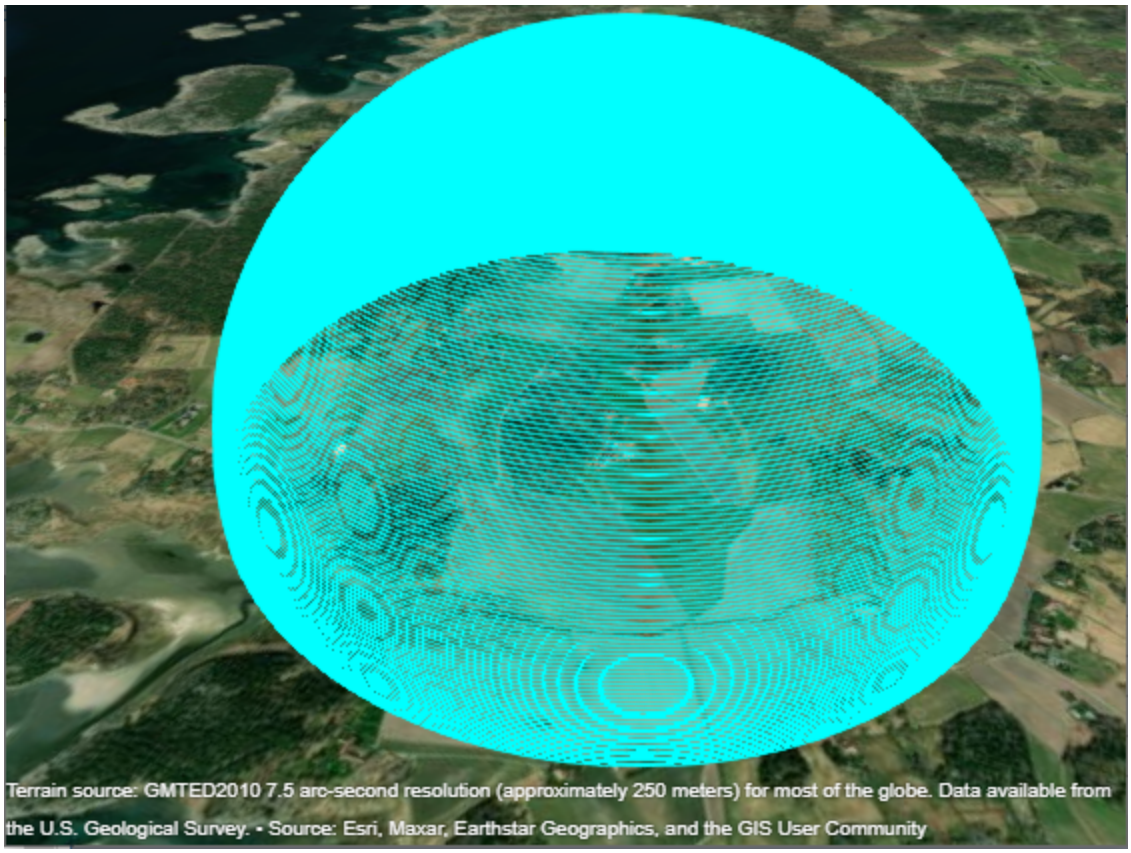


LOS/Multipath/NLOS Classifiers using Machine learning and Raytracing

A preliminary study to identify and address the Mulitpath error

Master's thesis in Complex Adaptive System

KARTHIK UPENDRA

DEPARTMENT OF SPACE, EARTH and ENVIRONMENT
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

Master's thesis 2024

LOS/Multipath/NLOS Classifiers using Machine learning and Raytracing

A preliminary study to identify and address the Mulitpath error

KARTHIK UPENDRA

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Space Earth and Environment
Division of Onsala Space Observatory
Research Unit of Space Geodesy and Geodynamics
Chalmers University of Technology
Gothenburg, Sweden 2024

LOS/Multipath/NLOS Classifiers using Machine learning and Raytracing
A preliminary study to identify and address the Mulitpath error
KARTHIK UPENDRA

Supervisor: Jan Johansson, Department of Space Earth and Environment
Examiner: Jan Johansson, Department of Space Earth and Environment

