ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάλυση και Προσομοίωση της Διαμόρφωσης OTFS σε Συστήματα Ασύρματων Επικοινωνιών Πέραν της Πέμπτης Γενιάς

Συγγραφέας: Χαράλαμπος Οικονομίδης
ice18390049

Επιβλέπων: Νικόλαος Μυριδάκης

Αθήνα, Οκτώβριος 2023

UNIVERSITY OF WEST ATTICA

FACULTY OF ENGINEERING

DEPARTMENT OF INFORMATICS AND COMPUTER ENGINEERING

DIPLOMA THESIS

Analysis and Simulation of OTFS Modulation for Wireless Communication Systems Beyond 5G

Author: Charalampos Oikonomidis
ice18390049

Supervisor: Nikolaos Myridakis

Athens, October 2023

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάλυση και Προσομοίωση της Διαμόρφωσης OTFS σε συστήματα ασύρματων επικοινωνιών πέραν της Πέμπτης Γενιάς

Analysis and Simulation of OTFS Modulation for wireless communication systems beyond 5G

Χαράλαμπος Οικονομίδης
ice18390049

Γλώσσα Συγγραφής: Αγγλικά

Μέλη Εξεταστικής Επιτροπής συμπεριλαμβανομένου και του Εισηγητή

Α/α	ΟΝΟΜΑ ΕΠΩΝΥΜΟ	ΒΑΘΜΙΔΑ/ΙΔΙΟΤΗΤΑ	ΥΠΟΓΡΑΦΗ
1	Νικόλαος Μυριδάκης	Επίκουρος Καθηγητής	Nikolaos Myridakis Digitally signed by Nikolaos Myridakis Date: 2023.10.03 10:20:33 +03'00'
2	Αντώνιος Μπόγρης	Καθηγητής	Antonios Bogris Digitally signed by Antonios Bogris Date: 2023.10.04 11:37:57 +03'00'
3	Παναγιώτης Καρκαζής	Αναπληρωτής Καθηγητής	Panagiotis Karkazis Digitally signed by Panagiotis Karkazis Date: 2023.10.06 12:27:20 +03'00'

Ημερομηνία Εξέτασης 03/10/2023

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος **Οικονομίδης Χαράλαμπος** του **Αναστασίου**, με αριθμό μητρώου **ice18390049** φοιτητής του **Πανεπιστημίου Δυτικής Αττικής** της **Σχολής Μηχανικών** του **Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών**, δηλώνω υπεύθυνα ότι:

«Είμαι ο συγγραφέας αυτής της Διπλωματικής Εργασίας και κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφή από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του διπλώματός μου».

Ο Δηλών

[illegible]

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να εκφράσω τις θερμές μου ευχαριστίες στο κύριο Νικόλαο Μυριδάκη για την εμπιστοσύνη που μου έδειξε δίνοντάς μου το παρόν θέμα, καθώς και για την καθοδήγηση και τις ιδέες που μου έδωσε για την εκπόνηση της διπλωματικής εργασίας μου. Το θέμα με βοήθησε να κατανοήσω καλύτερα το κομμάτι της ακαδημαϊκής έρευνας, δίνοντάς μου κίνητρο να συνεχίσω τις σπουδές μου, καθώς και να κοιτάξω προς την κατεύθυνση μίας πιο ερευνητικής καριέρας. Ευελπιστώ ότι κάποια στιγμή στο μέλλον θα έχουμε την δυνατότητα να συνεργαστούμε μαζί ξανά σε αντίστοιχα ερευνητικά και μη θέματα.

Επιπλέον, θα ήθελα να ευχαριστήσω την οικογένειά μου που με στήριξε καθ’ όλη την διάρκεια των σπουδών μου, επιτρέποντάς μου να αφοσιωθώ εξολοκλήρου στα μαθήματά μου και να ολοκληρώσω τις σπουδές μου με έναν αξιοπρεπή βαθμό.

Τέλος, θα ήθελα να ευχαριστήσω όλους τους καθηγητές του ιδρύματος, οι οποίοι πάντα είχαν διάθεση να λύνουν απορίες σε όποια ερώτηση και αν υπήρξε. Με βοήθησαν να κατανοήσω καλύτερα το πεδίο της επιστήμης των υπολογιστών, αφήνοντάς μου την θέληση της γνώσης και την επιθυμία να συνεχίσω τις σπουδές μου.

Οικονομίδης Χαράλαμπος

[illegible]

ΠΕΡΙΛΗΨΗ

Με την εξέλιξη της τεχνολογίας, αυξάνονται και οι ανάγκες που θα πρέπει να καλύπτει ένα επικοινωνιακό σύστημα. Οι τεχνολογίες που έχουν χρησιμοποιηθεί μέχρι στιγμής δεν είναι επαρκής, με αποτέλεσμα την ανάγκη υλοποίησης καινοτόμων τεχνολογιών. Η παρούσα διπλωματική εργασία ασχολείται με την ανάλυση μίας τέτοιας τεχνολογίας, την μορφοποίηση OTFS. Συγκεκριμένα, στο πρώτο κεφάλαιο γίνεται μία βιβλιογραφική ανασκόπηση στις τεχνολογίες που έχουν χρησιμοποιηθεί από την αρχή των τηλεπικοινωνιακών δικτύων, καθώς και των προσδοκιών μελλοντικών δικτύων. Στο δεύτερο κεφάλαιο, αρχικά γίνεται επεξήγηση κάποιων σημαντικών εννοιών για την καλύτερη κατανόηση του θέματος. Στη συνέχεια υπάρχει βιβλιογραφική αναφορά της τεχνολογίας OTFS, καθώς και η ανάλυσή της. Επιπλέον υπάρχει και η παρουσίαση και ανάλυση τεχνικών δεκτών που αναλογούν σε αυτή την τεχνολογία. Στο τρίτο κεφάλαιο υπάρχει η υλοποίηση ενός πλήρες συστήματος OTFS στο λογισμικό MATLAB, με δύο διαφορετικούς τρόπους. Στο τέταρτο κεφάλαιο υπάρχουν τα αποτελέσματα των προσομοιώσεων του συστήματος αυτού και η ανάλυσή τους. Στο πέμπτο και τελευταίο κεφάλαιο υπάρχει μία σύνοψη των αποτελεσμάτων, ιδέες για επόμενες έρευνες και προσωπικά σχόλια πάνω στην τεχνολογία. Έπειτα ακολουθούν τα αρχεία κώδικα που βρίσκονται στα παραρτήματα Α και Β, καθώς και οι βιβλιογραφία που χρησιμοποιήθηκε.

ΕΠΙΣΤΗΜΟΝΙΚΗ ΠΕΡΙΟΧΗ: Δίκτυα Ασύρματων Επικοινωνιών

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: OTFS, Doppler, OFDM, πέμπτη γενιά, πέρα της πέμπτης γενιάς, MATLAB, mmWave

ABSTRACT

With the evolution of technology, the requirements that communication systems must meet also increase. The technologies currently in use are no longer sufficient, necessitating the implementation of novel technologies. The present thesis concerns the analysis of one such novel technology, the OTFS modulation. Specifically, in the first chapter there is a literature review of all the technologies that have been used since the beginning of telecommunication networks, along with future network expectations. The second chapter begins by laying some fundamental knowledge that is necessary to properly understand the subject. Afterwards, there is a literature overview and analysis of OTFS modulation, followed by the different receiver technology that OTFS is accompanied by. In the third chapter, two different implementations in MATLAB of the OTFS system can be found. The fourth chapter contains the simulation results, along with their analysis. In the fifth and final chapter, a synopsis of the results that were presented in chapter four can be found, along with future work ideas and personal comments on the technology. Following chapter five, two appendices, A and B, are included which contain the code that was used, as well as all the references that the thesis was based on.

SCIENTIFIC FIELD: Wireless Communication Networks

KEYWORDS: OTFS, Doppler, OFDM, fifth generation, beyond fifth generation, MATLAB, mmWave

Table of Contents

List of Figures

List of Tables

List of Abbreviations

0G - Zero Generation

1G - First Generation

2G - Second Generation

3G - Third Generation

4G - Fourth Generation

5G - Fifth Generation

B5G - Beyond 5G

6G - Sixth Generation

MTS - Mobile Telephone Service

IMTS - Improved Mobile Telephone Service

VHF - Very High Frequency

UHF - Ultra High Frequency

NTT - Nippon Telegraph and Telephone

NMT - Nordic Mobile Telephone

AMPS - Advanced Mobile Phone System

RF - Radiofrequency

D-AMPS - Digital Advanced Mobile Phone System

GSM - Global System for Mobile Communications

TDMA - Time-Division Multiple Access

CDMA - Code-Division Multiple Access

GPRS - General Packet Radio Service

MMS - Multimedia Messaging Service

PSK - Phase Shift Keying

EDGE - Enhanced Data Rates for GSM Evolution

3GPP - 3rd Generation Partnership Project

UMTS	Universal Mobile Telecommunications System
W-CDMA	Wideband Code-Division Multiple Access
HSPA	High Speed Packet Access
HSDPA	High Speed Downlink Packet Access
HSUPA	High Speed Uplink Packet Access
QAM	Quadrature Amplitude Modulation
HARQ	Hybrid Automatic Repeat Request
FEC	Forward Error Correction
ARQ	Automatic Repeat Request
HSDPA+	Evolved HSDPA
DC-HSDPA	Dual-Carrier HSDPA
DC-HSUPA	Dual-Carrier HSUPA
MIMO	Multiple-Input Multiple-Output
MC-HSPA	Multi-Carrier HSPA
GGSN	Gateway GPRS Support Node
RNC	Radio Network Controller
ITU	International Telecommunication Union
IMT	International Mobile Telecommunications
LTE	Long Term Evolution
LTE+	LTE Advanced
VoIP	Voice over IP
OFDM	Orthogonal Frequency-Division Multiplexing
OFDMA	Orthogonal Frequency-Division Multiple Access
SNR	Signal to Noise Ratio
NAT	Network Address Translation
E-UTRA	Evolved Universal Terrestrial Radio Access

QoS - Quality of Service

CoMP - Coordinated Multi-Point

NR - New Radio

IoT - Internet of Things

mmWave - Millimeter Wave

ISPs - Internet Service Providers

SDN - Software Defined Networking

NFV - Network Function Virtualization

QKD - Quantum Key Distribution

URLLC - Ultra-Reliable Low Latency Communication

OTFS - Orthogonal Time Frequency Space

FFT - Fast Fourier Transform

IFFT - Inverse Fast Fourier Transform

DAC - Digital-to-Analog Converter

ADC - Analog-to-Digital Converter

ISI - Inter-Symbol Interference

ICI - Inter-Carrier Interference

LoS - Line-of-Sight

V2V - Vehicle-to-Vehicle

V2I - Vehicle-to-Infrastructure

V2x - Vehicle-to-Everything

SFT - Symplectic Fourier Transform

SFFT - Finite Symplectic Fourier Transform

ISFFT - Inverse Finite Symplectic Fourier Transform

MRC - Maximal Ratio Combining

DFE - Decision Feedback Equalizer

RNPI - Residual Noise Plus Interference

MPA - Message Passing Algorithm

PMF - Probability Mass Function

MAP - Maximum A Posterior Probability

LMMSE - Linear Minimum Mean Square Error

BER - Bit-Error Rate

ZP - Zero Padding

EPA - Extended Pedestrian A

EVA - Extended Vehicular A

ETU - Extended Typical Urban

UE - User Equipment

AWGN - Additive White Gaussian Noise

DFT - Discrete Fourier Transform

Chapter 1 Introduction

1.1 History of Wireless Communications

1.1.1 The Beginning of Wireless Communications (0G)

Before the creation of small, fully portable digital devices that are able to communicate in incredibly fast speeds, wireless communications were very different than what they are today. Dating back to the 1940s, long before the creation of the cellular phone, analog radio systems were used for communication, named Mobile Radio Telephone. These are retroactively referred to as the pre-cellular communication systems or sometimes as the zero generation (0G).

One of the technologies used for these systems was the Mobile Telephone Service (MTS), that was first used in 1946, making it one of the earliest mobile telephone standards). Originally, the equipment that was necessary to use this service weighed around 36 kilograms, while supporting just 3 channels for all users. Clearly, the sheer weight of the equipment made it only viable to be used in combination with vehicles such as cars and trucks, being appropriately named car phones. It is important to note, that this service is not the same as the one police vehicles utilized, since it was a commercial service that was a part of the public switched telephone network, rather than a part of a closed network. Besides the system having very limited capacity, which was very quickly saturated with users, it had a variety of different problems. Being an analog system, functioning effectively the same way as standard radios (i.e., walkie-talkies), it was extremely prone to interference and network congestion. The system was also using a half-duplex operation, meaning that only one party could be talking at a time. The channel could easily be overtaken by another system that would be broadcasting a more powerful signal at the same frequency MTS used, which basically destroyed the connection and prevented phone calls. This, inherently, posed a significant issue, especially in highly populated areas such as large cities, the network was unable to carry out many channels in a geographic area due to congestion. This technology has been discontinued for more than three decades, however, many of its frequencies are now used for local paging services [1].

MTS was ultimately replaced by its successor, Improved Mobile Telephone Service (IMTS) in the mid-1960s. This service offered significant improvements to the original MTS, such as full-duplex operation, allowing simultaneous communication on both ends. IMTS operated in more channels, while utilizing three different frequency bands, ranging from very-high (VHF low and high bands) to ultra-high frequencies. A more "portable" equipment was also developed, allowing the phone to be stored in a briefcase. A feature that the service had, was the fact that it was able to be connected to a vehicles horn, giving it the ability to, when receiving a call, honk to alert the driver that could be far away from the phone (perhaps a very early version of today's ringtones). Despite the new features, this system also had very similar disadvantaged, such as very high interference sensitivity and low capacity for subscribers. In fact, the system was so saturated that a new user could be waiting for years to be able to use the service, having to wait for anyone cancelling their subscription to free up room in the channels. The lack of capacity had a significant toll on sales, making the production for the equipment very expensive, significantly reducing profits. This acted as the main driver for the research and creation of the cellular phones, since many saw the opportunity of investment and the possible profit of such technology. The goal was to be able to support a significantly higher number of users that were able to be subscribed to the service at one time, ultimately leading to the creation of the first generation of wireless cellular networks (1G) [1].

1.1.2 The First Generation (1G)

While also being analog, the first cellular network aimed to improve radio telephony in many ways. As we have already mentioned, technologies that were utilized in 0G had serious limitations and disadvantages, both on channel capacity and channel robustness. The first step that was taken to improve mobile communications was to figure out a way of better utilizing the available frequency spectrum the system had access to [1].

The introduction of dynamic channel allocation was introduced, aiming to stabilize and improve interference robustness while increasing the overall capacity for subscribers at any given time. To achieve this, equipment upgrades took place, improving both infrastructure (base stations) and user equipment (phones). To accommodate the needs of this generation, the coverage shifted from big transmission towers which could overlap with adjacent geographical areas to "cells" that compartmentalized an area into smaller regions that would handle the region's traffic independently. These new base stations improved coverage by having the ability to reuse frequencies between cells that were sufficiently distanced (so the frequencies would not overlap like in 0G), improving the frequency spectrum utilization significantly while reducing interference between the cells themselves. As we said, the network shifted from fixed, pre-registered frequencies to dynamic, on-demand frequency allocation to users, which lead to an astonishingly high increase in capacity and the reduction of the likelihood of network congestion. With areas now being separated and independent, it is obvious that when traveling between cells with different frequencies, a user would eventually be too far to transmit signals to the base station. To address this issue, a technique named "handover" was introduced, allowing moving users to switch from one base station to another. This meant that a user that was travelling had access to the phone service, even though they would be far away from the original base station that they were in. In short, a handover is the act of switching the base station that the user equipment is using to another one which has a more powerful signal (better reception). With this in mind, the concept of "roaming" was introduced, where users that were travelling to different countries were able to use the phone services. Of course, this required countries using the same (or compatible) technologies, marking the beginning of global technology standardization [2].

While there were efforts to standardize a specific technology, different countries did use different protocols and systems, albeit compatible. The first commercially available cellular network system, was developed and deployed in Japan in 1979 by the Nippon Telegraph and Telephone (NTT). While this was the first commercial and nationwide cellular network, it was not the first that was developed. Prior to this, Bell Labs built the first cellular network around Chicago in 1977 and it was trialed in 1978. The Nordic countries developed their own cellular network named the Nordic Mobile Telephone (NMT) and it was first launched in Sweden in 1981. In 1983, the first cellular network was launched in the United States, named the Advanced Mobile Phone System (AMPS). The AMPS was originally developed by Bell Labs, but was later modified by Bell Labs and Motorola. The following years, many other countries adopted the network, including Israel, Australia, Singapore and Pakistan. AMPS was the direct upgrade to IMTS, aimed to improve many of the limitations the previous system had, by utilizing the methods we mentioned previously. While succeeding to improve the previous generation, AMPS also had weaknesses due to the analog nature of the network. All analog systems are susceptible to static and noise by other, usually more powerful signals while also offering no protection from eavesdropping and frequency hijacking with the use of scanners and transmitters. In fact, analog systems were prone enough, that an old TV set that was able to tune in specific channels could hijack the frequency, effectively making spying on calls very easy with basic equipment [3].

In 1992, the first smartphone was introduced, named IBM Simon, which used the AMPS network. A few years later, the smartphone became commercially available to buyers, effectively making it the first widely used smartphone. With equipment capable of receiving and transmitting signals to an AMPS network now being available to the public, many malicious actors figured out ways of "cloning" subscriptions into other phones, tricking the network into making calls without paying. Essentially, they would intercept the Electronic Serial Number, which was a 12-digit number that was sent for billing purposes, and the Mobile Directory Number or Cellular Telephone Number, essentially the phone number. After having access to one of these pairs, they could then clone that combination onto a different phone, allowing them to make phone calls in different areas by posing as the original subscriber. The following years with the release of newer smartphones made this problem became significantly worse since smartphones that were capable of "cloning" right out-of-the-box (such as the Oki 900) were now available to the public (the Oki 900 could also eavesdrop without any further modifications needed). This issue ended up costing cellular carriers millions of dollars, forcing them to implement some sort of security features to the network [3].

The first security addition was the installation of a PIN system into the network, that would require the users to provide a PIN before making any call. Due to the nature of static PINs, eventually they were also forced to install a system called Radiofrequency (RF) Fingerprinting, which, essentially, determined whether a phone was different from one another while making a phone call, shutting down any phone that wouldn't match the original. The latter, ended up being impractical due to RF changes that would occur in the change of a battery in a phone or the replacement of a broken antenna. Since malicious actors required specialized receivers to obtain the pair required for cloning, a law was passed in the US forbidding the sale of any receiver that was capable of tuning into the frequencies that were occupied by the AMPS network, further reducing the number of cloned phones that existed. Improvements to the system were steadily being developed with the goal of improving the drawbacks of pure analog communication and eventually a new system was developed named the Digital AMPS (D-AMPS) [4].

1.1.3 The Second Generation (2G)

The second generation of cellular networks was the result of the further development of AMPS, the D-AMPS, which was the evolution of the pre-existing system to a fully digital one. Initially, many of the already established AMPS networks were partially converted to D-AMPS, which happened a few years before its standardization. D-AMPS (also referred to as Time-division Multiple Access – TDMA) was commercially deployed in 1993 and was the standard that 2G networks initially used [3] [5].

In North America, D-AMPS was later evolved to IS-54 and IS-136. In Europe, another standard was being developed based on TDMA, the Global System for Mobile Communications (GSM), or more accurately the TDMA-based GSM. GSM development began in the early 1980s as a standard for digital cellular voice telecommunications in France. The following years, 13 more European countries signed a memorandum of understanding, to develop and deploy a common cellular system across Europe. Following this, the European Union passed rules mandating the GSM as the standard to be used for telecommunications for the member countries. After the agreement, the first GSM technical specifications were produced, with the first networks being deployed in France and Germany, followed by Italy and the UK. The first practical GSM mobile phones became available to the public in the beginning of the 1990s [6].

With the continuation of development, new services were developed, like fax, data and SMS messaging, which were commercially launched in the middle of the 1990s. A GSM network also became available in the United States (although it was not the one that was primarily used) and a year later pre-paid SIM cards were available to the public. At that point, the subscribers for GSM exceeded 10 million world-wide, which shows an extreme increase in network capacity over the previous generation [6]. In the United States a new modulation technique was being developed called Code-division Multiple Access (CDMA), which was used in a new cellular technology named cdmaOne (IS-95). Unlike TDMA, CDMA permits several radios to transmit in the same frequency, without causing interference between one another by using a special coding scheme (thus the getting the name), where each transmitter would have an assigned code which would then be decoded at the receiver to extract the data. This resulted in further improvements of the utilization of the frequency spectrum, increasing capacity even more [7].

The 2G systems eventually implemented a packet-switched domain in addition to the previous circuit-switched domain. Those services were named General Packet Radio Service (GPRS) and they provided the new feature of Multimedia Messaging Service (MMS). This very system would later go on and allow access to the internet. With further development of modulation schemes, the 8 Phase Shift Keying (8 PSK) encoding was introduced, which allowed the transmission of 3 times the information without increasing the symbol rate. This led to the creation of Enhanced Data rates for GSM Evolution (EDGE), which was effectively an overlay or an extension to the already existing GSM, offering higher transfer rates for the supported devices. Further enhancements of GPRS lead to the creation of new standards that were used for the following generations. By the end of this generation the benefits over the previous one was [8]:

1. Fully digitally encrypted phone conversations (between the user equipment and the cellular base station, not throughout the whole network) that made the act of eavesdropping difficult and not easily possible.
2. Significant improvement of the utilization of the available radio frequency spectrum, increasing the number of subscribers and concurrent users significantly.
3. Introduction of services such as SMS and later MMS with the improvement of technology.

Despite the fact that 2G has been around for over 2 decades, it still is an available service in many countries, while its successor (3G) is being shut down. GSM networks are still used as a fallback service in many countries, primarily to the fact that newer generation infrastructure has not yet been implemented (like rural areas) but there was pressure being applied by organizations so the used frequencies should be reallocated for reuse in newer generations. By 2035, GSM is set to become obsolete [9].

1.1.4 The Third Generation (3G)

As with every new generation, the third generation (3G) brought significant improvements in multiple aspects of the previous one, focusing primarily on voice quality improvements and higher data transfer speeds. Around the end of the 1990s, a partnership between a number of standards organizations was formed by the name of the 3rd Generation Partnership Project (3GPP). The goal of this partnership was the long term, combined effort to have unified standards, specifications and technologies for 3G. They were also responsible for the maintenance of said standards and technologies [10].

With the combined effort of 3GPP, a new system was created named the Universal Mobile Telecommunications System (UMTS). The radio access technology that was used was based of CDMA, Wideband CDMA (W-CDMA) with the goal to offer a greater spectral efficiency and bandwidth to mobile network operators. W-CDMA allowed the existence of features such as streaming and broadband internet access due to the speeds it was able to carry data at. With its deployment in the early 2000s, features such as mobile TV and video calling were available. UMTS required the implementation of new base stations and frequency allocations, unlike EDGE that was a direct upgrade on GPRS's technical implementation which allowed reuse of already deployed infrastructure. There was also the creation of USB modems that could be used with portable computers, allowing them to connect to the internet via 3G [11].

As the development of the technologies continued, W-CDMA received several upgrades. The amalgamation of those upgrades and protocols created the High-Speed Packet Access (HSPA) which was a combination of the High-Speed Downlink Packet Access (HSDPA) and High-Speed Uplink Packet Access (HSUPA). With the introduction of HSPA, the generation progress was annotated as 3.5G or 3G+. The HSDPA protocol was aimed at improving the data rates and reducing the latency at the downlink, while also enhancing the support for high-performance packet data applications. The protocol achieved this by implementing a higher-order modulation, like 16 Quadrature Amplitude Modulation (16QAM). The implementation of this protocol to the existing UMTS system was as simple as a software update. The equivalent upgrade to the uplink shortly followed which was the HSDPA protocol. Similarly, the protocol was aimed on improving the uplink data rate while reducing the latency. This was achieved by allowing multi-code transmissions, reducing the time between transmissions and implementing fast hybrid automatic repeat request (HARQ) which was a combination of high-rate forward error correction (FEC) and automatic repeat request (ARQ) error-control [12], making retransmissions more effective [13].

HSPA continued to be improved and eventually its second phase was deployed, the HSPA+. This phase was the evolution of the 2 protocols that made it up, that were now named Evolved HSDPA (HSDPA+), Dual-Carrier HSDPA (DC-HSDPA) for the downlink and Dual-Carrier HSUPA (DC-HSDPA) for the uplink. HSDPA+ could now support multiple-input multiple-output (MIMO), as well as up to 64QAM. This allowed for a big increase in the theoretical transfer speeds that could be supported by a single carrier in good conditions, by increasing the throughput gain of the system. In the late 2000s, a new study was conducted by 3GPP, which lead to the creation of the DC-HSDPA. The basic idea of this version was to implement a multicarrier feature, with the goal to achieve better resource utilization and higher spectral efficiency across downlink carriers. Initially, support for only two carriers simultaneously was released. For user equipment to benefit from this protocol, additional antennas were added, as well as support for 64QAM transmission. To accommodate for the improved downlink, subsequently DC-HSUPA was released to improve the uplink. Similar to DC-HSDPA, it focused on increasing the simultaneous uplink carriers to improve spectral efficiency across the uplink and better utilize the available resources. Combining the two protocols marked the evolution of HSPA to HSPA+, with new user equipment that was fitting with more antennas to accommodate for concurrent, multi-carrier usage. Studies for usage of more than two carriers were conducted, eventually creating an 8-carrier HSPA with 4 x 4 MIMO (4 downlink antennas – 4 uplink antennas), reaching very high transfer rates. This standard was named Multi-Carrier HSPA (MC-HSPA)

With HSPA+, an option for a flattened all-IP architecture was an option, essentially having the base stations bypass legacy elements for the data connections, by directly connecting to the network via IP. This reduced the overall cost of the network while increasing its speed. In this architecture, the "user plane" would be directly connected from the base station to the Gateway GPRS support node (GGSN), using any available link. Due to the fact that the data flow would bypass the Radio Network Controller (RNC), the architecture was simplified while reducing operational costs and delays. This system would later go on and be used in 3GPP Long Term Evolution (LTE) that we will see in fourth generation networks. The first commercially available network to implement HSPA+ using this architecture was created by Nokia Siemens Networks and was named appropriately as the Internet HSPA (I-HSPA) [14].

Besides significant upgrades in transfer speeds and spectrum utilization, along with introduction on new features available to the end users, security received significant upgrades over 2G. In 3G, the encryption that previously was mostly between the user equipment and the base station, now was network wide, by implementing strong encryption algorithms that protected user data and communications. Data like voice calls and internet traffic were now protected to unauthorized access and eavesdropping. A variety of authentication procedures were also implemented to ensure that both the user equipment and the network were legitimate. These procedures helped to prevent man-in-the-middle attacks and unauthorized network access. Protection for the data integrity that was being transmitted was also installed, preventing data tampering by keeping the transmitted data unaltered. A variety of firewalls and intrusion detection systems along with network monitoring and auditing were also utilized to detect and respond quickly to any security breaches the system might had. Many of the new security features, such as encryptions and authentications required cryptographic keys that were now stored inside SIM cards, making cloning highly difficult. Finally, there were steps taken in protecting the subscriber’s privacy, such as location privacy, by heavily restricting access to location information [15].

The third generation brought a lot of new features to end users, while improving quality and security of the networks, laying the foundation for newer, even more powerful technologies, systems and networks to take its place in the following generations. To sum up, some of the key elements that 3G implemented were:

1. High enough speeds for mobile TV, video calls and internet surfing.
2. Significant increase to spectrum efficiency and resource utilization using new modulation techniques and MIMO protocols.
3. The first flattened all-IP architecture commercially available network.
4. Significant integrity and user privacy security implementations.

While the adaptation of 3G was a worldwide success, the system is set to completely shut down within this decade, while most countries have already shut down 3G accessibility. The driver for the global obsoletion of 3G was the fact that 4G is now being exclusively used and the services that 3G provided are now embedded in the newer generation, thus making it almost completely unused. By shutting the system down, they allow its frequencies to be reused and reallocated for use in newer generations [16].

1.1.5 The Fourth Generation (4G)

With increasing expectations of each new generation and technology, the International Telecommunication Union (ITU) mandated a list of minimum requirements that a system must meet in order to be considered a fourth-generation system. Those specifications were defined in the International Mobile Telecommunications (IMT) Advanced and which meant that a 4G system must:

- Be based on an all-IP packet switched network.
- Have specific peak data rates for high mobility and low mobility applications.
- Be able to dynamically share and utilize network resources (also known as cell optimization).
- Use scalable channel bandwidths that were in a specified frequency range.
- Have a spectral efficiency for both downlink and uplink that is between a specified range.
- Similarly, have a specific system spectral efficiency in indoor cases for both downlink and uplink, that is within a specified limit.
- Support and have smooth handovers across heterogeneous networks.
- Have specific physical layer technology on both base stations and user equipment.

One of the first systems that were commercially available was the 3GPP Long Term Evolution (LTE). It is important to note that LTE was not fully compliant with the above specifications, specifically it did not meet the specified peak data rates, thus being considered by many as a pre-4G system. The LTE standard was first released in Norway, followed by Sweden and has since been deployed in most parts of the world. Due to LTE not being fully compliant, many argued whether it should be considered a 4G system (along with WiMAX that was the equivalent standard released in South Korea) [17].

4G exclusively supported all-IP based communications (similar to I-HSPA), such as IP telephony (also referred to as Voice over IP - VoIP). Previous radio technologies were completely abandoned in all candidate systems and were replaced by OFDM multi-carrier transmissions along with other frequency-domain equalization schemes. Like in 3G, MIMO communication techniques were also utilized, increasing transfer rates and spectral efficiency along with the implementation of smart antennas, which were antennas that had smart signal processing algorithms [18]. The aforementioned additions, allowed the 4G systems to transmit at very high bit-rates despite the existence of multi-path radio propagation (referred to also as echoes) [19].

For the candidate systems to achieve the mandated specifications, it was observed that all of the proposed systems had the following key features [20]:

- The physical layer techniques that systems used were:
  - MIMO communication techniques to achieve very high spectral efficiency, including smart antenna, multi-antenna and multi-user MIMO.
  - Frequency-domain equalization like multi-carrier modulation, i.e., Orthogonal Frequency-Division Multiplexing (OFDM) in the downlink or single-carrier equalization in the uplink to avoid complex equalizations while utilizing the frequency selective channel property.
  - Frequency-domain statistical multiplexing like Orthogonal Frequency-Division Multiple Access (OFDMA) in the uplink, to allow for bit rate variation based on the channel conditions at any given time.
  - Turbo FEC codes to minimize the Signal to Noise Ratio (SNR) at the reception side.

- Channel- dependent scheduling to have the ability to use the time-varying channel.
- Adaptive modulation techniques to match the channel conditions for the best possible performance.
- Utilization of the Mobile IP communication protocol which allows any mobile device to move from one network to another without changing its IP address [21] improving overall system mobility.
- IP-based femtocells that would be connected to the fixed internet broadband infrastructure.

With the continuous increase of devices that are connected to the internet and the finite amount of IPv4 addresses, 4G systems were initially recommended to have IPv6 support across the whole network. In 2009, Verizon posted specifications, requiring all 4G devices to support IPv6 in order to connect to their network. IPv6 removed the need for network address translation (NAT), a technique that has a variety of limitations, including limitations of certain internet protocols, but 4G networks kept some support for NAT in order to have legacy support for older IPv4 systems that were using it [22].

In this generation, many improvements for security also took place, including stronger encryption algorithms and better authentication, as well as IP-based security since the network was now completely IP based. Overall, the systems that were in place for 3G received upgrades and improvements, with the addition to the aforementioned IP-based security. This included improved firewalls, detection systems, network monitoring and auditing of the whole network. The network also reduced the signaling overhead, preventing attacks like Denial of Service and was able to separate between services and traffic, isolating critical infrastructure and user data from one another [23].

The systems introduced a highly inconvenient situation for users that traveled internationally, essentially with the roaming capabilities. For a subscriber to make or receive 4G voice calls, they had to have a matching frequency band (since not all of the systems used the same bands and, in some cases, they required unblocking) and the matching enablement settings for the local carrier and/or country. In certain cases, making 4G calls in another carrier’s network is impossible without a software update that is specific to both the carrier and the model in question making this highly inconvenient for both carriers and subscribers. Some of the networks may fall back to 2G or 3G given a matching frequency band is available which, as we mentioned earlier, is the reason why 2G services still are available [24].

With LTE, technically, not being a 4G system, 3GPP began to develop a majorly improved version that would comply and even exceed the specifications that were set by ITU. The standard was named LTE Advanced (LTE+). 3GPP released the technical report “Requirements for further advancements for Evolved Universal Terrestrial Radio Access (E-UTRA) (LTE-Advanced)” [25] that was based on both ITU’s and operators’ requirements for LTE+. Many technical considerations were taken, including the following:

- The continuous improvement of the LTE radio technology and architecture.
- Variety of scenarios and performance requirements for compatibility with legacy systems.
- Backwards compatibility with already existing LTE systems, making existing infrastructure able to function as LTE+ systems as well, with any exceptions handled directly by 3GPP themselves.

With the above technical specifications in place, research began to create the ideal system that would meet or even exceed the standards. One of the features that was quickly developed, was the ability to utilize advanced topology networks, making the network a mix of macro, pico and femtocells [26], increasing the coverage and the capacity of the system significantly. LTE+ also introduced an ultra-wide bandwidth multi-carrier system to support very high data rates, exceeding the peak data rates that were set by ITU. Throughout its research, a variety of different proposals were made and studied, that can be roughly categorized as:

- Carrier aggregation, to increase the individual data rate of users.
- Enhanced precoding techniques and better FEC methods.
- Asymmetric bandwidth allocation for the frequency-division duplex.
- Autonomous network configuration, with the use of Self Organizing Networks (SONs) methodologies.
- Systems and methods to manage and suppress interference.

Eventually, LTE+ was able to use up to sixteen antennas (8x8 MIMO – 8 for the downlink and 8 for the uplink), as well as utilize up to 128QAM in the downlink [27]. With further advancements, the constellation size increased to 256QAM, allowing even higher download speeds in good conditions.

The technologies began to slowly being implemented to existing networks thanks to the backwards compatibility LTE+ had with LTE, and in 2011, trials began on equipment. Slowly, vendors that already had LTE networks install received software upgrades to accommodate the ongoing improvements. In 2012, the deployment of small cells in areas with high traffic or poor reception took place, utilizing this way the advanced network topologies that LTE+ supported, increasing the Quality of Service (QoS) for subscribers. As development continued, there was concentration to improve the support of the advanced topologies, which resulted in the release of the Coordinated Multi-Point operation (CoMP), allowing user equipment to better utilize the existence of multiple cells. The increase of cells also meant the increase of handovers which could cause inconsistencies. Later releases of LTE+ aimed to address this issue, while including support for 256QAM in the downlink [28].

It is expected, that by 2025, 55% of the worldwide telecommunication market will still be 4G technologies, while 5G will consist of 25% of the market [29]. LTE+ is still primarily used throughout the world, while also being the primary fallback service for 5G networks when the conditions are not fit. Overall, 4G brought incredible breakthroughs in wireless telecommunication networks and laid the foundation for even bigger breakthroughs for the next generations.

1.1.5 The Fifth Generation (5G)

The most recent commercially available wireless cellular network generation is 5G. The fifth generation of cellular networks began deploying worldwide in 2019, offering a plethora of upgrades over its predecessor, 4G. 5G networks are, essentially, the continuous evolution of LTE+, which was named LTE+ Pro. This standard is considered a pre-5G technology and with its release it brought new concepts to cellular networks, such as License Assisted Access, which allowed the sharing of both licensed and unlicensed spectrum. It also implemented various technologies associated with 5G, such as 256QAM constellations along with massive MIMO [30].

3GPP began by defining new standards for this generation in the same way it did for 4G. For a system to be considered 5G, it must use the new 5G New Radio (NR) software. Some of the key technical specifications that were set for 5G NR are:

- Incorporation of the Massive MIMO technology, which would enable the use of multiple antennas on both the base station and the user equipment, enhancing the overall spectral efficiency, capacity, coverage and network performance.

- Support for a wide range of frequency bands, including low, mid and high-frequency bands. This includes sub-6 GHz bands and millimeter wave (mmWave) bands, allowing flexibility in deployment by having diverse options to meet different requirements.
- Use of advanced beamforming techniques to focus radio signals toward specific user devices, increasing signal strength and reducing interference. This improved the efficiency and reliability of communication in high-frequency bands like mmWave.
- Low enough latency in order to support mission critical applications that require uninterrupted and robust data exchange, such as autonomous vehicles and remote surgery.
- Very high data rates, peaking up to multiple gigabits per second, allowing for extremely fast downloads and video streaming, improving the overall user experience.
- Support for a massive number of simultaneously connected devices, promoting the improvement of Internet of Things (IoT) applications, both industrial and small-scale (like smart homes).
- Significant reduction of the overall power consumption, reducing the overall impact of mobile communications on the environment.
- Implement network slicing techniques, such as Software Defined Networking (SDN) and Network Function Virtualization (NFV), in order to create multiple virtual networks on a single physical network infrastructure.

All of the candidate systems for 5G have to use and meet the above specifications in order to be considered a 5G technology. Overall, 3GPP's targets are the following [31]:

- Have a significantly higher peak data rate.
- Have significantly higher mobility and network robustness.
- Have approximately 10% of 4G's energy consumption.
- Have approximately 4 times the spectral efficiency of 4G.

The idea is to develop a super-fast, super-reliable system with incredibly high coverage that will eventually compete with Internet Service Providers (ISPs) [32] [33].

The frequency ranges that are used by 5G NR are separated into 2 different groups:

- Frequency Range 1 (FR1), for bands within 410 MHz – 6 GHz.
- Frequency Range 2 (FR2), for bands within 24.250 GHz – 71 GHz.

The first group, FR1, that is also referred to as the sub-6 GHz, has a maximum channel bandwidth of 100 MHz due to scarcity of spectrum in this range. Due to this, the most widely used band is in the range of 3.3 – 4.2 GHz. Sometimes this range is also referred to as the mid-band, referring to the frequencies that have not being used in previous generations [34].

The second group, FR2, that is also referred to as mmWaves, has a channel bandwidth between 50 MHz and 400 MHz. The increase in bandwidth allows a significant increase to transfer speeds, reaching network throughputs up to 20 Gbps, while reducing the overall network latency. They also have significantly lower overhead cost which theoretically should also reduce the cost per network connection. Unfortunately, mmWave does carry some significant disadvantages. Due to the wave's nature (being very short), it does not offer a very high obstruction penetration unlike other bands, requiring the user equipment to effectively be in the line-of-sight of the base station to utilize its benefits.

Generally, high frequency radio waves also have a reduced range of about 300 meters, requiring a denser placement of the cells. Due to the denser placements, smaller cells must be considered, requiring a higher number of antennas in each cell [31].

4G, as we mentioned, established a very strong multi-carrier modulation technique named OFDM. 5G is also based on the same modulation technique along with 5G NR. While 5G does share the same mobile networking principles with 4G, the introduction of 5G NR allows OFDM to deliver a much higher degree of flexibility and scalability. OFDM, as we will see later on in Chapter 2, is an extremely power modulation technique that is being considered for use in future generation networks as well, with new modulation techniques that are based on OFDM being developed and tested. Even though 5G uses the same modulation technique, it is important to note that 4G compatible user equipment cannot be used with 5G, due to the physical layer upgrades it received with its development. The next expected evolution of 5G is expected to be 5G-Advanced (5G+) [32].

While the new generation did bring updated security features, such as even stronger encryption algorithms, many security concerns began to arise, even before the commercialization of 5G, with papers being released stating that with 5G a new era of security threats will begin. Issues related to the infrastructure equipment also began to arise, with the European Commission stating against the usage of a single supplier for 5G infrastructure, especially from those based outside of the EU. Due to political conflicts, several countries like the US and the UK took action against the usage of Chinese infrastructure, with the fear of potential espionage [35].

With the increase of signal power along with the increase of frequency bands, concerns about the emitted electromagnetic interference began to arise. Issues regarding open bands, like the 6 GHz band that is used in Wi-Fi, could experience interference with the emission of 5G signals, resulting in signal drops or reduced data transfers. This issue is also present on a variety of electronic devices, such as TVs, radios and more importantly medical equipment, like pacemakers and other medical implants resulting in malfunctions which could endanger individuals dependent on such systems [36]. Systems like radar altimeters on aircraft in the US could also experience such malfunctions, due to the fact that they operate between 4.2 and 4.4 GHz. In Europe, 5G bands use lower frequencies, but similar concerns were expressed, recommending 5G devices to be turned off or placed in airplane mode during flights [37]. European networks that function in frequencies between 3.4 and 4.2 GHz, can interfere with satellite signals, although this interference can be mitigated with the use of proper filters [38].

