Отчет: частотно-временная синхронизация

Все изменения в коде производились в Synchronization и Experiment's parameters.

Experiment's parameters

Частота расстройки увеличина до 10кГц.

Synchronization

Для обеспечения частотной синхронизации необходимо компенсировать частотную расстройку. Для этого был задействован вектор freqSeq, который представляет собой комплексную экспоненту с частотой, заданной переменной toneFreq. Особенность freqSeq состоит в том, что разность фаз между соседними отсчетами постоянна, что позволяет нам определить грубую временную синхронизацию.

Определим разность фаз между последовательными элементами векторного сигнала rxSignal, получив phaseDifferences. Далее посчитаем скользящее среднее movingAvg с окном windowSize, равным размеру freqSeq, для определения ярко выраженного пика, который указывает на полное попадание в окно вектора freqSeq. Затем найдем индекс максимального значения movingAvg:

```
% Phase Difference Calculation
phaseDifferences = angle(rxSignal(2:length(rxSignal)) .* conj(rxSignal(1:length(rxSignal)-1)));

% Calculate moving average of phaseDifferences using a window size equal to length of freqSeq
windowSize = length(freqSeq);
movingAvg = movmean([zeros(1, windowSize), phaseDifferences, zeros(1, windowSize)], windowSize);

% Peak movingAvg finding
[~,movingAvgMaxIdx] = max(abs(movingAvg));
```

Ввиду того что мы для корректного подсчета скользящего среднего дополнили phaseDifferences нулями, а вектор rxSignal может иметь меньше число элементов, чем скользящее среднее, сделаем проверку:

```
% Skip it if we went out of the array rxSignal
if ( movingAvgMaxIdx + windowSize/2 > length(rxSignal))
    syncFailed(seedIdx, snrDbIdx) = 1;
    continue;
end
```

Далее выполним анализ в частотной области. Вектор freqSeq несет в себе информацию о расстройке частоты. Поэтому вычислим длину необходимой нам freqSeq последовательности, для этого вычтем из найденного ранее индекса число нулей. Далее вычислим fft, найдем индекс максимального значения, согласуем отсчеты и частоты и, наконец, вычислим частотный сдвиг (расстройку по частоте):

```
% FFT Analysis
fftLength = movingAvgMaxIdx - windowSize/2;
fftAmplitudeSignal = abs(fft(rxSignal(1:fftLength)));

% Peak fft signal finding
[~, fftAmplitudeSignalMaxIdx] = max(fftAmplitudeSignal);

% Finding the shift
fftFrequencies = (0:fftLength-1) * (sampleRate / fftLength);
frequencyShift = fftFrequencies(fftAmplitudeSignalMaxIdx) - toneFreq;
```

Остается только скомпенсировать данный частотный сдвиг путем умножения принятого сигнала rxSignal на комплексную экспоненту:

```
% Frequency compensation
rxSignal = rxSignal .* exp(1j*2*pi*(-frequencyShift)/sampleRate*(1:length(rxSignal)));
```

Полученный результат:

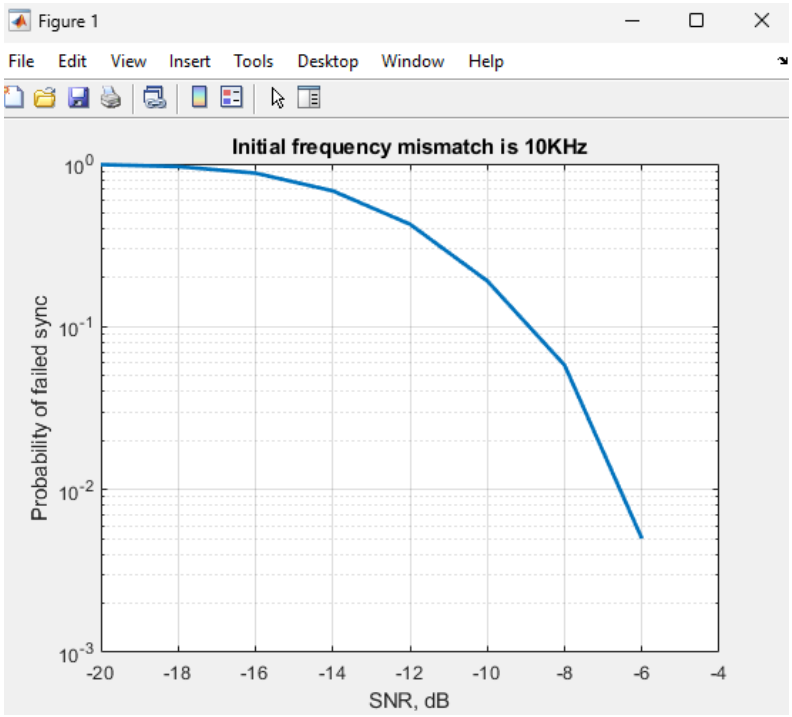

Рис. 1: Зависимость вероятности неуспешной синхронизации от SNR при частотной расстройке 10КГц и при числе сидов 1000

График не отображает значения дальше -6 дБ, из-за нулевой вероятности ошибочной синхронизации, поэтому увеличим выборку seedCount до 100000 и увидим дальнейшее уменьшение вероятности:

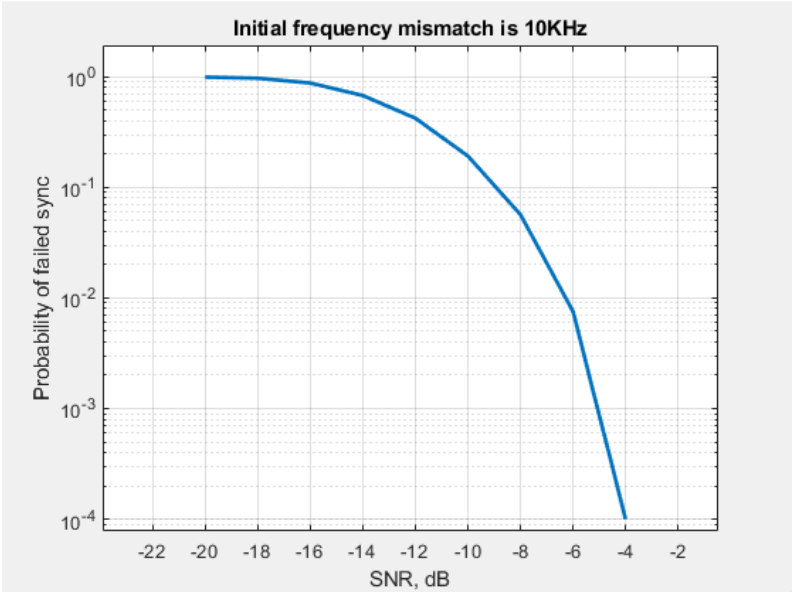

Рис. 2: Зависимость вероятности неуспешной синхронизации от SNR при частотной расстройке 10КГц и при числе сидов 100000