Master's Thesis 2024
Jan Johansson, Department of Space Earth and Environment
Division of Onsala Space Observatory
Research Unit of Space Geodesy and Geodynamics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Discritised Azimuth-Elevation space around the receiver. Diagram not to scale.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2024

LOS/Multipath/NLOS Classifiers using Machine learning and Raytracing
A preliminary study to identify and address the Mulitpath error
KARTHIK UPENDRA
Department of Space Earth and Environment
Chalmers University of Technology

Abstract

This thesis explores the application of machine learning algorithms to address one of the challenges in localization in GNSS called the Multipath errors. The approach involves data collection via a receiver mounted on an excavator which is placed in front of the building, which acts as one of the sources for mulitpath error. In order to perform supervised machine learning, wireless communication tool box within Matlab is used for Raytracing simulation to label the data.

Drawing inspiration from existing literature, we introduce a novel feature, the 'difference in range acceleration between code and carrier signals,' which exhibits promising distribution and metrics when analyzed with respect to ray-tracing results.

The support vector machine (SVM) achieves an average class-wise recall and precision of approximately 75% on the recorded measurments. Additionally, we explore the use of an Autoencoder with a 1D-CNN layer to extract new features aimed at enhancing classification performance. By conducting three different simulations with varying data sorting methods, we demonstrate how sorting the data and the machine learning algorithm can influence the learned features which in turn impacts classification performance.

Lastly, in order to take advantage of the insights gained from different sorting methods, we transform the problem from a Multiclass to Multi-label-Multi-output classification problem, wherein we utilize a deep neural network architecture with GRU units to classify the signals. Although complicated in terms of data restructuring and handling, the network demonstrated robust performance, achieving more than 85% average class-wise recall and precision and exceeding 96% for signals labeled LOS.

Keywords: Machine learning, Multipath error, GNSS, Convolutional neural network (CNN), Recurrent neural network (RNN), Gated recurrent unit (GRU), 1D CNN, Autoencoders, Raytracing, Deep neural network (DNN).

Acknowledgements

I express my sincere thanks to Professor Jan Johansson for his invaluable guidance and belief in my capabilities throughout this thesis. Special appreciation to my family for their unwavering support. I also acknowledge ChatGPT for its role in structuring and refining my thoughts, contributing to the clarity of this work. Additionally, I appreciate the prompt assistance from the Matlab team to provide a workaround for the bug related to the raytracing simulation.

Karthik Upendra, Gothenburg, March 2024

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

CNR	Carrier-to-Noise Ratio
CNN	Convolutional Neural Network
ECOC	Error Correcting Output Codes
GLONASS	Globalnaya Navigazionnaya Sputnikovaya Sistema
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GRU	Gated Recurrent Unit
IGS	International GNSS Service
JOSM	Java OpenStreetMap Editor
KNN	K-Nearest Neighbors
LOS	Line of Sight
NavIC	Navigation with Indian Constellation
NLOS	Non-Line of Sight
NRTK	Network-Based Real-Time Kinematic Positioning
PNT	Positioning, Navigation, and Timing
PPP	Precise Point Positioning
PVT	Position, Velocity, and Time
QZSS	Quasi-Zenith Satellite System
RBF	Radial Basis Function
RINEX	Receiver INdependent EXchange
RMSE	Root Mean Squared Error
SAE	Sparse Auto-Encoder
SBAS	Satellite Based Augmentation System
SBR	Shooting Bouncing Rays
SMOTE	Synthetic Minority Over-sampling Technique
SVM	Support Vector Machine
SSI	Signal Strength Indicator

Contents

List of Acronyms **ix**

List of Figures **xv**

List of Tables **xxi**

1 Introduction **1**
1.1 Project . . . 2
1.2 Problem definition . . . 2
1.3 Objective . . . 2
1.4 Method . . . 3
1.5 Limitations . . . 3
1.6 Thesis outline . . . 3

2 Literature review **5**

3 Hardware **9**

4 Measurements **11**
4.1 Placement and location of the receiver . . . 11
4.2 GPS Measurements . . . 13
4.3 Galileo Measurements . . . 16

