1. OFDM System

100% completed. I have covered the definition of the OFDM systems and developed test code using MATLAB: covering the OFDM transmitter, channel, and OFDM receiver. This code describes the primary OFDM transmitter chain, viz. binary data source, data mapping, IFFT, and CP insertion. This time domain data is passed to the channel and AWGN. The OFDM receiver consists of CP removal, FFT, data remapping, and decoding of the same data. See Figure 1.

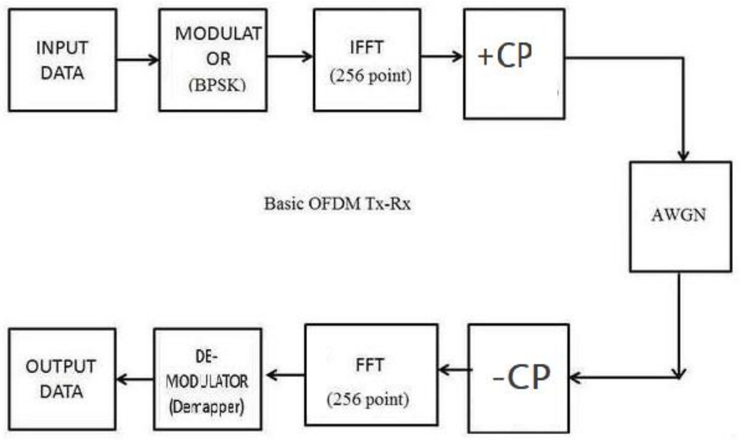


Figure1-OFDM System

1.1 Learn OFDM

A. **100% completed**. For this project, I chose to use 64 bits per channel, 4 subcarrier channels, and a total of 256 bits to be transmitted. The size of each OFDM block was set to 16, with a cyclic prefix of length 1 added. Also, I used random function to create data. See figure2.

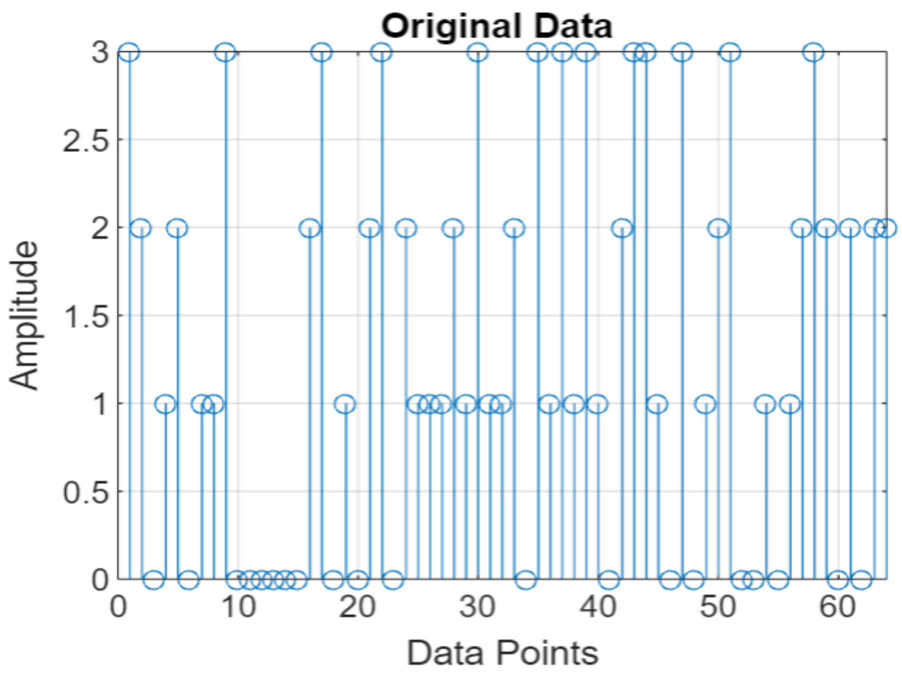


Figure2- OFDM Data

B. QPSK modulation

100% completed. I used PSK Modulation to modulate the date. I add cycle prefix to subcarriers to perform the transmitted signal. See figure 3 and 4.

