A Project Report

On

IMPLEMENTATION OF INDOOR FSO COMMUNICATION IN MATLAB

BY

Arnav Tripathi

2022AAPS0501H

Under the supervision of

BALASUBRAMANIAN MALAYAPPAN

SUBMITTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS OF

ECE F376: DESIGN PROJECT

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI (RAJASTHAN)

HYDERABAD CAMPUS

(MARCH 2025)

ACKNOWLEDGMENTS

I would like to take this opportunity to express my deepest gratitude to all those who supported and guided me throughout the journey of this project.

First and foremost, I am profoundly thankful to my project instructor, Prof. Balasubramanian Malayappan, whose expertise, thoughtful feedback, and unwavering encouragement have been instrumental in the successful completion of this work. His ability to explain complex concepts with clarity and his dedication to nurturing a spirit of inquiry and innovation in his students have greatly inspired me throughout this project.

A heartfelt thanks to my teammate, Souvik, whose collaboration and support proved invaluable during the project. His readiness to share resources and assist in tackling challenges strengthened the overall quality of our work. Working alongside him provided a collaborative environment that encouraged brainstorming and problem-solving.

Lastly, I extend my sincere appreciation to everyone who contributed to this project in any capacity. Whether through direct assistance, insightful discussions, or offering words of encouragement, your contributions have made a meaningful impact on this endeavour. Your collective efforts not only guided me in achieving the project's objectives but also motivated me to push the boundaries of my understanding and deliver my very best.

This project has been a rewarding experience, and I am truly grateful for the opportunity to learn, explore, and grow under the guidance and support of such exceptional individuals.

Thank you.

Birla Institute of Technology and Science-Pilani,

Hyderabad Campus

Certificate

This is to certify that the project report entitled "**IMPLEMENTATION OF INDOOR FSO COMMUNICATION IN MATLAB"** submitted by Mr. Arnav Tripathi (ID No. 2022AAPS0501H) in partial fulfilment of the requirements of the course EEE F376, Design Project Course, embodies the work done by him under my supervision and guidance.

Date:

Dr. BALASUBRAMANIAN MALAYAPPAN

Asst. Professor, EEE Dept., BITS- Pilani, Hyderabad Campus

ABSTRACT

This project explores the implementation of Indoor Free-Space Optical (FSO) communication systems using MATLAB simulation techniques. The research focuses particularly on analysing baseline wander phenomenon-a critical challenge in digital optical communication where the reference voltage level shifts unpredictably at the receiver end. Through systematic modelling and simulation, this work demonstrates how high-pass filtering, while effective for mitigating ambient light interference, inadvertently introduces baseline wander effects that can compromise signal integrity.

The project develops a comprehensive simulation framework incorporating transmitter impulse response, high-pass filtering, and matched filtering to quantify these effects under various operating conditions. By examining multiple system parameters including bit rate, sampling frequency, and sequence patterns, the research provides valuable insights into optimization strategies for indoor FSO systems.

The results reveal characteristic distribution patterns of matched filter outputs that illustrate how baseline wander distorts signal detection, particularly during extended sequences of identical bits, offering practical guidelines for improving signal detection reliability in indoor optical wireless communication applications.

CONTENTS

Baseline Wander without FLI

Introduction

In digital communication systems, baseline wander refers to the unwanted shifting of the reference voltage level that occurs at the receiving end. This reference level-often called the baseline-serves as the crucial threshold against which digital signals are interpreted as either high (1) or low (0). When this baseline unexpectedly fluctuates or drifts over time, it compromises the receiver's ability to accurately interpret the incoming digital information.

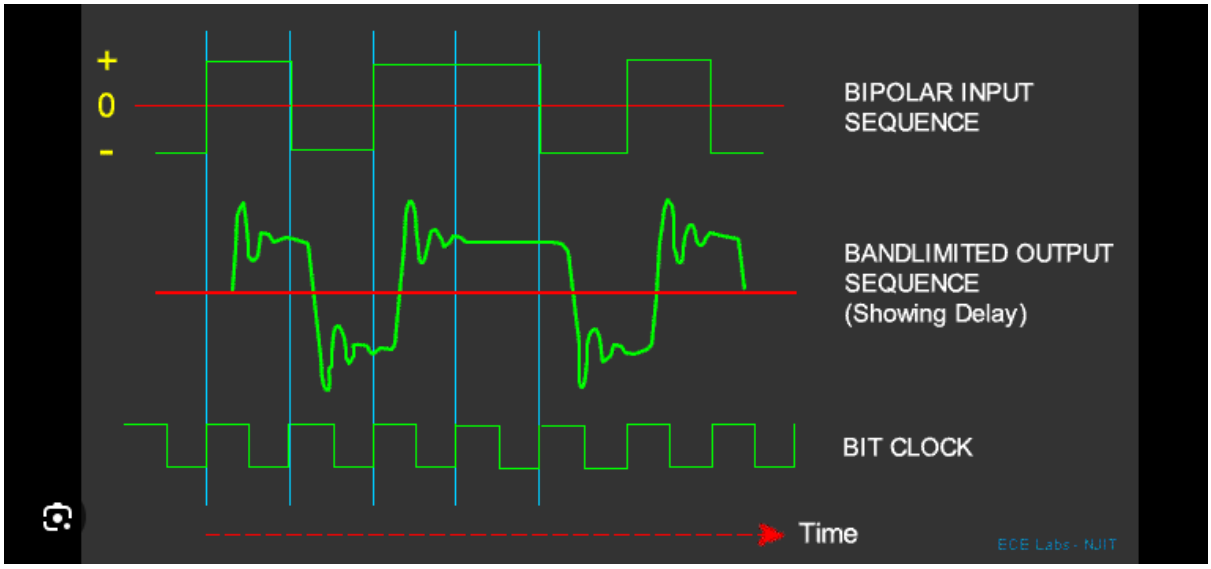


Figure 1 - Illustration of Baseline Wander

Since we know that utilisation of a High-Pass Filter is a common way to tackle interference caused by ambient light sources like the Sun, etc. Unfortunately, the presence of a high-pass filter causes baseline wander.

High-pass filters (HPFs) in digital communication systems trigger baseline wander, a gradual deviation in the signal's reference voltage by destabilising the balance between low-frequency signal content and the system's DC baseline. This drift emerges from three interrelated factors:

1. **HPF Frequency Characteristics**: AC-coupled systems block DC components through capacitors, creating a floating baseline dependent on the signal's average voltage.
2. **Data Pattern Dependency**: Extended sequences of identical bits (e.g., prolonged 0s or 1s) shift the cumulative DC offset, causing capacitors to charge/discharge slowly due to the HPF's time constant.
3. **Capacitor Limitations**: The HPF's cutoff frequency (f_c=1/2πRC) dictates how quickly the system adapts to DC changes. Lower *fc* values (e.g., 0.8 Hz) exacerbate drift by slowing baseline stabilisation.

