MQP 17196

Customized NS-3 Simulation for Bandwidth Assessment of Rural Communities

A Major Qualifying Project Report:

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Bachelor of Science

in

Electrical and Computer Engineering

by

Alexandra Maslen

Mara Nunez

Date: April 6, 2021

APPROVED:

Professor Alexander Wyglinski, Major Project Adviser

Professor Shamsnaz Bhada, Co-Adviser

Abstract

In the year 2021, there are still millions of Americans that do not have a reliable broadband internet connection or lack a connection entirely. This report presents the research and building of a 4G LTE network that future teams can further update and advance as the demand for internet increases. The goal of this major qualifying project is to create the connection setup between base stations and user equipment with set bandwidths. The team attempted to create this connection with many different programs but settled with using the NS-3 network simulator to achieve this goal. This project contributes a starting point for ongoing research and adaptations of 4G LTE networks, through the tutorial chapter and methods.

Acknowledgments

We would like to thank our advisors, Professor Wyglinski and Professor Bhada, both of whom had patience with us and provided assistance and insight to this topic. Both gave us guidance and the confidence to complete this project. We would also like to thank Kuldeep Gill, a graduate student, who provided us with a base code for the project. He also guided us when working with a new and unfamiliar program. His assistance and knowledge of NS-3 coding was greatly appreciated.

Authorship Page

Both authors Alexandra and Mara contributed an equal amount of time towards the project and report completion.

Executive Summary

The world relies on the internet to connect to all aspects of life. However, people still lack a basic connection even in their own home. Rural areas of the US such as Missouri, where the land is occupied by farmland, families still have no internet connection in their household, even in the year 2021. In an environment where an unexpected pandemic impacted the world and forced people to work from home, internet access became an obstacle for families to function and support themselves during these difficult times. There is a need for far better network connections, and this project aims to find a starting block to answer these demands.

The team coded two households to one base station in a 10 second time frame, which can be seen in Appendix A. The ten second time frames allowed the team to run the code multiple times with different bandwidth each run, and obtain results quickly. To construct the code, the main components included were lteHelper, ListPositionAllocator, MobilityHelper, and the PointToPointHelper. These helpers create the LTE internet, install the internet to the eNodeB and UE, set the coordinate positions of the eNodeB and UEs, and establish the constant positional model of the UEs. The PointToPointHelper becomes the most important component because this is where the data rate is set and controls the total bandwidth supplied to the base station.

The bandwidth was changed six times in order to see the behaviors of the uplink and downlink speeds when the data rate was set to 20 Mbps, 10 Mbps, 9 Mbps, and 5Mbps, . Table A shows the values of the physical resource block (PRB) that sets the values of the uplink and downlink bandwidths and the corresponding frequencies. Even with the small error in the code

that does not allow the downlink speeds to exceed ~5Mbps, the behaviors are as expected. The downlink and uplink rates did not affect each other, allowing the ability to run each PRB value at each uplink and downlink bandwidth only one. Appendix D contains all set data rates and the corresponding uplink and downlink speeds results

Table A: PRB Values

PRB	**MHz**
6	1.4
15	3
25	5
50	10
75	15
100	20

At the conclusion of the project, a recommendation is made for the future project group. The next step to continue advancing the LTE system is to focus on setting individual bandwidths for the households and running them simultaneously. The idea is that the households do not have the same demand rate, one may have a family of 5 with teenagers and the other has an elderly couple. Set the data rates so the households receive the amount of data they use, and nothing more. The project will continue to change as technology continues to advance, but once the starting blocks are set the project can adapt in order to meet demand.

Table of Contents

Table of Figures

Table of Tables

1. Introduction

1.1. Motivation

In the year 2020, the COVID-19 virus became a worldwide pandemic with a significant impact on families, businesses, and academia. To prevent further spread of the virus and keep everyone safe and healthy, the world was put on lockdown, which forced everyone to work from home. This left many people in a predicament where they did not have a stable internet connection. Currently, about 14 million Americans have zero access to the internet and about 25 million have an unreliable and slow internet connection [1]. Currently, the estimated cost to provide broadband to all Americans is about $80 billion. [2]. This value would also vary depending on the type of broadband that is installed.

The world relies on the internet ever since the smartphone was created. The impact of COVID-19 made it abundantly clear that the "current system for supporting access to high-speed broadband has failed" [2]. Increased numbers of individuals working from home, created connectivity challenges for struggling families. The lack of internet access has created a struggle for many families, while trying to support oneself during such difficult times.

1.2. Current State of The Art

The outdated, currently designed broadband was meant for a telephone-oriented world, leading to the problem of needing to remove all the old technology and installing new internet-oriented technology [2]. The most common and easily accessible broadband connections in rural communities are Satellite and DSL [1]. Easy to access technologies, do not necessarily mean a good connection. Satellite and DSL tend to not give subscribers the bandwidth their lives may demand. DSL and Satellite are the oldest forms of any connection, both are explained more in the tutorial chapter. DSL has not been updated to meet the new internet economy demand. Satellite cannot supply an adequate connection unless weather conditions are perfect. A study showed that when it came to matching advertised internet connection speeds, DSL operated at significantly lower speeds, and Satellite speeds continued to drop as " new customer additions and increased usage per customer cut into current capacity" [3]. These broadband connections are not designed for the new and improved technological world. The household bandwidth demands are too high for these types of broadband to properly deliver, further stating the need for new technology.

1.3. Project Contributions

The goal of this project is to create a starting block for a 4G LTE network simulation that connects the base stations and users with set bandwidths. The idea is to have this basic setup completed and working, so that future project teams may continue adding new and improved features, in order to one day have a scalable working network that could be implemented to answer the rising high-speed internet demand.

An accurate LTE network simulation could provide data to help with the advancement of these networks being set up in rural areas. The simulation could potentially help identify the optimal locations for the base stations, which in turn could minimize the cost to set up an LTE network. The financial impact would allow many more areas to have access to an LTE network and broadband internet speeds.

2. Tutorial

2.1. Broadband Network

A rural broadband network is what gives people their internet at home. Currently, the number of people lacking a broadband connection of at least 25Mbps/3Mbps is about 21.3 million [4]. The definition of a broadband network is a "high-speed Internet access that is always on and faster than the traditional dial-up access," yet still so many people do not have this connection. It will cost almost 19 billion dollars just to connect "every unserved rural school, library, health provider and community college" [5] that doesn't already have a high-speed broadband connection. However, this number does not take into account households. To provide connections to all rural communities could potentially have a significant financial impact on businesses, families, or the government. However, there are many different forms of broadband networks in place; such as digital subscriber line (DSL), cable, fiber, wireless, satellite, and broadband over powerlines (BPL) [6]. With each new method, an opportunity might present itself that will help provide rural broadband access to all.

A digital subscriber line or DSL "is a wireline transmission technology that transmits data faster over traditional copper telephone lines already installed to homes and businesses" [6]. The connection is accessed through the use of standard telephone lines, but that makes it "restricted in bandwidth and data rate" [7] since the lines were only made for voice signals. There are two types of DSL transmission technologies, Asymmetrical Digital Subscriber Line (ADSL) and symmetrical Digital Subscriber Line (SDSL). ADSL is "used primarily by residential customers … who receive a lot of data but do not send much" [6]. This outdated technology does not support applications such as Skype or video gaming, applications that are widely used today.[8]. SDSL on the other hand is “used typically by businesses for services such as video conferencing, which need significant bandwidth both upstream and downstream” [6]. ADSL is the technology type of DSL that is limited to rural areas. Even if ADSL becomes fully available to all, it does not have a strong enough or big enough bandwidth that would meet the household needs.

In [9], the article indicates that “the superior technology to deliver Internet and network services is optical fiber that carries a communications signal from an operator’s equipment all the way to a home, business or enterprise.”. As the demand for internet continues to rise, fiber optic becomes more popular because it allows for much faster internet connection speeds while also going for longer distances than other options [9]. The article also indicates that the availability of fiber connections is still small because of the high installation costs [11]. Fiber installation can cost from $1 to $6 per foot, and while the cost of the material is easy to calculate, estimating the installation costs of fiber-optic internet is more complex [10]. To complicate matters further, when the fiber network is connected to someone’s home, the internet speeds can be higher than what the receiver actually needs, leaving them to pay more for something they do not use to its fullest potential [11].

A fiber line works by using light for network transmission, instead of data transmission over a frequency. This is why a fiber connection allows for faster internet speeds. With the speed of light it “enables speeds up to 1,000 Megabits per second on fiber-optic networks -- almost 100 times faster than the US broadband average of 11.7 Megabit per second” [11]. It is important to note that even though fiber seems like the best option for internet connectivity, businesses are not willing to invest in this technology, while DSL is still available. As technology is improved and developed, the copper wired broadband may be replaced with fiber optic cable, which would decrease the cost of available networks. However, it is also important to note that although this technology is available, it will take a significant time to make the changes to aging technology available today.[11]. Therefore, other network connections such as wireless broadband could be more feasible for trying to supply people with internet.

Using a radio link connection, wireless broadband can be connected to a home or business whether it is a fixed location or mobile. With similar speeds to that of DSL, wireless broadband will be used in more remote locations rather than DSL. Focusing on a fixed location, wireless broadband internet often requires a direct line-of-sight from the transmitter to the receiver; line-of-sight will be explained more later. Wireless network services “have been offered using both licensed spectrum and unlicensed devices” [6], meaning that it could be provided by a licensed service provider that set up the system. Rural areas that do not have licensed service providers have “thousands of small Wireless Internet Services Providers" supplying a connection of about 1 Mbps to unlicensed devices [6]. Because those connections are from unlicensed devices, the connection might not always be reliable.

Satellite broadband connection is another form of wireless broadband. Satellite broadband connection is identified as satellites orbiting the earth are the connection. Unlike other broadband connections, satellite connections are a bit more challenging. Weather, line-of sight, and latency are just a few of the issues related to satellite connectivity. Those factors have an effect on the downstream and upstream speeds [6]. Weather conditions, such as clouds, can prevent a direct line-of-sight to the satellite and cause a delay, which creates a higher ping rate.

Satellite broadband providers implemented a policy that states that there is only so much bandwidth that can be used a month before the connection starts to slow down [12]. However, because satellites rotate around the earth, it has the ability to provide internet access to isolated and unreachable areas [13]. Although this is a viable option, the satellite bandwidth does not meet current societal demands.

Broadband over powerlines or BPL is “the delivery of broadband over the exciting low- and medium-voltage electric power distribution network” [6]. This is stating that the power lines connected to people’s homes that supply their electricity can also be the source of internet. There is a cost benefit of using a resource developed in 1928, to provide a reliable connection to homes. However, the demand for faster internet connection suggests that the 1928 energy cable no longer can support the current internet demands. In [14] global trial, assessing BPL initiative, supports the conclusion that, “the technology is not viable as a means of delivering broadband internet access.” BPL was a reliable source of internet connection during a period of low demand. However, in today's society where the need for data access is constant and demanding, BPS is not a viable solution for future development.

Rural broadband has become “essential to modern agriculture.” For farmers and ranchers, it has become their source of connecting to “commodity markets, communicate with customers, and access new markets around the world” [15]. Currently, only a few areas actually have access to rural broadband, but there are plans to further expand. The demand for rural broadband is high now since more people are in need of it [16].

2.2. Long Term Evolution (LTE)

Long Term Evolution, better known by its acronym LTE, is the leading technology that enables the use of the fourth generation of mobile networks known as 4G [17]. 4G LTE was created as an improvement upon 3G that allowed for moving larger packets of data to create a more streamlined service with reduced latency. LTE typically operates at frequencies between 700 MHz and 2.6 GHz. The lower frequencies allow the signal to travel a further distance, while the higher frequencies provide faster speeds [18]. LTE can offer speeds of up to 300 Mbps downlink (DL) and 75 Mbps uplink (UL), which would be qualified under the broadband definition [17]. As of 2020, the global LTE population coverage was 83%, including being used by the four largest mobile network carriers in the US [19].