5 Post processing using GipsyX **21**
5.1 General statistics about the dataset . . . 22
5.2 Distribution of residuals with respect to raytracing results . . . 23
5.3 Comparison of Residuals with Ray Tracing Labels . . . 25

6 Ray tracing **27**
6.1 Environment model . . . 28
6.2 Labels and classes . . . 28
6.3 Discritized space in Azimuth-elevation domain . . . 30
6.4 Limitation but not as significant as initially thought . . . 31
6.5 Labeling for discretized azimuth-elevation space . . . 33
6.6 Labeling the observations . . . 35

7 Feature analysis and engineering with respect to Classes **39**
7.1 Range-velocity . . . 39

7.2 Range-acceleration . . . 40
7.3 Differencing of range-velocity between code and carrier signals . . . 42
7.4 Differencing of range-acceleration between code and carrier signals . . 43
7.5 Indicator variable . . . 49
7.6 Signal strength indicator . . . 49
7.7 Difference of Signal strength indicator . . . 50

8 Support vector machine 53
8.1 Dataset . . . 53
8.1.1 Cleaning the Data set . . . 54
8.1.2 Handling imbalanced Data set . . . 54
8.2 Feature selection . . . 55
8.3 SVM for multiclass classification . . . 56
8.4 k-Fold cross validation . . . 57
8.5 Result . . . 57

9 CNN based Auto encoders with SVM for Multi-class classification 61
9.1 Dataset and features . . . 61
9.2 Autoencoder structure . . . 62
9.3 Svm for classification . . . 62
9.4 Results . . . 63
9.5 Conclusion . . . 65

10 Overview of different classification problems 67
10.1 Binary classification . . . 67
10.2 Multi class classification . . . 68
10.3 Multi label classification . . . 68
10.4 Multi-Label Multi-Output Classification . . . 69
10.5 Conclusion . . . 69

11 Multi-Label-Multi-output classification using DNN with GRU units 71
11.1 Data Processing . . . 72
11.1.1 Restructuring and cleaning the data . . . 72
11.1.2 Error-Correcting Output Codes (ECOC) encoding . . . 72
11.2 Imbalance in dataset . . . 73
11.2.1 Heuristic approach to handle imbalance dataset . . . 74
11.2.2 Standardisation of the data . . . 76
11.3 GRU neural network structure . . . 76
11.4 Result . . . 76

12 Conclusion 79

Bibliography 83

A Appendix A I

B Appendix B V
B.1 Dense layer based Autoencoders with SVM for Multi-class classification V

B.2 Dataset and features V
B.3 Autoencoder structure V
B.4 Svm for classification V
B.5 Results VI

C Appendix C **IX**

List of Figures

3.1 Images of the receiver (a) and antenna (b) used in the GNSS measurements. 9

4.1 Placement of the Receiver on the excavator and its surrounding environment. Image (a) provides an overview of the excavator and its surroundings. To the south of the excavator is the building, whereas the terrain is situated to the west highlighted in yellow. Towards the north-west direction, there is a tree (highlighted in red), and to the east lies an open field. Image (b) shows the relative positions of the building and tree in relation to the excavator, with the terrain also visible. Image (c) illustrates the receiver's placement on the top of the excavator. Finally, image (d) shows the open field located to the east of the excavator. 12

4.2 Range measurements from all the tracked GPS satellites on C1C (left figure) and L2W (right figure) channels on 4th September. The frequency of the measurements are 1sec. The color bar indicates the signal strength indicator. 13

4.3 Polar plot of satellite trajectories and range measurements, from all the tracked GPS satellites, on all days, on C1C (left) and L2W (right) channels. The frequency of the measurements are 1sec. 14

4.4 Polar plot of satellite trajectories and signal strength indicator(SSI), from all the tracked GPS satellites, on all days, on C1C (left) and L2W (right) channels. The frequency of the measurements are 1sec. . 15

4.5 Density of measurements at different Azimuth-elevation for GPS constellation. It shows more data are collected at low elevation within approximately 300 to 60 deg Azimuth. 16

