\phi) =$$
$$= \frac{1}{2} m(t) \cdot \cos(2\pi \Delta \text{ft} + \Delta\phi) + \text{слагаемое } c \text{ удвоенной частотой}.$$

После ФНЧ слагаемое на удвоенной частоте исчезнет:

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

Для начала рассмотрим случай, когда присутствует только расстройка по фазе ($\Delta \text{ft} = 0$). Восстановленный сигнал примет вид:

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

Результат совпадает со случаем идеальной демодуляции за исключение того, что теперь присутствует дополнительное масштабирование на $\cos(\Delta\phi)$. Так как модуль косинуса меньше единицы, мы получаем дополнительное ослабление сигнала. В крайнем случае, когда $\Delta\phi = 90$ градусов, сигнал на выходе ФНЧ будет равен нулю.

Теперь рассмотрим, к чему приведет ненулевая расстройка по частоте. На выходе ФНЧ получим:

$$s_{\text{LPF}}(t) = \frac{1}{2} m(t) \cdot \cos(2\pi \Delta \text{ft}).$$

Таким образом, восстановленное сообщение масштабируется на множитель $\cos(2\pi \Delta \text{ft})$, амплитуда которого будет изменяться с частотой Δf. 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 оскажения сигнала.

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

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

% дополнительные расчеты
SignalFs = AudioReader.SampleRate;              % получаем частоту дискретизации модулир
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'}, ...
```

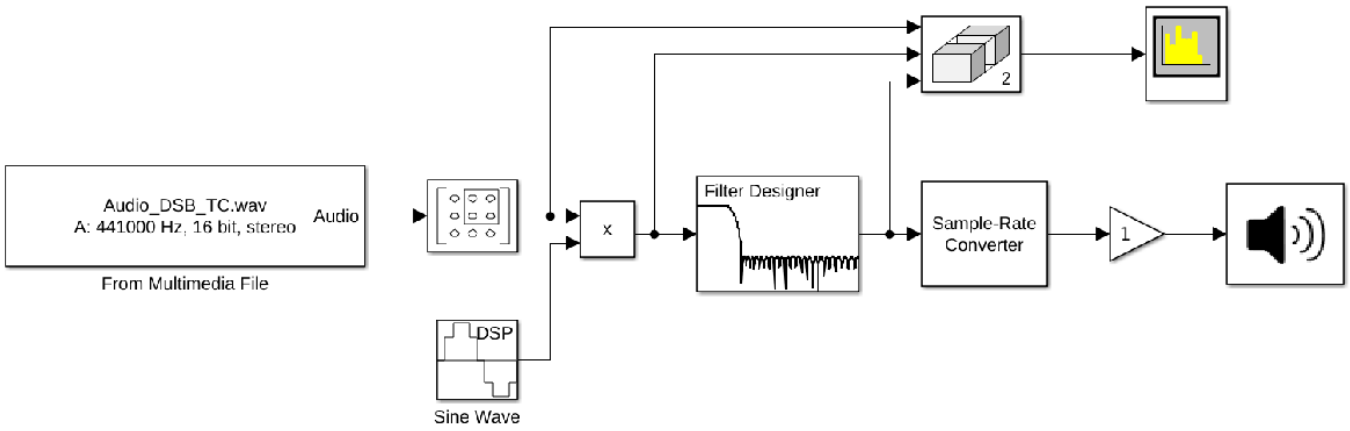
```
    'SampleIncrement', SignalFs/SignalFrameSize ...
    );

Message = [];

% запуск симуляции
for i = 1:FramesNumber
    % считывание отсчетов AM-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 АМ-сигнала. Синим - спектр на выходе смесителя. Можно увидеть, что он состоит из информационного сообщения и сигнала на удвоенной частоте несущей. После фильтрации остается только информационное сообщение (красный цвет).

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

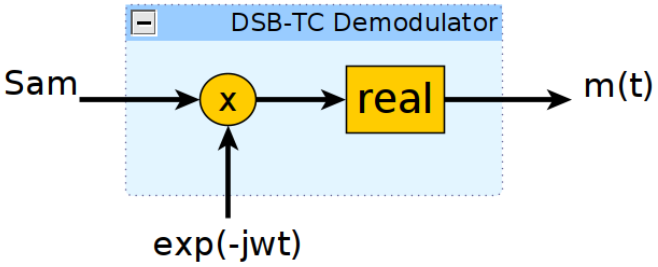
На слух можно оценить искажения аудиосигнала из-за рассогласования по фазе и частоте. При расстройке по фазе сигнал слышится слабее, но он не искажен. Расстройка по частоте приводит к ухудшению качества и появлению свистящих звуков.