1.2 Beyond 5G (B5G)

1.2.1 What is B5G

The term “Beyond 5G” refers to the research and development efforts that are focused on technologies that go beyond the original capabilities of 5G systems. While the term implies that the ultimate goal is the eventual development of the next generation of mobile networks, the primary focus is to lay foundations for new technologies, that eventually will become new generation systems [39].

1.2.2 The Importance of B5G

With B5G, the combined efforts of multiple organizations allow researchers to work on new technologies by providing the appropriate funding and promoting innovation. All of the parties primarily aim to advance the research and development, while also looking into solution to address the global connectivity challenges the current generation (5G) is facing. Overall, the existence of B5G benefits everyone, including research institutes by providing appropriate funding, all involved technology companies by developing new equipment to accommodate new standards (which can also be given to researchers to work and develop on actual hardware), as well as every user that utilizes mobile networks, especially the ones that use the more sophisticated features, such as high-speed video streaming and autonomous vehicle development [40].

1.2.3 The B5G Concepts

It is clear, that the ultimate goal is the evolution of 5G into 5G+ which will ultimately lead to the sixth generation of cellular networks (6G). 6G is already (even though early) under development and it is set to be released by 2030, providing even faster speeds, lower latencies (especially in high mobility situations) and support for emerging applications [41].

Another concept that is being encompassed is Terahertz (THz) communications. As the name suggests, researchers are looking into utilizing extremely high frequencies for use in communications. While the data speeds that these frequencies achieve is extreme, they do have a variety of challenges that need to be address. Similar to mmWaves, THz signals grow weaker over large distances, effectively making the high-speed transmissions (which is the main reason for their development) highly difficult. Another issue, again similar to mmWaves, is the shallow penetration of obstacles, requiring effectively line-of-site to the base station as well. Finally, current limitations to equipment is making the technology less effective (in a sense a bottleneck) due to the inability of decoding a huge amount of data fast enough [42].

More communication-based concepts are being examined, such as utilizing the quantum principles to enable more secure and faster communication. One example is the use of Quantum Key Distribution (QKD), which provides an unbreakable encryption for data transmissions [43].

There is also research for advancements in satellite communications, which would allow connectivity for areas that pose a big technical and financial challenge, increasing the coverage and improving global connectivity. Combined with THz communication techniques, China successfully launched 13 satellites with the intended goal being to verify THz communications in space [44].

The use for artificial intelligence is also under consideration, specifically for autonomous network optimization. AI will be used to better handle the dynamic resource allocation, with the ultimate goal to improve the user experience. Better resource allocation will also increase the overall spectral efficiency of the system, allowing a greater number of devices to be connected at one time [45]. In combination with AI, IoT is another focus of B5G, aiming to integrate IoT networks in feature generation systems, expanding the capabilities of IoT applications [46].

Finally, different techniques and methods are being researched and developed in order to achieve the Ultra-Reliable Low Latency Communication (URLLC) that is in 5G's specifications. With reliable, low-latency networks, the implementation of mission critical applications, such as autonomous vehicles and remote medical procedures can be achieved [47]. One of the problems that the current generation is facing, is the significant performance loss in high mobility scenarios, such as a vehicle on the highway or a passenger in a high-speed train. This occurs due to the instability of the channel, requiring adaptive modulation to reduce constellation sizes. A technology was developed and is now under trials with the name Orthogonal Time Frequency Space (OTFS) modulation [48].

1.3 Thesis Objective

The goal of this thesis is to present, analyze and simulate OTFS modulation. Beginning from Chapter 2, we will start by laying the foundation by describing the OFDM modulation and its limitations, providing fundamental knowledge, such as the Fourier Transform and the Doppler effect. We will then describe the OTFS modulation scheme and its benefits over OFDM, providing the mathematical properties that the system is based on. Following this we will present different detector algorithms that can be found in the literature. In Chapter 3, we will discuss the implementation of the aforementioned algorithms in the simulation software MATLAB, analyzing the code while matching the equivalent mathematical equations that we will present in Chapter 2. We will also present full OTFS systems that use both synthetic and LTE channel models. In Chapter 4 we will present the simulation results, while giving context on why the parameters that were used were chosen. We will compare the different detector algorithms, as well as compare different constellation sizes in various speeds and channel conditions. Chapter 5 will be the final chapter and it will be the conclusion to the thesis. In that chapter we will provide a synopsis of the results, possible future work to be done along with some personal comments about the technology. At the end of the document, annexes containing the full MATLAB implementation can be found.

Chapter 2 Orthogonal Time Frequency Space (OTFS) Modulation

2.1 Laying the Foundation

2.1.1 Fourier Transform and the Fast Fourier Transform Algorithm

In signal processing, the Fourier Transform is a very important (and very commonly used) tool that allows us to move between the time and frequency domains, revealing very important information about the signals. In digital signal processing, the Discrete Fourier Transform is used, which can be described with the following mathematical formula:

$$\boldsymbol{y_{k+1} = \sum_{j=0}^{n-1} \omega^{jk} x_{j+1}} \tag{1}$$

where $\omega = e^{-2\pi i/n}$ is one of the n complex roots of unity and i is the imaginary unit. The Fourier transform also has different variations, some of which we will see when describing the OTFS system.

Various algorithms were developed that allowed Fourier Transform to be implemented in both software and hardware, which ultimately led to the creation of the Fast Fourier Transform (FFT). The FFT is a smart computational technique of sequentially combining progressively larger weighted sums of data samples, which in certain scenarios proved to be up to 50 times faster than previous Fourier Transform implementations [49]. The Fourier Transform is a reversable procedure, named appropriately as the Inverse Fourier Transform. The FFT is also able to calculate the inverse of the original procedure which is also appropriately named as the Inverse FFT (IFFT). The algorithm is implemented in many modulation techniques (like OFDM) and is also readily available in most simulation and mathematical software [50].

2.1.2 Orthogonal Frequency-Division Multiplexing (OFDM) Modulation

As we mentioned in Chapter 1, OFDM was originally used in 4G systems, due to its high spectral efficiency and the high data rates that was able to transmit. This modulation scheme was capable enough to also be used in 5G and now research and development continues, aiming to reuse it in applications beyond 5G. Combined with adaptive modulation, OFDM had the capability of adapting very well to poor channel conditions, maintaining an acceptable latency and error count, which is an essential part for mission critical applications planned to be available in future generations.

A basic OFDM transmitter system functions by initially receiving a sampled digital signal. The signal is then mapped (or modulated) using M-ary QAM (8,16,32 etc.) and passed to the IFFT algorithm. The resulting digital signal is split into two parts, the real part and the imaginary part. Both parts are then converted into an analog signal using a Digital-to-Analog Converter (DAC), with the imaginary part being encoded (with codes meeting the orthogonality property). Both of the parts are then convoluted into a carrier signal, which is finally transmitted.

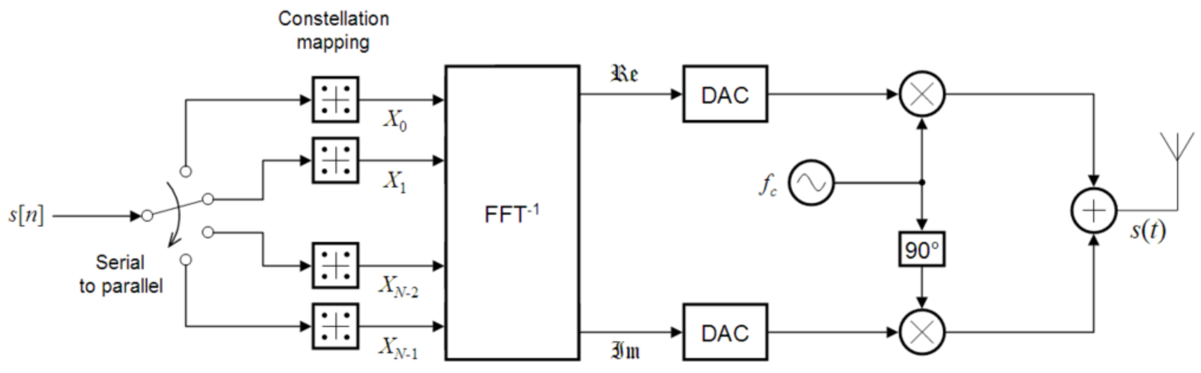

Figure 1: OFDM Transmitter System [62]

The reverse operations are followed at the receiver, initially receiving the carrier signal, separating the data from it. The resulting imaginary part is then decoded (using the same code as the one used when it was transmitted) and both of the parts are converted back into a digital signal using an Analog-to-Digital Converter (ADC) before being passed to the FFT algorithm. The resulting digital signal, depending on the QAM constellation, goes through a symbol detection operation (or simply - is demodulated), giving us the original signal, carrying the information that we initially transmitted.

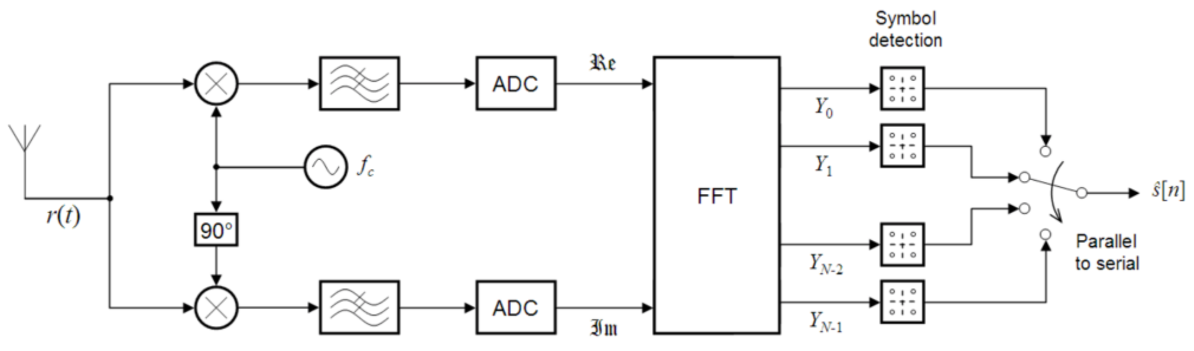

Figure 2: OFDM Receiver System [63]

It is important to note, that the aforementioned systems describe ideal transmitters and receivers. In reality, more aspects are implemented in a signal, such as FECs to reduce total errors, cyclic prefixes for time synchronization and addition of guard intervals between symbols to eliminate Inter-Symbol Interference (ISI) [51].

2.1.3 The Doppler Shift

The Doppler shift is a physics phenomenon that was first described by Christian Doppler in 1842. Doppler shift (or simply Doppler) refers to the apparent change in the frequency of a waveform in relation to an observer moving relative to the wave source [52]. Many of us have experienced this phenomenon in our day to day lives, with the most common example being the horn of a car changing in pitch depending on whether the car is moving away, towards or staying stationary in relation to us.

To better visualize this effect in a transmitted signal, we must first understand how a signal is transmitted. A single transmitted signal can be described as an expanding balloon that begins at the transmitter and expanding outwards (or in a 2D plane as an expanding circle). This is also known as signal broadcasting. Now assume that a signal is being transmitted, in fixed time intervals. Assuming that the transmitter is in a 2D plane, then the figure below shows the broadcasting of a signal (in this case an audio signal although this is irrelevant).

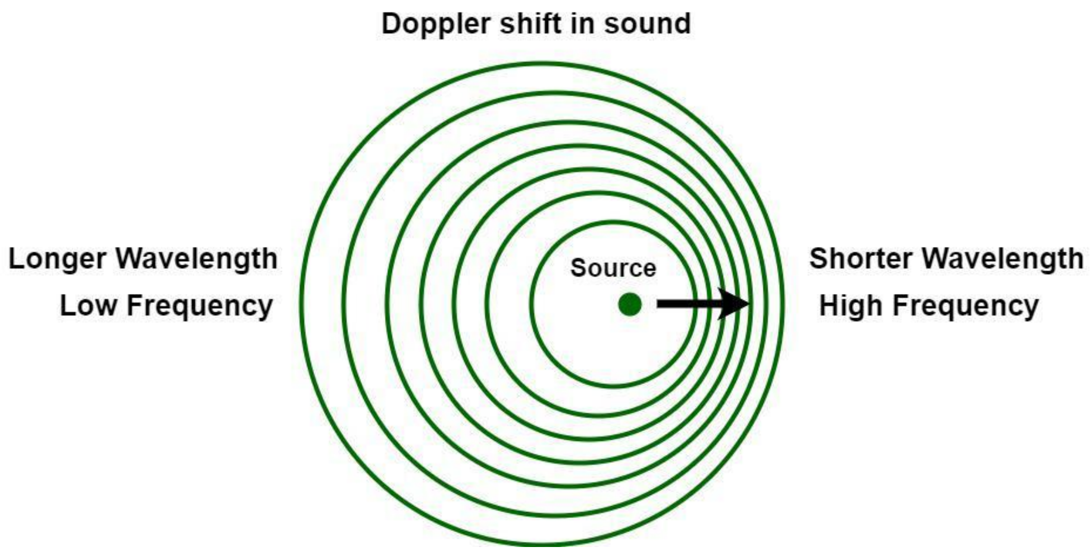

Figure 3: Doppler Shift in Sound [64]

We can describe the effect that the signals have when moving towards the observer as the “bunching up” of the transmitted signals. This effectively increases the frequency of the signals, which means that (assuming that the observer is the receiver) the transmitted interval appears lower. Similarly, when moving away from the observer, the signals can be described as “spreading out”, decreasing the frequency of the signals, effectively making the transmitted interval appear higher. This, especially in multi-carrier modulation techniques (like OFDM) causes Inter-Carrier Interference (ICI) and in combination of fading signals and signal copies (or echoes), can have various implications in telecommunication systems, both for the user equipment (i.e., a phone) and the stationary infrastructure, such as significantly reduced data transfer speeds, increase of errors and loss of communication (phone call drops) [53]. In simple terms, while low-mobility (or low-Doppler) user equipment, such as pedestrians, can enjoy very reliable and fast transfer speeds, the same cannot be said for high-mobility (or high-Doppler) user equipment, such as cars and trains [54]. It is important to note, that the increase in errors and reduction of transfer speeds does not only affect the overall user experience of subscribers, but also prevents the deployment of mission critical applications that experience high-Doppler like autonomous vehicles.

2.1.4 OFDM’s Limitations in High Mobility Scenarios

While OFDM has proven to be a very powerful modulation technique, capable to be used in systems that require ever-increasing standards and specifications, does have a significant limitation. As we mentioned previously, in high-mobility, so when there are significant Doppler shifts or simply the existence of high-Dopplers, multi-carrier modulations do suffer from ICI which can have significant impact in the overall system performance. While OFDM is capable of filtering out some of the Doppler shifts, when a high number of them are present it fails to successfully filter them, causing reduction in performance due to channel conditions. With the continuous development of technology, utilization of a large number of antennas (known as massive MIMO that we have already mentioned) has reduced the overall effect that Doppler shifts have, stabilizing the system. This technique has proven to be effective when we are dealing with line-of-sight (LoS) situations, but in urban scenarios where LoS is not possible, due to signal echoes, even with massive MIMO and smart antennas, the reduction of Doppler shift is very challenging [53], which shifted research in the development of new modulation technique, such as OTFS.

2.2 Introduction to OTFS

2.2.1 What is OTFS

OTFS is a recently emerged modulation technique that has been designed in the delay-Doppler domain. Due to this, OTFS is capable of fully utilizing the channel diversity of both the time and frequency domains, leading to the reduction of the overall system overheard, while also simplifying the physical layer. It achieves this by converting the fading, time-varying channels into time-independent channels with consistent channel gain across the symbols, resulting in a system that is stable for usage in mission critical (delay-sensitive) applications, which is a key specification for the future generation of networks. OTFS has been specifically developed for use in vehicle-to-vehicle (V2V) applications, like autonomous vehicle networks that we mentioned earlier, as well as vehicle-to-infrastructure (V2I) for reliable and fast connections when cruising in high speeds (i.e., in a car or a fast train), or more generally vehicle-to-everything (V2x) applications. We will procced in the analysis of OTFS, including the mathematical and physics properties that is based on, as presented in the paper “Orthogonal Time Frequency Space Modulation”, written by R. Hadani, S. Rakib, S. Kons, M. Tsatsanis, A. Monk, C. Ibars, J. Delfeld, Y. Hebron, A. J. Goldsmith, A.F. Molisch, and R. Calderbank and published on the 1st of August in 2018 [48].

2.2.2 Delay-Doppler Representation of Channels and Signals

Since the release of P. Bello’s paper “Characterization of Randomly Time-Variant Linear Channels” [55], it is known that a time-varying propagation channel can be represented by its:

- Time-varying impulse response.
- Time-varying transfer function.
- Doppler-variant impulse response.
- Doppler-variant transfer function (although this is rarely used).

Based on the propagation physics, the Doppler-variant impulse response is the best way to represent the channel. The complex baseband Doppler-variant impulse response $h_c(\tau, \nu)$ describes the channel response to an impulse, at delay τ and Doppler ν. The signal that is received when a signal $s(t)$ is transmitted over this channel is given by:

$$\boldsymbol{r(t) = \iint h_c(\tau,\nu)e^{j2\pi\nu(t-\tau)}s(t-\tau)d\tau d\nu} \tag{2}$$

Based on **(2)**, the received signal is the superposition of the echoes of the transmitted signal, where each echo is delaying by the path delay τ, frequency shifted by Doppler shift ν and weighted by the time-independent complex valued delay-Doppler impulse response $h_c(\tau,\nu)$ for that particular τ and ν.

In principle, **(2)** can be understood as a linear operator $\Pi_h(\cdot)$ that depends on the impulse response function $h = h_c(\tau,\nu)$. This operator operates on the input signal $s(t)$ to generate the output signal $r(t)$, that is:

$$\boldsymbol{\Pi_h(s){:}\ \ \mathbf{s(t)} \overset{\Pi_h}{\rightarrow} r(t)} \tag{3}$$

In mathematics, the parameterization of the operator $h \rightarrow \Pi_h$ is defined as the Heisenberg transform.

The fundamental idea behind the Heisenberg transform centers on the twisted convolution property, which can be seen as an extension of the convolution property of the Fourier transform into the non-commutative domain. The statement of the twisted convolution property is outlined in the following Lemma.

Lemma 1. Twisted convolution property. *Let Π_{h_1} and Π_{h_2} be two Heisenberg operators parameterized by functions $h_1(\tau,\nu)$ and $h_2(\tau,\nu)$ as defined in* **(2)** *and* **(3)**, *applied in composition to a signal $s(t)$. Then we have:*

$$\boldsymbol{\Pi_{h_2}\left(\Pi_{h_1}\big(s(t)\big)\right) = \Pi_h(s(t))} \tag{4}$$

where $h(\tau,\nu) = h_2(\tau,\nu) *_\sigma h_1(\tau,\nu)$ is the twisted convolution of $h_1(\tau,\nu)$ and $h_2(\tau,\nu)$, defined as:

$$\boldsymbol{h(\tau,\nu) = \iint h_2(\tau',\nu')h_1(\tau-\tau',\nu-\nu')e^{j2\pi\nu'(\tau-\tau')}d\tau' d\nu'} \tag{5}$$

Proof: Let

$$\boldsymbol{r_1(t) = \iint h_1(\tau,\nu)e^{j2\pi\nu(t-\tau)}s(t-\tau)d\tau d\nu} \tag{6}$$

$$\boldsymbol{r(t) = \iint h_2(\tau,\nu)e^{j2\pi\nu(t-\tau)}r_1(t-\tau)d\tau d\nu} \tag{7}$$

Substituting **(6)** into **(7)** we obtain after some algebra manipulations:

$$\boldsymbol{r(t) = \iint h(\tau,\nu)e^{j2\pi\nu(t-\tau)}s(t-\tau)d\tau d\nu} \tag{8}$$

with $h(\tau,\nu)$ given by **(5)**. ■

All modulations involving time-frequency characteristics (also known as multi-carrier modulations), can be unified under a model comprised by the following components:

- A lattice (also termed a grid) Λ in the time-frequency domain that samples the time and frequency axes at integer multiples of T and Δf respectively, that is:
$$\boldsymbol{\Lambda = \{(nT, m\Delta f): n, m \in \mathbb{Z}\}} \tag{9}$$
- A packet burst with total duration of NT seconds and total bandwidth of $M\Delta f$ Hz.
- A 2D sequency of modulated symbols $X[n,m]$ that we wish to transmit over a given packet burst, parametrized along a finite number of points of the lattice Λ with indices $n = 0,\dots,N-1$ and $m = 0,\dots,M-1$.
- A transmit pulse $g_{tx}(t)$ and associated receive pulse $g_{rx}(t)$ whose inner product is bi-orthogonal with respect to translations by integer multiples of T and frequency Δf, that is:

$$\boldsymbol{\int e^{-j2\pi m\Delta f(t-nT)} g_{rx}^{*}(t-nT)g_{tx}(t)dt = \delta(m)\delta(n)} \tag{10}$$

Time-frequency modulator: A modulator that operates in the time-frequency domain incorporating the above components, takes a 2D symbol sequence $X[n,m]$ defined on lattice Λ and maps it to a transmitted signal $s(t)$. The transmitted signal is formed by superimposing delay-and-modulate operations that are applied to the transmit pulse $g_{tx}(t)$ in the following manner:

$$\boldsymbol{s(t) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} X[n,m]e^{j2\pi m\Delta f(t-nT)}g_{tx}(t-nT)} \tag{11}$$

The rule expressed in **(11)**, can also be understood as a Heisenberg transform, in the same way that the channel can be interpreted as such in **(3)**. This gives us a Heisenberg operator $\Pi_X(\cdot)$ that depends on the symbol sequence $X[n,m]$ which is applied to the transmitted pulse shape $g_{tx}(t)$, which is:

$$\boldsymbol{s(t) = \Pi_X(g_{tx})} \tag{12}$$

The above interpretation enables us to view the received signal as the combination of two Heisenberg operators – one associated with the modulation rule and the other with the channel. The combination of the two operators (with **(2)** corresponding to the channel and **(12)** to the modulation rule), results in the received signal in the form $r(t) = \Pi_{h_c}\big(\Pi_X(g_{tx})\big) + \tilde{v}(t)$ where $\tilde{v}(t)$ is additive noise at the receiver input. By applying the twisted convolution property of Lemma 1, we have that $\Pi_{h_c}\big(\Pi_X(g_{tx})\big) = \Pi_{h_c *_\sigma X}(g_{tx})$, therefore $r(t)$ can be written more explicitly as:

$$\boldsymbol{r(t) = \iint f(\tau,\nu) e^{j2\pi\nu(t-\tau)} g_{tx}(t-\tau) d\tau d\nu + \tilde{v}(t)} \tag{13}$$

Time-frequency demodulator: To obtain adequate information for the symbol detection, we perform matched filtering on the received signal $r(t)$ using the channel-distorted pulses that carry the information (assuming that the additive noise is white and Gaussian). The matched filter initially calculates the cross-ambiguity function between the received signal $r(t)$ and the received pulse $g_{rx}(t)$. This function is denoted as $A_{g_{rx},r}(\tau,\nu)$ and is given by:

$$\boldsymbol{A_{g_{rx},r}(\tau,\nu) \triangleq \int e^{-j2\pi\nu(t-\tau)} g_{rx}^*(t-\tau) r(t) dt} \tag{14}$$

The cross-ambiguity function can be understood as a two-dimensional (delay-Doppler) correlation function. To obtain the matched filter output, the cross-ambiguity function is sampled along the points of the lattice Λ, i.e., at integer multiples of T and frequency Δf, resulting in a 2D sequence:

$$\boldsymbol{\hat{Y}[n,m] = A_{g_{rx},r}(\tau,\nu)|_{r=nT,\nu=m\Delta f}} \tag{15}$$

The above function, serves as a transform that maps the 1D continuous function $r(t)$ to the 2D sequence $\hat{Y}[n,m]$. This particular transform acts as an inverse of the discrete Heisenberg transform and in mathematics is referred to as the discrete Wigner transform. The discrete Wigner transform can be seen as the generalization of the OFDM demodulator, which corresponds to the mapping of a received OFDM signal back into modulated symbols on the frequency grid.

Time-frequency input-output relation: As we already established in **(13)**, the input of the matched filter $r(t)$ can be represented as the sum of a noise term $\tilde{v}(t)$ and a signal term $\Pi_f(g_{tx})$. The signal term is obtained as the Heisenberg operator that depends on the impulse response of $f(\tau,\nu)$ applied to the pulse shape $g_{tx}(t)$. This results in the output of the matched filter, prior to sampling, being the combination of two terms:

$$\boldsymbol{\hat{Y}(t,f) = A_{g_{rx},\Pi_f(g_{tx})}(\tau,\nu)_{|\tau=t,\nu=f} + A_{g_{rx},\tilde{v}}(\tau,\nu)_{|\tau=t,\nu=f}} \tag{16}$$

The second term on the right side represents the contribution of noise, which we denote as $V(t,f)$. The first term corresponds to the matched filter output in the absence of noise, which we denote as $Y(t,f)$. Upon direct calculation, it is revealed that the noise-free component can be expressed as the twisted convolution of three terms:

$$\boldsymbol{Y(t,f) = h_c(\tau,\nu) *_\sigma A_{g_{rx},g_{tx}}(\tau,\nu)} \tag{17}$$

Replacing the first term of **(16)** with **(17)**, results in:

$$\boldsymbol{\hat{Y}(t,f) = h_c(\tau,\nu) *_\sigma X[n,m] *_\sigma A_{g_{rx},g_{tx}}(\tau,\nu) + V(t,f)} \tag{18}$$

The estimation of the modulation symbols from the matched filter output can be obtain by the evaluation of the continuous function $\hat{Y}(t,f)$ along the points of the lattice Λ, which is:

$$\widehat{\boldsymbol{Y}}[\boldsymbol{n},\boldsymbol{m}] = \widehat{\boldsymbol{Y}}(\boldsymbol{t},\boldsymbol{f})|_{\boldsymbol{t}=\boldsymbol{nT},\boldsymbol{f}=\boldsymbol{m}\boldsymbol{\Delta f}} \tag{19}$$

To determine the end-to-end input output relation, we will first begin by considering the simplified scenario of an ideal channel $h_c(\tau,\nu) = \delta(\tau)\delta(\nu)$. In this case, the direct calculation of the right-hand side of **(18)**, results in:

$$\widehat{\boldsymbol{Y}}[\boldsymbol{n},\boldsymbol{m}] = \sum_{\boldsymbol{n}'=0}^{\boldsymbol{N}-1}\sum_{\boldsymbol{m}'=0}^{\boldsymbol{M}-1} \boldsymbol{X}[\boldsymbol{n}',\boldsymbol{m}'] \times \boldsymbol{A}_{\boldsymbol{g}_{rx},\boldsymbol{g}_{tx}}\big((\boldsymbol{n}-\boldsymbol{n}')\boldsymbol{T},(\boldsymbol{m}-\boldsymbol{m}')\boldsymbol{\Delta f}\big) + \boldsymbol{V}[\boldsymbol{n},\boldsymbol{m}] \tag{20}$$

where $V[n,m] = V(t,f)|_{t=nT,f=m\Delta f}$ is the sampling of the noise term along the lattice Λ. By using the bi-orthogonality condition described in **(10)**, we receive the following result:

$$\widehat{\boldsymbol{Y}}[\boldsymbol{n},\boldsymbol{m}] = \boldsymbol{X}[\boldsymbol{n},\boldsymbol{m}] + \boldsymbol{V}[\boldsymbol{n},\boldsymbol{m}] \tag{21}$$

We will now consider the matched filter output for more general channels that are described by a non-trivial impulse response $h_c(\tau,\nu)$. The impulse response has finite support, which is bound by the maximum delay and Doppler spreads $(\tau_{\max},\nu_{\max})$ of the reflectors/scatterers. For simplicity, we assume that the bi-orthogonality condition described in **(10)** holds robustly, meaning that the cross-ambiguity function vanishes in a neighborhood of each non-zero lattice point $(nT, m\Delta f)$ with size at least as large as the support of the channel response, which can be described as $A_{g_{rx},g_{tx}}(\tau,\nu) = 0$ for $\tau \in (nT - \tau_{\max}, nT + \tau_{\max}), \nu \in (m\Delta f - \nu_{\max}, m\Delta \mathrm{f} + \nu_{\max})$. Under the aforementioned assumptions, a direct calculation of the right-hand side of **(18)** results in the generalization of **(20)**, which can be seen below.

$$\widehat{\boldsymbol{Y}}[\boldsymbol{n},\boldsymbol{m}] = \sum_{\boldsymbol{n}'=0}^{\boldsymbol{N}-1}\sum_{\boldsymbol{m}'=0}^{\boldsymbol{M}-1} \boldsymbol{X}[\boldsymbol{n}',\boldsymbol{m}'] \times \iint \boldsymbol{e}^{-\boldsymbol{j}2\boldsymbol{\pi\nu\tau}} \boldsymbol{h}_{\boldsymbol{c}}(\boldsymbol{\tau},\boldsymbol{\nu}) \boldsymbol{A}_{\boldsymbol{g}_{rx},\boldsymbol{g}_{tx}}((\boldsymbol{n}-\boldsymbol{n}')\boldsymbol{T} - \boldsymbol{\tau},(\boldsymbol{m}-\boldsymbol{m}')\boldsymbol{\Delta f}-\boldsymbol{\nu})\boldsymbol{e}^{-\boldsymbol{j}2\boldsymbol{\pi}(\boldsymbol{m}'\boldsymbol{\Delta f\tau}-\boldsymbol{nT\nu})}\boldsymbol{d\tau d\nu} + \boldsymbol{V}[\boldsymbol{n},\boldsymbol{m}] \tag{22}$$

It can be easily verified that thanks to the robustness of the bi-orthogonality condition, only the zero term $n' = n, m' = m$ remains in the right-hand side of **(22)**, leading us to:

$$\widehat{\boldsymbol{Y}}[\boldsymbol{n},\boldsymbol{m}] = \boldsymbol{H}[\boldsymbol{n},\boldsymbol{m}]\boldsymbol{X}[\boldsymbol{n},\boldsymbol{m}] + \boldsymbol{V}[\boldsymbol{n},\boldsymbol{m}] \tag{23}$$

where the complex gain factor $H[n,m]$ is given by:

$$\boldsymbol{H}[\boldsymbol{n},\boldsymbol{m}] = \iint \boldsymbol{e}^{-\boldsymbol{j}2\boldsymbol{\pi\nu\tau}} \boldsymbol{h}_{\boldsymbol{c}}(\boldsymbol{\tau},\boldsymbol{\nu})\boldsymbol{e}^{-\boldsymbol{j}2\boldsymbol{\pi}(\boldsymbol{m\Delta f\tau}-\boldsymbol{nT\nu})}\boldsymbol{d\tau d\nu} \tag{24}$$

The Symplectic Fourier Transform (SFT) is a variant of the 2D Fourier transform that is naturally linked to the Fourier kernel $e^{-j2\pi(m\Delta f\tau - nT\nu)}$ used in **(24)** for converting between the delay-Doppler and time-frequency channel representations. We will specifically focus on a finite version of the transform called the Finite Symplectic Fourier Transform which is denoted by SFFT. The input to this transform is a 2D periodic sequence $x_p[k,l]$ with periods (M,N) and the output of the SFFT is a 2D periodic sequence $X_p[n,m] = \mathrm{SFFT}(x_p[k,l])$ with periods (N,M). It is important to note that the periods of the output are in reverse order.

The output and input sequences should be regarded as defined, respectively, along the points of the time-frequency lattice Λ (**(9)**) and the reciprocal delay-Doppler lattice $\Lambda^{\perp}$ that samples the delay axis at integer multiples of $\Delta\tau = \frac{1}{M\Delta f}$ and the Doppler axis at integer multiples of $\Delta\nu = \frac{1}{NT}$, i.e.:

$$\boldsymbol{\Lambda^{\perp} = \{(k\Delta\tau, l\Delta\nu): k, l \in \mathbb{Z}\}} \tag{25}$$

The output sequence is given by the following Fourier summation formula:

$$\boldsymbol{X_p[n,m] = \sum_{k=0}^{M-1}\sum_{l=0}^{N-1} x_p[k,l]e^{-j2\pi(\frac{mk}{M}-\frac{nl}{N})}} \tag{26}$$

It is important to note that the SFFT couples the frequency variable with the delay variable and the time variable with the Doppler variable using a minus sign. This particular coupling between the variables is referred to in mathematics as symplectic coupling.

The inverse to the SFFT transform (known as ISFFT), which is denoted by $x_p[k,l] = \mathrm{SFFT}^{-1}(X_p[n,m])$ is given by a similar summation formula:

$$\boldsymbol{x_p[k,l] = \frac{1}{MN}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} X_p[n,m]e^{-j2\pi(\frac{ln}{N}-\frac{km}{M})}} \tag{27}$$

The primary property of the SFFT is that it switches between the circular convolution and point-wise multiplication of periodic sequences, similar to the well-known convolution property of the conventional finite Fourier transform. This property is summarized in the following theorem.

Theorem 1. Symplectic convolution property. *Let $x_1[k,l]$ and $x_2[k,l]$ be periodic 2D sequences with periods (M,N). In addition, let $X_1[n,m] = SFFT(x_1[k,l])$ and $X_2[n,m] = SFFT(x_2[k,l])$ be the corresponding Fourier transforms. The following relation holds:*

$$\boldsymbol{SFFT(x_1[k,l] \circledast x_2[k,l]) = X_1[n,m]\, X_2[n,m]} \tag{28}$$

where $\circledast$ denotes 2D circular convolution.

Proof: Based on the definition of the SFFT, it is straight-forward to verify that translation in delay-Doppler converts into a linear phase in time-frequency:

$$\boldsymbol{\mathrm{SFFT}(x_2[k-k', l-l']) = X_2[n,m]e^{-j2\pi(\frac{mk\prime}{M}-\frac{nl\prime}{N})}} \tag{29}$$

Based on this result we can evaluate the SFFT of a circular convolution as:

$$\mathbf{SFFT}\left\{\sum_{k'=0}^{M-1}\sum_{l'=0}^{N-1} x_1[k',l']\, x_2[(k-k')modM,(l-l')modN]\right\}$$
$$= \sum_{k'=0}^{M-1}\sum_{l'=0}^{N-1} x_1[k',l'] X_2[n,m] e^{-j2\pi\left(\frac{mk'}{M}-\frac{nl'}{N}\right)} \tag{30}$$
$$= X_1[n,m]X_2[n,m]$$

yielding the desired result. ■

2.2.3 The OTFS System

It is well-established in the OTFS literature that the input symbols to an OTFS system are to be modulated using M-ary QAM. Based on the mathematics behind OTFS that we presented above and are also well-established in the OTFS literature, we are able to design a block diagram that describes, albeit in a high-level, an OTFS system.

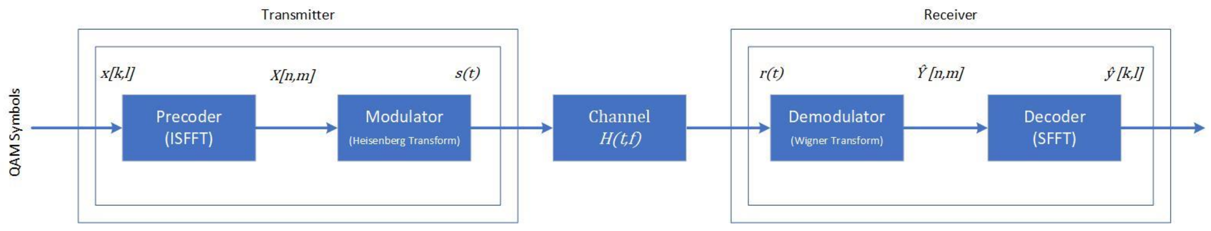

Figure 4: OTFS System Block Diagram [56]

Also established in the literature, is the fact that the modulator and demodulator are simply described as the "OFDM modulator" and "OFDM demodulator" respectively. Based on this fact, it is apparent that an OTFS system could be compatible with an LTE system, making its implementation as an overlay to a pre-existing OFDM system easy. In fact, the only thing that is necessary to be implemented is the precoder (or the ISFFT) which will take the modulated symbols from the delay-Doppler domain to the time-frequency domain and the decoder (or the SFFT) which will reverse the previous process. To summarize, the steps that are taken in an OTFS system to modulate the signals are:

1. Map the symbols using M-ary QAM.
2. Apply the ISFFT (**(27)**) to take said symbols from the delay-Doppler domain to the time-frequency domain.
3. Use the Heisenberg transform (**(11)** and **(12)** or simply use the "OFDM modulation") to modulate the signal.
4. Transmit the signal over the channel.

To demodulate the above signals, the same steps (but reversed) are taken, as listed below:

1. Receive the signal from the channel (**(2)**).
2. Use the Wigner transform (**(15)** or simply use the "OFDM demodulation") to demodulate the signal.
3. Apply the SFFT (**(26)**) to take the result from the demodulation to the delay-Doppler domain from the time-frequency domain.
4. Use M-ary QAM symbol detection to retrieve the symbols.

Due to the constant shift between domains, it might be difficult to understand where this transition happens so to better visualize the transition from delay-Doppler to time-frequency domain and vice versa, we can outline the domains in the block diagram that we presented above (**Figure 4**) [56].

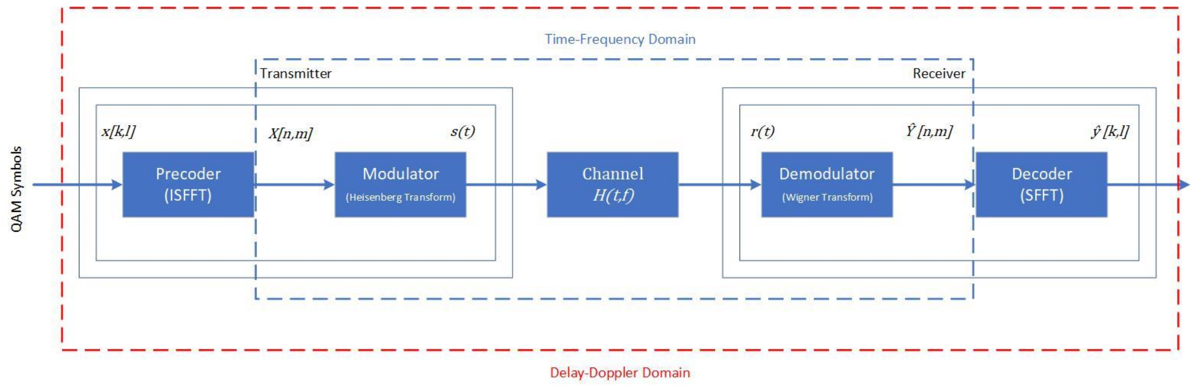

Figure 5: OTFS System Block Diagram (Domain Visualization) [56]

2.3 OTFS Detection Methods and Algorithms

2.3.1 Maximal Ratio Combining (MRC) in the Delay-Doppler Domain

When we discussed the limitations of OFDM, we mentioned that with the use of massive MIMO and smart antennas, reduction to the effect of the Doppler shifts can be achieved. What we did not mentioned is that in CDMA systems (such as OFDM), a special type of receiver is used to improve the reception in a multipath fading environment (like cities). These receivers are called Maximal Ratio Combining (MRC) Rake Receivers. This type of receivers, collect all of the faded and delayed echoes of the original signal and combine them to a single signal, successfully improving the reception.

By implementing the above detection method to OTFS, we are able to not only have a system that is able to efficiently operate in high-Doppler but relatively low-propagation channels (like highways), but also a system that can function well in a high-propagation channel with a non-insignificant presence of Doppler shifts. Such channels are city avenues or above-ground metro systems, which are moving relatively quickly while also being surrounded by large building and other obstacles that cause reflections, scattering and diffractions. The MRC algorithms that we will be discussing have been taken from the paper "Low Complexity Iterative Rake Decision Feedback Equalizer for Zero-Padded OTFS Systems", written by T. Thaj and E. Viterbo and was published in 2020 [57] [58].

The following pseudo-algorithm describes an MRC rake received in the delay-Doppler domain.