LTE works by connecting user equipment (UE) to the base stations, called eNodeB, which performs resource management, and this is all done through the Evolved UMTS Terrestrial Radio Access Network (E-UTRAN) And Evolved Packet Core (EPC), both will be further explained later. Each UE can connect to one base station at a time, but can also be passed along to different base stations via handovers [20]. A handover is a procedure that allows a UE to remain connected to the network while moving between base stations. The base stations operate the handover decisions through measurements sent from the UE. The handover occurs when a better network is found for the UE. Handovers are important for allowing user mobility within an LTE network [21].

Such complex LTE modules can be created because the LTE system architecture consists of two components, the E-UTRAN and the EPC. The E-UTRAN allows for “high spectral efficiency, high peak data rates, short round trip times as well as flexibility in frequency and bandwidth" [22] and is where the UEs and eNodeBs are comprised [23]. EPC is the core network of the LTE system and it relies on circuit-switching (CS). Circuit-switching is the establishment "between the calling and called parties" of the network [24]. Comprised in the EPC is the Mobility Management Entity (MME), the Serving Gateway (SGW), the Packet Data Network Gateway (PGW), the Home Subscriber Server (HSS), the Access Network Discovery and Selection Function (ANDSF), and the Evolved Packet Data Gateway (ePDG). Descriptions of all these components can be found in [23]. Without these components in the architecture of the LTE system this project would not be successful.

2.3. Wireless Propagation

Wireless propagation is the transmission of signals from a transmitter to a receiver. Multipath propagation is a form of wireless propagation that occurs when the signal from a transmitter takes multiple paths to the receiver. These signals may go through reflection, diffraction, or scattering. This creates multiple paths that arrive with different strengths in different amounts of time. Generally, the longer the path, the weaker the signal will be when it arrives. Multipath propagation is useful when there is no direct path for the signal to travel [25]. As a signal propagates it incurs path loss. Path loss is how much signal strength is lost as it travels between the transmitter and the receiver. Free-space path loss can be defined by the equation:

$$PL = \frac{(4\pi\ f\ d)^{\ \ 2}}{c^2} \tag{1}$$

Where f is the frequency, d is the distance from transmitter to receiver, and c is the speed of light.

With propagation there are two different types of transmission paths, line of sight and non-line of sight. Line of sight (LOS) occurs when there is no obstruction between the transmitter and receiver, which for LTE would be the base station and the user. LOS is the ideal condition for transmitting as the transmission can follow the shortest path to the receiver via a straight line [26]. LOS is demonstrated in Figure 2.1 with one base station having line of sight to three different households.

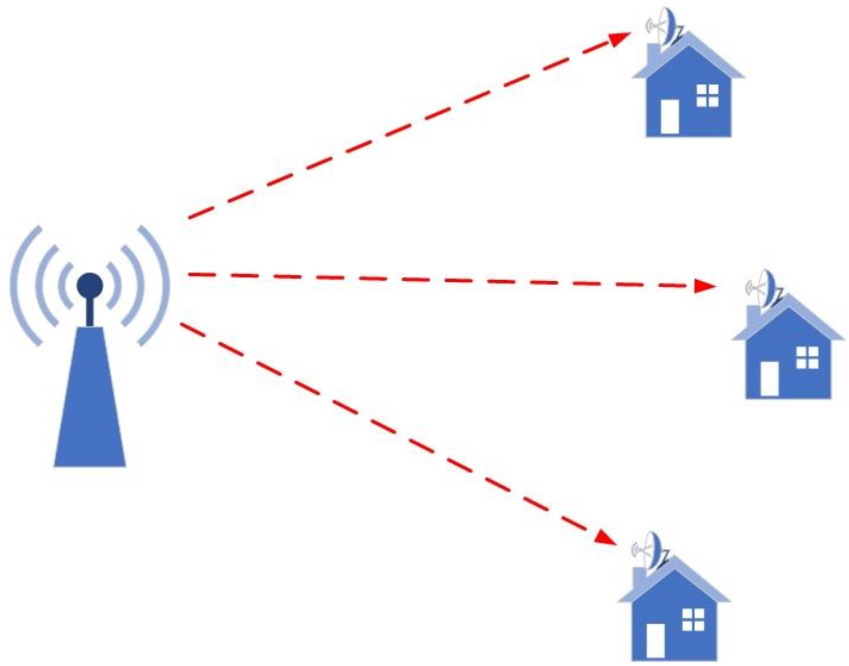

Figure 2.1: Line of Sight

Non-line of sight (NLOS) occurs when there is an obstruction directly between the transmitter and receiver. This obstruction creates a situation where the direct straight-line transmission path cannot be used. This would be solved with multipath propagation allowing the transmission to arrive to the receiver through one of the reflected paths [26]. NLOS is demonstrated in Figure 2.2 where a large building obstructs the direct path from the base station to the house. A path that has reflected off another large building is the path that would be used to get the signal to the house.

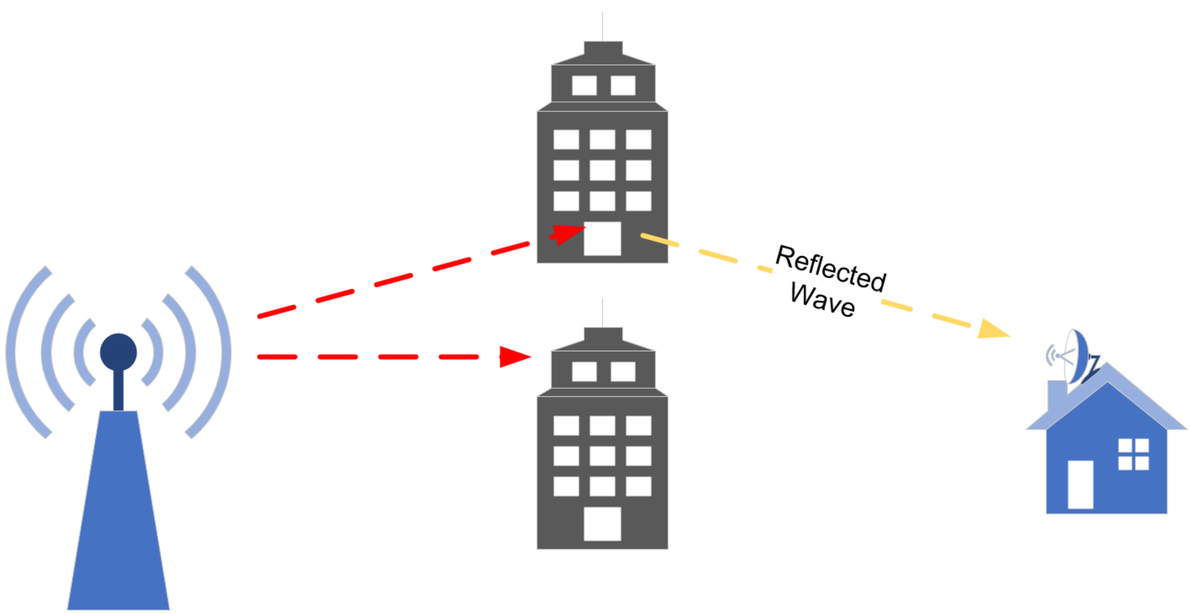

Figure 2.2: Non-Line of Sight

2.4. Simulations

This project relied on programming to make the end results possible. There were two simulators that the team looked into using for this project, Matlab and NS-3. The process to find which one would work best was trial and error. There are many requirements that the simulators needed to meet for them to be used, so that the team may properly create a working network.

Matlab is not a simulator but rather a programming platform that can analyze data, develop algorithms, and create models and applications. Matlab allows for ideas from research to be produced and deployed "to enterprise applications and embedded devices, as well as integrating with Simulink and Model-Based Design" [27]. This platform comes with many toolboxes, such as an LTE Toolbox, that provide the functions and algorithms that allow the users to properly create a project. With the LTE Toolbox, proper end-to-end communications links can be created [28]. This program, however, has a challenge of allowing for multiple users to connect to one base station making it not the best choice for the project.

NS-3 is a network simulator that is for discrete-event internet systems. The purpose of ns-3 being created was for there to be "a free and open-source simulation environment suitable for network research" [29]. Using base models from the ns-3 web page, it is possible to download the basic setup of an LTE network that can later be manipulated for any project. NS-3 is a better choice than other simulators for this project because "it is a 4G wireless communications standard developed by the 3rd Generation Partnership Project that's designed to provide up to 10x the speeds of 3G networks" [30]. NS-3 also allows the user to look at the network's upload and download links for more information, allowing the user to know how well their program is running and if it needs to be improved. NS-3 allows for many more commands and delivers better results than any other program the team has looked at making it the best choice for the project.

The process to set up the NS-3 simulator is all done in the command terminal on the PC. The setup time does take about an hour, but all commands that are needed can be found in [31]. There are a few ways to install the NS-3 simulator that can be found in this source, but the best way is to start at “Building with build.py”. Before hitting run on each function the command “sudo” needs to be before it. More specifically the command should be “sudo ./build.py --enable-examples --enable-tests” and then hit enter to run. Continue to do this one line at a time until finished at the debugging section of [31]. Once the setup is done and NS-3 is working correctly then the process of working on a project can begin. There should now be folders of examples of code and everything just downloaded in the file explore on the pc. Coding will all be done in a .cc file written in C++. When it comes time to run the code in NS-3 the directory is pointing to the location of the .cc file. For example, a lte.cc file is located in a scratch folder within the NS-3 folder library, see Figure 2.3 for the location specifics.

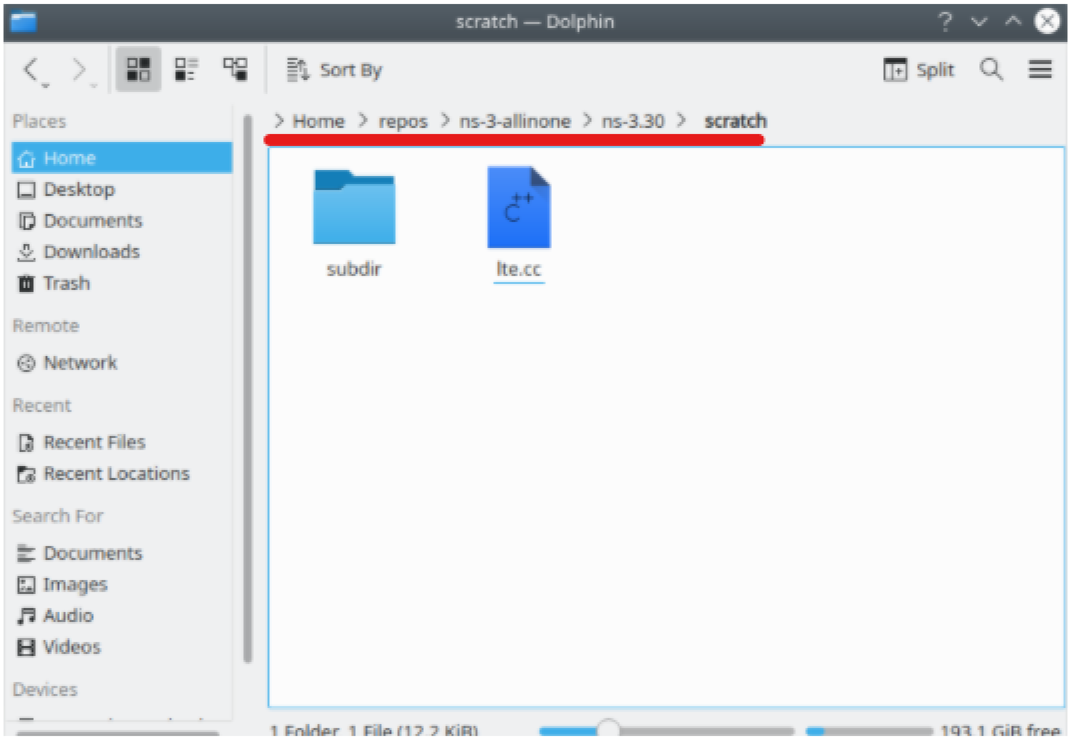

Figure 2.3: Folder Library Example

When running in NS-3 the directory needs to be connected to the files underlined in Figure 2.3, see Figure 2.4 for NS-3 directory example.