В файле *DSB_SC_Receiver_Part_1_1.slx* представлена Simulink модель когерентного демодулятора, который ранее был реализован ввиде скрипта.



2. Когерентная демодуляция при квадратурном приеме

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



То есть, принятый АМ-сигнал, который является комплексным, нужно умножить на мнимую экспоненту, а потом просто выделить действительную часть.

В аналитическом виде описанные выше преобразования можно представить следующим образом. Пусть принятый АМ-сигнал имеет вид:

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

где, как и ранее, $m(t)$ - информационное сообщение, f_c - частота несущей.

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

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

В идеале сигнал после умножения получится действительным и будет состоять только из информационного сообщения. Поэтому выделение действительной части не внесет в сигнал никаких изменений:

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

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

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

После вычисления действительной части получим:

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

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

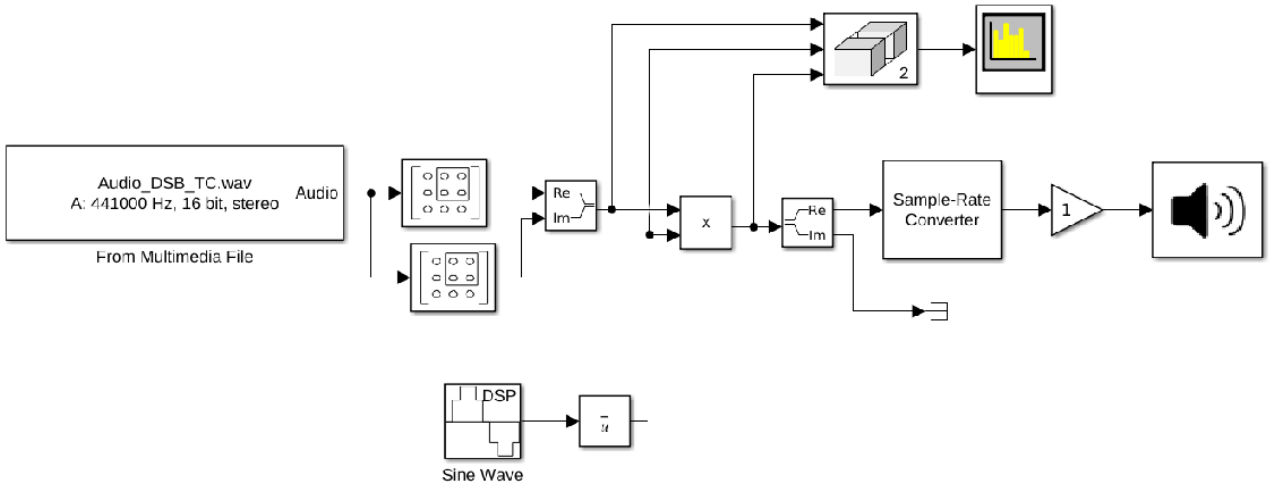
```
clc; clear; close all;

