# Dynamic Code Selection Method for Content Transfer in Deep Space Network

Rojina Adhikary
*University of Mississippi*

# Dynamic Code Selection Method for Content Transfer in Deep Space Network

A Dissertation
presented in partial fulfillment of requirements
for the degree of Doctor of Philosophy
in the Department of Electrical Engineering
The University of Mississippi

by

Rojina Adhikary

May 2019

## ABSTRACT

Space communications feature large round-trip time delays (for example, between 6.5 and 44 minutes for Mars to Earth and return, depending on the actual distance between the two planets) and highly variable data error rates, for example, bit error rate (BER) of $10^{-5}$ is very common and even higher BERs on the order of $10^{-1}$ is observed in the deep-space environment. We develop a new content transfer protocol based on RaptorQ codes and turbo codes together with a real-time channel prediction model to maximize file transfer from space vehicles to the Earth stations. While turbo codes are used to correct channel errors, RaptorQ codes are applied to eliminate the need for negative-acknowledgment of the loss of any specific packet. To reduce the effect of channel variation, we develop a practical signal-to-noise ratio (SNR) prediction model that is used to periodically adjust the turbo encoder in distant source spacecraft. This new protocol, termed as dynamic code selection method (DCSM), is compared with two other methods: turbo based *genie* method (upper bound of DCSM performance) assuming that the channel condition is perfectly known in advance and a *static* method in which a fixed turbo encoder is used throughout a communication pass. Simulation results demonstrate that the *genie* can increase telemetry channel throughput expressed in terms of the total number of successfully delivered files during a communication pass by about 20.3 % and DCSM achieves more than 99 % of *genie*, compared to the *static* approach being used currently. Logistic regression and neural networks are also used to develop two different SNR condition prediction models. Comparative performance analysis of these channel prediction models are conducted.

## ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my adviser Prof. Daigle for the continuous support, guidance, and encouragement during the course of my masters and PhD studies. My heartfelt thank goes to him for the invaluable opportunity to pursue my graduate studies in the United States.

I am especially grateful to Prof. Cao for his insightful suggestions and inputs over numerous discussions. I am grateful to Prof. Vish and Dr. Wang for all the valuable discussions we had during our departmental presentation sessions. I am thankful to Prof. Alidaee for consenting to be a member of my dissertation defense committee.

This work was supported in part by NASA EPSCoR program under grants NNX13AB31A and NNX14AN38A. My sincere gratitude goes to Mr. Kamal Oudrhiri of Jet Propulsion Laboratory (JPL) Communications Architectures and Research Section, Radio Science Group and Mr. Philip Tsao of JPL Communications Architectures and Research Section, Signal Processing and Networks Group for helping with the advanced water vapor radiometer data used in this work. My sincere acknowledgments to Mr. Philip Tsao for his helpful suggestions and comments.

Lastly, I would like to thank my family for their continuous support. I am thankful to my loving husband Dr. Amrit Kharel for all the support and encouragement. Without him, I would not be able to come to this beautiful point in my life.

TABLE OF CONTENTS

## LIST OF FIGURES

# LIST OF TABLES

## LIST OF ABBREVIATIONS

| | |
|---|---|
| **ACK** | acknowledgment |
| **ADU** | application data unit |
| **ARQ** | automatic repeat request |
| **ASM** | attached synchronization marker |
| **AWVR** | advanced water vapor radiometer |
| **BER** | bit error rate |
| **BPSK** | binary phase shift keying |
| **BWG** | beam wave guide |
| **CCSDS** | Consultative Committee for Space Data Systems |
| **CFDP** | CCSDS file delivery protocol |
| **CLTUs** | command link transmission units |
| **CRC** | Cyclic Redundancy Check |
| **CTS** | clear to send |
| **DCSM** | dynamic code selection method |
| **DDOR** | delta-differential one-way ranging |
| **DSCCs** | deep space communication complexes |
| **DSN** | deep space network |
| **DSS** | deep space stations |
| **EEP** | equal error protection |
| **ESA** | European Space Agency |
| **ESI** | encoded symbol identifier |

|  |  |
|---|---|
| **FEC** | forward error correction |
| **FRR** | frame rejection rate |
| **GDS** | ground data system |
| **GEO** | geosynchronous orbit |
| **GN** | ground network |
| **GSFC** | Goddard Space Flight Center |
| **HEF** | high efficiency |
| **IEEE** | Institute of Electrical and Electronics Engineers |
| **IF** | intermediate frequency |
| **JPL** | Jet Propulsion Laboratory |
| **LCP** | left circular polarization |
| **LEO** | low Earth orbit |
| **LT** | Luby transform |
| **MOCs** | Mission operation centers |
| **MOS** | mission operations system |
| **MRO** | Mars Reconnaissance Orbiter |
| **MSPA** | Multiple Spacecraft Per Antenna |
| **NAK** | negative-acknowledgment |
| **NAKs** | negative-acknowledgments |
| **NEN** | near Earth network |
| **ODY** | Mars Odyssey |
| **OQPSK** | offset quadrature phase shift keying |
| **PAC** | partial autocorrelation |

| | |
|---|---|
| **PACF** | partial autocorrelation function |
| **PDS** | Planetary Data System |
| **PSNR** | peak signal-to-noise ratio |
| **PDUs** | protocol data units |
| **QoS** | quality of service |
| **QPSK** | Quadrature phase shift keying |
| **RCP** | right circular polarization |
| **RFI** | radio frequency interference |
| **RTT** | round-trip time |
| **RTTs** | round-trip times |
| **SDA** | sub-carrier demodulator assembly |
| **SCMF** | space command message file |
| **SLE** | space link extension |
| **SN** | space network |
| **SNR** | signal-to-noise ratio |
| **SPC** | signal processing center |
| **SSA** | symbol synchronizer assembly |
| **TDRS** | tracking and data relay satellites |
| **TTC** | tracking, telemetry, and command |
| **UEP** | unequal error protection |
| **URT** | unequal recovery time |
| **VLBI** | very long baseline interferometry |

# CHAPTER 1

# INTRODUCTION

In this thesis, we present a new content delivery protocol for space communications, termed as DCSM, so that more efficient communication between distant spacecraft and deep space network (DSN) stations could be achieved. First, the objective is to develop prototypes of algorithms that can be developed into fully functional protocols that can be deployed in real deep space networks.

All deep-space missions–defined as those operating at or beyond the orbit of the Earth's Moon–require some form of telecommunications network with a ground system to transmit data to and receive data from the spacecraft. The DSN is one of the largest and most sophisticated of such networks.

NASA missions in low Earth orbit (LEO) communicate through either the near Earth network (NEN) or the space network (SN), both operated by the NASA Goddard Space Flight Center (GSFC). Formerly known as the ground network (GN), the NEN provides telemetry, command, ground-based tracking, and data and communications services to a wide range of customers. The NEN provides these services to customers with satellites in LEO, geosynchronous orbit (GEO), lunar orbit and missions using multiple frequency bands. The NEN utilizes both NASA-owned and commercial stations around the world to ensure proper service to customers. The SN has a number of tracking and data relay satellites (TDRS) in geosynchronous orbits. In addition, the European Space Agency (ESA) operates a number of ground stations that may be used to track NASA deep space missions during the hours after launch.

The deep space communication channel over which content transfer takes place is characterized by large round-trip times (RTTs), intermittent connectivity, and highly variable propagation channels. Terrestrial and space weather variation causes continuous changes in atmospheric attenuation and the noise temperature of the DSN station receiver antenna system, which in turn varies the received SNR. Atmospheric attenuation generally increases with increasing frequency, except in the vicinity of the water vapor line at 22.235 GHz and the oxygen band at 55 - 65 GHz. Weather effects at K-band (18 - 27 GHz) and Ka-band (27 - 40 GHz) are larger than at X-band (8 - 12 GHz), which in turn are larger than at S-band (2 - 4 GHz) and L-band (1 - 2 GHz) [1].

Apart from the exploration of the solar system and beyond, effective communication among the Mars orbiters and the Earth stations is a main focus of the scientific community at the moment. On a daily basis, a number of telemetry images and data files prepared by the Mars rovers need to be telemetered to the DSN stations. These files are transmitted to an orbiter during a scheduled communication pass between the rover and the orbiter using the Proximity-1 protocol [2]. These files are then safely stored in the orbiter memory and are relayed to the DSN stations over scheduled communication passes between the orbiter and a DSN station.

In the current deep space communication scenario, standard method of commanding the spacecraft in deep space applications is to perform *background sequencing*, in which the spacecraft is commanded once every four weeks to reconfigure its parameters. The concept of using *mini-sequencing* to vary telecommunication parameters such as modulation index, data rate profile parameters of distant spacecraft on weekly basis for Ka-band demonstration was introduced with the Mars Reconnaissance Orbiter (MRO). During the primary science phase of the MRO, links are designed to use at most two data rates with a minimum availability of 90% [3],

[4]. Using a pre-determined, fixed rate transmission scheme results in disruption of data continuity due to bad weather. At this time, use of real-time commands is very limited because of possible interaction with the planned sequences that must be validated before being uploaded [5]. However, as plans are being made for the Mars landing, Mars tourism and communication with other distant planets, use of a protocol that features real-time channel condition prediction and real-time commands to dynamically control encoders used at the orbiters may provide substantial improvement in communications quality.

Essential details of DSN required for understanding of this work, DSN operations, and services provided by the DSN along with the significant body of research relative to X-band and Ka-band communications, and unequal error protections for telemetry image transmission is presented in Chapter 2. In Chapter 3, we first present the rationale behind our approach for handling long round trip delays and weather degradation on RF waves involved in deep space communication. We discuss our developed protocol DCSM, in detail, and assess its performance from a theoretical perspective. We also elaborate the benefits and necessity of employing a dynamic code rate selection mechanism for improved deep space communication. In Chapter 4 and Chapter 5, we utilize the logistic regression and neural networks, respectively, to develop two separate channel bit-SNR prediction models. In Chapter 4, first, the necessary details of the logistic regression required to understand our work is presented. Details of how the logistic regression is utilized in our scenario is given, and the obtained model parameters and the effectiveness of the developed prediction model is discussed in the later part of the chapter. Similarly, in Chapter 5, first, the necessary details of the artificial neural networks required to understand our work is presented. Details of how the neural networks is utilized in our scenario is given, and the obtained model parameters and the effectiveness of the developed prediction model is discussed in the later part of the chapter. In Chapter 6, conclusion of the work is presented.

# CHAPTER 2

# DEEP SPACE NETWORK

The DSN is an international network of ground stations (antennas, transmitters, receivers, and associated systems) that operated intensively at S-band (2 - 4 GHz) in the 1960s and 1970s, moving into X-band (8 - 12 GHz) in the 1980s and 1990s, and more into Ka-band (27 - 40 GHz) in the 2000s. The DSN supports interplanetary-spacecraft missions and radio- and radar-astronomy observations for the exploration of the Solar System and beyond. The DSN consists of three deep space communication complexes (DSCCs) placed approximately 120° apart around the world: at Goldstone, in California's Mojave Desert; near Madrid, Spain; and near Canberra, Australia [6]. Each complex has a signal processing center (SPC) and a number of antennas (deep space stations (DSS)), including a 70 m antenna, a 34 m high efficiency (HEF) antenna, and multiple 34 m beam wave guide (BWG) antennas. Each antenna in the DSN is capable of sending commands to one spacecraft at a time. It also has the support infrastructure and personnel needed to operate and maintain the antennas.

The DSN is a multi-mission system, which provides space communication services, i.e., acquisition and/or transport of tracking, telemetry, and command (TTC) data over the space links, as well as observational science utilizing those links. The DSN operates on a multi-mission basis, serving many flight projects and experiments concurrently. Functional view of the DSN in the context of mission operations are shown in Figure 2.1. Mission operation centers (MOCs) are customer facilities that house the mission operations system (MOS). The MOS carries out the ground engineering functions necessary to operate a mission such as planning, sequence and command generation, navigation, and analysis of flight system performance and behavior. The flight project's or experiment investigation's MOS operates largely as a dedicated operations system. Depending on the project, MOCs may be located at JPL in Pasadena, California, at the customer's home institution, at a spacecraft vendor's site or some combination thereof.

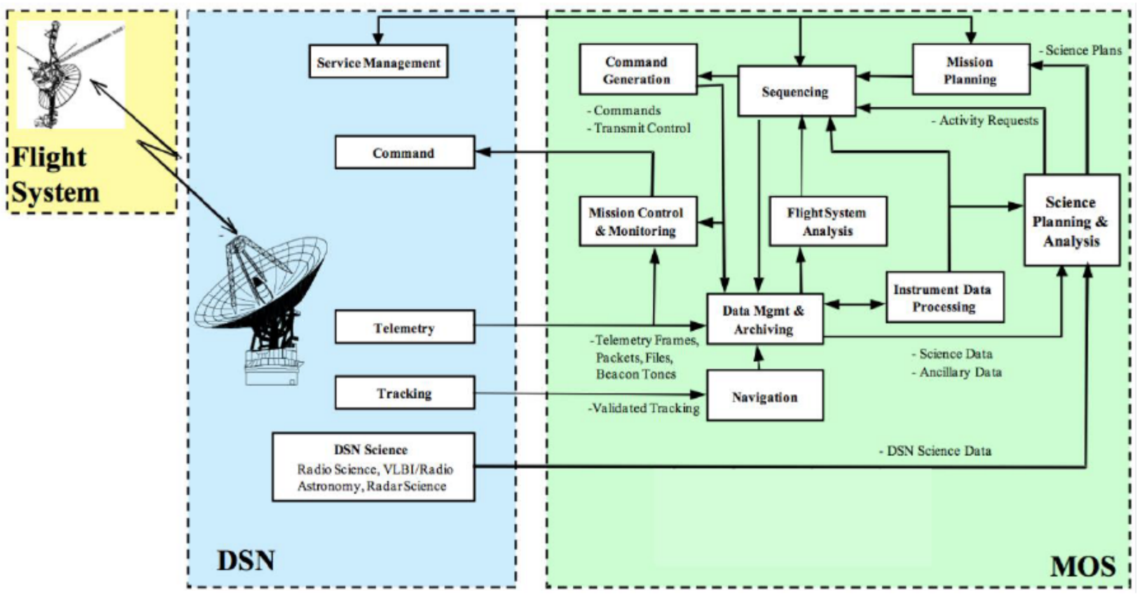


Figure 2.1: A functional view of the DSN in the context of mission operations.

## 2.1 DSN Services

In this section, we discuss about the various capabilities available from the DSN to support flight projects and experiment investigations. The capabilities described here are focused on deep space missions, near-Earth missions above GEO distance, and ground-based observational science and other mission domains [7]. The capabilities provided by the DSN are classified as data services and engineering support, details of which are covered below.

### 2.1.1 Data Services

Service is a self-contained function, which accepts one or more requests and returns one or more responses through a well-defined, standard interface. Data services provided by the DSN are mission operations services that relate directly to the transport of mission data over space-ground communications links, and to the acquisition of observational data pertaining to such links. Standard data services offered are categorized as command services, telemetry services, tracking services, calibration and modeling services, radio science services, radio astronomy or VLBI services, and radar science services. All these services are further categorized based on type of service provided.

Command services are further categorized into command radiation service and command delivery service. Telemetry services are further categorized into telemetry frame service, telemetry packet service, telemetry file service, beacon tone service, and relay service. Tracking services are categorized into validated radio metric data service and delta-DOR service. Calibration and modeling services are categorized into platform calibration service and media calibration service. Radio science services are further categorized into experiment access service and data acquisition service. Radio astronomy services are further categorized into signal capturing service, VLBI data acquisition service, and VLBI data correlation service. Similarly, radio science services are further categorized into experiment access service and data acquisition service. Brief description of command services and telemetry services are covered below.

#### 2.1.1.1 Command Services

The command services transmit information from the ground to the spacecraft. Command data transmitted typically includes commands, sequence loads, and flight software loads, and any other types of data elements. The command link is characterized by a low data rate and a relatively low volume of data of extremely high quality to assure a minimum of errors of transmission to avoid misinterpretation of commands.

This service family is functionally divided into command radiation service (more rudimentary) and command delivery service (more comprehensive).

##### *2.1.1.1.1 Command Radiation Service*

DSN operates this service in either a stream mode or a file mode. In the stream mode, data in the form of command link transmission units (CLTUs) is received from a customer's MOS and radiated, as prescribed in the space link extension (SLE) data PDU, to a spacecraft during a pass as a series of individual data units (that could be concatenated). In file mode, a command file, i.e., the space command message file (SCMF) is stored at DSN DSCCs prior to, or during, a pass and radiated to the spacecraft at a customer-specified time (radiation is reliant on an operator). Both modes ensure timely radiation of command data; however, error-free command delivery to the spacecraft is not guaranteed. The Consultative Committee for Space Data Systems (CCSDS) space link extension CLTU is the standard interface protocol in stream mode, with the DSN Command Radiation.

##### *2.1.1.1.2 Command Delivery Service*

This service includes the functionality of lower level services, i.e., the command radiation service, and the reliable command delivery feature. It accepts command files from a project's MOS in either real-time or at any point prior to the time designated for radiation. The command files are stored at the DSOC until positive confirmation of successful file delivery to the spacecraft. Using the standard CCSDS file delivery protocol (CFDP), this service executes command radiation while providing reliable error-free delivery of command data to a spacecraft. The reliable command delivery is accomplished through the selective retransmission scheme of CFDP. Therefore, missions subscribing to this service must implement that capability on the spacecraft compliant to the CFDP standard. This service requires the project to send the CFDP response directives (acknowledgment (ACK), NAK) that are received from the spacecraft via telemetry to the DSN.

#### 2.1.1.2 Telemetry Services

Telemetry services acquire telemetry data from a CCSDS compliant space link, extract communications protocol data structures, and deliver them to a customer's MOS. This data usually consists of science data, engineering data, and imaging data. Science data convey information gained from scientific experiments onboard the spacecraft. These data are moderate in volume but very valuable, with stringent quality and transmission accuracy requirements. Engineering data report the status of spacecraft instruments and systems. These data are low in volume and need to be of only moderate quality. Imaging data are of high volume. Because of the redundancy present in planetary scenes, imaging data need be of only moderate quality.

Three different levels of services are available, namely, *telemetry frame service*, *telemetry packet service*, and *telemetry file service*. Subscription to a particular level automatically includes the lower level services. In telemetry frame service, successfully received and decoded frames (both data and filler frames) are delivered to customer's MOS. In telemetry packet service, packets are extracted from frames and delivered to the customer's MOS. The telemetry file service recovers files transmitted according to the CFDP. The CFDP is a content-independent protocol, which requires no knowledge about the content or structure of a transferred file. The time needed to deliver a final, complete file via the CFDP is a function of the file size, data rate, and round-trip-light-time effect. The service extracts protocol data units (PDUs) from packets and re-assembles the PDUs into files. The resulting files are then made available to the customer's MOS, along with the meta-data contained within the transaction and the metadata summarizing the file contents and the time of receipt of the initial and final PDUs. The telemetry file service can operate in either of two basic modes:

- Unacknowledged mode
- Acknowledged mode (reliable transfer) - This mode guarantees complete file transfer within the parameters established for the protocol. The DSN will automatically notify the MOS if PDUs are missing or malformed. The DSN can also automatically respond to the spacecraft with ACK or NAK PDUs. Acknowledged mode requires use of an uplink and that the spacecraft cooperate in accord with the CFDP specification.

The DSN provides the *beacon tone service* for the flight project MOS to monitor the high-level state of the spacecraft according to the beacon tones generated and transmitted by the spacecraft. There are some missions, which have a long cruise and/or require frequent visibility during periods when the downlink signals drops below threshold for normal telemetry. In these scenarios, the beacon tone service offers a useful mechanism for the MOS to gain some minimum knowledge about the health and safety of its spacecraft. The DSN is capable of acquiring and detecting the 4-tone Beacon Monitoring signals at SNRs down to 5 dB-Hz, with detection time up to 1000 seconds. The detected tone will be forwarded to the project MOS as a message. However, the interpretation of the detected tone is the responsibility of the MOS.

The DSN also provides *relay data service* to deliver telemetry data relayed by an relaying spacecraft to a ground station. An example is a lander data being relayed by another orbiting spacecraft, or an spacecraft data being relayed by another spacecraft. The DSN would extract the relayed data from a relaying telemetry stream and deliver the data to the source mission's MOC.

Since telemetry services primarily involve data acquisition and delivery, accountability requires measures of the quantity, continuity, and latency of the data delivered. Various attributes of telemetry services metrics are discussed below.

#### *2.1.1.2.1 Quantity*

Quantity is defined as the volume of acceptable data units delivered by the service. For telemetry, data units are frames, packets, and files. Quantity is measured as the percentage of the total data return expected from the execution of the service instances, over a period of one month. The DSN routinely achieves 95% delivery during the life of mission; however, up to 98% is achievable provided there is sufficient justification and special arrangements are made, e.g., for supporting mission critical events. Use of acknowledged telemetry file service will significantly improve the percentage of original data delivered.

The quantity metrics do not take into account data loss due to insufficient data margin for the space link, adverse weather conditions, Solar conjunction, and certain spacecraft events or anomalies, (e.g., occultation, spacecraft off-pointing during downlink, tracking mode change, and sequence errors).

Users of any frequency band will experience some outages due to adverse weather conditions. The extent of outages depends strongly on the user's assumptions about weather when configuring their spacecraft, and the application of data management techniques such as CFDP. Data return strategy should be designed to be tolerant of weather caused data delays or gaps.

#### *2.1.1.2.2 Quality*

Quality is defined as the error rate for the delivered data units over the end-to-end path. A major contributing factors to telemetry quality is *frame rejection rate (FRR)*. A frame is rejected when it fails to decode or fails a checksum process. Telemetry data units typically use error-detecting and/or error-correcting codes such as Reed Solomon, convolutional, turbo, or some combination of these. Customers must determine the acceptable frame rejection rate for their circumstances, select a coding scheme accordingly, and maintain an adequate link margin in order to ensure the required performance. Another contributing factor to telemetry quality is undetected error rate introduced by ground equipment. Its value is less than $4 \times 10^{-12}$ and can therefore typically be ignored.

#### *2.1.1.2.3 Continuity*

Continuity is defined as the number of gaps in the set of data units delivered to a customer during a scheduled pass. A gap is defined as the loss of one or more consecutive data units. The DSN routinely provides a gap rate less than or equal to 8 gaps per pass provided:

- The bit energy-to-noise spectral density ratio $E_b/N_0$ is sufficient for a FRR $\leq 1 \times 10^{-5}$ at all times during the pass.
- There are no spacecraft anomalies throughout the pass.
- There are no change in the telemetry data rate or link configuration (1-way, 2-way, 3-way) during the pass that would necessitate a reacquisition.
- No radio frequency interference (RFI) events occur during the pass.

#### *2.1.1.2.4 Latency*

Latency for telemetry is defined as the delay between a data unit's (frame, packet, or file) reception at a specified point, i.e., an antenna or antennas, and its delivery to another point where it becomes accessible to a customer, i.e., the customer's MOS. However, the latency is a function of the bandwidth between the DSN and the customer's MOS, which is not under the control of the DSN. Three grades of quality of service (QoS) are defined for telemetry, each corresponding to a specific delivery mode.

- Grade 1 - Online Timely: Data delivery is initiated immediately upon ingestion, and the data delivery rate is equal to the data ingest rate throughout the duration of the service instance.
- Grade 2 - Online Complete: Data delivery is initiated immediately upon ingestion, but the data delivery rate may be less than the data ingest rate due to outbound bandwidth constraints, resulting in delayed delivery of data units.
- Grade 3 - Offline Complete: Data delivery may be initiated at any time after ingestion starts, including after ingestion is complete.

#### *2.1.1.2.5 Telemetry Data Acquisition Throughput*

Depending on the link performance, coding scheme, modulation method, and other factors, the maximum telemetry acquisition or capture rate supported by the DSN is:

- 10 Mbps (deep space) and 150 Mbps (near Earth Ka-band) for convolutional encoded data (r= 1/2, k= 7) concatenated with Reed-Solomon encoding;
- 1.6 Mbps for turbo encoded data (1 Mbps for 1/6 codes);

- The minimum supported data rate is 10 bps (uncoded), but it is recommended that the data rate be at least 40 bps for a timely acquisition.

However, since multiple missions are simultaneously tracked by each DSCC, the achievable telemetry data acquisition throughput for a given mission is constrained by the maximum aggregate data capture rate of 180 Mbps at the DSCC.

### 2.1.2 Engineering Support

DSN engineering personnel are made available to support customers in conducting pre-project studies, mission design, MOS / ground data system (GDS) development, integration and test, as well as mission operations. Support can be provided during any phase of the mission life-cycle. Various engineering support activities offered by the DSN are listed below:

- System engineering support
- Advanced mission planning support
- Emergency mission operations center support
- Emergency limited continuity of operations
- RF compatibility test support
- Mission system test support
- Spectrum and frequency management support
- Spacecraft search support
- Initial acquisition

## 2.2 DSN Stations - Operating Modes And Characteristics

To protect the equipment from power saturation damage, the DSN has a constraint on the maximum received signal strength. The DSN system can support a maximum signal level of -85 dBm (in normal mode) and in special support mode, e.g., Launch Early Orbit Phase, Earth flyby or Earth return operations, etc., the DSN system can be configured in a low-gain mode to accommodate a maximum signal level of -65 dBm.

DSN signal capture efficiency is influenced by several factors including: station-operating mode (diplexed vs. non-diplexed), aperture size, operating frequency, and various station configurations. The standard configuration for TT&C is one ground antenna with dedicated telemetry, tracking, and command equipment interacting with a single spacecraft. In this configuration, dual communication links, e.g., simultaneous X- and Ka-band links, between a spacecraft and a DSN station can also be accommodated. In addition to the standard one-station configuration, there are other operating modes, details of which are covered below.

### 2.2.1 Multiple Spacecraft Per Antenna (MSPA)

Multiple spacecraft per antenna (MSPA) is a special configuration wherein multiple receivers are connected to a single DSN antenna permitting the simultaneous reception of signals from two or more spacecraft. MSPA makes more efficient use of DSN facilities by enabling simultaneous data capture services to several spacecraft, provided that they are all within the Earth station's beam width. MSPA is not a service; it is a capability for resolving some schedule conflicts.

Presently, the DSN can receive signals from two spacecraft simultaneously in a 2-MSPA configuration. MSPA design limits uplink transmissions to a single spacecraft at a time. Thus, only one spacecraft can operate in a two-way coherent mode, all others must be in one-way non-coherent. Only the spacecraft having the uplink can be commanded. MSPA users can agree to share the uplink, switching during the pass. Approximately 30 minutes are required to reconfigure the uplink to operate with a different spacecraft resulting in 30 minutes plus RTT before coherent communication takes effect. Listed below are requirements for users of MSPA:

- All spacecraft must be within the beam width of the requested DSN station
- All spacecraft must operate on different uplink and downlink frequencies and have polarizations (up and down) consistent with the antenna's capability
- High quality (2-way) radio metric data can only be obtained from the spacecraft operating in the coherent mode.

#### 2.2.2 Antenna Arraying

Antenna arraying is a special configuration wherein the signals from two or more DSN antennas are combined to create the performance of an antenna larger than either. Arraying is also available for combining signals with different polarizations, i.e., right circular polarization (RCP), left circular polarization (LCP), etc. Combining is performed at an intermediate frequency (IF) resulting in improved performance of both the carrier and data channels. Like MSPA, antenna arraying is a capability, not a service.

#### 2.2.3 Interferometry Tracking

Interferometry tracking is an operating mode in which two stations, each at a different DSN site, are configured to perform spacecraft tracking using a very long baseline interferometry (VLBI) technique, i.e., Delta-DOR. It allows determination of the angular position (or plane-of-sky position) of a deep space spacecraft relative to a natural radio source by measuring the geometric time delay between received radio signals at the two stations.

### 2.2.4 Site Diversity

Site diversity is a special configuration in which multiple sites are scheduled to improve the certainty of achieving the desired service availability. This is normally done for critical events (e.g., orbit insertions, landings, etc.) and can be done in two ways:

- Deterministically: Sites are scheduled without reference to equipment or weather conditions,
- Adaptively: Sites are scheduled on short notice only when needed.

The ability to use adaptive scheduling techniques depends strongly on the customer's ability to adapt, the availability of resources, and/or the ability to find other customers who are willing to make arrangements to relinquish their resources on short notice.

## 2.3 DSN Telecommunications Systems

Deep space telemetry function involves the transmission of information from a spacecraft to earth, and the command function involves the transmission of information from the ground to the spacecraft. Tracking yields information on spacecraft position and velocity, the radio propagation medium, and the properties of the solar system, thus enabling trajectory monitoring and spacecraft navigation. These tracking data are characterized by a very low rate, a need for long-term stability, extreme accuracy of measurement, and by the extent of data processing required to turn the signal into information.

The performance of these three functions - telemetry, command, and tracking - depends on the amount of signal that is present relative to the noisy environment in which the signal must be detected, i.e., a ratio of signal-to-noise strength, and the degree of efficiency with which this signal-to-noise ratio is used. Advanced modulation techniques, coding schemes, modern antennas and transmitters, etc., all improve communication efficiency in their own ways. We present the details of telemetry system below.

### 2.3.1 Telemetry System

A large portion of telemetry data being captured is image data type, and these image files are of larger size as compared to other telemetry data files. To maximize the number of images acquired during the mission, virtually all image data will be compressed by the rover CPU (using either lossy or lossless compression) prior to placement into the telemetry stream [8]. Hence, different compression techniques are employed in spacecraft so that larger number of images could be stored on board. ICER is a wavelet-based image compressor designed for use with the deep-space channel [9]. ICER is a progressive image compressor that can provide lossless and lossy compression, and incorporates a sophisticated error containment scheme to limit the effects of data loss during transmission. Progressive compression means that as more compressed data are received, successively higher quality reconstructed images can be reproduced. Using ICER, one could send a small fraction of the data from a compressed image to get a low quality preview, and later send more of the data to get a higher-quality version if the image is deemed interesting.

Some unequal error protection (UEP) rateless code design methods have been proposed in recent years for image and multi-media transmissions with UEP requirements. In [10], authors proposed a modification in the structure of rateless codes to provide UEP and unequal recovery time (URT) properties by encoding message symbols by non-uniform selection of source symbols. They analyzed the performance of the proposed structure asymptotically. It was shown that UEP-rateless codes can provide very low error rates for more important bits with only a subtle loss on the performance of less important bits. Next, they focused on finite length rateless codes and derived upper and lower bounds on the maximum-likelihood decoding bit error rates of equal error protection (EEP) and UEP-rateless codes. In [11], a generalized version of UEP Luby transform (LT) codes having a larger set of parameters and hence a more flexible rateless coding scheme in introduced. Using the generalized version of UEP LT codes, a progressive transmission scheme is proposed and its performance is evaluated against two other major UEP LT codes in the literature for the progressive transmission scenario. The techniques presented by [11] can be adapted to work with the algorithm for image compression recommended by CCSDS. To design a LT-code with an unequal error protection, the bit stream produced by the image compression algorithm, i.e., ICER, recommended by CCSDS must be partitioned in $M$ disjoint sets of bits. Using the weighted approach, the LT-code produces $M$ different failure probabilities for each set of bits, $p_1, p_2, \cdots, p_M$ leading to a total probability of failure, $p$ which is an average of $p_1, p_2, \cdots, p_M$. In [12], authors analyze the problem of choosing the LT-code parameters to optimize two figure of merits: the probability of achieving a minimum acceptable peak signal-to-noise ratio (PSNR), and (b) the mean of PSNR, given that the minimum acceptable PSNR has been achieved. Given the rate-distortion curve achieved by CCSDS recommended algorithm, the work establishes a closed form of the mean of PSNR (given that the minimum acceptable PSNR has been achieved) as a function of $p_1, p_2, \cdots, p_M$.

The performance of a digital telemetry system is measured as a bit error probability, $P_b$ , or BER. The BER, in turn, depends on the bit energy-to-noise spectral density ratio $E_b/N_0$ and the inefficiencies of the demodulation/detection process. A typical deep space telemetry system is shown in Figure 2.2. The telemetry data at the spacecraft are encoded for energy efficiency, then used to phase-modulate a square-wave sub-carrier, which in turn phase-modulates a microwave sinusoidal carrier. Upon reception by the DSN, the signal is routed from antenna to receiver. The receiver tracks the carrier and translates the signal down to an intermediate frequency coherent with the station reference. The sub-carrier demodulator assembly (SDA) translates the signal down to baseband and demodulates the sub-carrier. The output of the SDA is synchronized by the symbol synchronizer assembly (SSA), which is part of the telemetry and command data handling subsystem and has two primary functions: a) detect uncoded data or, alternatively, to provide symbol integrations of coded data and b) to provide the timing or synchronization needed to perform function (a). The data are then decoded by a matching decoder.

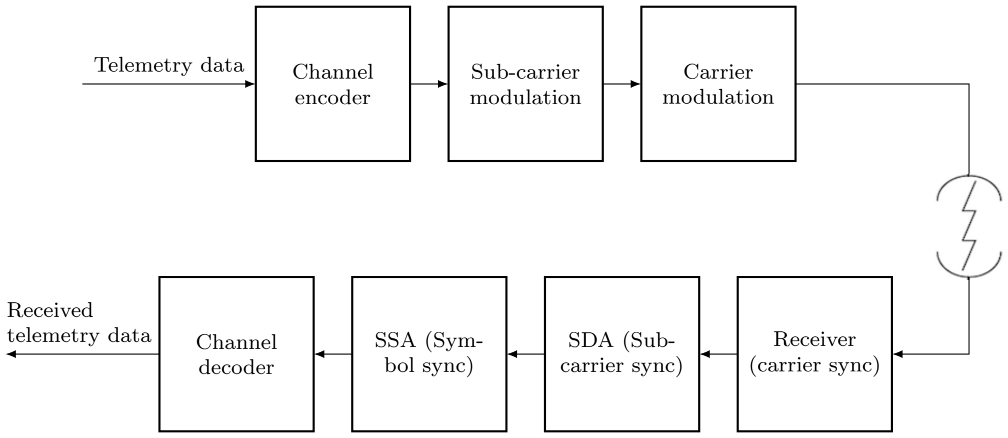


Figure 2.2: A typical deep space telemetry system.

The goal of the channel encoder and decoder is to map the input digital sequence into a channel input sequence, and, conversely, the channel output sequence into an output digital sequence, such that the effect of the channel noise is minimized. That is, the number of discrepancies (errors) between the output and input digital sequences is minimized. The approach is to introduce redundancy with the channel encoder and to use this redundancy at the decoder to reconstitute the input sequence as accurately as possible. Therefore, if an efficient code is selected and a good decoding scheme is used, we require less channel bit-SNR for a given bit error rate and a given data rate than if the data is transmitted uncoded.