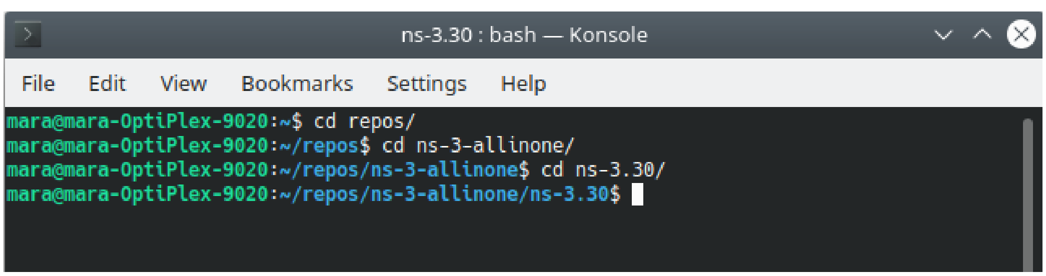

Figure 2.4: NS-3 Directory Example

The final step to run the lte.cc file is to use the command “sudo ./waf --run scratch/lte” and hit enter to run. The ./waf command only works within the ns-3.30 folder, so the run command must point to the scratch folder. If there is more than one .cc file within the scratch folder all files within the folder will be compiled when any one of them is run.

To understand a basic LTE setup in NS-3 see [32] for example code and detailed descriptions. Run the example code and understand the output that is obtained to be able to run more complex LTE setups later.

2.5. Chapter Summary

In summary this chapter provides all the information needed to achieve a basic understanding of the varying implementations of a rural broadband connection. This section covers the physical network connections, the characteristics that fabricate the network system, and LTE architecture. Understanding each part individually helps understand the setup of a network as a whole.

3. Proposed Approach

3.1. 30 House Layout

The original goal was to simulate a network with one base station providing LTE to 30 households all within a 3-5 mile range. The households' network usages would be set differently to represent different possible family situations. Possible household usages could represent a family with teenagers who all have multiple devices would show more usage throughout the day. Both parents would be using the network to work from home, while the teenagers would be using the network to attend school. This would be one of the higher load households. Then another household would be a retired couple, where they do not use as many devices and do not need to work during the day. The other 27 households would be set to usages that reflect different household situations. Some household usages would be similar in their usages. The total bandwidth that the base station provides to each household would be set at different values to see how that affects the throughput at each household. The total bandwidth would be set above the total usage of all the households combined, as well as below what all the households combined would use. This concept can be seen laid out in Figure 3.1. Each total bandwidth would be run for one hour simulations 24 times to simulate the change in bandwidth over a 24 hour period.

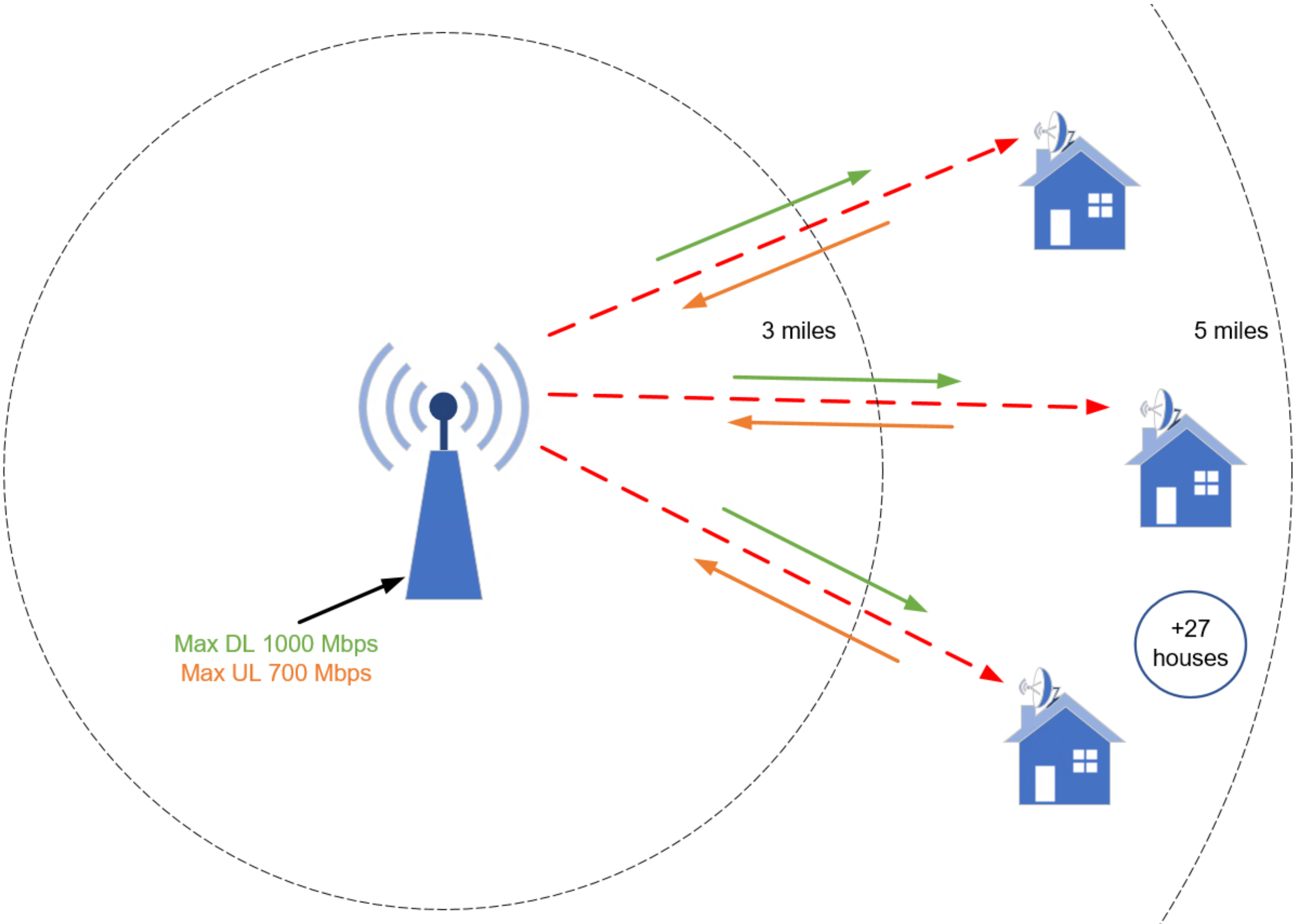

Figure 3.1: 30 Household Concept

3.2. Revised Approach

After working within the NS-3 Environment, it became apparent that the simulation times took significantly longer than real time. For example, a 10 second simulation with 30 households would take a few minutes to run. This made the original approach unrealistic for the time constraint. Running a simulation to get enough data to represent a day would take multiple days to simulate. This becomes an extra large problem since the simulation would need to be run multiple times at different total bandwidths. With keeping this in mind, the new approach moved on to that as seen in Figure 3.2. There would now be only 2 households instead of 30. The goal was separated into 3 parts. The first goal, labeled as 1 on Figure 3.2, was to control the total bandwidth provided to the base station. The second goal, labeled as 2 on Figure 3.2, was to control the bandwidth provided from the base station to the households. The third goal, labeled as 3 on Figure 3.2, was to control the household usages at real time within the simulations. This means the usages would change for each household as the simulation runs. This simulation would also be run for shorter amounts of time at 10 seconds. This allows for more versions of the simulation to be run with different parameters set.

3.3. Chapter Summary

This chapter outlines the team's proposed approach for creating a network simulator. It details the first idea for the concept of the simulator, as well as the second idea after learning more of the capabilities of NS-3. The second approach is the concept that will be outlined in the methodology section.

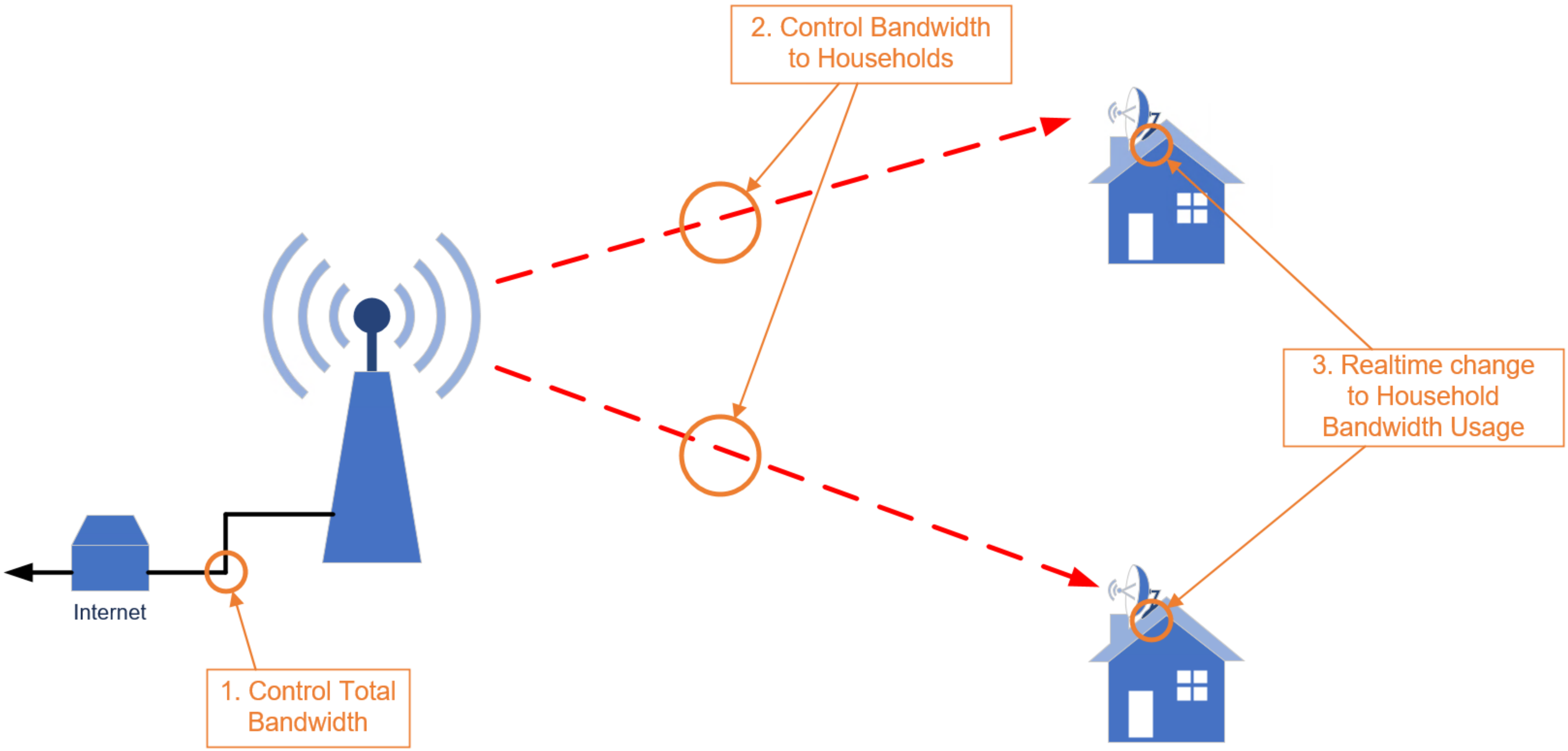

Figure 3.2: Revised Approach Concept

4. Methodology

4.1. Matlab

Matlab was the first program used to understand the simplest type of a multipath channel; a two ray propagation channel. With the two-ray channel, it gave the ability to simulate a line-of-sight propagation channel and calculate the variations of path loss in a system, the code can be seen in Appendix C. From doing such a simple network set up the understanding of the different variables that can affect the efficiency of a network became very clear. Many variables need to be taken into account when setting up even the simplest of multipath channels, things such as temperature and air pressure can affect the strength and speed of a connection. However, the team started with creating the two-ray channel in perfect conditions as a starting point that could be further advanced. However, after further understanding of Matlab's capability of coding a more complex network, the team sought out other programs more advanced; more detailed explanation as to why will be stated in the results.