4.6 Range measurements from all the tracked Galileo satellites on C1C (left figure) and L7Q (right figure) channels from 4th September.The frequency of the measurements are 1sec. The color indicates the signal strength indicator 17

4.7 Polar plot of satellite trajectories and range measurements, from all the tracked Galileo satellites, on all days, on C1C (left) and L7Q (right) channels. The frequency of the measurements are 1sec. Color indicates the range 18

4.8 Polar plot of satellite trajectories and range measurements, from all the tracked Galileo satellites, on all days, on C1C (left) and L7Q (right) channels. The frequency of the measurements are 1sec. Color indicates SSI 19

4.9 Density of measurements at different Azimuth-elevation for Galileo constellation. It shows more data are collected at low elevation within approximately 300 to 60 deg Azimuth 20

5.1 Total percentage of labels in residuals for each day, comparing GPS (top) and Galileo (bottom) constellations. Numeric values above each column indicate the total data points in residuals generated by GipsyX for corresponding days. 22

5.2 Distribution of residuals with respect to labels from raytracing for GPS constellation. In each subplot red curve indicates a Gaussian fit. 23

5.3 Residual distribution with respect to labels from raytracing for Galileo constellation. The red curve in each subplot represents a Gaussian fit. 24

5.4 Daily summary illustrating the mean and standard deviation of residuals with respect to labels for both GPS and Galileo constellations . . 25

5.5 Comparison of residuals with ray tracing labels for GPS and Galileo constellations. Fist and second row represents GPS and Galileo constellations, respectively. first column of figures displays the residuals without outliers, while the last column exclusively shows the outliers. The middle column represents the labels obtained from ray tracing. . 26

6.1 Comparison of True and Rendered Environment in Matlab. Image (a) and (b) shows the top-view of the true and rendered environment around the excavator, respectively. Image(c) and (d) shows comparison of the true and rendered building, excavator and tree, respectively 28

6.2 Example of labels generated from ray tracing when the excavator is pointing north. Ideally, the blue icon in images represents the satellite location and the red icon represent the receiver location. But for better simulation performance these are swapped, meaning the blue and the red icon represents the receiver and transmitter, respectively. 29

6.3 Discritised Azimuth-Elevation space around the receiver with a fixed distance of 500km around the receiver. Diagram not to scale. 31

6.4 Schematic illustrating the initally perceived concept when compared to more realistic schematic 32

6.5 Errors introduced due to projecting the satellite closer to over come the limitations in Matlab tool box. It indicates that the apparent limitations initially perceived in labeling the multipath component are not significant. 33

6.6 Results from the raytracing simulation are presented in pairs: Images a and b, c and d, e and f, and g and h correspond to results from the simulations conducted with the excavator pointing north, east, south, and west, respectively. The first columns indicate whether the data point characterized by Azimuth and elevation is Line of Sight (LOS) or Non-Line of Sight (NLOS). Meanwhile, the second columns indicate whether the data point consists of a multipath component. The combination of these two criteria results in the labels LOS, LOS+Multipath, NLOS, and NLOS+Multipath. 34

6.7 Labels assigned to each time epoch in the recorded measurements for the GPS constellation, determined using the lookup table with azimuth and elevation values. The displayed results span from the 4th to the 7th of September, providing a focused representation as the excavator's orientation remains constant, resulting in similar labeling patterns as observed on the 7th of September. 36

6.8 Labels assigned to each time epoch in the recorded measurements for the Galileo constellation, utilizing the lookup table based on azimuth and elevation values. The presented results cover the period from the 4th to the 7th of September, simplifying the representation while maintaining consistency due to the unchanged orientation of the excavator 37

6.9 Total percentage of labels in residuals for each day, comparing GPS (top) and Galileo (bottom) constellations. Numeric values above each column indi- cate the total data points in residuals generated by GipsyX for corresponding days. 38

7.1 **Range-velocity and histogram** for the **C1C** channel in the GPS constellation (top panel) and the Galileo constellation (bottom panel). In the left set of figures, the blue curve represents Range-velocity on the left axis, while the orange curve represents the elevation on the right axis. The right set of figures corresponds to the histogram of Range-velocity from the left panel. 40