Algorithm 1: MRC in delay-Doppler domain.

```
1  Input: $\mathbf{H}, \mathbf{D}_m, \mathbf{y}_m, \mathbf{x}_m = \mathbf{0}_N \quad \forall\, m = 0, \dots, M-1$
2  for $i = 1 : max\ iterations$ do
3      for $m = 0 : M' - 1$ do
4          for $l \in \mathcal{L}$ do
5              $\mathbf{b}_m^l = \boldsymbol{y}_{m+l} - \sum_{p \neq l} \mathbf{K}_{m+l,p} \cdot \hat{\mathbf{x}}_{m+l-p}$
6          end
7          $\mathbf{g}_m = \sum_{l \in \mathcal{L}} \mathbf{K}_{m+l,l}^{\dagger} \cdot \mathbf{b}_m^l$
8          $\mathbf{c}_m = \mathbf{D}_m^{-1} \cdot \mathbf{g}_m$
9          $\hat{\mathbf{x}}_m = \boldsymbol{\mathcal{D}}(\mathbf{c}_m)$  (or  $\hat{\mathbf{x}}_m = \mathbf{c}_m$) $\rightarrow$ soft estimate
10     end
11 end
12 Output: $\hat{\mathbf{x}}_m$
```

We will begin by analyzing the inputs to the algorithm. Input H represents the OTFS channel matrix, D_m represents the diagonal matrix for each, non-zero symbol, y_m represents the received signal matrix, containing the symbols and x_m represents the initial estimate of the transmitted signal vector for each symbol and is initialized to 0. With the above in mind, we can now begin analyzing each step that the algorithm executes. With x_m having initialized all symbol estimates to 0, the algorithm executes the following steps:

1. For $i = 1$ repeat the following until we reach our max iteration limit:
    a. For each symbol, which is denoted by m, in the range of $M - 1$, where M is our symbol count (or our constellation size), calculate the following:
        i. For each $l \in \mathcal{L}$, calculate the residual vector, b_m^l, by subtracting the sum of the products, with the sum being the multiplication of $\mathrm{K}_{m+l,p}$ and the corresponding $\hat{\mathrm{x}}_{m+l-p}$, where $p \neq l$, from y_{m+l}. Note that l is an element of the set $\mathcal{L}$, different each time we compute the above. If we enumerate set $\mathcal{L}$, then we can say that l begins at the index 0, with the index incrementing with each repetition until we reach the end of said set.
        ii. Compute an intermediate vector g_m by summing up the Hermitian transpose of the equivalent entries of matrix K, for each $l \in \mathcal{L}$, multiplied by b_m^l which we calculated previously.
        iii. Compute the MRC output c_m by multiplying the inversed matrix D_m^{-1} with the intermediate vector g_m.
        iv. Finally, assuming that we utilize the soft estimate, we can update the soft estimate of the symbol of the current iteration, storing it in vector $\hat{\mathrm{x}}_m$. It is important to mention that if we do use soft estimation, the final decision of the symbols will also be based on its probability (this is also known as a Decision Feedback Equalizer (DFE)).

After all of the iterations are finished, we will receive $\hat{\mathrm{x}}_m$, which will contain the soft estimate of each transmitted symbol that was received from the signal vector y_m.

2.3.2 Reduced Complexity MRC in the Delay-Time Domain

While all MRC algorithms are considered low-complexity, the above algorithm does compute more b_m^l vectors than necessary, meaning that the step that computes the residual vector is making redundant calculations. This, presumably, might not seem like a serious issue, but is important to understand that this receiver is meant to be executed each time a new signal is received. In B5G systems, especially when Terahertz transmission technology is being considered, delay at the receivers is a luxury those networks cannot afford to have. This means that the receivers must be as optimized as possible to efficiently detect signals and symbols, even when the signals are received in incredibly small intervals. With that being said, it leads to the obvious conclusion that we must consider reducing the complexity of the algorithms as much as possible to ensure the smooth operation of the physical layer (receiver).

We will begin by defining the residual noise plus interference (RNPI) term in the i-th iteration:

$$\boldsymbol{\Delta y_m^{(i)}} = \mathbf{y}_{\boldsymbol{m}} - \sum_{l \in \mathcal{L}} \mathbf{K}_{\boldsymbol{m,l}} \cdot \hat{\mathbf{x}}_{\boldsymbol{m-l}}^{(\boldsymbol{i})} \tag{31}$$

which can be perceived as the residual error in the received delay-Doppler domain symbols due to error in estimation of the transmitted symbols after reconstruction. Based on **(31)**, b_m^l computation that estimates x_m, for the i-th, can be expressed as:

$$\mathbf{b}_{\boldsymbol{m}}^{\boldsymbol{l}} = \boldsymbol{\Delta y_{m+l}^{(i)}} + \mathbf{K}_{\boldsymbol{m+l,l}} \cdot \hat{\mathbf{x}}_{\boldsymbol{m}}^{(\boldsymbol{i-1})} \tag{32}$$

If we now substitute **(32)** in the computation equation of g_m, we can avoid the direct calculation of b_m^l by writing g_m, for the i-th iteration, as:

$$\begin{aligned} \mathbf{g}_{\boldsymbol{m}}^{(\boldsymbol{i})} &= \sum_{l \in \mathcal{L}} \mathbf{K}_{\boldsymbol{m+l,l}}^{\dagger} \cdot \boldsymbol{\Delta y_{m+l}^{(i)}} + \left(\sum_{l \in \mathcal{L}} \mathbf{K}_{\boldsymbol{m+l,l}}^{\dagger} \cdot \mathbf{K}_{\boldsymbol{m+l,l}} \right) \cdot \hat{\mathbf{x}}_{\boldsymbol{m}}^{(\boldsymbol{i-1})} \\ &= \sum_{l \in \mathcal{L}} \mathbf{K}_{\boldsymbol{m+l,l}} \cdot \boldsymbol{\Delta y_{m+l}^{(i)}} + \mathbf{D}_{\boldsymbol{m}} \cdot \hat{\mathbf{x}}_{\boldsymbol{m}}^{(\boldsymbol{i-1})} \end{aligned} \tag{33}$$

Based on **(33)**, we can now rewrite the MRC output at the i-th iteration as:

$$\mathbf{c}_{\boldsymbol{m}}^{(\boldsymbol{i})} = \hat{\mathbf{x}}_{\boldsymbol{m}}^{(\boldsymbol{i-1})} + \mathbf{D}_{\boldsymbol{m}}^{-\mathbf{1}} \cdot \boldsymbol{\Delta g_m^{(i)}} \tag{34}$$

where:

$$\boldsymbol{\Delta g_m^{(i)}} = \sum_{l \in \mathcal{L}} \mathbf{K}_{\boldsymbol{m+l,l}}^{\dagger} \cdot \boldsymbol{\Delta y_{m+l}^{(i)}} \tag{35}$$

which is the maximal ration combining of the RNPI's in all the delay branches that contain a symbol estimation.

The complexity of **(31)** can be further reduced by storing and updating the initial RNPI vectors $\Delta \mathrm{y}_{m+l}^{(0)}$. The L number of vectors which contain a component of the last estimated symbol are updated based on the following:

$$\mathbf{\Delta y}_{m+l}^{(i)} \leftarrow \mathbf{\Delta y}_{m+l}^{(i)} - \mathbf{K}_{m+l,l} \cdot (\mathbf{x}_m^{(i)} - \mathbf{x}_m^{(i-1)}) \tag{36}$$

Assuming that the superscript ~ denotes the *N*-IFFT of a vector (i.e., $\tilde{a} = \mathrm{F}_N^{\mathrm{H}} \cdot a$), we can write **(34)**, **(35)** and **(36)** in the corresponding delay-time domain as follows:

$$\tilde{\mathbf{c}}_m^{(i)} = \tilde{\mathbf{x}}_m^{(i-1)} + \mathbf{\Delta}\tilde{\mathbf{g}}_m^{(i)} \oslash \tilde{\mathbf{d}}_m \tag{37}$$

$$\mathbf{\Delta}\tilde{\mathbf{g}}_m^{(i)} = \sum_{l \in \mathcal{L}} \tilde{\boldsymbol{\nu}}_{m+l,l}^{*} \circ \mathbf{\Delta}\tilde{\mathbf{y}}_{m+l}^{(i)} \tag{38}$$

$$\mathbf{\Delta}\tilde{\mathbf{y}}_{m+l}^{(i)} \leftarrow \mathbf{\Delta}\tilde{\mathbf{y}}_{m+l}^{(i)} - \tilde{\boldsymbol{\nu}}_{m+l,l}^{*} \circ (\tilde{\mathbf{x}}_m^{(i)} - \tilde{\mathbf{x}}_m^{(i-1)}) \tag{39}$$

where:

$$\tilde{\mathbf{d}}_m = \sum_{l \in \mathcal{L}} \tilde{\boldsymbol{\nu}}_{m+l,l}^{\dagger} \circ \tilde{\boldsymbol{\nu}}_{m+l,l} \tag{40}$$

With the above optimizations, we can rewrite Algorithm 1 as follows:

Algorithm 2: Reduced complexity MRC in delay-time domain.

```
1   Input: $\tilde{\mathbf{H}}, \tilde{\mathbf{d}}_m, \tilde{\mathbf{x}}_m^{(0)}, \tilde{\mathbf{y}}_m \quad \forall\, m = 0, \dots, M-1$
2   for $m = 0 : M' - 1$ do
3           $\mathbf{\Delta}\tilde{\mathbf{y}}_m^{(0)} = \tilde{\mathbf{y}}_m - \sum_{l \in \mathcal{L}} \tilde{\boldsymbol{\nu}}_{m,l} \circ \tilde{\mathbf{x}}_{m-l}^{(0)}$
4   end
5   for $i = 1 : max\ iterations$ do
6           $\mathbf{\Delta}\tilde{\mathbf{y}}^{(i)} = \mathbf{\Delta}\tilde{\mathbf{y}}^{(i-1)}$
7           for $m = 0 : M' - 1$ do
8                   $\mathbf{\Delta}\tilde{\mathbf{g}}_m^{(i)} = \sum_{l \in \mathcal{L}} \tilde{\boldsymbol{\nu}}_{m+l,l}^{*} \circ \mathbf{\Delta}\tilde{\mathbf{y}}_{m+l}^{(i)}$
9                   $\tilde{\mathbf{c}}_m^{(i)} = \tilde{\mathbf{x}}_m^{(i-1)} + \mathbf{\Delta}\tilde{\mathbf{g}}_m^{(i)} \oslash \tilde{\mathbf{d}}_m$
10                  $\tilde{\mathbf{x}}_m^{(i)} = \mathbf{F}_N^{\dagger} \cdot \boldsymbol{\mathcal{D}}\left(\mathbf{F}_N \cdot \tilde{\mathbf{c}}_m^{(i)}\right)$  (or  $\tilde{\mathbf{x}}_m^{(i)} = \tilde{\mathbf{c}}_m^{(i)}$)
11                  for $l \in \mathcal{L}$ do
12                          $\mathbf{\Delta}\tilde{\mathbf{y}}_{m+l}^{(i)} \leftarrow \mathbf{\Delta}\tilde{\mathbf{y}}_{m+l}^{(i)} - \tilde{\boldsymbol{\nu}}_{m+l,l} \circ (\tilde{\mathbf{x}}_m^{(i)} - \tilde{\mathbf{x}}_m^{(i-1)})$
13                  end
14          end
15          if ($\| \mathbf{\Delta}\tilde{\mathbf{y}}^{(i)} \| \geq \| \mathbf{\Delta}\tilde{\mathbf{y}}^{(i-1)} \|$) then EXIT
16  end
17  Output: $\hat{\mathbf{x}}_m = \boldsymbol{\mathcal{D}}(\mathbf{F}_N \cdot \tilde{\mathbf{x}}_m)$
```

We should note that it is possible for the algorithm to finish without completing all of the specified iterations, when the overall RNPI error stops reducing due to the estimation error. With that noted, we can begin analyzing algorithm 2 with more detail. To begin, the following inputs must be provided to the algorithm. Matrix $\tilde{\mathrm{H}}$ represents the channel in the delay-Doppler domain. $\tilde{\mathrm{d}}_m$ represents the elements of D_m, also in the delay-Doppler domain. $\tilde{\mathrm{x}}_m^{(0)}$ and $\tilde{\mathrm{y}}_m$ represent the delay-Doppler values, equivalent to the same values they represent in algorithm 1 (including the initialization before the execution). After the parameters are given (and the initialization is done) the steps that the algorithm executes are the following:

1. For each symbol, which is denoted by m, in the range of $M-1$, where M is the number of subcarriers, calculate the following:
    a. Subtract the interference from other paths, resulting in the initial estimation of the received signal.
2. For $i=1$ repeat the following until we reach our max iteration limit:
    a. Move the previous element of the RNPI to the current index in the vector $\Delta\tilde{y}_m$, or more simply, replace the current RNPI element with the previous one.
    b. For each symbol, which is denoted by m, in the range of $M-1$, where M is the number of subcarriers, calculate the following:
        i. Calculate $\Delta\tilde{g}_m^{(i)}$ by summing the elements of the Hadamard products of $\tilde{v}_{m+l,l}^{*}$ and $\Delta\tilde{y}_{m+l}^{(i)}$, for each $l \in \mathcal{L}$.
        ii. Compute $\tilde{c}_m^{(i)}$ by first performing the Hadamard division between the current RNPI element and the input $\tilde{d}_m$. Then add the previous symbol estimate to that result.
        iii. Update the current symbol estimate (soft estimate) with $\tilde{c}_m^{(i)}$.
        iv. For each $l \in \mathcal{L}$, update the residual received signals $(\Delta\tilde{y}_{m+l}^{(i)})$ by subtracting the interference term $\tilde{v}_{m+l,l} \circ (\tilde{x}_m^{(i)} - \tilde{x}_m^{(i-1)})$ with the previous calculation of $\Delta\tilde{y}_{m+l}^{(i)}$ (in coding terms we can say that we subtract itself with the interference term and store the result back into the same variable).
    c. Check whether the overall RNPI stops reducing. In that case finish the algorithm.

The result is the same as in algorithm 1, that being $\hat{x}_m$, which is the soft estimate of the transmitted signal vector for each symbol.

2.3.3 Delay-Time MRC Operation Principle using the Gauss-Seidel Method

The last algorithm that we will be discussing is another MRC algorithm, implemented by using the properties of Jacobi's and Gauss Seidel's methods for solving linear equations. By utilizing these methods, we can simplify the equations that the lines 5-16 of algorithm 2 implement as:

$$\tilde{\mathbf{x}}^{(i)} = \tilde{\mathbf{x}}^{(i-1)} + \tilde{\mathbf{D}}^{-1}\tilde{\mathbf{H}}^{\dagger}(\tilde{\mathbf{y}} - \tilde{\mathbf{H}}\tilde{\mathbf{x}}^{(i-1)}) \tag{41}$$

when using parallel updates. $\tilde{D}$ denotes a matrix that contains the diagonal elements of $\tilde{H}^{\dagger}\tilde{H}$. Using the permutation matrix P, we can shuffle the rows and columns of the delay-time matrix $\tilde{H}$ perfectly. We can then acquire time-domain matrix G, which is similar to $\tilde{H}$. This fact, allows us to parallelize **(41)**, enabling us to compute each of the independent time-domain blocks, denoted as G_n, as

$$\mathbf{s}_n^{(i)} = \mathbf{s}_n^{(i-1)} + \mathbf{D}_n^{-1}\mathbf{G}_n^{\dagger}(\mathbf{r}_n - \mathbf{G}_n\mathbf{s}_n^{(i-1)}) \tag{42}$$

where D_n is the matrix that contains the diagonal elements of $G_n^{\dagger}G_n$. The above equation can also be expressed in the form

$$\mathbf{s}_n^{(i)} = -\mathbf{T}_n^{\mathbf{J}}\mathbf{s}_n^{(i-1)}\mathbf{Q}_n^{\mathbf{J}} \cdot \mathbf{z}_n \tag{43}$$

where $\mathrm{T}_n^{\mathrm{J}} = \mathrm{D}_n^{-1} \cdot (\mathrm{L}_n + \mathrm{L}_n^{\dagger})$, $\mathrm{Q}_n^{\mathrm{J}} = \mathrm{D}_n^{-1}$, $\mathrm{z}_n = \mathrm{G}_n^{\dagger}\mathrm{r}_n$. L_n and $\mathrm{L}_n^{\dagger}$ denote the matrices that contain the strictly upper and lower parts of the Hermitian matrix $\mathrm{R}_n = \mathrm{G}_n^{\dagger}\mathrm{G}_n$. It is also important to note that the notation J refers to usage of the Jacobi method we mentioned earlier.

As presented in algorithm 2, the calculation of the latest symbol-vector estimate depends on the previously calculated estimates as well. We are now able to modify **(43)**, by using the Gauss Seidel (which will be annotated by GS) method as seen below:

$$\mathbf{s}_n^{(i)} = -\mathbf{T}_n^{\mathbf{GS}}\mathbf{s}_n^{(i-1)}\mathbf{Q}_n^{\mathbf{GS}} \cdot \mathbf{z}_n \tag{44}$$

where $\mathrm{T}_n^{\mathrm{GS}} = (\mathrm{D}_n + \mathrm{L}_n)^{-1} \cdot \mathrm{L}_n^{\dagger}$ and $\mathrm{Q}_n^{\mathrm{GS}} = (\mathrm{D}_n + \mathrm{L}_n)^{-1}$, while $\mathbf{z}_n$ remaining the same as before. The algorithm, based on the Gauss Sidel method, is the following:

Algorithm 3: MRC delay-time domain operation principle in the form of time domain Gauss-Seidel method.

```
1   Input: $\mathbf{r}, \mathbf{G}$
2   for $n = 0 : N - 1$ do
3       $\mathbf{R}_n = \mathbf{G}_n^{\dagger} \cdot \mathbf{G}_n$
4       $\mathbf{z}_n = \mathbf{G}_n^{\dagger} \cdot \mathbf{r}_n$
5       $\mathbf{L}_n = \text{strictly lower triangular part}\{\mathbf{R}_n\}$
6       $\mathbf{T}_n^{\mathbf{GS}} = (\mathbf{D}_n + \mathbf{L}_n)^{-1} \cdot \mathbf{L}_n^{\dagger}$
7       $\mathbf{Q}_n^{\mathbf{GS}} = (\mathbf{D}_n + \mathbf{L}_n)^{-1}$
8   end
9   $\hat{\mathbf{s}}^{(0)} = \mathbf{P} \cdot (\mathbf{I}_M \otimes \mathbf{F}_N^{\dagger}) \cdot \hat{\mathbf{x}}^{(0)}$
10  for $i = 1 : max\ iterations$ do
11      for $n = 0 : N - 1$ do
12          $\hat{\mathbf{s}}_n^{(i)} = -\mathbf{T}_n^{\mathbf{GS}}\hat{\mathbf{s}}_n^{(i-1)}\mathbf{Q}_n^{\mathbf{GS}} \cdot \mathbf{z}_n$
13      end
14      if ($\| \mathbf{r} - \mathbf{G} \cdot \hat{\mathbf{s}}^{(i)} \| \geq \| \mathbf{r} - \mathbf{G} \cdot \hat{\mathbf{s}}^{(i-1)} \|$) then EXIT
15  end
16  Output: $\hat{\mathbf{x}} = (\mathbf{I}_M \otimes \mathbf{F}_N) \cdot (\mathbf{P} \cdot \hat{\mathbf{s}}^{(i)})$
```

As before, we will begin by analyzing the inputs. Comparing the inputs directly to the previous algorithms, we can clearly see that not only we have less inputs, they also aren't equivalent to ones we had before. Specifically, r is our received signal in the form of a vector and G is the matrix that represents the channel response for multiple propagation paths in the delay-time domain. With that being said, the above algorithm executes the following steps:

1) For $n = 0$ in the range of $N - 1$ calculate the following:
   a) Calculate the Hermitian matrix R_n (for the current n) as we defined after **(43)**. This represents the n-th path's channel response.
   b) Calculate z_n, again as we defined after **(43)**. This represents the n-th path's signal.
   c) Assign the strictly lower triangular part of the n-th path's channel response (or the strictly lower triangular part of the Hermitian matrix R_n) to matrix L_n.
   d) Using the Gauss-Seidel method, compute $\mathrm{T}_n^{\mathrm{GS}}$ by multiplying the inverse of the sum of D_n (which is the matrix containing the diagonal elements of R_n) and L_n, with the strictly upper triangular part of R_n, denoted by $\mathrm{L}_n^{\dagger}$ (as defined earlier).
   e) Using the Gauss-Seidel method, compute $\mathrm{Q}_n^{\mathrm{GS}}$ which is the inverse of the sum D_n and L_n.

2) Compute the initial estimate of the transmitted signal $\hat{\mathrm{s}}^{(0)}$ by multiplying the permutation matrix P with the initial symbol estimate $\hat{\mathrm{x}}^{(0)}$ and the Kronecker product (denoted by $\otimes$) between I_M and $\mathrm{F}_N^{\dagger}$.
3) For $i = 1$ repeat the following until we reach our max iteration limit:
   a) For $n = 0$ in the range of $N - 1$ calculate the following:
      i) Using the Gauss-Seidel method, calculate $\hat{\mathrm{s}}_n^{(i)}$, as defined in **(44)**.
   b) Check whether the residual error is greater or equal to the one from the previous iteration. If so, exit the algorithm.

When the algorithm is finished, we get the same result as the previous algorithms, that being the vector $\hat{\mathrm{x}}$ which contains estimation of the symbols for the transmitted signals. It is important to note, that in algorithm 2 and 3, the final estimation is calculated in a different way, as described in those algorithms respectively.

2.3.4 Low-Complexity Message Passing (MP) Detection

In the paper "Interference Cancellation and Iterative Detection for Orthogonal Time Frequency Space Modulation" written by P. Raviteja, Khoa T. Phan, Yi Hong and Emanuele Viterbo, which was published in October 2018 [56], another algorithm was proposed for the receiver of an OTFS system. The algorithm is a Message Passing Algorithm (MPA). MPA is also an iterative algorithm, same as the MRC algorithms we discussed earlier. This group of algorithms work based on variable nodes and check nodes (or referred to the paper as observation nodes), which are responsible in detecting (and correcting) the transmitted signal, by going back and forth to those nodes, until either the expected result is achieved or until the operation exits [59]. Generally, these algorithms are also relatively low complexity, just like the MRC algorithms but overall tend to require more calculations and iterations, making them slower. The algorithm is described as follows:

Algorithm 4: MP algorithm for OTFS symbol detection

```
1   Input: Received signal $\mathbf{y}$, channel matrix $\mathbf{H}$.
2   Initialization: probability mass function (pmf) $\mathbf{p}_{c,d}^{(0)} = 1/Q$, $c = 1 \ldots, NM$, $d \in \mathcal{J}(c)$, iteration
    count $i = 1$.
3   repeat
4       -  Observation nodes $y[d]$ compute the means $\mu_{d,c}^{(i)}$ and variances $(\sigma_{d,c}^{(i)})^2$ of
           Gaussian random variables $\zeta_{d,c}^{(i)}$ using $\mathbf{p}_{c,d}^{(i-1)}$ and pass them to variable nodes $x[c]$,
           $c \in I(d)$.
5       -  Variable nodes $x[c]$ update $\mathbf{p}_{c,d}^{(i)}$ using $\mu_{d,c}^{(i)}$, $(\sigma_{d,c}^{(i)})^2$ and $\mathbf{p}_{c,d}^{(i-1)}$ and pass them to
           observation nodes $y[d]$, $d \in \mathcal{J}(c)$.
6       -  Compute convergence indicator $\eta^{(i)}$.
7       -  Update the decision on the transmitted symbols $\hat{\mathbf{x}}[c]$, $c = 1 \ldots, NM$ if needed.
8       -  $i = i + 1$.
9   until Stopping criteria;
10  Output: The decision on transmitted symbols $\hat{\mathbf{x}}[c]$
```

Unlike the MRC algorithms, this algorithm is described more abstractly, by describing each step along the way. We will begin by examining the stopping criteria of the MPA.

As stated in the paper, the algorithm stops when at least one of the following criteria is met:

1. $\eta^{(i)} = 1$.

2. $\eta^{(i)} < \eta^{(i^*)} - \varepsilon$, where i^* is the iteration index from $\{1, \dots, (i-1)\}$ for which $\eta^{(i^*)}$ is maximum.
3. Maximum number n_{iter} of iterations is reached.

It is stated that ε should be set to 0.2 to disregard small fluctuations of η. The first condition indicates that all of the received symbols have converged, making it the best-case scenario. The second condition prevents worse decisions, by regarding the previous iterations to its calculation. The third condition, obviously, serves as the worst-case scenario, where the algorithms must compute up to our maximum number of iterations we have set. It is stated, that the above algorithm typically converges within 20 iterations (this remark will be important for later).

The messages passing from variable nodes $x[c]$ to the observation nodes $y[d]$ are computed as follows:

$$\boldsymbol{y}[\boldsymbol{d}] = \boldsymbol{x}[\boldsymbol{c}]\boldsymbol{H}[\boldsymbol{d},\boldsymbol{c}] + \underbrace{\sum_{\boldsymbol{e}\in\boldsymbol{I}(\boldsymbol{d}),\boldsymbol{e}\neq\boldsymbol{c}} \boldsymbol{x}[\boldsymbol{e}]\boldsymbol{H}[\boldsymbol{d},\boldsymbol{e}] + \boldsymbol{z}[\boldsymbol{d}]}_{\boldsymbol{\zeta}_{\boldsymbol{d},\boldsymbol{c}}^{(i)}} \tag{45}$$

where the term, $\zeta_{d,c}^{(i)}$, is a Gaussian random variable which roughly models the mean $\mu_{d,c}^{(i)}$ and the variance $(\sigma_{d,c}^{(i)})^2$ of the interference, which can be computed as follows:

$$\boldsymbol{\mu}_{\boldsymbol{d},\boldsymbol{c}}^{(i)} = \sum_{\boldsymbol{e}\in\boldsymbol{I}(\boldsymbol{d}),\boldsymbol{e}\neq\boldsymbol{c}} \sum_{\boldsymbol{j}=1}^{\boldsymbol{Q}} \boldsymbol{p}_{\boldsymbol{e},\boldsymbol{d}}^{(i-1)}(\boldsymbol{a}_j)\boldsymbol{a}_j\boldsymbol{H}(\boldsymbol{d},\boldsymbol{e}) \tag{46}$$

$$(\boldsymbol{\sigma}_{\boldsymbol{d},\boldsymbol{c}}^{(i)})^2 = \sum_{\boldsymbol{e}\in\boldsymbol{I}(\boldsymbol{d}),\boldsymbol{e}\neq\boldsymbol{c}} \left(\sum_{\boldsymbol{j}=1}^{\boldsymbol{Q}} \boldsymbol{p}_{\boldsymbol{e},\boldsymbol{d}}^{(i-1)}(\boldsymbol{a}_j)\left|\boldsymbol{a}_j\right|^2 |\boldsymbol{H}(\boldsymbol{d},\boldsymbol{e})|^2 - \left| \sum_{\boldsymbol{j}=1}^{\boldsymbol{Q}} \boldsymbol{p}_{\boldsymbol{e},\boldsymbol{d}}^{(i-1)}(\boldsymbol{a}_j)\boldsymbol{a}_j\boldsymbol{H}(\boldsymbol{d},\boldsymbol{e}) \right|^2 \right) + \boldsymbol{\sigma}^2 \tag{47}$$

The update to the decision on the transmitted symbols is calculated by using a symbol-by-symbol maximum a posterior probability (MAP) detection rule, which is the following:

$$\begin{aligned} \hat{\mathbf{x}}[\boldsymbol{c}] &= \mathbf{argmax}\,\mathbf{Pr}\big(\boldsymbol{x}[\boldsymbol{c}] = \boldsymbol{a}_j \big| \mathbf{y}, \mathbf{H}\big) \\ &= \mathbf{argmax}\,\frac{\mathbf{1}}{\boldsymbol{Q}}\mathbf{Pr}\big(\mathbf{y}\big|\boldsymbol{x}[\boldsymbol{c}] = \boldsymbol{a}_j, \mathbf{H}\big) \\ &\approx \mathbf{argmax} \prod_{\boldsymbol{d}\in\boldsymbol{\mathcal{J}}(\boldsymbol{c})} \mathbf{Pr}\,\big(\boldsymbol{y}[\boldsymbol{d}]\big|\boldsymbol{x}[\boldsymbol{c}] = \boldsymbol{a}_j, \mathbf{H}\big) \end{aligned} \tag{48}$$

where $a_j \in \mathbb{A}$, with set $\mathbb{A}$ denoting all the possible symbols and a_j denoting all the transmitted symbols.

The convergence indicator $\eta^{(i)}$ is computed as follows:

$$\boldsymbol{\eta}^{(i)} = \frac{\mathbf{1}}{\boldsymbol{NM}} \sum_{\boldsymbol{c}=\mathbf{1}}^{\boldsymbol{NM}} \mathbb{I}(\mathbf{max}\, \boldsymbol{p}_{\boldsymbol{c}}^{(i)}(\boldsymbol{a}_{\boldsymbol{j}}) \geq \mathbf{1} - \boldsymbol{\gamma}) \tag{49}$$

for some small $\gamma > 0$, $a_j \in \mathbb{A}$ and where

$$\boldsymbol{p}_{\boldsymbol{c}}^{(i)}(\boldsymbol{a}_{\boldsymbol{j}}) = \prod_{\boldsymbol{e}\in\mathcal{J}(\boldsymbol{c})} \frac{\boldsymbol{\xi}^{(i)}(\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{j})}{\sum_{\boldsymbol{k}=\mathbf{1}}^{\boldsymbol{Q}} \boldsymbol{\xi}^{(i)}(\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{k})} \tag{50}$$

where $\boldsymbol{\xi}^{(i)}(\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{k}) = \exp\left(\frac{-\left|\boldsymbol{y}[e] - \mu_{e,c}^{(i)} - H_{e,c}a_k\right|^2}{\left(\sigma_{e,c}^{(i)}\right)^2}\right)$ and $\mathbb{I}(\cdot)$ is a Boolean function, resulting in 1 if the argument is true and 0 otherwise.

The pmf vector $\mathrm{p}_{c,d}^{(i)}$ can be updated as follows:

$$\mathbf{p}_{\boldsymbol{c},\boldsymbol{d}}^{(i)}(\boldsymbol{a}_{\boldsymbol{j}}) = \boldsymbol{\Delta} \cdot \tilde{\mathbf{p}}_{\boldsymbol{c},\boldsymbol{d}}^{(i)}(\boldsymbol{a}_{\boldsymbol{j}}) + (\mathbf{1} + \boldsymbol{\Delta}) \cdot \mathbf{p}_{\boldsymbol{c},\boldsymbol{d}}^{(\boldsymbol{i}-\mathbf{1})}(\boldsymbol{a}_{\boldsymbol{j}}) \tag{51}$$

where $a_j \in \mathbb{A}$ and $\Delta \in (0,1]$ is the damping factor, which is used to improve the performance by controlling the convergence speed and

$$\begin{aligned} \tilde{\mathbf{p}}_{\boldsymbol{c},\boldsymbol{d}}^{(i)}(\boldsymbol{a}_{\boldsymbol{j}}) &\propto \prod_{\boldsymbol{e}\in\mathcal{J}(\boldsymbol{c}),\boldsymbol{e}\neq\boldsymbol{d}} \mathbf{Pr}(\boldsymbol{y}[\boldsymbol{e}] | \boldsymbol{x}[\boldsymbol{c}] = \boldsymbol{a}_{\boldsymbol{j}}, \mathbf{H}) \\ &= \prod_{\boldsymbol{e}\in\mathcal{J}(\boldsymbol{c}),\boldsymbol{e}\neq\boldsymbol{d}} \frac{\boldsymbol{\xi}^{(i)}(\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{j})}{\sum_{\boldsymbol{k}=\mathbf{1}}^{\boldsymbol{Q}} \boldsymbol{\xi}^{(i)}(\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{k})} \end{aligned} \tag{52}$$

with $\boldsymbol{\xi}^{(i)}(\boldsymbol{e}, \boldsymbol{c}, \boldsymbol{k})$ equal as before.

2.3.5 Other Detection Algorithms in the Literature

In the literature for delay-Doppler-based communications, a few more algorithms have been proposed for the receivers. One of the algorithms is a single tap time-frequency equalizer, which while being low complexity, as we will see in the simulation results it performs very well compared to the other algorithms we will test, and a block-wise time-domain linear minimum mean square error (LMMSE), which while it is performing better than the time-frequency equalizer, it has both higher complexity and an overall higher bit-error rate (BER) than the other algorithms that we presented and analyzed previously. With the increase of specification requirements as the generations evolve, the two aforementioned algorithms do not serve as good candidates for systems, both due to low performance but also higher complexity than other algorithms implemented on receivers, both of which are necessary for fast speeds and reliable connections in B5G systems. With that being said, while we still take a look at the implementation of those algorithms in the simulation software, we will only use them to compare them against each of the algorithms presented previously [60].

Chapter 3 Implementation in MATLAB®

3.1 Laying the Foundation

3.1.1 The Simulation Software (MATLAB®)

MATLAB® is a programming platform that has been specifically designed to be used by engineers and scientists to design, model, analyze and test various types of applications. MATLAB® uses the MATLAB scripting language, specifically a matrix-based scripting language that allows for most natural expressions of computational mathematics [61]. MATLAB® offers a variety of packages to assist in the development of various applications, such as the telecommunications toolbox that provides premade functions such as qammod and qamdemod which, as the name implies, modulates a vector of information bits to QAM symbols, based on the constellation size we give it. It also has various of assisting functions, such as the implementation of the FFT which we have mentioned in the previous chapter, premade tools to add Gaussian white noise to a signal or to randomly create information bits for a signal, and tools to visually represent the data that are yielded from executing the scripts. Overall, the tools that MATLAB® provides are ideal for various telecommunication system implementations and we will be using it to run the simulations of the OTFS system, based on the receiver algorithms we discussed previously.

3.1.2 Synthetic (Ideal) Channel Representation

In the simulations, we will be using two different representations for the channels. The first representation is a synthetic (or an ideal) channel, that will be used exclusively for testing, specifically for the comparison of different QAM constellations between the MRC (algorithm 2) and MP (algorithm 4) algorithms. In MATLAB, we can create the channel as follows:

```
taps = 4;
delay_taps = [0 1 2 3];
Doppler_taps = [0 1 2 3];
pow_prof = (1/taps) * (ones(1,taps));
chan_coef = sqrt(pow_prof.*(sqrt(1/2)*(randn(1,taps)+1i*randn(1,taps)));
```

Each of the above variables represent the following:

- **taps**: This is the total number of paths (or taps) that appear within the channel. For testing, it will be set to 4.
- **delay_taps**: This serves as the delay profile of the channel. Each of the variable's elements represents the delays of each tap. For testing, the elements are set to 0, 1, 2 and 3 respectively.
- **Doppler_taps**: This serves as the Doppler profile of the channel. Each of the variable's elements represents the Doppler shift of each tap. For testing, the elements are also set to 0, 1, 2 and 3 respectively.
- **pow_prof**: This represents the power delay profile of each tap. Given that fact that we are creating an ideal channel, the taps have uniform power, thus calculating the power profile once.
- **chan_coef**: This represents the channel coefficients for each tap, which are calculated based on the power profile of each tap. Each of the coefficients is the magnitude and phase of the tap's contribution to the channel. The function randn is used to generate a complex Gaussian random number with zero mean and unit variance.

The variables will be used in the calculations of the channel output, simulating the transmitted signal passing through a channel with the above parameters. We will discuss that calculation later on.

3.1.3 3GPP LTE Propagation Channel Models

It is clear, that simulations with an ideal channel do not represent accurately the performance of a system that will be deployed in a highly complex and varying environment. Due to this, 3GPP created three different propagation channel models, to test and simulate systems in multipath fading environments. The models are the following:

- Extended Pedestrian A (EPA) model.
- Extended Vehicular A (EVA) model.
- Extended Typical Urban (ETU) model.

Due to the nature of OTFS, along with the use-case that it was developed for, it is obvious that we will not be using the low-propagation EPA model to for the simulations. The medium-propagation (EVA) and high-propagation (ETU) serves our purposes better to simulate a variety of different scenarios and environments, closer to the real world. In the following tables we can see the delay profiles of each of the aforementioned models (obviously excluding EPA).

Excess tap delay (ns)	Relative power (dB)
0	0.0
30	-1.5
150	-1.4
310	-3.6
370	-0.6
710	-9.1
1090	-7.0
1730	-12.0
2510	-16.9

Table 1: EVA Channel Model Delay Profile

Excess tap delay (ns)	Relative power (dB)
0	-1.0
50	-1.0
120	-1.0
200	0.0
230	0.0
500	0.0
1600	-3.0
2300	-5.0
5000	-7.0

Table 2: ETU Channel Model Delay Profile

Based on the above tables, the channels can now be implemented in MATLAB as follows:

```
one_delay_tap = 1/(M*delta_f);
one_doppler_tap = 1/(N*T);

delays = [0 30 150 310 370 710 1090 1730 2510]*10^(-9); %EVA model
%delays = [0 50 120 200 230 500 1600 2300 5000]*10^(-9); %ETU model

taps = length(delays); % number of delay taps
delay_taps = round(delays/one_delay_tap); %assuming no fraction for the delay

pdp = [0 -1.5 -1.4 -3.6 -0.6 -9.1 -7.0 -12.0 -16.9]; % EVA power delay profile
%pdp = [-1 -1 -1 0 0 0 -3 -5 -7]; % ETU power delay profile

pow_prof = 10.^(pdp/10);
pow_prof = pow_prof/sum(pow_prof); %normalization of power delay profile

chan_coef = sqrt(pow_prof).*(sqrt(1/2) * (randn(1,taps)+1i*randn(1,taps)));
%channel coef. for each path

max_UE_speed = max_speed*(1000/3600);
Doppler_vel = (max_UE_speed*car_fre)/(299792458);
max_Doppler_tap = Doppler_vel/one_doppler_tap;
Doppler_taps = (max_Doppler_tap*cos(2*pi*rand(1,taps))); %Doppler taps using
Jake's spectrum
```

Clearly, the number of taps increases to 9 (since the delay profiles have 9 different elements). Let us analyze the variables present in the above snippet of code:

- **one_delay_tap**: This variable represents the delay of one subcarrier in the delay domain. In the calculation, variable M is our constellation size and variable delta_f is the subcarrier spacing.
- **one_doppler_tap**: This variable represents the Doppler shift of one subcarrier in the Doppler domain. In the calculation, variable N is the number of symbols and variable T is the one-time symbol duration in an OTFS frame.
- **delay_taps**: This variable, like in the synthetic channel, represents the delay profile for the channel. Unlike in the synthetic channel, this variable is not static, but is calculated by using the 3GPP excess tap delay (which corresponds to the model we are using, in the above case EVA) and the one_delay_tap variable that we calculated earlier, assuming no fraction for the delay by rounding the result.
- **pow_prof**: This variable represents the power delay profile of each tap of the corresponding channel model, converted to linear scales from the dB values, followed by a normalization.
- **chan_coef**: This variable represents the channel coefficient of each tap, which is calculated the same way as in the synthetic channel, with the only difference being the fact that the power delay profile is not uniform.
- **max_UE_speed**: This variable represents the maximum speed of user equipment (UE) in km/h.
- **Doppler_vel**: This variable represents the maximum Doppler velocity, which is given by the multiplication of the maximum UE speed with carrier frequency, divided by the speed of light.
- **max_Doppler_tap**: This variable represents the maximum number of Doppler shifts in one tap.
- **Doppler_taps**: This variable, like in the synthetic channel, represents the Doppler profile of the channel. Unlike the synthetic channel where the variable is static, in this implementation its generated by using Jake’s spectrum.

Besides the fact that the propagation channel (in result and the taps) are more realistic, this channel implementation also allows us to change the UE speed to accommodate for different scenarios.

3.2 OTFS Sample Code

3.2.1 The OTFS_Sample_Code Package

The following OTFS MATLAB implementation was presented in a tutorial at ICC2019 in Shanghai, May 24th 2019, by Y. Hong, E. Viterbo and A. Chockalingam (https://ecse.monash.edu/staff/eviterbo/). The code has been written by R. Patchava, Y. Hong and E. Viterbo (although there may have been some correction due to MATLAB version differences) and is freely distributed for educational and research purposes. The whole code package can also be found at the end of this document.

3.2.2 OTFS Modulation Function

We will begin by looking at the implementation of the OTFS modulation that we discussed in Chapter 2, that being the ISFFT followed by the Heisenberg transform. In MATLAB, this is implemented as follows:

```
X = fft(ifft(x).').'/sqrt(M/N); %%%ISFFT
s_mat = ifft(X.')*sqrt(M); % Heisenberg transform
s = s_mat(:);
```

We have already mentioned that M stands for the number of subcarriers and N for the number of symbols in time. The x is a variable that has the mapped signal, which (as is described in the literature) is done by running the qammod built-in function with the specified constellation size M_mod. It is important to note that the information bits that are to be modulated, are randomly decided each time a signal is to be transmitted. With that being said, the snipper modulates an input signal x as follows:

- The ISFFT is calculated, by firstly calculated the IFFT of variable x. That is passed into the FFT and the final result is yielded by dividing that with the square root of the number of subcarriers over our number of symbols in time. We should note that both the IFFT and the division between the two terms is element-wise, which is denoted with the symbol "." in MATLAB. The result is stored in variable X.
- The variable X is then passed into the IFFT (again, element-wise) and then multiplied by the square root the number of subcarriers. The result is stored in variable s_mat which is then transposed, giving us the OTFS modulated signal s.

It is important to note that the SFFT can be described as a double Fourier Transform, as described above. The resulting signal s is then passed (in this implementation) in a synthetic channel which we described previously.