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)

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

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

% генератор несущей
Carrier = dsp.SineWave(...
    '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 АМ-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 *DSB_SC_Receiver_Part_1_2.slx* представлена Simulink модель когерентного демодулятора в квадратурном случае.



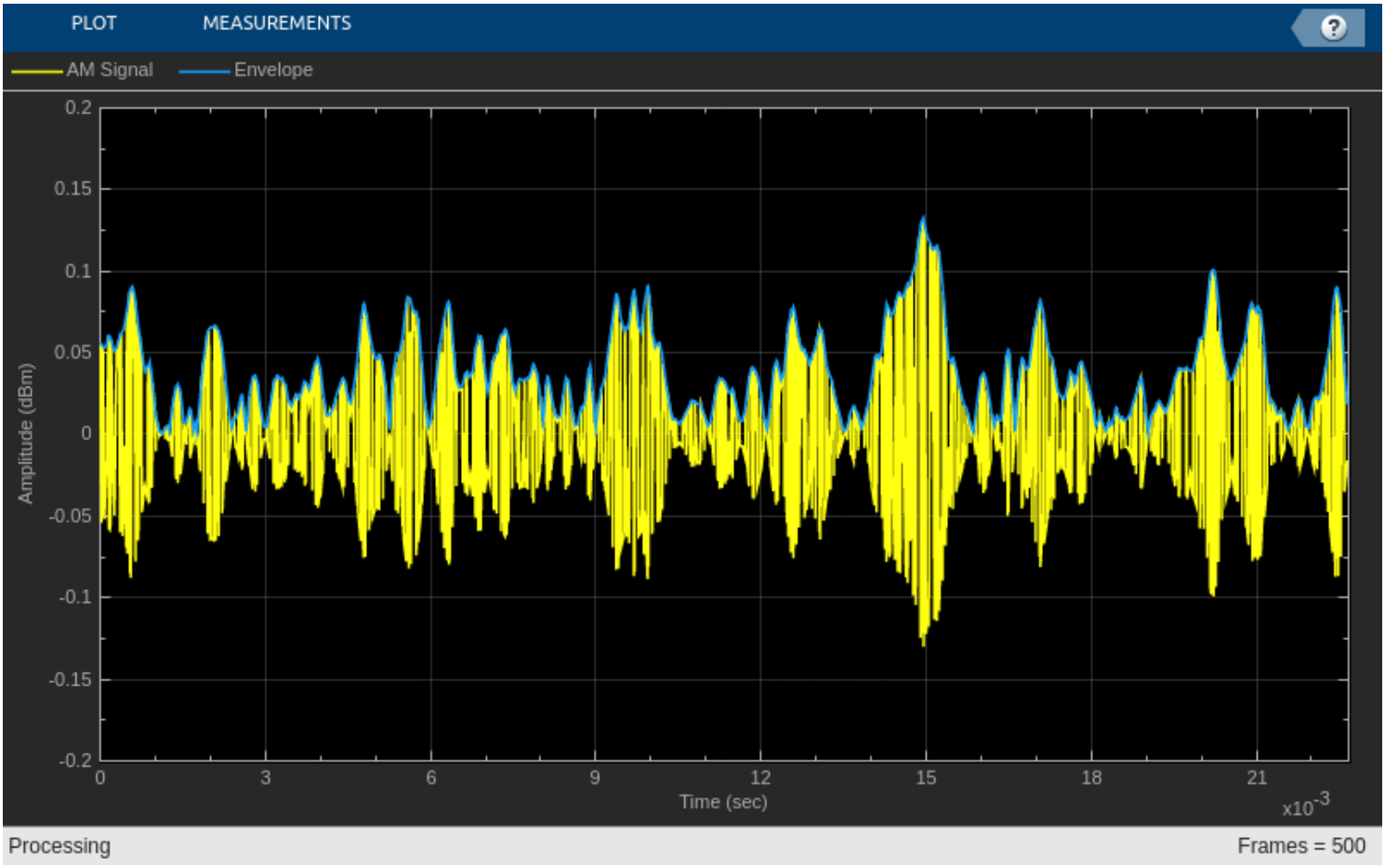
3. Некогерентная демодуляция АМ-сигнала с подавленной несущей

В случае 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. Если отсчеты информационного сообщения изменяют знак, то есть не выполнено условие $m(t) \geq 0$, то огибающая сигнала больше не совпадает с информационным сообщением. Таким образом, некогерентный демодулятор для приема сигнала с подавленной несущей применять нельзя.

Убедимся в этом на примере для случая квадратурного приема. Для получаения 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

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

% дополнительные расчеты
SignalFs = AudioReader.SampleRate;              % получаем частоту дискретизации модулир
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 AM-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,принятом сообщении действительная составляющая) и выходе (синий цвет) демодулятора. Огибающая принимает всегда положительные значения, а значит не совпадается с информационным сообщение в моменты времени, когда $m(t) < 0$.