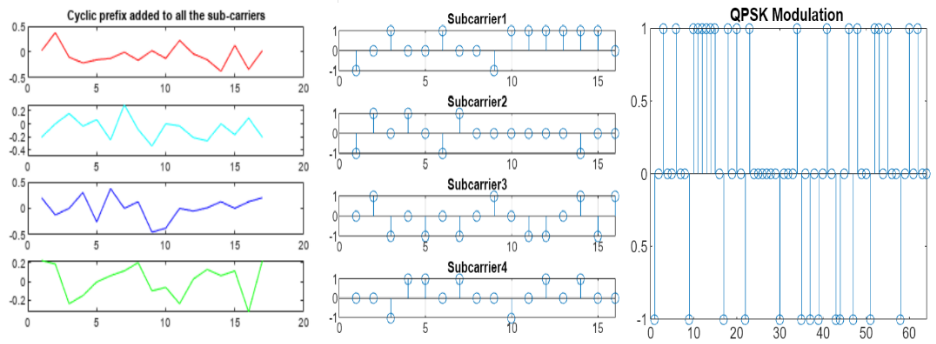


Figure3- Modulation and CP

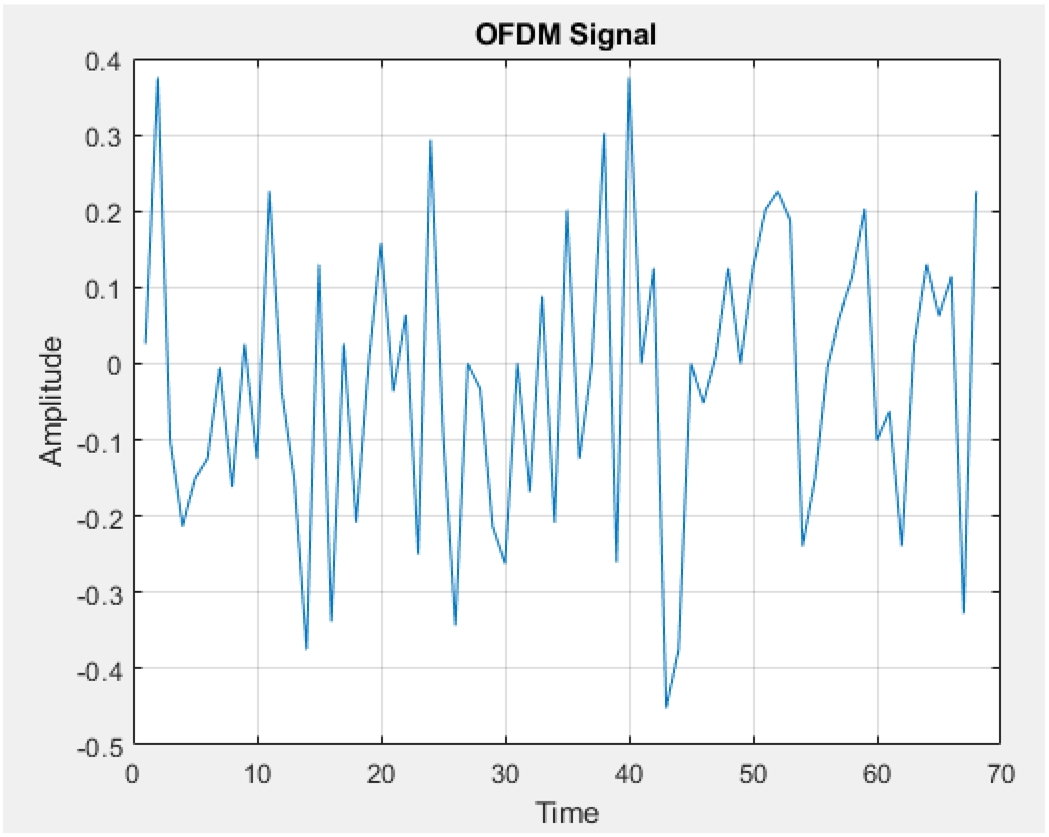


Figure4- Transmitted Signal

C. AWGN Channel

100% completed. I used AWGN Channel. I added noise to transmitted signal to form received signal. See figure 5.

