OFDM Systems:
Impact of Phase Noise on OFDM

209AS: Special Topics in Circuits and Embedded Systems

Silei Ma (UID: 604741646)
Pedro Vazquez Sierra (UID: 304775510)

Table of Contents:

Table of Figures:

Introduction

There are two effects that occur if the phase noise is present in an OFDM system:

1. Rotation of all demodulated subcarriers of an OFDM symbol by a common angle (common phase error), which comes from the DC value of the phase noise
2. Occurrence of the intercarrier interference, which comes from the deviations of the phase noise during one OFDM symbol from its DC value.

The more the information about the phase noise, the better it can be reduced. Therefore, in order to minimize the phase noise, it is necessary to get as much information about the phase noise waveform as possible. One of the simplest approaches would be to approximate the phase noise to a constant value, for instance its mean.

In this project, ‘small’ phase noise is not considered and we use the Wiener process phase noise model, which is found to be an appropriate description for the phase noise in oscillators.

Objective

The aim of this project is to analyze the impact of phase noise on OFDM systems. Local oscillators introduce phase noise that creates inter-carrier interference. The tasks of this project are the following ones:

The first step is to build an OFDM system. For this task, an OFDM transmitter with BW = 20MHz, 64 subcarriers, and CP length of 16 is considered. The channel is an AWGN channel.

The second step it to model the phase noise. Phase noise is introduced in the OFDM receiver. $\Delta \mathrm{f}_{3\mathrm{dB}} = \{50, 100, 150\}$ is considered.

The third step is to design a phase compensation algorithm.

Finally, it is necessary to study the symbol error rate with variations of SNR over [0,30] dB in the presence and absence of the compensation algorithm to evaluate the results.

System model:

Phase noise model is modeled as follows:

$\mathrm{r(n)} = \left(\mathrm{x(n)} \star \mathrm{h(n)}\right)\mathrm{e}^{\mathrm{j}\phi(\mathrm{n})} + \xi(\mathrm{n})$, assuming perfect frequency and timing synchronization.

$\mathrm{x(n)}$: Samples of the transmitted signal

$\mathrm{h(n)}$: Channel impulse response

$\phi(\mathrm{n})$: Phase noise process at the output of the mixer

$\xi(\mathrm{n})$: AWGN Noise

$\star$: Convolution

The discrete time equation for the Wiener phase noise process equation is:

$\phi(\mathrm{n}+1) = \phi(\mathrm{n}) + w(n)$

$\phi(\mathrm{t})$: Is modeled as Wiener process and denotes the phase noise process at sampling instant $\mathrm{nT_s}$, where $\mathrm{n} \in \mathbb{Z}$