We model the high-pass filter as a first-order RC filter, which has a 3-dB cut-on frequency and its impulse response is shown by *g(t).*

$$g_{\text{out}}(t) = \begin{cases} Ae^{-t/RC} & 0 \le t \le \tau \\ -A(e^{-\tau/RC} - 1)e^{-t/RC} & t > \tau \end{cases}$$

Note, that this impulse response is given for a single rectangle pulse with amplitude A and duration of τ. τ also represents the 'time constant' of the circuit used to model the HPF.

$$\boldsymbol{\tau = RC = \frac{1}{2\pi f}}$$

When we pass a sequence of pulses through a HPF, we observe that the output is contributed to by each and every pulse of the sequence individually. This is due to the principle of superposition of linear systems. For n bits $A_1A_2..A_n$

$$g_{out}(t)\Big|_{t=n\tau} = \sum_{i=1}^{n} A_i (e^{-2\pi f_c \tau} - 1)(e^{-2\pi f_c \tau})^{i-1}$$

Figure 2-HPF output calculation

Modelling

The experimental simulation setup for observing the effects of baseline wander due to the utilisation of a HPF is given.

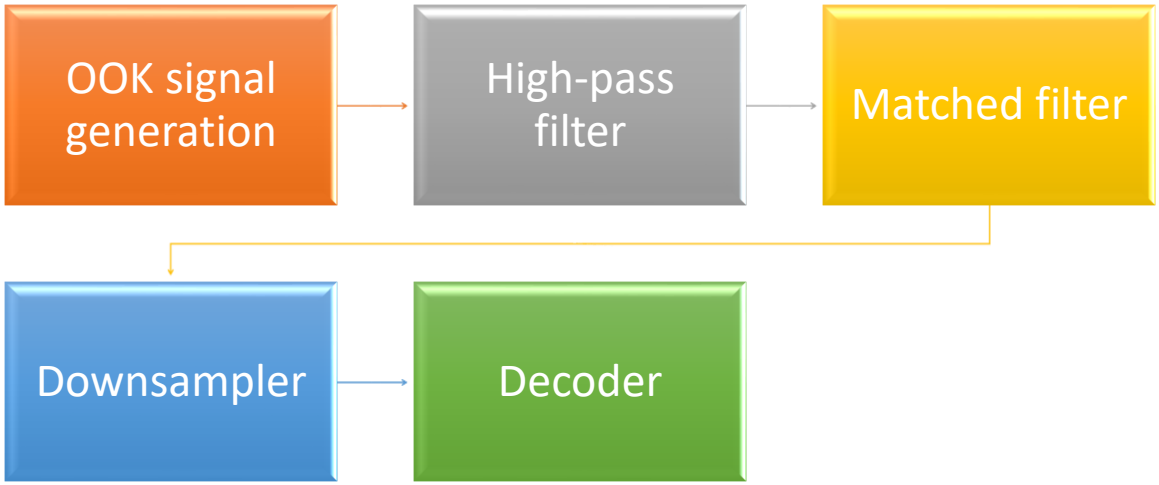


Figure 3- Block diagram of simulation

We use a HPF which has a cut-off frequency equal to 0.05*Rb (Rb = bitrate). The output of the HPF is fed to the matched filter. The MF has a threshold value is

$2 * R * Pavg\sqrt{T}$ and 0 for binary 1 and 0, respectively.

Note- Here R is the photodetector responsivity (conversion efficiency of light to current) and Pavg is the average received optical signal power.

We introduce a matched filter after the high-pass filter in order to improve the output signal so that it becomes less prone to decoding errors.

Working of a Matched Filter

In systems impacted by random additive noise, the matched filter represents the ideal linear filtering solution that optimizes the signal-to-noise ratio (SNR). This filter functions by performing correlation between a predefined signal pattern and an incoming unknown signal, enabling detection of the known pattern amid noise interference.

The filter effectively searches for specific signal "fingerprints" within noisy transmissions by maximizing the response when the incoming signal aligns with the anticipated waveform pattern, making it particularly valuable for digital communication systems where signal recovery must occur under non-ideal conditions.

From a mathematical perspective, this correlation technique is identical to applying a convolution between the incoming signal and the expected signal template that has been **time-inverted** and **complex-conjugated**.

```
%% Matched Filter (time-reversed version of tx filter)

matched_filter = tx_impulse(end:-1:1);
matched_output = filter(matched_filter, 1, hpf_output);
```

Figure 4-Code snippet showing MF implementation

The first line shows that “matched filter” is equated to the time inverted, ie. The sequence is now going from end to start instead of vice versa. The complex conjugate is the same as the original term as the term in question is purely real.

Simulation

```
%% Initialization
Rb = 1e3;                   % Bit rate (bits/sec)
Tb = 1/Rb;                  % Bit duration (seconds)
sig_length = 1e2;           % Number of bits
nsamp = 100;                % Samples per bit
Tsamp = Tb/nsamp;           % Sampling interval
Lsym = 1e2;                 % Number of symbols
fc_rb = 5e-2;               % Normalized cut-off frequency
A = 1;                      % Amplitude

%% Transmit Filter (Rectangular Pulse)
tx_impulse = ones(1, nsamp) * A;

%% HPF Design (Impulse Response)
fc = fc_rb * Rb;                        % Actual HPF cutoff (Hz)
t = Tsamp:Tsamp:10*Tb;                  % Time vector for HPF
hpf_impulse = zeros(1, length(t));  % Preallocate

hpf_impulse(1) = exp(-2*pi*fc*t(1));
for loop = 2:length(t)
    hpf_impulse(loop) = -1 * (exp(2*pi*fc*t(1)) - 1) * exp(-2*pi*fc*t(loop));
end

%% Generate OOK Signal
OOK = randi([0 1], 1, Lsym);      % Random binary sequence
OOK = 2 * OOK - 1;                % Map {0,1} → {-1,1}
signal = filter(tx_impulse, 1, upsample(OOK, nsamp));  % Rectangular pulse shaping

%% Pass through HPF (simulate AC coupling)
hpf_output = filter(hpf_impulse, 1, signal);

%% Matched Filter (time-reversed version of tx filter)

matched_filter = tx_impulse(end:-1:1);
matched_output = filter(matched_filter, 1, hpf_output);
%matched_filter = fliplr(tx_impulse);
%matched_output = filter(matched_filter, 1, hpf_output);  % Apply matched filter

%% Display first 20 symbols
disp('First 20 OOK symbols:');
disp(OOK(1:20));

%% Plotting
plotstart = 10*nsamp + 1;
plotfinish = plotstart + 15*nsamp;
t = 0:Tsamp:Tsamp*(plotfinish-plotstart);

figure;
subplot(411); plot(t, signal(plotstart:plotfinish), 'k');
title("Transmitted binary signal"); ylabel("Amplitude"); grid on;
```

```
subplot(412); plot(t, hpf_output(plotstart:plotfinish), 'k');
title("HPF output"); ylabel("Amplitude"); grid on;

subplot(413); plot(t, -signal(plotstart:plotfinish) + hpf_output(plotstart:plotfinish),
'k');
title("Baseline wander signal"); ylabel("Amplitude"); grid on;

%subplot(414); plot(t, matched_output(plotstart:plotfinish), 'k');
%title("Matched filter output"); xlabel("Time (s)"); ylabel("Amplitude"); grid on;
```