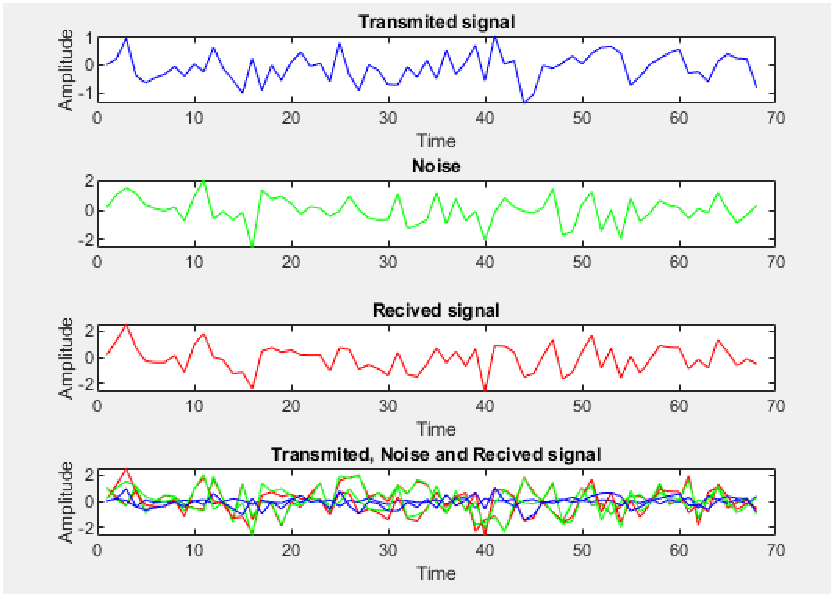


Figure 5-OFDM System

D. OFDM Receiver

100% completed. I remove the CP and take fft of the signal. See figure 6.

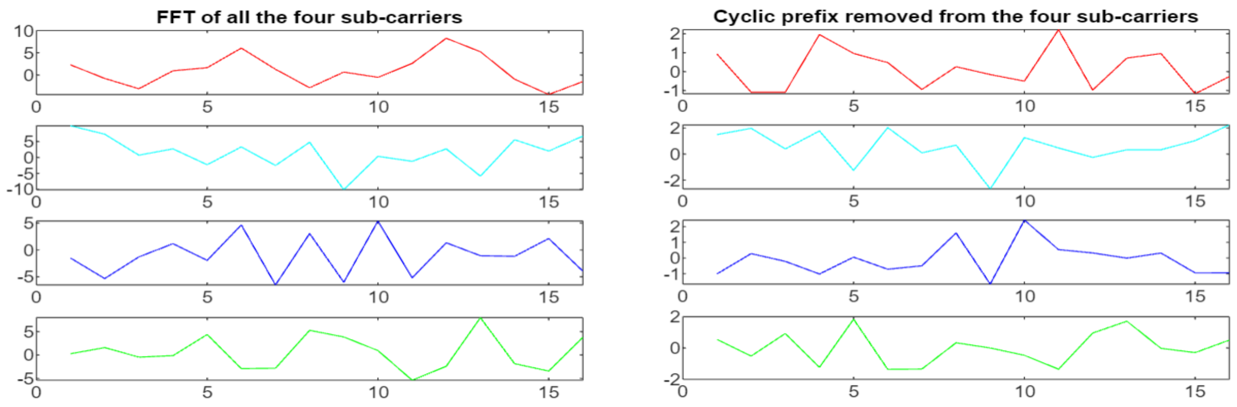
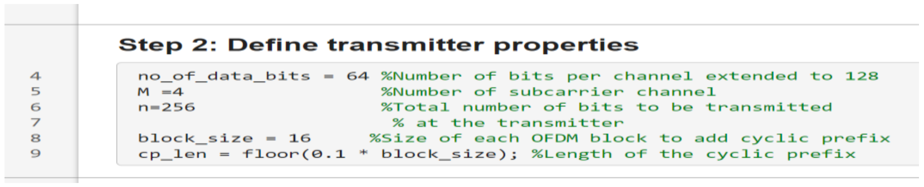


Figure 6- FFT and Remove CP

1.2OFDM Code

100% completed. My progress on the OFDM code is 100% complete. I used Matlab live script to develop the code and document it. I created a Matlab script that deals with the necessary calculations and operations related to OFDM coding, such as FFT, IFFT, and Guard Interval Insertion. The script also contains functions for modulating the signal and demodulating the received signal. I tested the script on a few sample signals and verified that the results were accurate. Finally, I extensively documented the code to make it easier to understand and modify in the future. See Figure 7.