#### 2.3.1.1 Received Signal Power

Received power $P_R$ by the communicating DSN antenna is computed by the following equation [13], [6]

$$P_R = P_T \, G_T \, L_T \, L_{TP} \, L_S \, L_A \, L_P \, L_{RP} \, G_R \, L_R \tag{2.1}$$

where $P_R$ is the received signal power at the input to the receiver or preamplifier, $P_T$ is the total transmitted power at transmitting antenna terminals, $L_T$ is the transmitting circuit loss between transmitting antenna terminals and radio case due to cabling, $G_T$ is the transmitting antenna gain, $L_{TP}$ is the pointing loss of the transmitting antenna, $L_S$ is the space loss, $L_A$ is the atmospheric attenuation, $L_P$ is the polarization loss between transmitting and receiving antennas due to mismatch in polarization patterns, $L_{RP}$ is the pointing loss of the receiving antenna, $G_R$ is the receiving antenna gain, and $L_R$ is the receiving circuit loss between receiving antenna and receiver due to cabling.

The space loss, or numerical ratio of received power to transmitted power between two antennas, is given by

$$L_S = \left( \frac{\lambda}{4\pi R} \right)^2 \tag{2.2}$$

where $\lambda$ is the wavelength of radio signal and $R$ is the distance between distant spacecraft and ground antennas. The transmitting antenna gain $G_T$ can be related to the effective antenna aperture $A_T$ as

$$G_T = \frac{4\pi A_T}{\lambda^2} \tag{2.3}$$

where the effective antenna aperture $A_T$ is related to the actual antenna aperture $A_t$ by the relations

$$A_T = \mu A_t \tag{2.4}$$

where $\mu$ is the antenna efficiency factor.

#### 2.3.1.2 Noise Spectral Density

Signal reception process introduces most of the noise that corrupts the receiving signal. The noise for an uplink is dominantly thermal, internal to the amplifier in the front end of the spacecraft receiver. For the downlink, thermal noise is abetted by random radiation picked up by the ground antenna. This random radiation includes that from the atmosphere, hot bodies in the field of view of the antenna, the 2.7 K cosmic background, and the portion of the ground seen by antenna side lobes. It is assumed that the receiving system noise has uniform spectral density in the frequency band containing the signal [13]. The one-sided noise spectral density $N_0$ is defined as

$$N_0 = K_b T_{op}, \tag{2.5}$$

where $K_b = 1.380622 \times 10^{-23}$ Watt/(Hz K) = –198.6 dBm/(Hz K) represents the Boltzmann's constant, and $T_{op}$ is the system operating noise temperature.

#### 2.3.1.3 Telemetry Modulation And Channel Bit-SNR

The DSN supports a wide range of telemetry modulation schemes [1]. Residual carrier BPSK using a square wave sub-carrier is the modulation scheme that has been most commonly employed for deep space telemetry. A residual carrier provides the ability to share downlink power to support additional functions such as two-way ranging and delta-differential one-way ranging (DDOR). Suppressed-carrier BPSK provides approximately the same performance at high data rates as residual-carrier BPSK and improved performance at some medium data rates. A disadvantage of suppressed-carrier BPSK is that telemetry must be disabled in order to perform DSN ranging or DDOR. Quadrature phase shift keying (QPSK) and offset quadrature phase shift keying (OQPSK) offer better bandwidth efficiency than BPSK. For a given binary symbol rate, a QPSK or OQPSK carrier occupies only half the bandwidth of a BPSK-modulated carrier (with no sub-carrier). QPSK and OQPSK have the same disadvantage as suppressed-carrier BPSK in that telemetry must be disabled in order to perform DSN ranging or DDOR. Carrier modulation indices are used to control the allocation of the amount of transmit power between carrier and data channels, and sub-carrier modulation indices are used to allocate the amount of transmit power on sub-carriers. The modulation index is established by applying a variable-amplitude voltage to the phase modulator in the exciter [1].

Sub-carriers provide a simple method for separating different types of data as well as ensuring no overlap between the modulated data's frequency spectra and RF carrier. Sub-carrier modulation suffers the disadvantages of greater spacecraft complexity, additional losses in the modulation and demodulation process, and a large occupied bandwidth. For deep space missions, also known as category B missions, either square-wave sub-carriers or no sub-carriers are used, and sine-wave sub-carriers are used for category A (near Earth) missions [14]. When a sub-carrier is present, the modulation index of the data on the sub-carrier is normally $90°$ leading to suppressed sub-carrier (no residual sub-carrier). This is done because there is no performance advantage to using a residual sub-carrier since the downlink channel are designed to track the data and ignores the presence or absence of a residual sub-carrier [1].

With no sub-carrier or square wave sub-carrier (no residual sub-carrier), when there is only a single telemetry channel present and no ranging modulation is used, the telemetry channel data directly modulates the carrier with modulation index $\theta_m$ [1], [6], [13]. The modulation index $\theta_m$ is the modulation index of the carrier, and thus allocates the total transmitted power $P_T$ to the carrier and to the data channel. The transmitted carrier power $P_{CT}$ and the transmitted data power $P_{DT}$ are given as

$$P_{CT} = P_T \cos^2 \theta_m, \tag{2.6}$$

$$P_{DT} = P_T \sin^2 \theta_m. \tag{2.7}$$

At the DSN station receiver antenna, received carrier power $P_{CR}$ and received data power $P_{DR}$ are given by

$$P_{CR} = P_R \cos^2 \theta_m, \tag{2.8}$$

$$P_{DR} = P_R \sin^2 \theta_m. \tag{2.9}$$

Note that larger the value of the carrier modulation index $\theta_m$, larger is the amount of power allocated to data channel. A system with $\theta_m < 90^\circ$ is called a residual carrier system, and a system with $\theta_m = 90^\circ$ is called a suppressed carrier system. In case of suppressed carrier, the (2.6) and (2.7) reduce to

$$P_{CT} = P_T \cos^2(90^\circ) = 0, \tag{2.10}$$

$$P_{DT} = P_T \sin^2(90^\circ) = P_T, \tag{2.11}$$

respectively. Thus leading to

$$P_{DR} = P_R. \tag{2.12}$$

The input energy per bit $E_b$ to noise spectral density $N_0$ ratio (bit-SNR) of the communication channel as measured at DSN station is [1]

$$\frac{E_b}{N_0} = \frac{P_{DR}}{N_0\, R_b}, \tag{2.13}$$

where $R_b$ is the channel transmission rate, and $N_0$ is one-sided noise spectral density referenced at the input to the receiver antenna's low-noise amplifier.

## 2.4 Conclusion

In this chapter, we discussed about DSN, services and supports offered by DSN, and DSN telecommunications systems in brief.

# CHAPTER 3

# DCSM

The proposed DCSM protocol consists of three key components: RaptorQ codes [15], [16], turbo codes [17], and a channel prediction model. Turbo codes operate at physical layer to correct the corrupted bits at the receiver end. RaptorQ codes, on the other hand, operate at application layer on application data units (ADUs) to reconstruct original source symbols from received encoded symbols. Since data recovery in RaptorQ does not depend upon receiving any specific encoded symbol but only on the number of distinct encoded symbols received, use of RaptorQ codes eliminates the need for negative-acknowledgments (NAKs) of specific lost packets.

Given that the most important factors that affect transmission quality are measurable at the DSN stations and significant computing resources can also be made available to execute fairly complex channel prediction models, to achieve high data rate, weather conditions at the DSN station are continuously tracked and the channel condition (channel bit-SNR) one RTT into the future is predicted using both the *a-priori* information and real-time channel condition. The reason behind predicting one RTT into the future is that a symbol encoded using a selection command launched from the DSN will reach back to the DSN station one RTT later; it takes a half RTT for turbo encoder selection command to reach the orbiter and another half RTT for the encoded telemetry packet (generated in response to the command) to reach the DSN station.

Based on the estimated channel bit-SNR (one RTT into the future), the turbo encoder, i.e., the turbo code rate $r$ and information block length $K$ (bits), that maximizes overall throughput of the channel is decided and the spacecraft is commanded continuously with real-time, non-interactive commands to reconfigure its turbo encoder. The telemetry data representing the RaptorQ encoded symbols will be transmitted in turbo encoded blocks specified by the CCSDS, where the amount of data per block $K$ and the coding rate $r$ for each block can be chosen on a block-by-block basis as required. At this time, the number of turbo information block sizes to be used in space communication as specified by the CCSDS is 4 and the number of coding rates is 4 yielding a total of 16 alternatives. It is also worth mentioning that the amount of data needed to command the spacecraft would be minimal, on the order of a few bits, to select from a small set of turbo encoder alternatives, and, in addition, the rate at which commands would be needed by the space vehicle is also very low.

We propose a simple and practical channel bit-SNR prediction model that uses *a-priori* information and is based on the idea of first order auto-regressive process AR(1). We demonstrate the performance of DCSM with real-time sequence and adaptive rate transmission by comparing with other two methods: a) *genie* method and b) *static* method. The DCSM, the *genie* and the *static* methods are divided into further sub-classes, details of which is covered in Section 3.3. The *genie* method represents an ideal scenario of a turbo code based adaptive rate transmission scheme along with a channel condition prediction model. In the *genie* method, we assume that the spacecraft uses (idealized) accurate channel condition prediction model and exactly knows the DSN station channel condition half-RTT in advance so that it can adjust the turbo encoder to achieve the maximum link capacity and hence provides an upper-bound in performance of a turbo based (with some automatic repeat request (ARQ) mechanisms) adaptive rate transmission scheme. In the *static* method, a turbo encoder with a fixed information block length $K$ and code rate $r$ is used throughout the duration of a communication pass. That is, a fixed, predetermined data rate is used throughout a communication pass. This method of using a fixed codeword length and code rate throughout a communication pass is a mimic of constant rate transmission scheme in which one set of parameters work for multiple days and mimics the current approach being used for X-band communication (with Mars Odyssey (ODY)). The concept of using two different rates per pass is used with the MRO mission in which 8920-bit information block length turbo codes and an algorithm called DR-90 (for dual rate per pass, and 90% availability) is used [3]. However, the data-rate profile for the allocated passes is determined at the beginning of each 28-day sequencing period. We show that our DCSM method can achieve, on average, 99.9% of the *genie*'s performance and has throughput improvement of 25% over the best possible *static* method. This high performance improvement suggests that the proposed protocol can be a good candidate for future deployment in deep space communications that involves significant RTT and noise levels.

The objective behind this work is to develop an optimal file transmission mechanism for DSN so that we can maximize the channel throughput and minimize the amount of time required to deliver each file. In order to guarantee minimum file delivery time for each file, it is necessary to give higher priority to transmission of symbols from earlier incomplete files over the current file transmission. That is, if feedback from previous files are received in the middle of current file transmission, the feedback message should be handled and resolved first. Two different issues arise when proposing an optimal file transmission using RaptorQ codes. The first is the arrangement used for file transmissions and the available options to consider are: a) serial transmission of files, b) file interleaving, and c) RaptorQ encoding all the files at once. The second is, whether it is beneficial to transmit some additional $y$ number of symbols. Considering trade offs between maximum channel throughput and the amount of time required for successful delivery of each file, serial transmission of separately RaptorQ encoded files along with proactive transmission of $y$ additional number of encoded symbols for each file appears to be the best option to consider, details of which are covered in the upcoming sections.

We expect the results of this research to contribute new insights both into the mechanism to chose the set of appropriate block sizes and coding rates, and the parameters for transmitting over specific time intervals. Considering trade-offs among coding rates, transmission time required, and encoding symbol error rates, we will develop optimal policies that achieve maximal average throughput.

The rest of the paper is organized as follows. In Section 3.1, we cover details of our trace based analysis, where Sections 3.1.1 through 3.1.4 cover numerical process of calculating channel bit-SNR value starting from wet path delay information. Section 3.2 describes the system model that includes the proposed protocol stack (Section 3.2.2), details on selecting best turbo encoder to be used and frame error rate (FER) that can be achieved by the turbo encoder for that channel bit-SNR condition (Section 3.2.3), and a data loss mitigation analysis associated with RaptorQ codes to guarantee successful reception of the required number of symbols without the need of NAK for each lost symbol (Section 3.2.4). The DCSM, the *genie* and the *static* methods are divided into further sub-classes, details of which is covered in Section 3.3. Section 3.4 presents our channel bit-SNR prediction model based on both *a-priori* and real-time information. Section 3.5 presents communication data rates corresponding to the selection of turbo encoder and compares, via simulation, the proposed DCSM method with its upper bound given by the *genie* method and the *static* method for MRO Ka-band communication scenario. Section 3.6 concludes the chapter.

Table 3.1: Table of symbols used in Section 3.1.

| Symbol | Meaning |
|---|---|
| $\theta$ | antenna elevation angle |
| $\theta_m$ | modulation index of carrier signal |
| $D_{\text{wet}}^{(f)}$ | zenith wet path delay measured by an antenna operating at frequency $f$ |
| $T_{\text{sky}}^{(f)}$ | zenith sky brightness temperature measured by an antenna operating at frequency $f$ |
| $T_{\text{sky}}^{(f)}(\theta)$ | sky brightness temperature measured by an antenna operating at frequency $f$ and at an elevation angle of $\theta$ |
| $L_{\text{atm}}^{(f)}$ | zenith atmospheric attenuation experienced by an antenna operating at frequency $f$ |
| $L_{\text{atm}}^{(f)}(\theta)$ | atmospheric attenuation experienced by an antenna operating at frequency $f$ and at an elevation angle of $\theta$ |
| $T_{\text{atm}}^{(f)}$ | zenith atmospheric noise temperature as seen by an antenna operating at frequency $f$ |
| $T_{\text{atm}}^{(f)}(\theta)$ | atmospheric noise temperature as seen by an antenna operating at frequency $f$ and at an elevation angle of $\theta$ |
| $T_{\text{op}}^{(f)}(\theta)$ | system operating noise temperature measured by an antenna operating at frequency $f$ and at an elevation angle of $\theta$ |

## 3.1 Trace Based Analysis

Sky brightness temperatures have been measured for more that 20 years at Madrid Deep Space Communication Complex (DSCC), 17 years at Goldstone DSCC and 9 years at Canberra DSCC [1]. For the trace based analysis and simulation, we are using recorded advanced water vapor radiometer (AWVR) measurement data of zenith wet-path delay $D_{\text{wet}}$ measurements. AWVR provides a precise measure of the line-of-sight, water-vapor-induced signal delay in the Doppler tracking of the Cassini spacecraft at the NASA Goldstone station. Zenith wet path delay $D_{\text{wet}}$ measurements taken every 30 secs during a communication pass with Cassini spacecraft at Goldstone DSCC at 31.4 GHz are archived in Planetary Data System (PDS) archive system in path delay data files. Along with zenith wet path delay measurements, the file contains time of measurement, antenna elevation, antenna azimuth, and zenith dry path delay measurements taken every 30 secs interval over the duration of a communication pass. In order to use those data for conducting trace based analysis of Earth-spacecraft communication and to design a future channel bit-SNR prediction method, first we need every 1 s interval data. Since we have enough number of samples of each communication pass to reconstruct the data, we use cubic basis-spline interpolation [18], [19] to fit those set of data points and generate a continuous signal representation. For this purpose, python functions *splrep* and *splev* of *scipy* module *interpolate* is used [20]. The function splrep is first used to compute a spline representation of the curve, and then the function splev is used to evaluate the spline at the desired points (every one second interval, for our problem). In Appendix B, the motivation behind using cubic spline interpolation is presented.

Using the traces of zenith wet path delay $D_{\text{wet}}$ and antenna elevation angle $\theta$ profile of a communication pass, zenith sky brightness temperature $T_{\text{sky}}$, atmospheric attenuation $L_{\text{atm}}(\theta)$ experienced by an antenna and system operating noise temperatures $T_{op}(\theta)$ at an elevation angle of $\theta$ of the pass can be obtained, details of which are presented in the following sub-sections. Receiver antenna received power levels, system operating noise temperatures, atmospheric attenuation and other factors are translated into frame error probabilities for all alternative turbo block sizes and coding rates.

### 3.1.1 Zenith Wet Path Delay To Zenith Sky Brightness Temperature

In this section we show the details of extracting zenith sky brightness temperature $T_{\text{sky}}^{(f)}$ for an antenna operating at a certain frequency $f$ using the AWVR zenith wet path delay $D_{\text{wet}}$ and antenna elevation angle $\theta$ data. Additional delay incurred to a signal by water content of the atmosphere is called wet path delay.

The zenith sky brightness temperature $T_{\text{sky}}^{(f)}$ as seen from the ground is defined as the noise contribution of the entire atmosphere plus the attenuated noise contribution of the cosmic microwave background along the direction of zenith for an antenna operating at a frequency $f$ [21]. In [22], authors present opacity vs wet path delay curves based on real measurement data over a period of one year at Goldstone site. Using measurements of J03 WVR-derived opacities versus GPS-derived wet path delay at the Goldstone site for 31.4 GHz channel, the zenith wet path delay $D_{\text{wet}}$ is converted into water vapor opacity $\tau^{(31.4)}$ [22]. The zenith sky brightness temperature $T_{\text{sky}}^{(31.4)}$ for 31.4 GHz channel is computed from the water vapor opacity $\tau^{(31.4)}$ by using the relation given in [23], which is

$$\tau^{(31.4)} = -\ln\left[\frac{275 - T_{\text{sky}}^{(31.4)}}{272}\right]. \tag{3.1}$$

Using temporal zenith wet path delay $D_{\text{wet}}$ data and antenna elevation $\theta$ profile of a communication pass for DSS-25 at Goldstone DSCC, we computed temporal zenith sky brightness temperature $T_{\text{sky}}^{(31.4)}$ for a 31.4 GHz channel and its plot is presented in the Figure 3.1.

### 3.1.2 Zenith Brightness Temperature To Zenith Attenuation

Presence of gaseous components and water (rain, clouds, snow and ice) in the troposphere and the ionosphere causes attenuation of waves as they propagate through the atmosphere. This attenuation experienced by a receiver antenna operating at a frequency $f$ along the direction of zenith is called zenith atmospheric attenuation and is denoted by $L_{\text{atm}}^{(f)}$. In this section, we present the details of converting the obtained zenith sky brightness temperature $T_{\text{sky}}^{(31.4)}$ to zenith atmospheric attenuation $L_{\text{atm}}^{(32)}$ experienced by an antenna operating at Ka-band (32 GHz) channel. First, the zenith atmospheric noise temperature $T_{\text{atm}}^{(31.4)}$ as seen by an antenna

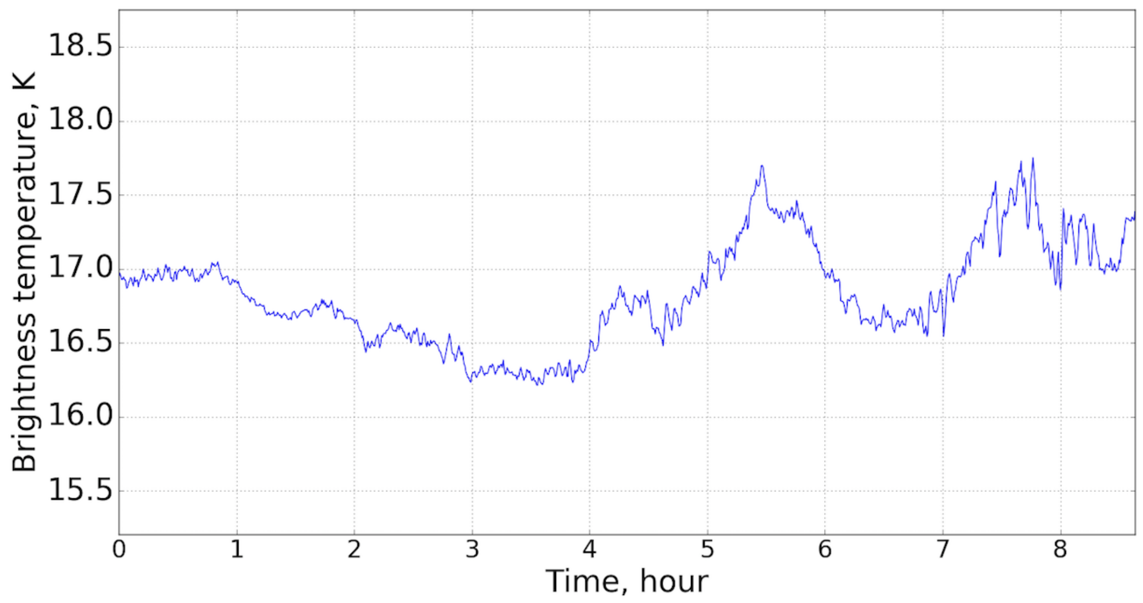


Figure 3.1: Zenith sky brightness temperature $T_{\text{sky}}^{(31.4)}$ (K) vs time (hour) plot of DSS-25 measured at 31.4 GHz during a communication pass on September 26, 2005.

operating at 31.4 GHz is computed from zenith sky brightness temperature $T_{\text{sky}}^{(31.4)}$ measurements by using the following formula [24]

$$T_{\text{atm}}^{(31.4)} = T_p \left( \frac{T_{\text{sky}}^{(31.4)} - T_{\text{cosmic}}}{T_p - T_{\text{cosmic}}} \right), \tag{3.2}$$

where $T_p$ is the physical temperature of the atmosphere and 275 K is a reasonable approximation of $T_p$, and $T_{\text{cosmic}}$ is cosmic microwave background temperature and its value is found to be a constant $T_{\text{cosmic}} = 2.725$ K.

Then we convert the zenith atmospheric noise temperature $T_{\text{atm}}^{(31.4)}$ at 31.4 GHz to atmospheric noise temperature $T_{\text{atm}}^{(32)}$ at 32 GHz using the following formula

$$T_{\text{atm}}^{(32)} = T_{\text{atm}}^{(31.4)} + 5 \left( 1 - \exp^{\left( -0.008\, T_{\text{atm}}^{(31.4)} \right)} \right). \tag{3.3}$$

Next we use the obtained $T_{\text{atm}}^{(32)}$ to calculate zenith atmospheric attenuation $L_{\text{atm}}^{(32)}$ at

32 GHz as follows

$$L_{\text{atm}}^{(32)} = \frac{T_p}{T_p - T_{\text{atm}}^{(32)}} = \frac{275}{275 - T_{\text{atm}}^{(32)}}, \tag{3.4}$$

which is a unit less quantity. The zenith atmospheric attenuation $L_{\text{atm}}^{(32)}$ experienced by the antenna can be converted to atmospheric attenuation $L_{\text{atm}}^{(32)}(\theta)$ at an elevation angle of $\theta$. This relation between $L_{\text{atm}}^{(f)}$ and $L_{\text{atm}}^{(f)}(\theta)$ holds true for any frequency $f$ of operation and is given by

$$L_{\text{atm}}^{(f)}(\theta) = \frac{L_{\text{atm}}^{(f)}}{\sin\theta}, \quad \text{dB}. \tag{3.5}$$

Using the computed temporal zenith sky brightness temperature $T_{\text{sky}}^{(31.4)}$ data for DSS-25 at Goldstone DSCC, we obtained the corresponding temporal zenith attenuation $L_{\text{atm}}^{(32)}$ for 32 GHz Ka-band and its plot is presented in Figure 3.2.

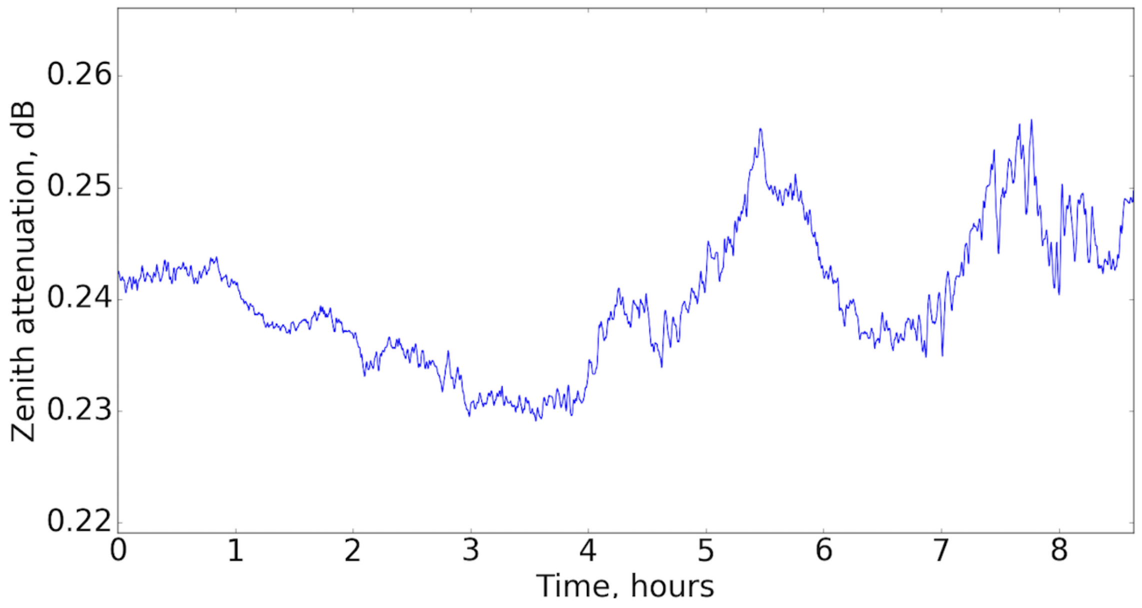


Figure 3.2: Zenith atmospheric noise temperature $L_{\text{atm}}^{(32)}$ (dB) vs time (hours) plot of DSS-25 at Ka-band (32 GHz) during a communication pass on September 26, 2005.

### 3.1.3 Zenith Attenuation To System Operating Noise Temperature

The receiver system operating noise temperature $T_{op}^{(f)}(\theta)$ varies as a function of antenna elevation angle $\theta$ due to changes in the path length through the atmosphere and ground noise received by the side-lobe pattern of the antenna at a given frequency $f$ of operation. The system operating noise temperature $T_{op}^{(f)}(\theta)$ consists of two parts, an antenna-microwave component $T_{\text{AMW}}^{(f)}(\theta)$ and a sky component $T_{\text{sky}}^{(f)}(\theta)$ as shown is (3.6). Here, all the operations described in this section holds true to any antenna operating at a frequency $f$. Hence, we get rid of the superscript $(f)$ part for the ease of representation here onwards. The antenna-microwave component $T_{\text{AMW}}(\theta)$ represents the contribution of the antenna and microwave hardware only, and the sky component $T_{\text{sky}}(\theta)$ represents the contribution of the atmospheric noise plus the cosmic microwave background noise. The receive system operating noise temperature is given by

$$\begin{aligned} T_{op}(\theta) &= T_{\text{AMW}}(\theta) + T_{\text{sky}}(\theta) \\ &= \left[T_1 + T_2 e^{-a\theta}\right] + \left[T_{\text{atm}}(\theta) + T'_{\text{cosmic}}(\theta)\right], \end{aligned} \tag{3.6}$$

where $T_1$, $T_2$, and $a$ are antenna-microwave noise temperature parameters [1]. The atmospheric noise temperature $T_{\text{atm}}(\theta)$ at an antenna elevation angle $\theta$ is computed using

$$T_{\text{atm}}(\theta) = T_M \left[1 - \frac{1}{L(\theta)}\right], \quad \text{K}, \tag{3.7}$$

here $T_M = 255 + 25 \times \text{CD}$ is the atmosphere mean effective radiating temperature (K), $0 \leq \text{CD} \leq 0.99$ is cumulative distribution, and $L(\theta)$ is atmosphere loss factor. The $L(\theta) > 1.0$ is a dimensionless quantity and is given as

$$L(\theta) = 10^{(L_{\text{atm}}(\theta)/10)}. \tag{3.8}$$

Here $L_{\text{atm}}(\theta)$ (in decibel) is the atmospheric attenuation experienced by the antenna at an elevation angle of $\theta$ and can be obtained from the zenith atmospheric attenuation $L_{\text{atm}}$ at the given frequency of operation using the relation given in (3.5).

The effective cosmic background noise $T'_{\text{cosmic}}$ is obtained using the cosmic microwave background noise temperature $T_{\text{cosmic}}$ and atmosphere loss factor $L(\theta)$ as

$$T'_{\text{cosmic}} = \frac{T_{\text{cosmic}}}{L(\theta)}, \quad \text{K}. \tag{3.9}$$

The obtained temporal receiver $T_{op}$ and $T_{sky}$ (K) plot for the DSS-25 is presented in Figure 3.3.

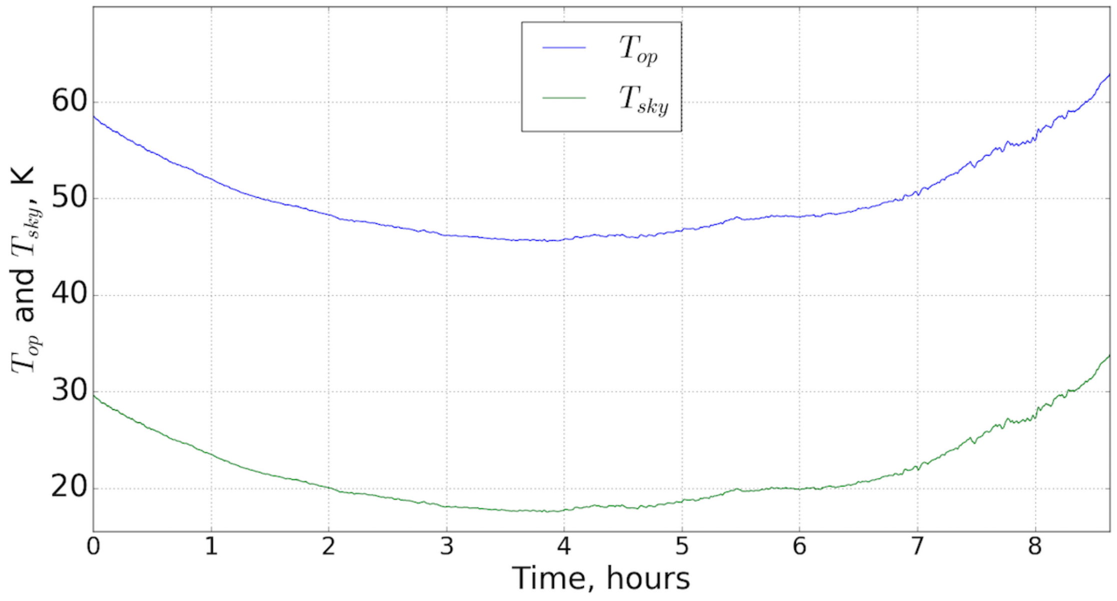


Figure 3.3: System operating noise temperature $T_{op}$ (K) and sky brightness temperature $T_{sky}$ (K) vs time (hours) plot of DSS-25 at 32 GHz during a communication pass on September 26, 2005.

### 3.1.4 Channel Bit-SNR

In this section, we cover the details of computing channel bit-SNR $E_b/N_0$ of the communication channel as measured at DSN station. Let us consider, the distant orbiter consists a transmit antenna with gain $G_T$ that radiates a power $P_T$ in the direction of the DSN station's receiver antenna. On its way, the radiated power suffers from free space path loss and atmospheric attenuation. Apart from free space loss and atmospheric attenuation, in practice, additional losses occur due to losses in the transmitting and receiving equipment, de-pointing losses, polarization mismatch losses. As we are evaluating baseline performance of our proposed methods, we assume that losses in transmitting and receiving equipment, de-pointing losses and polarization mismatch losses are negligible.

A receiving antenna operating at a frequency of $f$ and at an elevation angle $\theta$ with gain $G_R$ situated at a distance $R$ from the transmitting antenna receives a power $P_R$ given by the following expression

$$\begin{aligned} P_R &= P_T G_T \left( \frac{1}{L_{\mathrm{FS}}\, L_{\mathrm{atm}}(\theta)} \right) G_R \\ &= P_T G_T \left( \frac{\lambda}{4\pi R} \right)^2 \left( \frac{1}{L_{\mathrm{atm}}(\theta)} \right) G_R, \quad \text{(Watt)}. \end{aligned} \tag{3.10}$$

where $L_{\mathrm{FS}} = (4\pi R/\lambda)^2$ is called the free space loss, and $L_{\mathrm{atm}}(\theta)$ is atmospheric attenuation experienced by the antenna at an elevation angle of $\theta$. The $P_R$ is the total received signal power at the input to the low-noise amplifier which can be further divided into the received carrier power $P_C$ and received data power $P_D$ as

$$P_R = P_C + P_D.$$

Modulation types used on MRO for Ka-band telemetry transmissions are binary phase shift keying (BPSK) on a square wave sub-carrier with the sub-carrier modulating the carrier, and BPSK directly on the carrier (no sub-carrier) [5]. Details of modulation and modulation indices are presented in Section 2.3.1.3. As we know that, the carrier power $P_C$ and data power $P_D$ received by the receiver antenna is calculated from $P_R$ using the modulation indexes of the link and depends on the type of modulation used. In MRO, Ka-band communication is used for telemetry purpose only, and there is a single telemetry channel present. With no sub-carrier or square wave sub-carrier, when there is only a single telemetry channel present and no ranging modulation is used, the carrier power $P_C$ and data power $P_D$ received by the receiver antenna at the DSN station are given by [1]

$$P_C = P_R \cos^2 \theta_m,$$

$$P_D = P_R \sin^2 \theta_m,$$

here $\theta_m$ is carrier modulation index used at the MRO.

The input energy per bit $E_b$ to noise spectral density $N_0$ ratio (bit-SNR) of the communication channel as measured at DSN station is [1]

$$\frac{E_b}{N_0} = \frac{P_D}{N_0\, R_b} = \frac{P_R \sin^2 \theta_m}{N_0\, R_b}, \tag{3.11}$$

where $R_b$ is the channel transmission rate, and $N_0$ is one-sided noise spectral density referenced at the input to the receiver antenna's low-noise amplifier given as

$$N_0 = K_b\, T_{op}.$$

Here $K_b = 1.380622 \times 10^{-23}$ Watt/(Hz K) represents the Boltzmann's constant. Using (3.11), we generated temporal channel bit-SNR plot for the DSS-25 and is presented in Figure 3.4.