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

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

4. Когерентная демодуляция с помощью обычной фазовой автоподстройки частоты

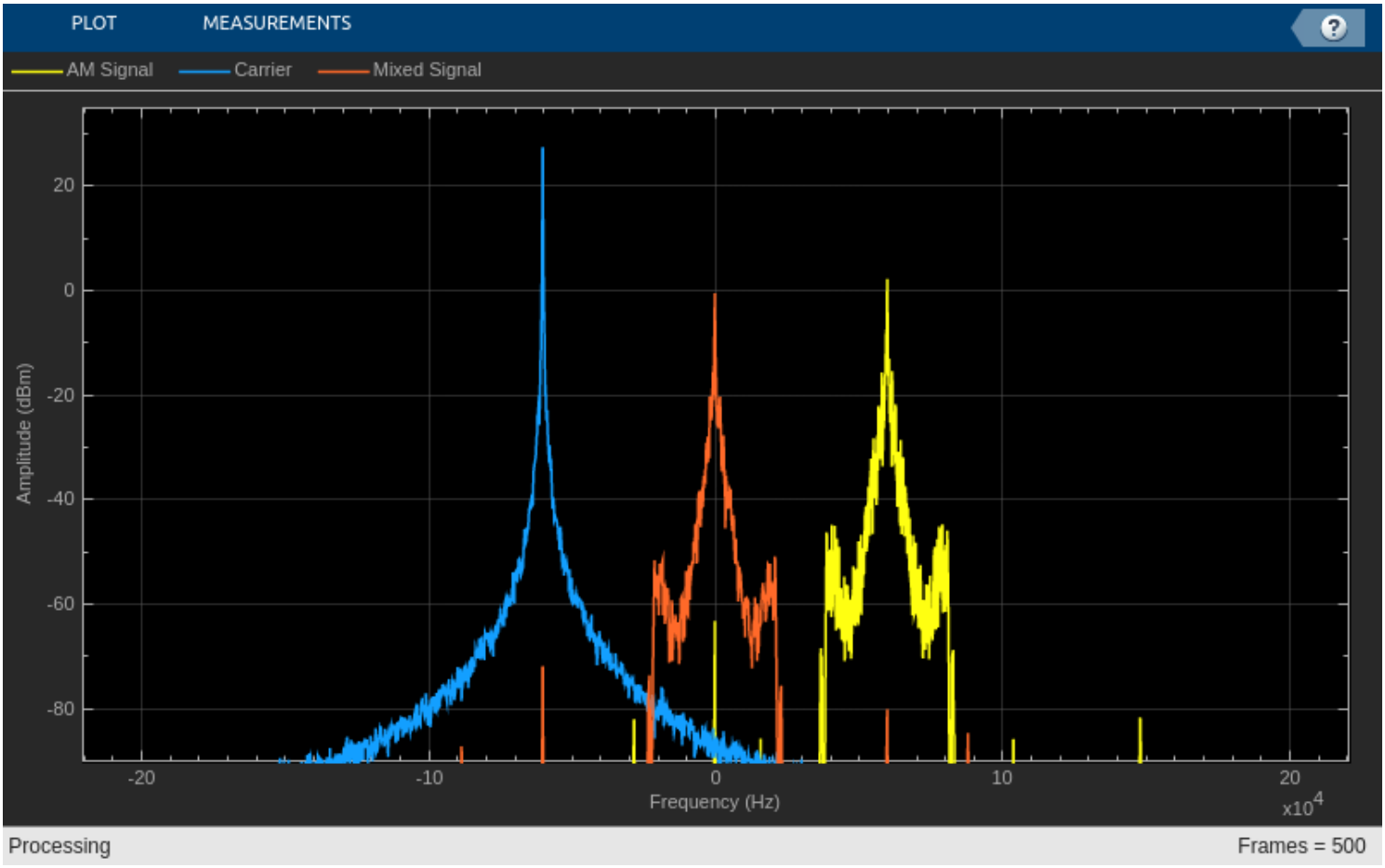
Рассмотренный в DSB TC Receiver Part 2 способ востановления несущей с помощью PLL для сигнала с подавленной несущей так же не применим. В принимаемом сигнале отсутствует в явном виде гармонический сигнал соответствующий несущей. Из-за этого фазовая автоподстройка частоты не может войти в режим синхронизации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

Убедимся в этом на примере для случая квадратурного приема. Воспользуемся тем же скриптом, который рассмотривается 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;

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

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

% фапч 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 AM-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 сигнал на вход PLL, не получится. Один из возможных вариантов - предварительно возвести сигнал в квадрат. Другой способ приема сигнала с подавленной несущей - схема Костаса.

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