$w(n)$: Gaussian random variable; $w(n) \sim N(0{,}4\pi\Delta f_{3dB}T_s)$

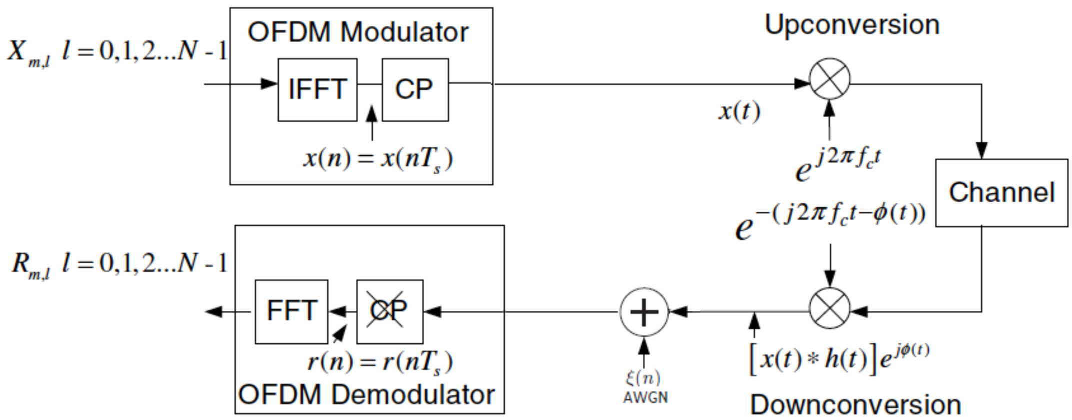


Figure 1: Block diagram of an OFDM transmission chain

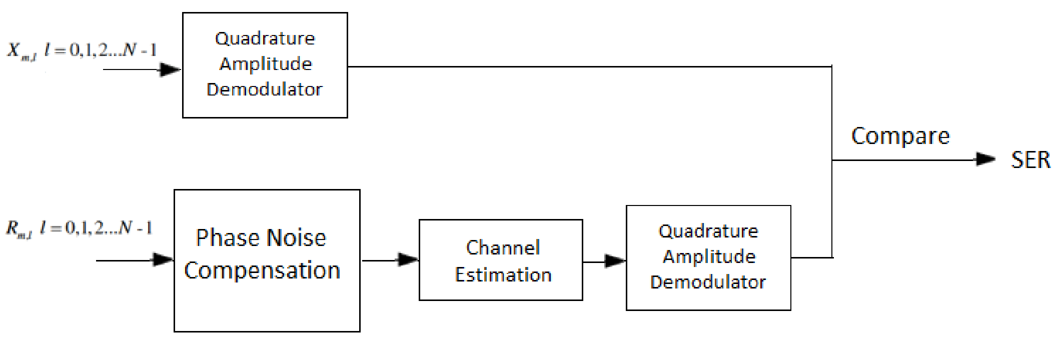


Figure 2: Symbol Error Rate Measurement Model

Compensation algorithm:

Phase noise can be decomposed in to Common Phase Error (CPE) and Intercarrier Interference (ICI). **We have estimated that using pilots in channel estimation can automatically remove CPE. Our algorithm will mostly focus on ICI compensation.**

These are the coefficients to consider for the ICI compensation algorithm:

1. u: Compensation order (up to K/2).

2. $R_{I_m I_m}$: K * K matrix

$$R_{I_m I_m}(n,p) = E\{I_m(n) I_m^*(p)\} = \frac{1}{K^2} \sum_{k=0}^{K-1} \sum_{l=0}^{K-1} e^{-\frac{|k-l|\sigma_W^2}{2}} e^{-j\frac{2\pi}{K}(nk-pl)} = \frac{1}{K^2} F(n,-p)$$

Where $F(n,p)$ represents the two-dimensional Discrete Fourier Transform of $e^{-\frac{|k-l|\sigma_W^2}{2}}$

3. $R_{J_m J_m}$: (2u+1) * (2u+1) matrix. Selected the (K/2-u)[th] to the (K/2+u)[th] rows and columns of $R_{I_m I_m}$

4. A_m: (K-2u) * (2u+1) matrix:

$$\underbrace{\begin{bmatrix} a_{l1} \\ a_{l2} \\ \cdot \\ \vdots \\ \cdot \\ a_{lk} \end{bmatrix}}_{A_m} \quad \text{where} \quad a_{l_i} = \begin{bmatrix} H_{m,l_i} X_{m,l_i} \\ H_{m,l_i-1} X_{m,l_i-1} \\ H_{m,l_i+1} X_{m,l_i+1} \\ \cdot \\ \cdot \\ \cdot \\ H_{m,l_i+u} X_{m,l_i+u} \end{bmatrix}^T$$

5. $R_{\varepsilon_m \varepsilon_m}$: (K-2u) * (K-2u) diagonal matrix:

$$R_{\varepsilon_m \varepsilon_m} = diag(E\{|\zeta'_{ICI}(l_1)|^2\} + \sigma_n^2, \ldots, E\{|\zeta'_{ICI}(l_k)|^2\} + \sigma_n^2)$$

Where k=K-2u and $E\{|\zeta'_{ICI}(q)|^2\} = \sum_{\substack{v=-\frac{K}{2} \\ v>|u|}}^{K/2-1} E\{|I_m(v)|^2\}$, which equals to the sum of v[th] diagonal elements of $R_{J_m J_m}$.

6. M: (2u+1) * (K-2u) matrix:

$$M = R_{J_m J_m} A_m^H (A_m R_{J_m J_m} A_m^H + R_{\varepsilon_m \varepsilon_m})^{-1}$$

7. $\tilde{J}_m$: (2u+1) dimensional vector:

$$\tilde{J}_m = M R_m$$

8. $\widetilde{\mathrm{U}}_{\mathrm{m}}$: (2u+1) dimensional vector (different from paper). It can be obtained by flipping the order of the conjugate of $\tilde{\mathrm{J}}_{\mathrm{m}}$.