3.2.3 Synthetic Channel Output Function

By passing the OTFS modulated signal into the channel, we introduce it to Additive White Gaussian Noise (AWGN), which results in the signal, before demodulation, to be altered (which will eventually lead to errors even though this is an ideal channel). The way that this is implemented in MATLAB can be seen below.

```
L = max(delay_taps);
s = [s(N*M-L+1:N*M);s]; %add one cp
s_chan = 0;
for itao = 1:taps
    s_chan = s_chan+chan_coef(itao)*circshift([s.*exp(1j*2*pi/M ...
              *(-L:-L+length(s)-1) ...
       *Doppler_taps(itao)/N).';zeros(delay_taps(end),1)],delay_taps(itao));
end
noise = sqrt(sigma_2/2)*(randn(size(s_chan)) + 1i*randn(size(s_chan)));
r = s_chan + noise;
r = r(L+1:L+(N*M)); %discard cp
```

In the above code we can also see the addition of a cyclic prefix (CP), which is to used along with the MP detector algorithm (algorithm 4) that we presented earlier. The code snippet executes the following:

- Variable L is initialized with the maximum value of all the delay taps. This represent the “last L” samples and will be used for the addition of the CP.
- The modulated signal s gets a CP added to it.
- Variable s_chan which, at the end, will represent the signal after it passed through the channel, is initialized with the value 0.
- For each of the taps, the variable s_chan for the current tap is calculated as follows:
    - The current value of s_chan is added to the channel coefficient of the current tap multiplied by the result of the circshift function. The function receives two parameters, the first being the signal (after the addition of the CP) multiplied by the complex channel coefficient with L numbers of zeros padded, and the second one being the current iteration’s delay_tap.
- The AWGN is calculated as shown above, with sigma_2 representing the variance of AWGN or simply the noise’s power. The result is added to signal s and stored in the variable r.
- The final result is given by removing the CP from signal r. The resulting signal is our transmitted signal after it passed through the channel.

After we receive the signal from the channel, we can then use the OTFS demodulation before passing that result into a detector (in our case that is an MPA detector).

3.2.4 OTFS Demodulation Function

Once the signal comes out of the channel, it is then passed into the demodulation function, which effectively reverses the effect of the modulation function we discussed previously. In MATLAB, this is implemented as follows:

```
r_mat = reshape(r,M,N);
Y = fft(r_mat)/sqrt(M); % Wigner transform
Y = Y.';
y = ifft(fft(Y).').'/sqrt(N/M); % SFFT
```

If we pay close attention, disregarding the reshape function, we can see that the operations are the reverse of the ones in the modulator function. Specifically, the following operations take place:

- The signal r which is the signal that comes out of the channel, gets reshaped into an MxN matrix and stored in variable r_mat.
- The variable Y is calculated by passing r_mat into the FFT (element-wise) and then divided by the square root of the number of subcarriers. The variable Y is then transposed. In comparison, the equivalent line of code used and IFFT and multiplication before transposing the result.

- The SFFT is calculated, by firstly calculated the FFT of variable Y. That is passed into the IFFT and the final result is yielded by dividing that with the square root of the number of subcarriers over the number of symbols in time. The operations, in comparison, are written in reverse order with FFT being first and IFFT being second (the opposite than before).

This results in variable y holding (mostly) the QAM modulated information bits. To reverse the QAM modulation, we need to pass the received symbols into a detection algorithm. In this implementation, as we have mentioned, we will be using algorithm 4 (MPA detection).

3.2.5 OTFS MP Detector Implementation

MPA will be one of the main algorithms that we will use later for the simulations, making it important to understand how it is implemented in MATLAB. First of all, the inputs to the function that will implement the MPA are the following.

- **N**: Number of symbols in time.
- **M**: Number of subcarriers.
- **M_mod**: Constellation size.
- **taps**: Number of taps.
- **delay_taps**: Delay profile of the channel.
- **Doppler_taps**: Doppler profile of the channel.
- **chan_coef**: Channel's coefficient.
- **sigma_2**: AWGN variance or simply noise power.
- **y**: The QAM modulated information bits after the signal has passed through the channel and has been demodulated.

We will begin by showing the initialization of a few variables that will be used throughout the implementation of the MPA. The implementation can be seen below:

```
yv = reshape(y,N*M,1);
n_ite = 200;
delta_fra = 0.6;
alphabet = qammod(0:M_mod-1,M_mod,'gray');

mean_int = zeros(N*M,taps);
var_int = zeros(N*M,taps);
p_map = ones(N*M,taps,M_mod)*(1/M_mod);

conv_rate_prev = -0.1;
```

When the function is called, the following variables are created as follows:

- **yv**: This variable represents the variable y (which is a matrix with N rows and M columns) as a vector, by using the reshape function, turning the original matrix into a vector with N*M elements.
- **n_ite**: This variable represents the maximum iterations that the algorithm can compute before exiting.
- **delta_fra**: This variable represents a fractional weight that will be used later to calculate the probability distribution. It is initialized with the value 0.6.
- **alphabet**: This variable represents all the possible symbols that the initial signal might have carried. It is calculated using the function qammod and the constellation size M_mod that we have passed as a function parameter.

- **mean_int and var_int**: These variables represent the mean and the variance of the interference respectively. They are both initialized with zeros (they are N*MxTaps matrixes). They also correspond to **(46)** and **(47)** respectively.
- **p_map**: This variable represents the probability of each possible symbol and is initialized by giving each symbol equal probability.
- **conv_rate_prev**: This variable represents the previous convergence rate and it is initialized to -0.1. This corresponds to **(49)**.

With the above variables being initialized, the main body of the algorithm begins to execute, which is a loop that iterates up to 200 times, the following parts of code. We will begin by presenting the implementation for the mean and variance updates. The code snippet is the following:

```
%% Update mean and var
    for ele1=1:1:M
        for ele2=1:1:N
            mean_int_hat = zeros(taps,1);
            var_int_hat = zeros(taps,1);
            for tap_no=1:taps
                m = ele1-1-delay_taps(tap_no)+1;
                add_term = exp(1i*2*(pi/M)*(m-1)*(Doppler_taps(tap_no)/N));
                add_term1 = 1;
                if ele1-1<delay_taps(tap_no)
                    n = mod(ele2-1-Doppler_taps(tap_no),N) + 1;
                    add_term1 = exp(-1i*2*pi*((n-1)/N));
                end
                new_chan = add_term * (add_term1) * chan_coef(tap_no);

                for i2=1:1:M_mod
                    mean_int_hat(tap_no) = mean_int_hat(tap_no) + ...
p_map(N*(ele1-1)+ele2,tap_no,i2) * alphabet(i2);
                    var_int_hat(tap_no) = var_int_hat(tap_no) + ...
p_map(N*(ele1-1)+ele2,tap_no,i2) * abs(alphabet(i2))^2;
                end
                mean_int_hat(tap_no) = mean_int_hat(tap_no) * new_chan;
                var_int_hat(tap_no) = var_int_hat(tap_no) * abs(new_chan)^2;
                var_int_hat(tap_no) = var_int_hat(tap_no) - ...
abs(mean_int_hat(tap_no))^2;
            end

            mean_int_sum = sum(mean_int_hat);
            var_int_sum = sum(var_int_hat)+(sigma_2);

            for tap_no=1:taps
                mean_int(N*(ele1-1)+ele2,tap_no) = mean_int_sum - ...
mean_int_hat(tap_no);
                var_int(N*(ele1-1)+ele2,tap_no) = var_int_sum - ...
var_int_hat(tap_no);
            end

        end
    end
```

The above code snippet begins by iterating from 1 up to the number of subcarriers M. Inside that iteration, the code iterates from 1 up to the number of symbols N. Inside the second loop, the following calculations happen:

- Two assisting variables, mean_int_hat and var_int_hat are initialized with zeros. Then, an iteration through all the taps begins.
    - Initially, the channel coefficient is updated, based whether the outmost iteration count (which is denoted by ele1) is greater than the current tap value of the delay

profile. Following this, another iteration begins that iterates up to the constellation size.

        - Inside that iteration, the two assisting variables mean_int_hat and var_int_hat are updated based on the probability of each symbol (variable p_map), the current outmost iteration number (ele1), the second outmost iteration number (ele2), the current tap that the algorithm is iterating (tap_no), the current number of the constellation size the algorithm is iteration (i2) and the previous value of the variables.
    - They two variables are updated further, depending on the new channel coefficient that was calculated for the current tap.
- The sum of the elements of the two variables is calculated and stored into two additional assisting variables, mean_int_sum and var_int_sum respectively. The algorithm again iterates through the taps.
    - The mean and the variance of the interference are calculated based on the two assisting variables mean_int_sum and var_int_sum, as well as the element that corresponds to the current tap the algorithm is iterating through of the assisting variables mean_int_hat and var_int_hat.

After the mean and the variance of the interference is updated, meaning that the algorithm has iterated through both the number of symbols in time and the number of subcarriers, the algorithm proceeds by initializing three additional variables as follows:

```
sum_prob_comp = zeros(N*M,M_mod);
dum_eff_ele1 = zeros(taps,1);
dum_eff_ele2 = zeros(taps,1);
```

The above variables are initialized with zeros and are the following:

- **sum_prob_comp**: This variable will be storing the sum of the probabilities that are computed. It will later be used for the update of the convergence rate.
- **dum_eff_ele1 and dum_eff_ele2**: These are two assisting variables that will be holding intermediate values of each of the iterations as we will see later.

Following the above initialization, the algorithm will iterate again through the number of subcarriers (which will again be referred as the outmost iteration or loop), inside of which it will be iterating through the number of symbols in time (which will, again, also be referred to as the second outmost iteration or loop). Inside the second outmost iteration, the computations shown in the code snippet below take place.

```
            dum_sum_prob = zeros(M_mod,1);
            log_te_var = zeros(taps,M_mod);
            for tap_no=1:taps

                if ele1+delay_taps(tap_no)<=M
                    eff_ele1 = ele1 + delay_taps(tap_no);
                    add_term = exp(1i*2*(pi/M)*(ele1- ...
1)*(Doppler_taps(tap_no)/N));
                    int_flag = 0;
                else
                    eff_ele1 = ele1 + delay_taps(tap_no)- M;
                    add_term = exp(1i*2*(pi/M)*(ele1-1- ...
M)*(Doppler_taps(tap_no)/N));
                    int_flag = 1;
                end
                add_term1 = 1;
                if int_flag==1
                    add_term1 = exp(-1i*2*pi*((ele2-1)/N));
                end
                eff_ele2 = mod(ele2-1+Doppler_taps(tap_no),N) + 1;
                new_chan = add_term * add_term1 * chan_coef(tap_no);

                dum_eff_ele1(tap_no) = eff_ele1;
                dum_eff_ele2(tap_no) = eff_ele2;
                for i2=1:1:M_mod
                    dum_sum_prob(i2) = abs(yv(N*(eff_ele1-1)+eff_ele2)- ...
mean_int(N*(eff_ele1-1)+eff_ele2,tap_no) - new_chan * alphabet(i2))^2;
                    dum_sum_prob(i2)= -(dum_sum_prob(i2)/var_int(N*(eff_ele1- ...
1)+eff_ele2,tap_no));
                end
                dum_sum = dum_sum_prob - max(dum_sum_prob);
                dum1 = sum(exp(dum_sum));
                log_te_var(tap_no,:) = dum_sum - log(dum1);
            end
            for i2=1:1:M_mod
                ln_qi(i2) = sum(log_te_var(:,i2));
            end
            dum_sum = exp(ln_qi - max(ln_qi));
            dum1 = sum(dum_sum);
            sum_prob_comp(N*(ele1-1)+ele2,:) = dum_sum/dum1;
            for tap_no=1:1:taps
                eff_ele1 = dum_eff_ele1(tap_no);
                eff_ele2 = dum_eff_ele2(tap_no);

                dum_sum = log_te_var(tap_no,:);
                ln_qi_loc = ln_qi - dum_sum;
                dum_sum = exp(ln_qi_loc - max(ln_qi_loc));
                dum1 = sum(dum_sum);
                p_map(N*(eff_ele1-1)+eff_ele2,tap_no,:) = ...
(dum_sum/dum1)*delta_fra + (1-delta_fra)*reshape(p_map(N*(eff_ele1- ...
1)+eff_ele2,tap_no,:),1,M_mod);
            end
```

The above snippet begins with the initialization of two additional assisting variables, dum_sum_prob (which later one will store the sum of the probabilities of the symbol candidates) and log_te_var (which will hold the logarithm of the previous variable), with zeros. After that, the following computations take place:

- The algorithm iterates the number of taps and computes:
    - Depending whether the sum of ele1 and the current iteration's delay profile element is less-equal or greater to the number of the number of subcarriers, the effective index is calculated, variable add_term is set and a (Boolean) flag depending on the criteria is set.
    - If the flag is set to 1, then the variable add_term1 is set to a specific value, otherwise its set to the number 1.
    - eff_ele2 is calculated, along with the new channel coefficient which is calculated by multiplied the variables add_term, add_term1 and the channel coefficient of the current tap.
    - The assisting variables dum_eff_ele1 and dum_eff_ele2 are set to the eff_ele1 and eff_ele2 variable values respectively and an iteration through the constellation size begins, which calculates the sum of the probabilities for the current element (the element that corresponds to the iteration number i2).
    - The probability of the candidate symbols is calculated, along with the calculation of its logarithm.
- After the completion of the iteration through the taps, another iteration through the constellation size begins, calculating ln_qi's elements corresponding to the iteration count, by summing the logarithms stored in variable log_te_var.

- The probability for the symbol candidates is then updated, before iterating through the taps again.
    - The effective indices are updated and finally the probability of each symbol is updated and stored in the p_map variable.

The above, as we stated repeat for each subcarrier and symbol in time. After those iterations finish, the convergence rate is updated, followed by a check for the algorithm's stopping criteria we discussed in algorithm 4. The above is implemented in MATLAB as follows:

```
conv_rate = sum(max(sum_prob_comp,[],2)>0.99)/(N*M);
if conv_rate==1
   sum_prob_fin = sum_prob_comp;
   break;
elseif conv_rate > conv_rate_prev
   conv_rate_prev = conv_rate;
   sum_prob_fin = sum_prob_comp;
elseif (conv_rate < conv_rate_prev - 0.2) && conv_rate_prev > 0.95
   break;
end
```

If the first or third criteria matches in any of the iterations (this refers to the iteration that all of the above code takes place in, which is an iteration until a specified limit is met) the algorithm stops and produces the final symbol estimates of the received signal. Otherwise, the variable that stores the previous convergence rate is either updated or remains the same before repeating the whole process that we described again, until either condition one or three becomes true or the iteration limit is reached. The code below shows how the final symbol estimates are calculated.

```
x_est = zeros(N,M);
for ele1=1:1:M
    for ele2=1:1:N
        [~,pos] = max(sum_prob_fin(N*(ele1-1)+ele2,:));
        x_est(ele2,ele1) = alphabet(pos);
    end
end
```

Its important to note, that the above snippet exist outside of the main loop of the algorithm and its only ever ran once.

3.3 Zero Padding (ZP) OTFS Algorithm Implementations

3.3.1 The ZP_OTFS_MRC_Detection Package

The following OTFS MATLAB implementation was published by IEEE on the 14th of October 2020 and is written by Y. Hong, E. Viterbo and T. Thaj (https://ecse.monash.edu/staff/eviterbo/). The code has been written by R. Patchava, Y. Hong and E. Viterbo (although there may have been some correction due to MATLAB version differences) and is freely distributed for educational and research purposes. The whole code package can also be found at the end of this document. This particular package has the implementation of all of the algorithms that have been presented in this paper, including the single tap time-frequency equalizer and the block-wise time-domain LMMSE algorithms that we only briefly mentioned. From the above, we will be focusing on the implementation of algorithm 2 (low-complexity MRC), while briefly explaining some key points of the other implementations.

3.3.2 OTFS Low-complexity MRC Detection Implementation

The key difference of the two code packages is the different receiver that is implemented in each one. The remaining points, such as the channel representation (both ideal and 3GPP models), the modulation and demodulation of OTFS, as well as the channel output remain the same, thus we will not be discussing them again. With that being said we will begin by discussing the inputs to the low-complexity MRC implementation.

- **N**: Symbols in time.
- **M**: Number of subcarriers.
- **M_mod**: Constellation size.
- **M_data and data_grid**: These variables represent the data positions of OTFS delay-Doppler data symbols.
- **no (or sigma_2)**: Variance of the AWGN.
- **r**: Channel output.
- **H_tf**: Channel in time-frequency domain.
- **nu_ml_tilda**: Channel in delay-time domain.
- **L_set**: A set with all of the unique delay profile values.
- **omega**: Damping parameter. This improves performance in the cost of higher iteration count. Should be reduced (increasing performance) in high order modulations such as 64QAM.
- **decision**: Flag that indicates the use of hard decision or soft decision.
- **init_estimate**: Flag that indicated whether a single tap initial estimate will be used or the estimate will begin at 0. The single tap estimate should not be used in higher order modulations such as 64QAM since it may produce inaccurate results.
- **n_ite_MRC**: The total number of iterations the MRC algorithm will perform.

The function begins by initializing and preparing variables that will be used later in the implementation. These preparations can be seen below:

```
%% Normalized DFT matrix
Fn=dftmtx(N);  % Generate the DFT matrix
Fn=Fn./norm(Fn);  % normalize the DFT matrix
%% Initial assignments
%Number of symbols per frame
N_syms_perfram=sum(sum((data_grid>0)));
%Arranging the delay-Doppler grid symbols into an array
data_array=reshape(data_grid,1,N*M);
%finding position of data symbols in the array
[~,data_index]=find(data_array>0);
%number of bits per QAM symbol
M_bits=log2(M_mod);
%number of bits per frame
N_bits_perfram = N_syms_perfram*M_bits;
%received delay-time samples
Y_tilda=reshape(r,M,N);
M_prime=M_data;
L_set=L_set+1; % Since matlab cannot work with zero indices
```

The above code snippet executes the following:

- A discrete Fourier transform (DFT) matrix is generated and then normalized.
- The number of symbols per frame is calculated.
- The variable data_grid is reshaped into a 1D vector.
- The position of the data symbols within the reshaped data_grid is found.
- The number of bits each QAM symbol has is calculated.
- With the knowledge of the bit count for each QAM symbol, the number of bits in a frame is calculated.

- The received signal is reshaped into an MxN matrix.
- Variable M_prime is initialized with the contents of variable M_data.
- L_set is incremented by 1 to avoid the number 0 (MATLAB indexing begins at 1, opposite to other programming languages where it begins at 0).

The function then proceeds to calculating the initial estimate depending on the flag variable init_estimate. The code snippet that computes the initial estimate can be seen below.

```
%% initial estimate using single tap TF equalizer
if(init_estimate==1)
    Y_tf=fft(Y_tilda).'; % delay-time to frequency-time domain
% equation (63) in [R1]
    X_tf=conj(H_tf).*Y_tf./(H_tf.*conj(H_tf)+no); % single tap equalizer
% equation (64) in [R1]
    X_est = ifft(X_tf.')*Fn; % SFFT
% equation (65) in [R1]
    X_est=qammod(qamdemod(X_est,M_mod,'gray'),M_mod,'gray');
    X_est=X_est.*data_grid;
    X_tilda_est=X_est*Fn';
else
    X_est=zeros(M,N);
    X_tilda_est=X_est*Fn';
end
x_m=X_est.';
x_m_tilda=X_tilda_est.';
```

The above code snippet, if the flag is set to 1, computes the initial estimation as follows:

- The reshaped received signal is taken from the delay-time domain to the frequency-time domain with the use of the fft function.
- The single tap equalizer is created and then its transformed using the SFFT.
- The estimation of the symbols is computed.

If the flag is not set to 1, then the estimation is simply set to 0. It is important to note, that the equations and the reference tag refer to this paper [57].

With the initial estimation computed, the algorithm will then begin its initial computation, as shown in the code snippet below.

```
%% initial computation
d_m_tilda=zeros(N,M);
y_m_tilda=reshape(r,M,N).';
delta_y_m_tilda=y_m_tilda;
for m=1:M_prime
    for l=L_set
        d_m_tilda(:,m)=d_m_tilda(:,m)+abs(nu_ml_tilda(:,m+(l-1),l).^2);
% equation (59) in [R1]
    end
end
for m=1:M
    for l=L_set
        if(m>=l)
            delta_y_m_tilda(:,m)=delta_y_m_tilda(:,m)- ...
nu_ml_tilda(:,m,l).*x_m_tilda(:,m-(l-1));
% Line 3 of Algorithm 2 in [R1]
        end
    end
end
x_m_tilda_old=x_m_tilda;
c_m_tilda=x_m_tilda;
```

The computation begins with the creation of three variables. The variable d_m_tilda is initialized with zeros, the variable y_m_tilda is the received signal that is reshaped into an MxN matrix and variable delta_y_m_tilda is initialized with variable y_m_tilda. Afterwards, a double iteration begins through all the data positions and the unique delay profiles. Within the double iteration, the variable d_m_tilda is updated. After the double iteration is completed, another double iteration begins through the number of subcarriers and the unique delay profiles. Within that double iteration, every time that variable m (which counts the iteration through the subcarriers) is greater or equal to 1, the variable delta_y_m_tilda is updated. Finally, two assisting variables are created that hold the old value of x_m_tilda, named x_m_tilda_old and c_m_tilda.

With the initial computation completed, the algorithm proceeds to the iterative computation, up until the specified iteration count that we have set. The iterative computation can be seen in the following code snippet.

```
    delta_g_m_tilda=zeros(N,M);
    for m=1:M_prime
        for l=L_set
            delta_g_m_tilda(:,m)=delta_g_m_tilda(:,m)+conj(nu_ml_tilda(:,m+(l-...
1),l)).*delta_y_m_tilda(:,m+(l-1));
% Line 8 of Algorithm 2 in [R1]
        end
        c_m_tilda(:,m)=x_m_tilda_old(:,m)+delta_g_m_tilda(:,m)./d_m_tilda(:,m);
% Line 9 of Algorithm 2 in [R1]
        if(decision==1)

x_m(:,m)=qammod(qamdemod(Fn*(c_m_tilda(:,m)),M_mod,'gray'),M_mod,'gray');
% Line 10 of Algorithm 2 in [R1]
            x_m_tilda(:,m)=(1-omega)*c_m_tilda(:,m)+omega*Fn'*x_m(:,m);
        else
            x_m_tilda(:,m)=c_m_tilda(:,m);
        end
        for l=L_set
% Line 11 of Algorithm 2 in [R1]
            delta_y_m_tilda(:,m+(l-1))=delta_y_m_tilda(:,m+(l-1))-...
nu_ml_tilda(:,m+(l-1),l).*(x_m_tilda(:,m)-x_m_tilda_old(:,m));
% Line 12 of Algorithm 2 in [R1]
        end
% Line 13 of Algorithm 2 in [R1]
        x_m_tilda_old(:,m)=x_m_tilda(:,m);
    end

    %% convergence criteria
    error(ite)=norm(delta_y_m_tilda);
    if(ite>1)
        if(error(ite)>=error(ite-1))
% Line 15 of Algorithm 2 in [R1]
            break;
        end
    end
```

All of the above operations correspond to lines in algorithm 2 that we have presented in this document, meaning that is not necessary to go through the MATLAB equivalent since it essentially implements the mathematical equations that we have discussed and analyzed. After the iterative computation is completed, whether that is by meeting the convergence criteria or reaching the max iteration count, the symbols are detected the same way as in the MPA implementation. It is important to note, that the flag decision, decides whether a soft or hard estimation will take place. If the flag is set to 1 then a hard estimation will be computed, otherwise a soft estimation will be computed instead.

3.3.3 Other Detection Algorithm Implementations

In the package, algorithm 1 and algorithm 3 are also available for use, which show minor differences between themselves, those mostly being different implementations of the iterative computation part. There is also another variant of algorithm 3 that has reduced complexity while also being based in the Gauss-Seidel method of iterative computation. The other MRC algorithms are implemented with the use of more iteration loops, making them less efficient with large numbers (large numbers as in the symbol in time count, total subcarriers, delay and Doppler taps etc.). Implemented in the package are also the two algorithms that we did not analyze. We will primarily use these algorithms to compare them against all the other available algorithms in the literature. While these two algorithms are not lengthy (in the sense of lines of code), the LMMSE does have large iterations and the single-tap equalizer is not as good as other algorithms that we have presented (BER-wise).

Chapter 4 OTFS Simulations

4.1 Simulation Parameters

4.1.1 Channel Parameter Decision

The channel models that will be used, as we have mentioned, are:

- Synthetic/Ideal channel to set a baseline comparison between the detectors that we will be testing.
- EVA model to test MRC in mid-propagation channel conditions, such as freeways or railroads.
- ETU model to test MRC in high-propagation channel conditions, such as crowded cities with tall buildings that cause a high number of signal echoes.

We will not be using the EPA model since it does not fit the scenarios in which OTFS is meant to perform well, such as high UE speed (a pedestrian cannot realistically go as fast as a vehicle).

4.1.2 Speed Parameter Decision

The speeds that will be used were decided upon (mostly) realistic scenarios that UE could be utilized for wireless communications. The speeds, depending on the channel, are:

- EVA model:
  - 110 km/h – This represents the typical highway speed.
  - 140 km/h – This represents a very fast highway speed.
  - 180 km/h – This represents the above-average speed of semi high-speed trains.
  - 260 km/h – This represents the above-average speed of high-speed trains.
  - 300 km/h – This represents the average speed of normal bullet trains.
  - 500 km/h – This represents the average speed of maglev trains.
- ETU model:
  - 30 km/h – This represents the typical inner-city speed.
  - 45 km/h – This represents a high inner-city speed.
  - 60 km/h – This represents the typical speed in large roads such as avenues.
  - 80 km/h – This represents a high speed in large roads.

The above speed ranges should provide realistic scenarios that we will be able to analyze, giving us the opportunity to further understand OTFS's capabilities.

4.1.3 Frequency and Spacing Parameter Decision

We will be utilizing frequencies in both FR1 and FR2 (mmWave), to have more versatile results, including the ability to compare the changes in BER between sub-mmWave and mmWave signals. For FR1 we will be using 4 GHz carrier frequency with 15 KHz spacing between the subcarriers. For FR2 we will be using 40 GHz carrier frequency (which is considered mmWave). To maintain proper spectral efficiency, we will need to adjust the spacing between the subcarriers as well. With the signal having a ten times higher frequency and the carrier frequency and subcarrier spacing being inversely related, we can estimate that the subcarrier spacing should be around 1.5 KHz. While the spacing decision is not completely dependent on relation, it will give us a good approximation for similar channel conditions.

4.1.4 Other Parameter Decision

The final parameters that we will be discussing are the SNR range, the constellation size, frame count, symbol in time count and the subcarrier number. To maintain somewhat realistic scenarios, we will be using eleven different SNR values, starting from 0 and going up to 20 with a 2 dB increment each time. The range can be understood as from inadequate to adequate reception. We will also be using three different constellation sizes (in a way to simulate adaptive modulation) which are QPSK (or 4QAM), 8QAM and 16QAM. The frame count is set to 1000, along with the symbol in time count and subcarrier number being set to 64 to maintain somewhat realistic transmissions.

4.2 Detector Comparison

We will begin by setting a baseline performance of all algorithms that are presented in the OTFS literature. The results can be seen in the figure below.

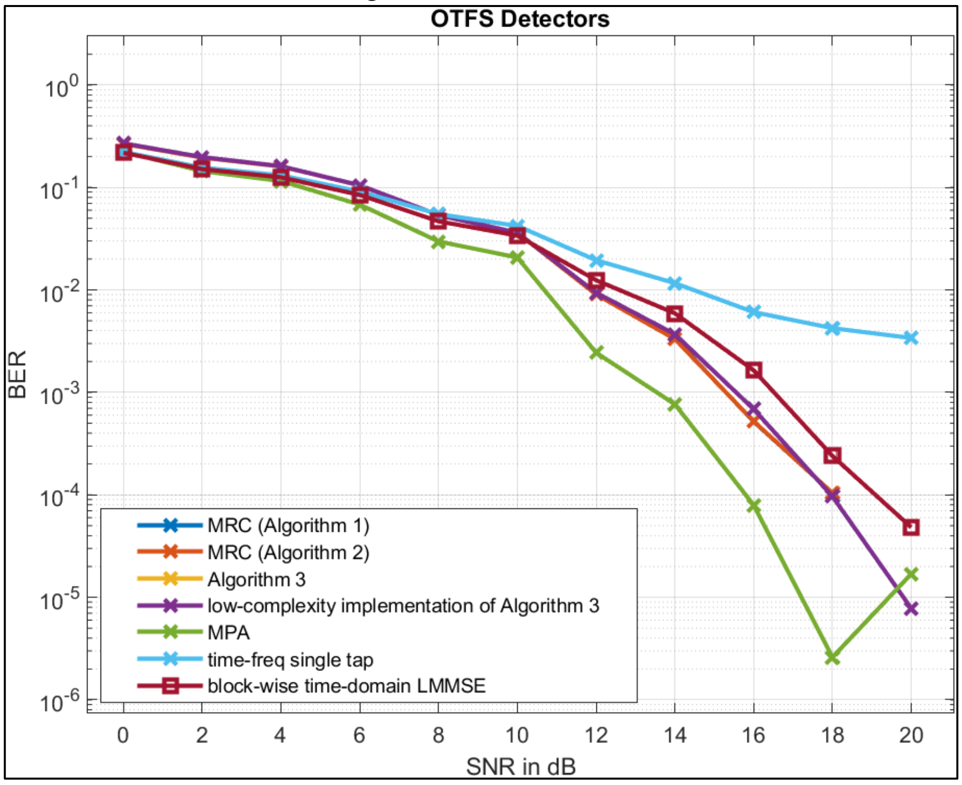

Figure 6: OTFS Detector Comparison

When we were discussing the detectors available in the literature, we said that the single-tap detector, while low-complexity, isn't performing well as it can clearly be seen. Also, the LMMSE algorithm, while performing better, it is still outperformed by all the other algorithms that we have discussed and in addition with its high complexity does not make a good candidate receiver. All of the MRC implementations perform almost identically, with the exception being algorithm 2, which performs very slightly better than the other three implementations. Finally, MPA seems to outperform the algorithms by a significant margin. It is important to note that less than 1000 frames were used along with QPSK in the above simulation due to the high complexity algorithms to keep the simulation time reduced. Simulation time was about 20 hours.

4.3 MRC-MPA Comparison in Ideal Channels

We will now focus on the implementation of algorithm 2 and the implementation of MPA. The parameters that both of the algorithms will utilize can be seen in the table below.

Frequency	4 GHz
Spacing	15 KHz
Number of Frames	1000
Symbols in Time	64
Number of Subcarriers	64
Channel Model	Synthetic/Ideal
Constellation Size	QPSK – 8QAM – 16QAM

Table 3: MRC-MPA Comparison Parameters

The MPA simulation results for all the constellation sizes combined can be seen below.

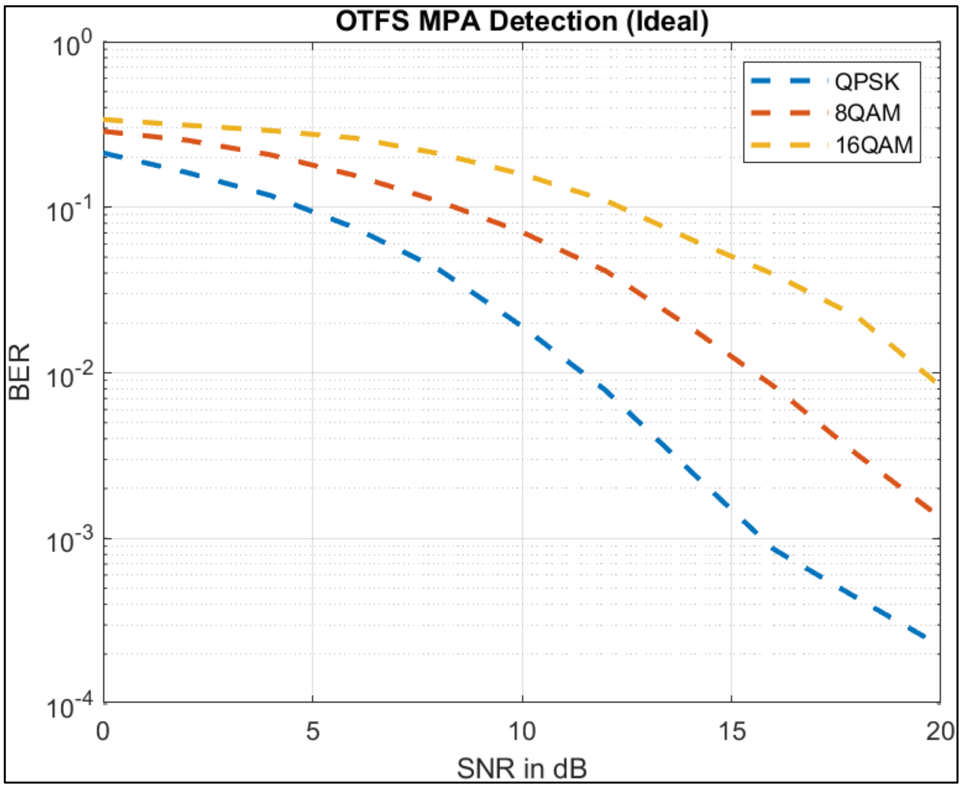

Table 4: OTFS MPA Detection (Ideal)

We can clearly see that increasing the constellation does also increase the BER. In the table below we can see the simulation results in higher detail.

SNR (dB)	**Bit-Error Rate (BER)**		
	QPSK	**8QAM**	**16QAM**
0	0.21261767578125	0.2875591634114583	0.33857049560546876
2	0.162085205078125	0.254031494140625	0.313110595703125
4	0.1175517578125	0.20712556966145834	0.2902836303710937
6	0.075234375	0.155687255859375	0.26161102294921873
8	0.0422978515625	0.10889005533854167	0.21124530029296876
10	0.0190762939453125	0.070724853515625	0.15828729248046874
12	0.0078104248046875	0.041233479817708335	0.10969305419921875
14	0.00258154296875	0.01892928059895833	0.06458331298828125
16	8.6279296875E-4	0.008323079427083335	0.03942620849609375
18	4.356689453125E-4	0.0032068684895833334	0.02182891845703125
20	2.230224609375E-4	0.0013299153645833332	0.00825360107421875

Table 5: OTFS MPA Detection (Ideal)

It is important to note that for all three constellations the simulation time was around 120 hours.

The MRC simulation results for all the constellation sizes combined can be seen below.

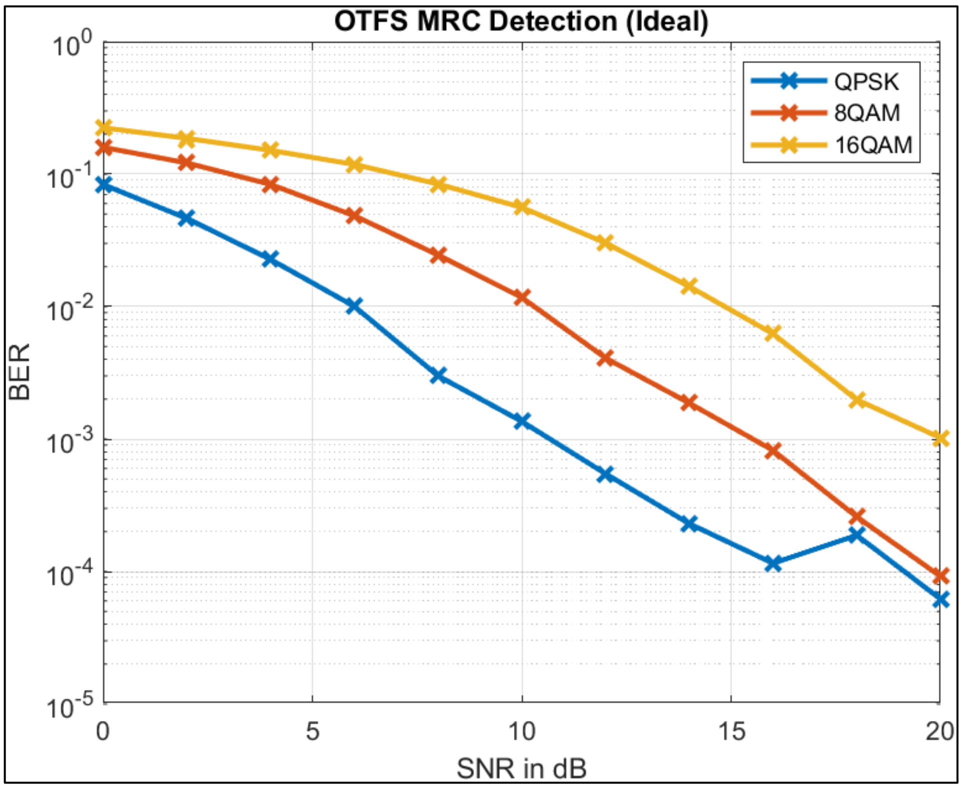

Table 6: OTFS MRC Detection (Ideal)

Similar results can be observed in the MRC simulations, although it is clear that a lower BER can be achieved. The table below shows the simulation results in higher detail.

SNR (dB)	Bit-Error Rate (BER)		
	QPSK	8QAM	16QAM
0	0.0833913225446429	0.159323846726191	0.223957938058036
2	0.0463051060267857	0.121507905505952	0.186284109933036
4	0.0226870814732143	0.0835908668154762	0.151117675781250
6	0.0100410156250000	0.0484647507440476	0.118429617745536
8	0.00299693080357143	0.0245164620535714	0.0841439034598214
10	0.00135797991071429	0.0117121465773810	0.0560512695312500
12	0.000547293526785714	0.00408872767857143	0.0302220982142857
14	0.000229492187500000	0.00186662946428571	0.0141683175223214
16	0.000115513392857143	0.000820777529761905	0.00622963169642857
18	0.000187918526785714	0.000258742559523810	0.00198032924107143
20	6.19419642857143e-05	9.18898809523810e-05	0.00102162388392857

Table 7: OTFS MRC Detection (Ideal)

It is clear, that MRC shows a significant overall improvement, contrary to the simulation with all of the detectors combined, which showed MPA outperform all other algorithms. The combined results can be seen in the figure below.

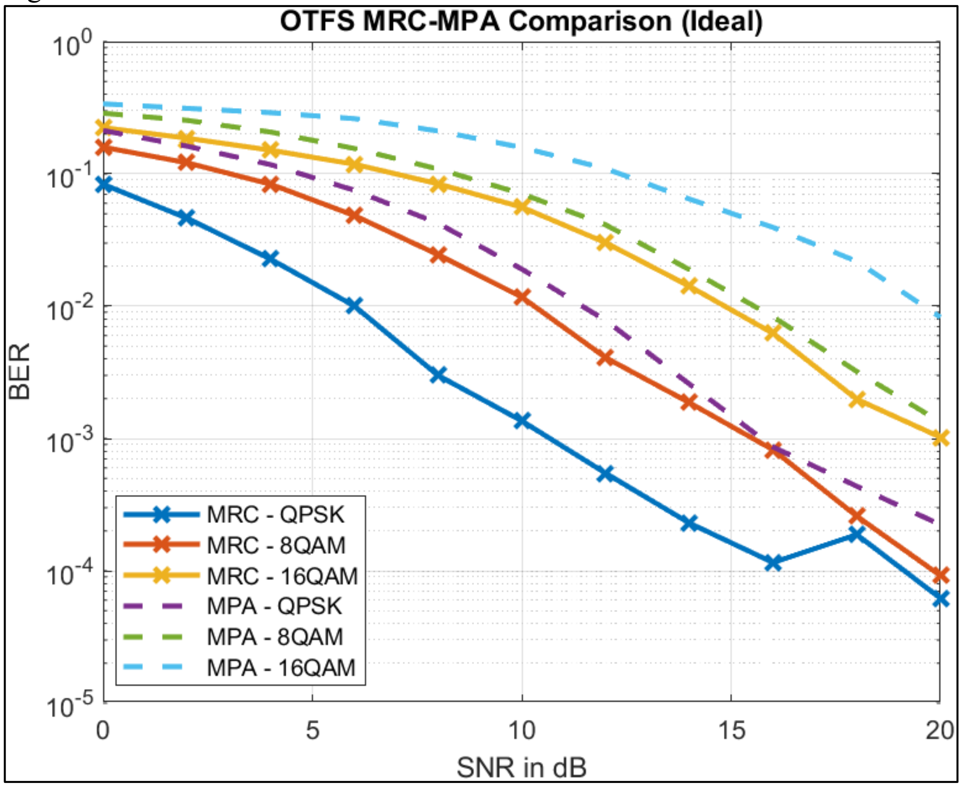

Figure 7: OTFS MRC-MPA Comparison (Ideal)

It is easily observable that QPSK using MPA detection is comparable to 8QAM using MRC, and 8QAM using MPA is comparable to 16QAM using MRC. The discrepancy between the overall comparison and the above graph might be due to differences in the parameters, importantly the count of frames. It is possible that the above results would be different if less frames were used, giving us similar results to the comparison between all the detectors. The table below shows the combined results of the above figure, rounded to 4 decimal digits.