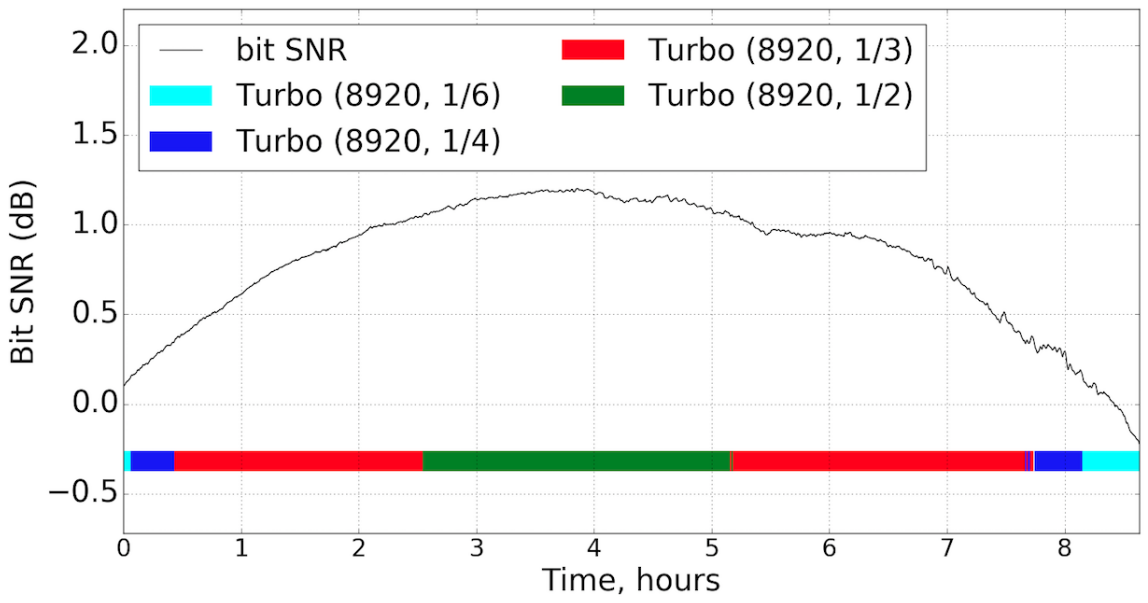
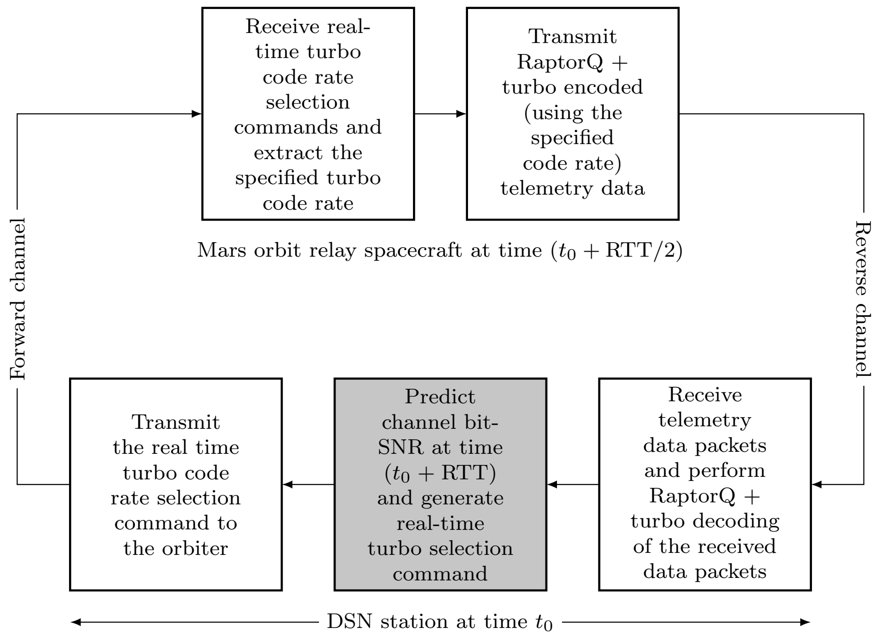


Figure 3.4: Channel bit-SNR (dB) vs time (hours) plot of DSS-25 during a communication pass on September 26, 2005.



Receive real-time turbo code rate selection commands and extract the specified turbo code rate
Transmit RaptorQ + turbo encoded (using the specified code rate) telemetry data
Mars orbit relay spacecraft at time $(t_0 + \mathrm{RTT}/2)$
Forward channel
Reverse channel
Transmit the real time turbo code rate selection command to the orbiter
Predict channel bit-SNR at time $(t_0 + \mathrm{RTT})$ and generate real-time turbo selection command
Receive telemetry data packets and perform RaptorQ + turbo decoding of the received data packets
DSN station at time $t_0$

Figure 3.5: Summary of encoding and decoding processes occurring at a DSN station and the distant spacecraft.

## 3.2 System Model

In the proposed DCSM method, two separate processes are simultaneously occurring at a communicating DSN station; a) channel condition prediction, appropriate turbo code rate selection and command generation, and real-time commanding of the distant spacecraft, and b) telemetry data reception and decoding. Similarly, two separate processes are simultaneously occurring at the distant spacecraft; a) real time command reception, decoding, and appropriate turbo encoder selection and b) RaptorQ plus turbo encoding and telemetry data transmission. Summary of the processes are presented in Figure 3.5. The spacecraft-to-DSN direction of communication channel is the reverse channel or the downlink path, and the DSN-to-spacecraft direction of communication channel that is used for commanding the spacecraft is the forward channel or uplink channel.

During communication with a distant spacecraft, channel condition prediction algorithm is continuously executed at DSN station to predict channel bit-SNR one RTT into the future. The turbo code that maximizes throughput for the predicted channel condition is decided, and real time turbo encoder selection commands are generated and transmitted to the distant spacecraft. The command is received by the distant spacecraft half RTT later and the specified turbo encoder is selected for telemetry encoding. The RaptorQ and turbo encoded packets are telemetered to the communicating DSN station. The DSN station receives the encoded packets exactly one RTT from the instant channel condition was predicted.

### 3.2.1 File Arrangement

As mentioned earlier, we are proposing serial transmission of separately RaptorQ encoded files along with proactive transmission of $y$ additional encoded symbols to mitigate possible packet losses for each file. In serial transmission of files, once the necessary symbols from file $i$ is transmitted, transmission of symbols from file $i + 1$ begins. At the receiver end, as soon as the number of corrupted symbols received for the $i$th file exceeds $y$, feedback message(s) requesting transmissions of additional encoded symbols of the $i$th file is sent over to the spacecraft. As soon as a feedback is received at the spacecraft, the current transmission is put to a temporary halt and the requested number of symbols from $i$th file are transmitted. Using this approach, we can guarantee file delivery in minimum amount of time for each file with reasonable processing and memory usage.

In file interleaving, $N$ transmission queues hold encoded symbols from $N$ interleaved files at a time. A proportion $p$ $(0 < p < 1)$ of symbols from each interleaved files are transmitted at a time until all the necessary symbols from all the $N$ files are transmitted. Once the first $N$ files are done, another set of $N$ files are interleaved for transmission and the process goes on. The purpose behind interleaving is to distribute errors among multiple files in case of burst errors, so that one file does not have to bear all the losses. However, interleaving requires relatively larger number of standby queues, and has larger memory consumption and complexity. This method may also lead to large number of incomplete files in the system at a time, i.e., certain portion of all the received files are missing, and may result in huge increase in file delivery time required for a file.

In third approach, instead of RaptorQ encoding each file individually, all the files to be transmitted are RaptorQ encoded at once, and then transmitted. As we can see that, this method will result in highest throughput in comparison to the other two options and has lowest complexity. However, real time delivery of a file is not possible and time to deliver each file might be very large. If files have different transmission priorities, its not possible to transmit and deliver files in priority orders.

From the general comparison presented here, the serial transmission of files appears to be the best option to consider in DSN in order to achieve maximum throughput and minimum file delivery time.

### 3.2.2 Protocol Stack: RaptorQ Over Turbo

In a typical space communication channel, the principal signal degradation are due to the loss of signal energy with distance, and also due to the thermal noise in the receiving system. Based on experiments, data measurements and historical data, some convolutional and block codes have been recommended by CCSDS standards to be used in space channels as they are capable of providing good communication over the channels. In Mars Odyssey, the telemetry signal is encoded with two codes, a Reed-Solomon $(255, 223)$ as outer code and a $(7, 1/2)$ convolutional code as inner code [25]. In MRO, three different coding types are available [4], [5]; a) $(255, 223)$ Reed-Solomon block code with interleaving depth of either 5 or 1, b) $(255, 223)$ Reed-Solomon and a $(7, 1/2)$ convolutional code with interleaving depth of 5, and c) turbo codes with block length 8920 bits and rates $1/2$, $1/3$, and $1/6$.

We propose to use RaptorQ class of fountain codes at the application layer and a dynamic turbo coding system at data link layer for the orbiter to communicate with DSN stations. The RaptorQ codes are the most advanced and efficient fountain codes designed to date [16], [15]. RaptorQ based content transfer protocols can generate very large numbers of encoded symbols but the receiver needs only a specific number of unique encoded symbols in order to recover the original content. As long as the receiver is able to collect a number of encoded symbols slightly more than the number of original source symbols, it can successfully recover the original file. The success of recovery only depends on the number of encoded symbols received, but not on any specific symbols. In essence, each encoded symbol is equivalent to a linear equation in which the source symbols are the variables and the encoded symbol is the value of the equation. The source symbols themselves are recovered through the equivalent of solving a linear system of equations, and the only requirement is to collect enough linear equations so that the system is linearly independent. As a result, RaptorQ-based protocol removes the need to NAK for any specific symbol and simplifies design of delay tolerant content transfer protocols.

Turbo codes are one of the most advanced forward error correction (FEC) codes that can achieve near-Shannon-limit error correction performance with reasonable coding and decoding complexity [17], [26]. Good turbo codes can come within approximately 0.8 dB of the theoretical limit at a BER of $10^{-6}$. According to the CCSDS recommended standard for telemetry synchronization and channel coding [27], turbo codes shall have nominal code rates selected from

$$r \in \{1/2, 1/3, 1/4, 1/6\}$$

and information block length selected from

$$K \in \{1784, 3568, 7136, 8920\}\,(\text{bits}).$$

With turbo codes, synchronization is accomplished by preceding each transfer frame with a rate dependent attached synchronization marker (ASM). For the recommended codes, the turbo decoder error floor, defined as a point where a further increase in the bit-SNR $E_b/N_0$ does not significantly increase a turbo decoder's performance, occurs at a BER of less than $10^{-7}$. For operation near this region, CCSDS recommends (optional) that the data content of each information block be reduced to allow for a 16 bit Cyclic Redundancy Check (CRC) as an independent check on the decoding process to be inserted at the end of the codeblock [1], [27], [28]. Serial concatenation of CRC encoding and turbo encoding is shown in Fig 3.6. Rate $r$ turbo encoded frame containing a information block of length $K$ bits is shown in Figure 3.7.

The RaptorQ encoder treats each file to be telemetered to DSN as an indi-

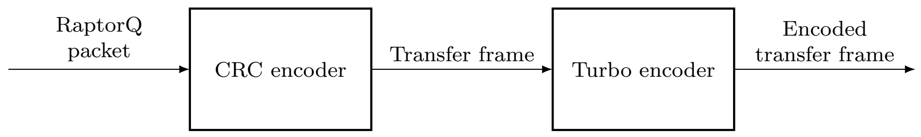


Figure 3.6: Turbo-CRC encoder.

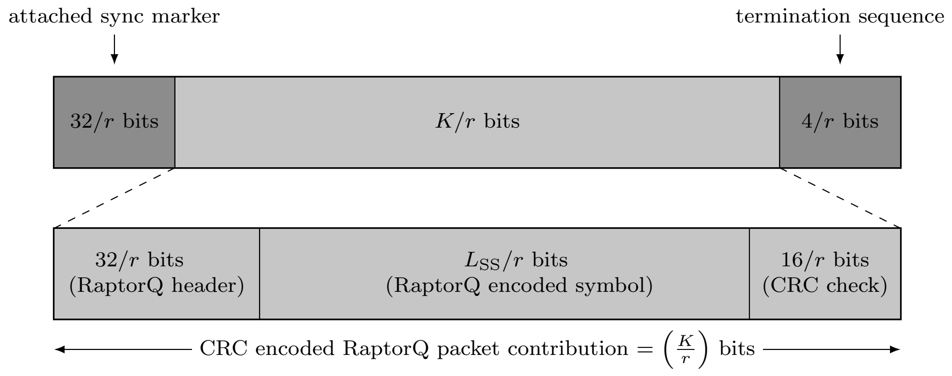

attached sync marker
termination sequence
$32/r$ bits
$K/r$ bits
$4/r$ bits
$32/r$ bits (RaptorQ header)
$L_{\mathrm{SS}}/r$ bits (RaptorQ encoded symbol)
$16/r$ bits (CRC check)
CRC encoded RaptorQ packet contribution = $\left(\frac{K}{r}\right)$ bits

Figure 3.7: Turbo encoded frame with rate dependent attached synchronization marker, a RaptorQ packet contribution, and termination sequence.

vidual source block and divides a file into $K_{\mathrm{SS}}$ source symbols of size $L_{\mathrm{SS}}$ bits. These source symbols are then RaptorQ encoded to generate $N_{\mathrm{ES}}$ encoded symbols of size $L_{\mathrm{SS}}$ bits (same size as that of a source symbol), where $K_{\mathrm{SS}} < N_{\mathrm{ES}}$. Each encoded symbol has an associated encoded symbol identifier (ESI) of size 32 bits to uniquely identify the symbol. A RaptorQ encoded symbol along with its ESI (packet header) forms a RaptorQ packet of size

$$L_{\mathrm{RQ}} = (L_{\mathrm{SS}} + 32) \quad \text{bits.}$$

Each RaptorQ packet is an application data unit (ADU) that is handed over to the CRC encoder to create a CRC encoded transfer frame and the transfer frame is handed over to turbo encoder for FEC coding before transmission. Selection of the turbo encoder for a particular packet is based on the channel prediction model and the choices are transmitted to the orbiter from the DSN as real-time commands. The turbo encoded frames are then transmitted over the telemetry channel from the orbiter to a DSN station. The length of the CRC encoded RaptorQ packet (transfer frame) $L_{\text{CRC}}$ and the length of a turbo encoded frame (in bits) is given by (3.12) and (3.13)

$$L_{\text{CRC}} = K = L_{\text{RQ}} + 16, \quad \text{and} \tag{3.12}$$

$$L_{\text{turbo}} = \left(\frac{36 + K}{r}\right) = \left(\frac{36 + L_{\text{RQ}} + 16}{r}\right). \tag{3.13}$$

At the DSN station, the receiver system first reconstructs a transmitted frame and hands it over to the turbo decoder. If the turbo decoder successfully decodes the frame, a clean CRC encoded RaptorQ packet is generated and passed over to the CRC decoder. The CRC decoder checks if there are any errors in the received RaptorQ packet, and hands over to the RaptorQ decoder for further processing if no error is detected, else the packet is discarded. Once the number of encoded symbols received by the RaptorQ decoder is sufficient, the original file is reconstructed. RaptorQ decoder requires $(K_{\text{SS}} + \Theta)$ encoded symbols to be able to successfully decode a source block with $K_{\text{SS}}$ source symbols, where $\Theta \in \{0, 1, 2, \cdots\}$ is the number of additional symbols (symbol overhead) required. Higher the symbol overheads, higher is the probability of successful decoding as shown in Table 3.2.

Table 3.2: Probability of successful decoding of a RaptorQ encoded source block for different overhead values.

| Overhead ($\Theta$) | Received encoding symbols ($K_{\text{SS}} + \Theta$) | Probability |
|---|---|---|
| 0 | $K_{\text{SS}}$ | 99% |
| 1 | $K_{\text{SS}} + 1$ | 99.99% |
| 2 | $K_{\text{SS}} + 2$ | 99.9999% |

### 3.2.3 Turbo Encoder Selection

In Section 3.1, necessary details of computing channel bit-SNR information starting with the wet-path delay information has been presented. The obtained value of channel bit-SNR is translated into frame error probabilities for all alternative turbo block sizes and code rates. In our DCSM scheme, turbo code selections are those that yield the highest instantaneous overall image reception rate for the channel bit-SNR.

Using the proposed channel condition prediction model (Section 3.4), channel bit-SNR one RTT into the future is predicted. Given the predicted value of channel bit-SNR over the 1 s period, the turbo encoder that maximizes telemetry channel throughput for that period is chosen. The rate of packet transmission from the orbiter is $R_b r_j/(K_i+36)$ packets/s. The rate at which correct packets are received at the DSN is $R_b r_j\ (1-P_E(\text{SNR},K_i,r_j))/(K_i+36)$ packets/s. Overall throughput is thus $R_b r_j\ (1-P_E(\text{SNR},K_i,r_j))$ bits/s. Mathematically, the selection of turbo encoder is obtained by maximizing the overall throughput, i.e.,

$$\max_{i,j}\ \ (1-P_E(\text{SNR},K_i,r_j))\ r_j, \tag{3.14}$$

with

$$K_i \in \{1784, 3568, 7136, 8920\},$$

$$r_j \in \{1/2, 1/3, 1/4, 1/6\},$$

where $K_i$ is the information block length, $r_j$ is the code rate, and $P_E(\text{SNR},K_i,r_j)$ is the frame error rate corresponding to the use of turbo encoder with $K_i$ and $r_j$ at the bit-SNR.

Using (3.14) against the performance of turbo codes presented in [1] and [27], we can obtain the dynamic code assignments shown in Table 3.3 for achieving highest

Table 3.3: Dynamic turbo code assignment for different channel bit-SNR ranges and their names.

| Name | Channel bit-SNR (dB) range | turbo code $(K, r_i)$ |
|---|---|---|
| Range A | $< -0.5$ | - |
| Range B | $[-0.5, -0.1)$ | $(8920, 1/6)$ |
| Range C | $[-0.1, 0.15)$ | $(8920, 1/4)$ |
| Range D | $[0.15, 0.85)$ | $(8920, 1/3)$ |
| Range E | $[0.85, 2.2)$ | $(8920, 1/2)$ |

link throughput. Out of the sixteen turbo code options recommended by CCSDS, under different channel conditions, link throughput is maximized when $K = 8920$ bits. This result is in line with what is expected with turbo codes because they always perform better with larger packet sizes [17]. Thus, the real-time commands that we generate at the DSN station can have payload of two bits only to reflect the selected code rate $r$.

As long as the turbo encoder to be used by the orbiter remains the same, there is no need to transmit the generated command continuously. We can send the commands periodically to ensure that the orbiter received our commands correctly. If the prediction model detects the bit-SNR moving from one range to another, which indicates the need to change the encoder, the selection command is automatically transmitted. In the cases where predicted channel bit-SNR is below the threshold to support any data reception at DSN, indicating weather dropout conditions, we have two separate options to consider:

1. The orbiter transmission parameter namely modulation index, is varied to achieve reasonable channel bit-SNR.

2. The orbiter is commanded to stop transmission until it receives a clear to send (CTS) command.

Although varying orbiter transmission system parameters will further improve our performance, we are not considering it at the moment.

### 3.2.4 Data Loss And Mitigation Approach

Space communications feature much noisier channels than terrestrial communications resulting in BERs with $10^{-5}$ being very common and even higher BERs in the order of $10^{-1}$ in the deep space environment [29]. This very high BERs contributes to higher loss of data, which means that all the transmitted encoded symbols are not successfully decoded at the receiver. Certain number of turbo encoded symbols are corrupted during the transmission, causing unsuccessful decoding of turbo packets and, in turn, RaptorQ symbol loss at the receiver.

The issue that we are addressing here is whether it is beneficial to transmit $(K_{\mathrm{SS}} + \Theta)$ or $(K_{\mathrm{SS}} + \Theta + y)$ encoded symbols from a file initially, where $y$ is the number additional symbols transmitted proactively to counter the effect of failed transmissions. We need to consider the trade-offs of transmission of $(K_{\mathrm{SS}} + \Theta)$ and $(K_{\mathrm{SS}} + \Theta + y)$ symbols in terms of channel throughput and time to deliver a file.

In the first approach, i.e., $y = 0$, we are not transmitting any additional symbols ever. It appears highly unlikely that all the transmitted $(K_{\mathrm{SS}} + \Theta)$ symbols will be received successfully at the earth station receiver due to underlying channel nature and associated FER. If no additional symbols are transmitted, even if one symbols gets corrupted over the downlink, Earth station receiver wont be able to decode the file and it has to send feedback requesting for additional symbols. This approach will result in highest link throughput as no transmission is being wasted, however, at the cost of increased file delivery time. The gain in throughput is very small in comparison to the delay in the file delivery time. Thus, it is really essential to transmit some optimal $y$ number of additional symbols along with the $(K_{\mathrm{SS}} + \Theta)$

symbols to get reasonable throughput and minimum possible file delivery time. This strong capability of being able to transmit additional symbols proactively comes from the use of RaptorQ code and if used efficiently overall performance of a transmission system can be enhanced drastically. Since a slightly larger number of RaptorQ encoded symbols are needed to recover all original source symbols, this redundancy must be quantified and reflected in the throughput calculation. In this section, we present our approach to achieve certain probability of successful data transmission by transmitting $y \in \{0, 1, 2, \cdots\}$ extra symbols so that the number of re-transmissions required is zero or kept to its minimum value.

Over the duration of a communication pass, for each symbol transmission and associated bit-SNR value, there is an associated turbo code rate $r^{(n)}$, with $n \in \{1, 2, \cdots, N_s\}$ and packet error probability $p^{(n)}$ that maximizes the channel throughput. Here, $r^{(n)}$ and $p^{(n)}$ respectively represent the turbo code rate and packet error rate associated with $n$th transmitted symbol, and $N_s$ represents the total number of symbols transmitted during the pass. Even though the bit-SNRs are correlated, the probability of a packet error $p^{(n)}$ at a bit-SNR value depends specifically upon the specific realization over the period that the packet is being transmitted. The realizations are drawn independently so the packet errors are independent. Thus, each symbol transmission is an independent trial and we have $N_s$ independent trials in a communication pass. Let $\tilde{y}_n$ be a random variable whose value is equal to 1 if trial $n$ is a failure, and 0 otherwise, represented as

$$\tilde{y}_n = \begin{cases} 1 & \text{failure,} \\ 0 & \text{otherwise.} \end{cases}$$

It follows that the number of failures out of the total $(K_{\text{SS}} + \Theta + y)$ transmitted symbols of a file represented by another random variable $\tilde{y}$ can be expressed as

$$\tilde{y} = \sum_{n=1}^{(K_{\text{SS}}+\Theta+y)} \tilde{y}_n, \tag{3.15}$$

and is Gaussian distributed. Thus, $\tilde{y}$ can be represented as

$$\begin{aligned} \tilde{y} &\sim \mathcal{N}\left( \sum_{n=1}^{(K_{\text{SS}}+\Theta+y)} E\left[\tilde{y}_n\right], \operatorname{Var}\left\{ \sum_{n=1}^{(K_{\text{SS}}+\Theta+y)} \tilde{y}_n \right\} \right), \\ \tilde{y} &\sim \mathcal{N}\left( \sum_{n=1}^{(K_{\text{SS}}+\Theta+y)} p^{(n)}, \sum_{n=1}^{(K_{\text{SS}}+\Theta+y)} p^{(n)}(1-p^{(n)}) \right), \end{aligned} \tag{3.16}$$

where $E\left[\tilde{y}_n\right] = p^{(n)}$ and $\operatorname{Var}\left\{\tilde{y}_n\right\} = p^{(n)}(1-p^{(n)})$.

Given that the number of failures represented by a random variable $\tilde{y}$ out of $(K_{\text{SS}} + \Theta + y)$ transmitted symbols of a file are Gaussian distributed, we compute the value of $y$ additional number of symbols to be transmitted apriori so that the future packet losses of that file are covered. To guarantee this, the value of $y$ should be chosen in such a way that the probability $P\left\{\tilde{y} \leq y\right\}$ that the number of failures are less than or equal to $y$ should be greater than or equal to some large $p_{\text{target}}$ target probability. Mathematically,

$$\begin{aligned}
P\left\{\tilde{y} \leq y\right\} &\geq p_{\text{target}}, \\
1 - P\left\{\tilde{y} \leq y\right\} &\leq 1 - p_{\text{target}}, \\
P\left\{\tilde{y} > y\right\} &\leq 1 - p_{\text{target}}, \\
P\left\{\frac{\tilde{y} - E\left[\tilde{y}\right]}{\sqrt{\operatorname{Var}\left\{\tilde{y}\right\}}} > \frac{y - E\left[\tilde{y}\right]}{\sqrt{\operatorname{Var}\left\{\tilde{y}\right\}}}\right\} &\leq 1 - p_{\text{target}}, \\
\mathrm{Q}\left(\frac{y - E\left[\tilde{y}\right]}{\sqrt{\operatorname{Var}\left\{\tilde{y}\right\}}}\right) &\leq 1 - p_{\text{target}}, \\
\left(\frac{y - E\left[\tilde{y}\right]}{\sqrt{\operatorname{Var}\left\{\tilde{y}\right\}}}\right) &\geq \mathrm{Q}^{-1}\left(1 - p_{\text{target}}\right), \\
y \geq \left(\sqrt{\operatorname{Var}\left\{\tilde{y}\right\}}\, \mathrm{Q}^{-1}\left(1 - p_{\text{target}}\right) + E\left[\tilde{y}\right]\right)&, \\
y = \left\lceil\left(\sqrt{\operatorname{Var}\left\{\tilde{y}\right\}}\, \mathrm{Q}^{-1}\left(1 - p_{\text{target}}\right) + E\left[\tilde{y}\right]\right)\right\rceil&. \qquad (3.17)
\end{aligned}$$

Basically, we can proceed in two different ways with computing and using $y$ values during a communication pass; a) using fixed $y$ value, and b) using adaptive $y$ value. Let us denote the value of $y$ for file $i$ by $y_i$ with $i \in \{1, 2, 3, \cdots, F_{\text{tx}}\}$, where $F_{\text{tx}}$ is the number of files transmitted in the pass. In *fixed approach*, a fixed $y$ value is used with all the files of a pass, i.e., $y_1 = y_2 = \cdots = y_{F_{\text{tx}}} = y$. At the beginning of each pass, the value of $y$ is computed utilizing the previous communication pass information, and relayed to the spacecraft.

In *adaptive approach*, $y$ is computed for each file separately at the spacecraft, i.e., $y_1, y_2, \cdots, y_{F_{\text{tx}}}$, on the basis of proportion of time different turbo encoders are used for telemetry and its corresponding FERs. The actual FER for each transmitted turbo coded packets cannot be known at the spacecraft. However, corresponding to the use of each turbo code rate, the maximum and minimum value of FER is a known quantity, and the average FER can be easily computed. Thus, the adaptive approach is further sub-divided into a) adaptive average, b) adaptive best, and c) adaptive worst approach depending on the FERs used for $y$ calculation. As an example, in adaptive best approach, for each turbo encoded packet, corresponding FER is assumed to be minimum and value of $y$ is computed using the minimum FERs.

## 3.3 Protocols

In this section, we present all the necessary details and sub-classes of the DCSM, the *genie* and the *static* approach, along with mathematical formulation of channel throughput and $y$ value computation for the methods. A pictorial representation of all the methods and their categories is presented in Figure 3.8.

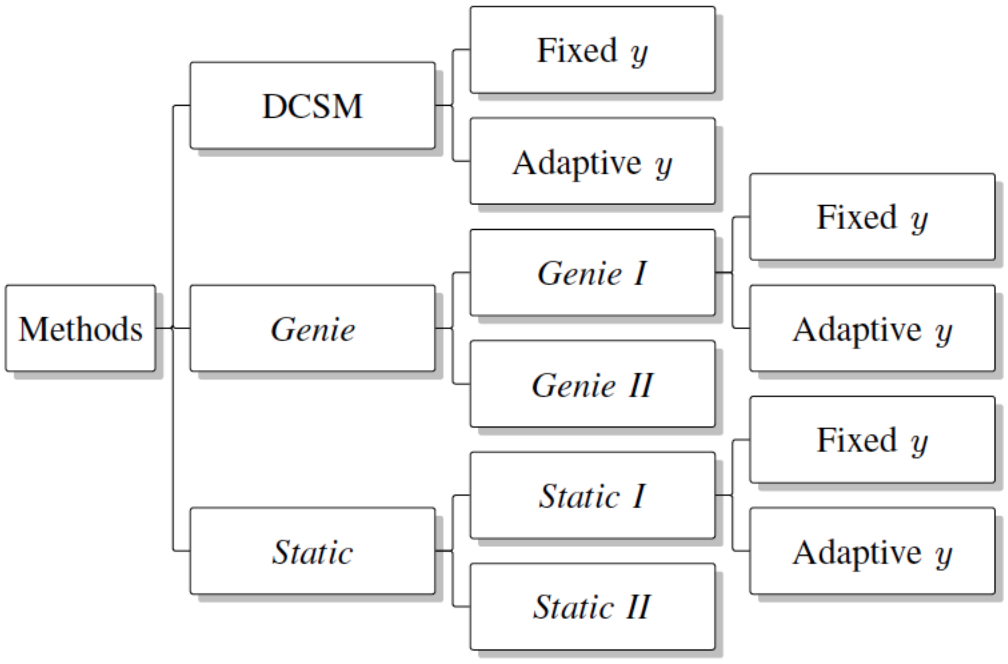


Figure 3.8: A tree showing the *genie*, the DCSM and the *static* method and their sub-categories.

### 3.3.1 DCSM

Main features of our proposed DCSM approach are RaptorQ at application layer, dynamic selection of turbo codes at physical layer, our proposed channel condition prediction model to predict channel bit-SNR and FER into the future, $y$ additional symbols transmitted for each files and feedback of lost packets. The DCSM can be further divided into a) adaptive DCSM, and b) fixed DCSM. A stepwise elaboration on execution of the DCSM is given below:

1. At the spacecraft, for each file to be telemetered to the Earth station, $(K_{\mathrm{SS}} + \Theta + y)$ number of RaptorQ encoded symbols are generated, and handed over to the physical layer for channel coding.

2. Appropriate turbo encoder is used to encode each RaptorQ encoded symbol. The turbo encoder selection process is summarized as follows:

   (a) At the Earth station, at time $t_0$ a prediction of future channel bit-SNR $X_{t_0+\mathrm{RTT}}$ that will occur at time $t_0 + \mathrm{RTT}$ is made.

   (b) A real-time command identifying the turbo encoder that maximizes channel throughput at time $t_0 + \mathrm{RTT}$ is sent over to the distant spacecraft.

   (c) The spacecraft receives the real-time command at time $t_0 + \mathrm{RTT}/2$, command and data handling part of the spacecraft extracts the required information. The specified turbo encoder is the appropriate turbo encoder to be used at time $t_0 + \mathrm{RTT}/2$.

3. Earth station keeps record of the number of packets successfully received and the number of packets lost from a file. These informations are easily extracted by comparing the number of successfully received packets and ESI of the RaptorQ encoded symbol contained in the last received packet.

4. Once the count of lost packets of that particular file becomes greater than $y$, as soon as further packet losses are recorded, feedback(s) requesting transmission of that number of additional packets of the file is generated and transmitted to the spacecraft.

5. The spacecraft continues with the transmission of $(K_{\text{SS}} + \Theta + y)$ symbols from each file in serial manner one after the another. As soon as it receives feedback on lost number of packets from earlier transmitted files, transmission of symbols from current file is put to halt and that number of new packets from the previous file is transmitted. Once done, the spacecraft continues with the current file.

6. As soon as the Earth station receives $(K_{\text{SS}} + \Theta)$ number of packets successfully, the file is successfully decoded and no further feedbacks for this file will be created.

During a communication pass, $g_i$ be the proportion of time during which the actual channel bit-SNR is such that the packets are transmitted at rate $r_i$, $r_i \in \{0, 1/2, 1/3, 1/4, 1/6\}$. Rate $r_i = 0$ represents the weather dropout conditions at Earth station and the channel bit-SNR is below the acceptable range ($< -0.5$ dB) to support any data reception. Accuracy of our channel condition prediction model has significant impact on performance of the DCSM approach. The channel bit-SNR prediction model predicts $d_i$ as the proportion of time during which channel bit-SNR is such that the packets are transmitted at rate $r_i$. If the prediction method used is highly accurate, the $d_i$ is going to be almost same as $g_i$. Over the duration of a communication pass of duration $T$ s, the total time spent transmitting at rate $r_i$ is $T'_i = d_i T$. However, in reality, it should have been $T_i = g_i T$. Thus, the total duration of a pass can be written as $T = \sum_{r_i} T'_i = \sum_{r_i} T_i$.

Length of a rate $r_i$ turbo encoded packet is

$$L_p^{(r_i)} = (36 + K)/r_i \quad \text{bits.} \tag{3.18}$$

Data transmission rate $R_d^{(r_i)}$ for turbo codes with code rate $r_i$ is related to the bits transmission rate $R_b$ of the channel as

$$R_d^{(r_i)} = \frac{(K - 32)}{(K + 36)} R_b \, r_i \quad \text{bps.} \tag{3.19}$$

Over the duration of a communication pass, the number of symbols (packets) transmitted $N_{\text{tx}}^{(d)}$ is given as

$$N_{\text{tx}}^{(d)} = \sum_{r_i \neq 0} r_i \, d_i \, T \, R_b \frac{1}{(K + 36)}. \tag{3.20}$$

The superscript $(d)$ is to represent that all this corresponds to the DCSM approach.

For a communication pass, let us define $p_{ii}$, $i \in \{A, B, C, D, E\}$ as the FER for the case when bit-SNR range is predicted correctly, and $p_{ij}$, $i, j \in \{A, B, C, D, E\}$ with $i \neq j$ as the FER for the case when bit-SNR range is incorrectly predicted. Out of the $(K_{\text{SS}} + \Theta + y)$ transmitted symbols of a file, lets assume that the bit-SNR range was accurately predicted for $k_{\text{ss}}$ number of transmitted symbols and remaining $((K_{\text{SS}} + \Theta + y) - k_{\text{ss}})$ symbols were transmitted when bit-SNR prediction was incorrect. Number of failures $\tilde{y}$ out of $(K_{\text{SS}} + \Theta + y)$ transmitted symbols of a file is Gaussian distributed with mean

$$E[\tilde{y}] = \left( \sum_{n=1}^{k_{\text{SS}}} p_{ii}^{(n)} + \sum_{n=k_{\text{SS}}+1}^{(K_{\text{SS}}+\Theta+y)} p_{ij}^{(n)} \right),$$

and variance

$$\mathrm{Var}\left\{\tilde{y}\right\} = \left(\sum_{n=1}^{k_{\mathrm{SS}}} p_{ii}^{(n)}(1-p_{ii}^{(n)}) + \sum_{n=k_{\mathrm{SS}}+1}^{(K_{\mathrm{SS}}+\Theta+y)} p_{ij}^{(n)}(1-p_{ij}^{(n)})\right)$$

, where superscript $(n)$ denotes the FER corresponding to the $n$th transmitted symbol. Theoretically, using (3.17) along with the $E\left[\tilde{y}\right]$ and the $\mathrm{Var}\left\{\tilde{y}\right\}$, the $y$ that need to be transmitted apriori in order to cover all the possible symbol losses of a file can be computed.

Similarly, the total number of failures represented by another random variable $\tilde{Y}^{(d)}$ out of the total symbol transmissions $N_{\mathrm{tx}}^{(d)}$ over the duration of a communication pass is also normal distributed. The $\tilde{Y}^{(d)}$ can be represented as sum of independent and identically distributed random variables $\tilde{y}_m$ representing the number of failures in $m$th transmitted file

$$\tilde{Y}^{(d)} = \sum_{m=1}^{F_{\mathrm{tx}}^{(d)}} \tilde{y}_m, \tag{3.21}$$

where $F_{\mathrm{tx}}^{(d)}$ is the total number of files transmitted over the duration of a communication pass. Average channel throughput $T_H^{(d)}$, in terms of number of symbols received at the receiver per s, is $T_H^{(d)} = (N_{\mathrm{tx}}^{(d)} - Y^{(d)})/T$, where $Y^{(d)}$ is the number of failed transmissions during the pass.