Instead of doing circular convolution $\tilde{R}_{m,N} = R_{m,N} \odot \widetilde{U}_m$, we use another way to estimate:

$$\tilde{R}_{m,l} = A_m \widetilde{\mathrm{U}}_{\mathrm{m}}$$

Where $l = l_1, l_2, \dots, l_k$. Other samples remain unchanged.

Development

These are the OFDM parameters:
Number of subcarriers, K = 64
The modulation level of QAM is M=64
Number of cyclix prefix samples is assumed to be CP=16
Number of symbols = 1000
Number of Bits per branch is calculated as log2(M)*(Nº Symbols)
Band Width = 20MHz
Pilot starts in 4 with 8 of gap

Transmitter:

The first step is to develop random numbers, for this we use the following function in Matlab for all the branches:

```
B = randi([0 1],NoBits,K);
```

After generating the random bits, we use the modulation level:

```
X = qammod(B,M,'UnitAveragePower',true,'InputType','bit');
```

Here we take advantage of the qammod function of Matlab to implement it directly, normalizing the unit of power and establishing byte as the input type.

The next step is to insert pilots, considering the start point as 4 and a gap between points of 8:

```
for s=1:NoSym
    Xp(s,:)=X(s,:);
    samples=X(s,:);
    pilot=(1+1j)/abs(1+1j);
    for i=start:gap:K
        samples(i)=pilot;
        Xp(s,i)=pilot;
    end
```

After this, it is necessary to use IFFT (Inverste Fourier Function Transformation) over the samples, which contains the pilots, with the size of the subcarriers to get x. After this, we add cyclix prefix by copying the last points of the OFDM symbol at the beginning. This will be the OFDM symbols. Finally we reshape them into series and we give the unit of energy to x, so the calculations in Matlab become faster.

```
    samples_IFFT = ifft(samples,K);
    samples_CP = [samples_IFFT(K-CP+1:K) samples_IFFT];
    OFDMsymbols(s,:) = samples_CP;
end
OFDMSeries =  reshape(OFDMsymbols,1,[]);
OFDMSeries = OFDMSeries./sqrt(mean(abs(OFDMSeries).^2));
```

Channel:

The multipath considered is the following one:

```
h = [1];
```

The output of the channel is the convolution of the series and the multipath considered:

```
channel_output = conv(OFDMSeries,h);
```

To moderate the noise power:
By theory we know that SNR=(Signal power/Noise power). We fixed the signal power to be 1, so (Signal power) = (1/SNR).

Therefore, this is the code for implementing the AWGN noise and the phase noise.
The Gaussian noise, wNoise, is assumed to be a random number with real part and imaginary part considering the mean and the length of the channel:

```
% AWGN
SNR_dB = SNR;
SNR_lin = 10^(SNR_dB/10);
noisePower = sum(abs(h).^2)./SNR_lin;
wNoise =
sqrt(0.5*noisePower).*randn(1,length(channel_output))+1j*sqrt(
0.5*noisePower).*randn(1,length(channel_output));

% Phase Noise
for s = 1:NoSym
    phi(1)=0;
    for n=1:(K+CP)
        Ts=1/BW;
        w(n) = normrnd(0,sqrt(4*pi*f3dB*Ts));
        phi(n+1) = phi(n) + w(n);
        PN(s,n)=exp(1j*phi(n));
    end
end
pNoise =  reshape(PN,1,[]);
```

The output signal is the following one:

```
y = channel_output.*pNoise + wNoise;
```

Receiver:
The first thing is to reshape the samples and make them parallel while removing the cyclic prefix imposed before:

```
samples_parallel = reshape(y(1:NoSym*(K+CP)),NoSym,[]);
samples_noCP = samples_parallel(:,CP+1:end);
```

After this, we do the FFT for the output.

For the phase noise compensation, we have followed the algorithm explained in compensation algorithm point.