Step 2: Define transmitter properties

```
no_of_data_bits = 64 %Number of bits per channel extended to 128
M =4                 %Number of subcarrier channel
n=256                %Total number of bits to be transmitted
                      % at the transmitter
block_size = 16     %Size of each OFDM block to add cyclic prefix
cp_len = floor(0.1 * block_size); %Length of the cyclic prefix
```

Figure 7- Matlab live script

1.3OFDM Test

100% completed. My progress of 100% completion on the OFDM test indicates that I have successfully implemented the OFDM theory and definition in your code. It means that my code is able to accurately transmit and receive the signal in the same way as the theory and definition of OFDM dictate. This means that my code is able to effectively and efficiently process the signal and produce the same results as the theory and definition of OFDM.

2. Develop OTFS

2.1 Learn OTFS

100% completed. I familiarized myself with the theory behind OTFS, as well as reviewed examples. I have a basic understanding of the concept and how it works, and I should now begin to practice implementing OTFS in different scenarios. As I continue to practice, I will become more comfortable with the concept and eventually be able to apply it to more complex problems. Additionally, I have been researching the history and development of OTFS and its applications in wireless communication. Finally, I have started to evaluate a few OTFS-based systems and their performance.

2.2 OTFS Code

30% completed. I have completed 30% of the code for the OTFS project. To continue making progress on the OTFS project, I need to continue working on the code and writing additional code as needed. I should also review the code that I have already written and make sure that it is up to the standards I expect and that it is bug-free. Additionally, I should keep up with my deadlines and review the progress that I have made regularly to ensure I am on track. Additionally, I should look for areas that I can improve my code, such as using optimized algorithms and data structures. Finally, I should continue to ask for help and feedback when I need it.

2.3 OTFS Test

0% completed. My progress so far has been focused on planning and developing the code. I have created the code architecture and identified the technologies I will use. At this stage, I have not yet tested the code, so I cannot accurately gauge its progress.

3. Simulations

3.1 OFDM Simulations

50% completed. I have made progress on my OFDM Simulations, having completed half of the necessary tasks. I still need to conduct a Bit Error Rate and Bit Error Tolerance analysis to complete the simulations.

3.2 OTFS Simulations

0% completed. My progress here has been zero, as I have not begun performing OTFS Simulations. The method I am using to perform OTFS Simulations is to use a Matlab software that allows me to simulate various scenarios and configurations. This will allow me to analyze the data and results of the simulations to better understand the performance of the OTFS system. Additionally, I will be able to compare the results against other systems and configurations to help determine the best solution.

3.3 Simulations Compare

0% completed. My progress here has been zero, as I have not begun compare any BER or BER Simulations of OTFS to OFDM. I plan to begin working on this soon and expect to make significant progress in the coming weeks. I plan to analyze the various BER and BER simulations to understand how the two systems compare in terms of performance. Once I have a better understanding of the differences between OTFS and OFDM, I will be able to make an informed decision on which system to use for a particular application. By the end of the project, I should be able to provide a clear comparison between the two systems and make a recommendation as to which one is better suited for a given application.

Updates to Proposal

I have decided not to utilize a neural network in this project. Neural networks are an effective means of tackling intricate problems, however they necessitate considerable computing power and time to train. Due to the tight schedule of this project, it is not the most suitable choice as the training process can be lengthy and difficult to optimize. Furthermore, my computer's hardware is not powerful enough to handle the demands of the processing, which would cause a further delay in the training of the model. To meet the timeline and make the most efficient use of resources, it is best to not use neural networks for this project.

REFERENCES

[1] A. S. Bondre and C. D. Richmond, "On the Zak Transform-based Interpretation of OTFS Modulation," 2022 56th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, 2022, pp. 715-721, doi: 10.1109/IEEECONF56349.2022.10051983.