SNR (dB)	Bit-Error Rate (BER)					
	QPSK		8QAM		16QAM	
	MRC	MPA	MRC	MPA	MRC	MPA
0	0.0834	0.2126	0.1593	0.2876	0.2240	0.3386
2	0.0463	0.1621	0.1215	0.2540	0.1863	0.3131
4	0.0227	0.1176	0.0836	0.2071	0.1511	0.2903
6	0.0101	0.0752	0.0485	0.1557	0.1184	0.2616
8	0.0030	0.0423	0.0245	0.1089	0.0841	0.2112
10	0.0014	0.0191	0.0117	0.0707	0.0561	0.1583
12	0.0006	0.0078	0.0041	0.0412	0.0302	0.1097
14	0.0002	0.0026	0.0019	0.0189	0.0142	0.0646
16	0.0001	8.628E-4	0.0008	0.0083	0.0062	0.0394
18	0.0002	4.357E-4	0.0003	0.0032	0.0020	0.0218
20	6.194E-5	2.231E-4	9.189E-5	0.0013	0.0010	0.0083

Table 8: OTFS MRC-MPA Comparison (Ideal)

Based on the significant difference on performance and the overall simulation times (MRC finishes in less than 15 minutes), the remainder of the simulations will be based on the MRC detection implementation (algorithm 2).

4.4 MRC in Mid-Propagation Channels (EVA)

4.4.1 QPSK Results using FR1

We will now simulate the MRC detection in mid-propagation channels, using 4 GHz carrier along with the constellation size of 4 (QPSK). The simulations will begin at 110 km/h and eventually reach 500 km/h, as we have mentioned earlier. For QPSK, the results can be seen in the following figure.

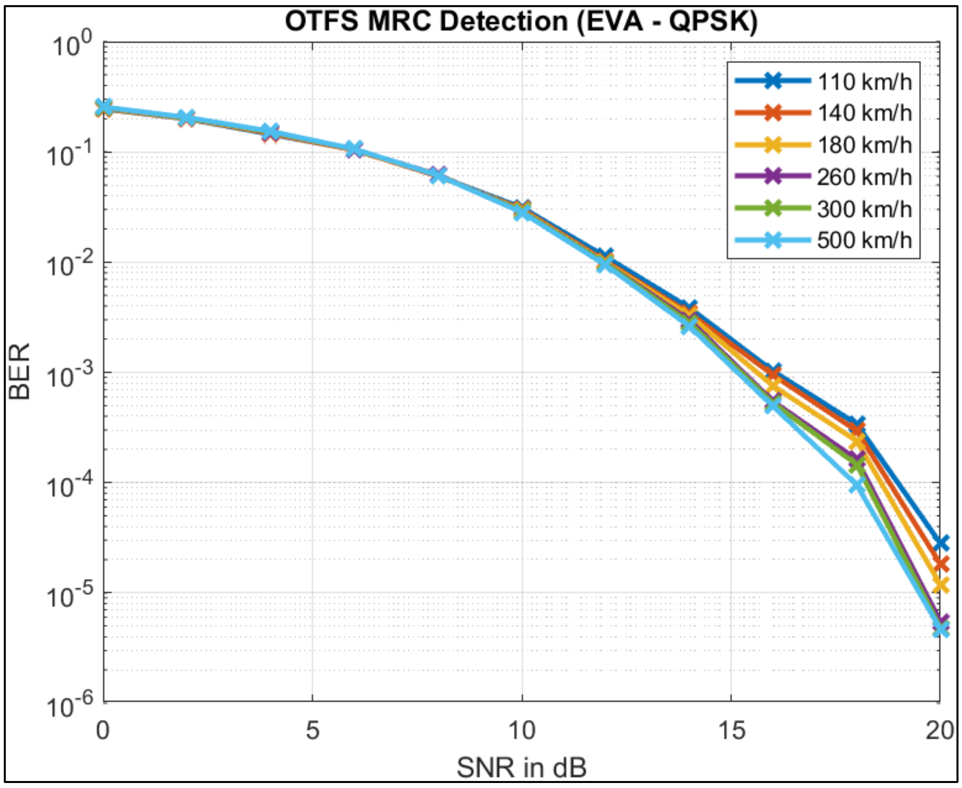

Figure 8: OTFS MRC Detection (EVA - QPSK)

As we can see, for the first 10 dB of SNR, regardless of speed the BER remains almost identical. Beyond 10 dB, we can observe that the graph begins to “split”. Counterintuitively, OTFS using MRC receivers seems to provide a lower BER the higher the speed of the UE is. The difference between 110 km/h and 500 km/h is very distinct, while 260, 300 and 500 km/h maintain the same BER albeit with marginal difference around 18 dB of SNR. In the following table we can see the exact values of the above figure.

SNR (dB)	**Bit-Error Rate (BER)**					
	User Equipment Speed (km/h)					
	110	**140**	**180**	**260**	**300**	**500**
0	0.2449	0.2469	0.2494	0.2524	0.2540	0.2555
2	0.1990	0.2006	0.2020	0.2035	0.2038	0.2057
4	0.1447	0.1470	0.1497	0.1513	0.1520	0.1537
6	0.1040	0.1037	0.1045	0.1053	0.1058	0.1061
8	0.0602	0.0601	0.0612	0.0619	0.0614	0.0612
10	0.0312	0.0303	0.0296	0.0290	0.0288	0.0283
12	0.0112	0.0102	0.0099	0.0097	0.0097	0.0094
14	0.0039	0.0035	0.0033	0.0030	0.0028	0.0026
16	0.0010	0.0009	0.0008	0.0005	0.0005	0.0005
18	0.0003	0.0003	0.0002	0.0002	0.0001	0.0001
20	2.799E-5	1.823E-5	1.185E-5	5.469E-6	4.818E-6	4.688E-6

Table 9: OTFS MRC Detection (EVA - QPSK)

It is very clear, that throughout all the speeds, for each increment in SNR, OTFS shows incredible stability, with the BER having minor differences between speed increase and SNR increase, which could also be cause due to differences between the simulations since the original signal is randomized and different each time.

4.4.2 8QAM Results using FR1

We will repeat the same simulation with a constellation size of 8 (8QAM). The results can be seen in the figure below.

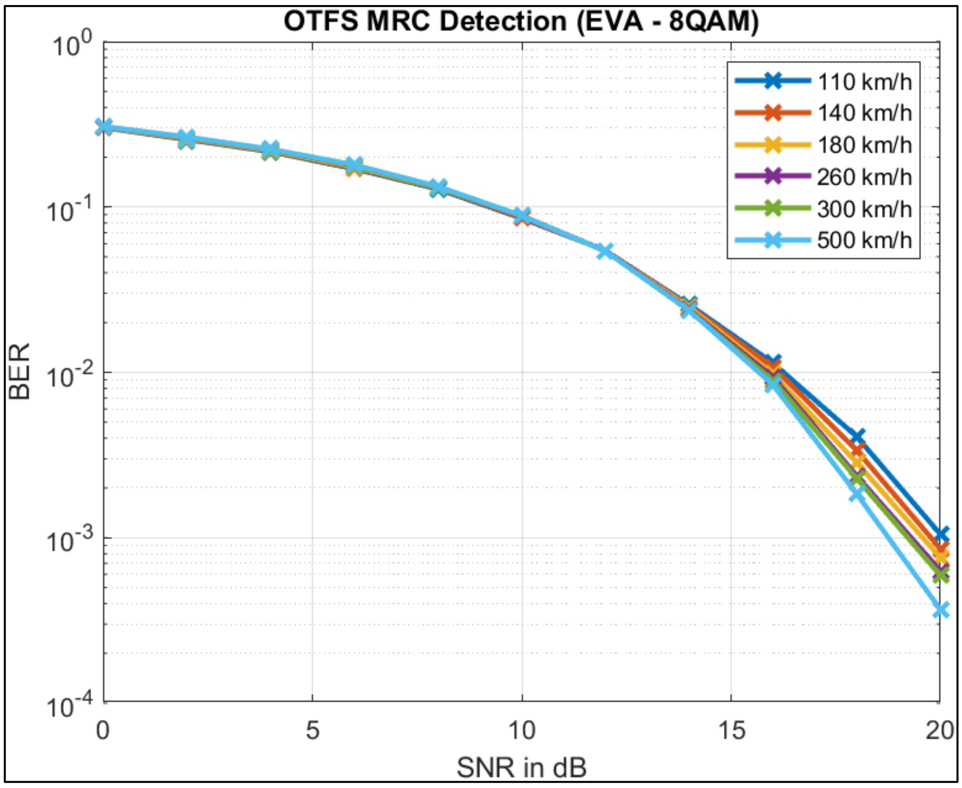

Figure 9: OTFS MRC Detection (EVA - 8QAM)

A similar remark can be made for 8QAM, although, as we can see it maintains stability until the 14 dB SNR point, after which the graph begins to "split" like before. It is also important to note that the 500 km/h line poses a higher difference between the 260 and 300 km/h lines, unlike before, though this might be completely coincidental. In the table below we can see the values in higher detail.

SNR (dB)	**Bit-Error Rate (BER)**					
	User Equipment Speed (km/h)					
	110	**140**	**180**	**260**	**300**	**500**
0	0.3018	0.3032	0.3049	0.3063	0.3068	0.3074
2	0.2558	0.2584	0.2609	0.2635	0.2641	0.2653
4	0.2158	0.2182	0.2196	0.2225	0.2233	0.2239
6	0.1701	0.1726	0.1749	0.1777	0.1785	0.1798
8	0.1282	0.1297	0.1311	0.1314	0.1310	0.1329
10	0.0854	0.0864	0.0887	0.0886	0.0887	0.0886
12	0.0543	0.0543	0.0537	0.0544	0.0541	0.0541
14	0.0256	0.0251	0.0247	0.0239	0.0236	0.0237
16	0.0114	0.0108	0.0098	0.0093	0.0089	0.0084
18	0.0041	0.0034	0.0028	0.0024	0.0023	0.0018
20	0.0010	0.0009	0.0008	0.0006	0.0006	0.0004

Table 10: OTFS MRC Detection (EVA - 8QAM)

4.4.3 16QAM Results using FR1

Once again, we will repeat the previous simulations with a constellation size of 16 (16QAM). The results can be seen in the figure below.

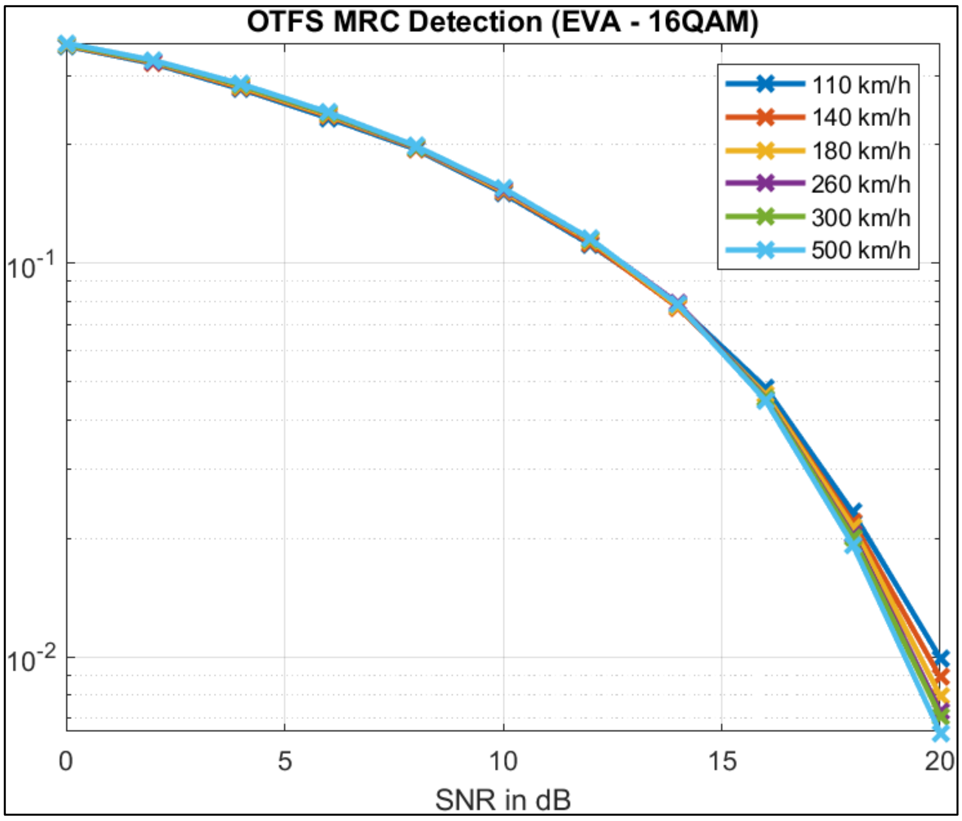

Figure 10: OTFS MRC Detection (EVA - 16QAM)

This time, the stability is maintained up until the 16 dB point, after which the graph once again begins to "split". Unlike 8QAM, the 500 km/h line seems to have a slight difference between the 260 and 300 km/h lines. In the table below we can see the values in higher detail.

SNR (dB)	Bit-Error Rate (BER)					
	User Equipment Speed (km/h)					
	110	140	180	260	300	500
0	0.3564	0.3582	0.3592	0.3608	0.3607	0.3628
2	0.3216	0.3235	0.3249	0.3260	0.3266	0.3281
4	0.2771	0.2794	0.2812	0.2833	0.2838	0.2859
6	0.2339	0.2367	0.2380	0.2398	0.2404	0.2421
8	0.1949	0.1963	0.1970	0.1974	0.1981	0.1989
10	0.1506	0.1521	0.1529	0.1544	0.1553	0.1554
12	0.1115	0.1124	0.1143	0.1150	0.1151	0.1154
14	0.0782	0.0772	0.0777	0.0791	0.0785	0.0785
16	0.0484	0.0465	0.0466	0.0456	0.0454	0.0449
18	0.0233	0.0220	0.0214	0.0204	0.0201	0.0193
20	0.0099	0.0089	0.0080	0.0073	0.0071	0.0064

Table 11: OTFS MRC Detection (EVA - 16QAM)

4.4.4 Result Comparison at Highest Speed using FR1

Since the highest speed provides the highest reduction in BER, we will compare the highest speeds of each constellation against each other. Combining the three lines gives us the following figure:

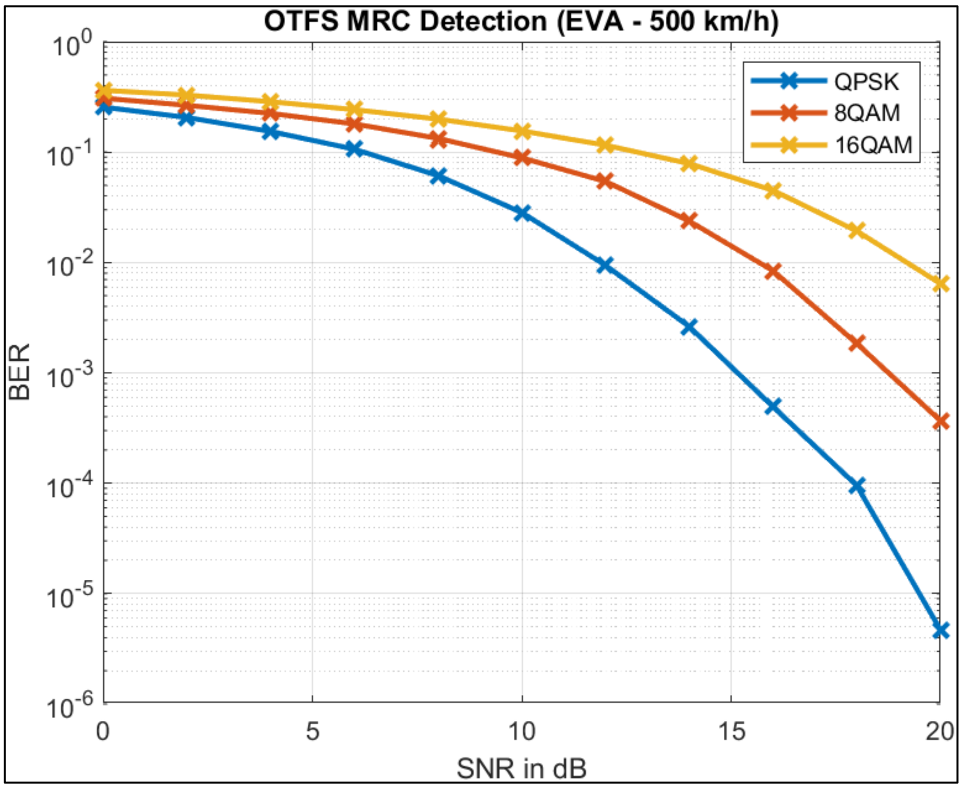

Figure 11: OTFS MRC Detection (EVA - 500 km/h)

As is to be expected, with the increase of the constellation size we can see the increase of the BER. QPSK appears to perform very well, given the speed that the UE is moving at. It is important to mention that all three constellations have similar BER up until 4 dB of SNR. The values can be seen in higher detail in the table below.

SNR (dB)	**Bit-Error Rate (BER)**		
	QPSK	**8QAM**	**16QAM**
0	0.255472005208333	0.307421180555556	0.362835221354167
2	0.205670052083333	0.265256076388889	0.328127083333333
4	0.153679166666667	0.223870920138889	0.285901953125
6	0.106076041666667	0.179778732638889	0.24205546875
8	0.0611868489583333	0.132947829861111	0.1988599609375
10	0.02832109375	0.0885748263888889	0.155369661458333
12	0.00940859375	0.0541227430555556	0.115376171875
14	0.002623046875	0.0236815972222222	0.0785119791666667
16	0.000492317708333333	0.00840034722222222	0.0448932942708333
18	9.53125E-5	0.00184236111111111	0.019273046875
20	4.6875E-6	0.000367447916666667	0.0064169921875

Table 12: OTFS MRC Detection (EVA - 500 km/h)

4.4.5 QPSK Results using mmWave

We will now repeat the previous simulations, again for QPSK but we will be increasing the carrier frequency to 40 GHz (mmWave) and reducing the spacing to 1.5 KHz. The results with the increased frequency can be seen in the figure below.

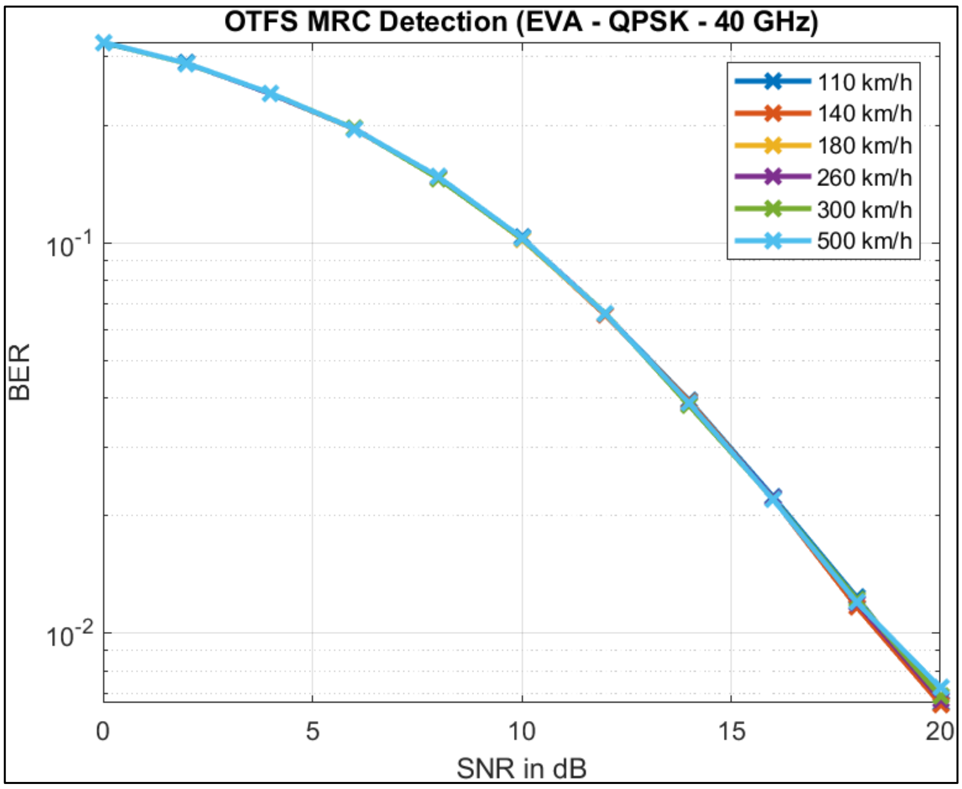

Figure 12: OTFS MRC Detection (EVA - QPSK - 40 GHz)

It is clear that there has been a significant increase in the BER, but it is easily observable that the stability in all speeds remains marginally equal. In the following table we can see the values in higher detail.

SNR (dB)	**Bit-Error Rate (BER)**					
	User Equipment Speed (km/h)					
	110	**140**	**180**	**260**	**300**	**500**
0	0.3259	0.3266	0.3266	0.3268	0.3266	0.3271
2	0.2889	0.2889	0.2884	0.2882	0.2883	0.2887
4	0.2410	0.2413	0.2417	0.2414	0.2421	0.2420
6	0.1958	0.1959	0.1961	0.1965	0.1964	0.1961
8	0.1470	0.1471	0.1473	0.1473	0.1464	0.1480
10	0.1032	0.1027	0.1022	0.1030	0.1025	0.1031
12	0.0656	0.0652	0.0656	0.0659	0.0660	0.0657
14	0.0394	0.0392	0.0387	0.0386	0.0383	0.0388
16	0.0223	0.0221	0.0220	0.0222	0.0220	0.0220
18	0.0123	0.0117	0.0122	0.0119	0.0122	0.0120
20	0.0066	0.0066	0.0069	0.0067	0.0069	0.0072

Table 13: OTFS MRC Detection (EVA - QPSK - 40 GHz)

4.4.6 8QAM Results using mmWave

Continuing on, we repeat the previous simulation with a constellation size of 8 (8QAM) while keeping the frequency the same. The results can be seen in the following figure.

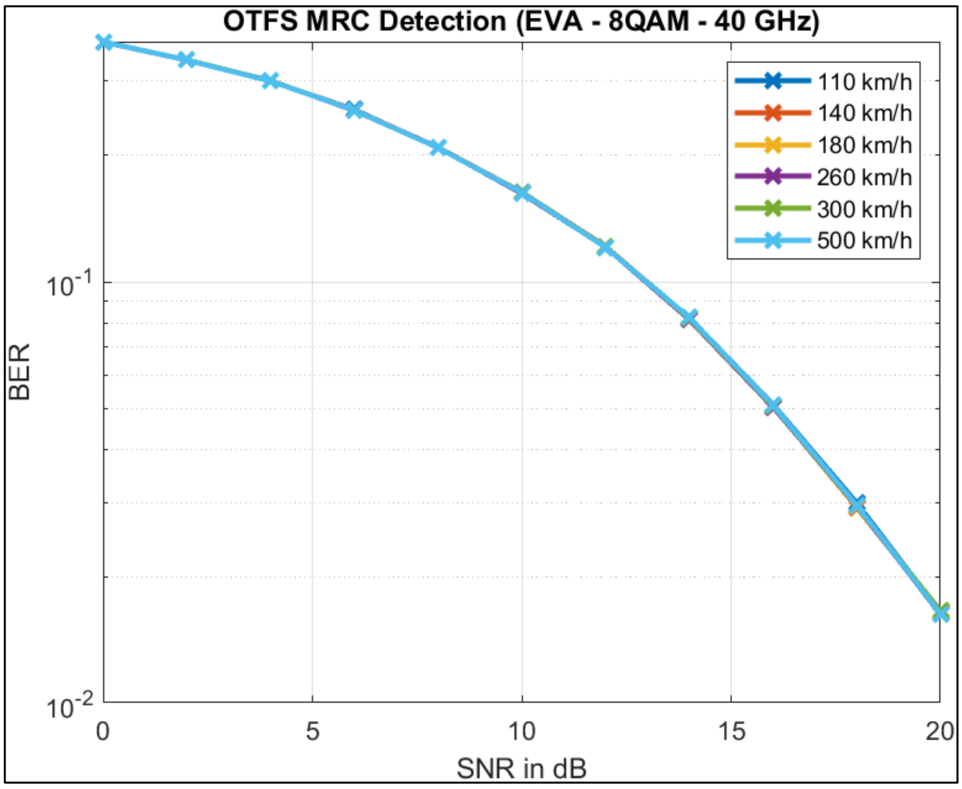

Figure 13: OTFS MRC Detection (EVA - 8QAM - 40 GHz)

Similarly, 8QAM maintains stability in all speeds, again, having marginal differences between them. We can see values in higher detail in the following tables.

SNR (dB)	Bit-Error Rate (BER)					
	User Equipment Speed (km/h)					
	110	140	180	260	300	500
0	0.3259	0.3266	0.3266	0.3268	0.3266	0.3271
2	0.2889	0.2889	0.2884	0.2882	0.2883	0.2887
4	0.2410	0.2413	0.2417	0.2414	0.2421	0.2420
6	0.1958	0.1959	0.1961	0.1965	0.1964	0.1961
8	0.1470	0.1471	0.1473	0.1473	0.1464	0.1480
10	0.1032	0.1027	0.1022	0.1030	0.1025	0.1031
12	0.0656	0.0652	0.0656	0.0659	0.0660	0.0657
14	0.0394	0.0392	0.0387	0.0386	0.0383	0.0388
16	0.0223	0.0221	0.0220	0.0222	0.0220	0.0220
18	0.0123	0.0117	0.0122	0.0119	0.0122	0.0120
20	0.0066	0.0066	0.0069	0.0067	0.0069	0.0072

Table 14: OTFS MRC Detection (EVA - 8QAM - 40 GHz)

4.4.7 16QAM Results using mmWave

Finally, we will run the previous simulations with a constellation size of 16 (16QAM), keeping the frequency the same. The results can be seen in the figure below.

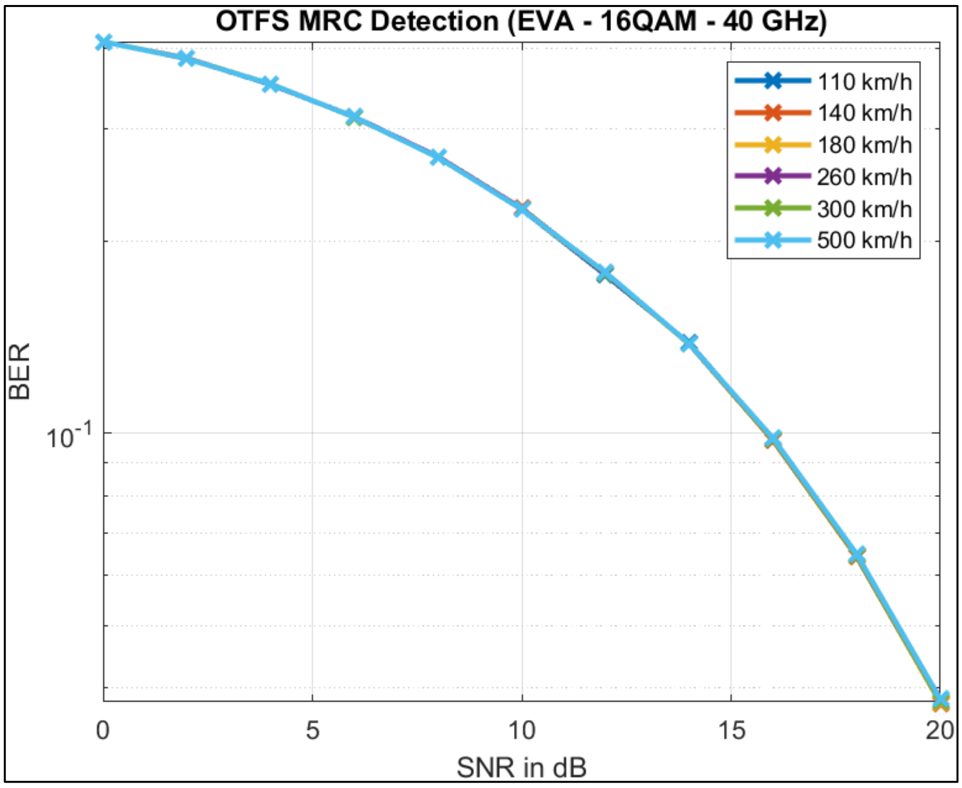

Figure 14: OTFS MRC Detection (EVA - 16QAM - 40 GHz)

Like the previous simulations, 16QAM also maintains stability in all speeds having marginal differences. The values can be seen with higher detail in the following tables.

SNR (dB)	**Bit-Error Rate (BER)**					
	User Equipment Speed (km/h)					
	110	**140**	**180**	**260**	**300**	**500**
0	0.4094	0.4093	0.4095	0.4095	0.4096	0.4094
2	0.3854	0.3851	0.3859	0.3856	0.3853	0.3852
4	0.3511	0.3515	0.3516	0.3515	0.3518	0.3516
6	0.3122	0.3120	0.3124	0.3125	0.3119	0.3123
8	0.2702	0.2700	0.2698	0.2706	0.2701	0.2699
10	0.2239	0.2247	0.2240	0.2246	0.2241	0.2239
12	0.1769	0.1778	0.1776	0.1776	0.1783	0.1785
14	0.1384	0.1379	0.1380	0.1382	0.1379	0.1381
16	0.0981	0.0975	0.0979	0.0981	0.0979	0.0984
18	0.0648	0.0642	0.0644	0.0644	0.0644	0.0648
20	0.0379	0.0379	0.0380	0.0384	0.0381	0.0384

Table 15: OTFS MRC Detection (EVA - 16QAM - 40 GHz)

4.4.8 Result Comparison at Highest Speed using mmWave

While there is almost no difference between the speeds, we will be using the highest speed to compare the constellations between each other. Combining the lines, we get the following result.

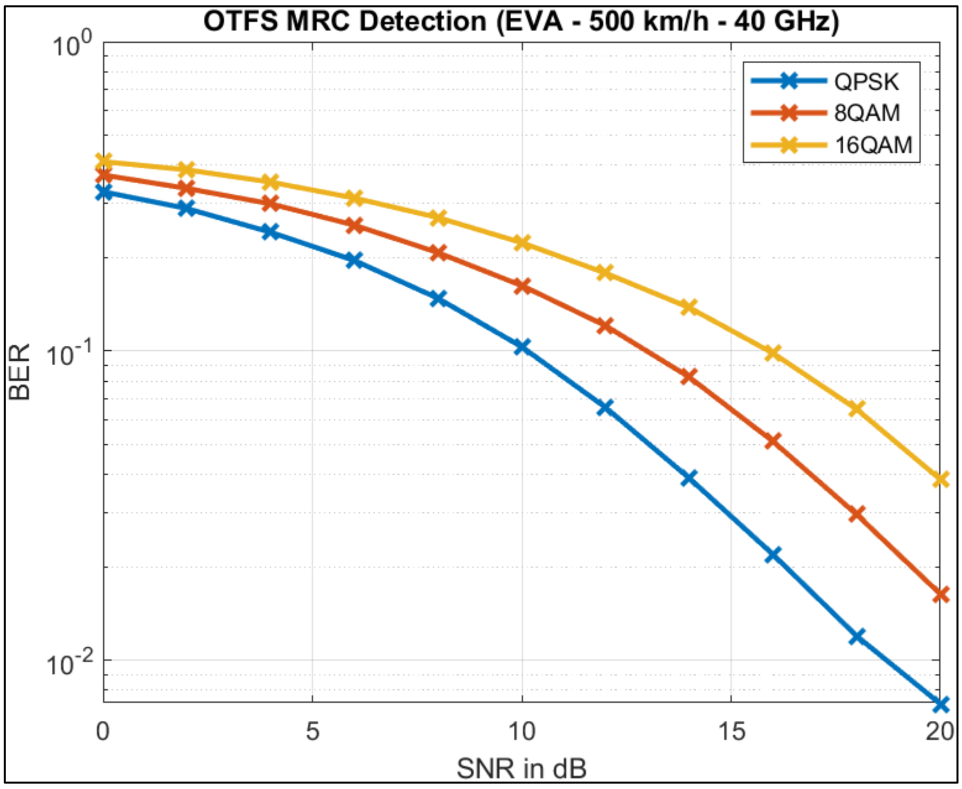

Figure 15: OTFS MRC Detection (EVA - 500 km/h - 40 GHz)

Once again, the increase of BER is to be expected as the constellation size increases. It appears that the different constellations maintain equal difference with the increase of SNR, beginning from 0 dB. Clearly, QPSK outperforms the other constellation sizes again. With that being said, the values can be seen in higher detail in the table below.

SNR (dB)	Bit-Error Rate (BER)		
	QPSK	8QAM	16QAM
0	0.327137630208333	0.370707638888889	0.409433463541667
2	0.288737109375	0.335421006944444	0.385205729166667
4	0.242023567708333	0.299066840277778	0.351564453125
6	0.196149869791667	0.25529140625	0.312286458333333
8	0.147985677083333	0.207860243055556	0.269937565104167
10	0.103121875	0.162723524305556	0.2238921875
12	0.0656953125	0.120738194444444	0.178537630208333
14	0.038803515625	0.0823085069444444	0.138113346354167
16	0.0219877604166667	0.0510250868055556	0.0983704427083333
18	0.0119766927083333	0.0295951388888889	0.0647857421875
20	0.00722591145833333	0.0163607638888889	0.0384206380208333

Table 16: OTFS MRC Detection (EVA - 500 km/h - 40 GHz)

4.4.9 Result Comparison of FR1 and mmWave at Highest Speed

To properly compare the two different frequency ranges, we will combine the highest speed figures of both frequencies into one and directly compare it. The combined graph can be seen below.

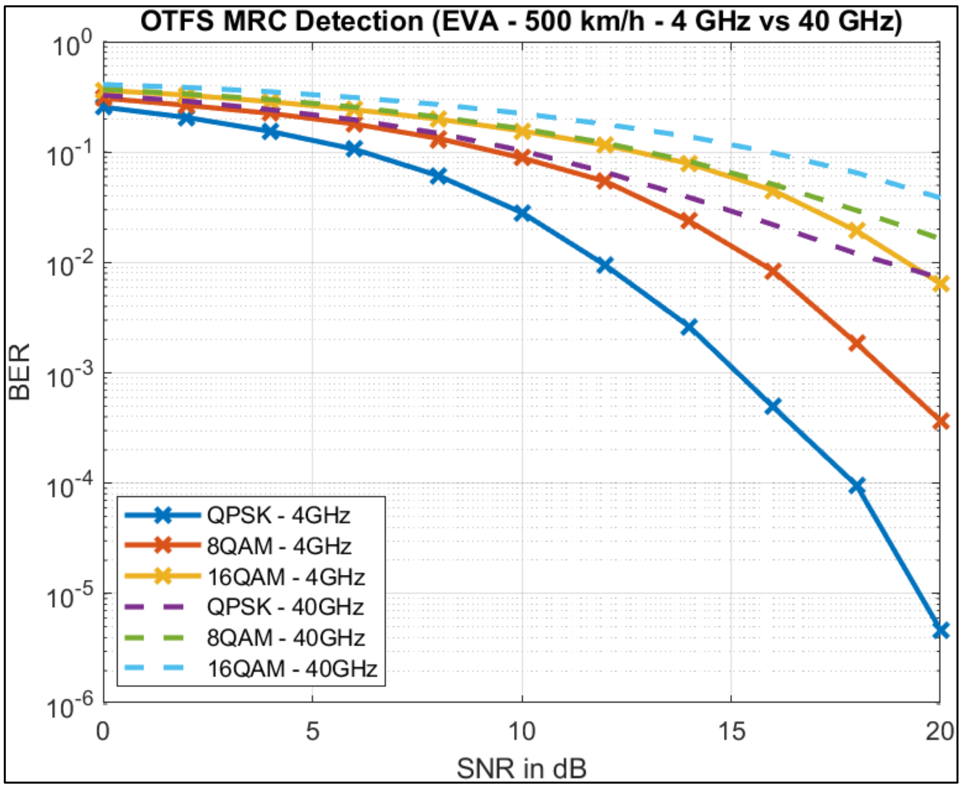

Figure 16: OTFS MRC Detection (EVA - 500 km/h - 4 GHz vs 40 GHz)

While the fact that 4 GHz carriers having lower BER was obvious, what is very interesting is the fact that 40 GHz QPSK and 8QAM is competing in BER with the 4 GHz 16QAM, up until the 16 dB mark. Another important remark to be made is that 40 GHz QPSK is marginally equal to 4 GHz 8QAM up until the 12 dB mark. Finally, we should point out that 40 GHz 16QAM does not fall far behind the 4 GHz 16QAM up until the 16 dB mark. While the current BER that is present is not enough to ensure a reliable connection, it is possible for mmWave to be used along with OTFS for faster (though less reliable) transfer speeds in medium propagation channels. For a better understanding, the values have been combined into a single table that can be seen below.

SNR (dB)	Bit-Error Rate (BER)					
	QPSK		8QAM		16QAM	
	4 GHz	40 GHz	4 GHz	40 GHz	4 GHz	40 GHz
0	0.2555	0.3271	0.3074	0.3271	0.3628	0.4094
2	0.2057	0.2887	0.2653	0.2887	0.3281	0.3852
4	0.1537	0.2420	0.2239	0.2420	0.2859	0.3516
6	0.1061	0.1961	0.1798	0.1961	0.2421	0.3123
8	0.0612	0.1480	0.1329	0.1480	0.1989	0.2699
10	0.0283	0.1031	0.0886	0.1031	0.1554	0.2239
12	0.0094	0.0657	0.0541	0.0657	0.1154	0.1785
14	0.0026	0.0388	0.0237	0.0388	0.0785	0.1381
16	0.0005	0.0220	0.0084	0.0220	0.0449	0.0984
18	0.0001	0.0120	0.0018	0.0120	0.0193	0.0648
20	4.688E-6	0.0072	0.0004	0.0072	0.0064	0.0384

Table 17: OTFS MRC Detection (EVA - 500 km/h - 4 GHz vs 40 GHz)

4.5 MRC in High-Propagation Channels (ETU)

4.5.1 QPSK Results using FR1

With the EVA simulations completed, it is important to also simulate high-propagation channel conditions to have a complete image of the effectiveness of OTFS (using an MRC detector). Like before, we will begin setting the constellation size to 4 (QPSK), the carrier frequency to 4 GHz and of course the channel model to ETU. Since we will be using a higher propagation channel, it is also important to reduce the speeds to more realistic in-city speeds. We can see the results in the following figure.

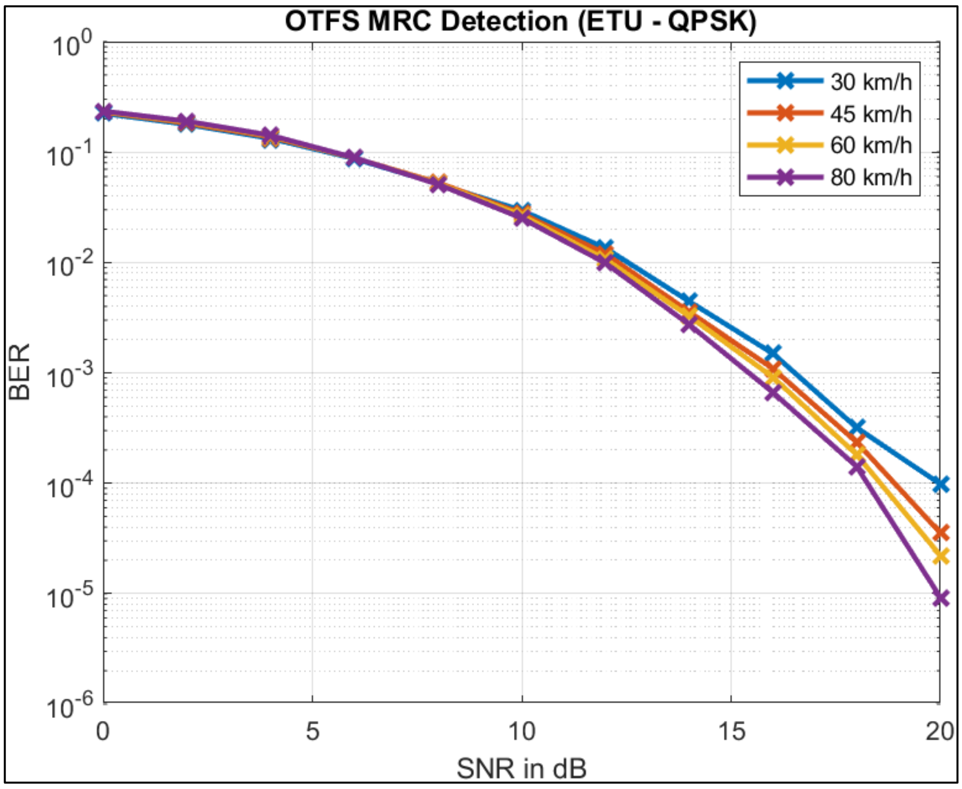

Figure 17: OTFS MRC Detection (ETU - QPSK)

Similar to QPSK in mid-propagation conditions, we can see OTFS being stable up until 10 dB of SNR, followed by the "splitting" of the lines. Like before, the BER decreases with the increase of speed and directly comparing the results we can notice that the BER is in fact similar with the high speeds in the EVA channel. Obviously, very high speeds that we tested for the EVA channel cannot be achieved in a city (even with the use of trains) due to residential area limitations along with high traffic in crowded areas, such as city centers. In the following table we can see the values in higher detail.