7.2 **Range-acceleration and histogram** for the **C1C** channel in the GPS constellation (top panel) and the Galileo constellation (bottom panel). In the left set of figures, the blue curve represents Range-acceleration on the left axis, while the orange curve represents the elevation on the right axis. The right set of figures corresponds to the histogram of Range-acceleration from the left panel. 41

7.3 **Range-acceleration and histogram** for the **L1C** channel in the GPS constellation (top panel) and the Galileo constellation (bottom panel). In the left set of figures, the blue curve represents Range-acceleration on the left axis, while the orange curve represents the elevation on the right axis. The right set of figures corresponds to the histogram of Range-acceleration from the left panel. 42

7.4 An illustration showing the effect of differencing **Range-velocity** on the distribution. The top and bottom panels of figures correspond to the distribution for the GPS and Galileo constellations, respectively. The first and second columns of figures represent the distribution of Range-velocity on the L1C and C1C channels, respectively. The last column represents the distribution after differencing these two Range-velocity. 43

7.5 An illustration showing the effect of differencing **Range-acceleration** on the distribution.The top and bottom panels of figures correspond to the distribution for the GPS and Galileo constellations, respectively. The first and second columns of figures represent the distribution of Range-acceleration on the L1C and C1C channels, respectively. The last column represents the distribution after differencing these two Range-acceleration. 44

7.6 GPS: Summary of statistics of differencing of Range-acceleration. Panel a and b shows the mean and standard deviation for each day for different class. In each panel the left and right side figure is for differencing of range acceleration between L1C-C1C and L2W-C2W, respectively. 46

7.7 Galileo: Summary of statistics of differencing of Range-acceleration. Panel a and b shows the mean and standard deviation for each day for different class. In each panel the left and right side figure is for differencing of range acceleration between L1C-C1C and L7Q-C7Q, respectively. 47

7.8 An illustration of the effect of differencing of Range-Acceleration in polar plot. Each row corresponds to different classes. The first and second column corresponds to Range-acceleration on L2W and C2W channel respectively for different classes. The last column corresponds the differencing of the first two columns. The colorbar in each plot corresponds to 3-Sigma region. The red marker in each plot corresponds to data points beyond 3-Sigma region. 48

7.9 SSI of L1C channel on the x-axis and C1C channel on the y-axis for GPS (left figure) and Galileo (right figure), respectively. The size of the bubbles represents the total number of data points in each class. . 50

7.10 The difference of SSI of L1C channel on the x-axis and C1C channel on the y-axis for GPS (left figure) and Galileo (right figure), respectively. The size of the bubbles represents the total number of data points in each class. 51

8.1 Total number of data points in each class across different days within the downsampled dataset. 54

8.2 Total number of data points in each class across different days in the balanced dataset 55

8.3 Averaged hinge loss across K-Fold where K=5 for 20 different iterations. 57

8.4 Confusion chart for training and test dataset. 58

8.5 True and predicted classes for training (top row) and for testing (bottom row) in polar plot. 59

9.1 Confusion chart for simulation with different neurons in the bottleneck layer. 64

11.1 Total number of data points in each class. Note: Class 3 consists of NLOS and NLOS+Multipath and class 4 represents satellite beyond horizon. 73
11.2 Total number of data points in each class. Note: Class 3 consists of NLOS and NLOS+Multipath and class 4 represents satellite beyond horizon. 74
11.3 3D-bubble chart showcasing proportion of classes in different label sets. 75
11.4 Confusion chart showing the results from DNN with GRU units. . . . 77

A.1 GPS: An illustration of the effect of differencing of Range-Acceleration in polar plot. Each row corresponds to different classes. The first and second column corresponds to Range-acceleration on L1C and C1C channel respectively for different classes. The last column corresponds the differencing of the first two columns. The colorbar in each plot corresponds to 3-Sigma region. The red marker in each plot corresponds to data points beyond 3-Sigma region II
A.2 Galileo: An illustration of the effect of differencing of Range-Acceleration in polar plot. Each row corresponds to different classes. The first and second column corresponds to Range-acceleration on L1C and C1C channel respectively for different classes. The last column corresponds the differencing of the first two columns. The colorbar in each plot corresponds to 3-Sigma region. The red marker in each plot corresponds to data points beyond 3-Sigma region III
A.3 Galileo: An illustration of the effect of differencing of Range-Acceleration in polar plot. Each row corresponds to different classes. The first and second column corresponds to Range-acceleration on L7Q and C7Q channel respectively for different classes. The last column corresponds the differencing of the first two columns. The colorbar in each plot corresponds to 3-Sigma region. The red marker in each plot corresponds to data points beyond 3-Sigma region IV