As we know that the algorithm to predict channel bit-SNR one round trip time into the future is continuously executed at the Earth station and optimal turbo code rate information is transmitted to the orbiter. Along with the turbo code rate, the value of $y$ for the corresponding file needs to be computed at the Earth station using predicted bit-SNRs and FERs, and transmitted to the orbiter so that $(K_{\mathrm{SS}} + \Theta + y)$ number of symbols could be transmitted from the orbiter. In order to be able to compute the value of $y$ using (3.17), we first need to compute $\mathrm{Var}\left\{\tilde{y}\right\}$ and $E\left[\tilde{y}\right]$, for which exact knowledge of $p_{ii}^{(n)}$ and $p_{ij}^{(n)}$ is needed, which is not possible to do in real implementation as we will need to keep track of every single bit-SNR predictions and corresponding FERs. There is a lot of complexity and uncertainty involved due to the involvement of prediction model, along with the need to keep track of every single bit-SNR predictions and corresponding FERs for computation of $y$. Due to which, we won't be able to compute the actual value of $y$ in real implementation, and need to come up with an alternative method to simplify the situation and compute the value of $y$. Thus, to simplify the situation, we propose fixed DCSM and adaptive DCSM as our alternative approach.

#### 3.3.1.1 Fixed DCSM

In this approach, we compute value of $y = y_1 = y_2 = \cdots = y_{F_{\text{tx}}^{(d)}}$ utilizing actual channel bit-SNR profile and our predicted bit-SNR profile of previous communication pass. By using statistics of the past communication pass, we obtain statistics of proportion of time $\alpha_i$ bit-SNR is in a particular range $i$, proportion of time $\beta_i$ we predicted bit-SNR in a particular range $i$, proportion of time $\beta_{ij}, i = j$ we predicted bit-SNR ranges correctly as well as incorrectly denoted by $\beta_{ij}, i \neq j$. This can be summarized in tabular form as presented in Table 3.4.

The information presented in the table is expressed in a matrix form as shown by the matrix $\boldsymbol{\beta}$ and is termed as range prediction matrix as

$$\boldsymbol{\beta} = [\beta_{ij}]\,, \text{ for } \quad i, j \in \{A, B, C, D, E\}\,. \tag{3.22}$$

By averaging over all the possible FER values of a particular bit-SNR range $i$, we obtain average FER corresponding to each ranges represented as $p_{i_{\text{avg}}}$. For a particular range $i$, actual average FER value is $p_{i_{\text{avg}}}$. However, due to the range estimation error, in reality, the average FER $\hat{p}_i$ experienced by the rate $r_i$ turbo

Table 3.4: An illustration of an actual and predicted bit-SNR occurrence proportion of a communication pass.

| | Predicted SNR range proportion | | | | |
|---|---|---|---|---|---|
| Proportion of SNRs | Range A | Range B | Range C | Range D | Range E |
| $\alpha_A$ | $\beta_{AA}$ | $\beta_{AB}$ | $\beta_{AC}$ | $\beta_{AD}$ | $\beta_{AE}$ |
| $\alpha_B$ | $\beta_{BA}$ | $\beta_{BB}$ | $\beta_{BC}$ | $\beta_{BD}$ | $\beta_{BE}$ |
| $\alpha_C$ | $\beta_{CA}$ | $\beta_{CB}$ | $\beta_{CC}$ | $\beta_{CD}$ | $\beta_{CE}$ |
| $\alpha_D$ | $\beta_{DA}$ | $\beta_{DB}$ | $\beta_{DC}$ | $\beta_{DD}$ | $\beta_{DE}$ |
| $\alpha_E$ | $\beta_{EA}$ | $\beta_{EB}$ | $\beta_{EC}$ | $\beta_{ED}$ | $\beta_{EE}$ |
| Proportion of pred. SNRs | $\beta_A$ | $\beta_B$ | $\beta_C$ | $\beta_D$ | $\beta_E$ |

encoded packets are going to be different from $p_{i_{\text{avg}}}$. Thus, using the range prediction matrix $\boldsymbol{\beta}$ of (3.22) and $p_{i_{\text{avg}}}$, we compute an estimate of FER $\hat{p}_i$ experienced by the rate $r_i$ turbo encoded packets as $\hat{p}_i = \sum_j \beta_{ij}\, p_{j_{\text{avg}}}$. Total time spent transmitting at rate $r_i$ is $\beta_i T$, and the total number of symbols sent at rate $r_i$ is $\hat{n}_i = r_i\, \beta_i\, T$. Therefore, the proportion of symbols $\pi_i$ that are sent at rate $r_i$ is

$$\pi_i = \frac{\hat{n}_i}{\sum_j \hat{n}_j} = \frac{r_i\, \beta_i\, T}{\sum_j r_j\, \beta_j\, T} = \frac{r_i\, \beta_i}{\sum_j r_j\, \beta_j}. \tag{3.23}$$

We can also assume that the proportion holds true for each file transmitted. Thus, the new mean and the variance of $\tilde{y}$ is

$$E\left[\tilde{y}\right] = \sum_i \pi_i \left(K_{\text{SS}} + \Theta + y\right) \hat{p}_i,$$

and

$$\operatorname{Var}\left\{\tilde{y}\right\} = \sum_i \pi_i \left(K_{\text{SS}} + \Theta + y\right) \hat{p}_i \left(1 - \hat{p}_i\right)$$

, respectively. The value of $y$ could be easily computed using $E\left[\tilde{y}\right]$, $\operatorname{Var}\left\{\tilde{y}\right\}$, and (3.17). As there is no other way for us to compute the actual value of $y$ for a pass practically, this is the simplest approach we could use to estimate the $y$.

#### 3.3.1.2 Adaptive DCSM

In this approach, $y$ is computed for each file separately at the spacecraft, i.e., $y_1, \cdots, y_{F_{\mathrm{tx}}}$, on the basis of the proportion of symbols $\pi_i^{(m)}$ transmitted at a particular turbo code rates $r_i$ over the duration of the $m$th file transfer and its corresponding FER values. Since, exact FER values corresponding to each turbo encoded packet transfer cannot be known at the spacecraft, we have to either go with best case FER $p_{i_{\min}}$, average case FER $p_{i_{\mathrm{avg}}}$ or worst case FER $p_{i_{\max}}$ values corresponding to the use of a particular rate $r_i$ turbo code.

If we consider best case FER for a communication pass, the mean is

$$E\left[\tilde{y}_m\right] = \sum_i \pi_i^{(m)} \left(K_{\mathrm{SS}} + \Theta + y_m\right) p_{i_{\min}}$$

and the variance is

$$\operatorname{Var}\left\{\tilde{y}_m\right\} = \sum_i \pi_i^{(m)} \left(K_{\mathrm{SS}} + \Theta + y_m\right) p_{i_{\min}} \left(1 - p_{i_{\min}}\right),$$

respectively. Similarly, we could use average FER scenario or worst FER scenario as well. For all these cases, $y_m$ is computed using corresponding $E\left[\tilde{y}_m\right]$, $\operatorname{Var}\left\{\tilde{y}_m\right\}$ and (3.17).

### 3.3.2 Genie I

The premise of the *genie* method is that the distant spacecraft has exact knowledge of actual channel bit-SNR condition and FERs corresponding to each transmitted turbo packets. Keeping this important fact under consideration, the *genie* approach is categorized into two types; a) *genie I*, and b) *genie II*. Main features of *genie I* are same as that of the DCSM except the fact that actual future channel condition at the Earth station is known to the spacecraft apriori. Stepwise execution of *genie I* is same as that of the DCSM presented above, except, in point 2), appropriate turbo encoder is selected based on exact knowledge of future channel bit-SNR condition.

Length $L_p^{(r_i)}$ and data transmission rate $R_d^{(r_i)}$ for different code rate $r_i$ turbo encoded packets are given by (3.18) and (3.19), respectively. Over the duration of a communication pass, the number of symbols $N_{\text{tx}}^{(gI)}$ transmitted are

$$N_{\text{tx}}^{(gI)} = \sum_{r_i \neq 0} g_i \, r_i \, T \, R_b \, \frac{1}{(K+36)}. \tag{3.24}$$

Here, the superscript $(gI)$ represents the value associated to the *genie I* method. The number of failed transmissions $\tilde{y}$ out of $(K_{\text{SS}} + \Theta + y)$ transmitted symbols of a file is Gaussian distributed given as

$$\tilde{y} \sim \mathcal{N}\left( \sum_{n=1}^{(K_{\text{SS}}+\Theta+y)} p_i^{(n)}, \sum_{n=1}^{(K_{\text{SS}}+\Theta+y)} p_i^{(n)}(1-p_i^{(n)}) \right), \tag{3.25}$$

where $p_i^{(n)}$ is the FER of $n$th transmitted symbol when optimal rate $r_i$ is used. Similar to that of the DCSM, average channel throughput $T_H^{(gI)}$, in terms of number of symbols received at the receiver per sec can be easily obtained. Details of fixed *genie I* and adaptive *genie I* are presented below.

### 3.3.2.1 Fixed genie I

Using statistics of the past communication pass, we obtain information on proportion of time $\alpha_i$ bit-SNR is in a particular range $i$. We now need to compute an estimate of FER $\hat{p}_i$ when bit-SNR is in range $i$. As we assume exact knowledge of actual channel bit-SNR in the *genie* approach, we have $\hat{p}_i = p_{i_{\text{avg}}}$.

Over the duration of a communication pass of $T$ s, the total time spent transmitting at rate $r_i$ is $T_i = g_i T$. Number of symbols sent at rate $r_i$ is $\hat{n}_i = r_i\, g_i\, T$. Therefore, the proportion of symbols $\pi_i$ transmitted at rate $r_i$ turbo packet is

$$\pi_i = \frac{\hat{n}_i}{\sum_j \hat{n}_j} = \frac{r_i\, g_i\, T}{\sum_j r_j\, g_j\, T} = \frac{r_i\, g_i}{\sum_j r_j\, \alpha_j}. \tag{3.26}$$

The simplified $E\left[\tilde{y}\right]$ and $\mathrm{Var}\left\{\tilde{y}\right\}$ is $\sum_i \pi_i \left(K_{\mathrm{SS}} + \Theta + y\right) \hat{p}_i$ and $\sum_i \pi_i \left(K_{\mathrm{SS}} + \Theta + y\right) \hat{p}_i \, (1 - \hat{p}_i)$, respectively.

#### 3.3.2.2 Adaptive genie I

This approach is same as has been described in Section 3.3.1.2 above. Details are same except the fact that actual channel bit-SNR profile is used here instead of predicted channel bit-SNRs of DCSM for calculations.

### 3.3.3 Genie II

To study the performance of *genie* approach without RaptorQ codes, we have *genie II* characterized by dynamic selection of turbo codes at physical layer, and actual knowledge of channel bit-SNR and FER. The capability to transmit $y$ additional symbols of each file, and feedback on the count of number of unsuccessfully received packets at the receiver with *genie I* was the result of RaptorQ in application layer. In *genie II*, depending on the value of future channel bit-SNRs, multiple retransmission of a packet is done proactively so that the probability of success of the transmitted packet is made 1. A stepwise elaboration on execution of the *genie II* is given below:

1. At the spacecraft, each file to be telemetered is divided into $N_{\mathrm{SS}}$ ADUs of size $K = 8920$ bits. These ADUs are handed over to dynamic turbo encoder in serial manner.

2. Appropriate turbo encoder is selected based on exact knowledge of channel bit-SNR condition at the Earth station RTT/2 into the future.

3. Depending on the value of FER corresponding to that bit-SNR, the packet is transmitted one or more times (as required) proactively so that the probability of success of the transmitted packet is made 1. As an example, a telemetry packet is being telemetered and its corresponding FER is going to be 0.5. This means that the probability that the packet transmission will succeed is 1/2. Now to make the probability of success of the packet equals 1.0, the packet is transmitted twice.

4. Once all the $N_{\mathrm{SS}}$ ADUs of the file is telemetered, begin transmissions of ADUs from the following file.

Length of a rate $r_i$ turbo encoded packet is given by (3.18). The data transmission rate $R_d^{(r_i)}$ for different code rate $r_i$ turbo code is given as $R_d^{(r_i)} = K\, R_b\, r_i/(K+36)$ bps. Over the duration of a communication pass, the number of symbols $N_{\mathrm{tx}}^{(gII)}$ transmitted is computed using (3.24).

The number of unsuccessful packet transmissions out of $N_{\mathrm{tx}}^{(gII)}$ packets transmitted is Gaussian distributed with mean $\sum_{n=1}^{N_{\mathrm{tx}}^{(gII)}} p_i^{(n)}$, and variance $\sum_{n=1}^{N_{\mathrm{tx}}^{(gII)}} p_i^{(n)}(1 - p_i^{(n)})$. Utilizing this, average channel throughput $T_H^{(gII)}$, in terms of number of symbols successfully received at the receiver per s can be easily obtained.

### 3.3.4 Static I

In *static* method, unlike in the DCSM and the *genie*, a fixed (8920, 1/2) turbo encoder is used at the physical layer throughout the communication pass. The *static* method can be categorized into two types; a) *static I*, and b) *static II*. Stepwise execution of *static I* is same as that of the DCSM, except, in point 2), turbo encoder is always kept fixed to (8920, 1/2) during communication.

The length of turbo encoded packets and data transmission rate $R_d$ remains same throughout the entire pass duration, and are given as

$$L_p = 2\,(36 + K) \quad \text{bits}, \tag{3.27}$$

$$R_d = \frac{(K-32)}{2\,(K+36)}\,R_b \quad \text{bps}. \tag{3.28}$$

The number of packets $N_{\text{tx}}^{(sI)}$ transmitted over the duration of a communication pass is given as

$$N_{\text{tx}}^{(sI)} = T\,R_b\,\frac{1}{2\,(K+36)}. \tag{3.29}$$

The number of failed transmissions $\tilde{y}$ out of $(K_{\text{SS}} + \Theta + y)$ transmitted symbols of a file is Gaussian distributed

$$\tilde{y} \sim \mathcal{N}\left(\sum_{n=1}^{(K_{\text{SS}}+\Theta+y)} p_{\frac{1}{2}}^{(n)}, \sum_{n=1}^{(K_{\text{SS}}+\Theta+y)} p_{\frac{1}{2}}^{(n)}(1-p_{\frac{1}{2}}^{(n)})\right), \tag{3.30}$$

where, the $p_{\frac{1}{2}}^{(n)}$ is FER of $n$th transmitted symbol with $(8920, 1/2)$ turbo code. Similar to that of the DCSM, average channel throughput $T_H^{(sI)}$, in terms of number of symbols received at the receiver per s can be easily obtained.

### 3.3.4.1 Fixed static I

The fact that a fixed $(8920, 1/2)$ turbo encoder is used throughout a communication pass can also be looked at as a case where an arbitrary channel prediction model wrongly predicted all the actual bit-SNR ranges as Range E. Thus, in this case, the range prediction matrix $\boldsymbol{\beta}$ has $\beta_{iE} = 1.0$ and $\beta_{ij} = 0.0$ for $j \in \{A, B, C, D\}$.

Using the $\boldsymbol{\beta}$ and $p_{\text{E}_{\text{avg}}}$ (average FER corresponding to range $E$), an estimate of FER $\hat{p}_{\text{E}}$ experienced by the rate 1/2 turbo encoded packets is $\hat{p}_{\text{E}} = p_{\text{E}_{\text{avg}}}$. Similarly, we have $\pi_{\text{E}} = 1$. The mean $E\left[\tilde{y}\right]$ and variance $\text{Var}\left\{\tilde{y}\right\}$ of $\tilde{y}$ is $(K_{\text{SS}} + \Theta + y)\,\hat{p}_{\text{E}}$ and

$(K_{\mathrm{SS}} + \Theta + y)\,\hat{p}_{\mathrm{E}}\,(1 - \hat{p}_{\mathrm{E}})$, respectively.

### 3.3.4.2 Adaptive static I

Since we have $\pi_{\mathrm{E}}^{(m)} = 1$ for all the files, value of $y_m$ is going to be same for all the files, that is, $y = y_1 = y_2 = \cdots = y_{F_{\mathrm{tx}}}$. Thus, the mean and the variance of $\tilde{y}$ is $E[\tilde{y}] = ((K_{\mathrm{SS}} + \Theta + y)\,p_{\mathrm{E_{min}}})$ and $\mathrm{Var}\{\tilde{y}\} = ((K_{\mathrm{SS}} + \Theta + y)\,p_{\mathrm{E_{min}}}\,(1 - p_{\mathrm{E_{min}}}))$, respectively, in best FER scenario. The mean and the variance of $\tilde{y}$ in case of average FER scenario and worst FER scenario can be obtained following the same approach.

### 3.3.5 Static II

To study the performance of *static* approach without RaptorQ codes, in this scheme, irrespective of the actual channel bit-SNR conditions at the Earth station, orbiter continuously transmits $(8920, 1/2)$ encoded turbo packets. The data transmission rate $R_d$ is $R_d = K\,R_b/(2\,(K + 36))$ bps. The number of packets $N_{\mathrm{tx}}^{(sII)}$ transmitted during the communication pass is same as given in (3.29). The number of failed transmissions $\tilde{y}$ out of $(K_{\mathrm{SS}} + \Theta + y)$ transmitted symbols of a file are also Gaussian distributed as given in (3.30).

## 3.4 Prediction Model

In this section, we propose a channel bit-SNR prediction model that uses apriori sky brightness temperature information and is based on the idea of first order autoregressive process AR(1). The fact that the weather conditions in an area have a certain consistency during a small period of time allows the design of a useful channel prediction model and it is sufficient to predict future channel conditions once every second. All the measurable factors that characterize channel condition can be expressed in terms of channel bit-SNR value. Hence, instead of developing a prediction model to predict sky brightness temperature, we develop a model that predicts channel bit-SNR values one RTT into the future. We proceed with the future channel bit-SNR prediction in two phases, details of which are covered below.

### 3.4.1 Preliminary Prediction

We denote actual temporal channel bit-SNR of a communication pass by a time series $\mathbf{X} = \{X_t : t \in \{0, 1, 2, \cdots, T_e\}\}$, where $X_t$ represents the actual channel bit-SNR at time $t$, and $t = 0$ and $t = T_e$ represent beginning and end of the communication pass respectively. For each $X_t$, we obtain an estimate $\hat{X}_t$, details of which are presented below. Thus obtained estimates of temporal channel bit-SNR $\mathbf{X}$ is denoted by another time series $\hat{\mathbf{X}} = \left\{\hat{X}_t : t \in \{0, 1, 2, \cdots, T_e\}\right\}$, where $\hat{X}_t$ represents the preliminary prediction of actual channel bit-SNR $X_t$.

NASA's DSN supports multiple spacecrafts and service users. All the services and activities requiring DSN are scheduled in three phases: long range planning and forecasting (starts six months to one year before execution), mid-range scheduling (starts roughly 4 - 5 months before execution), and near real-time scheduling (starts roughly 8 weeks before the execution through the execution) [30]. Thus for each communication pass, we have information of the Earth and the Mars geometry, pass duration, visibility, and elevation angle of communicating DSN antenna in advance. Let us denote antenna elevation angle over a duration of communication pass by a time series $\boldsymbol{\theta} = \{\theta_t : t \in \{0, 1, 2, \cdots, T_e\}\}$, where $\theta_t$ represents the DSN antenna elevation angle at time $t$. Another time series $\mathbf{R} = \{R_t : t \in \{0, 1, 2, \cdots, T_e\}\}$ represents the Earth-Mars range over the duration of the pass and $R_t$ is the Earth-Mars range at time $t$.

During communication with a distant spacecraft, the DSN site continuously records sky-brightness temperature measurements. Let us denote actual temporal sky brightness temperature of a communication pass as measured by the DSN site by a time series $\mathbf{T}_{\text{sky}} = \left\{T_t^{(\text{sky})} : t \in \{0, 1, 2, \cdots, T_e\}\right\}$, where $T_t^{(\text{sky})}$ is the actual sky-brightness temperature at time $t$. For each time index $t$, an average of sky-brightness temperature $\mathcal{T}_t^{(\text{sky})}$ over the duration of past '$N$' secs, is computed

$$\mathcal{T}_t^{(\text{sky})} = \sum_{i=0}^{N} \frac{T_{t-i}^{(\text{sky})}}{N}, \tag{3.31}$$

where $\mathcal{T}_t^{(\text{sky})}$ is the computed apriori sky brightness temperature for the instant $t$ and $\boldsymbol{\mathcal{T}}_t^{(\text{sky})} = \left\{\mathcal{T}_t^{(\text{sky})} : t \in \{0, 1, 2, \cdots, T_e\}\right\}$ represents the computed apriori sky-brightness temperature over the duration of the communication pass. For our simulation purpose we are using $N = 10$ secs. Since immediate past few seconds of weather condition has higher correlation with present condition as compared to the conditions few hours or days before, $N = 10$ secs gives a better approximation of $\mathcal{T}_t^{(\text{sky})}$.

For each time instant $t$, using $\mathcal{T}_t^{(\text{sky})}$ along with $\theta_{t+T_{\text{RTT}}}$ and $R_{t+T_{\text{RTT}}}$, we compute corresponding value of channel bit-SNR $\hat{X}_{t+T_{\text{RTT}}}$. The obtained $\hat{X}_{t+T_{\text{RTT}}}$ is an estimate of actual channel bit-SNR $X_{t+T_{\text{RTT}}}$. We use the obtained temporal channel bit-SNR estimate $\hat{\mathbf{X}}$ as prior channel bit-SNR information during real time prediction phase.

### 3.4.2 Real Time Prediction

In order to further correct and improve the accuracy of preliminary prediction, we utilize the real-time measurements and an autoregressive process of order one AR(1). In the first part of this section, we describe the motivation behind using AR(1) process in our prediction model and present its brief description. In the second part, we present our prediction algorithm and results demonstrating its prediction accuracy.

#### 3.4.2.1 AR(1) Process

We first verify that $\mathbf{X} = \{X_t : t \in \{0, 1, 2, \cdots, T_e\}\}$ can be defined by an AR(1) process by analyzing partial autocorrelation (PAC) of $\mathbf{X}$ of different communication passes. Details of partial autocorrelation function (PACF) is presented in Appendix A. For any time series $\mathbf{X}$, the PAC at lag $k$ is the autocorrelation between $X_t$ and $X_{t+k}$ with the linear dependence of $X_t$ on $X_{t+1}$ through $X_{t+k-1}$ removed. The PACF of $\mathbf{X}$ at lag $k$, denoted by $\alpha_{\mathbf{X}}(k)$ is defined as [31], [32]

$$\alpha_{\mathbf{X}}(k) = \mathrm{R}\left(\left(X_t - E\left[X_t | X_{t-1}, \cdots, X_{t-k+1}\right]\right), \left(X_{t-k} - E\left[X_{t-k} | X_{t-1}, \cdots, X_{t-k+1}\right]\right)\right), \tag{3.32}$$

where $k = 0, 1, 2, \cdots$, and

$$\mathrm{R}(X, Y) = \frac{E\left[\left(\mathbf{X} - E\left[\mathbf{X}\right]\right)\left(\mathbf{Y} - E\left[\mathbf{Y}\right]\right)\right]}{\sigma_{\mathbf{X}}\,\sigma_{\mathbf{Y}}}$$

represents the correlation between two time series $\mathbf{X}$ and $\mathbf{Y}$, and $E\left[\mathbf{X}\right]$ and $E\left[\mathbf{Y}\right]$ are the means respectively for $\mathbf{X}$ and $\mathbf{Y}$ and $\sigma_{\mathbf{X}}$, $\sigma_{\mathbf{Y}}$ are their standard deviations.

The PACF is very useful in identifying an autoregressive (AR) process. If our original process is autoregressive of order $p$ represented as AR($p$), then for $k > p$, we should have $\alpha_{\mathbf{X}}(k) = 0$. This provides a very useful test for whether or not a process is autoregressive. A series can be represented as a pure AR process, if we observe

1. the autocorrelation function dies out in an exponential or sinusoidal fashion, and

2. the partial autocorrelation cuts off after lag $p$.

From Figure 3.9, we can observe that autocorrelation function of channel bit-SNR of a communication pass dies out in sinusoidal fashion. From Figure 3.10, we can see that partial autocorrelation function of the channel bit-SNR with lag of one

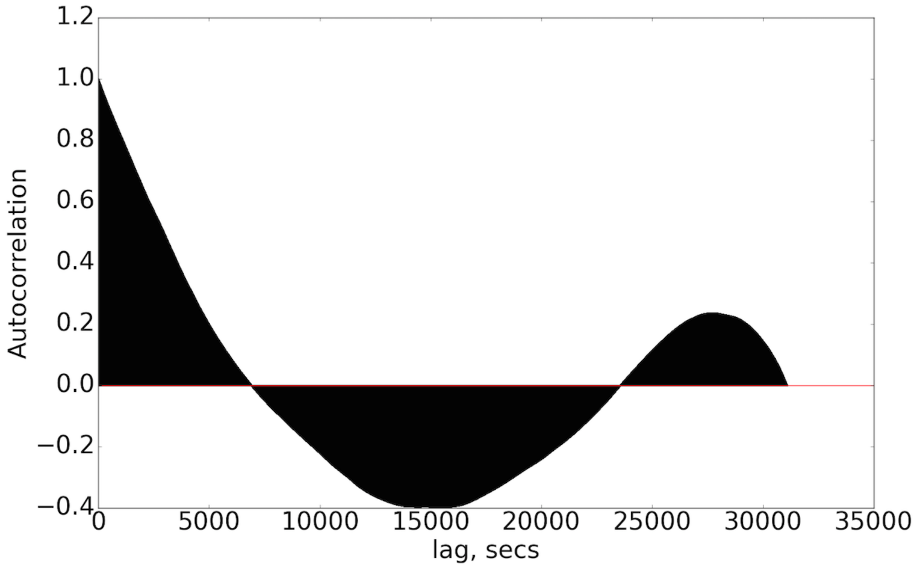


Figure 3.9: Autocorrelation of bit-SNR of a communication pass on September 26, 2005.

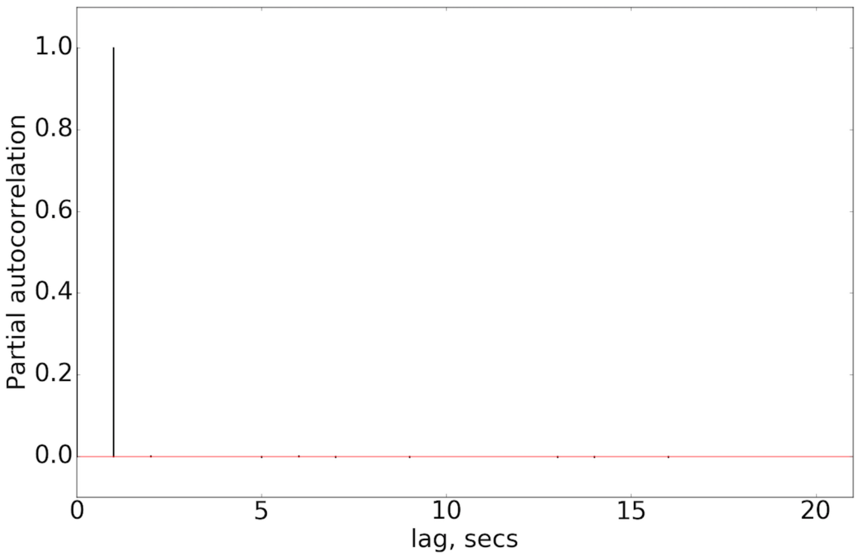


Figure 3.10: Partial autocorrelation of bit-SNR of a communication pass on September 26, 2005.

second is 1, i.e., $\alpha_{\mathbf{X}}(k) = 1$ for $k = 1$, and is 0 with lag more than a second, i.e., $\alpha_{\mathbf{X}}(k) \simeq 0$ for $k > 1$. This implies that the channel bit-SNR $X_t$ depends on $X_{t-1}$

but not on previous values and hence can be defined by an AR(1) process given as

$$X_t = c + \varphi\, X_{t-1} + \varepsilon_t\,, \tag{3.33}$$

where $c$ is a constant, $\varphi \in (0, 1)$ is a constant multiplicative factor, and $\varepsilon_t$ is a zero mean and constant variance white noise process at time $t$.

The $AR(1)$-process is a discrete time analogy of the continuous *Ornstein-Uhlenbeck* process and can be cast into Ornstein-Uhlenbeck equivalent form [33]. The Ornstein-Uhlenbeck process [34] is a stochastic Gauss–Markov process that describes the velocity of a massive Brownian particle under the influence of friction [35]. Utilizing the Ornstein-Uhlenbeck equivalent form of AR(1) process, we can approximate the relationship between $X_t$ and $X_{t+n}$ as given below

$$\mathrm{E}\left[X_{t+n}|X_t\right] = \mu\left[1 - (1 - \phi)^n\right] + X_t\,(1 - \phi)^n\,, \tag{3.34}$$

where $X_{t+n}$ represents bit-SNR value $n$ periods into the future, and $\mu$ and $|\phi| < 1$ are the model parameters. The $\mu = c/(1 - \varphi)$ is long term mean and $\phi = (1 - \varphi)$ is the rate of mean reversion of the process. Using equation (3.34), based on current state of the time series we can forecast its value an arbitrary number of periods into the future.

#### 3.4.2.2 Prediction Algorithm

At the communicating DSN station, for each time index $t$ during the period of communication, we compute the estimation error $e_t = X_t - \hat{X}_t$ between the actual measured bit-SNR $X_t$ and its preliminary estimation $\hat{X}_t$. The $e_t$ can also be characterized by an AR(1) process and hence the relation between $e_t$ and $e_{t+n}$ is given as

$$\mathrm{E}\left[e_{t+n}|e_t\right] = \mu_e\left[1 - (1 - \phi_e)^n\right] + e_t\,(1 - \phi_e)^n\,, \tag{3.35}$$

where $e_{t+n}$ represents estimation error $n$ periods into the future, $\mu_e$ is long term mean and $|\phi_e|< 1$ is rate of mean reversion of the process.

Utilizing the value of $e_t$ and (3.35), we obtain an estimation of estimation error one s into the future $\hat{e}_{t+1}$ and one RTT into the future $\hat{e}_{t+\mathrm{RTT}}$. We then correct the preliminary prediction $\hat{X}_{t+1}$ by $\hat{e}_{t+1}$ amount so that the updated value is $\hat{X}_{t+1} = (\hat{X}_{t+1} + \hat{e}_{t+1})$. Similarly, we update the preliminary prediction $\hat{X}_{t+\mathrm{RTT}}$ by $\hat{e}_{t+\mathrm{RTT}}$ amount so that the updated $\hat{X}^{(\mathrm{pred})}_{t+\mathrm{RTT}} = (\hat{X}_{t+\mathrm{RTT}} + \hat{e}_{t+\mathrm{RTT}})$. All the updated $\hat{X}^{(\mathrm{pred})}_{t+\mathrm{RTT}}$ are considered as our final prediction and is represented in time series form as $\hat{\mathbf{X}}_{\mathrm{pred}} = \left\{\hat{X}^{(\mathrm{pred})}_t : t \in \{0, 1, 2, \cdots, T_e\}\right\}$. The process is presented in Algorithm 1.

```
**Data:** $X_0$, $\hat{\mathbf{X}}$
initialization $t = 0$, $\hat{\mathbf{X}}_{\mathrm{pred}} = \mathbf{0}$;
while $t \leq T_e$ do
    read $X_t, \hat{X}_t$;
    initialize $e_t = X_t - \hat{X}_t$;
    predict $\hat{e}_{t+1}$, $\hat{e}_{t+\mathrm{RTT}}$;
    update $\hat{X}_{t+1} = \hat{X}_{t+1} + \hat{e}_{t+1}$;
    update $\hat{X}_{t+\mathrm{RTT}} = \hat{X}_{t+\mathrm{RTT}} + \hat{e}_{t+\mathrm{RTT}}$;
    save $\hat{X}^{(\mathrm{pred})}_{t+\mathrm{RTT}} = \hat{X}_{t+\mathrm{RTT}}$
end
```

**Algorithm 1:** Channel bit-SNR prediction algorithm.

One example of the predicted channel bit-SNR curve along with the actual bit-SNR curve of a communication pass obtained using Algorithm 1 is shown in Figure 3.11. We can see that our prediction model is capable of tracking the actual channel condition and predicting the rapid fluctuation efficiently.

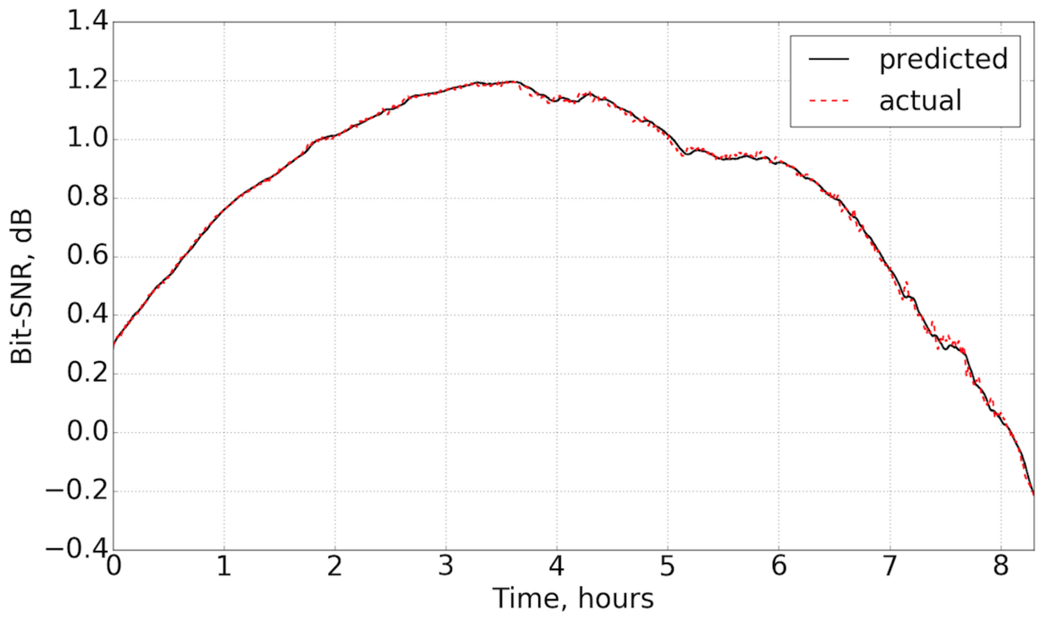


Figure 3.11: Predicted and actual temporal bit-SNR during a communication pass on September 26, 2005.