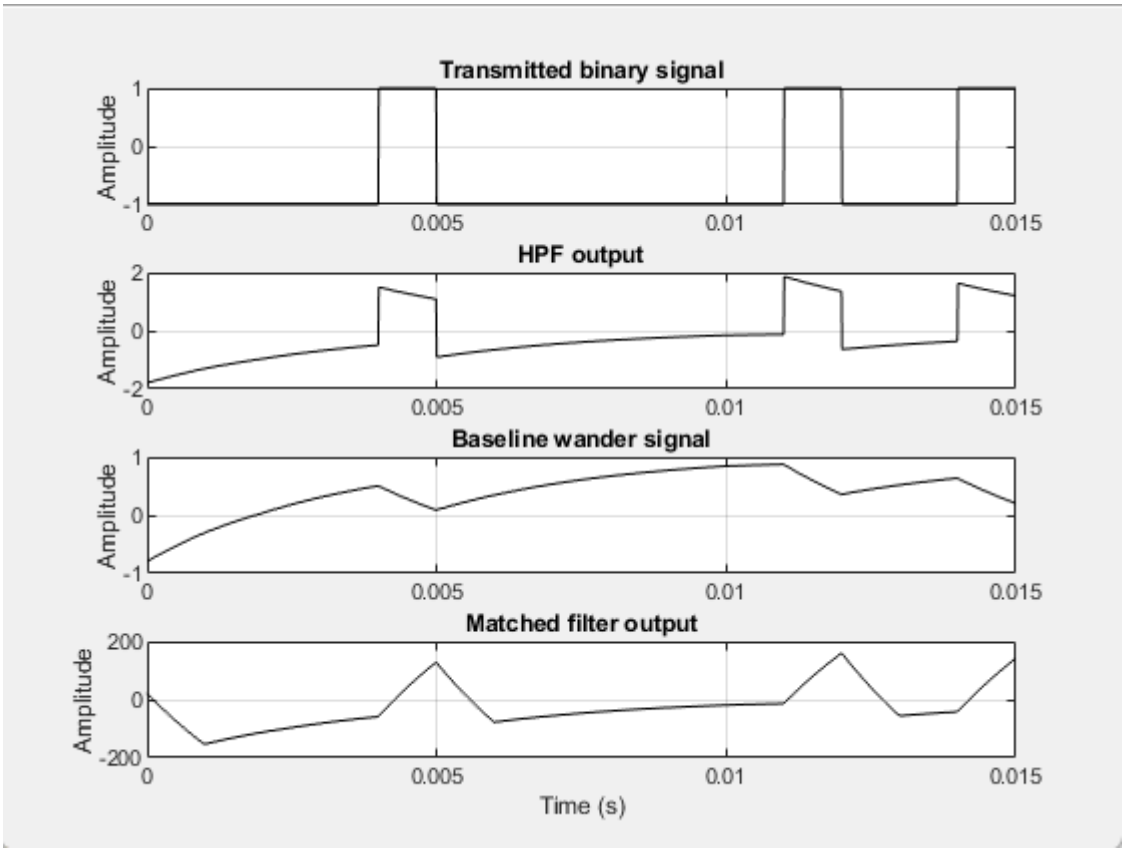


Figure 5-Program output

Observations

We observe from the above plots, the transmitted binary signal, the output at the HPF, the calculation and plotting of baseline wander and the matched filter output.

Even though it seems the HPF output alone is enough to decode the input signal with satisfactory accuracy, the problem arises when the input sequence is a long chain of 0s or 1s.

In that case, we observe, the baseline starts to change, that is, the baseline "wander" increases, if it reaches a sufficient level such that the input bit and received bit are on different sides of the decision threshold (signifying an error in detection).

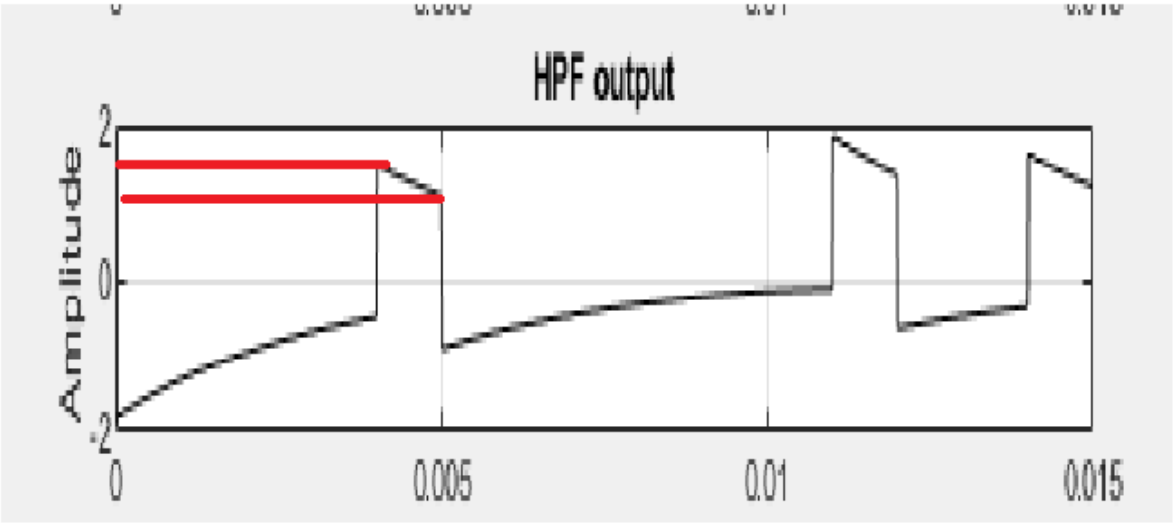


Figure 6-Flaws with HPF output

The drop in the HPF output(shown in the above figure) without there being a change from 0 to 1 or vice versa, is due to the long sequence of 0s or 1s.