4.2. NS-3 Network Simulator

NS-3 was selected to be used for the network simulation. Initially, the example codes provided from NS-3 were run and analyzed to learn the basics. Once the examples were understood, a simple code for a wired network was created to begin to understand the environment of NS-3. The simulation had ten nodes connected to a server. The code for the ten node simulation can be found in Appendix B. The nodes connected to the server with a point-to-point connection using NS-3's PointToPointHelper. The code was revised and shortened for a better understanding of the environment. The code was also made to be expandable to add additional nodes to keep future expansion in mind. This process allowed for the learning of fundamentals within NS-3 that would be used to implement the final wireless network.

4.3. NS-3 5G Module

When researching to properly code a 4G network a 5G module, created by a research institution named Centre Tecnologic de Telecomunicacions de Catalunya (CTTC) which is based in Barcelona, was discovered. To be granted access to the 5G module code the company asks that an email titled "Access to 5G-LENA code release" be composed and sent to info@5g-lena.cttc.es. In this email the following question must be answered:

1. Who are you?
2. Why do you need the software?
3. Are you from academia (in that case, the name of the institution is appreciated) or from industry?
4. Your Gitlab name (the part after the @. To obtain one, please register in https://www.gitlab.com).

Once the team was granted access a new goal was set for the project, and the process of understanding the commands within the code was the next task to complete. The 5G module was installed into NS-3 by following the steps of installation stated in GitLab files. However, an obstacle quickly became clear, there is little knowledge about 5G in the world and when it came to the code there were many functions that were unknown to the team. The team set a goal of two weeks to understand the commands within the code, with the condition that if there was still a lack of understanding then there needed to be a backup plan. After further discussion with the creators of the 5G model, the team's conclusion was to abort this option and execute our contingency plan.

4.4. NS-3 LTE Module

The NS-3 LTE module was selected to be used for the wireless network simulation after the 5G module was ruled out. For the LTE module, the process started the same as for the wired NS-3 simulation by running the LTE examples. This allowed for the additional understanding of the LTE specific commands. A basic LTE code was provided by WPI graduate student Kuldeep Gill. This code would then be adapted for the purposes within this project. The original code included moving targets and handovers which were not needed for this project. The core LTE connections of the given code were adapted into the new code with the revision of a constant mobility model.

The model was to consist of the layout from Figure 3.1, with 30 houses as the UE's within a 3-5 mile radius of one base station. The initial code began with only one house connected to the base station that ran for 1050 milliseconds as a proof of concept for a simple simulation. The code did not have any limitation on the bandwidth of the base station or a set usage for the node. The code's main components included the lteHelper, ListPositionAllocator, MobilityHelper, and the PointToPointHelper. The lteHelper allowed for the installation of the LTE network onto the eNodeB and UE. It then also allowed the eNodeB and UE to be attached to each other. This would also allow for expansion with additional eNodeBs where each UE could connect to a different eNodeB if necessary. The ListPositionAllocator allows for setting the location of each

UE and eNodeB. The positions are entered as coordinates and then converted into kilometers to be read by the positional allocator. Upon the expansion of the code with more UE nodes this step will be more important for the placement of the houses within the set radius. The MobilityHelper was necessary for setting the constant positional model. Since each UE was a house, there was no movement necessary. The PointToPointHelper is what creates the internet. This supplies the internet that is going into the base station to be supplied to the houses. The data rate set within the PointToPointHelper can be used to control the total bandwidth supplied to the base station. This is the value that would represent number 1 on Figure 3.2.

Once the one house simulation was working, it was expanded to the 30 house simulation. With the 30 house simulation, an error was thrown that can be seen in Figure 4.1.

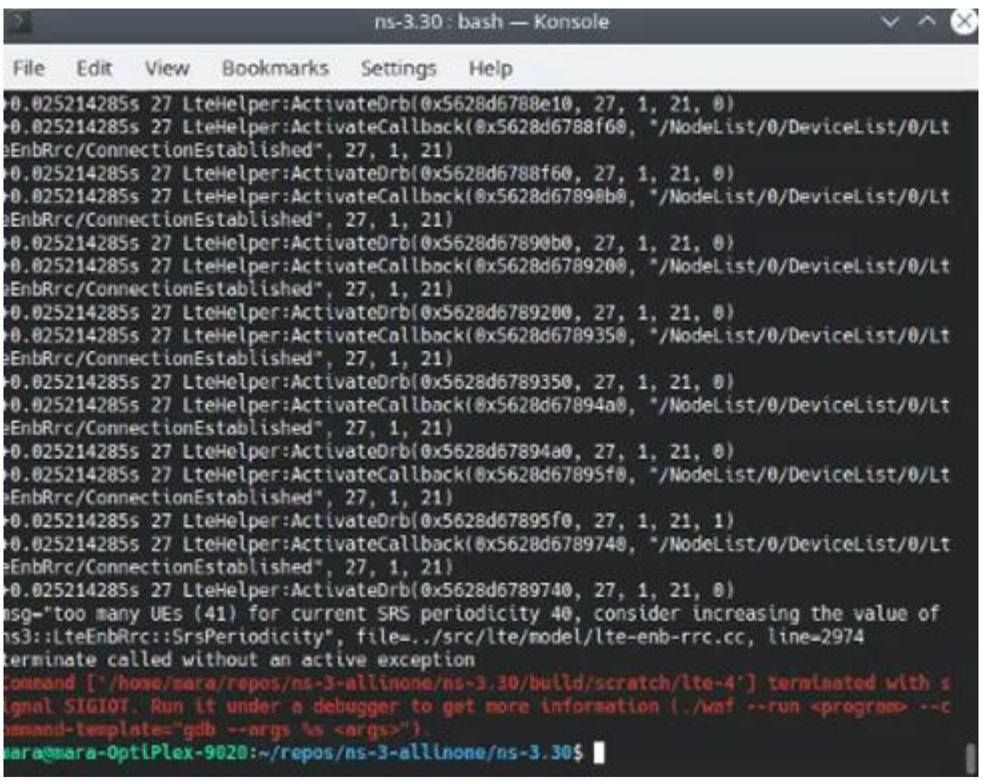

Figure 4.1: NS-3 SRS Periodicity Error

After troubleshooting the code and learning the error, it was discovered that the SRS periodicity was not set high enough in its default state. This value would have to be set within the code with the following line of code:

```
Config::SetDefault("ns3::LteEnbRrc::SrsPeriodicity", UintegerValue(320));
```

Once this value was set to 320, the code successfully ran with the 30 household nodes.

The original idea for the project was to run one-hour-long simulations with 30 households and show the varying network usage at each household for each hour. However, the simulation time within NS-3 is much greater than the real-time, where a second-long simulation could take a few minutes. Making 24 one-hour-long simulations not possible. This idea was replaced with a more feasible idea for run time. The simulation time would be for ten seconds with only 2 households rather than 30. This would allow the simulation to be run multiple times. Now that the simulation would be able to run many times, it could be run each time with varying bandwidth values. The new approach based on the capabilities of NS-3 is outlined in Figure 3.2.

Within the code, the RLC traces have been enabled. With the traces, the data for the throughput will be able to be extracted. The RLC traces file is broken up into uplink and downlink. Within each file, the number of bytes sent and received are tracked, as well as the time. With these two numbers, it is possible to calculate the bytes per second. With bytes per second, Mbps is able to be calculated by multiplying the bytes per second time $8x10^{-6}$. The code for this model can be found in Appendix A.

4.5. Chapter Summary

This chapter outlined the methodologies used to achieve the goal of the project. It details the changes and decisions that were made as the project progressed. The final design of the simulator was not the same as the original intention of the project.

5. Results

5.1. Matlab

The Matlab propagation channel gave the basic understanding and background knowledge needed to further code a more complex LTE network. When looking at the code in Appendix C, there are four different connection path losses that are coded. Path loss is the change in the strength of the network connection from the base station to the household.The first path loss calculated was L_total, the total path loss of all connection strength changes such as free space path loss. Lfs was just the general wired path loss from the base station to the household. Next was the L_2ray, which calculated signal path loss, and lastly was L_ref which calculated the free space pathloss. The team successfully obtained all variations of path loss within a LOS model, which can be seen in Figure 5.1.

Regardless of the values calculated, the team decided that Matlab was the least capable choice for the project because the program does not have a built-in LTE system nor does it allow for scalability. The team would have had to manually code the internet features from scratch and then start connecting to households, and the project time frame did not allow for that.

```
loschannel =

  phased.LOSChannel with properties:

          PropagationSpeed: 299792458
        OperatingFrequency: 3.5000e+09
         SpecifyAtmosphere: false
         TwoWayPropagation: true
                SampleRate: 1000000
     MaximumDistanceSource: 'Auto'


L_total =

  227.0107


Lfs =

  226.9671


L_2ray =

  109.2332


L_ref =

  113.4849
```

Figure 5.1: Total Loss, Path Loss, Signal Loss, and Free Space Path Loss

5.2. NS-3 LTE Module

After creating the code for the desired simulation, during testing it became apparent that the individual downlink speed for each UE did not exceed ~5 Mbps. This issue was troubleshooted for a substantial amount of time, but due to our time constraint no fix was ever discovered. The behavior of the eNodeB data rate changes was still shown below 5 Mbps. The selected data rates were: 20 Mbps, 10 Mbps, 9 Mbps, and 5 Mbps. These data rates were selected with the maximum downlink per a UE in mind. For the test there are two UEs. The first UE, labeled as UE 1, is located at the coordinates (3, -4). The second UE, labeled as UE 2, is located at the coordinates (3, 4). Each test consisted of running the code six times, once for each bandwidth value, with a different eNodeB downlink and uplink bandwidth each time. The downlink and uplink bandwidths were set by their physical resource block (PRB) values. A PRB is the smallest resource that can be assigned by the eNodeB scheduler. The available PRB values in NS-3 are 6, 15, 25, 50, 75, and 100. These six values were each set differently each time the code was run. The PRB values correspond to frequencies as seen in table 5.1.

Table 5.1: PRB Values

PRB	**MHz**
6	1.4
15	3
25	5
50	10
75	15
100	20

While testing, it was discovered that the downlink and uplink rates were not affected by the other. For example, if the downlink bandwidth was set to 25 PRBs, the downlink output would be the same if the uplink bandwidth was at 6, 15, or any of the other PRB values. Since the uplink and downlink do not affect the other's outcome, each PRB value only needed to be run once per data rate. The simulation was run six times at each data rate, once for each PRB value, each with the same value for uplink and downlink bandwidth for the eNodeB.

The first test was done with a 20 Mbps data rate. 20 Mbps was chosen because this was the theoretical highest throughput that could be set if the UE downlink was not capped at 5 Mbps. Below in Figure 5.2, the graph for the 20 Mbps simulation can be found.