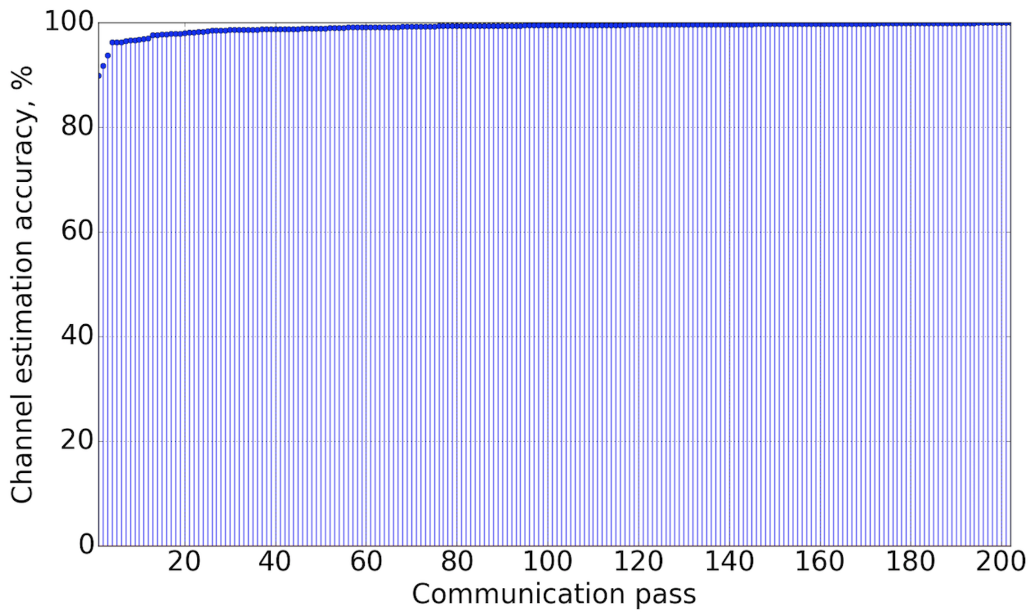


Figure 3.12: Percentage of correct channel bit-SNR estimations obtained with our channel bit-SNR prediction model for passes under consideration when RTT = 20 minutes.

The Table 3.5 demonstrates the % of estimation accuracy in individual ranges

Table 3.5: Channel bit-SNR estimation accuracy (in %) achieved with the AR(1) based prediction model.

(a) Communication pass on February 17, 2011 data.

| % of SNRs | Ranges | Predicted SNR range | | | | |
|---|---|---|---|---|---|---|
| | | Range A | Range B | Range C | Range D | Range E |
| 0.000 | Range A | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range B | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range C | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range D | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 100.000 | Range E | 0.000 | 0.000 | 0.000 | 0.000 | 100.000 |
| % of predicted SNR | | 0.000 | 0.000 | 0.000 | 0.000 | 100.000 |
| Percentage of correct estimation: 100.000 | | | | | | |

(b) Communication pass on December 15, 2001 data.

| % of SNRs | Ranges | Predicted SNR range | | | | |
|---|---|---|---|---|---|---|
| | | Range A | Range B | Range C | Range D | Range E |
| 7.196 | Range A | 98.909 | 1.091 | 0.000 | 0.000 | 0.000 |
| 4.907 | Range B | 1.417 | 97.577 | 1.006 | 0.000 | 0.000 |
| 3.749 | Range C | 0.000 | 0.000 | 100.000 | 0.000 | 0.000 |
| 15.751 | Range D | 0.000 | 0.000 | 0.826 | 98.589 | 0.584 |
| 68.397 | Range E | 0.000 | 0.000 | 0.000 | 0.128 | 99.872 |
| % of predicted SNR | | 7.187 | 4.867 | 3.929 | 15.616 | 68.402 |
| Percentage of correct estimation: 99.493 | | | | | | |

(c) Communication pass on September 5, 2005 data.

| % of SNRs | Ranges | Predicted SNR range | | | | |
|---|---|---|---|---|---|---|
| | | Range A | Range B | Range C | Range D | Range E |
| 5.566 | Range A | 100.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.017 | Range B | 100.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.011 | Range C | 100.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 41.109 | Range D | 2.970 | 0.375 | 0.214 | 82.887 | 13.554 |
| 53.2979 | Range E | 0.144 | 0.000 | 0.000 | 5.686 | 94.169 |
| % of predicted SNR | | 6.892 | 0.154 | 0.088 | 37.105 | 55.762 |
| Percentage of correct estimation: 89.830 | | | | | | |

and overall estimation accuracy during three communication passes. On the communication pass of December 15, 2001 (Table 3.5b), 7.196% of channel bit-SNR is observed in range A, 4.907% in range B, 3.749% in range C, 15.751% in range D, and 68.397% in range E. For the case when channel bit-SNR was actually in range A, 98.909% of time we predicted it to be in range A and 1.091% of time in range B. For the case when channel bit-SNR was actually in range B, 1.417% of time we predicted it in range A, 97.577% of time in range B, and 1.006% of time in range C. Similar predictions could be seen for other ranges as well. Overall, we were able to correctly predict channel bit-SNR 99.493% of time during the communication pass.

Using AWVR data of 202 different communication passes between year 2001 and 2011, we computed the percentage of correct estimation for each communication pass. The channel prediction algorithm is run once for each communication pass when RTT = 20 minutes. For different communication passes, different percentage of correct estimation is observed, with maximum and minimum percentage of correct estimation being 100% and 89.83% respectively. Average percentage of correct estimation obtained for all the passes with RTT = 20 minutes is 99.113% with the variance of 1.4549%. Percentage of correct estimation achieved in each communication pass are arranged in increasing order and plotted against the communication pass when RTT = 20 minutes is shown in Figure 3.12. Among these 202 communication passes, just on 1 pass, our prediction accuracy is less than 90% and for the rest of the 201 passes, we have prediction accuracy more than 90%.

To further evaluate the effectiveness of the prediction algorithm under different round trip time scenarios, we conducted future channel bit-SNR predictions for the same 202 communications passes when RTT = 6.5 minutes as well as RTT = 44 minutes. Percentage of correct estimation achieved in each communication pass are arranged in increasing order and plotted against the communication pass when

RTT = 6.5 minutes and RTT = 44 minutes is shown in Figure 3.13 and Figure 3.14, respectively. As expected, different percentages of correct estimation are observed for different communication passes. When the RTT = 6.5 minutes, maximum and minimum percentage of correct estimation is 100% and 90.182% respectively. Average percentage of correct estimation obtained for all the passes is 99.113% with the variance of 1.455%. When the RTT = 44 minutes, maximum and minimum percentage of correct estimation is 100% and 80.99% respectively. Average percentage of correct estimation obtained for all the passes is 99.064% with the variance of 2.684%. This shows that the prediction algorithm is sturdy enough to perform under various RTTs scenarios and is capable of correctly predicting with very high accuracy.

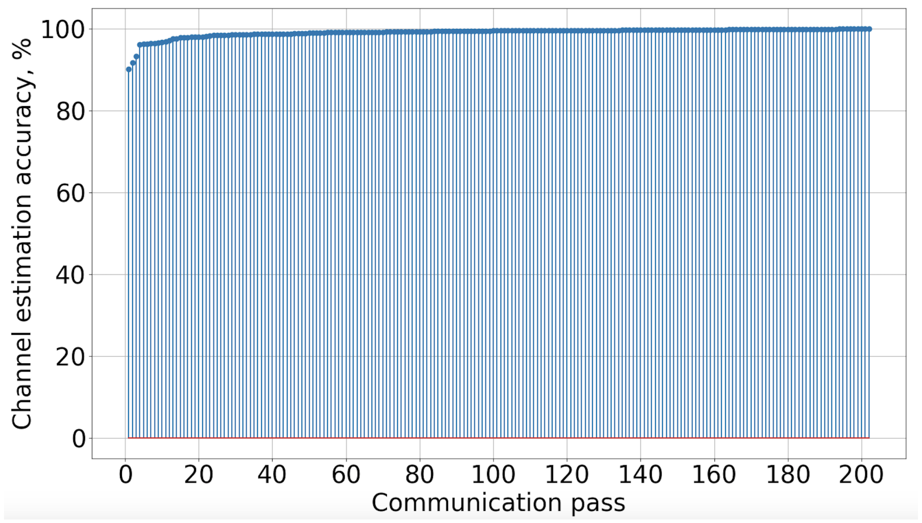


Figure 3.13: Percentage of correct channel bit-SNR estimations obtained with our channel bit-SNR prediction model for passes under consideration with RTT = 6.5 minutes.

## 3.5 Simulation Setup And Results

### 3.5.1 Simulation Setup And Communication Data Rates

In this section, we discuss in brief about the parameters used in our simulation setup, and channel symbol rates and corresponding achievable data rates for MRO

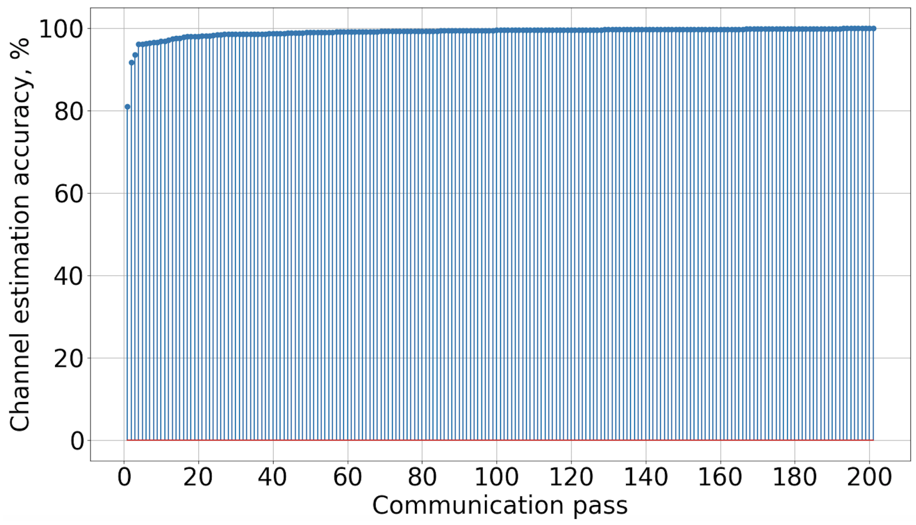


Figure 3.14: Percentage of correct channel bit-SNR estimations obtained with our channel bit-SNR prediction model for passes under consideration with RTT = 44 minutes.

Ka-band communication with the DCSM method. MRO supports both the X-band and Ka-band communications. If X-band and Ka-band carry different data types, the combined channel modulation rate (symbol rate) of the two bands can not exceed 6 Msps. If both bands carry identical data, the modulation rate on each band cannot exceed 6 Msps [5].

Table 3.6: Parameters used in the simulation setup.

| Parameters | Values |
|---|---|
| Transmission power (MRO's HGA) | 34 Watt |
| Transmit antenna gain (MRO's HGA) | 56.4 dBi |
| Receive antenna gain (DSS-25) | 79 dBi |
| Modulation index (MRO's HGA) | 21.09375 |
| Channel modulation rate (MRO's HGA) | 3 Msps |
| Earth-MRO distance | $18 \times 10^{10}$ m |
| File size | 50 MB |
| $\Theta$ (RaptorQ codes) | 5 |

We are considering only Ka-band communication scenario for our simulation purpose and are simulating data transmission between the MRO's high gain antenna (HGA) and 34-m beam-waveguide (BWG) antenna DSS-25 at Goldstone DSCC. In

MRO, BPSK modulation is used for Ka-band transmission, and it has a limited number of channel modulation rates and modulation index values. We are using 3 Msps as our channel modulation rate and hence the transmission rate $R_b$ is 3 Mbps. Distance between the Earth and the MRO is taken to be $18 \times 10^{10}$ m, equivalent to RTT of 20 mins. The parameters used in our simulations are summarized in Table 3.6. A summary of data rates for different channel symbol rates and turbo codes for DCSM, *Genie I* and *Static I* is given in the Table 3.7.

Table 3.7: Data rates, $R_d$, for different channel modulation rates and turbo codes with $K = 8920$ bits combination.

(a) Data rate $R_d$ for DCSM, *genie I* and *static I*.

| Turbo Codes | Data rate $R_d$ | | | |
|---|---|---|---|---|
| | 6 Msps | 4 Msps | 3 Msps | 2 Msps |
| $(8920, 1/6)$ | 992.4 kbps | 661.6 kbps | 496.2 kbps | 330.8 kbps |
| $(8920, 1/4)$ | 1.489 Mbps | 992.4 kbps | 744.3 kbps | 496.2 kbps |
| $(8920, 1/3)$ | 1.985 Mbps | 1.323 Mbps | 992.4 kbps | 661.6 kbps |
| $(8920, 1/2)$ | 2.977 Mbps | 1.985 Mbps | 1.489 Mbps | 992.4 kbps |

(b) Data rate $R_d$ for *genie II* and *static II*.

| Turbo Codes | Data rate $R_d$ | | | |
|---|---|---|---|---|
| | 6 Msps | 4 Msps | 3 Msps | 2 Msps |
| $(8920, 1/6)$ | 995.980 kbps | 663.986 kbps | 497.990 kbps | 331.993 kbps |
| $(8920, 1/4)$ | 1.494 Mbps | 995.980 kbps | 746.985 kbps | 497.990 kbps |
| $(8920, 1/3)$ | 1.992 Mbps | 1.328 Mbps | 995.980 kbps | 663.986 kbps |
| $(8920, 1/2)$ | 2.988 Mbps | 1.992 Mbps | 1.494 Mbps | 995.980 kbps |

The simulation is run once for each communication pass for all the above mentioned methods. Each file is considered of fixed size 50 MB, resulting in $K_{\text{SS}} = 45005$ with $L_{\text{SS}} = 8888$ bits for DCSM, *genie I*, and *static I*. We are using $\Theta = 5$ with the RaptorQ codes. In case of *genie II* and *static II*, each file is divided into 44844 turbo blocks of length $K = 8920$ bits. In the simulation setup, we do not consider any system losses and hence the results are baseline performance achieved with the proposed mechanism.

### 3.5.2 Background On Results

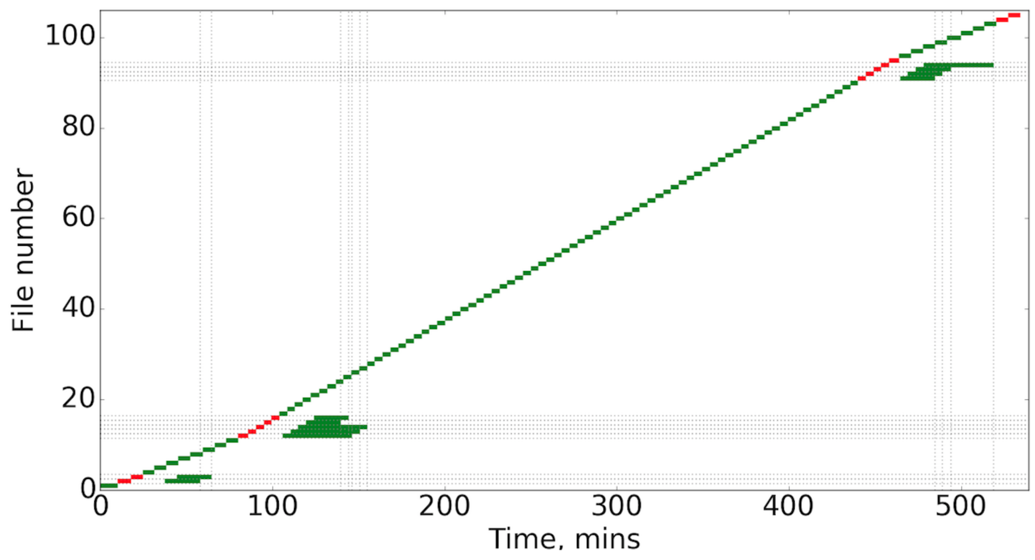


Figure 3.15: An example output showing serial transmissions of files during a communication pass on March 13 2008 using DCSM adaptive $y$ with average FER approach. The color green indicates successful decoding of the file and the red indicates that the number of clean packets received were not enough to decode the file and additional transmissions are needed.

A pictorial presentation of file transmissions during a communication pass is presented in Figure 3.15. Start and end of green/red line denotes the beginning and end of symbol transmission from a particular file, respectively, and length of the line denotes the period during which symbols of the particular file were transmitted. The horizontal red lines indicate that the number of clean packets received were not enough to decode the file and additional transmissions were needed. The horizontal green lines indicate successful decoding of a file and its length equals the total time required for the file transmission. It can be observed from the figure that the first file was successfully recovered in the first round of transmission. For the 2nd file,

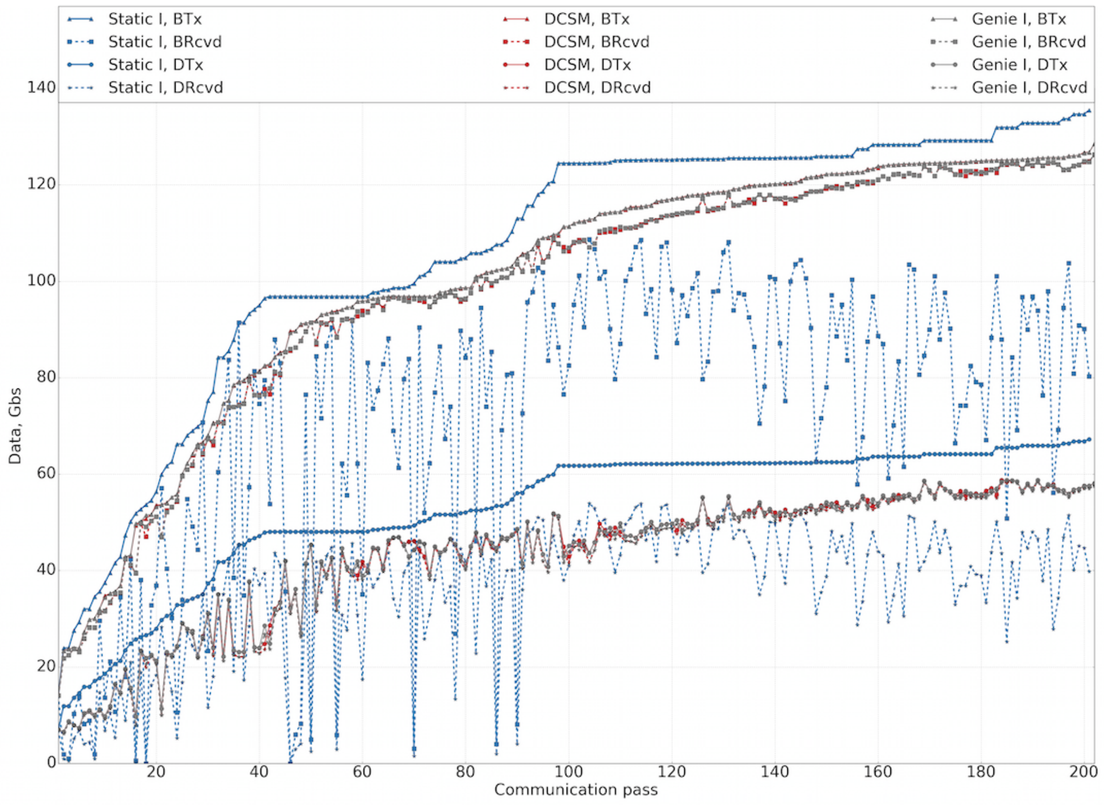


Figure 3.16: Bits transmitted ($B_{\text{Tx}}$), bits received ($B_{\text{Rcvd}}$), data transmitted ($D_{\text{Tx}}$), and data received ($D_{\text{Rcvd}}$) during all the communication passes with adaptive best approach.

symbols received during the first round were insufficient and additional transmissions were requested. Additional transmission began where its corresponding green line began, and took some RTTs to receive enough symbols to successfully decode the file. It is also worth mentioning that each retransmission request is not for requesting any specific symbols. It only tells the transmitter how many more symbols should be transmitted. The result is that files 4, 5, 6, and 7 are completed before file 2 and files 4, 5, 6, 7, and 8 were completed before file 3. It is essential to note that the uplink is used for real-time turbo encoder selection commands as well as for feedbacks requesting additional number of symbols from previously transmitted files. The feedback requests can be piggybacked on real-time turbo encoder selection commands.

For each communication pass, the total bits transmitted ($B_{\text{Tx}}$) and the total data transmitted ($D_{\text{Tx}}$) by the MRO (in Gb), and the total bits received ($B_{\text{Rcvd}}$) and the total data received ($D_{\text{Rcvd}}$) by DSS-25 (in Gb) over the duration of the pass is recorded. The $D_{\text{Tx}}$ is a measure of actual content of the file transmitted, excluding all the headers and trailers. The total $D_{\text{Tx}}$ and $D_{\text{Rcvd}}$ over the duration of a communication pass depends on a number of factors, namely, duration of the communication pass, antenna elevation angle, channel bit-SNRs, weather dropout conditions, and the accuracy with which channel bit-SNRs are predicted.

Plots showing $B_{\text{Tx}}$, $D_{\text{Tx}}$, $B_{\text{Rcvd}}$, and $D_{\text{Rcvd}}$ during each communication pass with adaptive $y$ approach using best FER is shown in Figure 3.16. For the clarity of presentation, the communication passes are arranged in increasing order in terms of the $D_{\text{Rcvd}}$ by the DSS-25. As an example, on the 120th communication pass, $B_{\text{Tx}}$ = 117.08 Gb and $D_{\text{Tx}}$ = 49.76 Gb, and $B_{\text{Rcvd}}$ = 113.67 Gb and $D_{\text{Rcvd}}$ = 48.87 Gb, over entire duration of the communication pass using the DCSM with adaptive best DCSM.

### 3.5.3 Intra-Method Performance Evaluation

By averaging over all 202 communication passes for each method, a summary of $B_{\text{Tx}}$, $D_{\text{Tx}}$, $B_{\text{Rcvd}}$, and $D_{\text{Rcvd}}$ during each communication pass is presented in Tables 3.8, 3.9, and 3.10. The $F_{\text{TxEqv}}$ and $F_{\text{RcvdEqv}}$ represent the equivalent number of files transmitted by the MRO and received by the DSS-25, respectively, obtained by dividing $D_{\text{Tx}}$ and $D_{\text{Rcvd}}$ by file size (50 MB). The $F_{\text{Tx}}$ is the actual number of files transmitted and $F_{\text{RcvdSuccess}}$ is the number of successfully received files over the duration of a communication pass.

From Table 3.8, it can be observed that with the *static II* method, on average, $B_{\text{Tx}} = 106.181$ Gb, $D_{\text{Tx}} = 52.877$ Gb, and $D_{\text{Rcvd}} = 35.302$ Gb. The $D_{\text{Tx}}$ and $D_{\text{Rcvd}}$ are equivalent to $F_{\text{TxEqv}} = 132.192$ and $F_{\text{RcvdSuccess}} = 88.256$, respectively. However, the actual number of files transmitted and successfully received are $F_{\text{Tx}} = 132.19$ and $F_{\text{RcvdSuccess}} = 53.705$, respectively.

With the *static I* method, on average, the $B_{\text{Tx}}$, $D_{\text{Tx}}$, $B_{\text{Rcvd}}$, and $D_{\text{Rcvd}}$ obtained with all the sub-classes are almost the same. However, there is huge difference in performance in terms of $F_{\text{Tx}}$ and $F_{\text{RcvdSuccess}}$. Looking at the $F_{\text{RcvdSuccess}}$ values, we can observe that the fixed *static I* approach results in larger number of $F_{\text{RcvdSuccess}}$ than that obtained with the adaptive *static I* approach. Since our primary objective is to deliver maximum number of files successfully over the same duration of a communication pass, we can conclude that the fixed *static I* is superior to the adaptive *static I*.

As seen in the Table 3.8, the fixed *static I* has three different sub-classes given as

a) $y = y$: $y$ additional symbols with each file,

b) $y = 2y$: $2y$ additional symbols with each file, and

c) $y = y_{\text{min}}$: Minimum value $y_{\text{min}} = 453$ is obtained for a pass when bit-SNR of the

Table 3.8: Amount of data transferred (in Gb) and number of file transmissions on average during a communication pass with *static* approach.

| Data | *Static I* | | | | | | *Static II* |
|---|---|---|---|---|---|---|---|
| | Fixed $y$ | | | Adaptive $y$ | | | |
| | $y$ | $2y$ | $y_{\min}$ | Best FER | Average FER | Worst FER | |
| $B_{\text{Tx}}$ | 106.769 | 106.769 | 106.769 | 106.769 | 106.769 | 106.769 | 106.181 |
| $D_{\text{Tx}}$ | 52.979 | 52.979 | 52.979 | 52.979 | 52.979 | 52.979 | 52.877 |
| $B_{\text{Rcvd}}$ | 71.625 | 71.625 | 71.625 | 71.625 | 71.625 | 71.625 | 70.890 |
| $D_{\text{Rcvd}}$ | 35.541 | 35.541 | 35.541 | 35.541 | 35.541 | 35.541 | 35.302 |
| $D_{\text{Tx}}/B_{\text{Tx}}$ | 0.4962 | 0.4962 | 0.4962 | 0.4962 | 0.4962 | 0.4962 | 0.4980 |
| $D_{\text{Rcvd}}/B_{\text{Tx}}$ | 0.3329 | 0.3329 | 0.3329 | 0.3329 | 0.3329 | 0.3329 | 0.3325 |
| $D_{\text{Rcvd}}/D_{\text{Tx}}$ | 0.671 | 0.671 | 0.671 | 0.671 | 0.671 | 0.671 | 0.6676 |
| $D_{\text{Rcvd}}/B_{\text{Rcvd}}$ | 0.4962 | 0.4962 | 0.4962 | 0.4962 | 0.4962 | 0.4962 | 0.4980 |
| $F_{\text{TxEqv}}$ | 132.449 | 132.449 | 132.449 | 132.449 | 132.449 | 132.449 | 132.192 |
| $F_{\text{RcvdEqv}}$ | 88.852 | 88.852 | 88.852 | 88.852 | 88.852 | 88.852 | 88.256 |
| $F_{\text{Tx}}$ | 90.355 | 87.338 | 94.378 | 98.736 | 93.240 | 72.589 | 132.190 |
| $F_{\text{RcvdSuccess}}$ | 83.194 | 80.660 | 85.369 | 66.490 | 79.629 | 66.193 | 53.705 |

Table 3.9: Amount of data transferred (in Gb) and number of file transmissions on average during a communication pass with DCSM approach.

| Data | DCSM | | | | | |
|---|---|---|---|---|---|---|
| | Fixed $y$ | | | Adaptive $y$ | | |
| | $y$ | $2y$ | $y_{\min}$ | Best FER | Average FER | Worst FER |
| $B_{\text{Tx}}$ | 100.569 | 100.569 | 100.569 | 100.569 | 100.569 | 100.569 |
| $D_{\text{Tx}}$ | 43.377 | 43.377 | 43.377 | 43.377 | 43.377 | 43.377 |
| $B_{\text{Rcvd}}$ | 98.132 | 98.132 | 98.131 | 98.133 | 98.133 | 98.132 |
| $D_{\text{Rcvd}}$ | 42.696 | 42.696 | 42.696 | 42.697 | 42.696 | 42.696 |
| $D_{\text{Tx}}/B_{\text{Tx}}$ | 0.4313 | 0.4313 | 0.4313 | 0.4313 | 0.4313 | 0.4313 |
| $D_{\text{Rcvd}}/B_{\text{Tx}}$ | 0.4245 | 0.4245 | 0.4245 | 0.4245 | 0.4245 | 0.4245 |
| $D_{\text{Rcvd}}/D_{\text{Tx}}$ | 0.9843 | 0.9843 | 0.9843 | 0.9843 | 0.9843 | 0.9843 |
| $D_{\text{Rcvd}}/B_{\text{Rcvd}}$ | 0.4351 | 0.4351 | 0.4351 | 0.4351 | 0.4351 | 0.4351 |
| $F_{\text{TxEqv}}$ | 108.443 | 108.443 | 108.443 | 108.443 | 108.443 | 108.443 |
| $F_{\text{RcvdEqv}}$ | 106.741 | 106.741 | 106.740 | 106.741 | 106.741 | 106.741 |
| $F_{\text{Tx}}$ | 102.861 | 98.809 | 106.129 | 108.038 | 105.849 | 81.349 |
| $F_{\text{RcvdSuccess}}$ | 100.505 | 97.467 | 102.628 | 76.840 | 102.301 | 81.205 |

Table 3.10: Amount of data transferred (in Gb) and number of file transmissions on average during a communication pass with *genie* approach.

| Data | *Genie I* | | | | | | *Genie II* |
|---|---|---|---|---|---|---|---|
| | Fixed $y$ | | | Adaptive $y$ | | | |
| | $y$ | $2y$ | $y_{\min}$ | Best FER | Average FER | Worst FER | |
| $B_{\text{Tx}}$ | 100.565 | 100.565 | 100.565 | 100.565 | 100.565 | 100.565 | 100.565 |
| $D_{\text{Tx}}$ | 43.377 | 43.377 | 43.377 | 43.377 | 43.377 | 43.377 | 43.533 |
| $B_{\text{Rcvd}}$ | 98.188 | 98.187 | 98.187 | 98.187 | 98.188 | 98.188 | 98.187 |
| $D_{\text{Rcvd}}$ | 42.720 | 42.720 | 42.720 | 42.720 | 42.720 | 42.720 | 42.873 |
| $D_{\text{Tx}}/B_{\text{Tx}}$ | 0.4313 | 0.4313 | 0.4313 | 0.4313 | 0.4313 | 0.4313 | 0.4329 |
| $D_{\text{Rcvd}}/B_{\text{Tx}}$ | 0.4248 | 0.4248 | 0.4248 | 0.4248 | 0.4248 | 0.4248 | 0.4263 |
| $D_{\text{Rcvd}}/D_{\text{Tx}}$ | 0.9848 | 0.9848 | 0.9848 | 0.9848 | 0.9848 | 0.9848 | 0.9848 |
| $D_{\text{Rcvd}}/B_{\text{Rcvd}}$ | 0.4351 | 0.4351 | 0.4351 | 0.4351 | 0.4351 | 0.4351 | 0.4366 |
| $F_{\text{TxEqv}}$ | 108.443 | 108.443 | 108.443 | 108.443 | 108.443 | 108.443 | 108.833 |
| $F_{\text{RcvdEqv}}$ | 106.799 | 106.799 | 106.799 | 106.799 | 106.800 | 106.799 | 107.183 |
| $F_{\text{Tx}}$ | 102.963 | 98.950 | 106.176 | 108.023 | 105.886 | 81.367 | 108.446 |
| $F_{\text{RcvdSuccess}}$ | 100.634 | 97.608 | 102.689 | 77.307 | 102.314 | 81.245 | 62.673 |

entire pass duration is in Range E, and is predicted with 100% accuracy.

Among the sub-classes of fixed *static I*, fixed *static I* with $y_{\min}$ results in larger number of $F_{\text{RcvdSuccess}}$. Hence, we can conclude that fixed *static I* with $y_{\min}$ is superior to other *static I* subclasses, which in turn is superior to *static II* approach.

Similar to that with the *static* approach, fixed DCSM with $y_{\min}$ is superior to other DCSM sub-classes (Table 3.9), and fixed *genie I* with $y_{\min}$ is superior to other *genie* sub-classes (Table 3.10). For all the three methods, we observed that fixed approach with $y_{\min}$ is the best option to be considered (within each method) to achieve highest number of successful file deliveries. Complexity of implementation of fixed approach with $y_{\min}$ is drastically low (in comparison to the other sub-classes), as we can always use the same $y = 453$ for all the files of all the communication passes. For each file, we are adding 1.00644% of additional symbols, i.e., 1.00644%

of 45010 is 453, by transmitting $y = 453$ additional symbols, which is very negligible given the performance improvement that is achieved.

### 3.5.4 Inter-Method Performance Evaluation

For fixed *static I* with $y_{\min}$, on average, in order to transmit 1 bit of information, MRO needs to transmit 2.015 bits. The $D_{\text{Rcvd}}$ by DSS-25 is about 33.29% and 67.1% of $B_{\text{Tx}}$ and $D_{\text{Tx}}$, respectively. On average, in order to transmit 1 bit of information, MRO needs to transmit 2.318 bits each for both the fixed DCSM and fixed *genie I* with $y_{\min}$. With the fixed DCSM with $y_{\min}$, the $D_{\text{Rcvd}}$ by DSS-25 is about 42.45% and 98.43% of $B_{\text{Tx}}$ and $D_{\text{Tx}}$, respectively, and with the fixed *genie I* with $y_{\min}$, the $D_{\text{Rcvd}}$ by DSS-25 is about 42.48% and 98.48% of $B_{\text{Tx}}$ and $D_{\text{Tx}}$, respectively.

Over the duration a communication pass, on average, $D_{\text{Rcvd}}$ with *genie I* is greater than that with DCSM, and in turn with *static I* with fixed $y_{\min}$. On average, the number of file successfully received with fixed *static I*, DCSM, and *genie I* with $y_{\min}$ represented by $F^{(\text{sI})}_{\text{RcvdSuccess}}$, $F^{(\text{d})}_{\text{RcvdSuccess}}$ and $F^{(\text{gI})}_{\text{RcvdSuccess}}$, respectively, is 85.369, 102.628, and 102.689. Compared to the $F^{(\text{sI})}_{\text{RcvdSuccess}}$, 17.259 and 17.32 additional files are received successfully with the DCSM and *genie I* with fixed $y_{\min}$. That is, over the same duration of a communication pass, on average, 20.217 % and 20.288 % of additional files are successfully received at the Earth station respectively with the DCSM and *genie I* with fixed $y_{\min}$. The $F^{(\text{d})}_{\text{RcvdSuccess}}$ is about 99.94% of that received with $F^{(\text{gI})}_{\text{RcvdSuccess}}$ with fixed $y_{\min}$. This implies that the overall channel throughput expressed in bits/s as well as successful files/s, is highest with the *genie*, lowest with the *static* and DCSM closely following the *genie*.