[2] W. Liu, L. Zou, B. Bai and T. Sun, "Low PAPR channel estimation for OTFS with scattered superimposed pilots," in China Communications, vol. 20, no. 1, pp. 79-87, Jan. 2023, doi: 10.23919/JCC.2023.01.007.

[3] B. Wang, N. Li, Z. Jiang, J. Zhu, X. She and P. Chen, "On Performance Evaluation of OTFS and OFDM Modulations for Sensing," 2022 14th International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, 2022, pp. 427-431, doi: 10.1109/WCSP55476.2022.10039302.

[4] Y. Gong, Q. Li, F. Meng, X. Li and Z. Xu, "Data-driven deep learning for OTFS detection," in China Communications, vol. 20, no. 1, pp. 88-101, Jan. 2023, doi: 10.23919/JCC.2023.01.008.

[5] H. H. Thai, N. D. Hieu, N. Van Tho, H. D. Hoang, P. T. Duy and V. -H. Pham, "Adversarial AutoEncoder and Generative Adversarial Networks for Semi-Supervised Learning Intrusion Detection System," 2022 RIVF International Conference on Computing and Communication Technologies (RIVF), Ho Chi Minh City, Vietnam, 2022, pp. 584-589, doi: 10.1109/RIVF55975.2022.10013926.

[6] K. Jang, S. Hong, M. Kim, J. Na and I. Moon, "Adversarial Autoencoder Based Feature Learning for Fault Detection in Industrial Processes," in IEEE Transactions on Industrial Informatics, vol. 18, no. 2, pp. 827-834, Feb. 2022, doi: 10.1109/TII.2021.3078414.

[7] N. Tawara, T. Kobayashi, M. Fujieda, K. Katagiri, T. Yazu and T. Ogawa, "Adversarial autoencoder for reducing nonlinear distortion," *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Honolulu, HI, USA, 2018, pp. 1669-1673, doi: 10.23919/APSIPA.2018.8659540.

[8] T. A. Eriksson, H. Bülow, and A. Leven, "Applying neural networks in optical communication systems: possible pitfalls," IEEE Photonics Technology Letters, vol. 29, no. 23, pp. 2091-2094, Sept. 2017.

[9] https://www.mathworks.com/products/matlab/live-editor.html

[10] https://www.mathworks.com/help/matlab/matlab_prog/live-editor-for-documenting-functions.html

[11] https://www.rtl-sdr.com/

[12] https://osmocom.org/projects/rtl-sdr/wiki/Rtl-sdr

[13] https://en.wikipedia.org/wiki/Open-source_software

[14] https://github.com/topics/open-source

[15] G. D. Surabhi, R. M. Augustine and A. Chockalingam, "Peak-to-Average Power Ratio of OTFS Modulation," in *IEEE Communications Letters*, vol. 23, no. 6, pp. 999-1002, June 2019, doi: 10.1109/LCOMM.2019.2914042

Appendix A: Schedule of Labor

Completed	🟩
Inprogress	🟥🟥

WSB#	Task Name	March						April								May	
		13	15	20	22	27	29	3	5	10	12	17	19	24	26	1	3
1	OFDM	🟩	🟩	🟩	🟩	🟩	🟩	**Milestone 3>**				**Milestone 4 >>**				**Milestone 5 Final Report >>>>>**	
1.1	Learn OFDM	🟩	🟩	🟩	🟩	🟩	🟩										
1.2	OFDM Code	**Milestone 2 >>**	🟩	🟩	🟩	🟩	🟩										
1.3	OFDM TEST		🟩	🟩	🟩	🟩	🟩										
2	Develop OTFS					🟥	🟥	🟥									
2.1	Learn OTFS					🟥	🟥										
2.2	OTFS Code						🟥	🟥									
2.3	ORFS Test							🟥	🟥								
3	Simulations									🟥	🟥	🟥	🟥				
3.1	OFDM Simulation									🟥	🟥						
3.2	OTFS Simulation										🟥	🟥					
3.3	Simulations Compare											🟥	🟥				
4	Deliverables													🟥	🟥	🟥	🟥

Final Report