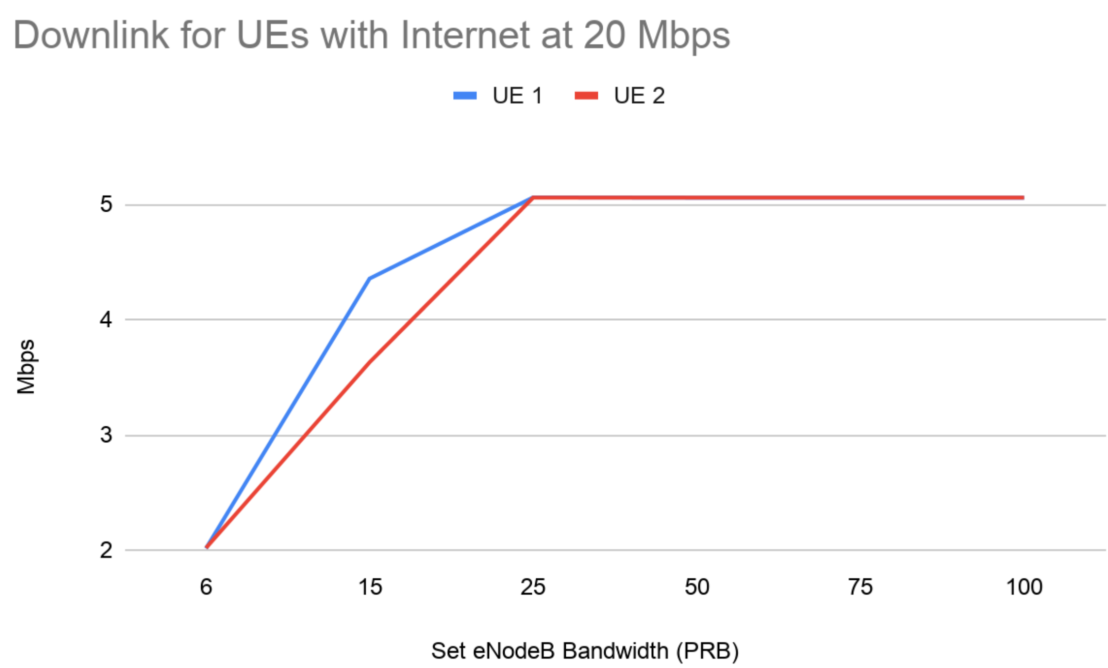

Figure 5.2: 20 Mbps Downlink Graph

The general behavior at the 20 Mbps data rate was as expected. At the lower DL bandwidth, the throughput speed is limited by the eNodeB DL bandwidth. As the eNodeB bandwidth increases, the throughput increases, until it levels off at 5 Mbps. UE 1 hits a higher throughput at a lower bandwidth than UE 2, however once they reach their max of 5 Mbps they are the same.

The second test was done with a data rate of 10 Mbps. This rate was chosen, because it is about the maximum data rate for the eNodeB to supply the maximum throughput that both UEs would use. Each UE uses a maximum of about 5 Mbps, so the maximum provided from the eNodeB would be 10 Mbps. The graph for 10 Mbps can be seen below in Figure 5.3.

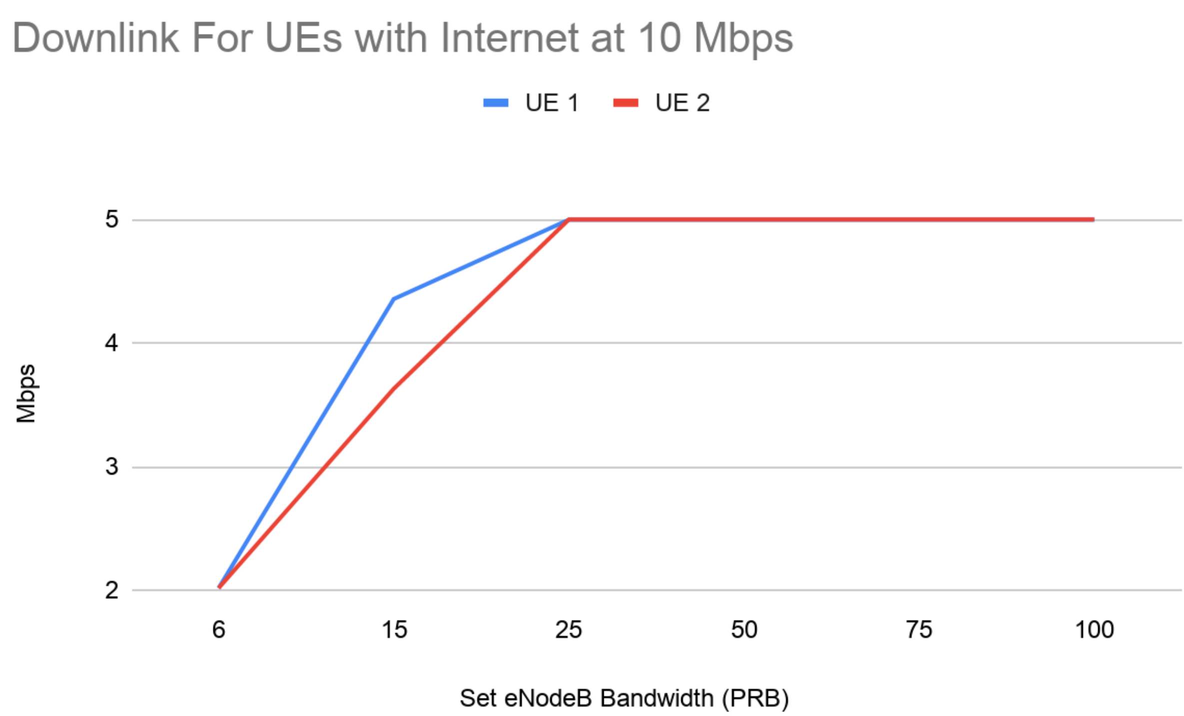

Figure 5.3: 10 Mbps Downlink Graph

The behavior for the 10 Mbps data rate was also as expected. It was extremely similar to the 20 Mbps data rate. The largest difference is that now neither UE exceeded 5 Mbps, as the total provided to both was 10 Mbps. While running the simulation with the 20 Mbps data rate, the throughput raises slightly above 5 Mbps. UE 1 has a maximum DL of 4.9970272 Mbps, and UE 2 has a maximum DL of 4.9987392 Mbps. Both of these throughputs occurred while the DL bandwidth was set to 25.

The third test was done with a data rate of 9 Mbps. This value was selected, because it is slightly below 10 Mbps. This was done to see what would happen to the maximum throughput values across all of the bandwidths. The graph for the 9 Mbps data rate can be seen below in Figure 5.4.

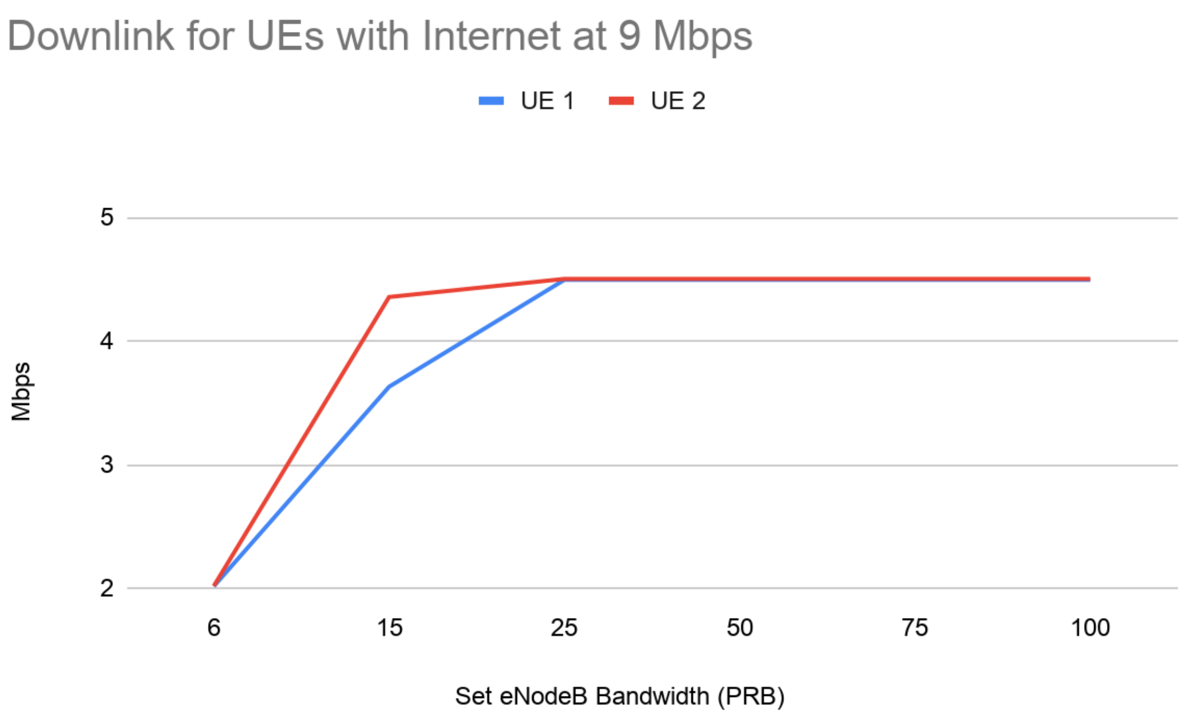

Figure 5.4: 9 Mbps Downlink Graph

The behavior for the 9 Mbps data rate was also as expected. Each UE now hit a max throughput of about 4.5 Mbps, instead of 5, which is half of the total bandwidth supplied by the eNodeB. This shows that each UE was weighed evenly. The eNodeB processed the information from both UEs and distributed the bandwidth evenly across both UEs. Additionally, during this simulation, UE 2 hit a higher value than UE 1 at the bandwidth of 15. For both the 20 Mbps and 10 Mbps simulation, this was the opposite. UE 2 also held a higher maximum throughput at 4.5019728 Mbps. While UE 1 had a maximum throughput of 4.494368 Mbps. Once again, both of these values occurred while the DL bandwidth was set to 25. This is the value at which the maximum throughput is first achievable, with not being capped by the bandwidth parameters that are set. After this, the value goes down slightly by less than 0.00003 Mbps due to already hitting the peak.

The fourth test was done with a data rate of 5 Mbps. This data rate was selected, because it is significantly lower than the maximum value needed for each UE. Instead of each UE being able to have a throughput of 5 Mbps, they now had to share a total of 5 Mbps. The graph for this test can be seen below in Figure 5.5.

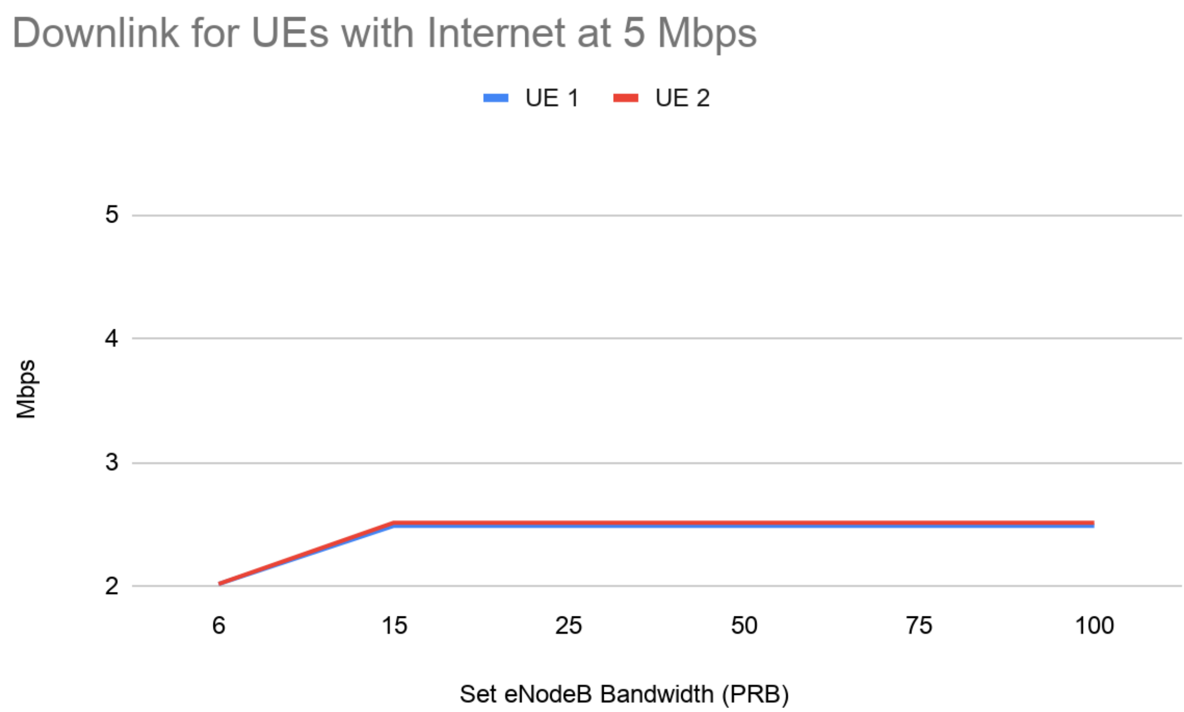

Figure 5.5: 5 Mbps Downlink Graph

This test once again, behaved as expected. Both UEs hit their maximum throughput at about 2.5 Mbps, exactly half of the supplied 5 Mbps. This shows the two UEs once again sharing the bandwidth evenly. This test behaved more similarly to the 9 Mbps test, in that UE 2 hit the higher data rate sooner than UE 1 did. UE 2 then stayed at a slightly higher data rate than UE 1. This can be seen in the zoomed in graph below in Figure 5.6.