Another important measure of performance is the total time required to successfully deliver a file. A plot showing the number of files successfully delivered by

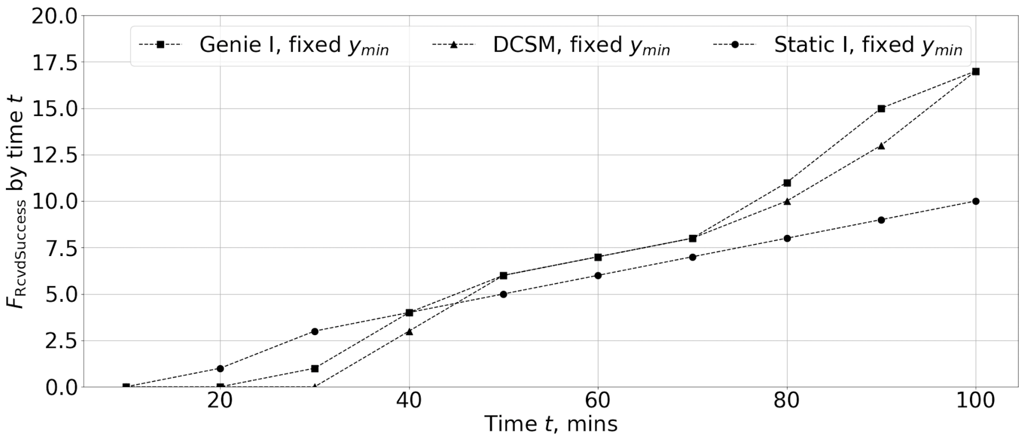
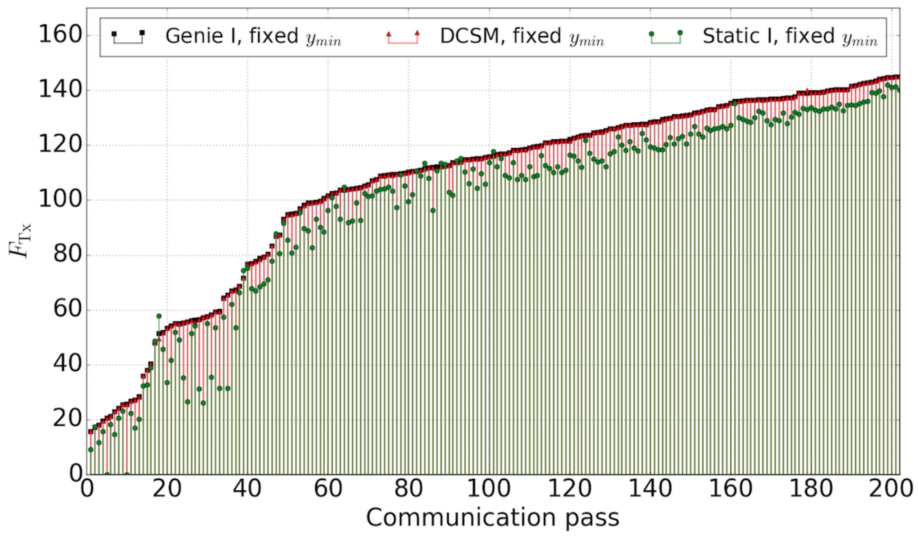


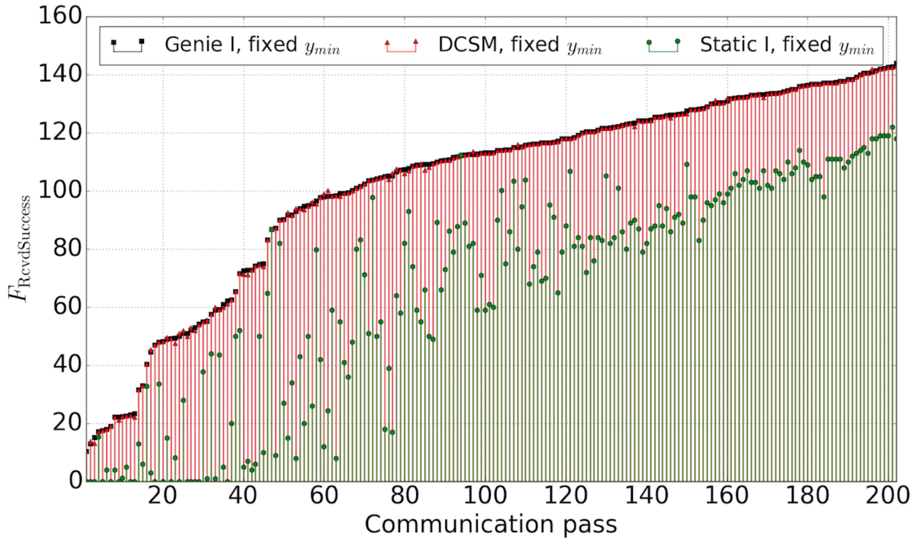
Figure 3.17: A plot showing the number of files successfully received at Earth station by time $t = 100$ minutes on a communication pass of December 9, 2001.

time $t = 100$ minutes during a communication pass is presented in Figure 3.17. Except for some cases where delivery of a file fails during its first round of transmission and needs additional round of transmissions, time required to successfully deliver the file with *genie I* is comparable to that of the DCSM with fixed $y_{\text{min}}$. The number of files successfully delivered by some time $t$ rapidly increases as we proceed through the communication due to the adaptive data transmission scheme of *genie I* and DCSM. Variable time is required to deliver each file depending on the turbo code rate used throughout the communication pass. However, due to the fixed data transmission rate scheme of *static* method, time required for *static I* with fixed $y_{\text{min}}$ is almost constant throughout the duration of a communication pass, except for when transmission of additional number of symbols of the file occurs. Thus, the *genie* and the DCSM approaches are better to the *static* approach in terms of the amount of time required to successfully deliver a file as well.

Figure 3.18a and 3.18b represent the number of files (arranged in increasing order with reference to the performance of the *genie* method) successfully transmitted by the MRO and received by the DSS-25, respectively, using *static I*, DCSM and *genie I* with fixed $y_{\text{min}}$. In Figure 3.18a, on 100th communication pass 117.772, 116.056,



(a) Successfully transmitted from MRO.



(b) Successfully received by DSS-25.

Figure 3.18: Total number of files of size 50 MBs successfully transmitted from MRO and successfully received by DSS-25.

and 116.063 files were transmitted from MRO with the *static I*, DCSM, and *genie I* with fixed $y_{\min}$, respectively. At the receiver end, only 61.0, 113.075, and 113.111 files were received successfully by the DSS-25 as shown in Figure 3.18b.

In general, performance of DCSM is very close to that of *genie* in terms of overall channel throughput, total number of files successfully delivered to Earth station receiver, and time required to successfully deliver each file. From all the results presented here, we can clearly see that use of *static* method results in huge loss of transmission power, channel bandwidth and other associated costly resources that could be utilized efficiently if we use channel condition prediction mechanisms, vary the turbo codes dynamically and periodically use reverse channel to send feedbacks on lost packets to distant spacecraft. We can see that at the cost of some additional bandwidth, we can achieve very high performance gain as the amount of data loss is drastically reduced due to the use of real time channel prediction mechanism, which can be done efficiently with the simple DCSM protocol we proposed in this work.

## 3.6 Conclusion

In this chapter, we presented a new content delivery protocol for deep space communications that incorporates RaptorQ codes, turbo codes, and a practical channel prediction model. We have shown that this protocol can cope with the issues of large RTT and the dynamic noise environment of space communications. We have also given an upper bound on the performance that can be achieved by incorporating real time channel condition prediction and a dynamic code rate selection strategy.

# CHAPTER 4

# LOGISTIC REGRESSION BASED PREDICTION MODEL

In the first part of this chapter, we present a brief introduction to logistic regression-based data analysis approach and classification problems. In the second part, we present details on using logistic regression to design our channel condition prediction model.

## 4.1 Introduction to Logistic Regression

Logistic regression [36] is one of the most popular and most widely used supervised learning algorithm used for classification problems. Supervised learning is a process of an algorithm learning from the labeled training dataset. That is, we are given a training data set, containing input feature values and corresponding output, and the objective is to infer a functional relation between them. We fit our model on historical data (past measurements) and develop a model, and use the model to predict the future values. Supervised learning problems are categorized into *regression* and *classification* problems. In a regression problem, the objective is to find a function to predict continuous numerical values, meaning that we try to map input variables into output values using some continuous function. For example, given a picture of a person, we have to predict their age on the basis of the given picture and the function inferred from past data. Classification is the problem of identifying to which set of categories a new observation belongs. That is, we try to map input variables into discrete categories and predict results in a discrete output. For example, given an email, we have to predict whether the email is spam or not.

In statistics, logistic regression is a classification algorithm the objective of which is to predict the output variable $y$ that we can think of as taking on two values either zero or one, based on the value of input variables denoted by $x$. The output $y = 0$ is termed as class zero or negative class, and the output $y = 1$ is called the class one or positive class. The definition and assignment of the two classes to zero and one is arbitrary and it doesn't really matter. However, the general standard is that a negative class conveys the absence and the positive class conveys the presence of what we are looking for. The classification problem with just two classes is called binary classification problem. The classification problem where $y$ may take more than two values is called a multiclass classification problem. We will be developing our prediction model as a binary classification problem, details of which are presented later.

### 4.1.1 Model Representation

We will use $x^{(i)}$ to denote the input variables (features) of the $i$th training example, and $y^{(i)}$ to denote the output or target variable of the $i$th training example that we are trying to predict. A pair $(x^{(i)}, y^{(i)})$ is called a training example, and the dataset, i.e., a list of $m$ training examples $(x^{(i)}, y^{(i)})$; $i = 1, \cdots, m$, that we will be using to learn is called a training set represented as $(x, y)$. We will use $x_j$ to denote feature $j$, $j = 1, \cdots, n$, and $x_j^{(i)}$ to denote the value of feature $j$ in the $i$th training example, and $n$ is the total number of features. The superscript $(i)$ in the notation is simply an index into the training set. We will also use $X$ to denote the space of input values, and $Y$ to denote the space of output values. For the ease of understanding, an illustration of input features, output variable and training example is presented in Table 4.1.

Our goal is, given a training set and input features, to learn a function $h_\theta$ :

Table 4.1: An illustration of training dataset containing $n$ input features and $m$ training examples.

| | Input features | | | | Output |
|---|---|---|---|---|---|
| $i$ | $x_1^{(i)}$ | $x_2^{(i)}$ | $\cdots$ | $x_n^{(i)}$ | $y^{(i)}$ |
| 1 | $x_1^{(1)}$ | $x_2^{(1)}$ | $\cdots$ | $x_n^{(1)}$ | $y^{(1)}$ |
| 2 | $x_1^{(2)}$ | $x_2^{(2)}$ | $\cdots$ | $x_n^{(2)}$ | $y^{(2)}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\ddots$ | $\vdots$ | $\vdots$ |
| $m$ | $x_1^{(m)}$ | $x_2^{(m)}$ | $\cdots$ | $x_n^{(m)}$ | $y^{(m)}$ |

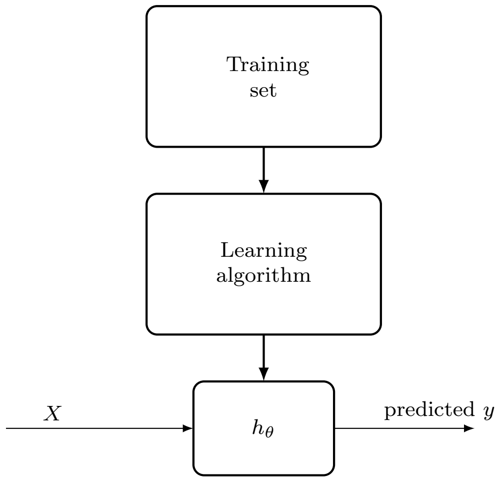


Figure 4.1: Supervised learning process.

$X \to Y$ so that $h_\theta(x)$ is a good predictor for the corresponding value of $y$. The pictorial representation of the process is represented as in Figure 4.1. For historical reasons, this function $h_\theta$ is called a hypothesis.

For instance, if we are trying to build a spam classifier for email, then $x^{(i)}$ may be some features of a piece of email, and $y \in \{0, 1\}$ may be 1 if it is a piece of spam mail, and 0 otherwise. Given $x^{(i)}$, the corresponding $y^{(i)}$ is also called the label for the training example.

Logistic regression has the property that the output, i.e., the predictions, of logistic regression are always between zero and one. The logistic regression hypotheses denoted by $h_\theta(x)$ also satisfies $0 \leq h_\theta(x) \leq 1$, and is given as

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n), \tag{4.1}$$

with

$$g(z) = \frac{1}{1 + e^{-z}}, \tag{4.2}$$

where $x_1, x_2, \cdots, x_n$ are the $n$ input features, and $\theta_0, \theta_1, \cdots, \theta_n$ are $(n+1)$ real-valued parameters of the hypothesis that characterize the contribution of each input features. The $g(z)$ is a *sigmoid function*, also called the *logistic function*, and is given as shown in Figure 4.2.

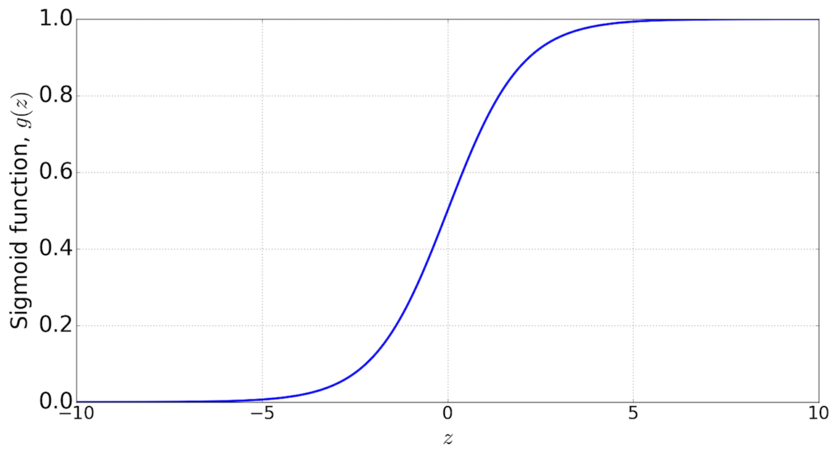


Figure 4.2: Sigmoid function.

Using vectorization, the $h_\theta(x)$ in (4.1) can be concisely represented as

$$h_\theta(x) = g(\theta^T x), \tag{4.3}$$

with the hypothesis parameter vector $\theta^T$ given as

$$\theta^T = \begin{bmatrix} \theta_0 & \theta_1 & \ldots & \theta_n \end{bmatrix},$$

and the input features vector $x$ given as

$$x = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}.$$

The sigmoid function $g(z)$, maps any real number to the $(0, 1)$ interval, making it useful for transforming an arbitrary-valued function into a function better suited for classification. The value of $h_\theta(x)$ is thus the probability that the output $y$ is 1, given $x$ and parameterized by $\theta$. That is,

$$h_\theta(x) = P\{y = 1|x; \theta\} = 1 - P\{y = 0|x; \theta\}. \tag{4.4}$$

As an example, in case of the spam classifier for email, for given $x$, $h_\theta(x) = 0.8$ signifies that there is 80% chance that the email is a spam.

### 4.1.2 Decision Boundary

Decision boundary is the line that separates the area where $y = 0$ and where $y = 1$. It is created by the hypothesis function. The output of the hypothesis function is translated as follows to get discrete 0 or 1 classification

$$h_\theta(x) \geq 0.5 \rightarrow y = 1,$$
$$h_\theta(x) < 0.5 \rightarrow y = 0.$$

The statements signify that

$$h_\theta(x) = g(\theta^T x) \geq 0.5, \quad \text{for } \theta^T x \geq 0,$$
$$h_\theta(x) = g(\theta^T x) < 0.5, \quad \text{for } \theta^T x < 0.$$

Hence, the decision boundaries are

$$\theta^T x \geq 0 \Rightarrow y = 1,$$
$$\theta^T x < 0 \Rightarrow y = 0.$$

Although we use the training data set to fit and determine the parameters, decision boundary is a property of the hypothesis and of the parameters of the hypothesis, but not a property of the data set. That is, once we have particular values for the parameters $\theta_0, \theta_1, \cdots, \theta_n$ then the decision boundary is completely defined by those parameters. We don't need to plot a training set in order to plot the decision boundary.

## 4.2 Channel Condition Prediction Model

In this section, we present the details of how we modeled our future channel condition prediction problem as a classification problem and accuracy of the prediction results obtained.

In Table 3.3 of Chapter 3, we presented the dynamic code assignments for achieving highest link throughput. For the ease of discussion, the table is again presented below as Table 4.2.

In our AR(1) based channel condition prediction model, using present time sky brightness temperature information, along with the antenna elevation angle and Earth-Mars distance one round trip time into the future, we made a prediction of

Table 4.2: Dynamic turbo code assignment for distinct channel bit-SNR ranges and their names.

| Name | Channel bit-SNR (dB) range | turbo code |
|---|---|---|
| Range A | $< -0.5$ | - |
| Range B | $[-0.5, -0.1)$ | $(8920, 1/6)$ |
| Range C | $[-0.1, 0.15)$ | $(8920, 1/4)$ |
| Range D | $[0.15, 0.85)$ | $(8920, 1/3)$ |
| Range E | $[0.85, 2.2)$ | $(8920, 1/2)$ |

channel bit-SNR value one RTT into the future. From the Table 3.3, we see that as long as we predict channel bit-SNR ranges accurately, we do not need to find the exact value of channel bit-SNR. Thus, we first mapped each bit-SNR estimation to its corresponding range and computed the range estimation accuracy of the AR(1) based channel condition prediction model.

In logistic regression based channel condition prediction model, instead of predicting channel bit-SNR value, we directly predict the bit-SNR range one RTT (20 mins) into the future. Thus, we won't have knowledge of future bit-SNR value exactly, however, we will know the possible range to which bit-SNR corresponds.

### 4.2.1 Problem Formulation

We have five different bit-SNR ranges implying five separate output classes, i.e., multiple-class classification problem with five classes, represented as $y \in \{0, 1, 2, 3, 4\}$. We use multiclass classification one-vs-all approach and divide our problem into five binary classification problems; in each one, we predict the probability that $y$ is a member of one of our classes and obtain hypothesis for each class separately. That is, we modify those bit-SNR ranges, convert the problem into five binary classification problem, and obtain hypothesis $h_\theta(x)$ for each range separately.

Let us denote each bit-SNR range by $I$, where $I \in \{A, B, C, D, E\}$. To proceed with the problem for a particular range $I$, we first combine all other ranges together to form a new range denoted as range $I'$ so that the range $I$ becomes a positive class and the new range $I'$ becomes a negative class. We are basically choosing one class and then lumping all the others into a single second class. Using training data set, we then generate hypothesis $h_\theta^{(I)}(x)$ for that particular range. Thus, we convert the multiple-class classification problem into five separate binary classification problems and obtain five separate hypotheses that define boundary of the corresponding range.

As an example, to obtain hypothesis $h_\theta^{(A)}(x)$ for range $I = A$, we combine all the other ranges, i.e., $I \neq A$, and create a new range $A'$, as follows:

| Name | Channel bit-SNR (dB) range | Output $y$ | Class |
|---|---|---|---|
| Range $A$ | < -0.5 | 1 | Positive |
| Range $A'$ | $[-0.5, 2.2)$ | 0 | Negative |

Table 4.3: A summary of new ranges re-defined for the logistic regression problem formulation.

| Hypothesis | Name | bit-SNR (dB) range | Output $y$ | Class |
|---|---|---|---|---|
| $h_\theta^{(A)}(x)$ | Range $A$ | $< -0.5$ | 1 | Positive |
| | Range $A'$ | $[-0.5, 2.2)$ | 0 | Negative |
| $h_\theta^{(B)}(x)$ | Range $B$ | $[-0.5, -0.1)$ | 1 | Positive |
| | Range $B'$ | $< -0.5$ and $[-0.1, 2.2)$ | 0 | Negative |
| $h_\theta^{(C)}(x)$ | Range $C$ | $[-0.1, 0.15)$ | 1 | Positive |
| | Range $C'$ | $< -0.1$ and $[0.15, 2.2)$ | 0 | Negative |
| $h_\theta^{(D)}(x)$ | Range $D$ | $[0.15, 0.85)$ | 1 | Positive |
| | Range $D'$ | $< 0.15$ and $[0.85, 2.2)$ | 0 | Negative |
| $h_\theta^{(E)}(x)$ | Range $E$ | $[0.85, 2.2)$ | 1 | Positive |
| | Range $E'$ | $< 0.85$ | 0 | Negative |

The output $y$ takes 1 (positive class) if the bit-SNR value one RTT into the future belongs to range $A$, and takes 0 (negative class) if it belongs to the new range $A'$. Now, using training data set, we obtain hypothesis $h_\theta^{(A)}(x)$ that separates the range $A$ from range $A'$, i.e., rest of the other ranges. Similarly, we proceed individually with other ranges as well, i.e, $B, C, D, E$ and obtain hypotheses $h_\theta^{(B)}(x), h_\theta^{(C)}(x), h_\theta^{(D)}(x)$, and $h_\theta^{(E)}(x)$, respectively. A summary of all the new ranges that we define during the process is presented in the Table 4.3.

### 4.2.2 Input Features

At a given frequency of operation and for a given Earth-Mars distance, we begin with zenith wet-path delay $D_{\text{wet}}$ and antenna elevation angle $\theta$ data to compute channel bit-SNR value (Section 3.1). Along with zenith wet path delay measurements and antenna elevation, we also have data on time of measurement and antenna azimuth. Thus, for our logistic regression problem, primary input features are a) zenith wet path delay, b) antenna elevation, and c) antenna azimuth. As a part of input feature evaluation, we plotted zenith wet-path delay vs bit-SNR curve, antenna elevation angle vs bit-SNR curve, and antenna azimuth vs bit-SNR curve for each of the 202 communication passes in a single plot as shown in Figure 4.3, Figure 4.4, and Figure 4.5, respectively. Just by looking at the curves, we cannot observe a clear one-to-one mapping between these features and bit-SNR value, however we observed that through logistic regression analysis.

To determine the input features to use and to evaluate the effect of each of them individually, we first divided all the available data into a training data set and a test data set. Out of the 202 communication passes, we selected 101 communication passes as our training data set and another 101 communication passes as our test data set. Using the training set, we then conducted logistic regression analysis and obtained hypotheses functions $h_\theta^{(A)} : X \to Y$, $h_\theta^{(B)} : X \to Y$, $h_\theta^{(C)} : X \to Y$,

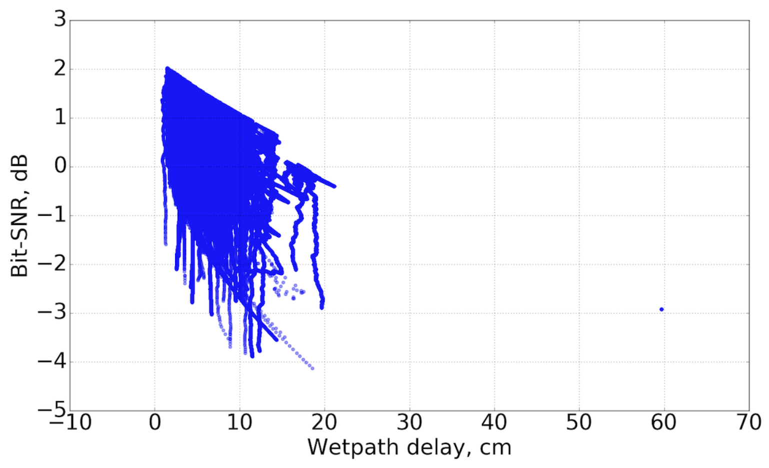


Figure 4.3: Zenith wet-path delay vs bit-SNR curve of all the 202 communication passes.

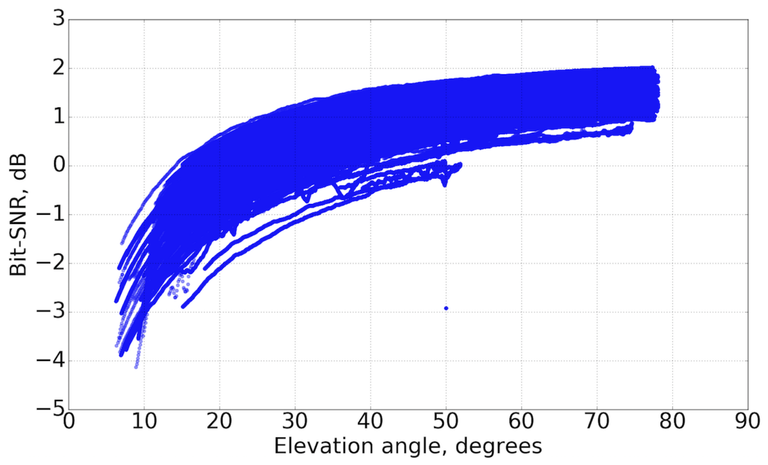


Figure 4.4: Antenna elevation angle vs bit-SNR curve of all the 202 communication passes.

$h_\theta^{(D)} : X \to Y$, and $h_\theta^{(E)} : X \to Y$ each with the following different input features

1. Two input feature case: $n = 2$ with

   - $x_1$ = antenna elevation angle and $x_2$ = antenna azimuth
   - $x_1$ = wet-path delay and $x_2$ = antenna azimuth
   - $x_1$ = antenna elevation angle and $x_2$ = wet-path delay

2. Three input feature case: $n = 3$ with

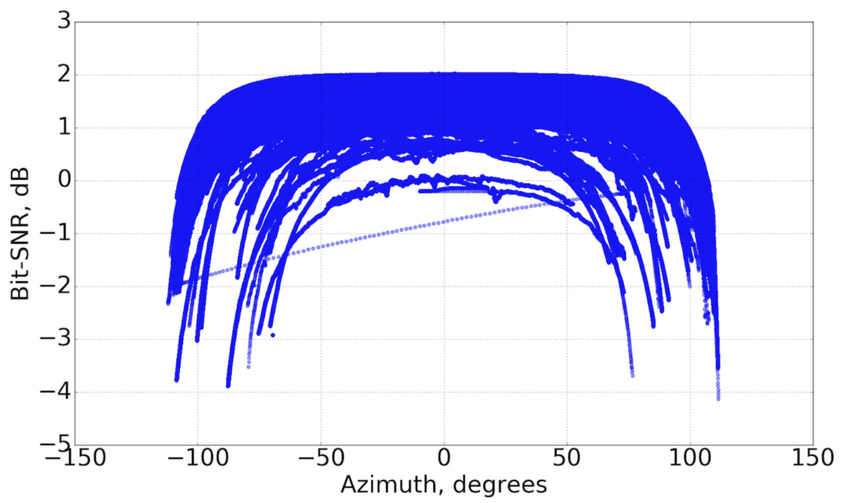


Figure 4.5: Antenna azimuth vs bit-SNR curve of all the 202 communication passes.

- $x_1$ = antenna elevation angle, $x_2$ = wet-path delay, and $x_3$ = antenna azimuth

3. Multiple input feature case: Secondary input features can be obtained by combining multiple primary features into one. As an example, we can combine two primary features $x_1$ and $x_2$ to form a new feature $x_3 = x_1 \cdot x_2$. We can use feature $x_1$ to form a new secondary feature $x_4 = x_1^2$, and so on.

### 4.2.3 Hypothesis Function

To fit our data, we can use different hypothesis function and choose the one that gives better fit as our hypothesis. We checked the behavior or curve of our hypothesis function by making it

1. a linear function,
2. a quadratic function,
3. a cubic function, and
4. a quartic function (polynomial with a degree of 4).

The linear hypothesis function with $n = 2$ and $n = 3$ has the following form

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \quad \text{for} \quad n = 2, \tag{4.5}$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \quad \text{for} \quad n = 3. \tag{4.6}$$

The quadratic hypothesis function with $n = 5$ (2 primary and 3 secondary features) and $n = 9$ (3 primary and 6 secondary features) has the following form

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1 x_2 + \theta_5 x_2^2 \quad \text{for} \quad n = 5, \tag{4.7}$$

$$\begin{aligned} h_\theta(x) =& \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_1^2 + \theta_5 x_1 x_2 + \theta_6 x_1 x_3 \\ & + \theta_7 x_2^2 + \theta_8 x_2 x_3 + \theta_9 x_3^2 \quad \text{for} \quad n = 9. \end{aligned} \tag{4.8}$$

The cubic hypothesis function with $n = 9$ (2 primary and 7 secondary features) and $n = 19$ (3 primary and 16 secondary features) has the following form

$$\begin{aligned} h_\theta(x) =& \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1 x_2 + \theta_5 x_2^2 + \theta_6 x_1^3 \\ & + \theta_7 x_1^2 x_2 + \theta_8 x_1 x_2^2 + \theta_9 x_2^3 \quad \text{for} \quad n = 9, \end{aligned} \tag{4.9}$$

$$\begin{aligned} h_\theta(x) =& \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_1^2 + \theta_5 x_1 x_2 + \theta_6 x_1 x_3 + \theta_7 x_2^2 \\ & + \theta_8 x_2 x_3 + \theta_9 x_3^2 + \theta_{10} x_1^3 + \theta_{11} x_1^2 x_2 + \theta_{12} x_1^2 x_3 + \theta_{13} x_1 x_2^2 \\ & + \theta_{14} x_1 x_2 x_3 + \theta_{15} x_1 x_3^2 + \theta_{16} x_2^3 + \theta_{17} x_2^2 x_3 + \theta_{18} x_2 x_3^2 + \theta_{19} x_3^3 \quad \text{for} \quad n = 19. \end{aligned} \tag{4.10}$$

The quartic hypothesis function with $n = 14$ (2 primary and 12 secondary features) and $n = 33$ (3 primary and 30 secondary features) has the following form

$$\begin{aligned} h_\theta(x) =& \theta_0 + \theta_1\, x_1 + \theta_2\, x_2 + \theta_3\, x_1^2 + \theta_4\, x_1\, x_2 + \theta_5\, x_2^2 + \theta_6\, x_1^3 + \theta_7\, x_1^2\, x_2 + \theta_8\, x_1\, x_2^2 \\ &+ \theta_9\, x_2^3 + \theta_{10}\, x_1^4 + \theta_{11}\, x_1^3\, x_2 + \theta_{12}\, x_1^2\, x_2^2 + \theta_{13}\, x_1\, x_2^3 + \ \theta_{14}\, x_2^4 \quad \text{for} \quad n = 14, \end{aligned} \tag{4.11}$$

$$\begin{aligned} h_\theta(x) =& \theta_0 + \theta_1\, x_1 + \theta_2\, x_2 + \theta_3\, x_3 + \theta_4\, x_1^2 + \theta_5\, x_1\, x_2 + \theta_6\, x_1\, x_3 + \theta_7\, x_2^2 + \theta_8\, x_2\, x_3 \\ &+ \theta_9\, x_3^2 + \theta_{10}\, x_1^3 + \theta_{11}\, x_1^2\, x_2 + \theta_{12}\, x_1^2\, x_3 + \theta_{13}\, x_1\, x_2^2 + \theta_{14}\, x_1\, x_2\, x_3 + \theta_{15}\, x_1\, x_3^2 \\ &+ \theta_{16}\, x_2^3 + \theta_{17}\, x_2^2\, x_3 + \theta_{18}\, x_2\, x_3^2 + \theta_{19}\, x_3^3 + \theta_{20}\, x_1^4 + \theta_{21}\, x_1^3\, x_2 + \theta_{22}\, x_1^3\, x_3 + \theta_{23}\, x_1^2\, x_2^2 \\ &+ \theta_{24}\, x_1^2\, x_2\, x_3 + \theta_{25}\, x_1^2\, x_3^2 + \theta_{25}\, x_1\, x_2^3 + \theta_{26}\, x_1\, x_2^2\, x_3 + \theta_{27}\, x_1\, x_2\, x_3^2 + \theta_{28}\, x_1\, x_3^3 \\ &+ \theta_{29}\, x_2^4 + \theta_{30}\, x_2^3\, x_3 + \theta_{31}\, x_2^2\, x_3^2 + \theta_{32}\, x_2\, x_3^3 + \theta_{33}\, x_3^4 \quad \text{for} \quad n = 33. \end{aligned} \tag{4.12}$$

### 4.2.4 Cost Function

For a given logistic regression problem with a given $n$ input features and hypothesis function $h_\theta(x)$, our objective is to choose or learn the parameters $\theta$ so that $h_\theta(x)$ is close to $y$ for our training set $(x, y)$, i.e., the set of $m$ training examples. This is achieved by minimizing logistic regression cost function for our training set until convergence.

The logistic regression cost function $J(\theta)$ has the following form

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(h_\theta(x^{(i)})) + (1 - y^{(i)})\log(1 - h_\theta(x^{(i)}))]. \tag{4.13}$$

Notice that when $y$ is equal to 1, then the second term $(1 - y^{(i)})\log(1 - h_\theta(x^{(i)}))$ becomes zero and will not affect the result. If $y$ is equal to 0, then the first term $y^{(i)}\log(h_\theta(x^{(i)}))$ will be zero and will not affect the result.

To minimize the cost function, we use gradient descent algorithm. Gradient descent is a first-order iterative optimization algorithm for finding the minimum of a function. The gradient descent algorithm is to repeat

$$\begin{aligned}\theta_j &:= \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta), \\ &:= \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})\, x_j^{(i)}, \end{aligned} \tag{4.14}$$

until convergence. The $\alpha$ is the learning rate of the gradient descent and $j = 0, 1, \cdots, n$ represents the feature index number.

It is possible that the form of our hypothesis $h_\theta(x)$ a) maps poorly to the trend of the data, or b) fits the available data but does not generalize well to predict new data. When the form of our hypothesis function $h_\theta(x)$ maps poorly to the trend of the data, its called under-fitting or high bias. It is usually caused by a function that is too simple or uses too few features. Thus, we need to consider higher-order polynomial functions as our hypothesis function to eradicate this problem. At the other extreme, over-fitting, or high variance, is caused by a hypothesis function that fits the available data but does not generalize well to predict new data. It is usually caused by a complicated function that creates a lot of unnecessary curves and angles unrelated to the data. Two main options to address the problem of over-fitting are a) reducing the number of features through proper selection, and b) regularization. By using regularization, we can keep all the features that we consider important, but reduce the magnitude of parameters $\theta_j$. Regularization works well when we have a lot of slightly useful features. An example of under-fit, proper-fit, and over-fit is shown in the Figure 4.6.

In order to avoid over-fitting, regularization is used in this formulation. The cost function given in (4.13) is without regularization. When using regularization,

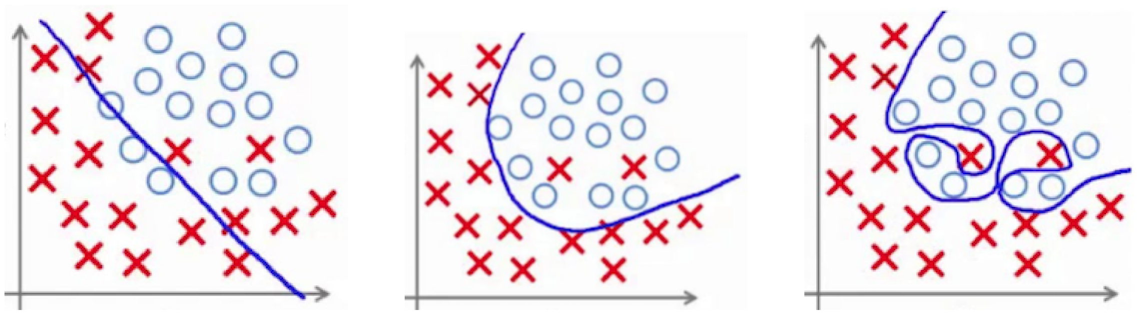


Figure 4.6: An example of under-fitting, proper-fit, and over-fitting.

the function changes to the following form

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2. \quad (4.15)$$

The gradient descent algorithm in this case is to repeat

$$\theta_0 := \theta_0 - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)}, \text{and} \quad (4.16)$$

$$\theta_j := \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\theta_j\right] \quad \text{for} \quad j \in \{1, 2, \cdots, n\}, \quad (4.17)$$

until convergence. The $\lambda$ is the regularization parameter. It determines how much the costs of our theta parameters are inflated. If the value of $\lambda$ is chosen to be too large, it may smooth out the hypothesis function too much and result in under-fitting. If $\lambda = 0$, the (4.15) reduces down to be same as (4.13). Similarly, if the $\lambda$ is very small, it won't be strong enough to regularize the cost-function, and the results will be similar to that of an un-regularized case.