Once the phase noise compensation is established, the next step is to program the channel estimation in the pilots selected and the symbol estimation for all the subcarriers using interpolation (it can also remove CPE):

```
for s=1:NoSym
    for i=start:gap:K
        channelEstimate(i)=pilot'*Y(s,i)./power(abs(pilot),2);
        channelEstimate((i-
(gap/2)+1):(i+(gap/2)))=channelEstimate(i);
    end
```

```
    % Symbol estimation
    for i=1:K
        X_hat(s,i) =
channelEstimate(i)'*Y(s,i)./power(abs(channelEstimate(i)),2);
    end
end
```

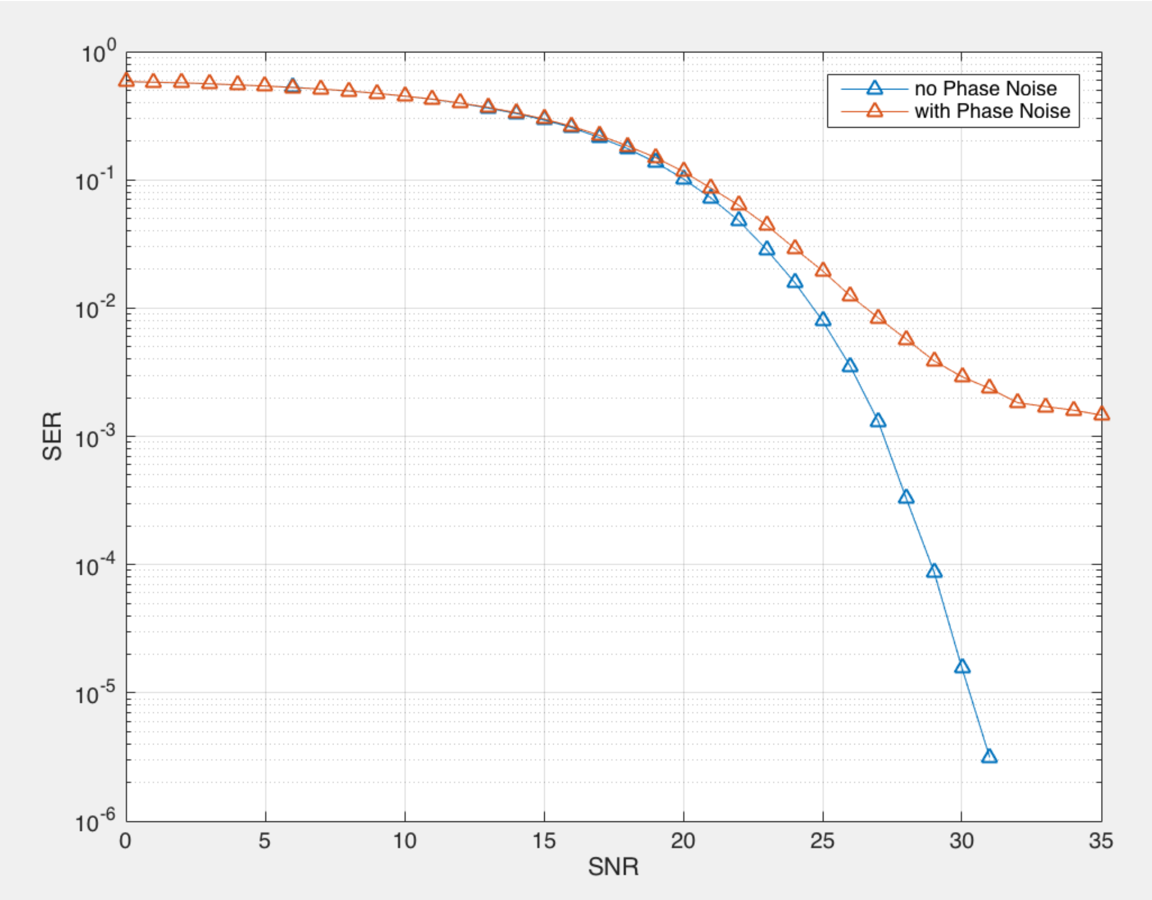
The next step is to demodulate using qamdemod to get the signal error ratio (SER):

```
XI_hat=qamdemod(Xp,M,'UnitAveragePower',true,'OutputType','int
eger');
RI_hat=qamdemod(X_hat,M,'UnitAveragePower',true,'OutputType','
integer');
```

The final step to access to the results is to plot the constellation diagram using ‘scatterplot’ and to measure the SER.