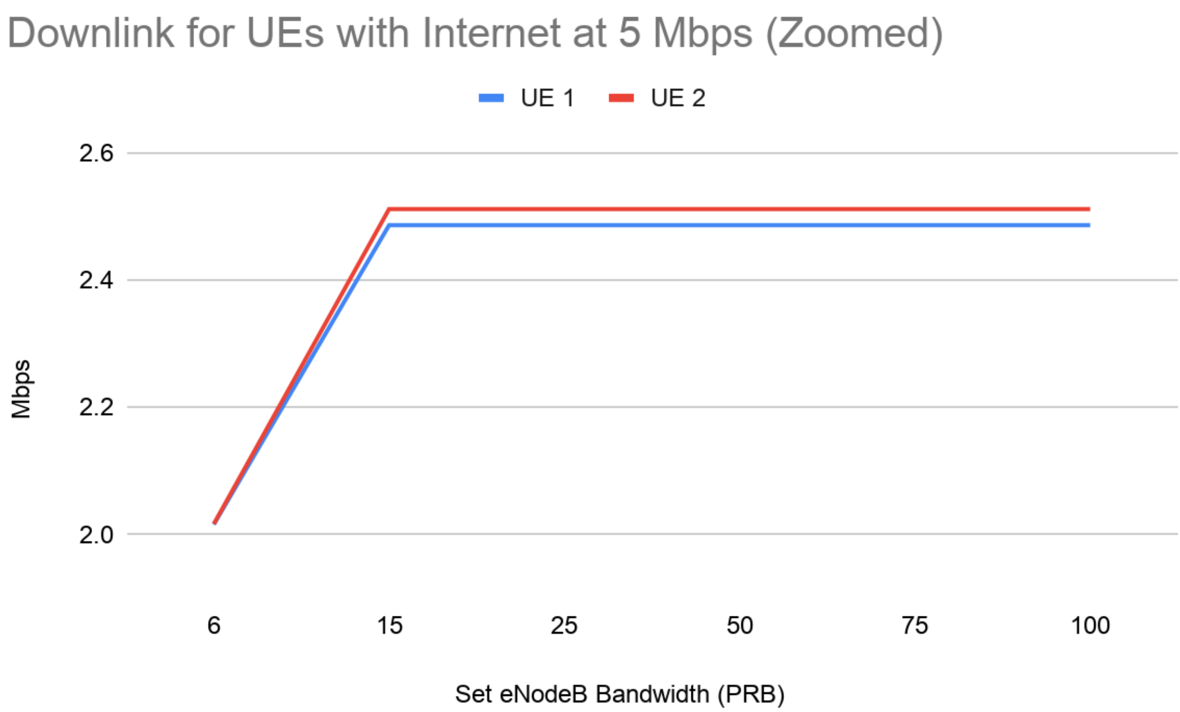

Figure 5.6: 5 Mbps Downlink Graph Zoomed

During this run of the simulation, UE 2 had a maximum throughput of 2.5116128. UE 1 had a maximum throughput of 2.4862704. These values both occurred while the bandwidth was set to 15, rather than 25 like the previous tests. The UEs always hit their highest throughput at the first bandwidth value which allowed for the maximum.

Throughout all of the simulations at different data rates, the uplink values remained consistent. This is due to the fact that the data rate attribute in the PointToPointHelper sets the data rate used for the transmission of packets only. This means that the value only affects the data sent to each UE from the eNodeB, which means it only affects the DL values. The graph for the UL throughput remained constant for all of the data rates. The UL graph can be seen in Figure 5.7.

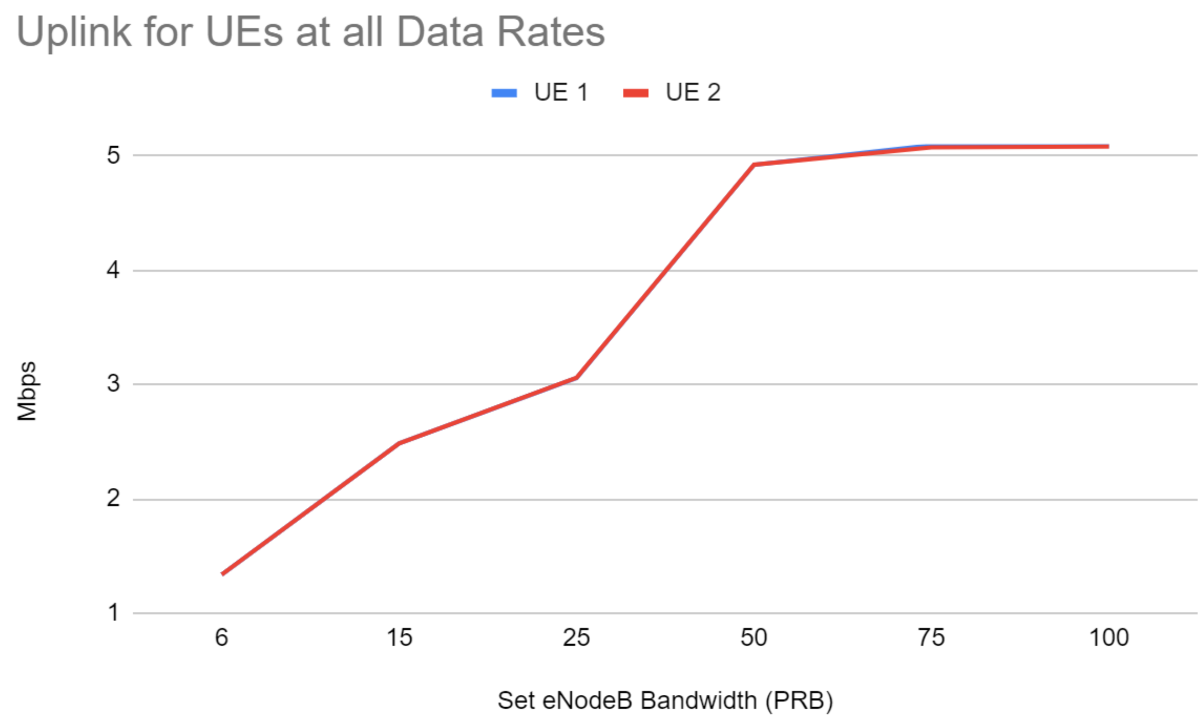

Figure 5.7: Downlink Graph at all Data Rates

For the downlink throughput, the maximum was also about 5 Mbps. Additionally for UL, both UEs were closer in value at every bandwidth. This is why the graph appears as almost one line.

The results for the most part showed the expected behavior. At the data rates that were above the maximum throughputs for each UE, UE 1 was favored. For the data rates that limited the maximum throughput for each UE, UE 2 was favored. The data outputs that formed all of the graphs can be found in Appendix D. For the three goals in Figure 3.2, the final simulation fully met goal number one. The second goal was partially met. It was met in the sense that the bandwidth sent from the eNodeB to all the UEs was set. However, each bandwidth was not controlled individually. The third goal was not met. This feature would have been very complex to add, and due to the time constraints it was not able to be added.

5.3. Chapter Summary

This chapter outlined the results from the methodologies that were used from Chapter 4. The final NS-3 simulation results were shown and analyzed to show the UE bandwidth behaviors. The eNodeB actively distributed the bandwidth evenly between the two UEs. It limited each UE's maximum throughput evenly when the total data rate was lower than the requested bandwidth for the UEs. The simulation behavior was all as expected and scaled accordingly with the set parameters.

6. Conclusion

6.1. Future Recommendations

This project can be expanded upon to become a very powerful tool for getting LTE to communities that lack broadband internet access. The simulation would need to be expanded to control the household usage at runtime. This is goal number 3 from Figure 3.2. Controlling the individual UE usage is possible, however, time did not allow for it to be done in this project. Additionally, the throughput being maxed at ~5 Mbps limited the testable bandwidths. Fixing this to allow for higher throughput would also be important to create a more accurate simulation.

To further improve the depth of the simulation, more households could be added as well as more accuracy to where the households are located. More households as UEs would more accurately represent a rural communities usage of the internet, and the behavior of the base stations delivery of throughput to each UE.

6.2. Final Conclusions

The goal of this project was to develop an LTE network connection between base stations and households while controlling the bandwidth. From the data rate parameter that was controlled, the uplinks stayed the same. While the dowlinks were affected by both the data rate and max bandwidth values. The implementation of an LTE network is far from being completed. However, this project establishes the beginning framework needed to be able to move forward with the future recommandation. Since the project proved to be more complex than originally, the next steps are exceptional for a future MQP because it provides a strong learning curve for future members to overcome.

References

[1] P. Holslin, “What Type of Internet Do You Have at Home?,” *HighSpeedInternet.com*, 19-Mar-2021. [Online]. Available: https://www.highspeedinternet.com/resources/ compare-internet-provider-types#:~:text=DSL%2C%20fiber%2C%20and%20cable%20are,customers%20in%20the%20United%20States.

[2] T. Wheeler, “5 steps to get the internet to all Americans,” *Brookings*, 27-May-2020. [Online]. Available: https://www.brookings.edu/research/5-steps-to-get-the-internet -to-all-americans/#:~:text=While%20at%20the%20FCC%2C%20we,to%20deliver%20broadband%20to%20everyone.

[3] J. Brodkin, “Millions in US still living life in Internet slow lane,” *Ars Technica*, 05-Dec-2016. [Online]. Available: https://arstechnica.com/information-technology/2016/ 12/millions-in-us-still-living-life-in-internet-slow-lane/.

[4] F. C. Commission, “2019 Broadband Deployment Report,” Tech. Rep., 2019. [Online]. Available: https://docs.fcc.gov/public/attachments/FCC-19-44A1.pdf

[5] D. Schaffhauser, “Cost to Connect Rural America: $19 Billion or Less,” *THE Journal*, 20-Feb-2018. [Online]. Available: https://thejournal.com/articles/2018/02/20/cost-to-connect-rural-america-19-billion-or-less.aspx.

[6]“Types of Broadband Connections,” *Federal Communications Commission*, 24-Jun-2014. [Online]. Available: https://www.fcc.gov/general/types-broadband-connections#:~:text=The term broadband commonly refers,Fiber.

[7] “Digital Subscriber Lines,” *Digital Subscriber Lines - an overview | ScienceDirect Topics*. [Online]. Available: https://www.sciencedirect.com/topics/ engineering/digital-subscriber-lines.

[8] T. Cooper, “DSL Internet In the United States at a Glance,” *BroadbandNow*, 21-Dec-2020. [Online]. Available: https://broadbandnow.com/DSL.

[9]“What is Fiber Broadband?,” *Fiber Broadband Association*, 15-Jun-2018. [Online]. Available: https://www.fiberbroadband.org/page/what-is-ftth#:~:text=The superior technology to deliver, and reliable than older technologies.

[10] T. Collins, “How Much Does it Cost to Install Fiber Optic Internet in My Building?,” *Business Telecommunications for the Washington DC Metro Area - Atlantech Online*, 15-Jan-2020. [Online]. Available: https://www.atlantech.net/blog/how-much-does-it-cost-to-install-fiber-optic-internet- in-my-building.