## 4.3 Channel Condition Prediction Results

For eight different hypotheses represented by equation (4.5) through (4.12), presented in Section 4.2.3, we formulated our logistic regression problem as discussed in Section 4.2.1. We are using regularization in our problem formulation with five

Table 4.4: A summary of all the logistic regression analysis conducted.

| $n$ | Primary features | Regularization parameter $\lambda$ | Hypothesis |
|---|---|---|---|
| 2 | $x_1$ = antenna elevation angle<br>$x_2$ = wet-path delay | 0, 0.1, 1, 10, 100 | (4.5) |
| 2 | $x_1$ = antenna elevation angle<br>$x_2$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.5) |
| 2 | $x_1$ = wet-path delay<br>$x_2$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.5) |
| 3 | $x_1$ = antenna elevation angle<br>$x_2$ = wet-path delay<br>$x_3$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.6) |
| 5 | $x_1$ = antenna elevation angle<br>$x_2$ = wet-path delay | 0, 0.1, 1, 10, 100 | (4.7) |
| 5 | $x_1$ = antenna elevation angle<br>$x_2$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.7) |
| 5 | $x_1$ = wet-path delay<br>$x_2$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.7) |
| 9 | $x_1$ = antenna elevation angle<br>$x_2$ = wet-path delay | 0, 0.1, 1, 10, 100 | (4.9) |
| 9 | $x_1$ = antenna elevation angle<br>$x_2$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.9) |
| 9 | $x_1$ = wet-path delay<br>$x_2$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.9) |
| 9 | $x_1$ = antenna elevation angle<br>$x_2$ = wet-path delay<br>$x_3$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.8) |
| 14 | $x_1$ = antenna elevation angle<br>$x_2$ = wet-path delay | 0, 0.1, 1, 10, 100 | (4.11) |
| 14 | $x_1$ = antenna elevation angle<br>$x_2$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.11) |
| 14 | $x_1$ = wet-path delay<br>$x_2$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.11) |
| 19 | $x_1$ = antenna elevation angle<br>$x_2$ = wet-path delay<br>$x_3$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.10) |
| 33 | $x_1$ = antenna elevation angle<br>$x_2$ = wet-path delay<br>$x_3$ = antenna azimuth | 0, 0.1, 1, 10, 100 | (4.12) |

different values of $\lambda$ given as $\lambda \in \{0, 0.1, 1, 10, 100\}$. That is, for each case with $n$ input features, we conducted regression analysis with $\lambda \in \{0, 0.1, 1, 10, 100\}$, separately utilizing our training set. We are using $\lambda = 0$ as well in order to examine if the regularization is needed at all or not for our problem.

Overall, we conducted logistic regression analysis for 80 different cases, which is summarized in Table 4.4. The fourth column 'Hypothesis' represents the equation that defines the corresponding hypothesis form.

Among all these different logistic regression formulations, we achieve highest estimation accuracy with quadratic hypothesis function with $n = 9$ features as represented by (4.8) when $\lambda = 0$ followed by $\lambda = 0.1$, $\lambda = 1$, $\lambda = 10$ and $\lambda = 100$. The list of hypothesis obtained for $\lambda = 0$ case is given as below

$$\begin{aligned}
h_\theta^{(A)}(x) = {} & 3.6658 + 0.1663\, x_1 - 0.18427\, x_2 - 0.039374\, x_3 - 0.085769\, x_1^2 + 0.24355\, x_1\, x_2 \\
& + 8.7428 \times 10^{-3}\, x_1\, x_3 - 0.15164\, x_2^2 - 0.01172\, x_2\, x_3 + 3.0657 \times 10^{-5}\, x_3^2, \\
h_\theta^{(B)}(x) = {} & - 6.0353 + 2.0304\, x_1 - 4.2893\, x_2 - 0.079279\, x_3 - 0.13088\, x_1^2 + 0.52784\, x_1\, x_2 \\
& + 9.5559 \times 10^{-3}\, x_1\, x_3 - 0.51502\, x_2^2 - 0.01859\, x_2\, x_3 - 2.2897 \times 10^{-4}\, x_3^2, \\
h_\theta^{(C)}(x) = {} & - 2.7881 + 1.0825\, x_1 - 2.9887\, x_2 - 0.045779\, x_3 - 0.067003\, x_1^2 + 0.33904\, x_1\, x_2 \\
& + 5.2547 \times 10^{-3}\, x_1\, x_3 - 0.41138\, x_2^2 - 0.012909\, x_2\, x_3 - 2.0249 \times 10^{-4}\, x_3^2, \\
h_\theta^{(D)}(x) = {} & - 1.1447 + 0.72336\, x_1 - 3.6949\, x_2 - 0.033315\, x_3 - 0.023825\, x_1^2 + 0.14242\, x_1\, x_2 \\
& + 1.975 \times 10^{-3}\, x_1\, x_3 - 0.055359\, x_2^2 - 5.1793 \times 10^{-3}\, x_2\, x_3 - 3.1994 \times 10^{-5}\, x_3^2, \\
h_\theta^{(E)}(x) = {} & - 9.5155 \times 10^{-3} + 2.7395 \times 10^{-3}\, x_1 - 0.033308\, x_2 - 7.8312 \times 10^{-4}\, x_3 \\
& + 6.717 \times 10^{-3}\, x_1^2 - 0.02676\, x_1\, x_2 - 3.7892 \times 10^{-4}\, x_1\, x_3 - 0.061824\, x_2^2 \\
& - 2.8362 \times 10^{-4}\, x_2\, x_3 - 2.9898 \times 10^{-4}\, x_3^2.
\end{aligned}$$

Table 4.5: Channel bit-SNR estimation accuracy (in %) achieved with the logistic regression based prediction model with RTT = 20 minutes.

(a) Communication pass on September 6, 2005 data.

| % of SNRs | Ranges | Predicted SNR range | | | | |
|---|---|---|---|---|---|---|
| | | Range A | Range B | Range C | Range D | Range E |
| 0.000 | Range A | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range B | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range C | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range D | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 100.000 | Range E | 0.000 | 0.000 | 0.000 | 0.000 | 100.000 |
| % of predicted SNR | | 0.000 | 0.000 | 0.000 | 0.000 | 100.000 |
| Percentage of correct estimation: 100.000 | | | | | | |

(b) Communication pass on June 21, 2009 data.

| % of SNRs | Ranges | Predicted SNR range | | | | |
|---|---|---|---|---|---|---|
| | | Range A | Range B | Range C | Range D | Range E |
| 0.000 | Range A | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range B | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range C | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 12.181 | Range D | 0.000 | 0.000 | 0.000 | 71.236 | 28.764 |
| 87.819 | Range E | 0.000 | 0.000 | 0.000 | 1.524 | 98.476 |
| % of predicted SNR | | 0.000 | 0.000 | 0.000 | 10.016 | 89.984 |
| Percentage of correct estimation: 95.158 | | | | | | |

(c) Communication pass on August 2, 2011 data.

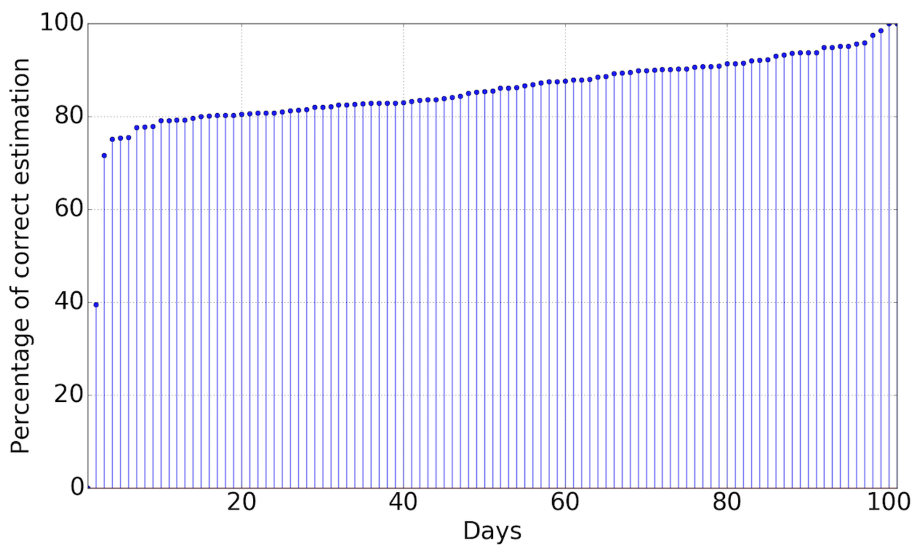
| % of SNRs | Ranges | Predicted SNR range | | | | |
|---|---|---|---|---|---|---|
| | | Range A | Range B | Range C | Range D | Range E |
| 0.000 | Range A | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 39.008 | Range B | 0.000 | 0.000 | 0.000 | 100.000 | 0.000 |
| 60.992 | Range C | 0.000 | 0.000 | 0.000 | 100.000 | 0.000 |
| 0.000 | Range D | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range E | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| % of predicted SNR | | 0.000 | 0.000 | 0.000 | 100.000 | 0.000 |
| Percentage of correct estimation: 0.000 | | | | | | |

Up to this point, we presented the details of the classification problem and learnt the logistic regression parameters with the help of the training data set. We now evaluate the accuracy of the developed logistic regression prediction model, defined by hypothesis $h_\theta^{(A)}(x)$ through $h_\theta^{(E)}(x)$, to predict channel bit-SNR values 20 mins into the future for all the communication passes in our test data set. The prediction model is run once for each communication pass of the data set, percentage of correct range estimation for each of them is obtained with RTT = 20 minutes. The Table 4.5 demonstrates the % of estimation accuracy in individual bit-SNR ranges and overall estimation accuracy during three test communication passes.

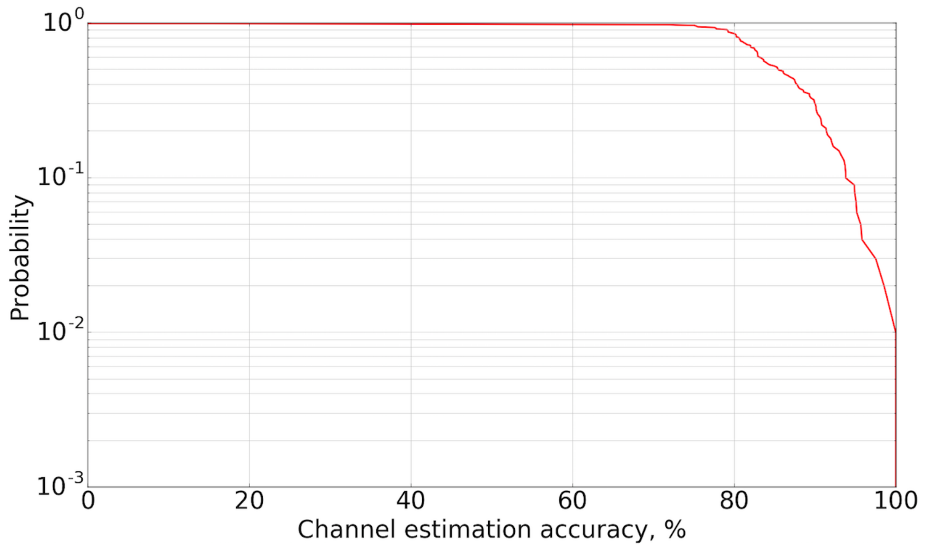
For different communication passes, varying percentage of correct estimation is observed, with maximum and minimum percentage of correct estimation being 100% and 0% respectively, as shown in Table 4.5a and Table 4.5c with RTT = 20 minutes.

Percentage of correct estimation achieved during each communication pass when RTT = 20 minutes are arranged in increasing order and plotted as shown in Figure 4.7a. Among these 101 test communication passes, on 2 passes, our prediction accuracy is less than 70% with least being 0%. For 99 passes, we have prediction accuracy more than 70%, and in turn, prediction accuracy is more than 80% for 86 communication passes. Figure 4.7b shows corresponding complementary CDF plot of percentage of correct estimation of all the test passes. From the plot we can see that for more than 85% of the passes we can predict channel conditions with more than 80% accuracy. We also computed mean and variance of the percentage of correct estimation of all these passes. The mean of percentage of correct estimation is 84.935 % and its variance is 128.856 %.

To further evaluate the effectiveness of the prediction algorithm under different round trip time scenarios, we conducted future channel bit-SNR predictions



(a) Percentage of correct channel bit-SNR estimation.



(b) Complementary CDF of percentage of correct estimation.

Figure 4.7: Plots obtained using the logistic regression based channel bit-SNR range prediction model for our test dataset with RTT = 20 minutes.

for the same 101 test communications passes when RTT = 6.5 minutes as well as RTT = 44 minutes. Percentage of correct estimation achieved in each communication pass are arranged in increasing order and plotted against the communication pass when RTT = 6.5 minutes and RTT = 44 minutes is shown in Figure 4.8 and Figure 4.9, respectively. As expected, different percentages of correct estimation are observed for different communication passes. When the RTT = 6.5 minutes, maximum and minimum percentage of correct estimation is 100% and 2.653% respectively. Average percentage of correct estimation obtained for all the passes is 91.698% with the variance of 112.362%. When the RTT = 44 minutes, maximum and minimum percentage of correct estimation is 100% and 0% respectively. Average percentage of correct estimation obtained for all the passes is 76.958% with the variance of 169.897%. This shows that the accuracy of logistic regression based prediction algorithm is dependent on the RTTs scenarios. Prediction accuracy is better for smaller RTT scenarios and it gradually decreases as the RTT between transmitter and receiver increases.

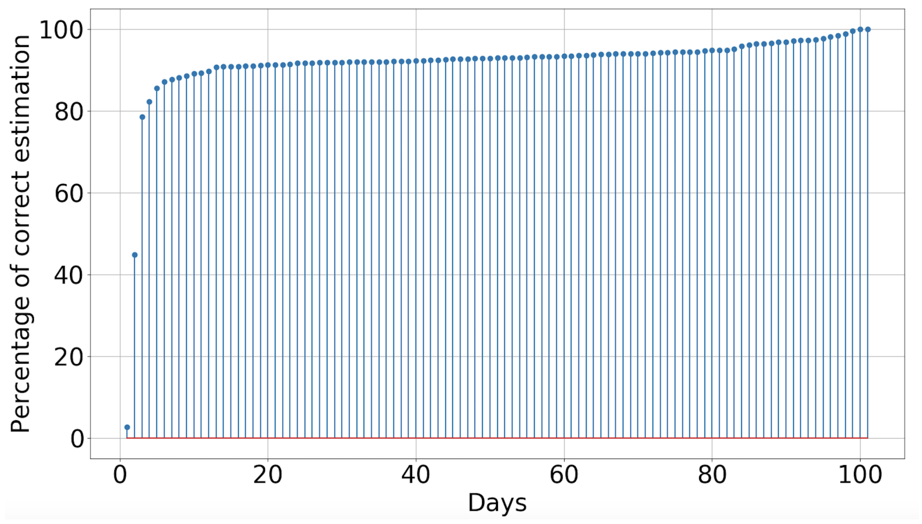


Figure 4.8: Percentage of correct channel bit-SNR estimations obtained with the logistic regression based channel bit-SNR range prediction model for our test dataset with RTT = 6.5 minutes.

Taking a closer look at the nature of estimation errors, we have noticed that most of the estimation errors occur when the actual future channel bit-SNR corresponds to range B and C, and we estimate it to be range D. It signifies that our obtained hypotheses $h_\theta^{(B)}(x)$ and $h_\theta^{(C)}(x)$ are not strong enough to clearly define their corresponding ranges. Possible reasons behind this are

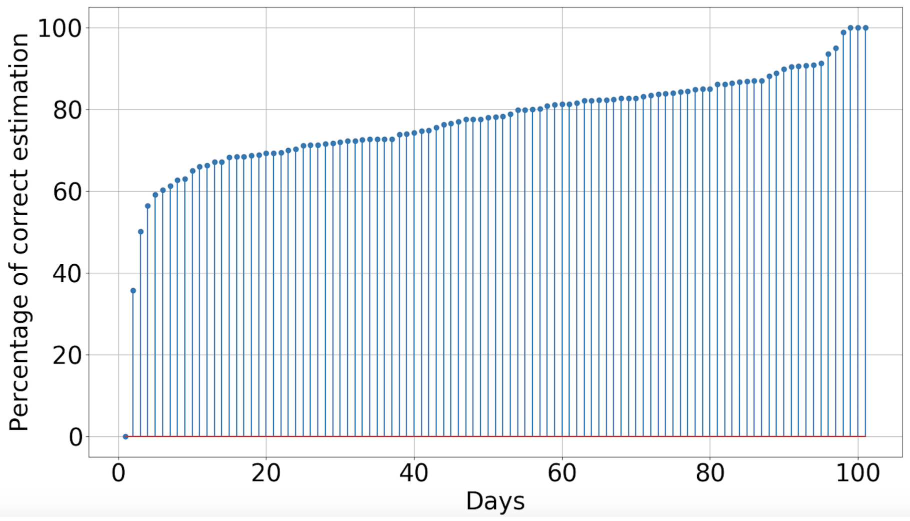


Figure 4.9: Percentage of correct channel bit-SNR estimations obtained with the logistic regression based channel bit-SNR range prediction model for our test dataset with RTT = 44 minutes.

1. Lesser number of test data corresponding to range B and C, resulting in very weak hypothesis for these ranges, and
2. The span of range B and C are -0.5 to -0.1 dB, and -0.1 to 0.15 dB, respectively. Compared to range A, D and E, bit-SNR span of the range B and C are smaller.

The reason 1) can be attributed to the reason 2) as well. In our case, we are limited by the 202 number of communication data passes, out of which 101 are used as training data set and rest of the 101 passes are used as test data set. Even with the limited set of data, for more than 85% of the test passes we can predict channel conditions with more than 80% accuracy. This is pretty impressive in itself, given the simplicity of the prediction model. If we had larger the number of test data, with sufficient number of examples belonging to each bit-SNR ranges, obtained hypotheses can potentially be stronger and capable of capturing all possible patterns. In real DSN communications, they have huge library of past measurements (more than 20 years at Madrid DSCC, 17 years at Goldstone DSCC and 9 years at Canberra DSCC

[1]), utilizing which, it is possible that a very accurate logistic regression based future channel bit-SNR range prediction model could be obtained.

As we cannot make changes to the nature of the data, we could use some stronger learning algorithms and see how their results compare to that of the logistic regression. Thus, similar to what we did in this chapter, we will develop a neural network based prediction algorithm and see if we can further improve the accuracy of predictions.

## 4.4 Conclusion

In this chapter, we employed logistic regression to predict the value of future channel bit-SNR ranges. We obtained hypotheses for numbers of input features from 2 to 33 and four different type of hypothesis functions. We found that most accurate hypotheses are obtained when we have 9 secondary input features obtained from 3 primary features, a quadratic hypothesis function given by (4.8), and $\lambda = 0$.

# CHAPTER 5

# NEURAL NETWORKS BASED PREDICTION MODEL

In the first part of this chapter, a brief introduction to dense neural networks (DNN) based data analysis approach and classification problems is given. In DNN, every node in each layer is connected to every node in the following layer. In the second part, details on training DNN and using them to design a channel condition prediction model are given.

## 5.1 Introduction to Neural Networks

Neural networks, also known as an artificial neuron network, is the supervised learning algorithm of choice for many classification and machine learning problems. Neural networks is a much better way than other supervised learning algorithms to learn complex nonlinear hypotheses even when the input feature space $n$ is large. The origins of neural networks was as algorithms that try to mimic the networks of neurons in the brain. Work on artificial neural net models has a long history. Development of detailed mathematical models began more than 70 years ago with the work of [37], [38], [39] and [40]. At a very simple level, neurons are computational units that take electrical inputs from dendrites and channel the computed outputs to axons.

### 5.1.1 Model Representation

Neural networks use a very simple model of what a neuron does, in which a single neuron is modeled as just a logistic unit as shown in Figure 5.1. This

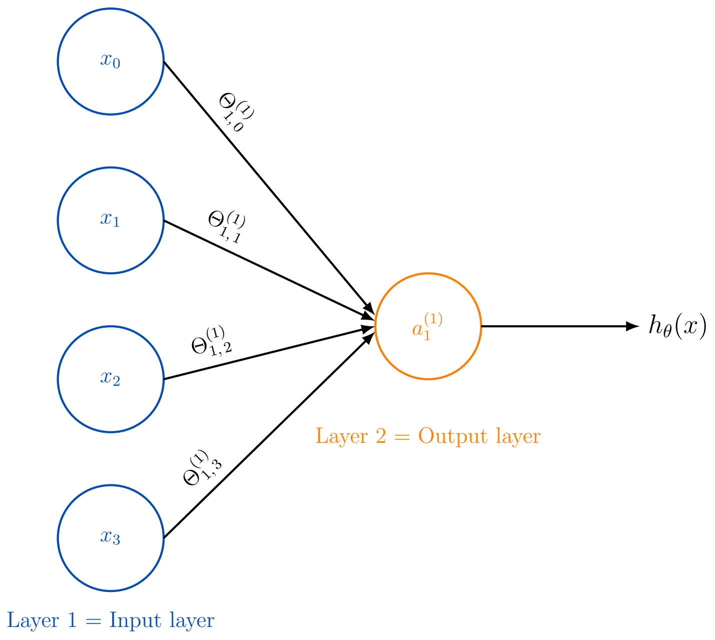


Figure 5.1: A simple representation of the body of a single neuron represented by the orange circle, with three inputs $x_1, x_2, x_3$ and an output $h_\theta(x)$.

shows that the neuron takes three ($n = 3$) inputs $x_1, x_2$, and $x_3$ via *input wires*, does some computation and outputs the value of a hypothesis function $h_\theta(x)$ on its *output wire*. The value of $x_0$ is always 1, and is called a *bias unit*. In neural networks terminology, the neuron has one *input layer* with three input nodes ($x_1, x_2$, and $x_3$) and a bias unit $x_0$, and one *output layer*. The first layer that is used to input the features is called the input layer. The final layer that outputs the value computed by a hypothesis is called the output layer. Neural networks hypotheses are represented by a sigmoid or logistic function $g(z)$(equation (4.2)), that was used for logistic regression classification problem in Section 4.1.1, and is given as

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}, \tag{5.1}$$

where $x$ is our input vector and $\theta$ is the parameter vector given as

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix},$$

and

$$\theta^T = \begin{bmatrix} \theta_0 & \theta_1 & \theta_2 & \theta_3 \end{bmatrix}.$$

A neural network is a group of different neurons strung together. Training a neural network basically means choosing all of the parameters or weights by repeating two key steps, forward propagation and back propagation, which will be covered in later parts of the chapter.

A neural network with three inputs, a *hidden layer* and three output classes represented is presented in Figure 5.2. The layer in-between the input layer and the output layer is called the hidden layer. A neural network can have more than one hidden layer as well, as per requirement.

The hidden layer nodes are also known as intermediate nodes or *activation units*. The value $a_u^{(l)}$ at the output of the $u$th, $u \in \{0, 1, \cdots, U_l - 1\}$ intermediate node in layer $l$, $l \in \{1, 2, \cdots, L\}$ is obtained using the activation function of the form $a_u^{(l)} = g(z_u^{(l)})$. The $a_u^{(l)}$ signifies activation of unit $u$. $U_l$ represents the number of units in layer $l$, and $L$ is the total number of layers in the network. Given $n$ input features and $C$ output classes, the number of hidden layers and the number of activation units per hidden layer can be varied as per requirement to achieve desired prediction accuracy. Addition of intermediate layers in neural networks results in more complex non-linear hypotheses. Each layer $l$ has its own matrix of parameters (weight) $\Theta^{(l)}$ that controls function mapping from layer $l$ to layer $l+1$. It is essential

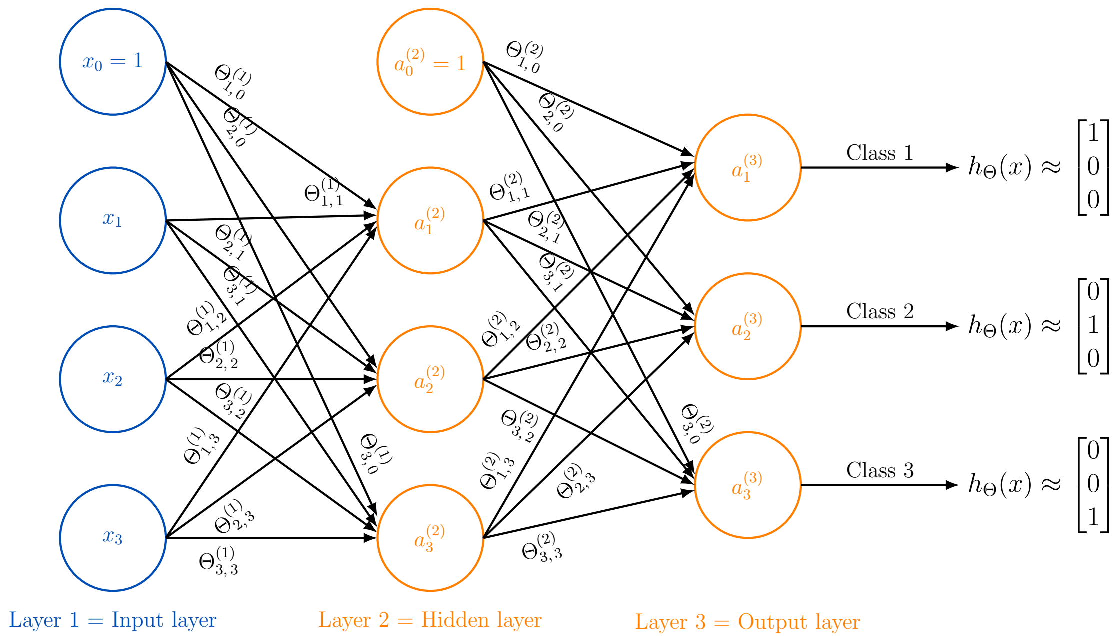


Figure 5.2: A neural network with three inputs $x_1, x_2, x_3$, a hidden layer, and three output classes. The $h_\Theta(x) \in \mathbb{R}^3$.

to understand that every input or activation in current layer is connected to every node in the following layer. If the network has $U_l$ units in layer $l$ and $U_{l+1}$ units in layer $l+1$, then $\Theta^{(l)}$ will be of dimension $(U_{l+1} - 1) \times U_l$ and is given as

$$\Theta^{(l)} = \left[\Theta^{(l)}_{v,u}\right] = \begin{bmatrix} \Theta^{(l)}_{1,0} & \Theta^{(l)}_{1,1} & \cdots & \Theta^{(l)}_{1,(U_l-1)} \\ \Theta^{(l)}_{2,0} & \Theta^{(l)}_{2,1} & \cdots & \Theta^{(l)}_{2,(U_l-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \Theta^{(l)}_{(U_{l+1}-1),0} & \Theta^{(l)}_{(U_{l+1}-1),1} & \cdots & \Theta^{(l)}_{(U_{l+1}-1),(U_l-1)} \end{bmatrix}, \qquad (5.2)$$

where $\Theta^{(l)}_{v,u}$ represents the weight that controls mapping from unit $u$ of layer $l$ to unit $v$ of layer $l+1$. The values for each of the activation nodes $a^{(l)}$ of layer $l = 2$ is obtained as follows

$$a^{(2)} = \begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \\ \vdots \\ a_{(U_2-1)}^{(2)} \end{bmatrix} = \begin{bmatrix} g(z_1^{(2)}) \\ g(z_2^{(2)}) \\ \vdots \\ g(z_{(U_2-1)}^{(2)}) \end{bmatrix} \quad \text{for } l = 2, \tag{5.3}$$

with

$$z^{(2)} = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ \vdots \\ z_{(U_2-1)}^{(2)} \end{bmatrix} = \begin{bmatrix} \Theta_{1,0}^{(1)} & \Theta_{1,1}^{(1)} & \cdots & \Theta_{1,(U_1-1)}^{(1)} \\ \Theta_{2,0}^{(1)} & \Theta_{2,1}^{(1)} & \cdots & \Theta_{2,(U_1-1)}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ \Theta_{(U_2-1),0}^{(1)} & \Theta_{(U_2-1),1}^{(1)} & \cdots & \Theta_{(U_2-1),(U_1-1)}^{(1)} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{U_1-1} \end{bmatrix} = \Theta^{(1)}\, x. \tag{5.4}$$

The values for each of the activation nodes of layer $2 < l < L$ is obtained as follows

$$a^{(l)} = \begin{bmatrix} a_1^{(l)} \\ a_2^{(l)} \\ \vdots \\ a_{(U_l-1)}^{(l)} \end{bmatrix} = \begin{bmatrix} g(z_1^{(l)}) \\ g(z_2^{(l)}) \\ \vdots \\ g(z_{(U_l-1)}^{(l)}) \end{bmatrix} \quad \text{for } 2 < l < L, \tag{5.5}$$

with

$$z^{(l)} = \begin{bmatrix} z_1^{(l)} \\ z_2^{(l)} \\ \vdots \\ z_{(U_l-1)}^{(l)} \end{bmatrix} = \begin{bmatrix} \Theta_{1,0}^{(l-1)} & \Theta_{1,1}^{(l-1)} & \cdots & \Theta_{1,(U_{(l-1)}-1)}^{(l-1)} \\ \Theta_{2,0}^{(l-1)} & \Theta_{2,1}^{(l-1)} & \cdots & \Theta_{2,(U_{(l-1)}-1)}^{(l-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \Theta_{(U_l-1),0}^{(l-1)} & \Theta_{(U_l-1),1}^{(l-1)} & \cdots & \Theta_{(U_l-1),(U_{(l-1)}-1)}^{(l-1)} \end{bmatrix} \begin{bmatrix} a_0^{(l-1)} \\ a_1^{(l-1)} \\ \vdots \\ a_{U_{(l-1)}-1}^{(l-1)} \end{bmatrix} = \Theta^{(l-1)}\, a^{(l-1)} \tag{5.6}$$

For $l = L$, there are $C$ output classes indexed by $c \in \{1, 2, \cdots, C\}$ with

$U_L = C$, then the equations (5.5) and (5.6) can be re-written as

$$h_\Theta(x) = \begin{bmatrix} h_\Theta(x)_1 \\ h_\Theta(x)_2 \\ \vdots \\ h_\Theta(x)_C \end{bmatrix} = \begin{bmatrix} a_1^{(L)} \\ a_2^{(L)} \\ \vdots \\ a_C^{(L)} \end{bmatrix} = \begin{bmatrix} g(z_1^{(L)}) \\ g(z_2^{(L)}) \\ \vdots \\ g(z_C^{(L)}) \end{bmatrix}, \tag{5.7}$$

and

$$\begin{bmatrix} z_1^{(L)} \\ z_2^{(L)} \\ \vdots \\ z_C^{(L)} \end{bmatrix} = \begin{bmatrix} \Theta_{1,0}^{(L-1)} & \Theta_{1,1}^{(L-1)} & \cdots & \Theta_{1,(U_{(L-1)}-1)}^{(L-1)} \\ \Theta_{2,0}^{(L-1)} & \Theta_{2,1}^{(L-1)} & \cdots & \Theta_{2,(U_{(L-1)}-1)}^{(L-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \Theta_{C,0}^{(L-1)} & \Theta_{C,1}^{(L-1)} & \cdots & \Theta_{C,(U_{(L-1)}-1)}^{(L-1)} \end{bmatrix} \begin{bmatrix} a_0^{(L-1)} \\ a_1^{(L-1)} \\ \vdots \\ a_{U_{(L-1)}-1}^{(L-1)} \end{bmatrix}, \tag{5.8}$$

respectively. Here $h_\Theta(x)_c$ denotes the hypothesis that results in the $c^{th}$ output. The output layer does not have a bias unit, and hence the indexing is from 1 to $U_L = C$, unlike from 1 to $U_{(l-1)}$ as was used for hidden layers. The resulting hypothesis $h_\Theta(x)$ is

$$h_\Theta(x) = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{5.9}$$

and it represents that the output corresponds to class 1, i.e., $h_\Theta(x)_1 = 1$. Combining all the information provided above, a dense neural networks setup with $n$ features,

$L-2$ hidden layers and $C$ output classes has the following form:

$$\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \rightarrow \begin{bmatrix} a_0^{(2)} \\ a_1^{(2)} \\ \vdots \\ a_{U_2}^{(2)} \end{bmatrix} \rightarrow \cdots \rightarrow \begin{bmatrix} a_0^{(L-1)} \\ a_1^{(L-1)} \\ \vdots \\ a_{U_{L-1}}^{(L-1)} \end{bmatrix} \rightarrow \begin{bmatrix} h_\Theta(x)_1 \\ h_\Theta(x)_2 \\ \vdots \\ h_\Theta(x)_C \end{bmatrix}. \tag{5.10}$$

Basically, a set of weights is applied to the input data, then the activation of each layer is computed sequentially, and finally the output is computed. This process of applying a set of weights to an input data and calculating an output is called forward propagation in neural networks.

### 5.1.2 Cost Function and Back Propagation Algorithm

As defined in Chapter 4, we will use $x^{(i)}$ to denote the input features of the $i$th training example, and $y^{(i)}$ to denote the output or target variable of the $i$th training example that we are trying to predict. A pair $(x^{(i)}, y^{(i)})$ is called a training example, and the dataset, i.e., a list of $m$ training examples $(x^{(i)}, y^{(i)})$; $i = 1, 2, \cdots, m$, that we will be using to learn is called a training set represented as $(x, y)$. We will use $x_j$ to denote feature $j$, $j = 1, 2, \cdots, n$, and $x_j^{(i)}$ to denote the value of feature $j$ in the $i$th training example. We will also use $y_c$ to denote output class $c$, $c \in \{1, 2, \cdots, C\}$, and $y_c^{(i)}$ to denote the value of output class $c$ in the $i$th training example. The superscript $(i)$ in the notation is simply an index into the training set. We will also use $X$ to denote the space of input values, and $Y$ to denote the space of output values. Our goal is, given a training set, to learn weight matrices $\Theta^{(l)}$ for each layer $l$ in the network, i.e., $\Theta^{(2)}, \Theta^{(3)}, \cdots, \Theta^{(L-1)}$, so that $h_\Theta(x)_c$ is a good predictor for the corresponding value of $y_c$.

Cost function for neural networks is a generalization of the regularized cost function (4.15) that we used for logistic regression in Chapter 4, and is given as

$$J(\Theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{c=1}^{C}\left[y_c^{(i)}\,\log\left(h_\Theta(x^{(i)})_c\right) + (1-y_c^{(i)})\,\log\left(1-h_\Theta(x^{(i)})_c\right)\right]$$
$$+\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{u=0}^{(U_l-1)}\sum_{v=1}^{(U_{l+1}-1)}\left(\Theta_{v,u}^{(l)}\right)^2, \tag{5.11}$$

where $\lambda$ is the regularization parameter for our cost function. We now need to minimize the cost function using an optimal set of parameters $\Theta$ for our training set so that we can obtain best parameter matrices $\Theta^{(2)}, \Theta^{(3)}, \cdots, \Theta^{(L-1)}$ resulting in best predictor for each output classes $c$. That is,

$$\min_\Theta J(\Theta). \tag{5.12}$$

In order to use either gradient descent or one of the advance optimization algorithms, we need to compute partial derivative terms of $J(\Theta)$ given as

$$\frac{\partial}{\partial\Theta_{i,j}^{(l)}}J(\Theta), \tag{5.13}$$

where $\Theta_{i,j}^{(l)} \in \mathbb{R}$ are the parameters of the neural network.