SNR (dB)	**Bit-Error Rate (BER)**			
	User Equipment Speed (km/h)			
	30	**45**	**60**	**80**
0	0.2239	0.2305	0.2327	0.2352
2	0.1781	0.1827	0.1869	0.1903
4	0.1326	0.1367	0.1399	0.1418
6	0.0872	0.0884	0.0886	0.0890
8	0.0515	0.0528	0.0518	0.0510
10	0.0296	0.0277	0.0267	0.0251
12	0.0135	0.0120	0.0107	0.0098
14	0.0044	0.0036	0.0032	0.0027
16	0.0015	0.0011	0.0009	0.0007
18	0.0003	0.0002	0.0002	0.0001
20	0.0001	3.530E-5	2.204E-5	9.208E-6

Table 18: OTFS MRC Detection (ETU - QPSK)

Like in the EVA simulation, it is very clear that OTFS maintains a very similar BER up until 10 dB of SNR. After the 10 dB mark, while the lines begin to "split", the do maintain relatively close BER. While some results, again, may differ between simulations due to the randomization of variables, such as the signal, the simulations do provide a rough idea of the effectiveness of OTFS regardless.

4.5.2 8QAM Results using FR1

Maintaining the same parameters and increasing the constellation size to 8 (8QAM). Repeating the previous simulation for 4 GHz carrier frequency, yields the results that can be seen in the following figure.

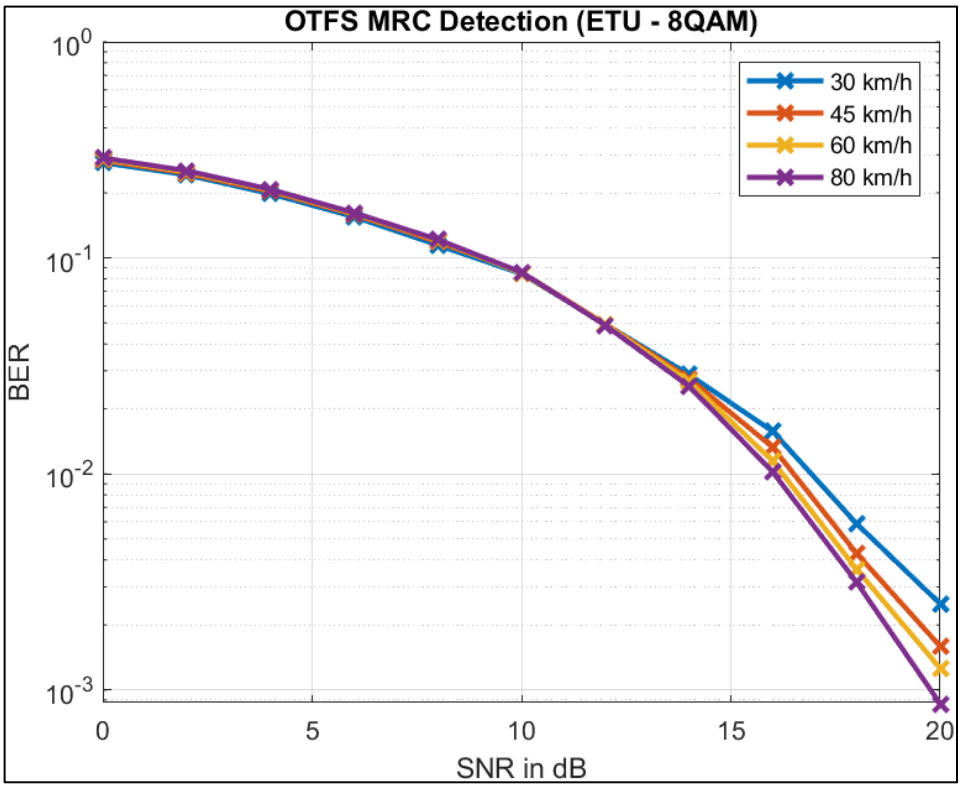

Figure 18: OTFS MRC Detection (ETU - 8QAM)

Like in the EVA channel simulations, 8QAM poses stability up to a higher number of dBs than QPSK. Specifically, we can see that up to 12 dB of SNR the BER is marginally equal among the different speeds, with the lines beginning to "split" after 14 dB of SNR. Again, the higher speed poses a lower BER compared to slower speeds. The results can be seen in higher detail in the following table.

SNR (dB)	**Bit-Error Rate (BER)**			
	User Equipment Speed (km/h)			
	30	**45**	**60**	**80**
0	0.2759	0.2828	0.2869	0.2904
2	0.2424	0.2468	0.2504	0.2534
4	0.1981	0.2031	0.2061	0.2078
6	0.1551	0.1589	0.1604	0.1619
8	0.1147	0.1186	0.1209	0.1217
10	0.0845	0.0858	0.0849	0.0858
12	0.0493	0.0494	0.0492	0.0487
14	0.0288	0.0277	0.0268	0.0255
16	0.0157	0.0133	0.0115	0.0102
18	0.0059	0.0043	0.0036	0.0032
20	0.0025	0.0016	0.0013	0.0009

Table 19: OTFS MRC Detection (ETU - 8QAM)

4.5.3 16QAM Results using FR1

Once again, we will repeat the previous simulation for a constellation size of 16 (16QAM) while keeping the other parameters the same. The results can be seen in the following figure.

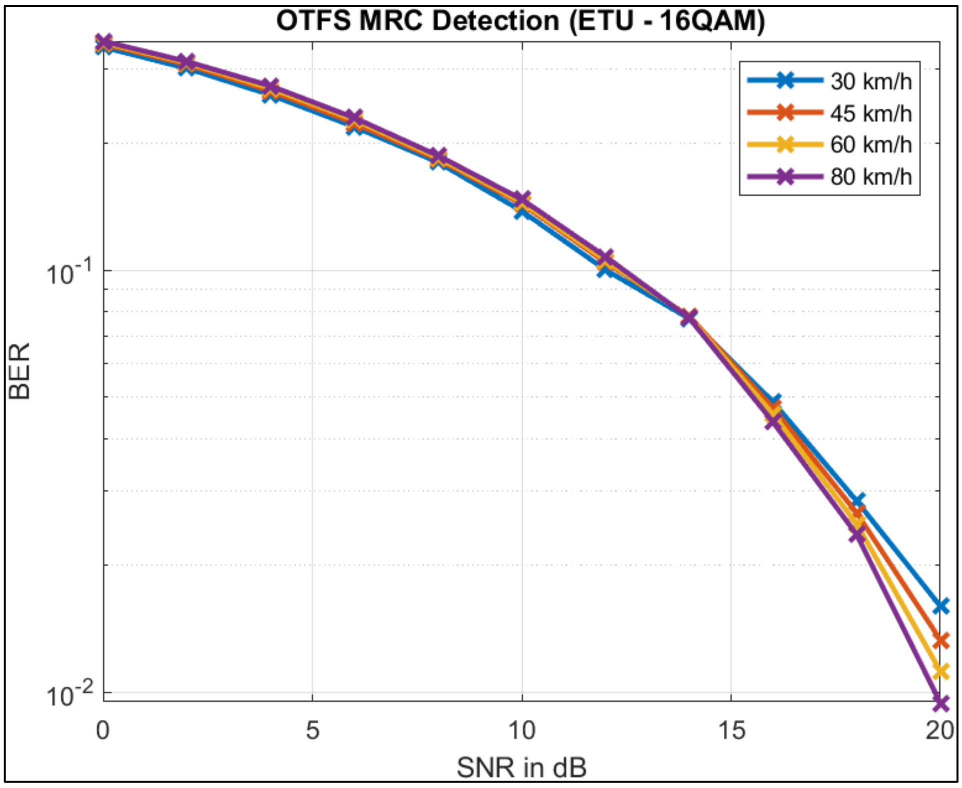

Figure 19: OTFS MRC Detection (ETU - 16QAM)

Unlike in the EVA channel simulation with 16QAM, we can see that the different speeds do have, although minimal, differences in the BER, with the slower speed having initially a lower BER and the higher speed having initially a higher BER. At the 14 dB mark, the BER of all the speeds becomes marginally equal before starting to "split" and posing the same effect that all others simulations have shown (that being the fact that the highest speed has the lowest BER). The values can be seen in the following table in higher detail.

SNR (dB)	**Bit-Error Rate (BER)**			
	User Equipment Speed (km/h)			
	30	**45**	**60**	**80**
0	0.3385	0.3445	0.3472	0.3497
2	0.3015	0.3078	0.3108	0.3129
4	0.2606	0.2664	0.2703	0.2739
6	0.2191	0.2236	0.2276	0.2301
8	0.1805	0.1832	0.1850	0.1872
10	0.1383	0.1439	0.1453	0.1475
12	0.1003	0.1046	0.1058	0.1078
14	0.0769	0.0779	0.0775	0.0771
16	0.0488	0.0472	0.0456	0.0437
18	0.0284	0.0266	0.0251	0.0236
20	0.0160	0.0133	0.0112	0.0094

Table 20: OTFS MRC Detection (ETU - 16QAM)

4.5.4 Result Comparison at Highest Speed using FR1

In the ETU model, it is apparent that the highest speed provides the best performing results. With that being said, we will be comparing the highest speeds among the 3 different constellation sizes, like we did in the EVA model. The results can be seen in the following figure.

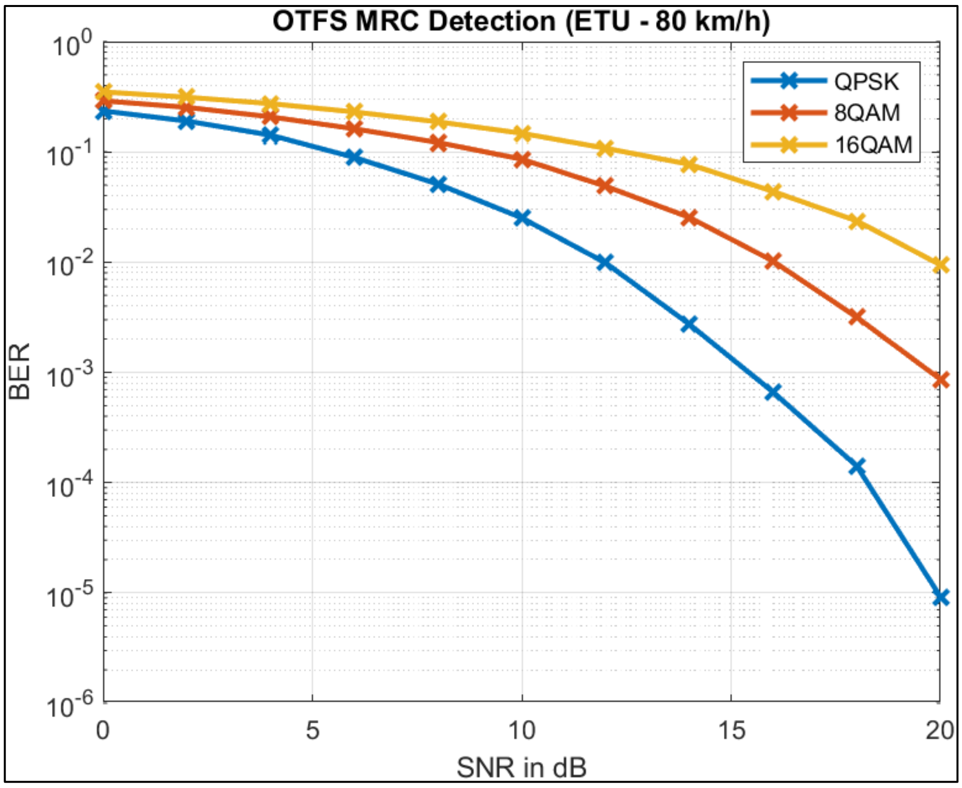

Figure 20: OTFS MRC Detection (ETU - 80 km/h)

It is easily observable that in high-propagation conditions, MRC performs roughly the same as in the mid-propagation channel, though, that comparison is not analogous to the actual conditions since in the ETU model we have significantly lower speeds (both speed ranges were chosen based on realistic scenarios). The values of the above figure can be seen in higher detail in the table below.

SNR (dB)	Bit-Error Rate (BER)		
	QPSK	8QAM	16QAM
0	0.235165178571429	0.290375651041667	0.349743024553571
2	0.190330357142857	0.253363932291667	0.312949846540179
4	0.141791573660714	0.207770647321429	0.273892578125
6	0.0889684709821429	0.161895647321429	0.230126255580357
8	0.0510327845982143	0.121650111607143	0.187181012834821
10	0.0251474609375	0.0858243117559524	0.147450474330357
12	0.0098447265625	0.0487483258928571	0.107826660156250
14	0.00273270089285714	0.0254594494047619	0.0771177455357143
16	0.000663504464285714	0.0102150297619048	0.0437147739955357
18	0.000141322544642857	0.00315848214285714	0.0236342075892857
20	9.20758928571429E-6	0.000863374255952381	0.00942696707589286

Table 21: OTFS MRC Detection (ETU - 80 km/h)

4.5.5 QPSK Results using mmWave

In the ETU model, we will increase the carrier frequency to 40 GHz and begin the simulation with a constellation size of 4 (QPSK), to simulate mmWave in high-propagation conditions. The results can be seen in the following figure.

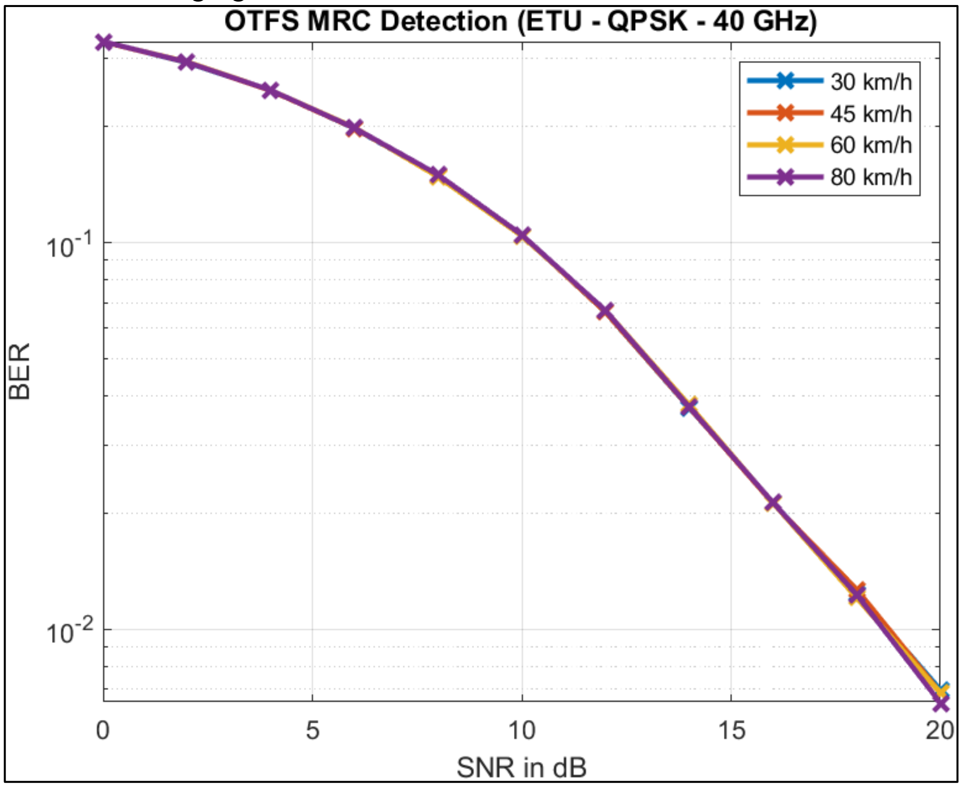

Figure 21: OTFS MRC Detection (ETU - QPSK - 40 GHz)

Similar to mmWave in mid-propagation conditions, the speeds are marginally equal. Compared to the EVA model simulation, the BER is very similar, though again, this comparison is not analogous. The values of the above figure can be seen in higher detail in the following table.

SNR (dB)	**Bit-Error Rate (BER)**			
	User Equipment Speed (km/h)			
	30	**45**	**60**	**80**
0	0.3311	0.3306	0.3308	0.3312
2	0.2924	0.2930	0.2938	0.2931
4	0.2478	0.2474	0.2480	0.2480
6	0.1975	0.1977	0.1986	0.1981
8	0.1491	0.1493	0.1480	0.1499
10	0.1048	0.1047	0.1041	0.1048
12	0.0660	0.0661	0.0667	0.0667
14	0.0374	0.0374	0.0380	0.0376
16	0.0213	0.0211	0.0212	0.0213
18	0.0124	0.0127	0.0120	0.0123
20	0.0070	0.0068	0.0068	0.0064

Table 22: OTFS MRC Detection (ETU - QPSK - 40 GHz)

4.5.6 8QAM Results using mmWave

We will repeat the previous simulation for a constellation size of 8 (8QAM). The results can be seen in the following figure.

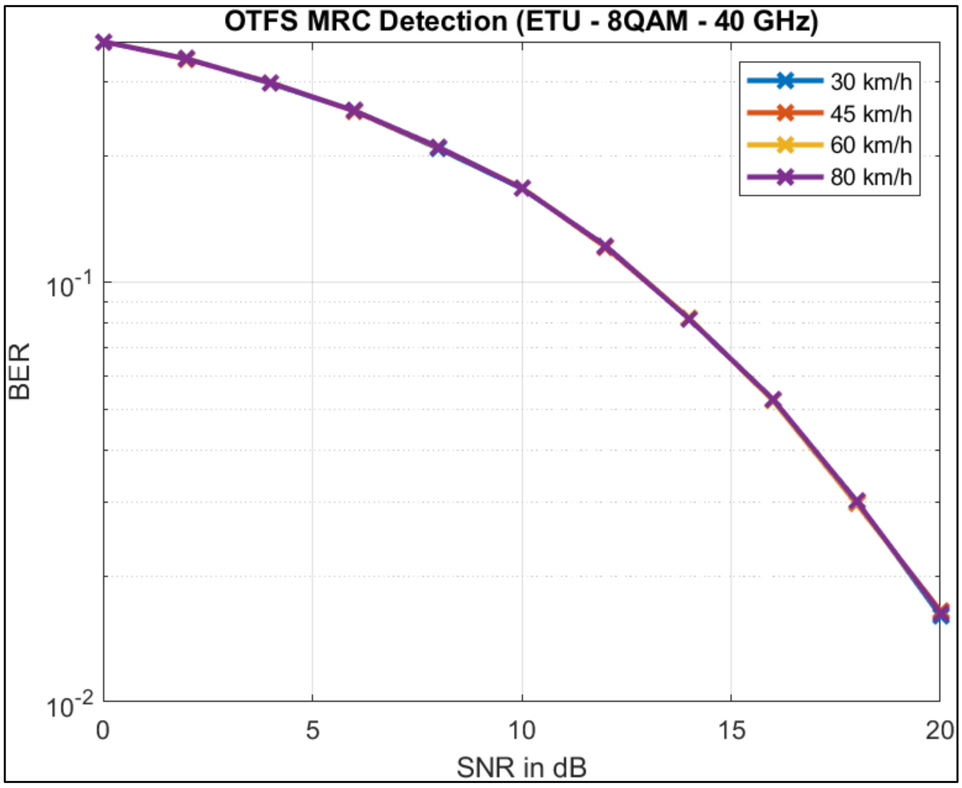

Figure 22: OTFS MRC Detection (ETU - 8QAM - 40 GHz)

Again, all the different speeds are marginally equal, similar to what we observed in the EVA model. The values of the above figure can be seen in higher detail in the table below.

SNR (dB)	**Bit-Error Rate (BER)**			
	User Equipment Speed (km/h)			
	30	**45**	**60**	**80**
0	0.3734	0.3734	0.3734	0.3735
2	0.3403	0.3392	0.3402	0.3398
4	0.2970	0.2986	0.2977	0.2981
6	0.2555	0.2553	0.2561	0.2558
8	0.2082	0.2092	0.2098	0.2100
10	0.1673	0.1679	0.1678	0.1674
12	0.1222	0.1213	0.1221	0.1222
14	0.0821	0.0821	0.0822	0.0818
16	0.0526	0.0523	0.0523	0.0528
18	0.0302	0.0298	0.0300	0.0302
20	0.0161	0.0166	0.0164	0.0164

Table 23: OTFS MRC Detection (ETU - 8QAM - 40 GHz)

4.5.7 16QAM Results using mmWave

Finally, we will repeat the previous simulations for a constellation size of 16 (16QAM). The results can be seen in the following figure.

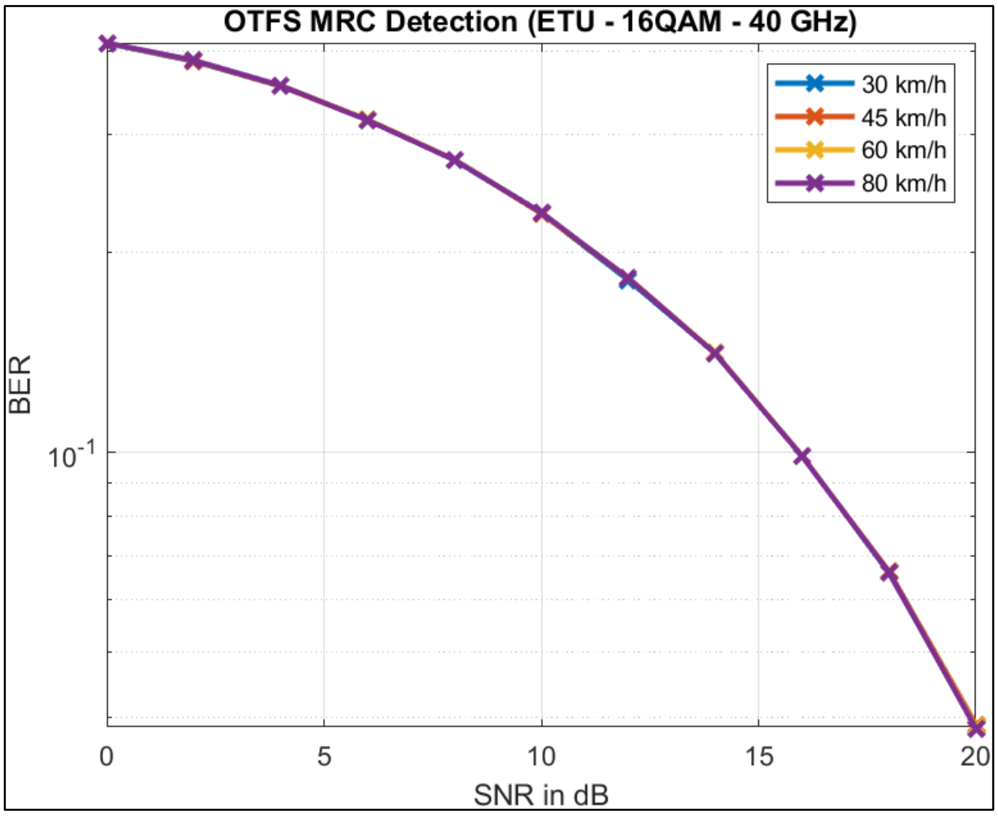

Figure 23: OTFS MRC Detection (ETU - 16QAM - 40 GHz)

As is to be expected, the speeds are marginally equal, posing stability throughout the increase of the SNR. The values of the above figure can be seen in the following table.

SNR (dB)	**Bit-Error Rate (BER)**			
	User Equipment Speed (km/h)			
	30	**45**	**60**	**80**
0	0.4119	0.4123	0.4124	0.4126
2	0.3870	0.3877	0.3880	0.3884
4	0.3551	0.3553	0.3560	0.3560
6	0.3163	0.3158	0.3163	0.3157
8	0.2749	0.2753	0.2755	0.2751
10	0.2288	0.2286	0.2291	0.2293
12	0.1815	0.1826	0.1830	0.1828
14	0.1410	0.1410	0.1411	0.1407
16	0.0989	0.0987	0.0988	0.0987
18	0.0659	0.0662	0.0660	0.0660
20	0.0386	0.0390	0.0386	0.0385

Table 24: OTFS MRC Detection (ETU - 16QAM - 40 GHz)

4.5.8 Result Comparison at Highest Speed using mmWave

Similar to the 4 GHz results, we will be combining the highest speeds into one figure, as shown below.

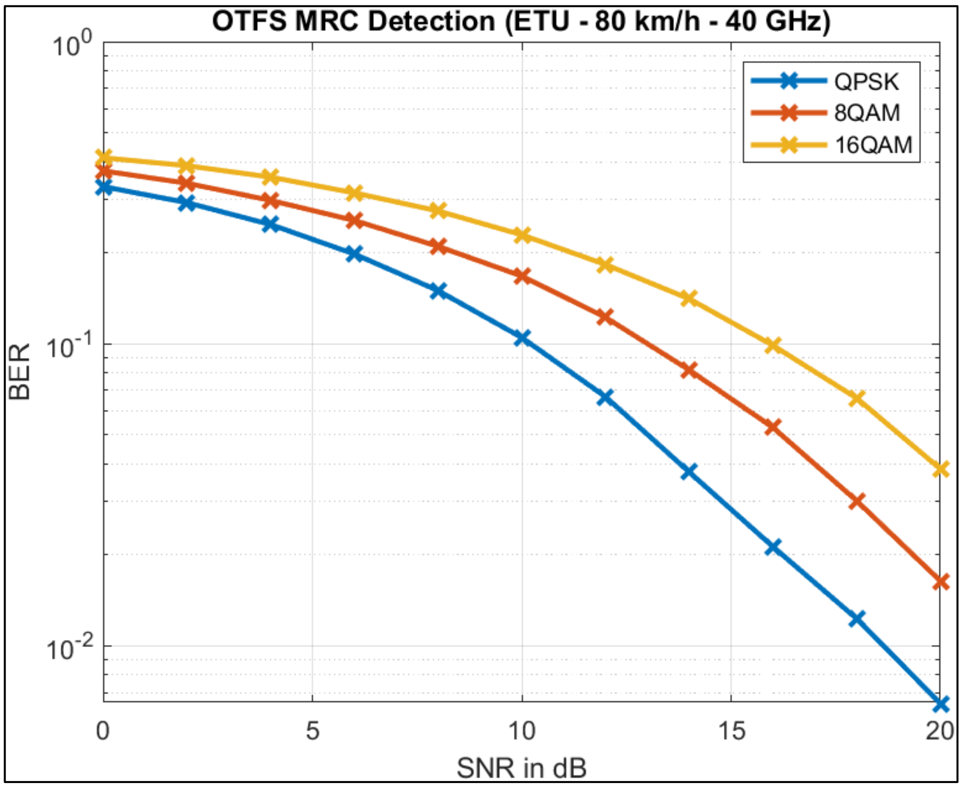

Figure 24: OTFS MRC Detection (ETU - 80 km/h - 40 GHz)

Comparing this figure with the equivalent for EVA, we can see that OTFS in high-propagation conditions performs roughly the same as in mid-propagation conditions while using mmWave carrier frequencies. Again, the comparison is not analogous but the speeds do represent real life scenarios that OTFS could be used in. The values of the above figure can be seen in the following table.

SNR (dB)	Bit-Error Rate (BER)		
	QPSK	8QAM	16QAM
0	0.331191685267857	0.373455171130952	0.412564871651786
2	0.293121512276786	0.339817150297619	0.388405343191964
4	0.247998883928571	0.298132719494048	0.356003208705357
6	0.198091936383929	0.255817429315476	0.315709612165179
8	0.149861188616071	0.210007719494048	0.275054338727679
10	0.104775111607143	0.167421781994048	0.229306361607143
12	0.0667088448660714	0.122199125744048	0.182797502790179
14	0.0375633370535714	0.0817979910714286	0.140716796875
16	0.0212678571428571	0.0527623697916667	0.0986843610491071
18	0.0122836216517857	0.0301697358630952	0.0660078822544643
20	0.00643708147321429	0.0163740699404762	0.0384674944196429

Table 25: OTFS MRC Detection (ETU - 80 km/h - 40 GHz)

4.5.9 Result Comparison of FR1 and mmWave at Highest Speed

As a final comparison, we will be combining the results for the highest speeds when using a 4 GHz carrier frequency and the results for the highest speeds when using a 40 GHz carrier frequency. The combined figure can be seen below.

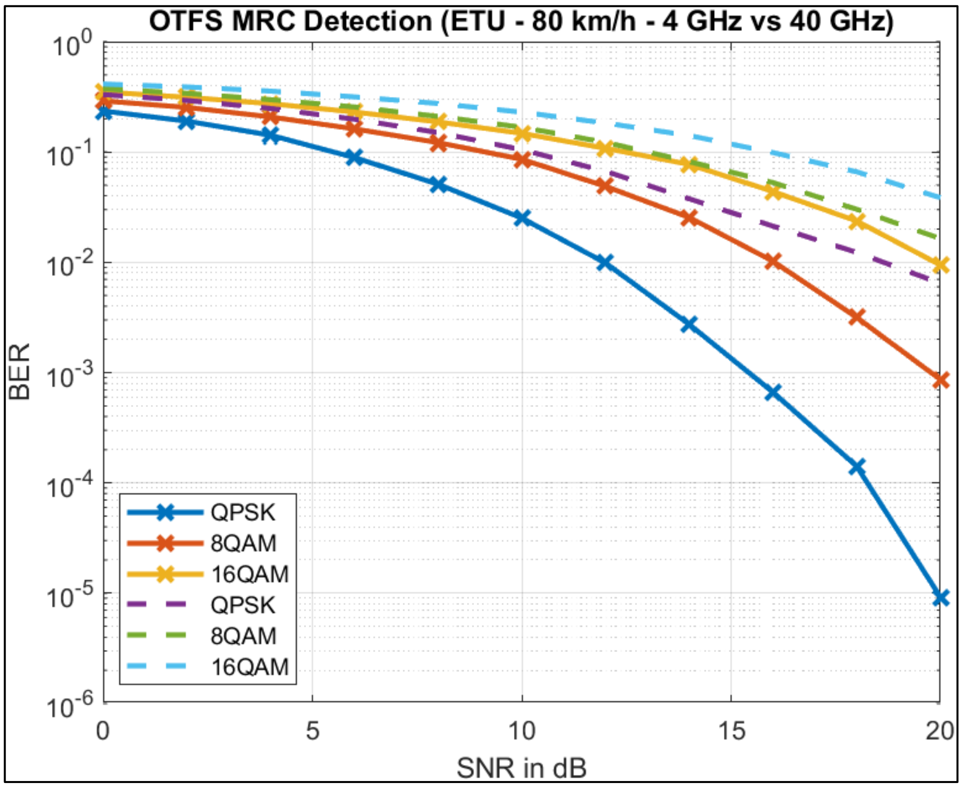

Figure 25: OTFS MRC Detection (ETU - 80 km/h - 4 GHz vs 40 GHz)

As expected, the QPSK and 8QAM using 4 GHz carriers outperform both 16QAM with 4 GHz carriers and all the constellation sizes using mmWave. An important remark to be made is the fact that 4 GHz 16QAM poses a very similar BER with 40 GHz 8QAM, while also having a higher BER than the 40 GHz QPSK. This was also noted in the EVA channel with similar results. The values can be seen in higher detail in the following table.

SNR (dB)	Bit-Error Rate (BER)					
	QPSK		8QAM		16QAM	
	4 GHz	40 GHz	4 GHz	40 GHz	4 GHz	40 GHz
0	0.2352	0.3312	0.2904	0.3735	0.3497	0.4126
2	0.1903	0.2931	0.2534	0.3398	0.3129	0.3884
4	0.1418	0.2480	0.2078	0.2981	0.2739	0.3560
6	0.0890	0.1981	0.1619	0.2558	0.2301	0.3157
8	0.0510	0.1499	0.1217	0.2100	0.1872	0.2751
10	0.0251	0.1048	0.0858	0.1674	0.1475	0.2293
12	0.0098	0.0667	0.0487	0.1222	0.1078	0.1828
14	0.0027	0.0376	0.0255	0.0818	0.0771	0.1407
16	0.0007	0.0213	0.0102	0.0528	0.0437	0.0987
18	0.0001	0.0123	0.0032	0.0302	0.0236	0.0660
20	9.208E-6	0.0064	0.0009	0.0164	0.0094	0.0385

Table 26: OTFS MRC Detection (ETU - 80 km/h - 4 GHz vs 40 GHz)

Chapter 5 Thesis Conclusion

5.1 Result Synopsis

The comparison between all the detectors that exist in OTFS's literature did yield interesting results. Out of the seven different algorithm implementations, MPA seemed to outperform the rest by a significant margin, with all of the MRC implementations performing roughly equally and with time-frequency single tap and the block-wise LMMSE implementation falling behind the competition by a significant margin. From the above result, we chose to continue into further simulations with the use of the MPA and MRC implementations (algorithm 4 and 2 respectively).

Both of the chosen implementations were simulated over ideal channel conditions, with equal symbol in time count and subcarrier number, with a 4 GHz carrier. As we saw, this resulted in a discrepancy between the comparison of all the detectors. Specifically, in the previous comparison, using QPSK, MPA outperformed MRC significantly while now, again using QPSK, MPA was significantly outperformed by MRC. The difference between the results might be caused due to the difference in frame count, with the comparison of all detector being 400 and the rest of the simulations being 1000. MPA seems to perform roughly equal to MRC when MPA is using QPSK and 8QAM and MRC is using 8QAM and 16QAM. This, of course, does impact the overall performance in the highest achievable transfer speeds if MPA is used, since a smaller constellation must be utilized in order to keep the BER low.

The MRC simulations did yield interesting results, on which a few important remarks can be made. The first important remark is the fact that OTFS, in both mid and high-propagation channel conditions seems stable in all speeds, meaning that regardless of Doppler frequencies, OTFS appears mostly unaffected. What is very interesting to point out is the fact that with more Doppler frequencies (i.e., high speeds) OTFS performed better than with less Doppler frequencies in the same channel conditions. What is also important to point out, is the fact that the results in both EVA and ETU channel models are very similar. While comparing the two models, as we already mentioned, is not the most accurate comparison due to the speed differences, they both represent real life scenarios where the chosen speeds are very plausible.

OTFS also proved to be able to maintain stability in all speeds with the use of mmWave frequencies (specifically 40 GHz). While decreased performance was expected, overall QPSK and 8QAM using 40 GHz frequencies performed very similarly to 16QAM using 4 GHz frequencies. Another important remark to be made for the use of mmWave frequencies is that unlike FR1, where the BER was reducing as the speed increased, the BER remains marginally equal regardless of speed. This can be seen in both channel models. It is important to note that all of the simulations can have slightly different results due to the randomness of the input signals, that are generated before each simulation is ran.

5.2 Personal Opinion

While OTFS is still a relatively new technology, in my opinion it does show promising results for usage in applications where high error tolerance (like mission critical applications) must be present. An example where both high error tolerance and high speeds are present is an autonomous vehicle network, where fast moving cars must quickly and reliably relay information such as location, speed, direction and intent (changing lanes, moving to exits etc) to other nearby vehicles. While there are a few points that can be improved, as of now OTFS is a strong candidate for B5G systems and could be used as an overlay (if not completely replace) for OFDM systems.

5.3 Future Work Recommendations

There are many research topics that OTFS should be a part of. While we did simulate OTFS in two different condition models in various speeds, many of the results should be refined and examined further, in order to maximize the possible BER and have definitive answers on whether OTFS should be considered a candidate. A few of the key topics that could be covered are the following:

- **OTFS Simulations using Massive MIMO/Smart Antennas**: There are continuous efforts in the development and enhancement of massive MIMO and smart antennas in order to decrease the BER in high-propagation channels. Specifically, these technologies have proven to be highly effective where many echoes (or copies) of the signals are received at one time. This can further improve OTFS's performance in high-propagation channels.
- **OTFS Simulations using Different Detector Algorithms**: In this thesis we examined an OTFS implementation that specifically used the MRC detection method. While indeed MRC proved to be more effective than the other algorithms, further examination of the other algorithms should be conducted and compared. With continuous research, new detection algorithms that are better than MRC could be developed, reducing the BER and increasing the performance of the overall system more.
- **OTFS Physical Implementation and Use in an Autonomous Vehicle Network Prototype**: As we have mentioned, OTFS appears to be an ideal candidate for communication between equipment traveling at high speeds. With the simulations proving to provide a relatively reliable connection, a physical prototype of a mission critical application, such as an Autonomous Vehicle network could be created and tested, providing real life information about OTFS's capabilities for such applications.
- **OTFS and OFDM Direct Comparison**: While new technologies, such as Massive MIMO and Smart Antennas are continuously optimized, a more direct comparison between OTFS and OFDM using these technologies should take place, to prove that OTFS still provides higher robustness in high-speed conditions. Along with those technologies, with the creation of new algorithms, new simulations should be conducted and be compared against current knowledge.

All in all, there are many topics that OTFS should participate in before it is concluded that it could eventually replace OFDM systems or be used as an overlay in said systems. With continuous research, OTFS will surely be a candidate to replace the current systems for generations B5G.