[11] T. Cooper, “Fiber-Optic Internet In the United States at a Glance,” *BroadbandNow*, 30-Mar-2020. [Online]. Available: https://broadbandnow.com/Fiber.

[12] “What is Satellite Internet?,” *Plug Things In*, 2021. [Online]. Available: http://www.plugthingsin.com/internet/satellite/#:~:text=Since you have to send,obstructions can affect your signal.&text=Satellite internet is relatively expensive.

[13] “What is Satellite Broadband?,” *ESOA*. [Online]. Available: https://www.esoa.net/broadband/what-is-satellite-broadband.asp#:~:text=With satellite broadband you can,in the most isolated areas.&text=A reliable solution - rather than,about 22,000 miles in space.

[14] M. Courtney, “Whatever happened to Broadband over Power Line?,” *RSS*, 15-Oct-2013. [Online]. Available: https://eandt.theiet.org/content/articles/2013/10/whatever-happened-to-broadband-over-power-line/.

[15] “Rural Broadband,” *American Farm Bureau Federation - The Voice of Agriculture*. [Online]. Available: https://www.fb.org/issues/infrastructure/broadband/.

[16] Congressional Research Service and B. E. Humphreys, Version 3. 2019.

[17] A. Ghayas, “What is LTE and how does it work? Commsbrief,” Commsbrief, 15-Aug-2020. [Online]. Available: https://commsbrief.com/what-is-lte-and-how-does-it-work/.

[18] N. Swanner, “LTE guide: what LTE is, how it works, and how it affects you,” Android Authority, 22-Oct-2013. [Online]. Available: https://www.androidauthority.com/4g-lte-guide-146347/.

[19] P. by S. O'Dea, “Mobile coverage by technology worldwide 2011-2026,” Statista, 19-Jan-2021. [Online]. Available: https://www.statista.com/statistics/1016292/mobile-coverage-by-technology-worldwide/#statisticContainer.

[20] Tutorialspoint.com, “LTE Network Architecture,” Tutorialspoint. [Online]. Available: https://www.tutorialspoint.com/lte/lte_network_architecture.htm.

[21] J. Agrawal, P. Mor, J. M. keller, R. Patel and P. Dubey, "Introduction to the basic LTE handover procedures," 2015 International Conference on Communication Networks (ICCN), Gwalior, India, 2015, pp. 197-201, doi: 10.1109/ICCN.2015.39.

[22] “LTE,” *3GPP A Global Initiative*, 2021. [Online]. Available: https://www.3gpp.org/technologies/keywords-acronyms/98-lte.

[23] "Full LTE architecture and components," *YateBTS*, 06-Jan-2021. [Online]. Available: https://yatebts.com/documentation/concepts/lte-concepts/#:~:text=A%20standard%20LTE%20system%20architecture,also%20known%20as%20an%20EPC.&text=the%20Evolved%20Universal%20Terrestrial%20Radio%20Access%20(E%2DUTRA).

[24] F. Firmin, "The Evolved Packet Core," *3GPP A Global Initiative*, 2021. [Online]. Available: https://www.3gpp.org/technologies/keywords-acronyms/100-the-evolved-packet-core.

[25] R. W. Heath, *Introduction to wireless digital communication: a signal processing perspective*. Boston: Prentice Hall, 2017.

[26] RF Wireless World, "RF Wireless World," LOS vs NLOS | Difference between LOS and NLOS wireless channels. [Online]. Available: https://www.rfwireless-world.com/Terminology/LOS-vs-NLOS-wireless-channel.html.

[27] "What Is MATLAB?," *MATLAB & Simulink*. [Online]. Available: https://www.mathworks.com/discovery/what-is-matlab.html.

[28] "LTE Toolbox," MATLAB. [Online]. Available: https://www.mathworks.com/products/lte.html.

[29] "What is ns-3," *ns*. [Online]. Available: https://www.nsnam.org/about/what-is-ns-3/.

[30] Ns3 Simulations, "LTE Simulation NS3," Ns3 Projects, 07-Apr-2020. [Online]. Available: https://ns3simulation.com/lte-simulation-ns3/.

[31] NS-3, "Getting Started," *NS-3 A Discrete-Event Network Simulator*, 29-Mar-2021. [Online]. Available: https://www.nsnam.org/docs/tutorial/html/getting-started.html.

[32] NS-3, "User Documentation," *NS-3 A Discrete-Event Network Simulator*, 29-Mar-2021. [Online]. Available: https://www.nsnam.org/docs/models/html/lte-user.html.

Appendix A: NS-3 Final Code

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/mobility-module.h"
#include "ns3/lte-module.h"
#include "ns3/applications-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/config-store-module.h"
#include "ns3/config-store.h"
#include "ns3/propagation-module.h"


using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("LenaX2HandoverMeasures");

void
NotifyConnectionEstablishedUe (std::string context,
                               uint64_t imsi,
                               uint16_t cellid,
                               uint16_t rnti)
{
  std::cout << context
            << " UE IMSI " << imsi
            << ": connected to CellId " << cellid
            << " with RNTI " << rnti
            << std::endl;
}

void
NotifyHandoverStartUe (std::string context,
                       uint64_t imsi,
                       uint16_t cellid,
                       uint16_t rnti,
                       uint16_t targetCellId)
{
  std::cout << context
            << " UE IMSI " << imsi
            << ": previously connected to CellId " << cellid
            << " with RNTI " << rnti
            << ", doing handover to CellId " << targetCellId
```

```
            << std::endl;
}

void
NotifyHandoverEndOkUe (std::string context,
                       uint64_t imsi,
                       uint16_t cellid,
                       uint16_t rnti)
{
  std::cout << context
            << " UE IMSI " << imsi
            << ": successful handover to CellId " << cellid
            << " with RNTI " << rnti
            << std::endl;
}

void
NotifyConnectionEstablishedEnb (std::string context,
                                uint64_t imsi,
                                uint16_t cellid,
                                uint16_t rnti)
{
  std::cout << context
            << " eNB CellId " << cellid
            << ": successful connection of UE with IMSI " << imsi
            << " RNTI " << rnti
            << std::endl;
}

void
NotifyHandoverStartEnb (std::string context,
                        uint64_t imsi,
                        uint16_t cellid,
                        uint16_t rnti,
                        uint16_t targetCellId)
{
  std::cout << context
            << " eNB CellId " << cellid
            << ": start handover of UE with IMSI " << imsi
            << " RNTI " << rnti
            << " to CellId " << targetCellId
            << std::endl;
}
```

```
void
NotifyHandoverEndOkEnb (std::string context,
                        uint64_t imsi,
                        uint16_t cellid,
                        uint16_t rnti)
{
  std::cout << context
            << " eNB CellId " << cellid
            << ": completed handover of UE with IMSI " << imsi
            << " RNTI " << rnti
            << std::endl;
}



int
main (int argc, char *argv[])
{
   LogLevel logLevel = (LogLevel)(LOG_PREFIX_ALL | LOG_LEVEL_ALL);

   LogComponentEnable ("LteHelper", logLevel);
   LogComponentEnable ("EpcHelper", logLevel);
   LogComponentEnable ("EpcEnbApplication", logLevel);
   LogComponentEnable ("EpcX2", logLevel);


   LogComponentEnable ("LteEnbRrc", logLevel);
   LogComponentEnable ("LteEnbNetDevice", logLevel);
   LogComponentEnable ("LteUeRrc", logLevel);
   LogComponentEnable ("LteUeNetDevice", logLevel);

    LogComponentEnable ("NoOpHandoverAlgorithm", logLevel);

  ConfigStore inputConfig;
  inputConfig.ConfigureDefaults ();


    std::vector<float> x_cor = {3, 3};
    std::vector<float> y_cor = {-4, 4};

    std::vector<float> enb_x = {0};
    std::vector<float> enb_y = {0};
```

```
    //converting to Kms...
    std::transform(x_cor.begin(), x_cor.end(), x_cor.begin(),
                   std::bind(std::multiplies<float>(), std::placeholders::_1,
1000));

    std::transform(y_cor.begin(), y_cor.end(), y_cor.begin(),
                   std::bind(std::multiplies<float>(), std::placeholders::_1,
1000));

    std::transform(enb_x.begin(), enb_x.end(), enb_x.begin(),
                   std::bind(std::multiplies<float>(), std::placeholders::_1,
1000));

    std::transform(enb_y.begin(), enb_y.end(), enb_y.begin(),
                   std::bind(std::multiplies<float>(), std::placeholders::_1,
1000));


  uint16_t numberOfUes = 2;
  uint16_t numberOfEnbs = 1;
  uint16_t numBearersPerUe = 6;

  double speed = 20;        // m/s
  double enbTxPowerDbm = 46.0;

  // change some default attributes so that they are reasonable for
  // this scenario, but do this before processing command line
  // arguments, so that the user is allowed to override these settings
  Config::SetDefault ("ns3::UdpClient::Interval", TimeValue (MilliSeconds (10)));
  Config::SetDefault ("ns3::UdpClient::MaxPackets", UintegerValue (10000000));
  Config::SetDefault ("ns3::LteHelper::UseIdealRrc", BooleanValue (true));

  // Command line arguments
  CommandLine cmd;
  //cmd.AddValue ("simTime", "Total duration of the simulation (in seconds)",
simTime);
  cmd.AddValue ("speed", "Speed of the UE (default = 20 m/s)", speed);
  cmd.AddValue ("enbTxPowerDbm", "TX power [dBm] used by HeNBs (default = 46.0)",
enbTxPowerDbm);

  cmd.Parse (argc, argv);
```

```
  Ptr<LteHelper> lteHelper = CreateObject<LteHelper> ();
  Ptr<PointToPointEpcHelper> epcHelper = CreateObject<PointToPointEpcHelper> ();
  lteHelper->SetEpcHelper (epcHelper);
  lteHelper->SetSchedulerType ("ns3::RrFfMacScheduler");

  lteHelper->SetHandoverAlgorithmType
("ns3::NoOpHandoverAlgorithm");//A3RsrpHandoverAlgorithm

  lteHelper->SetAttribute ("PathlossModel", StringValue
("ns3::FriisSpectrumPropagationLossModel"));
  uint8_t bandwidthdown = 100;
  uint8_t bandwidthup = 100;
  lteHelper->SetEnbDeviceAttribute ("DlBandwidth", UintegerValue
(bandwidthdown));
  lteHelper->SetEnbDeviceAttribute ("UlBandwidth", UintegerValue (bandwidthup));

  Ptr<Node> pgw = epcHelper->GetPgwNode ();

  // Create a single RemoteHost
  NodeContainer remoteHostContainer;
  remoteHostContainer.Create (1);
  Ptr<Node> remoteHost = remoteHostContainer.Get (0);
  InternetStackHelper internet;
  internet.Install (remoteHostContainer);

  // Create the Internet
  PointToPointHelper p2ph;
  p2ph.SetDeviceAttribute ("DataRate", DataRateValue (DataRate ("10Mb/s")));
  p2ph.SetDeviceAttribute ("Mtu", UintegerValue (9000));
  p2ph.SetChannelAttribute ("Delay", TimeValue (Seconds (0.010)));
  NetDeviceContainer internetDevices = p2ph.Install (pgw, remoteHost);
  Ipv4AddressHelper ipv4h;
  ipv4h.SetBase ("1.0.0.0", "255.0.0.0");
  Ipv4InterfaceContainer internetIpIfaces = ipv4h.Assign (internetDevices);
  Ipv4Address remoteHostAddr = internetIpIfaces.GetAddress (1);


  // Routing of the Internet Host (towards the LTE network)
  Ipv4StaticRoutingHelper ipv4RoutingHelper;
  Ptr<Ipv4StaticRouting> remoteHostStaticRouting =
ipv4RoutingHelper.GetStaticRouting (remoteHost->GetObject<Ipv4> ());
```

```
  // interface 0 is localhost, 1 is the p2p device
  remoteHostStaticRouting->AddNetworkRouteTo (Ipv4Address ("7.0.0.0"), Ipv4Mask
("255.0.0.0"), 1);