In neural networks, back propagation algorithm is used to compute partial derivative terms of our cost function as shown in (5.13). In back propagation, the margin of error of the output is measured and the weights are adjusted accordingly to decrease the error. Neural networks repeat both forward and back propagation until the weights are chosen to accurately predict an output. Given a training set $\left\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\right\}$, the back propagation algorithm can be summarized as follows:

1. Set $\Delta_{i,j}^{(l)} = 0$ for all $l$, $i$ and $j$.

2. For $i = 1$ to $i = m$:

   - Set $a^{(1)} = x^i$, where $a^{(l)}$ is a vector of activations for layer $l$.
   - Perform forward propagation to compute $a^{(l)}$, for $l = 2, 3, \cdots, L$.
   - Using $y^{(i)}$ and the obtained value of $a^{(L)}$, compute $\delta^{(L)} = a^{(L)} - y^{(i)}$.
   - Using $\delta^{(L)}$, compute the value of $\delta^{(L-1)}, \delta^{(L-2)}, \cdots, \delta^{(2)}$, using the following relation

   $$\begin{aligned}\delta^{(l)} &= ((\Theta^{(l)})^T \delta^{(l+1)}) \ \circ g'(z^{(l)}), \\ &= ((\Theta^{(l)})^T \delta^{(l+1)}) \ \circ a^{(l)} \ \circ (1 - a^{(l)}). \end{aligned} \tag{5.14}$$

     where $M_1 \circ M_2$ denotes element-wise multiplication of two matrices $M_1$ and $M_2$.
   - Update $\Delta_{i,j}^{(l)}$ as follows:

   $$\Delta_{i,j}^{(l)} := \Delta_{i,j}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}.$$

3. Compute the value of $D_{i,j}^{(l)}$ as

$$D_{i,j}^{(l)} = \begin{cases} \dfrac{1}{m}\left(\Delta_{i,j}^{(l)} + \lambda \Theta_{i,j}^{(l)}\right) & \text{if } j \neq 0, \\ \\ \dfrac{1}{m}\Delta_{i,j}^{(l)} & \text{if } j = 0. \end{cases}$$

   The case of $j = 0$ corresponds to our bias terms and hence it does not have a regularization term.

4. The obtained $D_{i,j}^{(l)}$ is the value of $\frac{\partial}{\partial \Theta_{i,j}^{(l)}} J(\Theta)$, that is,

$$\frac{\partial}{\partial \Theta_{i,j}^{(l)}} J(\Theta) = D_{i,j}^{(l)}. \tag{5.15}$$

We are using *fminunc* function of matlab to minimize the cost function $J(\Theta)$.

## 5.2 Channel Condition Prediction Model

We have covered all the details essential to understand a neural network and how it functions. We now know how a neural network is trained for a given training set, inputs and output classes, and how its weights are obtained using the forward propagation and the backward propagation algorithm. In the this section, we will look into the details of neural network based channel condition prediction model.

Similar to what we did in logistic regression based channel condition prediction model, instead of predicting channel bit-SNR value, we directly predict the possible bit-SNR range one RTT (20 mins) into the future. Thus, we won't have knowledge of future bit-SNR value exactly, however, we will know the possible range to which bit-SNR corresponds to. This is an application of neural networks to develop a classification problem.

As we know that we have five different bit-SNR ranges defined for dynamic turbo code assignments, we are defining each range as an output class for our neural network. Thus, we have five output classes $C = 5$ indexed as $c \in \{1, 2, 3, 4, 5\}$, where

$$\begin{aligned} &\text{range } A \rightarrow \text{class } 1, \\ &\text{range } B \rightarrow \text{class } 2, \\ &\text{range } C \rightarrow \text{class } 3, \\ &\text{range } D \rightarrow \text{class } 4, \\ &\text{range } E \rightarrow \text{class } 5. \end{aligned}$$

The resulting $h_\Theta(x) \in \mathbb{R}^5$ and is given as

$$h_\Theta(x) = \begin{bmatrix} h_\Theta(x)_1 \\ h_\Theta(x)_2 \\ h_\Theta(x)_3 \\ h_\Theta(x)_4 \\ h_\Theta(x)_5 \end{bmatrix}. \tag{5.16}$$

For the obtained output to correspond to a particular class $c$, $h_\Theta(x)_c = 1$ and 0 otherwise.

From our logistic regression analysis, we know that the highest prediction accuracy is achieved when we were using $n = 9$ with three primary features given as a) $x_1$ = zenith wet path delay, b) $x_2$ = antenna elevation, and c) $x_3$ = antenna azimuth. Unlike in logistic regression, we don't need to deal with the secondary features ourselves. That part is taken care of by hidden layers and the number of units in hidden layers of neural network. Hence, we are considering these three primary features as our input features, i.e., $n = 3$.

In neural networks, by default, one hidden layer is used for problems with smaller complexity. If one hidden layer is found to be insufficient for the problem, only then we move to multiple hidden layers. Thus, our neural network contains one hidden layer resulting in $L = 3$, that is, one input layer, one hidden layer and one output layer. There is not a standard number of units to be used per hidden layer, so we need to explore and try a lot of different combinations. Usually more the number of units per hidden layer, better it is (must balance with cost of computation as it increases with more hidden units). In general, the number of units per hidden layer should be

1. In-between the input layer size $n$ and the output layer size $C$.

2. Less than twice the number of inputs, that is, $< 2n$.

3. 70 to 90 % of the size of input layer, that is, $0.7\,n$ to $0.9\,n$.

As we can see that, there is no clear standard to follow, we have experimented with different number of units $U_2$ in the hidden layer. Just to make sure we examine with all the possible values of hidden units, we have $U_2 \in \{4, 5, 6, \cdots, 20\}$. We are also examining results for different values of regularization parameter $\lambda$, $\lambda \in \{0, 1, 2, \cdots, 10\}$. Details of all the properties that characterize our neural network is presented in Table 5.1.

Table 5.1: A summary of properties that characterize the neural network based channel prediction model.

| Property | Count | Values |
|---|---|---|
| Input features | $n = 3$ | $x_1$ = zenith wet path delay, $x_2$ = antenna elevation, and $x_3$ = antenna azimuth |
| Layers | $L = 3$ | 1 input layer, 1 hidden layer, and 1 output layer |
| Output classes | $C = 5$ | $c \in \{1, 2, 3, 4, 5\}$ |
| Units in hidden layer | $U_2$ (for multiple values) | $U_2 \in \{4, 5, 6, \cdots, 20\}$ |
| Regularization parameter | $\lambda$ (for multiple values) | $\lambda \in \{0, 1, 2, \cdots, 10\}$ |

## 5.3 Channel Condition Prediction Results

Using our training data set, we first trained our neural network for different values of $U_2$ and with different $\lambda$ as summarized in Table 5.1, with fixed $n = 3$, $L = 3$, and $C = 5$. Overall, we conducted neural network training for analysis for 187 different cases. Among all these different neural network formulations, we achieve highest training accuracy with $U_2 = 19$ and $\lambda = 1$. The obtained matrix of parameters $\Theta^{(1)}$ that controls mapping from layer 1 to layer 2 is of dimension $18 \times 4$ and is given as

$$\Theta^{(1)} = \begin{bmatrix} 0.083603 & 1.3069 & -1.3507 & -0.31406 \\ -7.2511 & 0.83695 & -1.3552 & -0.029151 \\ -0.17891 & 0.02127 & -1.0001 & 0.10472 \\ 0.45291 & -1.2242 & 2.3033 & -0.13007 \\ -0.067605 & 0.88882 & -1.4872 & 0.17819 \\ -9.6341 & 0.79873 & -1.9879 & -0.030151 \\ 6.0304 & -0.72619 & 1.6612 & 0.027693 \\ -0.17823 & 0.093676 & 0.85516 & 0.041489 \\ -0.36722 & 0.051129 & -0.50701 & -0.068896 \\ 0.81317 & -1.0042 & 3.4173 & 0.2269 \\ -0.36227 & 0.7304 & -1.5379 & -0.2473 \\ -5.1255 & 0.40031 & -2.0819 & -0.015761 \\ 0.24244 & 1.1364 & -1.1595 & -0.22285 \\ 0.089431 & -0.84213 & 0.5323 & -0.10267 \\ 0.46715 & 1.0859 & -3.7471 & 0.17632 \\ 6.8336 & -0.40881 & 1.6452 & 0.01649 \\ -0.054916 & 0.041571 & -0.0095006 & 0.079679 \\ 0.10552 & 1.1424 & -1.4058 & -0.32295 \end{bmatrix}, \qquad (5.17)$$

and $\Theta^{(2)}$ that controls mapping from layer 2 to layer 3 is of dimension $5 \times 19$. $(\Theta^{(2)})^T$ is given as

$$(\Theta^{(2)})^T = \begin{bmatrix} 0.64081 & -1.9222 & -2.0818 & -3.5496 & -2.9174 \\ 0.63864 & -1.712 & -1.7382 & 1.541 & 0.17406 \\ -7.3261 & 7.7397 & 9.4587 & 2.1935 & -1.3466 \\ -1.7116 & -1.7265 & -1.6584 & -1.9933 & 2.9853 \\ 0.45519 & 3.2046 & 1.3016 & -4.446 & -2.2666 \\ 0.26591 & -1.7694 & -0.70944 & -2.0909 & 1.7505 \\ -3.2625 & -8.2508 & -6.7571 & 11.279 & -0.99235 \\ 1.9181 & 3.9097 & -4.989 & -2.0265 & -4.1784 \\ -1.7953 & -2.4094 & -1.6825 & -1.0258 & 1.1769 \\ 0.63543 & -1.715 & -2.0152 & -2.9939 & 1.137 \\ 1.06 & 0.064952 & 0.23846 & 1.6145 & -3.7566 \\ -0.10419 & -1.5828 & -2.0229 & -1.9115 & 0.80917 \\ -0.68982 & -1.453 & -2.726 & -6.057 & 3.926 \\ -1.1941 & -2.4313 & 1.0541 & 2.1333 & -0.41928 \\ 2.3765 & 0.027802 & -2.4623 & -1.7732 & -1.8766 \\ 0.28496 & 0.86435 & -1.8023 & -4.8672 & 3.7188 \\ 1.653 & -0.075019 & 1.0833 & 4.9304 & -7.8526 \\ 1.5582 & -0.41236 & -0.55314 & -0.75314 & -1.8731 \\ -0.23315 & 0.22887 & -0.39377 & -3.144 & 1.1319 \end{bmatrix}. \tag{5.18}$$

We now evaluate the accuracy of the developed neural networks based prediction model, defined by $\Theta^{(1)}$ and $\Theta^{(2)}$, to predict channel bit-SNR values 20 mins into the future for all the communication passes in our test data set. The prediction model is run once for each communication pass of the data set, percentage of correct

Table 5.2: Channel bit-SNR estimation accuracy (in %) achieved with the neural networks based prediction model.

(a) Communication pass on September 6, 2005 data.

| % of SNRs | Ranges | Predicted SNR range | | | | |
|---|---|---|---|---|---|---|
| | | Range A | Range B | Range C | Range D | Range E |
| 0.000 | Range A | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range B | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range C | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range D | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 100.000 | Range E | 0.000 | 0.000 | 0.000 | 0.000 | 100.000 |
| % of predicted SNR | | 0.000 | 0.000 | 0.000 | 0.000 | 100.000 |
| Percentage of correct estimation: 100.000 | | | | | | |

(b) Communication pass on June 21, 2009 data.

| % of SNRs | Ranges | Predicted SNR range | | | | |
|---|---|---|---|---|---|---|
| | | Range A | Range B | Range C | Range D | Range E |
| 0.000 | Range A | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range B | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range C | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 12.181 | Range D | 0.000 | 0.000 | 0.000 | 72.425 | 27.575 |
| 87.819 | Range E | 0.000 | 0.000 | 0.000 | 0.817 | 99.183 |
| % of predicted SNR | | 0.000 | 0.000 | 0.000 | 9.540 | 90.460 |
| Percentage of correct estimation: 95.924 | | | | | | |

(c) Communication pass on March 26, 2009 data.

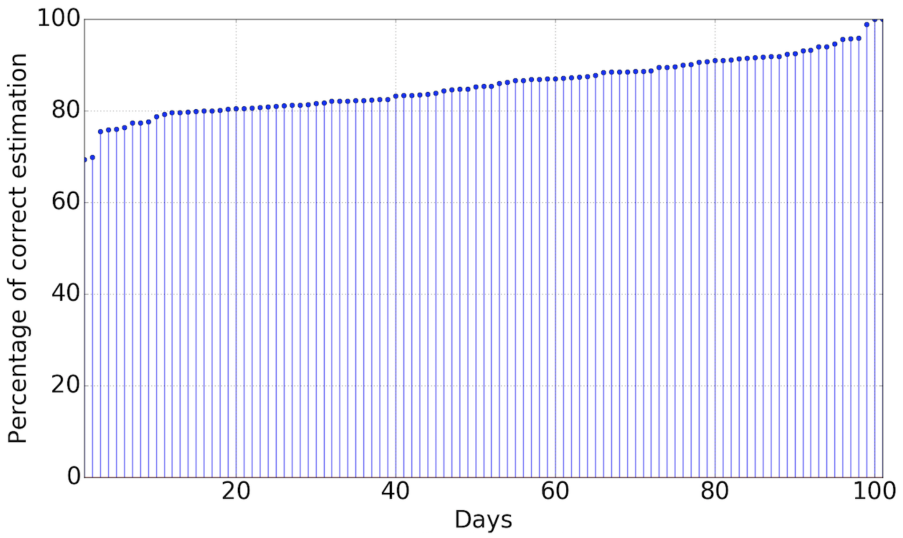
| % of SNRs | Ranges | Predicted SNR range | | | | |
|---|---|---|---|---|---|---|
| | | Range A | Range B | Range C | Range D | Range E |
| 0.000 | Range A | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 39.008 | Range B | 0.000 | 0.000 | 0.000 | 100.000 | 0.000 |
| 60.992 | Range C | 0.000 | 0.000 | 0.000 | 100.000 | 0.000 |
| 0.000 | Range D | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 0.000 | Range E | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| % of predicted SNR | | 10.700 | 9.184 | 10.625 | 35.321 | 34.169 |
| Percentage of correct estimation: 69.390 | | | | | | |

range estimation for each of them is obtained with RTT = 20 minutes. The Table 5.2 demonstrates the % of estimation accuracy in individual bit-SNR ranges and overall estimation accuracy during three test communication passes. For different communication passes, different percentage of correct estimation is observed, with maximum and minimum percentage of correct estimation being 100% and 69.390% respectively, as shown in Table 5.2a and Table 5.2c.

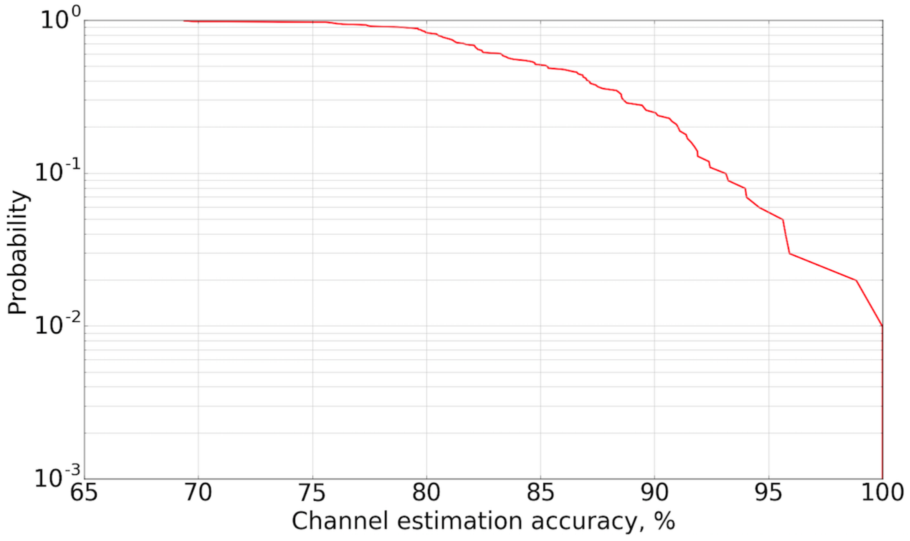
Percentage of correct estimation achieved during each communication pass when RTT = 20 minutes are arranged in increasing order and plotted as shown in Figure 5.3a. Among these 101 test communication passes, on 2 passes, our prediction accuracy is less than 70% with least being 69.390%. For 99 passes, we have prediction accuracy more than 70%, and in turn, prediction accuracy is more than 80% for 84 communication passes. Figure 5.3b shows corresponding complementary CDF plot of percentage of correct estimation of all the test passes. From the plot we can see that for more than 84% of the passes we can predict channel conditions with more than 80% accuracy. Similar to that of logistic regression, we also computed mean and variance of the percentage of correct estimation of all these passes. The mean of percentage of correct estimation is found to be 85.568 % and its variance is 36.328 %. Compared to the logistic regression, we have higher mean and smaller variance value signifying more accurate predictions and higher consistency of prediction accuracy. Similar to that of the logistic regression based prediction model, prediction accuracy is better for smaller RTT scenarios and it gradually decreases as the RTT between transmitter and receiver increases.

## 5.4 Conclusion

In this chapter, a dense neural network based future channel bit-SNR range prediction algorithm is presented. We obtained weight matrix $\Theta^{(1)}$ and $\Theta^{(2)}$ for 3



(a) Percentage of correct channel bit-SNR estimation.



(b) Complementary CDF of percentage of correct estimation.

Figure 5.3: Plots obtained using a neural networks based channel bit-SNR range prediction model for our test dataset.

input features, 3 network layers, and 5 output classes with $\lambda \in \{0, 1, 2, \cdots, 10\}$ and $U_2 \in \{4, 5, \cdots, 20\}$. Among all these cases, we found that the highest prediction accuracy is obtained when we have 19 nodes in the hidden layer and $\lambda = 1$. Estimation accuracy achieved with the neural network is higher than that achieved with logistic regression model for our test set containing 101 communication passes.

# CHAPTER 6

# CONCLUSION

In this chapter, we conclude the work presented in this dissertation. We evaluated the use of RaptorQ protocol at the application layer and a dynamic turbo coding system at the data link layer along with the implementation of real-time commands for deep space communication channels to achieve adaptive data transmission rate. We presented a DCSM protocol characterized by three key components: RaptorQ codes, turbo codes, and a channel bit-SNR prediction model that uses apriori information and is based on the idea of first-order auto-regressive process AR(1). We argued that in order to implement real-time commands, there is no need to develop a weather prediction model or predict the overall weather condition of an area. We can achieve that by using a future channel bit-SNR prediction model instead. We also presented a brief analysis of RaptorQ and turbo based data loss mitigation approach that could be employed in the spacecraft to guarantee successful delivery of a file and minimize the need for retransmission of lost symbols. We also evaluated the theoretical upper bound of turbo based dynamic code rate selection method and demonstrated that use of DCSM is very beneficial for efficient space communication. We also concluded that serial file transmission policy, along with a mechanism to inform the transmitter about the number of lost symbols plays a major role in the overall performance of any content transfer approach.

Performance of the DCSM method directly depends on the accuracy of the channel condition prediction mechanism used. Considering the fact that we have enough past measurement data available for conducting some data analysis, using state-of-the-art data analysis tools to develop strong channel condition prediction models seemed like a very appealing idea. Hence, we developed and analyzed a logistic regression and a neural networks based channel condition prediction models and presented their results. Comparatively evaluating the prediction accuracy achieved using these three prediction models, the AR(1) based approach is the best, the neural networks based approach comes in second followed by the logistic regression based model. The AR(1) based prediction algorithm is sturdy enough to perform under various RTT scenarios and is capable of predicting with very high accuracy, however, the logistic regression and neural networks based prediction algorithm have variable performances with different RTT scenarios. The strength of the AR(1) based approach comes from the fact that we utilize apriori information about the upcoming pass to begin the prediction process. However, the neural networks and the logistic regression based model use only long-term historical data, which fails to take advantage of predictable channel behavior.

# BIBLIOGRAPHY

# BIBLIOGRAPHY

[1] Jet Propulsion Laboratory. *DSN Telecommunications Link Design Handbook*. Jet Propulsion Laboratory, 2000.

[2] CCSDS Secretariat. Proximity-1 Space Link Protocol - Data Link Layer, December 2013.

[3] S. Shambayati, F. Davarian, and D. Morabito. Link design and planning for Mars Reconnaissance Orbiter (MRO) Ka-band (32 GHz) telecom demonstration. In *2005 IEEE Aerospace Conference*, pages 1559–1569, March 2005.

[4] S. Shambayati, J. S. Border, D. D. Morabito, and R. Mendoza. MRO Ka-band Demonstration: Cruise Phase Lessons Learned. In *2007 IEEE Aerospace Conference*, pages 1–17, March 2007.

[5] Jim Taylor, Dennis K. Lee, and Shervin Shambayati. Mars Reconnaissance Orbiter Telecommunications. *DESCANSO Design and Performance Summary Series*, September 2006.

[6] Jim Taylor, editor. *Deep Space Communications*. DEEP SPACE COMMUNICATIONS AND NAVIGATION SERIES. DESCANSO, 2014.

[7] Timothy Pham Jeff Berner. Deep Space Network Services Catalog. DSN No. 820-100, February 2015.

[8] J. N. Maki, J. F. Bell, K. E. Herkenhoff, S. W. Squyres, A. Kiely, M. Klimesh, M. Schwochert, T. Litwin, R. Willson, A. Johnson, M. Maimone, E. Baumgartner, A. Collins, M. Wadsworth, S. T. Elliot, A. Dingizian, D. Brown, E. C. Hagerott, L. Scherr, R. Deen, D. Alexander, and J. Lorre. Mars Exploration Rover Engineering Cameras. *Journal of Geophysical Research: Planets*, 108(E12):n/a–n/a, 2003. 8071.

[9] A. Kiely and M. Klimesh. The ICER progressive wavelet image compressor. In *IPN Progress Report*, pages 1–46, 2003.

[10] N. Rahnavard, B. N. Vellambi, and F. Fekri. Rateless Codes With Unequal Error Protection Property. *IEEE Transactions on Information Theory*, 53(4):1521–1532, April 2007.

[11] S. S. Arslan, P. C. Cosman, and L. B. Milstein. Generalized Unequal Error Protection LT Codes for Progressive Data Transmission. *IEEE Transactions on Image Processing*, 21(8):3586–3597, Aug 2012.

[12] F. S. Marques, C. Schwartz, M. S. Pinho, and W. A. Finamore. Designing an efficient LT-code with unequal error protection for image transmission. In Lorenzo Bruzzone, editor, *SPIE remote sensing*, page 96431H, oct 2015.

[13] NASA JPL. *Deep Space Telecommunications Systems Engineering.* JPL Publication 82-76, 1983.

[14] Tien M. Nguyen Warren L. Martin. CCSDS - SFCG Efficient Modulation Mmethods Study A Comparison of Modulation Schemes Phase 1: Bandwidth Uuilization, September 1993.

[15] A. Shokrollahi. Raptor codes. *IEEE Transactions on Information Theory*, 52(6):2551–2567, June 2006.

[16] Amin Shokrollahi and Michael Luby. Raptor Codes. *Foundations and Trends® in Communications and Information Theory*, 6(3–4):213–322, 2011.

[17] C. Berrou and A. Glavieux. Near optimum error correcting coding and decoding: turbo-codes. *IEEE Transactions on Communications*, 44(10):1261–1271, Oct 1996.

[18] I. J. Schoenberg. *I. J. Schoenberg Selected Papers*, chapter Contributions to the Problem of Approximation of Equidistant Data by Analytic Functions, pages 3–57. Birkhäuser Boston, Boston, MA, 1988.

[19] M. Unser. Splines: a perfect fit for signal and image processing. *IEEE Signal Processing Magazine*, 16(6):22–38, Nov 1999.

[20] The Scipy community. Interpolation, June 2017.

[21] David Morabito, Longtao Wu, and Stephen Slobin. Weather Forecasting for Ka-band Operations: Initial Study Results. Technical report, Jet Propulsion Laboratory, August 2016.

[22] S. J. Keihm, Y. Bar-Sever, and J. Liljegren. Water Vapor Radiometer - Global Positioning System Comparison, Measurements and Calibration of the 20 to 32 Gigahertz Tropospheric Water Vapor Absorption Model. Technical report, Jet Propulsion Laboratory, February 2001.

[23] G. M. Resch. Inversion algorithms for water vapor radiometers operating at 20.7 and 31.4 GHz. In E. C. Posner, editor, *The Telecommunications and Data Acquisition Report*, February 1984.

[24] Shervin Shambayati. Atmosphere Attenuation and Noise Temperature at Microwave Frequencies. In Macgregor S. Reid, editor, *Low-Noise Systems in the Deep Space Network*, chapter 6, pages 255 – 281. Deep Space Communications And Navigation Series, 2008.

[25] Andrea Barbieri Andre Makovsky and Ramona Tung. Odyssey Telecommunications. *DESCANSO Design and Performance Summary Series*, October 2002.

[26] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding: Turbo-codes. In *Communications, 1993. ICC '93 Geneva. Technical Program, Conference Record, IEEE International Conference on*, volume 2, pages 1064–1070 vol.2, May 1993.

[27] CCSDS 130.1-G-2. TM Synchronization and Channel Coding - Summary of Concept and Rationale. Informational report, CCSDS, November 2012.

[28] CCSDS 131.0-B-2. TM Synchronization and Channel Coding. Informational report, CCSDS, August 2011.

[29] Ian F. Akyildiz, Özgür B. Akan, Chao Chen, Jian Fang, and Weilian Su. InterPlaNetary Internet: State-of-the-art and Research Challenges. *Computer Networks*, 43(2):75–112, October 2003.

[30] Mark D. Johnston, Daniel Tran, Belinda Arroyo, Sugi Sorensen, Peter Tay, Butch Carruth, Adam Coffman, and Mike Wallace. Automating Mid- and Long-Range Scheduling for NASA's Deep Space Network. SpaceOps 2012, June 2012.

[31] Richard A. Davis Peter J. Brockwell. *Introduction to Time Series and Forecasting*. Springer, second edition, 2002.

[32] Gwilym M. Jenkins George E. P. Box and Gregory C. Reinsel. *Time Series Analysis, Forecasting and Control*. Wiley Series in Probability and Statistics. Wiley, fourth edition, 2008.

[33] Thomas Mikosch. *Elementary Stochastic Calculus with Finance in View*, volume 6. World Scientific Publishing Co. Pte. Ltd., 1998.

[34] G. E. Uhlenbeck and L. S. Ornstein. On the Theory of the Brownian Motion. *Phys. Rev.*, 36:823–841, Sep 1930.

[35] Enrico Bibbona, Gianna Panfilo, and Patrizia Tavella. The Ornstein - Uhlenbeck process as a model of a low pass filtered white noise. *Metrologia*, 45(6):S117, 2008.

[36] D. R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society*, 20(2):215 – 242, 1958.

[37] Warren S. McCulloch and Walter Pitts. A logical calculus of the idea immanent in nervous activity. *Bulletin of Mathematical Biology*, 5(10.1007/BF02478259):115–133, 1943.

[38] D. O. Hebb. *The Organization of Behavior*. John Wiley & Sons, New York, 1949.

[39] Frank Rosenblatt. *Principles of neurodynamics; perceptrons and the theory of brain mechanisms*. Spartan Books, 1962.

[40] R. Lippmann. An introduction to computing with neural nets. *IEEE ASSP Magazine*, 4(2):4–22, Apr 1987.

[41] R. Willett. Sampling Theory and Spline Interpolation. Connexions Web site, 2003, September 24.

APPENDICES

# APPENDIX A

# PARTIAL AUTOCORRELATION FUNCTION

A time series is a set of values observed sequentially through time. The *correlation* $\mathrm{R}(\mathbf{X}, \mathbf{Y})$ between two time-series $\mathbf{X} = \{X_t : t \in T\}$ and $\mathbf{Y} = \{Y_t : t \in T\}$, where $t$ refers to the time period, $T = \{1, 2 \cdots, n\}$ represents the index set, and $X_t$ and $Y_t$ refers to the value of the time-series $\mathbf{X}$ and $\mathbf{Y}$ at time $t$, respectively, is defined as

$$\rho = \mathrm{R}(\mathbf{X}, \mathbf{Y}) = \frac{E\left[\left(\mathbf{X} - E\left[\mathbf{X}\right]\right)\left(\mathbf{Y} - E\left[\mathbf{Y}\right]\right)\right]}{\sigma_{\mathbf{X}}\,\sigma_{\mathbf{Y}}} = \frac{\mathrm{Cov}\left(\mathbf{X},\, \mathbf{Y}\right)}{\sigma_{\mathbf{X}}\,\sigma_{\mathbf{Y}}}, \tag{A.1}$$

where $E$ is the expectation operator, $E[\mathbf{X}]$ and $E[\mathbf{Y}]$ are the means respectively for $\mathbf{X}$ and $\mathbf{Y}$ and $\sigma_{\mathbf{X}}$, $\sigma_{\mathbf{Y}}$ are their standard deviations.

*Autocorrelation* of a time series $\mathbf{X}$ is the correlation between the original series and its lagged version. Autocorrelation coefficient $\rho_k$ of a lag $k$ is computed using the following expression

$$\rho_k = \mathrm{R}\left(X_t, X_{t-k}\right) = \frac{\frac{1}{n-k}\sum_{t=k+1}^{n}\left(X_t - E\left[\mathbf{X}\right]\right)\left(X_{t-k} - E\left[\mathbf{X}\right]\right)}{\sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(X_t - E\left[\mathbf{X}\right]\right)^2}\sqrt{\frac{1}{n-k}\sum_{t=k+1}^{n}\left(X_{t-k} - E\left[\mathbf{X}\right]\right)^2}}, \tag{A.2}$$

For any time series $\mathbf{X}$, the *partial autocorrelation* at lag $k$, denoted by $\alpha_{\mathbf{X}}(k)$, is the autocorrelation between $X_t$ and $X_{t-k}$ with the linear dependence of $X_t$ on $X_{t-1}$ through $X_{t-k+1}$ removed. *Partial autocorrelation function* of $\mathbf{X}$ is constructed by calculating the partial correlation between $X_t$ and $X_{t-1}$, $X_t$ and $X_{t-2}$, and so on, statistically adjusting out the influence of intermediate lags. For example, the partial autocorrelation of lag four is the partial correlation between $X_t$ and $X_{t-4}$ after statistically removing the influence of $X_{t-1}$, $X_{t-2}$, and $X_{t-3}$ from both $X_t$ and $X_{t-4}$. The partial autocorrelation function of $\mathbf{X}$ is defined by [31], [32]

$$\alpha_{\mathbf{X}}(k) = \pi_{k,k} = \mathrm{R}\left(\left(X_t - E\left[X_t | X_{t-1}, \cdots, X_{t-k+1}\right]\right), \left(X_{t-k} - E\left[X_{t-k} | X_{t-1}, \cdots, X_{t-k+1}\right]\right)\right), \tag{A.3}$$

where $k = 0, 1, 2, \cdots$. The function consists of the sequence $\pi_{0,0}, \pi_{1,1}, \pi_{2,2}, \pi_{3,3}, \cdots$. At no lag, i.e., $k = 0$, both the autocorrelation coefficient $\rho_0$ and partial autocorrelation coefficient $\pi_{0,0}$ is 1, that is,

$$\pi_{0,0} = \rho_0 = 1.$$

At the first lag, i.e., $k = 1$, the autocorrelation coefficient $\rho_1$ and partial autocorrelation coefficient $\pi_{1,1}$ are the same

$$\pi_{1,1} = \rho_1.$$

Partial autocorrelation coefficient of lag $k > 1$ is computed by using Durbin-Levinson's recursive formula

$$\pi_{k,j} = \pi_{k-1,j} - \pi_{k,k}\, \pi_{k-1,k-j}, \tag{A.4}$$

$$\pi_{k,k} = \frac{\rho_k - \sum_{j=1}^{k-1} \pi_{k-1,j}\, \rho_{k-j}}{1 - \sum_{j=1}^{k-1} \pi_{k-1,j}\, \rho_{k-j}}. \tag{A.5}$$

The PACF is very useful in identifying an autoregressive (AR) process. If our original process is autoregressive of order $p$ represented as AR(p), then for $k > p$, we should have $\pi_{k,k} = 0$. This provides a very useful test for whether or not a process is autoregressive. A series can be represented as a pure AR process, if we observe

1. the autocorrelation function dies out in an exponential or sinusoidal fashion,

and

2. the partial autocorrelation cuts off after lag $p$.

# APPENDIX B

# SAMPLING THEORY AND SPLINE INTERPOLATION

In this appendix, Shannon's classical sampling theory is compared to digital to analog signal reconstruction using spline interpolation [41]. In the spline method, the signal is reconstructed using sample-weighted cardinal splines [18], [19] as opposed to sample-weighted sinc functions.

Shannon's sampling theory tells that if we have a band limited signal $s(x)$ that has been sampled at the Nyquist rate, then the signal can be reconstructed from its discrete samples $s\,[k]$ with the following relation

$$s(x) = \sum_{k \in \mathbb{Z}} s\,[k]\,\mathrm{sinc}(x-k) \tag{B.1}$$

There are several desirable properties of the sinc function that make this strategy effective. First of all, the sinc function vanishes at all integers except at the origin. Secondly, $\mathrm{sinc}(0) = 1$. As a result, if $F_s$ is the sampling frequency, then $s(xF_s) = s\,[k]$. The disadvantage of this approach is that it depends on the initial assumption that the signal is band limited, but frequently we rely on only a finite number of samples, which cannot completely describe a band limited signal. As a result, we can only find an approximate estimate of the signal $s(x)$.

As described above, having only a finite number of samples leads to inaccuracies in estimating $s(x)$. Using cardinal splines instead of sinc functions can lessen the magnitude of the errors. The $n^{\mathrm{th}}$ cardinal spline, $\eta^n$, gives piecewise polynomial interpolation with order $n$ polynomials. Like the sinc function, each cardinal spline vanishes at all integers except the origin, and $\eta^n(0) = 1$. Furthermore $\lim_{n\to\infty}\eta^n(x) = \text{sinc}(x)$. This means that cardinal splines can be used for signal reconstruction from samples just as sinc functions are used. Specifically,

$$s(x) = \sum_{k\in\mathbb{Z}} s\,[k]\ \eta^n\ \text{sinc}(x-k). \tag{B.2}$$

The spline interpolation is found to be more smoother than the sinc interpolation. This is because the support of the cardinal splines is more compact than that of the sinc function. In fact, to compute the value of $s(x)$ (when $s(x)$ is a polynomial signal) with an error of less than 1%, one would need $\mathcal{O}(100)$ sinc functions, but just $n+1$ basis-splines for exact evaluation.

# VITA

Rojina Adhikary obtained a Bachelor of Engineering degree in Electronics and Communication Engineering from Pulchowk Campus, Institute of Engineering, Tribhuvan University, Nepal in the year 2010.

She joined the University of Mississippi in January 2011 and received M.S. in engineering science degree with an emphasis in Telecommunications from the Department of Electrical Engineering in Spring 2014. She is currently pursuing her Ph.D. at the University of Mississippi. During her master's and Ph.D. studies, she has worked as a teaching assistant and research assistant at the Department of Electrical Engineering. Her research interest includes localization of wireless sensor networks, rate-less codes, wireless communications, and deep space communications. She is a student member of Institute of Electrical and Electronics Engineers (IEEE).