B.1 Confusion chart for predicting the classes from the learned representation of the Autoencoder with varying neurons in the bottleneck layer.VIII

C.1 Confusion chart for simulation using 3 features and an Autoencoder with 1D CNN. X

List of Tables

6.1 Truth Table for LOS, NLOS, and Multipath 30

8.1 Features used for the SVM. 56
8.2 Coding Matrix for One-Versus-One Classification 57

9.1 Description of Layers in the Autoencoder. 62
9.2 Training Progress Metrics 63

10.1 Typical input data for Binary Classification problem 67
10.2 Example of Multiclass Classification 68
10.3 Example of Multiclass Classification 69
10.4 Example of Multi-Label Multi-Output Classification 69
10.5 Overview of classification problems 70

11.1 Code book used to encode each classes 72
11.2 Description of Layers used in DNN. 76

B.1 Description of Layers in the Autoencoder. VI
B.2 Training Progress Metrics for all three Autoencoders, with sim1, sim2 and sim3 corresponds to training with 14, 7 and 3 neurons in the bottleneck layer. VI

C.1 Features used in this chapter for training the Autencoder with 1D CNN. IX
C.2 Training Progress Metrics IX

1

Introduction

Global Navigation Satellite System (GNSS) is a satellite-based navigation system that enables precise positioning, navigation, and timing (PNT) services. It has become an essential technology with widespread applications across various industries and continues to evolve to meet the growing demands of modern navigation and positioning requirements. The GNSS system uses a constellation of satellites in space, ground control stations, and mobile receivers to enable a wide range of applications and services.

The most well-known GNSS system is the Global Positioning System (GPS), developed during the early 1970s by the United States and declared fully operational in 1995. GLONASS constellation from Russia was also developed during the same time. There are other existing and emerging GNSS systems, including Galileo (European Union), BeiDou (China), QZSS (Japan), and NavIC (India). These systems operate independently and offer similar functionalities but with varying levels of coverage and accuracy [2].

The GNSS satellites transmit signals that are received by user-receivers on Earth. These receivers use the signals from multiple satellites to determine the user's precise position, velocity, and time. By trilaterating the signals from different satellites, the receiver can calculate its position with good accuracy.

GNSS technology has numerous applications in various sectors. In transportation, it is widely used for navigation in cars, ships, and aircraft, enabling accurate route planning, tracking, and positioning. It is also crucial in surveying, mapping, and geodetic applications, aiding in precise positioning and measurement. In addition, GNSS plays a vital role in timing synchronization for telecommunications, financial systems, and scientific research.

The ongoing advancements in GNSS technology are focused on improving accuracy, reliability, and availability, particularly in challenging environments like urban canyons or dense forests, where challenges like Multipath error arise. This thesis aims to delve deeper into the analysis of Multipath error in order to develop effective mitigation techniques.

1.1 Project

Over the past 30 years, many countries have invested in building a national infrastructure of GNSS reference stations. In this regard, Sweden has been a pioneering country and has built up a network of GNSS reference stations, SWEPOS, which is now run by the National Land Survey's head office in Gävle. Data are streamed in real time from more than 500 GNSS stations across the country and is made available to users who require position accuracy at the centimeter level in real time. The technology is commonly referred to as Network-Based Real Time Kinematic Positioning (NRTK).

In collaboration with the Swedish Transport Administration and RISE Research Institutes in Sweden, the Chalmers Institution Space, Earth and Environment (SEE) is now conducting a study of several of the most significant sources of error in this technology. One of the biggest remaining problems is related to the GNSS antenna and how it is mounted on, for example, an excavator or road machine. Since GNSS technology is based on the reception of satellite signals, it is desirable to receive the signal directly (as intended), but there is also a great risk that the signals are received after one or more reflections on other decals of the excavator.