  /*
   * Network topology:
   *
   *      |     + ----------------------------------------------------------->
   *      |     UE
   *      |
   *      |               d                   d                   d
   *    y |     |-------------------x-------------------x-------------------
   *      |     |                 eNodeB              eNodeB
   *      |   d |
   *      |     |
   *      |     |                                             d = distance
   *            o (0, 0, 0)                                   y = yForUe
   */

  NodeContainer ueNodes;
  NodeContainer enbNodes;
  enbNodes.Create (numberOfEnbs);
  ueNodes.Create (numberOfUes);
  Ptr<ListPositionAllocator> initialAllocenb = CreateObject
<ListPositionAllocator> ();
  Ptr<ListPositionAllocator> initialAllocue = CreateObject
<ListPositionAllocator> ();
  // Install Mobility Model in eNB

  for (uint16_t i = 0; i < numberOfEnbs; i++)
    {
           initialAllocenb->Add(Vector(enb_x[i],enb_y[i],10));
    }
  MobilityHelper enbMobility;
  enbMobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel");
  enbMobility.SetPositionAllocator (initialAllocenb);
  enbMobility.Install (enbNodes);

  // Install Mobility Model in UE
  for (uint32_t i = 0;i<ueNodes.GetN();i++){
 initialAllocue->Add(Vector(x_cor[i],y_cor[i],1.5));
 }
  MobilityHelper ueMobility;
```

```
  ueMobility.SetMobilityModel
("ns3::ConstantPositionMobilityModel");//ConstantVelocityMobilityModel
  ueMobility.SetPositionAllocator(initialAllocue);
  ueMobility.Install (ueNodes);


  // Install LTE Devices in eNB and UEs
  Config::SetDefault ("ns3::LteEnbPhy::TxPower", DoubleValue (enbTxPowerDbm));
  NetDeviceContainer enbDevs = lteHelper->InstallEnbDevice (enbNodes);
  NetDeviceContainer ueDevs = lteHelper->InstallUeDevice (ueNodes);


  // Install the IP stack on the UEs
  internet.Install (ueNodes);
  Ipv4InterfaceContainer ueIpIfaces;
  ueIpIfaces = epcHelper->AssignUeIpv4Address (NetDeviceContainer (ueDevs));



   for (uint16_t j=0; j<2; j++){
       lteHelper->Attach (ueDevs.Get(j), enbDevs.Get (0));
 }


  NS_LOG_LOGIC ("setting up applications");

  // Install and start applications on UEs and remote host
  uint16_t dlPort = 10000;
  uint16_t ulPort = 20000;

  // randomize a bit start times to avoid simulation artifacts
  // (e.g., buffer overflows due to packet transmissions happening
  // exactly at the same time)
  Ptr<UniformRandomVariable> startTimeSeconds =
CreateObject<UniformRandomVariable> ();
  startTimeSeconds->SetAttribute ("Min", DoubleValue (0));
  startTimeSeconds->SetAttribute ("Max", DoubleValue (0.010));

  for (uint32_t u = 0; u < numberOfUes; ++u)
    {
      Ptr<Node> ue = ueNodes.Get (u);
      // Set the default gateway for the UE
      Ptr<Ipv4StaticRouting> ueStaticRouting = ipv4RoutingHelper.GetStaticRouting
```

```
(ue->GetObject<Ipv4> ());
      ueStaticRouting->SetDefaultRoute (epcHelper->GetUeDefaultGatewayAddress (), 
1);


      for (uint32_t b = 0; b < numBearersPerUe; ++b)
        {
          ++dlPort;
          ++ulPort;

          ApplicationContainer clientApps;
          ApplicationContainer serverApps;

          NS_LOG_LOGIC ("installing UDP DL app for UE " << u);
          UdpClientHelper dlClientHelper (ueIpIfaces.GetAddress (u), dlPort);
            dlClientHelper.SetAttribute ("MaxPackets", UintegerValue 
(1250000000));//added
          clientApps.Add (dlClientHelper.Install (remoteHost));
          PacketSinkHelper dlPacketSinkHelper ("ns3::UdpSocketFactory",
                                               InetSocketAddress 
(Ipv4Address::GetAny (), dlPort));
          serverApps.Add (dlPacketSinkHelper.Install (ue));

          NS_LOG_LOGIC ("installing UDP UL app for UE " << u);
          UdpClientHelper ulClientHelper (remoteHostAddr, ulPort);
            ulClientHelper.SetAttribute ("MaxPackets", UintegerValue 
(100000000));//added
          clientApps.Add (ulClientHelper.Install (ue));
          PacketSinkHelper ulPacketSinkHelper ("ns3::UdpSocketFactory",
                                               InetSocketAddress 
(Ipv4Address::GetAny (), ulPort));
          serverApps.Add (ulPacketSinkHelper.Install (remoteHost));




          Ptr<EpcTft> tft = Create<EpcTft> ();
          EpcTft::PacketFilter dlpf;
          dlpf.localPortStart = dlPort;
          dlpf.localPortEnd = dlPort;
          tft->Add (dlpf);
          EpcTft::PacketFilter ulpf;
          ulpf.remotePortStart = ulPort;
```

```
        ulpf.remotePortEnd = ulPort;
        tft->Add (ulpf);


        EpsBearer bearer (EpsBearer::NGBR_VIDEO_TCP_DEFAULT);
        lteHelper->ActivateDedicatedEpsBearer (ueDevs.Get (u), bearer, tft);

        Time startTime = Seconds (startTimeSeconds->GetValue ());
        serverApps.Start (startTime);
        clientApps.Start (startTime);

      } // end for b
  }


  //lteHelper->EnablePhyTraces ();
  //lteHelper->EnableMacTraces ();
  lteHelper->EnableRlcTraces ();
  //lteHelper->EnablePdcpTraces ();



  // connect custom trace sinks for RRC connection establishment and handover
notification
  Config::Connect ("/NodeList/*/DeviceList/*/LteEnbRrc/ConnectionEstablished",
                   MakeCallback (&NotifyConnectionEstablishedEnb));
  Config::Connect ("/NodeList/*/DeviceList/*/LteUeRrc/ConnectionEstablished",
                     MakeCallback (&NotifyConnectionEstablishedUe));
  //Config::Connect ("/NodeList/*/DeviceList/*/LteEnbRrc/HandoverStart",
  //                 MakeCallback (&NotifyHandoverStartEnb));
  //Config::Connect ("/NodeList/*/DeviceList/*/LteUeRrc/HandoverStart",
  //                 MakeCallback (&NotifyHandoverStartUe));
  //Config::Connect ("/NodeList/*/DeviceList/*/LteEnbRrc/HandoverEndOk",
   //                MakeCallback (&NotifyHandoverEndOkEnb));
  //Config::Connect ("/NodeList/*/DeviceList/*/LteUeRrc/HandoverEndOk",
  //                 MakeCallback (&NotifyHandoverEndOkUe));



  Simulator::Stop (Seconds (10));//simTime
  Simulator::Run ();

  Simulator::Destroy ();
```

```
  return 0;

}
```

Appendix B: NS-3 tennodes.cc

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include <iostream>
#include <string>
#include <cstring>

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("FirstScriptExample");

int
main (int argc, char *argv[])
{
  CommandLine cmd;
  cmd.Parse (argc, argv);

  Time::SetResolution (Time::NS);
  LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
  LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);

  NodeContainer nodes;
  nodes.Create (12);


  PointToPointHelper pointToPoint;
  pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
  pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));

  NetDeviceContainer devices;
  devices = pointToPoint.Install (nodes.Get(0), nodes.Get(1));

  InternetStackHelper stack;
  stack.Install (nodes);

  Ipv4AddressHelper address;
  address.SetBase ("2.1.1.1", "255.255.255.1");

  Ipv4InterfaceContainer interfaces = address.Assign (devices);
```

```
UdpEchoServerHelper echoServer (9);


ApplicationContainer serverApps = echoServer.Install (nodes.Get (1));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (30.0));

UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (5.0)));
echoClient.SetAttribute ("PacketSize", UintegerValue (2048));

ApplicationContainer clientApps = echoClient.Install (nodes.Get (0));
clientApps.Start (Seconds (2.0));
clientApps.Stop (Seconds (30.0));



int nodeNum = 2;

char a = '5';


for (int i = 0; i < 10; i++)
{

    char ip[] = {'2','.','1','.','1','.',a};

    Ipv4AddressHelper address2;
    address2.SetBase ("2.1.1.1", "255.255.255.1", ip);

    NetDeviceContainer devices2;
    devices2 = pointToPoint.Install (nodes.Get(nodeNum), nodes.Get(1));

    Ipv4InterfaceContainer interfaces2 = address2.Assign (devices2);

    UdpEchoClientHelper echoClient2 (interfaces2.GetAddress (1), 9);
    echoClient2.SetAttribute ("MaxPackets", UintegerValue (1));
    echoClient2.SetAttribute ("Interval", TimeValue (Seconds (5.0)));
    echoClient2.SetAttribute ("PacketSize", UintegerValue (1024));

    ApplicationContainer clientApps2 = echoClient2.Install
```

```
(nodes.Get(nodeNum));
      clientApps2.Start (Seconds (2.0));
      clientApps2.Stop (Seconds (30.0));

      nodeNum++;

      a = a + 4;


  }

  Simulator::Run ();
  Simulator::Destroy ();
  return 0;
}
```

Appendix C: Matlab Code

```
fc = 3.5e9;
c = physconst('lightspeed');
waveform = phased.RectangularWaveform;
wav = waveform();
Rt = 3219;
az = 0;
el = 0; % 0 degree elevation
pos_tx = [0;0;0];
pos_rx = [Rt*cosd(el)*cosd(az);Rt*cosd(el)*sind(az);Rt*sind(el)];
vel_tx = [0;0;0];
vel_rx = [0;0;0];

loschannel = phased.LOSChannel(...
    'PropagationSpeed',c,...
    'OperatingFrequency',fc,...
    'TwoWayPropagation', true)
y = loschannel(wav,pos_tx,pos_rx,vel_tx,vel_rx);
L_total = pow2db(bandpower(wav))-pow2db(bandpower(y)) %The total loss
Lfs = 2*fspl(Rt,c/fc) %the path loss


tworaychannel = phased.TwoRayChannel('PropagationSpeed',c,...
    'OperatingFrequency',fc);

pos_base=[0;0;60];
pos_house=[3219;0;3];
vel_base=[0;0;0];
vel_house=[0;0;0];
y2ray=tworaychannel(wav,pos_base,pos_house,vel_base,vel_house);

L_2ray = pow2db(bandpower(wav))-pow2db(bandpower(y2ray)) %The signal loss
suffered in this channel
L_ref = fspl(norm(pos_house-pos_base),c/fc) %The free space path loss
```

Appendix D: Results Outputs

Results for 5 Mbps

https://docs.google.com/spreadsheets/d/e/2PACX-1vS3irTHKtdIcDLAUjrhcLJjK5yZN4RrJPN8c_fcpKKwbQgj8FhxAEhW2EOfrLJr849v8S8oQw9arMIJ/pubhtml

Results for 9 Mbps

https://docs.google.com/spreadsheets/d/e/2PACX-1vRxCAxsvQ6CaNRduGSdYaNpH260MeCMFLaZf7KJAxQDvvM88LBqZVLXFDBLaSfKVvlVSTwDasdi9FdN/pubhtml

Results for 10 Mbps

https://docs.google.com/spreadsheets/d/e/2PACX-1vSpWZf8J54nttm8q3-bWmbB7yUw25bgEgzvQtuGFFLknYsaVeJYzgZmKgg9el8TpRZ75QpBfe2SM_5h/pubhtml

Results for 20 Mbps

https://docs.google.com/spreadsheets/d/e/2PACX-1vRsNZ8gEuz4sPErI6PKZbbwo7G2fl1VcLj5AtxIYKdEJY-UleA4QIOcG1LRscU5AzY3rrymXFxfxUS_/pubhtml