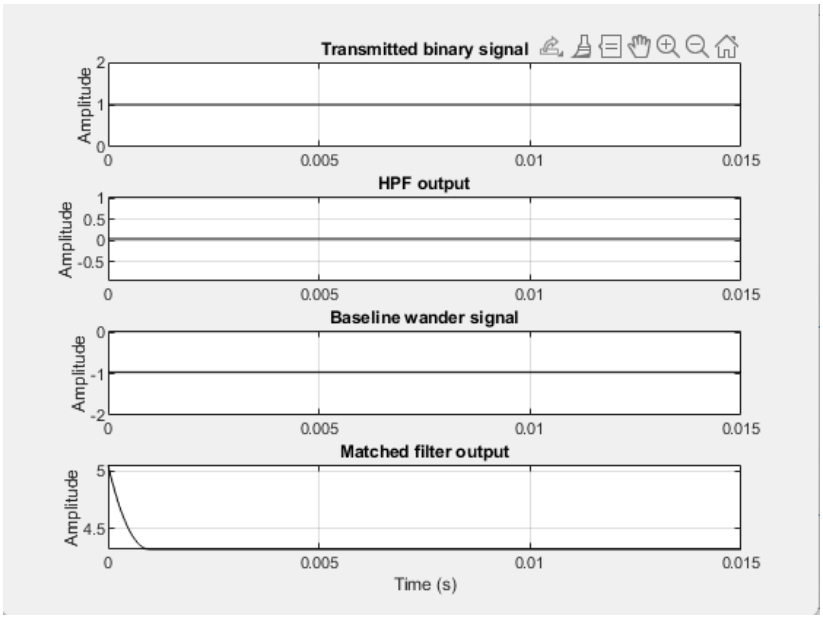


Figure 7- If input sequence is 1s

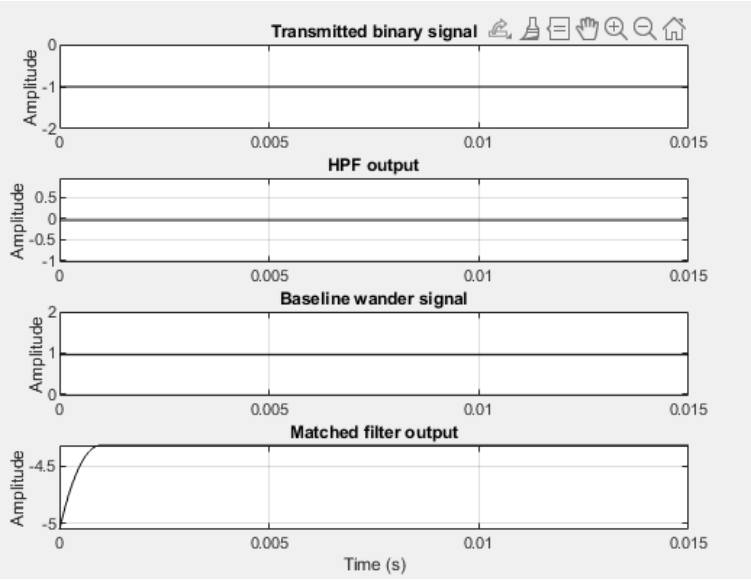


Figure 8- If input sequence is 0s

We can also utilise the slope of the HPF output for decoding the received bit, but for a sufficiently long unchanged sequence, the slope becomes zero, from which point it would be impossible to guess whether the long sequence was of 0s or 1s to begin with.

In figures 7 & 8, we see the slope is zero for both HPF outputs, be it long stream of 0s or 1s. Hence, we can say that slope is not a good judgment parameter for decoding of bits passed through an HPF.

Parameters affecting BWL

- Rb or BitRate - The lower the bitrate the slower the i/p, hence more chance of Baseline wander.

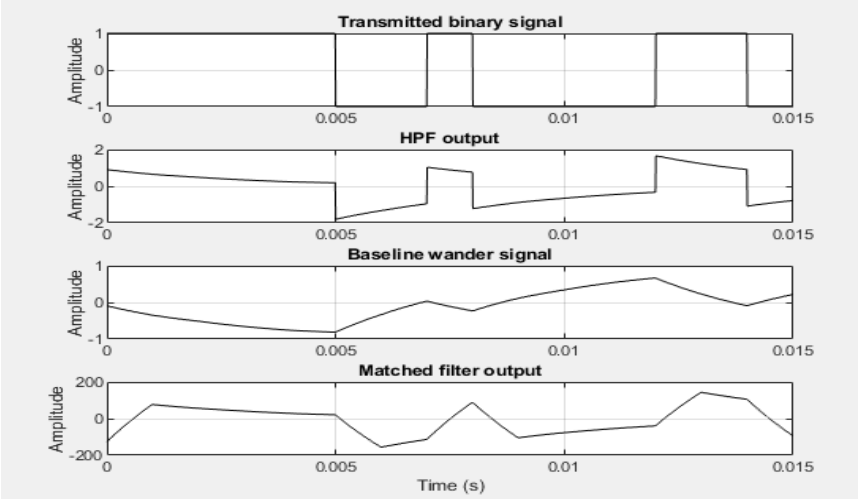


Figure 9 - Rb = 1e3

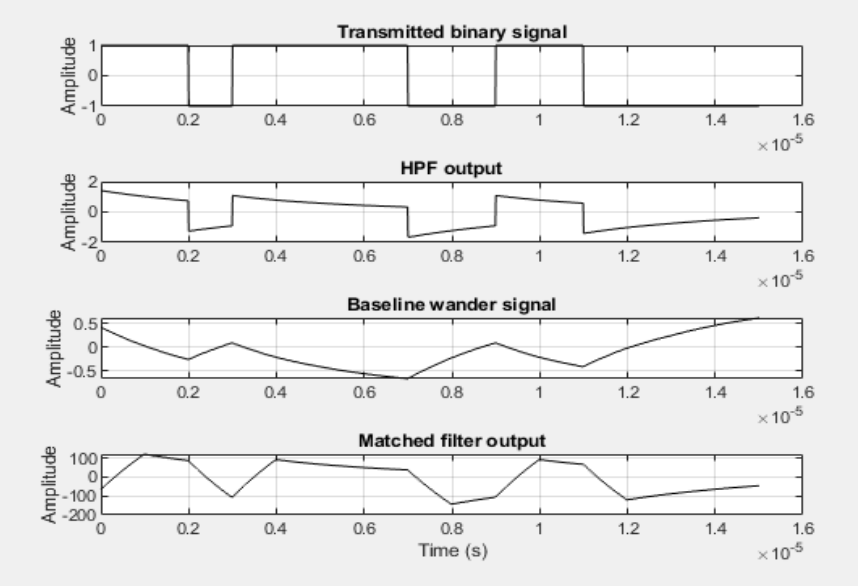


Figure 10 - Rb = 1e6

- nsamp – It reflects the number of samples per symbol, hence, graph smoothens once it goes up.