Chapter 6 Appendices

Appendix A: OTFS_Sample_Code Package

A.1 OTFS_sample_code.m

```
%
% Copyright (c) 2018, Raviteja Patchava, Yi Hong, and Emanuele Viterbo,
Monash University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%    list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%    this list of conditions and the following disclaimer in the
documentation
%    and/or other materials provided with the distribution.
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%     - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%     - Freely distributed for educational and research purposes
%%

clc
clear all
close all
tic
%% OTFS parameters%%%%%%%%%%
% number of symbol
N = 64; %64 min
% number of subcarriers
M = 64; %64 min
% size of constellation
M_mod = 4;
M_bits = log2(M_mod);
% average energy per data symbol
eng_sqrt = (M_mod==2)+(M_mod~=2)*sqrt((M_mod-1)/6*(2^2));
% number of symbols per frame
N_syms_perfram = N*M;
```

```
% number of bits per frame
N_bits_perfram = N*M*M_bits;

SNR_dB = 0:2:20;
SNR = 10.^(SNR_dB/10);
noise_var_sqrt = sqrt(1./SNR);
sigma_2 = abs(eng_sqrt*noise_var_sqrt).^2;
%%
rng(1)
N_fram = 10^3; %^4
err_ber = zeros(length(SNR_dB),1);
for iesn0 = 1:length(SNR_dB)
    for ifram = 1:N_fram
        %% random input bits generation%%%%%
        data_info_bit = randi([0,1],N_bits_perfram,1);
        data_temp = bi2de(reshape(data_info_bit,N_syms_perfram,M_bits));
        x = qammod(data_temp,M_mod,'gray'); %removed initial phase (not
supported anymore)
        x = reshape(x,N,M);

        %% OTFS modulation%%%%
        s = OTFS_modulation(N,M,x);

        %% OTFS channel generation%%%%
        [taps,delay_taps,Doppler_taps,chan_coef] = OTFS_channel_gen(N,M);

        %% OTFS channel output%%%%%
        r =
OTFS_channel_output(N,M,taps,delay_taps,Doppler_taps,chan_coef,sigma_2(iesn
0),s);

        %% OTFS demodulation%%%%
        y = OTFS_demodulation(N,M,r);

        %% message passing detector%%%%
        x_est =
OTFS_mp_detector(N,M,M_mod,taps,delay_taps,Doppler_taps,chan_coef,sigma_2(i
esn0),y);

        %% output bits and errors count%%%%%
        data_demapping = qamdemod(x_est,M_mod,'gray'); %removed initial
phase (not supported anymore)
        data_info_est =
reshape(de2bi(data_demapping,M_bits),N_bits_perfram,1);
        errors = sum(xor(data_info_est,data_info_bit));
        err_ber(iesn0) = errors + err_ber(iesn0)
        ifram
    end
end
err_ber_fram = err_ber/N_bits_perfram./N_fram
semilogy(SNR_dB, err_ber_fram,'-*','LineWidth',2);
title(sprintf('OTFS'))
ylabel('BER'); xlabel('SNR in dB');grid on

toc
```

A.2 OTFS_mp_detector.m

```
%
% Copyright (c) 2018, Raviteja Patchava, Yi Hong, and Emanuele Viterbo,
Monash University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%    list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%    this list of conditions and the following disclaimer in the
documentation
%    and/or other materials provided with the distribution.
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%     - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%     - Freely distributed for educational and research purposes
%%
function x_est =
OTFS_mp_detector(N,M,M_mod,taps,delay_taps,Doppler_taps,chan_coef,sigma_2,y
)

yv = reshape(y,N*M,1);
n_ite = 200;
delta_fra = 0.6;
alphabet = qammod(0:M_mod-1,M_mod,'gray'); %removed initial phase (not
supported anymore)

mean_int = zeros(N*M,taps);
var_int = zeros(N*M,taps);
p_map = ones(N*M,taps,M_mod)*(1/M_mod);

conv_rate_prev = -0.1;
for ite=1:n_ite
    %% Update mean and var
    for ele1=1:1:M
        for ele2=1:1:N
            mean_int_hat = zeros(taps,1);
            var_int_hat = zeros(taps,1);
            for tap_no=1:taps
                m = ele1-1-delay_taps(tap_no)+1;
                add_term = exp(1i*2*(pi/M)*(m-1)*(Doppler_taps(tap_no)/N));
                add_term1 = 1;
                if ele1-1<delay_taps(tap_no)
                    n = mod(ele2-1-Doppler_taps(tap_no),N) + 1;
                    add_term1 = exp(-1i*2*pi*((n-1)/N));
                end
                new_chan = add_term * (add_term1) * chan_coef(tap_no);

                for i2=1:1:M_mod
                    mean_int_hat(tap_no) = mean_int_hat(tap_no) +
p_map(N*(ele1-1)+ele2,tap_no,i2) * alphabet(i2);
                    var_int_hat(tap_no) = var_int_hat(tap_no) +
p_map(N*(ele1-1)+ele2,tap_no,i2) * abs(alphabet(i2))^2;
                end
                mean_int_hat(tap_no) = mean_int_hat(tap_no) * new_chan;
                var_int_hat(tap_no) = var_int_hat(tap_no) *
abs(new_chan)^2;
                var_int_hat(tap_no) = var_int_hat(tap_no) -
abs(mean_int_hat(tap_no))^2;
            end

            mean_int_sum = sum(mean_int_hat);
            var_int_sum = sum(var_int_hat)+(sigma_2);

            for tap_no=1:taps
                mean_int(N*(ele1-1)+ele2,tap_no) = mean_int_sum -
mean_int_hat(tap_no);
                var_int(N*(ele1-1)+ele2,tap_no) = var_int_sum -
var_int_hat(tap_no);
            end

        end
    end
    %% Update probabilities
    sum_prob_comp = zeros(N*M,M_mod);
    dum_eff_ele1 = zeros(taps,1);
    dum_eff_ele2 = zeros(taps,1);
    for ele1=1:1:M
        for ele2=1:1:N
            dum_sum_prob = zeros(M_mod,1);
            log_te_var = zeros(taps,M_mod);
            for tap_no=1:taps

                if ele1+delay_taps(tap_no)<=M
                    eff_ele1 = ele1 + delay_taps(tap_no);
                    add_term = exp(1i*2*(pi/M)*(ele1-
1)*(Doppler_taps(tap_no)/N));
                    int_flag = 0;
                else
                    eff_ele1 = ele1 + delay_taps(tap_no)- M;
                    add_term = exp(1i*2*(pi/M)*(ele1-1-
M)*(Doppler_taps(tap_no)/N));
                    int_flag = 1;
                end
                add_term1 = 1;
                if int_flag==1
                    add_term1 = exp(-1i*2*pi*((ele2-1)/N));
                end
                eff_ele2 = mod(ele2-1+Doppler_taps(tap_no),N) + 1;
                new_chan = add_term * add_term1 * chan_coef(tap_no);

                dum_eff_ele1(tap_no) = eff_ele1;
                dum_eff_ele2(tap_no) = eff_ele2;
                for i2=1:1:M_mod
                    dum_sum_prob(i2) = abs(yv(N*(eff_ele1-1)+eff_ele2)-
mean_int(N*(eff_ele1-1)+eff_ele2,tap_no) - new_chan * alphabet(i2))^2;
                    dum_sum_prob(i2)= -
(dum_sum_prob(i2)/var_int(N*(eff_ele1-1)+eff_ele2,tap_no));
                end
                dum_sum = dum_sum_prob - max(dum_sum_prob);
                dum1 = sum(exp(dum_sum));
                log_te_var(tap_no,:) = dum_sum - log(dum1);
            end
            for i2=1:1:M_mod
                ln_qi(i2) = sum(log_te_var(:,i2));
            end
            dum_sum = exp(ln_qi - max(ln_qi));
            dum1 = sum(dum_sum);
            sum_prob_comp(N*(ele1-1)+ele2,:) = dum_sum/dum1;
            for tap_no=1:1:taps
                eff_ele1 = dum_eff_ele1(tap_no);
                eff_ele2 = dum_eff_ele2(tap_no);

                dum_sum = log_te_var(tap_no,:);
                ln_qi_loc = ln_qi - dum_sum;
                dum_sum = exp(ln_qi_loc - max(ln_qi_loc));
                dum1 = sum(dum_sum);
                p_map(N*(eff_ele1-1)+eff_ele2,tap_no,:) =
(dum_sum/dum1)*delta_fra + (1-delta_fra)*reshape(p_map(N*(eff_ele1-
1)+eff_ele2,tap_no,:),1,M_mod);
            end

        end
    end
    conv_rate =  sum(max(sum_prob_comp,[],2)>0.99)/(N*M);
    if conv_rate==1
        sum_prob_fin = sum_prob_comp;
        break;
    elseif conv_rate > conv_rate_prev
        conv_rate_prev = conv_rate;
        sum_prob_fin = sum_prob_comp;
    elseif (conv_rate < conv_rate_prev - 0.2) && conv_rate_prev > 0.95
        break;
    end
end
x_est = zeros(N,M);
for ele1=1:1:M
    for ele2=1:1:N
        [~,pos] = max(sum_prob_fin(N*(ele1-1)+ele2,:));
        x_est(ele2,ele1) = alphabet(pos);
    end
end
end
```

A.3 OTFS_modulation.m

```
%
% Copyright (c) 2018, Raviteja Patchava, Yi Hong, and Emanuele Viterbo,
Monash University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%    list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%    this list of conditions and the following disclaimer in the
documentation
%    and/or other materials provided with the distribution.
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%     - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%     - Freely distributed for educational and research purposes
%%
function s = OTFS_modulation(N,M,x)
%% OTFS Modulation: 1. ISFFT, 2. Heisenberg transform
X = fft(ifft(x).').'/sqrt(M/N); %%%ISFFT
s_mat = ifft(X.')*sqrt(M); % Heisenberg transform
s = s_mat(:);
end
```

A.4 OTFS_demodulation.m

```
%
% Copyright (c) 2018, Raviteja Patchava, Yi Hong, and Emanuele Viterbo,
Monash University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%    list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%    this list of conditions and the following disclaimer in the
documentation
%    and/or other materials provided with the distribution.
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%     - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%     - Freely distributed for educational and research purposes
%%
function y = OTFS_demodulation(N,M,r)
%% OTFS demodulation: 1. Wiegner transform, 2. SFFT
r_mat = reshape(r,M,N);
Y = fft(r_mat)/sqrt(M); % Wigner transform
Y = Y.';
y = ifft(fft(Y).').'/sqrt(N/M); % SFFT
end
```

A.5 OTFS_channel_gen.m

```
%
% Copyright (c) 2018, Raviteja Patchava, Yi Hong, and Emanuele Viterbo,
Monash University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%    list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%    this list of conditions and the following disclaimer in the
documentation
%    and/or other materials provided with the distribution.
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%     - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%     - Freely distributed for educational and research purposes
%%
function [taps,delay_taps,Doppler_taps,chan_coef] = OTFS_channel_gen(N,M)
%% Channel for testing%%%%%
%channel with 4 taps of uniform power%%%
taps = 4;
delay_taps = [0 1 2 3];
Doppler_taps = [0 1 2 3];
pow_prof = (1/taps) * (ones(1,taps));
chan_coef = sqrt(pow_prof).*(sqrt(1/2) * (randn(1,taps)+1i*randn(1,taps)));
%%%%%%%%%%%%%%%%%%%%

end
```

A.6 OTFS_channel_output.m

```
%
% Copyright (c) 2018, Raviteja Patchava, Yi Hong, and Emanuele Viterbo,
Monash University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%%
function r =
OTFS_channel_output(N,M,taps,delay_taps,Doppler_taps,chan_coef,sigma_2,s)
%% wireless channel and noise
L = max(delay_taps);
s = [s(N*M-L+1:N*M);s];%add one cp
s_chan = 0;
for itao = 1:taps
    s_chan = s_chan+chan_coef(itao)*circshift([s.*exp(1j*2*pi/M ...
        *(-L:-L+length(s)-
1)*Doppler_taps(itao)/N).';zeros(delay_taps(end),1)],delay_taps(itao));
end
noise = sqrt(sigma_2/2)*(randn(size(s_chan)) + 1i*randn(size(s_chan)));
r = s_chan + noise;
r = r(L+1:L+(N*M));%discard cp
end
```

Appendix B: ZP_OTFS_MRC_Detection Package

B.1 ZP_OTFS_MRC_system.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%    list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%    this list of conditions and the following disclaimer in the
documentation
%    and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%    publication.
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%     - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%     - Freely distributed for educational and research purposes

%   [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%   [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%   [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285
```

```
close all
clear all
rng(1)
%% OTFS parameters%%%%%%%%%%
% N: number of symbols in time
N = 64;
% M: number of subcarriers in frequency
M = 64;
% M_mod: size of QAM constellation
M_mod = 16;
M_bits = log2(M_mod);
% average energy per data symbol
eng_sqrt = (M_mod==2)+(M_mod~=2)*sqrt((M_mod-1)/6*(2^2));

%% delay-Doppler grid symbol placement
% ZP length  should be set to greater than or equal to maximum value
% of delay_taps
length_ZP = M/8;
% data positions of OTFS delay-Doppler domain data symbols  in the 2-D grid
M_data = M-length_ZP;
data_grid=zeros(M,N);
data_grid(1:M_data,1:N)=1;
% number of symbols per frame
N_syms_perfram = sum(sum(data_grid));
% number of bits per frame
N_bits_perfram = N_syms_perfram*M_bits;



% Time and frequency resources
car_fre = 40*10^9;% Carrier frequency
delta_f = 1.5*10^3; % subcarrier spacing: 15 KHz
T = 1/delta_f; %one time symbol duration in OTFS frame

% SNR and variance of the noise
% SNR = P/\sigma^2; P: avg. power of albhabet transmitted
SNR_dB = 0:2:20;
SNR = 10.^(SNR_dB/10);
sigma_2 = (abs(eng_sqrt)^2)./SNR;



%% Initializing simulation error count variables

N_fram = 1000; %1000
est_info_bits_MRC=zeros(N_bits_perfram,1);
err_ber_MRC = zeros(1,length(SNR_dB));
no_of_detetor_iterations_MRC= zeros(length(SNR_dB),1);
avg_no_of_iterations_MRC=zeros(1,length(SNR_dB));


%% Normalized DFT matrix
Fn=dftmtx(N);  % Generate the DFT matrix
Fn=Fn./norm(Fn);  % normalize the DFT matrix
current_frame_number=zeros(1,length(SNR_dB));

for iesn0 = 1:length(SNR_dB)
    for ifram = 1:N_fram
        current_frame_number(iesn0)=ifram;
        %% random input bits generation%%%%%
        trans_info_bit = randi([0,1],N_syms_perfram*M_bits,1);
        %%2D QAM symbols generation %%%%%%%%
        data=qammod(reshape(trans_info_bit,M_bits,N_syms_perfram),
M_mod,'gray','InputType','bit');
        X = Generate_2D_data_grid(N,M,data,data_grid);


        %% OTFS modulation%%%%
        X_tilda=X*Fn';                      %equation (2) in [R1]
        s = reshape(X_tilda,N*M,1);         %equation (4) in [R1]


        %% OTFS channel generation%%%% The user can use either the
synthetic channel model or the 3GPP channel by uncommenting the
corresonding piece of code.
        %% synthetic channel model with equal power paths with delays
[0,l_max] and Dopplers [-k_max,k_max]
                % taps=4;
                % %l_max=delay_spread;
                % l_max=4;
                % k_max=4;
                % chan_coef=1/sqrt(2)*(randn(1,taps)+1i.*randn(1,taps));
                % delay_taps=randi(taps,[1,l_max+1]);
                % delay_taps=sort(delay_taps-min(delay_taps));  %% random
delay shifts in the range [0,l_max]
                % Doppler_taps=k_max-2*k_max*rand(1,taps);   %% uniform
Doppler profile [-k_max,k_max]

        % channel model following 3GPP standard
         max_speed=80;  % km/hr

[chan_coef,delay_taps,Doppler_taps,taps]=Generate_delay_Doppler_channel_par
ameters(N,M,car_fre,delta_f,T,max_speed);
         L_set=unique(delay_taps);

gs=Gen_discrete_time_channel(N,M,taps,delay_taps,Doppler_taps,chan_coef);
%equation (14) in [R1]

        %% channel output%%%%%
        r=zeros(N*M,1);
        l_max=max(delay_taps);
        for q=1:N*M
            for l=(L_set+1)
                if(q>=l)
                    r(q)=r(q)+gs(l,q)*s(q-l+1);  %equation (18) in [R1]
                end
            end
        end
        noise= sqrt(sigma_2(iesn0)/2)*(randn(size(s)) + 1i*randn(size(s)));
        r=r+noise;

        %% OTFS demodulation%%%%
        Y_tilda=reshape(r,M,N);             %equation (5) in [R1]
        Y = Y_tilda*Fn;                     %equation (6) in [R1]

        %% Generate delay-time channel vectors from the time domain
channel.
        [nu_ml_tilda]=Gen_delay_time_channel_vectors(N,M,l_max,gs);  %
equation (42) in [R1]

        %% Generate block-wise time-frequency domain channel
        [H_tf]=Generate_time_frequency_channel_ZP(N,M,gs,L_set);
        %% MRC delay-time detection in [R1,R2,R3]

        n_ite_MRC=50; % maximum number of MRC detector iterations
        %damping parameter - reducing omega improves error performance at
the cost of increased detector iterations
        omega=1;
        if(M_mod==64)
            omega=0.25;     % set omega to a smaller value (for example:
0.05) for modulation orders greater than 64-QAM
        end
        decision=1;         %1-hard decision, 0-soft decision
        init_estimate=1;    %1-use the TF single tap estimate as the
initial estimate for MRC detection, 0-initialize the symbol estimates to 0
at the start of MRC iteration
        %(Note: it is recommended to set init_estimate to 0 for higher
order modulation schemes like 64-QAM or 256-QAM as the single tap equalizer
estimate may be less accurate)

        [est_info_bits_MRC,det_iters_MRC,data_MRC] =
MRC_delay_time_detector(N,M,M_data,M_mod,sigma_2(iesn0),data_grid,r,H_tf,nu
_ml_tilda,L_set,omega,decision,init_estimate,n_ite_MRC);

        %% errors count%%%%%
        errors_MRC = sum(xor(est_info_bits_MRC,trans_info_bit));
        err_ber_MRC(iesn0) = err_ber_MRC(iesn0) + errors_MRC;

avg_ber_MRC(iesn0)=err_ber_MRC(iesn0).'/length(trans_info_bit)/ifram;


        %%  iterations count

no_of_detetor_iterations_MRC(iesn0)=no_of_detetor_iterations_MRC(iesn0)+det
_iters_MRC;

avg_no_of_iterations_MRC(iesn0)=no_of_detetor_iterations_MRC(iesn0)/ifram;


        %% DISP error performance details
         clc

disp('####################################################################'
)
        fprintf('ZP-OTFS-(N,M,QAM size)');disp([N,M,M_mod]);
        display(SNR_dB,'SNR (dB)');
        display(current_frame_number,'Number of frames');
        display(avg_ber_MRC,'Average BER - Delay-time domain Maximal Ratio
Combining (MRC)');
        display(avg_no_of_iterations_MRC,'Average number of iterations for
the delay-time domain MRC detector');

disp('####################################################################'
)

    end

end

figure(1)
semilogy(SNR_dB,avg_ber_MRC,'-x','LineWidth',2,'MarkerSize',8)
title(sprintf('OTFS MRC Detection'))
```

```
legend('MRC (Algorithm 2 in [R1])')
grid on
xlabel('SNR in dB')
ylabel('BER')
```

B.2 ZP_OTFS_compare_detectors.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285
```

```
close all
clear all
rng(1)
%% OTFS parameters%%%%%%%%%%
% N: number of symbols in time
N = 64;
% M: number of subcarriers in frequency
M = 64;
% M_mod: size of QAM constellation
M_mod = 4;
M_bits = log2(M_mod);
% average energy per data symbol
eng_sqrt = (M_mod==2)+(M_mod~=2)*sqrt((M_mod-1)/6*(2^2));

%% delay-Doppler grid symbol placement
% ZP length  should be set to greater than or equal to maximum value
% of delay_taps
length_ZP = M/16;
% data positions of OTFS delay-Doppler domain data symbols  in the 2-D grid
M_data = M-length_ZP;
data_grid=zeros(M,N);
data_grid(1:M_data,1:N)=1;
% number of symbols per frame
N_syms_perfram = sum(sum(data_grid));
% number of bits per frame
N_bits_perfram = N_syms_perfram*M_bits;



% Time and frequency resources
car_fre = 4*10^9;% Carrier frequency
delta_f = 15*10^3; % subcarrier spacing: 15 KHz
T = 1/delta_f; %one time symbol duration in OTFS frame

% SNR and variance of the noise
% SNR = P/\sigma^2; P: avg. power of albhabet transmitted
SNR_dB = 0:2:20;
SNR = 10.^(SNR_dB/10);
sigma_2 = (abs(eng_sqrt)^2)./SNR;



%% Initializing simulation error count variables

N_fram = 100; %1000
Initialize_error_count_variables;

%% Normalized DFT matrix
Fn=dftmtx(N);  % Generate the DFT matrix
Fn=Fn./norm(Fn);  % normalize the DFT matrix
current_frame_number=zeros(1,length(SNR_dB));

for iesn0 = 1:length(SNR_dB)
    for ifram = 1:N_fram
        current_frame_number(iesn0)=ifram;
        %% random input bits generation%%%%%
        trans_info_bit = randi([0,1],N_syms_perfram*M_bits,1);
        %%2D QAM symbols generation %%%%%%%%
        data=qammod(reshape(trans_info_bit,M_bits,N_syms_perfram),
M_mod,'gray','InputType','bit');
        X = Generate_2D_data_grid(N,M,data,data_grid);
        %% OTFS modulation%%%%
        X_tilda=X*Fn';
        s = reshape(X_tilda,N*M,1);


        %% OTFS channel generation%%%% The user can use either the
synthetic channel model or the 3GPP channel by uncommenting the
corresonding piece of code.
        %% synthetic channel model with equal power paths with delays
[0,l_max] and Dopplers [-k_max,k_max]
%          taps=4;
%          l_max=delay_spread;
%          k_max=4;
%          chan_coef=1/sqrt(2)*(randn(1,taps)+1i.*randn(1,taps));
%          delay_taps=randi(taps,[1,l_max+1]);
%          delay_taps=sort(delay_taps-min(delay_taps));  %% random delay
shifts in the range [0,l_max]
%          Doppler_taps=k_max-2*k_max*rand(1,taps);   %% uniform Doppler
profile [-k_max,k_max]
%          L_set=unique(delay_taps);
        %% channel model following 3GPP standard
        max_speed=500;  % km/hr

[chan_coef,delay_taps,Doppler_taps,taps]=Generate_delay_Doppler_channel_par
ameters(N,M,car_fre,delta_f,T,max_speed);
        L_set=unique(delay_taps);



        %% channel output%%%%%

[G,gs]=Gen_time_domain_channel(N,M,taps,delay_taps,Doppler_taps,chan_coef);
        [H,H_tilda,P]= Gen_DD_and_DT_channel_matrices(N,M,G,Fn);
        r=zeros(N*M,1);
        noise= sqrt(sigma_2(iesn0)/2)*(randn(size(s)) + 1i*randn(size(s)));
        l_max=max(delay_taps);
        for q=1:N*M
            for l=(L_set+1)
                if(q>=l)
                    r(q)=r(q)+gs(l,q)*s(q-l+1);
                end
            end
        end
        r=r+noise;

        %% OTFS demodulation%%%%
        Y_tilda=reshape(r,M,N);
        Y = Y_tilda*Fn;
        y=reshape(Y.',N*M,1);

        %% test: the received time domain signal can be generated element
in the matrix form (using r=G.s+noise).
        %          r_test=G*s+noise;
        %          test_delay_time_matrix_error=norm(r_test-r)
        %% test: the received DD domain signal can be generated in the
matrix form (using y=H.x+noise).
        %          noise_DD=kron(eye(M),Fn)*P'*noise;
        %          x=reshape(X.',N*M,1);
        %          y_test=H*x_vec+noise_DD;
        %         text_delay_Doppler_matrix_error=norm(y_test-y)

        %% Generate delay-time and delay-Doppler channel vectors from the
time domain channel.
%         [nu_ml_tilda]=Gen_delay_time_channel_vectors(N,M,l_max,gs);

[nu_ml_tilda,nu_ml,K_ml]=Gen_DT_and_DD_channel_vectors(N,M,L_set,gs);

        %% Generate block-wise time-frequency domain channel
        [H_tf]=Generate_time_frequency_channel_ZP(N,M,gs,L_set);

        %% Different detection methods

        n_ite_MRC=15; % maximum number of MRC detector iterations
(Algorithm 2 in [R1])
        n_ite_algo3=15; % maximum number of matched filtered Guass Seidel
(Algorithm 3 in [R1])
        n_ite_MPA=15; % maximum number of MPA detector iterations
        %damping parameter - reducing omega improves error performance at
the cost of increased detector iterations
        omega=1;
        if(M_mod>=64)
            omega=0.25;     % set omega to a smaller value (for example:
0.05) for modulation orders greater than 64-QAM
        end
        decision=1; %1-hard decision, 0-soft decision
        init_estimate=1; %1-use the TF single tap estimate as the initial
estimate for MRC detection, 0-initialize the symbol estimates to 0 at the
start of MRC iteration
        %(Note: it is recommended to set init_estimate to 0 for higher
order modulation schemes like 64-QAM as the single tap equalizer estimate
may be less accurate)

        %MRC detectors in [R1]

        [est_info_bits_MRC,det_iters_MRC,data_MRC] =
MRC_delay_time_detector(N,M,M_data,M_mod,sigma_2(iesn0),data_grid,r,H_tf,nu
_ml_tilda,L_set,omega,decision,init_estimate,n_ite_MRC);
        [est_info_bits_Algo1,det_iters_Algo1,data_Algo1]=
Algorithm1_detector(N,M,M_data,M_mod,sigma_2(iesn0),data_grid,r,H_tf,L_set,
omega,decision,init_estimate,n_ite_MRC,K_ml,Y);
        [est_info_bits_Algo3,det_iters_Algo3,data_Algo3] =
Algorithm3_detector(N,M,M_mod,sigma_2(iesn0),data_grid,r,G,omega,decision,i
nit_estimate,n_ite_algo3);

[est_info_bits_Algo3_low_complexity,det_iters_Algo3_low_complexity,data_Alg
o3_low_complexity] =
Algorithm3_low_complexity_detector(N,M,M_mod,sigma_2(iesn0),data_grid,r,H_t
f,gs,L_set,omega,decision,init_estimate,n_ite_algo3);

        % Other detectors in the literature

        [est_info_bits_MPA,det_iters_MPA,data_MPA] =
MPA_detector(N,M,M_mod,sigma_2(iesn0),data_grid,y,H,n_ite_MPA);
        [est_info_bits_1tap,data_1tap] =
TF_single_tap_equalizer(N,M,M_mod,sigma_2(iesn0),data_grid,Y,H_tf);
        [est_info_bits_LMMSE,data_LMMSE] =
Block_LMMSE_detector(N,M,M_mod,sigma_2(iesn0),data_grid,r,gs,L_set);


        %% errors count%%%%
        count_errors_per_frame;


        %% DISP error performance details
        display_errors_per_frame;



    end

end

figure(1)
semilogy(SNR_dB,avg_ber_Algo1,'-x','LineWidth',2,'MarkerSize',8)
hold on
semilogy(SNR_dB,avg_ber_MRC,'-x','LineWidth',2,'MarkerSize',8)
hold on
semilogy(SNR_dB,avg_ber_Algo3,'-x','LineWidth',2,'MarkerSize',8)
hold on
semilogy(SNR_dB,avg_ber_Algo3_low_complexity,'-
x','LineWidth',2,'MarkerSize',8)
hold on
semilogy(SNR_dB,avg_ber_MPA,'-x','LineWidth',2,'MarkerSize',8)
hold on
semilogy(SNR_dB,avg_ber_1tap,'-x','LineWidth',2,'MarkerSize',8)
hold on
semilogy(SNR_dB,avg_ber_LMMSE,'-s','LineWidth',2,'MarkerSize',8)
legend('MRC (Algorithm 1 in [R1])','MRC (Algorithm 2 in [R1])','Algorithm 3
in [R1]','low-complexity implementation of Algorithm 3 in
[R1]','MPA','time-freq single tap in [R3]','block-wise time-domain LMMSE in
[R3]')
grid on
xlabel('SNR in dB')
ylabel('BER')
```

B.3 Algorithm1_detector.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285
```

```matlab
function [est_bits,ite,x_data] =
Algorithm1_detector(N,M,M_data,M_mod,no,data_grid,r,H_tf,L_set,omega,decisi
on,init_estimate,n_ite_MRC,K_ml,Y)
%% Normalized DFT matrix
Fn=dftmtx(N);  % Generate the DFT matrix
Fn=Fn./norm(Fn);  % normalize the DFT matrix
%% Initial assignments
%Number of symbols per frame
N_syms_perfram=sum(sum((data_grid>0)));
%Arranging the delay-Doppler grid symbols into an array
data_array=reshape(data_grid,1,N*M);
%finding position of data symbols in the array
[~,data_index]=find(data_array>0);
%number of bits per QAM symbol
M_bits=log2(M_mod);
%number of bits per frame
N_bits_perfram = N_syms_perfram*M_bits;
%received delay-time samples
Y_tilda=reshape(r,M,N);
M_prime=M_data;
L_set=L_set+1; % Since matlab cannot work with zero indices
%% initial estimate using single tap TF equalizer
if(init_estimate==1)
    Y_tf=fft(Y_tilda).'; % delay-time to frequency-time domain
% equation (63) in [R1]
    X_tf=conj(H_tf).*Y_tf./(H_tf.*conj(H_tf)+no); % single tap equalizer
% equation (64) in [R1]
    X_est = ifft(X_tf.')*Fn; % SFFT
% equation (65) in [R1]
    X_est=qammod(qamdemod(X_est,M_mod,'gray'),M_mod,'gray');
    X_est=X_est.*data_grid;
else
    X_est=zeros(M,N);
end
x_m_hat=X_est.';

%% MRC detector    %% Algorithm 1 in [R1]
%% initial computation
D_m=zeros(N,N,M);
D_m_inv=zeros(N,N,M);
y_m=Y.';
% RNPI_error=y_m;
for m=1:M_prime
    for l=L_set
        D_m(:,:,m)=D_m(:,:,m)+K_ml(:,:,m+l-1,l)'*K_ml(:,:,m+l-1,l);
% equation (47) in [R1]
    end
    D_m_inv(:,:,m)=inv(D_m(:,:,m));
end
c_m=x_m_hat;
b_m_l=zeros(N,M,length(L_set));

%% iterative computation
for ite=1:n_ite_MRC
    g_m=zeros(N,M);
    RNPI_error=y_m;
    for m=1:M_prime
        for l=L_set
            b_m_l(:,m,l)=y_m(:,m+l-1);
            for p=L_set
                if(l~=p)
```

```
                        if(m+(l-p)>0)
                            b_m_l(:,m,l)=b_m_l(:,m,l)-K_ml(:,:,m+l-
1,p)*x_m_hat(:,m+(l-p));                                  % Line 5 of
Algorithm 1 in [R1]
                        end
                    end
                end
                g_m(:,m)=g_m(:,m)+K_ml(:,:,m+(l-1),l)'*b_m_l(:,m,l);
% Line 7 of Algorithm 1 in [R1]
            end
            c_m(:,m)=D_m_inv(:,:,m)*g_m(:,m);
% Line 8 of Algorithm 1 in [R1]
            if(decision==1)
                x_m_hat(:,m)=(1-
omega)*c_m(:,m)+omega*qammod(qamdemod((c_m(:,m)),M_mod,'gray'),M_mod,'gray'
);              % Line 9 of Algorithm 1 in [R1]
            else
                x_m_hat(:,m)=c_m(:,m);
            end

            for l=L_set
                if(m>=l)
                    RNPI_error(:,m)=RNPI_error(:,m)-
reshape(K_ml(:,:,m,l),N,N)*x_m_hat(:,m-(l-1));                          %
equation (50) in [R1] - residual interference
                end
            end
        end
        %% convergence criteria
        error(ite)=norm(RNPI_error);
        if(ite>1)
            if(error(ite)>=error(ite-1))
                break;
            end
        end
    end
    if(n_ite_MRC==0)
        ite=0;
    end
    %% detector output bits
    X_est=x_m_hat.';
    x_est=reshape(X_est,1,N*M);
    x_data=x_est(data_index);
    est_bits=reshape(qamdemod(x_data,M_mod,'gray','OutputType','bit'),N_bits_pe
    rfram,1);

    end
```

B.4 MRC_delay_time_detector.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%    list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%    this list of conditions and the following disclaimer in the
documentation
%    and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%    publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%     - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%     - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285
```

```
function [est_bits,ite,x_data] =
MRC_delay_time_detector(N,M,M_data,M_mod,no,data_grid,r,H_tf,nu_ml_tilda,L_
set,omega,decision,init_estimate,n_ite_MRC)
%% Normalized DFT matrix
Fn=dftmtx(N);  % Generate the DFT matrix
Fn=Fn./norm(Fn);  % normalize the DFT matrix
%% Initial assignments
%Number of symbols per frame
N_syms_perfram=sum(sum((data_grid>0)));
%Arranging the delay-Doppler grid symbols into an array
data_array=reshape(data_grid,1,N*M);
%finding position of data symbols in the array
[~,data_index]=find(data_array>0);
%number of bits per QAM symbol
M_bits=log2(M_mod);
%number of bits per frame
N_bits_perfram = N_syms_perfram*M_bits;
%received delay-time samples
Y_tilda=reshape(r,M,N);
M_prime=M_data;
L_set=L_set+1; % Since matlab cannot work with zero indices
%% initial estimate using single tap TF equalizer
if(init_estimate==1)
    Y_tf=fft(Y_tilda).'; % delay-time to frequency-time domain
% equation (63) in [R1]
    X_tf=conj(H_tf).*Y_tf./(H_tf.*conj(H_tf)+no); % single tap equalizer
% equation (64) in [R1]
    X_est = ifft(X_tf.')*Fn; % SFFT
% equation (65) in [R1]
    X_est=qammod(qamdemod(X_est,M_mod,'gray'),M_mod,'gray');
    X_est=X_est.*data_grid;
    X_tilda_est=X_est*Fn';
else
    X_est=zeros(M,N);
    X_tilda_est=X_est*Fn';
end
x_m=X_est.';
x_m_tilda=X_tilda_est.';


%% MRC detector    %% Algorithm 2 in [R1] (or Algotithm 3 in Chapter 6,
[R2])
%% initial computation
d_m_tilda=zeros(N,M);
y_m_tilda=reshape(r,M,N).';
delta_y_m_tilda=y_m_tilda;
for m=1:M_prime
    for l=L_set
        d_m_tilda(:,m)=d_m_tilda(:,m)+abs(nu_ml_tilda(:,m+(l-1),l).^2);
% equation (59) in [R1]
    end
end
for m=1:M
    for l=L_set
        if(m>=l)
            delta_y_m_tilda(:,m)=delta_y_m_tilda(:,m)-
nu_ml_tilda(:,m,l).*x_m_tilda(:,m-(l-1));
% Line 3 of Algorithm 2 in [R1]
        end
    end
end
```

```
x_m_tilda_old=x_m_tilda;
c_m_tilda=x_m_tilda;

%% iterative computation
for ite=1:n_ite_MRC
% Line 5 of Algorithm 2 in [R1]
    delta_g_m_tilda=zeros(N,M);
    for m=1:M_prime
        for l=L_set

delta_g_m_tilda(:,m)=delta_g_m_tilda(:,m)+conj(nu_ml_tilda(:,m+(l-
1),l)).*delta_y_m_tilda(:,m+(l-1));                     % Line 8 of Algorithm
2 in [R1]
        end

c_m_tilda(:,m)=x_m_tilda_old(:,m)+delta_g_m_tilda(:,m)./d_m_tilda(:,m);
% Line 9 of Algorithm 2 in [R1]
        if(decision==1)

x_m(:,m)=qammod(qamdemod(Fn*(c_m_tilda(:,m)),M_mod,'gray'),M_mod,'gray');
% Line 10 of Algorithm 2 in [R1]
            x_m_tilda(:,m)=(1-omega)*c_m_tilda(:,m)+omega*Fn'*x_m(:,m);
        else
            x_m_tilda(:,m)=c_m_tilda(:,m);
        end
        for l=L_set
% Line 11 of Algorithm 2 in [R1]
            delta_y_m_tilda(:,m+(l-1))=delta_y_m_tilda(:,m+(l-1))-
nu_ml_tilda(:,m+(l-1),l).*(x_m_tilda(:,m)-x_m_tilda_old(:,m));    % Line 12
of Algorithm 2 in [R1]
        end
% Line 13 of Algorithm 2 in [R1]
        x_m_tilda_old(:,m)=x_m_tilda(:,m);
    end

    %% convergence criteria
    error(ite)=norm(delta_y_m_tilda);
    if(ite>1)
        if(error(ite)>=error(ite-1))
% Line 15 of Algorithm 2 in [R1]
            break;
        end
    end
end
if(n_ite_MRC==0)
    ite=0;
end
%% detector output bits
X_est=(Fn*x_m_tilda).';
x_est=reshape(X_est,1,N*M);
x_data=x_est(data_index);
est_bits=reshape(qamdemod(x_data,M_mod,'gray','OutputType','bit'),N_bits_pe
rfram,1);                                                 % Line 17 of
Algorithm 2 in [R1]

end
```

B.5 Algorithm3_detector.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash 
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are 
met:
%
% 1. Redistributions of source code must retain the above copyright notice, 
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright 
notice,
%   this list of conditions and the following disclaimer in the 
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes 
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS 
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE 
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE 
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL 
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR 
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED 
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF 
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from: 
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision 
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on 
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi: 
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector 
for Orthogonal Time Frequency Space Modulation 2020 IEEE Wireless 
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi: 
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications: 
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285

function [est_bits,ite,x_data] = 
Algorithm3_detector(N,M,M_mod,no,data_grid,r,G,omega,decision,init_estimate
,n_ite)
%% Normalized DFT matrix
Fn=dftmtx(N);  % Generate the DFT matrix
```

```
Fn=Fn./norm(Fn);  % normalize the DFT matrix
%% Initial assignments
%Number of symbols per frame
N_syms_perfram=sum(sum((data_grid>0)));
%Arranging the delay-Doppler grid symbols into an array
data_array=reshape(data_grid,1,N*M);
%finding position of data symbols in the array
[~,data_index]=find(data_array>0);
%number of bits per QAM symbol
M_bits=log2(M_mod);
%number of bits per frame
N_bits_perfram = N_syms_perfram*M_bits;
%received delay-time samples
Y_tilda=reshape(r,M,N);
[Gn_block_matrix,Tn_block_matrix,zn_block_vector,H_tf]=Generate_Algo3_GS_it
eration_matrices(N,M,G,r);
%% initial time-frequency low complexity estimate assuming ideal pulses
if(init_estimate==1)
    Y_tf=fft(Y_tilda).'; % delay-time to frequency-time domain
% equation (63) in [R1]
    X_tf=conj(H_tf).*Y_tf./(H_tf.*conj(H_tf)+no); % single tap equalizer
% equation (64) in [R1]
    X_est = ifft(X_tf.')*Fn; % SFFT
% equation (65) in [R1]
    X_est=qammod(qamdemod(X_est,M_mod,'gray'),M_mod,'gray');
    X_est=X_est.*data_grid;
    X_tilda_est=X_est*Fn';
else
    X_est=zeros(M,N);
    X_tilda_est=X_est*Fn';
end
X_tilda_est=X_tilda_est.*data_grid;


%% Matched Filter Gauss Siedel algorithm (Algorithm 3 in [R1])
error=zeros(n_ite);
x_soft=zeros(M,N);

for ite=1:n_ite
    for i=1:N
        Tn=Tn_block_matrix(:,:,i);
        zn=zn_block_vector(:,i);
        sn_prev=X_tilda_est(:,i);
        sn_next=-Tn*sn_prev+zn;
%step 12 of Algorithm 3 in [R1]
        x_soft(:,i)=sn_next.*data_grid(:,i);
        X_tilda_est(:,i)=(x_soft(:,i));
    end
    if(decision==1)
        x_m=x_soft*Fn;
        X_tilda_est=(1-
omega)*X_tilda_est+omega*((qammod(qamdemod(x_m,M_mod,'gray'),M_mod,'gray').
*data_grid)*Fn');  %equation (50) in [R1] (or equivalently equation (27) in
[R2])
    end
    for i=1:N
        Gn=Gn_block_matrix(:,:,i);
        Y_tilda_est(1:M,i)=Gn*(X_tilda_est(1:M,i));
    end
    error(ite)=sum(sum(abs(Y_tilda_est-Y_tilda).^2))./N_syms_perfram;
%step 14 of Algorithm 3 in [R1]
```

```
    if(ite>1)
        if(error(ite)>=error(ite-1))
            break;
        end
    end
end
if(n_ite==0)
    ite=0;
end
%% detector output likelihood calculations for turbo decode
X_est=X_tilda_est*Fn;
x_est=reshape(X_est,1,N*M);
x_data=x_est(data_index);
est_bits=reshape(qamdemod(x_data,M_mod,'gray','OutputType','bit'),N_bits_pe
rfram,1);

end
```

B.6 Algorithm3_low_complexity_detector.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285
```

```
function [est_bits,ite,x_data] =
Algorithm3_low_complexity_detector(N,M,M_mod,no,data_grid,r,H_tf,gs,L_set,o
mega,decision,init_estimate,n_ite)
%% Normalized DFT matrix
Fn=dftmtx(N);  % Generate the DFT matrix
Fn=Fn./norm(Fn);  % normalize the DFT matrix
%% Initial assignments
%Number of symbols per frame
N_syms_perfram=sum(sum((data_grid>0)));
%Arranging the delay-Doppler grid symbols into an array
data_array=reshape(data_grid,1,N*M);
%finding position of data symbols in the array
[~,data_index]=find(data_array>0);
data_location=reshape(data_grid,N*M,1);
%number of bits per QAM symbol
M_bits=log2(M_mod);
%number of bits per frame
N_bits_perfram = N_syms_perfram*M_bits;
%received delay-time samples
Y_tilda=reshape(r,M,N);
%% initial time-frequency low complexity estimate assuming ideal pulses
if(init_estimate==1)
    Y_tf=fft(Y_tilda).'; % delay-time to frequency-time domain
% equation (63) in [R1]
    X_tf=conj(H_tf).*Y_tf./(H_tf.*conj(H_tf)+no); % single tap equalizer
% equation (64) in [R1]
    X_est = ifft(X_tf.')*Fn; % SFFT
% equation (65) in [R1]
    X_est=qammod(qamdemod(X_est,M_mod,'gray'),M_mod,'gray');
    X_est=X_est.*data_grid;
    X_tilda_est=X_est*Fn';
else
    X_est=zeros(M,N);
    X_tilda_est=X_est*Fn';
end
X_tilda_est=X_tilda_est.*data_grid;

%% Matched Filter Gauss Siedel algorithm
error=zeros(n_ite);
s_est=reshape(X_tilda_est,N*M,1);
delta_r=r;
d=zeros(N*M,1);
for q=1:N*M
    for l=(L_set+1)
        if(data_location(q)==1)
            d(q)=d(q)+abs(gs(l,q+(l-1))).^2;      % time-domain equivalent
of equation (59) in [R1]
        end
    end
end
for q=1:N*M
    for l=(L_set+1)
        if(q>=l)
            delta_r(q)=delta_r(q)-gs(l,q)*s_est(q-(l-1)); % time-domain
equivalent of line 3 of Algorithm 2 in [R1]
        end
    end
end
for ite=1:n_ite
    delta_g=zeros(N*M,1);
    s_est_old=s_est;
    for q=data_index
            for l=(L_set+1)
                delta_g(q)=delta_g(q)+conj(gs(l,q+(l-1)))*delta_r(q+(l-1));
% time-domain equivalent of line 8 of Algorithm 2 in [R1]
            end
            s_est(q)=s_est_old(q)+delta_g(q)/d(q);
% time-domain equivalent of line 9 of Algorithm 2 in [R1]
            for l=(L_set+1)
% time-domain equivalent of line 11 of Algorithm 2 in [R1]
                delta_r(q+(l-1))=delta_r(q+(l-1))-gs(l,q+(l-1))*(s_est(q)-
s_est_old(q)); % time-domain equivalent of line 12 of Algorithm 2 in [R1]
            end
% time-domain equivalent of line 13 of Algorithm 2 in [R1]
    end
    s_est_old=s_est;
    if(decision==1)
        X_est=reshape(s_est,M,N)*Fn;

X_tilda_est=((qammod(qamdemod(X_est,M_mod,'gray'),M_mod,'gray').*data_grid)
*Fn');
        s_est=(1-omega)*s_est+omega*reshape(X_tilda_est,N*M,1);
    end
    for q=data_index
            for l=(L_set+1)
% time-domain equivalent of line 11 of Algorithm 2 in [R1]
                delta_r(q+(l-1))=delta_r(q+(l-1))-gs(l,q+(l-1))*(s_est(q)-
s_est_old(q));   % time-domain equivalent of line 12 of Algorithm 2 in [R1]
            end
% time-domain equivalent of line 13 of Algorithm 2 in [R1]
    end
    error(ite)=norm(delta_r);
    if(ite>1)
        if(error(ite)>=error(ite-1))
% time-domain equivalent of line 15 of Algorithm 2 in [R1]
            break;
        end
    end
end
if(n_ite==0)
    ite=0;
end
%% detector output likelihood calculations for turbo decode
X_tilda_est=reshape(s_est,M,N);
X_est=X_tilda_est*Fn;
x_est=reshape(X_est,1,N*M);
x_data=x_est(data_index);
est_bits=reshape(qamdemod(x_data,M_mod,'gray','OutputType','bit'),N_bits_pe
rfram,1);

end
```