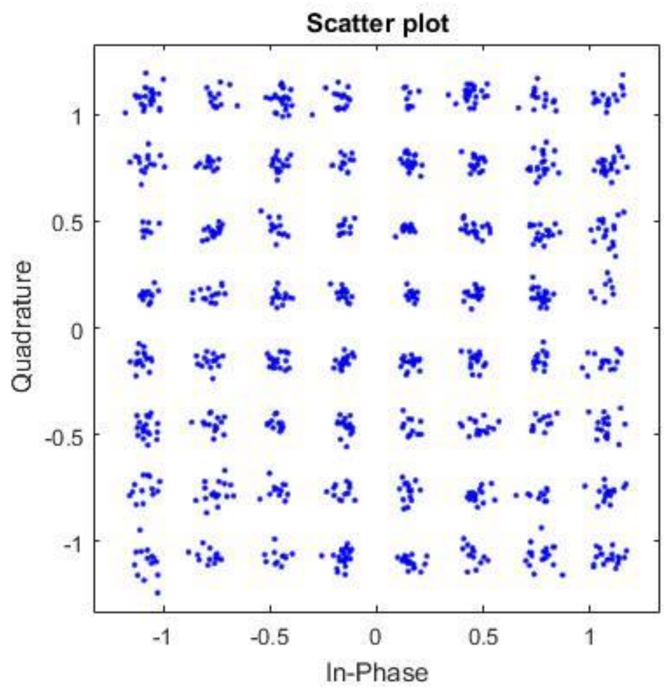
Results

In the following image, it is possible to see the difference between SER vs SNR before and after adding the phase noise.

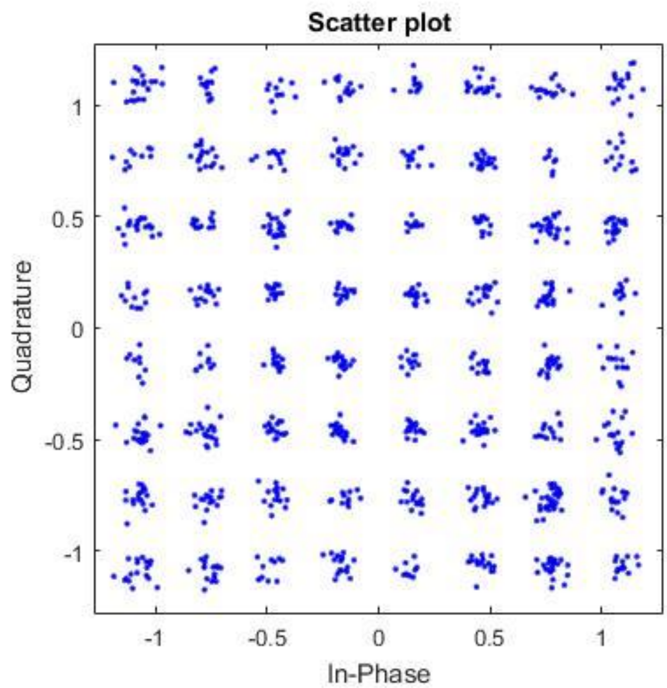


In the following two images it is possible to observe the constellation diagrams with and without compensation for the phase noise:

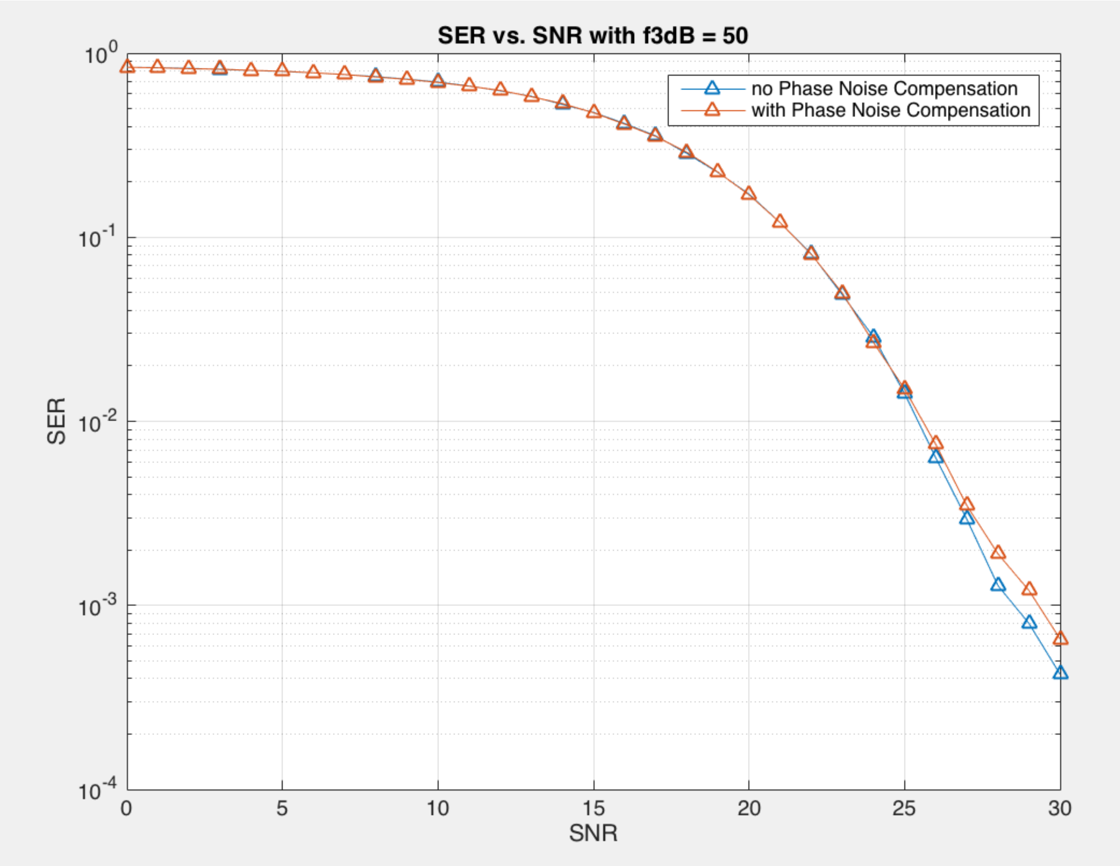
With no compensation

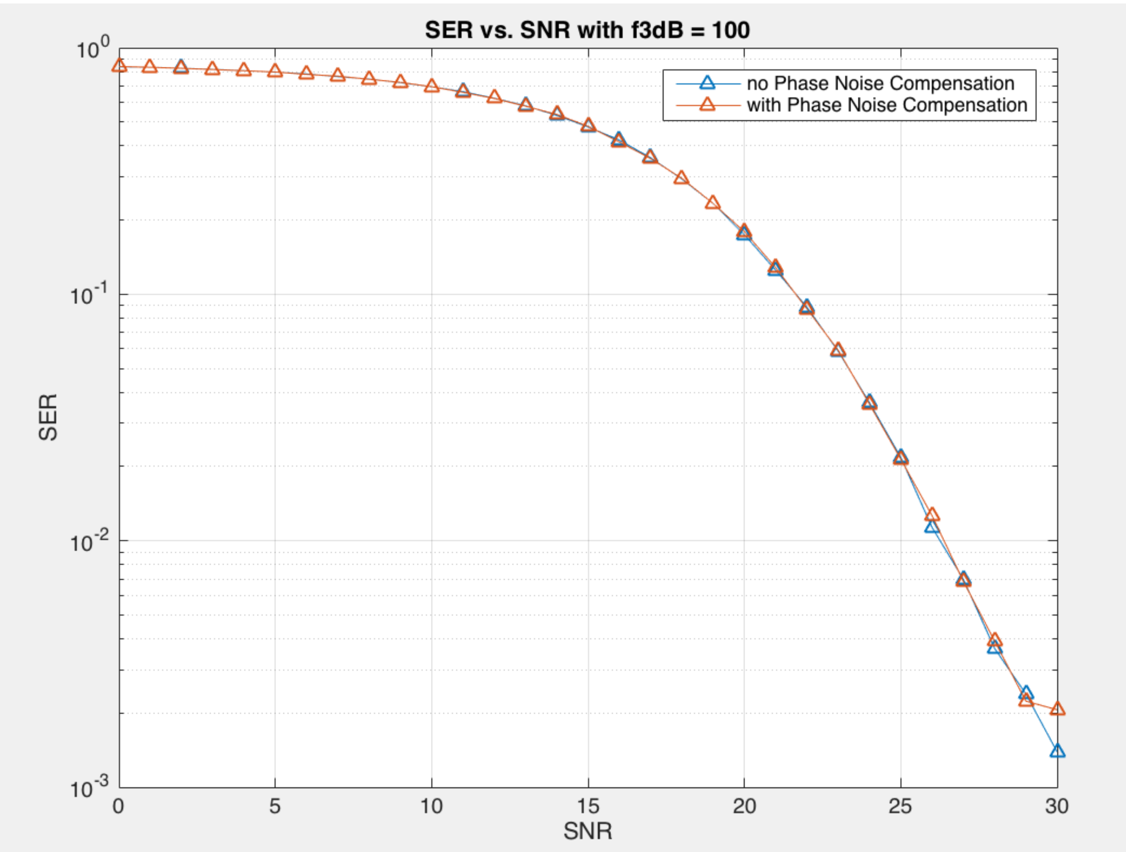


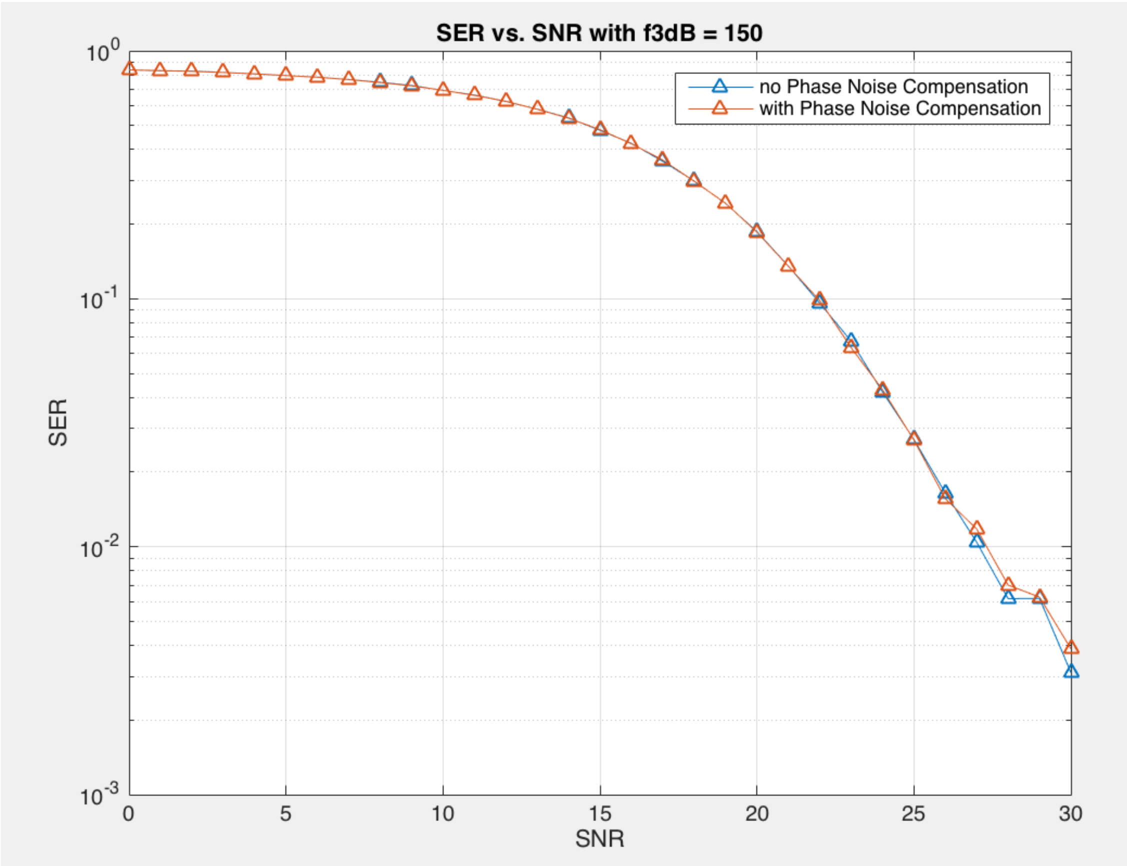
With compensation



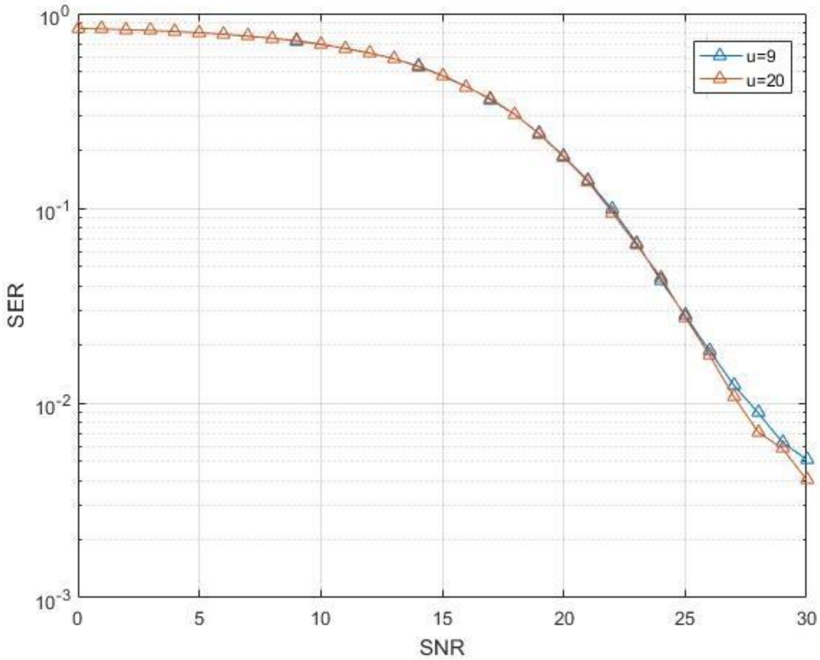
In the following images, it is possible to observe SER vs SNR for different values of phase noise:



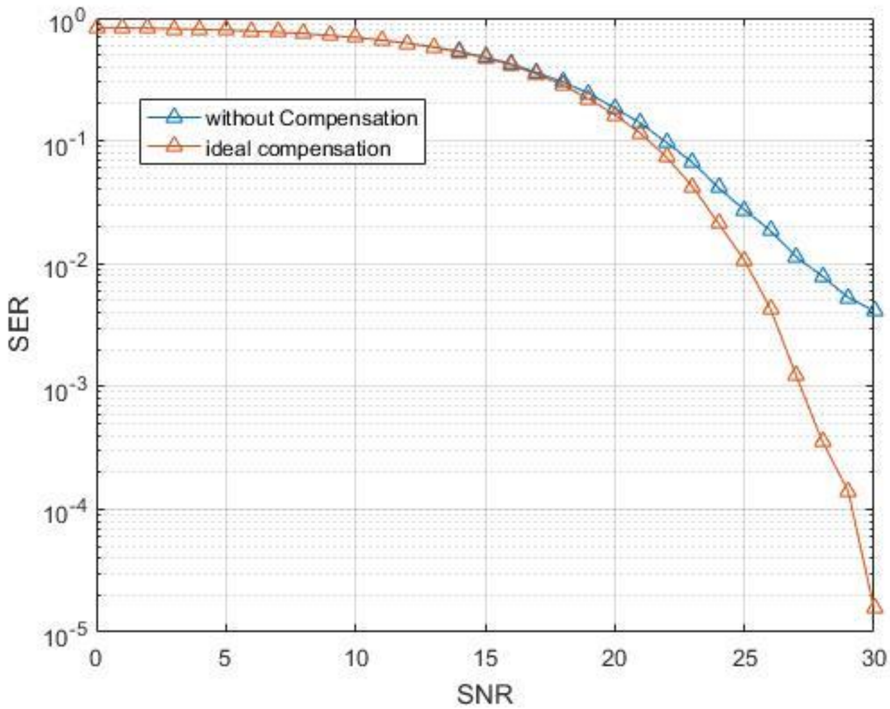



SER vs. SNR with f3dB = 150
no Phase Noise Compensation
with Phase Noise Compensation
SER
SNR
10^0
10^{-1}
10^{-2}
10^{-3}
0
5
10
15
20
25
30

In the following image, it is possible to observe that with higher order of composition we get larger reduction of SER, especially when the SNR is high as well.



The next figure shows the SER vs SNR of the ideal compensation of the phase noise in the case that we know the value of the phase noise:



Conclusions

After establishing the compensation algorithm, the SER doesn’t decrease much and even increases a little compared with the original value in high SNR region, which is not as we expected. Since we are not supposed to know either the input signal or the value of phase noise we applied, the algorithm in the paper is more like to compensate unknown phase noise with phase moving which has the same variance. In our model, it can be understood as introducing a “phase noise” again. Because the value of phase noise is random, we can sometimes observe a reduction of SER, sometimes observe an increment.

Another reason could be a misunderstanding in some equations in the reference used for the phase noise compensation due to the rough organization of the paper. In the results section, it is possible to observe the ideal case of how this compensation should have been.

Even though we have not been able to find the ideal compensation we consider that the learning process followed to build the system and build the phase noise compensation algorithm has been great overall. These factors lead us to know that the algorithm used is not completely correct. However, we will continue developing the algorithm outside the class to find the right solution to this issue.