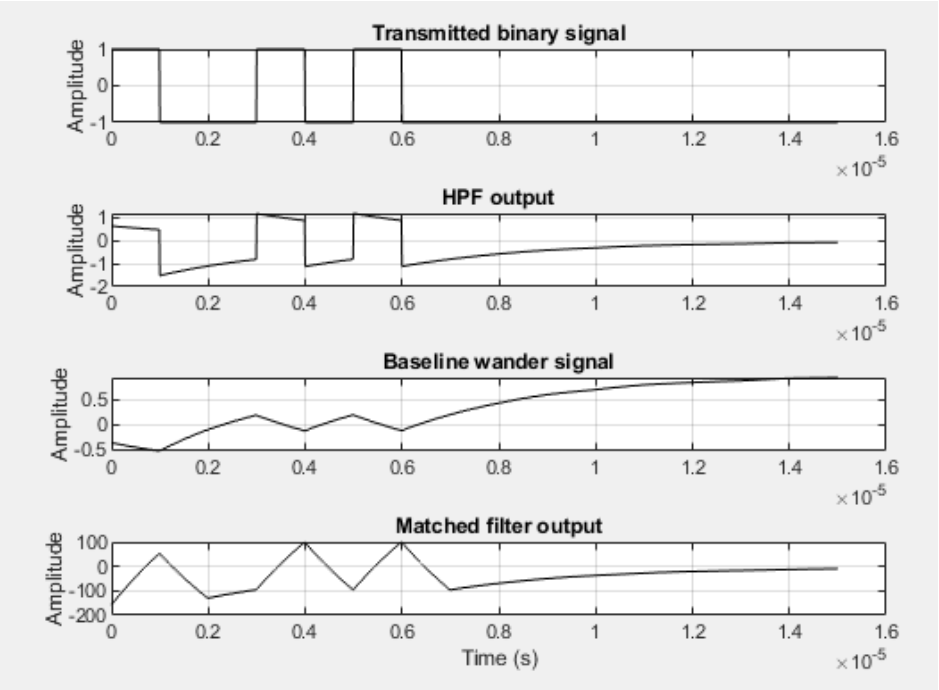


Figure 11 – nsamp = 100

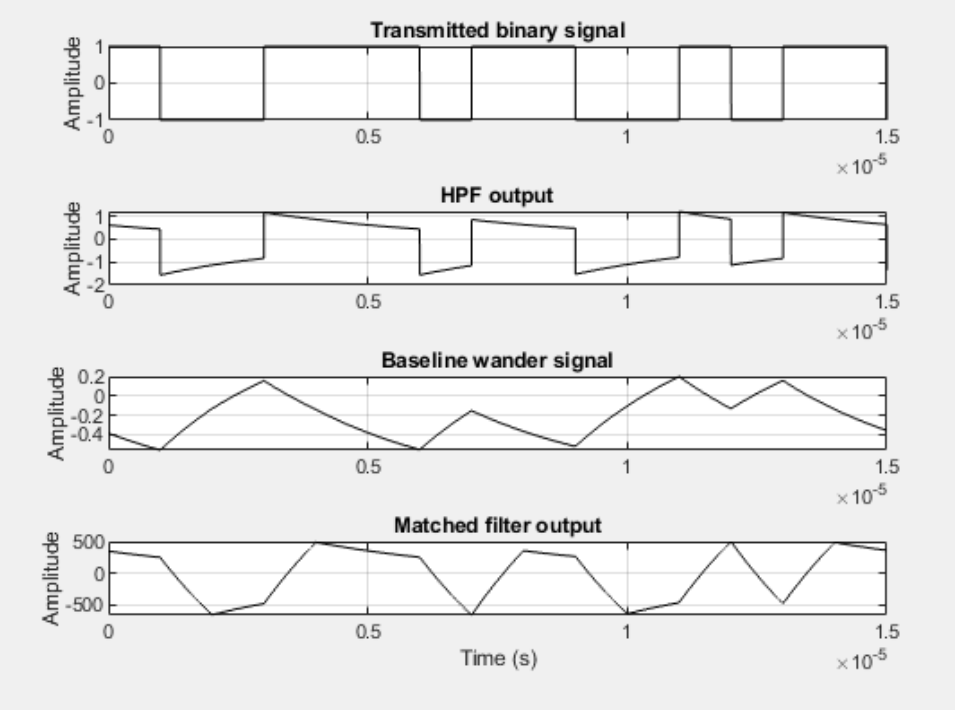


Figure 12 - nsamp = 500

Distribution plots due to BWL

Modelling

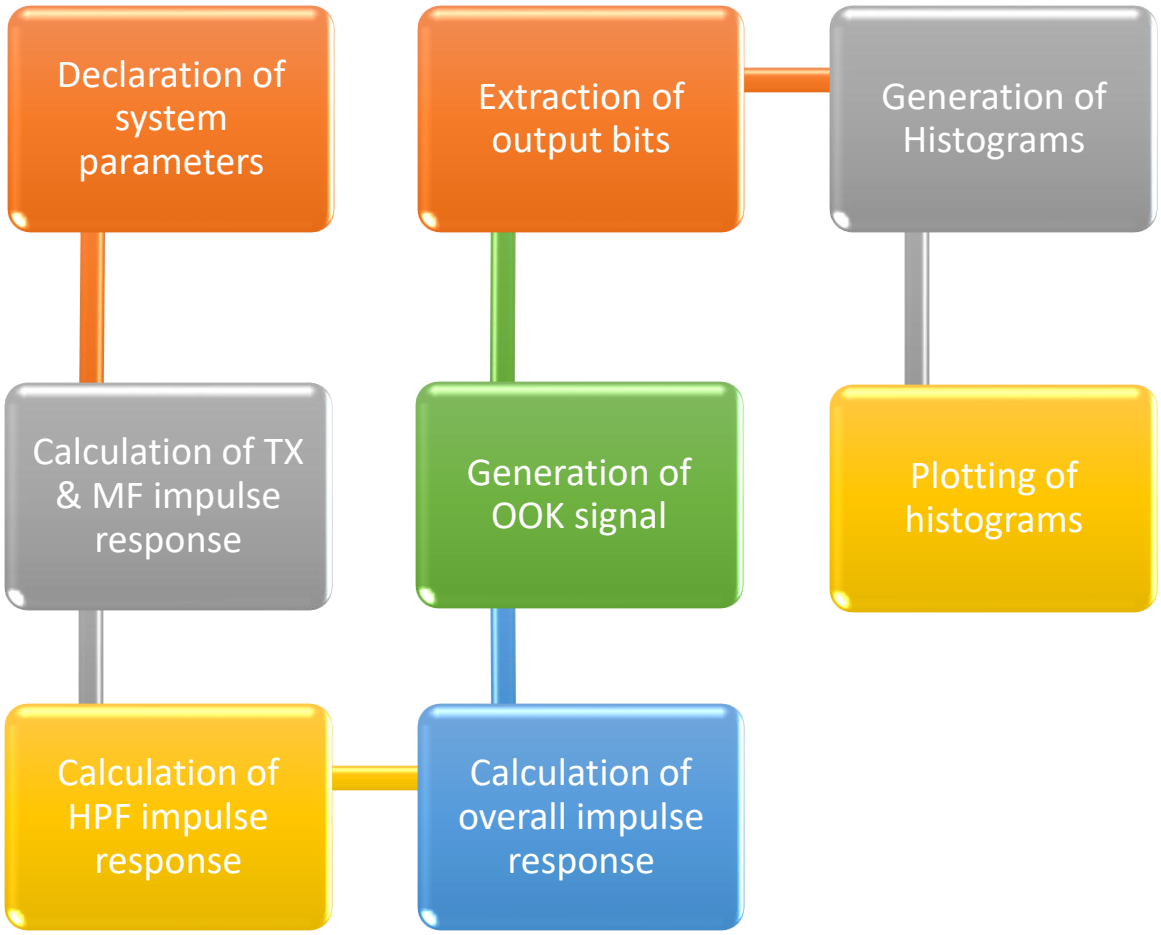


Figure 13 - Flow of code/process

Declaration of System Parameters

- Rb = 1e6 -> **Bit rate**
- Tb = 1/Rb; -> **Bit duration**
- nsamp = 50; -> **Samples per symbol**
- Tsamp = Tb/nsamp; -> **Sampling time**
- Lsym = 1e5; -> **Number of bits**
- fc_rb = 1e-3; -> **Normalized cut-off frequency**
- fc = fc_rb * Rb; -> **Actual cut-on frequency of HPF**
- p_ave = 1; -> **Average power of transmitted signal**
- p_peak = 2 * p_ave;-> **Peak power of TX'd signal for OOK**

Calculation of TX & MF Filter Response

- **Transmitter impulse –** The impulse response of the transmitted impulse is a sequence of ones; this has been done to ensure that we can effectively study the effects of baseline wandering**.** This is because the effects of baseline wander are the most prominent when we input a long sequence of 1s or 0s. We multiply the sequence of ones with **p_peak**, which is the peak power for the transmitted sequence.

```
tx_impulse=ones(1,nsamp)*p_peak;
```

- **Matched filter impulse -** The impulse response of the matched filter is also a sequence of ones, but divided by $\sqrt{T_b}$, this is done due to previous assumptions made in the text referred.

```
mf_impulse=ones(1,nsamp)*(1/sqrt(Tb));
```

Calculation of HPF Impulse Response

- Similar to the previous section, we model the HPF's impulse response as first order RC circuit.

$$g_{out}(t) = \begin{cases} Ae^{-t/RC} & 0 \le t \le \tau \\ -A(e^{-\tau/RC} - 1)e^{-t/RC} & t > \tau \end{cases}$$

Figure 14 - Impulse Response of HPF

Implementation for HPF impulse response: -

```
hpf_impulse(1) = 1*exp(-2*pi*fc*t(1));

for loop = 2:length(t)
   hpf_impulse(loop) = -1*(exp(2*pi*fc*t(1))-1)*exp
(-2*pi*fc*t(loop));
end
```

Figure 15 - Code for HPF impulse response

Calculation of Overall Impulse Response

We determine the discrete-time equivalent impulse response by combining the transmitter filter, receiver filter, and HPF components. Since this combined impulse response quickly diminishes to zero over time, it can be shortened to a specific length without significantly compromising accuracy. The Bit Error Rate (BER) can then be estimated using this shortened response length J, calculated by finding the average error rate across all possible symbol sequences of the same length.

The discrete-time equivalent impulse response is deliberately truncated to span only J bits in duration. This impulse response incorporates the behaviour of the HPF component within the system.