B.7 MPA_detector.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R4]. P. Raviteja, K. T. Phan, Y. Hong and E. Viterbo,``Interference
Cancellation and Iterative Detection for Orthogonal Time Frequency Space
Modulation in IEEE Transactions on Wireless Communications, vol. 17, no.
10, pp. 6501-6515, Oct. 2018, doi: 10.1109/TWC.2018.2860011.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285



function [est_bits,ite,x_est] =
MPA_detector(N,M,M_mod,no,data_grid,y,H,n_ite)
%% Initial assignments
%Number of symbols per frame
N_syms_perfram=sum(sum((data_grid>0)));
%Arranging the delay-Doppler grid symbols into an array
data_array=reshape(data_grid,1,N*M);
```

```
%finding position of data symbols in the array
[~,data_index]=find(data_array>0);
% data start delay index in the delay-Doppler grid
M_bits=log2(M_mod);
N_bits_perfram = N_syms_perfram*M_bits;

alphabet=qammod(0:M_mod-1,M_mod,'gray').';
x_est=zeros(M*N,1);
ind_ba=zeros(N,M*N);
ind_ab=zeros(N,M*N);
length_ba=zeros(M*N,1);
length_ab=zeros(M*N,1);
for b=1:N*M
    h_ba=H(b,:);
    ind_sparse=find(abs(h_ba)>1e-6);
    ind_sparse=sort(ind_sparse);
    length_ba(b)=length(ind_sparse);
    ind_ba(1:length_ba(b),b)=ind_sparse;
end
for a=1:N*M
    h_ab=H(:,a);
    ind_sparse=find(abs(h_ab)>1e-6);
    length_ab(a)=length(ind_sparse);
    ind_ab(1:length_ab(a),a)=ind_sparse;
end

taps=max(max(length_ab),max(length_ba));
p_map = ones(N*M,N*M,M_mod)*(1/M_mod);

delta_fra=0.7;
conv_rate_prev = -0.1;
mean_int = zeros(N*M,N*M);
var_int = zeros(N*M,N*M);


for ite=1:n_ite
    for b=1:M*N
        mean_int_hat = zeros(taps,1);
        var_int_hat = zeros(taps,1);
        for tap_no=1:taps
            if(ind_ba(tap_no,b)~=0)
                new_chan= H(b,ind_ba(tap_no,b));
                for i2=1:1:M_mod
                    mean_int_hat(tap_no) = mean_int_hat(tap_no) +
p_map(b,ind_ba(tap_no,b),i2) * alphabet(i2);
                    var_int_hat(tap_no) = var_int_hat(tap_no) +
p_map(b,ind_ba(tap_no,b),i2) * abs(alphabet(i2))^2;
                end
                mean_int_hat(tap_no) = mean_int_hat(tap_no) * new_chan;
                var_int_hat(tap_no) = var_int_hat(tap_no) *
abs(new_chan)^2;
                var_int_hat(tap_no) = var_int_hat(tap_no) -
abs(mean_int_hat(tap_no))^2;
            end
        end

        mean_int_sum = sum(mean_int_hat);
        var_int_sum = sum(var_int_hat)+(no);
        for tap_no=1:taps
            if(ind_ba(tap_no,b)~=0)
```

```
                mean_int(b,ind_ba(tap_no,b)) = mean_int_sum -
mean_int_hat(tap_no);
                var_int(b,ind_ba(tap_no,b)) = var_int_sum -
var_int_hat(tap_no);
            end
        end
    end
    sum_prob_comp = zeros(M*N,M_mod);

    for a=1:M*N
        dum_sum_prob = zeros(M_mod,1);
        log_te_var=zeros(taps,M_mod);
        for tap_no=1:taps
            c=ind_ab(tap_no,a);
            if(c~=0)
                new_chan= H(c,a);
                for i2=1:1:M_mod
                    dum_sum_prob(i2) = abs(y(c)- mean_int(c,a) - new_chan *
alphabet(i2))^2;
                    dum_sum_prob(i2)= -(dum_sum_prob(i2)/var_int(c,a));
                end
                dum_sum = dum_sum_prob - max(dum_sum_prob);
                dum1 = sum(exp(dum_sum));
                log_te_var(tap_no,:) = dum_sum - log(dum1);
            end
        end
        %
        for i2=1:1:M_mod
            ln_qi(i2) = sum(log_te_var(:,i2));
        end
        dum_sum = exp(ln_qi - max(ln_qi));
        dum1 = sum(dum_sum);
        sum_prob_comp(a,:) = dum_sum/dum1;
        for tap_no=1:1:taps
            c=ind_ab(tap_no,a);
            if(c~=0)
                dum_sum = log_te_var(tap_no,:);
                ln_qi_loc = ln_qi - dum_sum;
                dum_sum = exp(ln_qi_loc - max(ln_qi_loc));
                dum1 = sum(dum_sum);
                p_map(c,a,:) = (dum_sum/dum1)*delta_fra + (1-
delta_fra)*reshape(p_map(c,a,:),1,M_mod);

            end
        end
    end
    conv_rate =  sum(max(sum_prob_comp,[],2)>0.99)/(N*M);
    if conv_rate==1
        sum_prob_fin = sum_prob_comp;
        break;
    elseif conv_rate > conv_rate_prev
        conv_rate_prev = conv_rate;
        sum_prob_fin = sum_prob_comp;
    elseif (conv_rate < conv_rate_prev - 0.2) && conv_rate_prev > 0.95
        break;
    end

end
for a=1:M*N
    [m1,m2]=max(sum_prob_fin(a,:));
    x_est(a)=alphabet(m2);
```

```
end
X_est=reshape(x_est,N,M).';
%% detector output
x_est=reshape(X_est,1,N*M);
x_data=x_est(data_index);
est_bits=reshape(qamdemod(x_data,M_mod,'gray','OutputType','bit'),N_bits_pe
rfram,1);

end
```

B.8 Block_LMMSE_detector.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%    list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%    this list of conditions and the following disclaimer in the
documentation
%    and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%    publication.
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%     - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%     - Freely distributed for educational and research purposes

%References

%   [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285



function [est_bits,x_data] =
Block_LMMSE_detector(N,M,M_mod,noise_var,data_grid,r,gs,L_set)
%% Normalized DFT matrix
Fn=dftmtx(N);   % Generate the DFT matrix
Fn=Fn./norm(Fn);   % normalize the DFT matrix
%% Initial assignments
%Number of symbols per frame
N_syms_perfram=sum(sum((data_grid>0)));
%Arranging the delay-Doppler grid symbols into an array
data_array=reshape(data_grid,1,N*M);
```

```
%finding position of data symbols in the array
[~,data_index]=find(data_array>0);
%number of bits per QAM symbol
M_bits=log2(M_mod);
%number of bits per frame
N_bits_perfram = N_syms_perfram*M_bits;
%received time domain blocks
sn_block_est=zeros(M,N);
Gn=zeros(M,M);
%% block-wise LMMSE detection
for n=1:N
    for m=1:M
        for l=L_set+1
            if(m>=l)
                Gn(m,m-l+1)=gs(l,m+(n-1)*M);
            end
        end
    end
    rn=r((n-1)*M+1:n*M);
    Rn=Gn'*Gn;
    sn_block_est(:,n)=(Rn+noise_var.*eye(M))^(-1)*(Gn'*rn);
end
X_tilda_est=sn_block_est;
%% detector output
X_est=X_tilda_est*Fn;
x_est=reshape(X_est,1,N*M);
x_data=x_est(data_index);
est_bits=reshape(qamdemod(x_data,M_mod,'gray','OutputType','bit'),N_bits_pe
rfram,1);
end
```

B.9 TF_single_tap_equalizer

```
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285


function [est_bits,x_data] =
TF_single_tap_equalizer(N,M,M_mod,noise_var,data_grid,Y,H_tf)
%% Normalized DFT matrix
Fn=dftmtx(N);  % Generate the DFT matrix
```

```
Fn=Fn./norm(Fn);  % normalize the DFT matrix
%% Initial assignments
%Number of symbols per frame
N_syms_perfram=sum(sum((data_grid>0)));
%Arranging the delay-Doppler grid symbols into an array
data_array=reshape(data_grid,1,N*M);
%finding position of data symbols in the array
[~,data_index]=find(data_array>0);
M_bits=log2(M_mod);
N_bits_perfram = N_syms_perfram*M_bits;

%% initial time-frequency low complexity estimate
Y_tf=fft((Y*Fn')).'; % ISFFT
X_tf=conj(H_tf).*Y_tf./(H_tf.*conj(H_tf)+noise_var); % single tap equalizer
X_est = ifft(X_tf.')*Fn; % SFFT

%% detector output likelihood calculations for turbo decode
x_est=reshape(X_est,1,N*M);
x_data=x_est(data_index);
est_bits=reshape(qamdemod(x_data,M_mod,'gray','OutputType','bit'),N_bits_pe
rfram,1);

end
```

B.10 Initialize_error_count_variables.m

```
est_info_bits_MRC=zeros(N_bits_perfram,1);
est_info_bits_Algo1=zeros(N_bits_perfram,1);
est_info_bits_Algo3=zeros(N_bits_perfram,1);
est_info_bits_Algo3_low_complexity=zeros(N_bits_perfram,1);
est_info_bits_MPA=zeros(N_bits_perfram,1);
est_info_bits_1tap=zeros(N_bits_perfram,1);
est_info_bits_LMMSE=zeros(N_bits_perfram,1);

err_ber_MRC = zeros(1,length(SNR_dB));
err_ber_Algo1 = zeros(1,length(SNR_dB));
err_ber_Algo3 = zeros(1,length(SNR_dB));
err_ber_Algo3_low_complexity = zeros(1,length(SNR_dB));
err_ber_MPA = zeros(1,length(SNR_dB));
err_ber_1tap = zeros(1,length(SNR_dB));
err_ber_LMMSE = zeros(1,length(SNR_dB));

avg_ber_MRC=zeros(1,length(SNR_dB));
avg_ber_Algo1=zeros(1,length(SNR_dB));
avg_ber_Algo3=zeros(1,length(SNR_dB));
avg_ber_Algo3_low_complexity=zeros(1,length(SNR_dB));
avg_ber_MPA=zeros(1,length(SNR_dB));
avg_ber_1tap=zeros(1,length(SNR_dB));
avg_ber_LMMSE=zeros(1,length(SNR_dB));

det_iters_MRC=0;
no_of_detetor_iterations_MRC= zeros(length(SNR_dB),1);
avg_no_of_iterations_MRC=zeros(1,length(SNR_dB));

det_iters_Algo1=0;
no_of_detetor_iterations_Algo1= zeros(length(SNR_dB),1);
avg_no_of_iterations_Algo1=zeros(1,length(SNR_dB));

det_iters_Algo3=0;
no_of_detetor_iterations_Algo3= zeros(length(SNR_dB),1);
avg_no_of_iterations_Algo3=zeros(1,length(SNR_dB));

det_iters_Algo3_low_complexity=0;
no_of_detetor_iterations_Algo3_low_complexity= zeros(length(SNR_dB),1);
avg_no_of_iterations_Algo3_low_complexity=zeros(1,length(SNR_dB));

det_iters_MPA=0;
no_of_detetor_iterations_MPA= zeros(length(SNR_dB),1);
avg_no_of_iterations_MPA=zeros(1,length(SNR_dB));
```

B.11 Generate_time_frequency_channel_ZP.m

```
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%    list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%    this list of conditions and the following disclaimer in the
documentation
%    and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%    publication.
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%     - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%     - Freely distributed for educational and research purposes

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation�� 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285

function [H_t_f]=Generate_time_frequency_channel_ZP(N,M,gs,L_set)
H_t_f=zeros(N,M); % Time-frequency single tap channel matrix
Fm=dftmtx(M);
Fm=Fm./norm(Fm);
Gn=zeros(M,M);
for n=1:N
    for m=1:M
```

```
            for l=L_set+1
                if(m>=l)
                    Gn(m,m-l+1)=gs(l,m+(n-1)*M);  %equation(42) in [R1]
                end
            end
        end
        H_t_f(n,1:M)=diag(Fm*Gn*Fm').';
    end

    end
```

B.12 Generate_delay_Doppler_channel_parameters.m

```
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation�� 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285


function
[chan_coef,delay_taps,Doppler_taps,taps]=Generate_delay_Doppler_channel_par
ameters(N,M,car_fre,delta_f,T,max_speed)
one_delay_tap = 1/(M*delta_f);
```

```
one_doppler_tap = 1/(N*T);

%delays = [0 30 70 90 110 190 410]*10^(-9);%EPA model
%delays = [0 30 150 310 370 710 1090 1730 2510]*10^(-9);%EVA model
delays = [0 50 120 200 230 500 1600 2300 5000]*10^(-9);%ETU model


taps = length(delays);% number of delay taps
delay_taps = round(delays/one_delay_tap);%assuming no fraction for the
delay

%pdp = [0 -1.0 -2.0 -3.0 -8.0 -17.2 -20.8];% EPA power delay profile
%pdp = [0 -1.5 -1.4 -3.6 -0.6 -9.1 -7.0 -12.0 -16.9];% EVA power delay
profile
pdp = [-1 -1 -1 0 0 0 -3 -5 -7];% ETU power delay profile


pow_prof = 10.^(pdp/10);
pow_prof = pow_prof/sum(pow_prof);%normalization of power delay profile
chan_coef = sqrt(pow_prof).*(sqrt(1/2) *
(randn(1,taps)+1i*randn(1,taps)));%channel coef. for each path
max_UE_speed = max_speed*(1000/3600);
Doppler_vel = (max_UE_speed*car_fre)/(299792458);
max_Doppler_tap = Doppler_vel/one_doppler_tap;
Doppler_taps = (max_Doppler_tap*cos(2*pi*rand(1,taps)));%Doppler taps using
Jake's spectrum

end
```

B.13 Generate_Algo3_GS_iteration_matrices.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation�� 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285


function
[Gn_block_matrix,Tn_block_matrix,zn_block_vector,H_t_f]=Generate_Algo3_GS_i
teration_matrices(N,M,G,r)
```

```
% Generate time-domain GS iteration matrices for low complexity iterative
detection
Gn_block_matrix=zeros(M,M,N);
Tn_block_matrix=zeros(M,M,N);
Qn_block_matrix=zeros(M,M,N);
zn_block_vector=zeros(M,N);

H_t_f=zeros(N,M); % Time-frequency single tap channel matrix
Fn=dftmtx(M);
Fn=Fn./norm(Fn);
for n=1:N
    rn=r((n-1)*M+1:n*M);
    Gn_block_matrix(:,:,n)=G((n-1)*M+1:n*M,(n-1)*M+1:n*M);
    Gn=Gn_block_matrix(:,:,n);
    H_t_f(n,1:M)=diag(Fn*Gn*Fn').';  % Generate time-frequency channel
matrix for low complexity initial estimate in [R1,R3]
    Rn=Gn'*Gn;
    Dn=diag(diag(Rn));
    Ln=tril(Rn,-1);
    Un=triu(Rn,1);
    Qn=(Dn+Ln)^(-1);
    Tn=Qn*Un;
    Tn_block_matrix(:,:,n)=Tn;
    Qn_block_matrix(:,:,n)=Qn;
    zn_block_vector(:,n)=Qn*Gn'*rn;
end

end
```

B.14 Generate_2D_data_grid.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation�� 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285



function X = Generate_2D_data_grid(N,M,x_data,data_grid)
x_vec=zeros(N*M,1);
```

```
data_array=reshape(data_grid,1,N*M);
[~,data_pos]=find(data_array>0);
x_vec(data_pos)=x_data;
X=reshape(x_vec,M,N);

end
```

B.15 Gen_time_domain_channel.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation�� 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285



function
[G,gs]=Gen_time_domain_channel(N,M,P,delay_taps,Doppler_taps,chan_coef)
```

```
z=exp(1i*2*pi/N/M);
l_max=max(delay_taps);
gs=zeros(l_max+1,N*M);
G=zeros(N*M,N*M);

for q=0:N*M-1
    for i=1:P
        g_i=chan_coef(i);
        l_i=delay_taps(i);
        k_i=Doppler_taps(i);
        gs(l_i+1,q+1)=gs(l_i+1,q+1)+g_i*z^(k_i*(q-l_i));  % equation (16)
in [R1]
    end
end

for q=0:N*M-1
    for l=0:l_max
        if(mod(q,M)>=l) % due to ZP per block
            G(q+1,q+1-l)=gs(l+1,q+1);                      % equation (42)
in [R1]
        end
    end
end

end
```

B.16 Gen_DT_and_DD_channel_vectors.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation�� 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285


function
[nu_ml_tilda,nu_ml,K_ml]=Gen_DT_and_DD_channel_vectors(N,M,L_set,gs)
l_max=max(L_set);
```

```
nu_ml_tilda=zeros(N,M,l_max+1);
nu_ml=zeros(N,M,l_max+1);
K_ml=zeros(N,N,M,l_max+1);
Fn=dftmtx(N);
Fn=Fn./norm(Fn);
for m=1:M
    for l=(L_set+1)
        for n=1:N
            nu_ml_tilda(n,m,l)=gs(l,m+(n-1)*M);         %equation(42) in
[R1]
        end
        nu_ml(:,m,l)=Fn*nu_ml_tilda(:,m,l);             %Section III-A in
[R1]
        K_ml(:,:,m,l)=Fn*diag(nu_ml_tilda(:,m,l))*Fn';  %Section III-A in
[R1]
    end
end

end
```

B.17 Gen_discrete_time_channel.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation�� 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285
```

```
function
[gs]=Gen_discrete_time_channel(N,M,P,delay_taps,Doppler_taps,chan_coef)
z=exp(1i*2*pi/N/M);
l_max=max(delay_taps);
gs=zeros(l_max+1,N*M);
for q=0:N*M-1
    for i=1:P
        g_i=chan_coef(i);
        l_i=delay_taps(i);
        k_i=Doppler_taps(i);
        gs(l_i+1,q+1)=gs(l_i+1,q+1)+g_i*z^(k_i*(q-l_i));  % equation (16)
in [R1]
    end
end

end
```

B.18 Gen_delay_time_channel_vectors.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%   list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%   this list of conditions and the following disclaimer in the
documentation
%   and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%   publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation�� 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285


function [nu_ml_tilda]=Gen_delay_time_channel_vectors(N,M,l_max,gs)
nu_ml_tilda=zeros(N,M,l_max+1);
for n=1:N
```

```
    for m=1:M
        for l=1:(l_max+1)
            nu_ml_tilda(n,m,l)=gs(l,m+(n-1)*M);  %equation(42) in [R1]
        end
    end
end

end
```

B.19 Gen_DD_and_DT_channel_matrices.m

```
%
% Copyright (c) 2021, Tharaj Thaj, Emanuele Viterbo, and  Yi Hong, Monash
University
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions are
met:
%
% 1. Redistributions of source code must retain the above copyright notice,
this
%    list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
notice,
%    this list of conditions and the following disclaimer in the
documentation
%    and/or other materials provided with the distribution.
% 3. The reference listed below should be cited if the corresponding codes
are used for
%    publication..
%
%THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND
%ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
%WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
%DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR
%ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
%(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
%LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
AND
%ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
%(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS
%SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
%
%    - Latest version of this code may be downloaded from:
https://ecse.monash.edu/staff/eviterbo/
%    - Freely distributed for educational and research purposes
%References

%  [R1]. T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision
Feedback Equalizer for Zero-Padded OTFS Systems," in IEEE Transactions on
Vehicular Technology, vol. 69, no. 12, pp. 15606-15622, Dec. 2020, doi:
10.1109/TVT.2020.3044276.
%  [R2]. T. Thaj and E. Viterbo,``Low Complexity Iterative Rake Detector
for Orthogonal Time Frequency Space Modulation�� 2020 IEEE Wireless
Communications and Networking Conference (WCNC), 2020, pp. 1-6, doi:
10.1109/WCNC45663.2020.9120526.
%  [R3]. Y. Hong, T. Thaj, E. Viterbo, ``Delay-Doppler Communications:
Principles and Applications'', Academic Press, 2022, ISBN:9780323850285


function [H,H_tilda,P]= Gen_DD_and_DT_channel_matrices(N,M,G,Fn)
P=zeros(N*M,N*M);
for j=1:N
```

```
    for i=1:M
        E=zeros(M,N);
        E(i,j)=1;
        P((j-1)*M+1:j*M,(i-1)*N+1:i*N)=E;  % row-column interleaver matrix
in equation (35) in [R1]
    end
end
H_tilda=(P'*G*P);
H=kron(eye(M),Fn)*(P'*G*P)*kron(eye(M),Fn'); %using equation (33) and (40)
in [R1]
end
```

B.20 display_errors_per_frame.m

```
clc
fprintf('ZP-OTFS-(N,M,QAM size)');disp([N,M,M_mod]);
display(current_frame_number,'Number of frames');
display(SNR_dB,'SNR (dB)');
display(avg_ber_MRC,'Average BER - Delay-time domain Maximal Ratio
Combining (MRC)');
display(avg_ber_Algo1,'Average BER - Algorithm 1 in [R1]');
display(avg_ber_Algo3,'Average BER - Algorithm 3 in [R1]');
display(avg_ber_Algo3_low_complexity,'Average BER -low-complexity Algorithm
3 in [R1]');
display(avg_ber_MPA,'Average BER - Message passing algorithm (MPA)');
display(avg_ber_1tap,'Average BER - Single tap TF equalizer');
display(avg_ber_LMMSE,'Average BER - LMMSE equalizer');
display(avg_no_of_iterations_MRC,'Average number of iterations for the
delay-time domain MRC detector');
display(avg_no_of_iterations_Algo1,'Average number of iterations for the
delay-Doppler domain MRC detector');
display(avg_no_of_iterations_Algo3,'Average number of iterations for
Algorithm 3 in [R1]');
display(avg_no_of_iterations_Algo3_low_complexity,'Average number of
iterations for low-complexity Algorithm 3 in [R1]');
display(avg_no_of_iterations_MPA,'Average number of iterations for the MPA
detector');
```

B.21 count_errors_per_frame.m

```
%% errors count%%%%%
        errors_MRC = sum(xor(est_info_bits_MRC,trans_info_bit));
        errors_Algo1 = sum(xor(est_info_bits_Algo1,trans_info_bit));
        errors_Algo3 = sum(xor(est_info_bits_Algo3,trans_info_bit));
        errors_Algo3_low_complexity =
sum(xor(est_info_bits_Algo3_low_complexity,trans_info_bit));
        errors_MPA = sum(xor(est_info_bits_MPA,trans_info_bit));
        errors_1tap = sum(xor(est_info_bits_1tap,trans_info_bit));
        errors_LMMSE = sum(xor(est_info_bits_LMMSE,trans_info_bit));

        err_ber_MRC(1,iesn0) = err_ber_MRC(1,iesn0) + errors_MRC;
        err_ber_Algo1(1,iesn0) = err_ber_Algo1(1,iesn0) + errors_Algo1;
        err_ber_Algo3(1,iesn0) = err_ber_Algo3(1,iesn0) + errors_Algo3;
        err_ber_Algo3_low_complexity(1,iesn0) =
err_ber_Algo3_low_complexity(1,iesn0) + errors_Algo3_low_complexity;
        err_ber_MPA(1,iesn0) = err_ber_MPA(1,iesn0) + errors_MPA;
        err_ber_1tap(1,iesn0) = err_ber_1tap(1,iesn0) + errors_1tap;
        err_ber_LMMSE(1,iesn0) = err_ber_LMMSE(1,iesn0) + errors_LMMSE;


avg_ber_MRC(1,iesn0)=err_ber_MRC(1,iesn0).'/length(trans_info_bit)/ifram;

avg_ber_Algo1(1,iesn0)=err_ber_Algo1(1,iesn0).'/length(trans_info_bit)/ifra
m;

avg_ber_Algo3(1,iesn0)=err_ber_Algo3(1,iesn0).'/length(trans_info_bit)/ifra
m;

avg_ber_Algo3_low_complexity(1,iesn0)=err_ber_Algo3_low_complexity(1,iesn0)
.'/length(trans_info_bit)/ifram;

avg_ber_MPA(1,iesn0)=err_ber_MPA(1,iesn0).'/length(trans_info_bit)/ifram;

avg_ber_1tap(1,iesn0)=err_ber_1tap(1,iesn0).'/length(trans_info_bit)/ifram;

avg_ber_LMMSE(1,iesn0)=err_ber_LMMSE(1,iesn0).'/length(trans_info_bit)/ifra
m;


        %%  iterations count


no_of_detetor_iterations_MRC(iesn0)=no_of_detetor_iterations_MRC(iesn0)+det
_iters_MRC;

avg_no_of_iterations_MRC(iesn0)=no_of_detetor_iterations_MRC(iesn0)/ifram;

no_of_detetor_iterations_Algo1(iesn0)=no_of_detetor_iterations_Algo1(iesn0)
+det_iters_Algo1;

avg_no_of_iterations_Algo1(iesn0)=no_of_detetor_iterations_Algo1(iesn0)/ifr
am;

no_of_detetor_iterations_Algo3(iesn0)=no_of_detetor_iterations_Algo3(iesn0)
+det_iters_Algo3;

avg_no_of_iterations_Algo3(iesn0)=no_of_detetor_iterations_Algo3(iesn0)/ifr
am;
```

```
no_of_detetor_iterations_Algo3_low_complexity(iesn0)=no_of_detetor_iteratio
ns_Algo3_low_complexity(iesn0)+det_iters_Algo3_low_complexity;

avg_no_of_iterations_Algo3_low_complexity(iesn0)=no_of_detetor_iterations_A
lgo3_low_complexity(iesn0)/ifram;

no_of_detetor_iterations_MPA(iesn0)=no_of_detetor_iterations_MPA(iesn0)+det
_iters_MPA;

avg_no_of_iterations_MPA(iesn0)=no_of_detetor_iterations_MPA(iesn0)/ifram;
```

Chapter 7 References

[1] S. Mukhopadhyay, V. Agarwal, S. Sharma and V. Gupta, "A Study On Wireless Communication Networks Based On Different Generations," *International Journal of Current Trends in Engineering & Research (IJCTER),* vol. 2, pp. 300-304, May 2016.

[2] H. Mehta, D. Patel, B. Joshi and H. Modi, "0G to 5G Mobile Technology: A Survey," in *ASCIET*, 2014.

[3] F. Blecher, "Advanced mobile phone service," *IEEE Transactions on Vehicular Technology,* vol. 29, no. 2, pp. 238-244, 1980.

[4] W. Jyhi-Kong, C. Han-Tsung, S. Lir-Fan and Y. Wei-Pang, "Clone terminator: an authentication service for advanced mobile phone system," in *1995 IEEE 45th Vehicular Technology Conference. Countdown to the Wireless Twenty-First Century*, 1995.

[5] G. Hooper and A. Sicher, "Advanced TDMA digital AMPS mobile data and messaging capabilities," in *Proceedings of COM'96. First Annual Conference on Emerging Technologies and Applications in Communications*, 1996.

[6] M. Rahnema, "Overview of the GSM system and protocol architecture," *IEEE Communications Magazine,* vol. 31, no. 4, pp. 92-100, 1993.

[7] D. Terasawa and E. Tiedemann, "cdmaOne(R) (IS-95) technology overview and evolution," in *1999 IEEE Radio Frequency Integrated Circuits Symposium (Cat No.99CH37001)*, Anaheim, CA, USA, 1999.

[8] T. Newe and T. Coffey, "Security protocols for 2G and 3G wireless communications.," in *Proceedings of the 1st Intenational Symposium on Information and Communication Technologies*, Dublin, Ireland, 2003.

[9] C. Cox, "Circuit Switched Fallback," in *An Introduction to LTE: LTE, LTE-Advanced, SAE, VoLTE and 4G Mobile Communications*, Wiley Telecom, 2014, pp. 349-369.

[10] 3GPP, "Introducing 3GPP," 3GPP, 2022. [Online]. Available: https://www.3gpp.org/about-us/introducing-3gpp. [Accessed 9 August 2023].

[11] E. Dahlman, B. Gudmundson, M. Nilsson and A. Skold, "UMTS/IMT-2000 based on wideband CDMA," *IEEE Communications Magazine,* vol. 36, no. 9, pp. 70-80, 1998.

[12] A. Ahmed, A. Al-Dweik, Y. Iraqi, H. Mukhtar, M. Naeem and E. Hossain, "Hybrid Automatic Repeat Request (HARQ) in Wireless Communications Systems and Standards: A Contemporary Survey," *IEEE Communications Surveys & Tutorials,* vol. 23, no. 4, pp. 2711-2752, 2021.

[13] J. Prokkola, P. Perala, M. Hanski and E. Piri, "3G/HSPA Performance in Live Networks from the End User Perspective," in *2009 IEEE International Conference on Communications*, 2009.

[14] S. Parkvall, J. Skold, P. Beming and E. Dalhman, 3G Evolution: HSPA and LTE for Mobile Broadband, Academic Press, 2010, p. 648.

[15] 3GPP, "Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); 3G security; Security architecture," ETSI, 2013.

[16] F. Grijpink, A. Ménard, H. Sigurdsson and N. Vucevic, "The road to 5G: The inevitable growth of infrastructure cost," McKinsey & Company, 2018.

[17] K. Loa, C.-c. Wu, S.-t. Sheu, Y. Yuan, M. Chion, D. Huo and L. Xu, "IMT-advanced relay standards [WiMAX/LTE Update]," *IEEE Communications Magazine,* vol. 48, no. 8, pp. 40-48, 2010.

[18] J. Winters, "Smart antennas for wireless systems," *IEEE Personal Communications,* vol. 5, no. 1, pp. 23-27, 1998.

[19] A. Scaloni, P. Cirella, M. Sgheiz, R. Diamanti and D. Micheli, "Multipath and Doppler Characterization of an Electromagnetic Environment by Massive MDT Measurements From 3G and 4G Mobile Terminals," *IEEE Access,* vol. 7, pp. 13024-13034, 2019.

[20] S. Jun-Zhao, J. Sauvola and D. Howie, "Features in future: 4G visions from a technical perspective," in *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No.01CH37270)*, 2001.

[21] C. Perkins, "Mobile IP," *IEEE Communications Magazine,* vol. 35, no. 5, pp. 84-99, 1997.

[22] N. Seddigh, B. Nandy, R. Makkar and J. Beaumont, "Security advances and challenges in 4G wireless networks," in *2010 Eighth International Conference on Privacy, Security and Trust*, 2010.

[23] Wikipedia, "4G," 22 May 2023. [Online]. Available: https://en.wikipedia.org/wiki/4G. [Accessed 9 August 2023].

[24] A. Hikmaturokhman, L. S. Palupi, N. Amalia, A. R. Danisya and T. A. Nugraha, "4G LTE Evolved Packet Core Planning with Call Switch Fallback Technology," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC),* vol. 10, no. 1-6, pp. 133-136, February 2018.

[25] 3GPP, "Technical Specification Group Radio Access Network; Requirements for further advancements for Evolved Universal Terrestrial Radio Access (E-UTRA) (LTE-Advanced)," ETSI, 2023.

[26] T. Q. Quek, G. de la Roche, I. Guvenc and M. Kountouris, Small Cell Networks: Deployment, PHY Techniques, and Resource Management, Cambridge University Press, 2013.

[27] J. Wannstrom, "LTE-Advanced," 3GPP, 2012.

[28] E. Dahlman, S. Parkvall and J. Skold, 4G: LTE/LTE-Advanced for Mobile Broadband, 2 ed., Academic Press, 2013.

[29] P. Taylor, "Global market share of mobile telecom technology 2016-2025, by generation," Statista, 2023.

[30] A. Ghosh, A. Maeder, M. Baker and D. Chandramouli, "5G Evolution: A View on 5G Cellular Technology Beyond 3GPP Release 15," *IEEE Access,* vol. 7, pp. 127639-127651, 2019.

[31] A10 Staff, "5G Key Technologies," A10, 27 March 2019. [Online]. Available: https://www.a10networks.com/blog/5g-key-technologies/. [Accessed 9 August 2023].

[32] Qualcomm Technologies, Inc., "Everything you need to know about 5G.," Qualcomm Technologies, Inc., [Online]. Available: https://www.qualcomm.com/5g/what-is-5g. [Accessed 9 August 2023].

[33] Qualcomm Technologies, Inc., "Making 5G NR a reality; Leading the technology innovations for a unified, more capable 5G air interface," Qualcomm Technologies, Inc., 2016.

[34] E. Dahlman, S. Parkvall and J. Skold, 5G NR: The Next Generation Wireless Access Technology, Academic Press, 2020.

[35] T. Rühlig, J. Seaman and D. Voelsen, 5G and the US-China tech rivalry - a test for Europe's future in the digital age: how can Europe shift back from back foot to front foot?, Berlin: Stiftung Wissenschaft und Politik -SWP- Deutsches Institut für Internationale Politik und Sicherheit, 2019.

[36] I. Abubakar, J. Din, M. Alhilali and H. Y. Lam, "Interference and Electromagnetic Compatibility," *Indonesian Journal of Electrical Engineering and Computer Science,* vol. 5, no. 3, pp. 612-621, 2017.

[37] S. Maxim, "Electromagnetic interference hazards in flight and the 5G mobile phone: Review of critical issues in aviation security," *Transportation Research Procedia,* vol. 59, pp. 310-318, 2021.

[38] A. Al-Jumaily, A. Sali, V. P. G. Jiménez, F. P. Fontán, M. J. Singh, A. Ismail, Q. Al-Maatouk, A. M. Al-Saegh and D. Al-Jumeily, "Evaluation of 5G Coexistence and Interference Signals in the C-Band Satellite Earth Station," *IEEE Transactions on Vehicular Technology,* vol. 71, no. 6, pp. 6189-6200, 2022.

[39] M. Gonokami, H. Morikawa, H. Fujiwara, K. Iizumi, T. Kimura, K. Nemoto, A. Shinozaki, E. Takemura, H. Tokuda and Y. Uchinaga, "Beyond 5G Promotion Consortium," Beyond 5G Promotion Consortium, December 2020. [Online]. Available: https://b5g.jp/en/. [Accessed 10 August 2023].

[40] K. Nagai, "Beyond 5G: The New Era of Connectivity," NEC Global, 7 July 2023. [Online]. Available: https://www.nec.com/en/global/solutions/5g/blog/beyond-5g-the-new-era-of-connectivity.html. [Accessed 10 August 2023].

[41] S. Dang, O. Amin, B. Shihada and M.-S. Alouini, "What should 6G be?," *Nature Electronics,* vol. 3, no. 1, pp. 20-29, 2020.

[42] H. Elayan, O. Amin, R. M. Shubair and M.-S. Alouini, "Terahertz communication: The opportunities of wireless technology beyond 5G," in *2018 International Conference on Advanced Communication Technologies and Networking (CommNet)*, 2018.

[43] C. Wang and A. Rahman, "Quantum-Enabled 6G Wireless Networks: Opportunities and Challenges," *IEEE Wireless Communications,* vol. 29, no. 1, pp. 58-69, 2022.

[44] S. Chen, S. Sun and S. Kang, "System integration of terrestrial mobile communication and satellite communication —the trends, challenges and key technologies in B5G and 6G," *China Communications,* vol. 17, no. 12, pp. 156-171, 2020.

[45] A. B. Tareq, P. Ripon and N. Sabuzima, "A vision on the artificial intelligence for 6G communication," *ICT Express,* vol. 9, no. 2, pp. 197-210, 2023.

[46] Q. Zakria, L. N. Khoa, S. Nasir and M. S. Hafiz, "Towards 6G Internet of Things: Recent advances, use cases, and open challenges," *ICT Express,* vol. 9, no. 3, pp. 296-312, 2023.

[47] X. Yang, Z. Zho and B. Huang, "URLLC Key Technologies and Standardization for 6G Power Internet of Things," *IEEE Communications Standards Magazine,* vol. 5, no. 2, pp. 52-59, 2021.

[48] R. Hadani, S. Rakib, S. Kons, M. Tsatsanis, A. Monk, C. Ibars, J. Delfeld, Y. Hebron, A. J. Goldsmith, A. F. Molisch and R. A. Calderbank, "Orthogonal Time Frequency Space Modulation," *CoRR,* vol. abs/1808.00519, no. 1, 2018.

[49] W. Cochran, J. Cooley, D. Favin, H. Helms, R. Kaenel, W. Lang, G. Maling, D. Nelson, C. Rader and P. Welch, "What is the fast Fourier transform?," *Proceedings of the IEEE,* vol. 55, no. 10, pp. 1664-1674, 1967.

[50] MathWorks, "Fourier Transforms," MathWorks, 2023. [Online]. Available: https://www.mathworks.com/help/matlab/math/fourier-transforms.html. [Accessed 10 August 2023].

[51] G. Stuber, J. Barry, S. McLaughlin, L. Ye, M. Ingram and T. Pratt, "Broadband MIMO-OFDM wireless communications," *Proceedings of the IEEE,* vol. 92, no. 2, pp. 271-294, 2004.

[52] N. Giordano, College Physics: Reasoning and Relationships, Cengage Learning, 2009.

[53] A. Bazin, B. Jahan and M. Hélard, "Doppler Effect Reduction in an OFDM System Thanks to Massive MIMO," *IEEE Access,* vol. 6, pp. 38498-38511, 2018.

[54] M. Pössel, "Waves, motion and frequency: the Doppler effect," Einstein Online, 2011.

[55] P. Bello, "Characterization of Randomly Time-Variant Linear Channels," *IEEE Transactions on Communications Systems,* vol. 11, no. 4, pp. 360-393, 1963.

[56] P. Raviteja, K. T. Phan, Y. Hong and E. Viterbo, "Interference Cancellation and Iterative Detection for Orthogonal Time Frequency Space Modulation," *IEEE Transactions on Wireless Communications,* vol. 17, no. 10, pp. 6501-6515, 2018.

[57] T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Decision Feedback Equalizer for Zero-Padded OTFS Systems," *IEEE Transactions on Vehicular Technology,* vol. 69, no. 12, pp. 15606-15622, 2020.

[58] T. Thaj and E. Viterbo, "Low Complexity Iterative Rake Detector for Orthogonal Time Frequency Space Modulation," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, 2020.

[59] K. Sunil, P. Jayaraj and K. Soman, "Message Passing Algorithm: A Tutorial Review," *IOSR Journal of Computer Engineering (IOSRJCE),* vol. 2, no. 3, pp. 12-24, 2012.

[60] Y. Hong, T. Thaj and E. Viterbo, Delay-Doppler Communications: Principles and Applications, Academic Press, 2022.

[61] MathWorks, "What is MATLAB?," MathWorks, [Online]. Available: https://www.mathworks.com/discovery/what-is-matlab.html. [Accessed 10 August 2023].

[62] O. Filth, "Block diagram of an ideal OFDM transmitter," Wikipedia, 2006.

[63] O. Filth, "Block diagram of an ideal OFDM receiver," Wikipedia, 2006.

[64] GeekforGeeks, "Difference between Doppler Effect and Doppler Shift," GeekforGeeks, 2023.