When analyzing system performance, we consider K different bit sequences of length J. Each sequence contains individual bits with binary values (0 or 1). When any particular sequence passes through the communication system, the output from the matched filter is measured by sampling at the conclusion of the Jth bit period.

$$c_j = \begin{cases} p(t) \otimes r(t) \otimes g(t)\big|_{t=j\tau}, & 1 < j < J \\ 0, & \text{otherwise} \end{cases}$$

Figure 16 - Overall impulse response calculation

Here, p(t) is the impulse response of TX. r(t) is the impulse response of the matched filter(MF), and g(t) is the impulse response for the matched filter(MF).

```
temp1 = conv(tx_impulse, hpf_impulse);
temp2 = conv(temp1, mf_impulse);
temp2 = temp2 * Tsamp;
system_impulse = temp2(nsamp:nsamp:200 * nsamp); % Discrete impulse response
```

Figure 17 - Overall impulse response implementation

Generation of OOK signal

Since we utilise a bipolar OOK NRZ symbol for our simulation we must convert a sequence of randomly generated 0s and 1s to -1s and 1s, respectively. Hence we use –

$$OOK = 2 * OOK - 1$$

This equation yields -1 for a 0 and 1 for a 1.

Generation of Plotting of Histograms

We generate the histogram with 51 bins, with the Y-axis being the frequency of occurrence and the x-axis being the matched filter output.

We obtain 3 plots, one for 0s, another for 1s and the last one for overall signal.

```
nbin = 51; % Number of bins for histograms
[n_zero, xout] = histcounts(mf_output_zero, nbin);
[n_one, xout] = histcounts(mf_output_one, nbin);

% Find expected value for 1s
expect_one = xout(find(n_one == max(n_one), 1));

% Compute total histogram
n_total = n_zero + n_one;
```

Figure 18 - Generation of histograms

Simulation

```
%% Define System Parameters
Rb = 1e6;          % Bit rate (1 Mbps)
Tb = 1/Rb;         % Bit duration
nsamp = 50;        % Samples per symbol
Tsamp = Tb/nsamp;  % Sampling time
Lsym = 1e5;        % Number of bits
fc_rb = 1e-3;      % Normalized cut-off frequency
fc = fc_rb * Rb;   % Actual cut-on frequency of HPF

p_ave = 1;         % Average power of transmitted signal
p_peak = 2 * p_ave; % Peak power of TX'd signal for OOK
```

```
%% ***** Calculate Impulse Response of Filters *****
tx_impulse = ones(1, nsamp) * p_peak;  % Transmitter filter
mf_impulse = ones(1, nsamp) * (1 / sqrt(Tb)); % Matched filter impulse response

t = Tsamp:Tsamp:200*Tb; % Time vector

% Initialize HPF impulse response
hpf_impulse(1) = 1 * exp(-2 * pi * fc * t(1));

for loop = 2:length(t)
    hpf_impulse(loop) = -1 * (exp(2 * pi * fc * t(1)) - 1) * exp(-2 * pi * fc * t(loop));
end
```

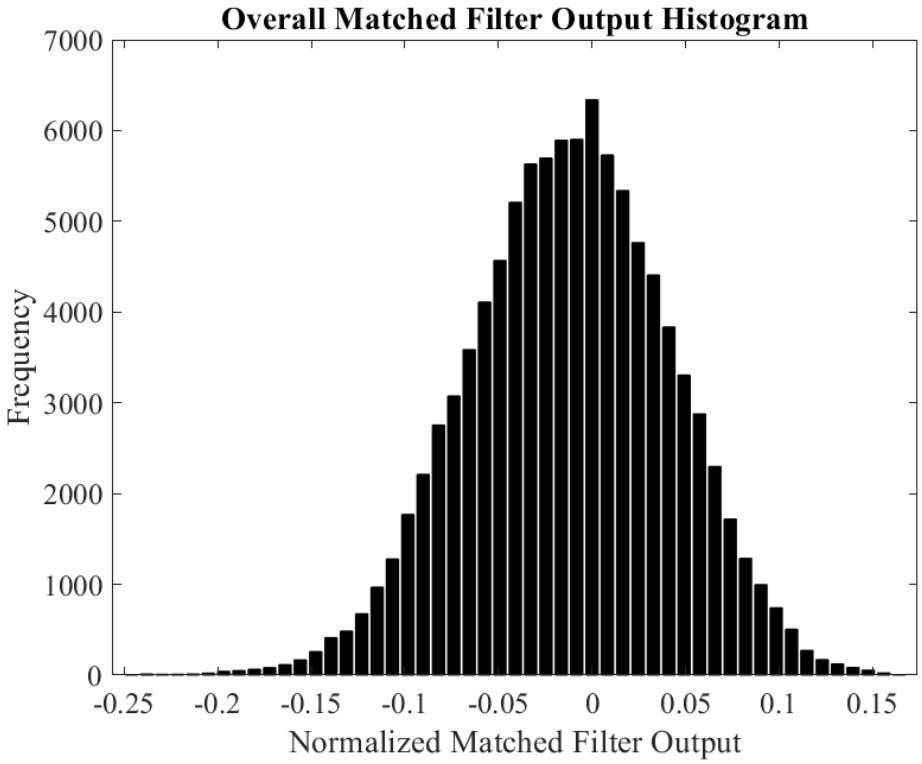
```
%% ***** Calculate Overall Impulse Response *****
temp1 = conv(tx_impulse, hpf_impulse);
temp2 = conv(temp1, mf_impulse);
temp2 = temp2 * Tsamp;
system_impulse = temp2(nsamp:nsamp:200 * nsamp); % Discrete impulse response
```

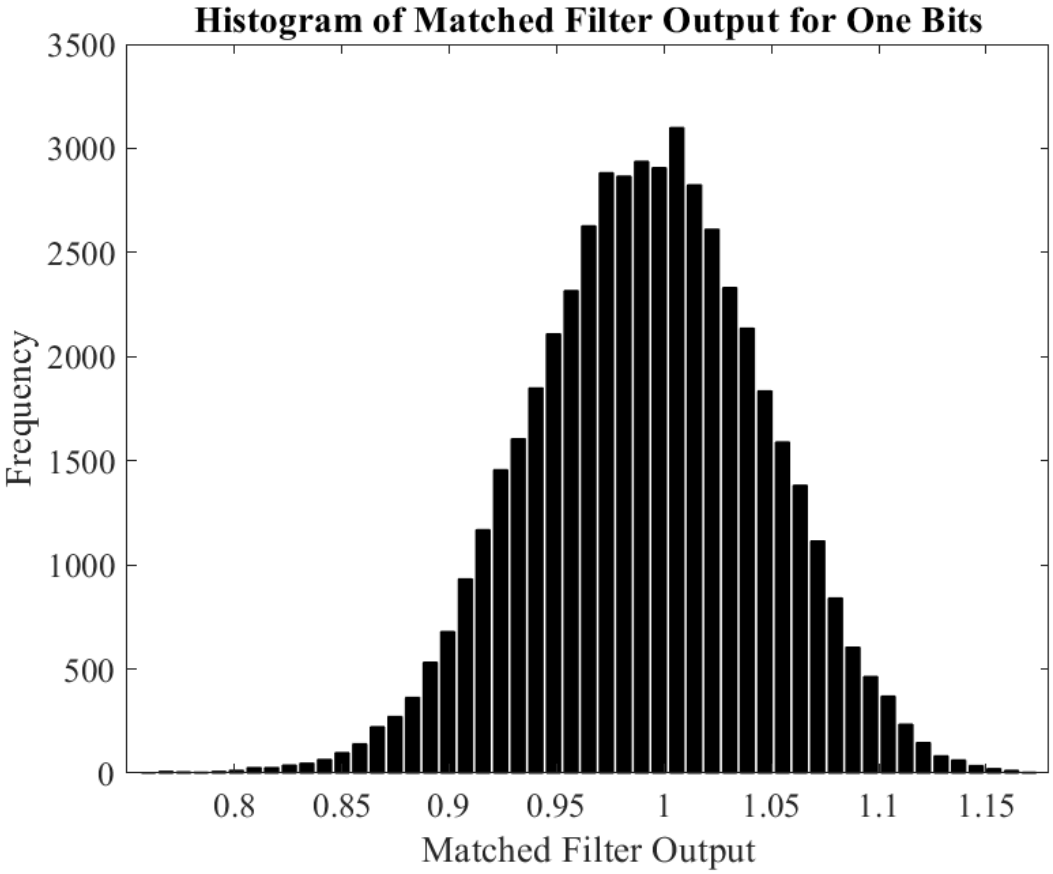
```
%% ***** Generate OOK Signal and Apply System Impulse Response *****
expected_one = 0.5 * p_peak * sqrt(Tb);
expected_zero = -0.5 * p_peak * sqrt(Tb);

OOK = randi([0, 1], 1, Lsym); % Generate random OOK sequence
OOK = 2 * OOK - 1; % Convert to -1 and 1 (remove DC components)

% Apply matched filter
mf_output = filter(system_impulse, 1, OOK) / (2 * expected_one);
%plot(t, matched_output(plotstart:plotfinish))

% Extract outputs for transmitted '1's and '0's
mf_output_one = mf_output(OOK == 1);
mf_output_zero = mf_output(OOK == -1);
```

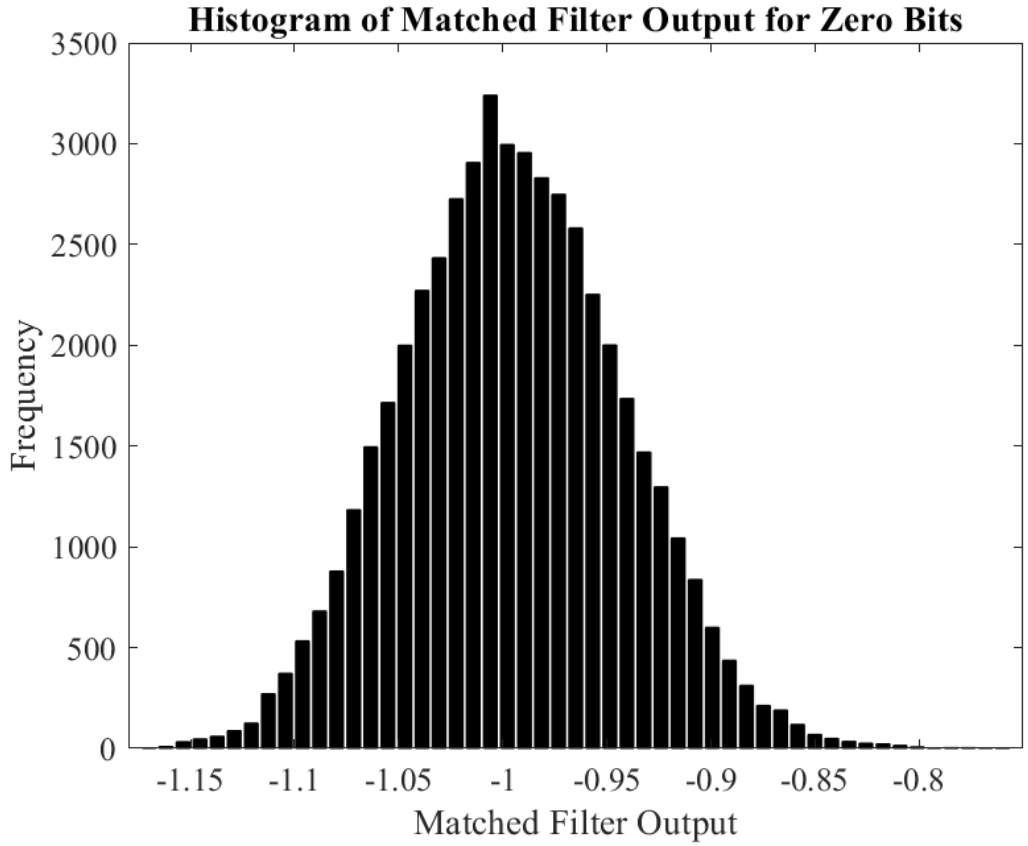
```
%% ***** Generate Histograms *****
nbin = 51; % Number of bins for histograms
[n_zero, xout] = histcounts(mf_output_zero, nbin);
[n_one, xout] = histcounts(mf_output_one, nbin);

% Find expected value for 1s
expect_one = xout(find(n_one == max(n_one), 1));

% Compute total histogram
n_total = n_zero + n_one;
```

```
%% ***** Plot Histograms *****
figure;
bar(-xout(1:end-1), n_zero, 'k'); % Histogram for '0' bits
title('Histogram of Matched Filter Output for Zero Bits');
xlabel('Matched Filter Output'); ylabel('Frequency');
set(gca, 'FontName', 'Times New Roman', 'FontSize', 12);

figure;
bar(xout(1:end-1), n_one, 'k'); % Histogram for '1' bits
title('Histogram of Matched Filter Output for One Bits');
xlabel('Matched Filter Output'); ylabel('Frequency');
set(gca, 'FontName', 'Times New Roman', 'FontSize', 12);

figure;
bar(xout(1:end-1) - expect_one, n_total, 'k'); % Combined histogram
title('Overall Matched Filter Output Histogram');
xlabel('Normalized Matched Filter Output'); ylabel('Frequency');
set(gca, 'FontName', 'Times New Roman', 'FontSize', 12);
```

Observation




Histogram of Matched Filter Output for One Bits
Frequency
Matched Filter Output
0
500
1000
1500
2000
2500
3000
3500
0.8
0.85
0.9
0.95
1
1.05
1.1
1.15


Histogram of Matched Filter Output for Zero Bits
Frequency
Matched Filter Output
0
500
1000
1500
2000
2500
3000
3500
-1.15
-1.1
-1.05
-1
-0.95
-0.9
-0.85
-0.8

Before drawing inferences, it is important to understand what exactly the graph represents.

- **X-Axis**
  - The x-axis plots the matched filter output, it reflects how closely the received waveform matches the expected shape — including effects of noise, filtering, distortion, and ISI.
  - For a bipolar NRZ, we observe that a value of -1 would denote a 0 was transmitted, and a +1 would denote a 1 was transmitted.
  - The values vary around +1 (for 1s) and -1 (for 0s) as the output is not the same as the input. It is affected by the effects of HPF; in our case, the effect of interest is baseline wander.
  - Ideally, the histogram plot should be just a central line centred at -1 (for 0s), at +1 (for 1s).
  - Analogy: It's like a Confidence Score
    - +1.0 → very confident it's a 1
    - -1.0 → very confident it's a 0
    - +0.3 → weak evidence of 1, possibly an error
    - -0.2 → uncertain, could be 0, possibly an error

- **Y-Axis**
  - It depicts the count of detection for each bit, for each matched filter output (bins of histogram define ranges between which the matched filter output lies).
  - A narrower plot would mean lesser baseline wander.

The most important parameter of study is ***fc_rb,*** which is the cutoff frequency divided by the bitrate. If we increase the value of ***fc_rb,*** we see that the value of baseline wander increases, that is, it becomes worse from a performance perspective.

Going back to the impulse response of the HPF (refer figure-13), which was modelled as a first order RC circuit.

As we know fc $\propto 1/\tau$, so when frequency increases, τ decreases. This means the filter responds more quickly, and its impulse response decays faster. This means, that lesser bits influence that output at a higher frequency.

If the HPF averages over many bits, long runs of 0s and 1s are smoothed out — the output stays closer to a consistent baseline. But with fewer bits averaged, the HPF output swings more dramatically whenever there's a run of identical bits. This violently changes the baseline, causing wander.

We can visualize the HPF as the convolution of the input bitstream with the HPF impulse response, where the number of bits involved in convolution decrease with increase in frequency.

When the value of f_c is low, the impulse response is many bit periods long, this affects the amount of ISI or Intersymbol interference present.

Conclusion

This project has successfully demonstrated the implementation and analysis of baseline wander effects in indoor FSO communication systems through MATLAB simulation. The investigation confirms that high-pass filtering, while necessary for reducing low-frequency noise and ambient light interference, introduces significant baseline drift that can degrade system performance. Several key findings emerged through our simulation work:

First, the severity of baseline wander strongly correlates with bit rate and sequence patterns, with lower bit rates and longer sequences of identical bits producing more pronounced effects. The simulation results clearly demonstrate that extended sequences of either 0s or 1s cause the reference level to drift substantially, creating potential decoding errors when the signal crosses the decision threshold.

Second, the matched filter implementation significantly improves signal detection compared to direct HPF output analysis, particularly when dealing with longer bit sequences. However, even this approach has limitations as shown in the histogram distributions of matched filter outputs, where the spread around ideal values (-1 for 0s, +1 for 1s) reveals the persistent influence of baseline wander.

The parameter analysis conducted in this work provides practical guidelines for system design, showing how adjustments to bit rate and sampling frequency can minimize baseline wander effects. These findings contribute to the broader understanding of indoor optical wireless communication challenges and offer a foundation for future work on adaptive threshold algorithms and advanced modulation schemes that could further mitigate baseline wander effects in practical FSO implementations.

References

- **Optical Wireless Communications: System and Channel Modelling with MATLAB ", CRCPress, 2017**
- **A. Tripathy, A. Dash and U. Bhanja, "Effect of High Pass Filtering and Matched Filtering on Baseline Wander," 2023 1st International Conference on Circuits, Power and Intelligent Systems (CCPIS), Bhubaneswar, India, 2023, pp. 1-4, doi: 10.1109/CCPIS59145.2023.10291460.**