1.2 Problem definition

Multipath error occurs when the signals transmitted by GNSS satellites reach the receiver's antenna through multiple paths, resulting in the reception of delayed and distorted signals. These multiple paths can be caused by reflections, diffraction, or scattering of signals from surrounding structures, such as buildings, trees, or terrain.

Due to these interaction of the signal with the environment, replicas of the desired signal arrive at the receivers which are delayed and are usually weaker in comparison to the direct signal. The receiver interprets these delayed signals as if they were coming directly from the satellite, leading to erroneous position calculations. Careful and sophisticated antenna design can reduce the Multipath error to a certain extent, however they are expensive.

1.3 Objective

The primary objective of this thesis is to analyze the Multipath error by utilizing collected measurement data. The focus will be on developing a classification system using machine learning techniques. The ultimate goal is to leverage this knowledge to use it in a signal processing algorithm to minimize the detrimental effects caused by Multipath error.

1.4 Method

In-order to do that, GNSS receiver mounted on an excavator will be used to collect data both when the excavator is stationary and when it is in motion. The collected measurements are then labeled as line of sight (LOS), line of sight plus Multipath (LOS + Multipath), non-line of sight (NLOS), and non-line of sight plus Multipath (NLOS+Multipath) reflection using raytracing function within the wireless communication toolbox in Matlab.

For obtaining the residual from the GNSS data, the software GipsyX (developed by CalTech and Jet Propulsion Laboratory under contract with NASA [1]) is used, which, among other things, results in a matrix with residuals as a function of the angle of incidence of the satellite signals.

The data collected from the receiver and processed by GipsyX is then analysed with respect to the labels to identify and engineer useful features to use it in machine learning algorithms. Then we apply machine learning algorithms including support vector machine (SVM), autoencoders, 1D convolution neural network (1D-CNN), and Gated Recurrent Unit (GRU) to train and classify the signals.

1.5 Limitations

The location and number of tests and measurements is limited due to the availability of the excavator. The 3D model of the environment used in raytracing simulation is approximate. The raytracing algorithm is from the wireless communication toolbox in Matlab which has a limitation in that the distance between the receiver and transmitter cannot be more than 500 km great circle radius. Some physical phenomena such as refraction, diffraction and scattering are not part of the raytracing simulation. Hyper parameter searches and optimisation are not the focus of this thesis. This thesis uses measurements from a single location. However to obtain a better generalisation from the machine learning model, separate measurements from different location is recommended.

1.6 Thesis outline

The thesis begins with an Introduction, providing an overview of our research topic, objectives, and significance. Following this, the Literature Review chapter explores the application of machine learning in GNSS, specifically focusing on addressing multipath errors. The third chapter details the Hardware Used for Measurements, while the subsequent Measurements chapter presents collected data, particularly the pseudorange and Signal Strength Indicator (SSI) for GPS and Galileo constellations. The results from Postprocessing of Measurements using GipsyX are discussed in the fourth chapter. The fifth chapter delves into Raytracing Simulation to label data for supervised machine learning, and the Feature Analysis chapter compares measurements with simulation results. Subsequently, the Support Vector Machine

chapter introduces SVM for classification, followed by the Autoencoder chapter exploring new feature generation. The Multi-label Multi-output Classification chapter consolidates insights from previous approaches to improve the classifcation results. Finally, the cocnlusion chapter concludes the thesis, summarizing the key findings.

2
Literature review

The performance of GNSS localization excels in open areas with good satellite visibility, but faces challenges in urban environments due to reduced satellite visibility caused by multipath errors. Various methods have been explored in the literature to improve GNSS localization by addressing multipath errors, ranging from receiver concepts, intricate antenna designs, advanced signal processing techniques, and positioning algorithms [2].

In [2], the authors provide a comprehensive overview of the technological trends in GNSS user equipment, encompassing both hardware and algorithms integrated into systems. On the contrary, in [7], the authors present various use cases where machine learning is explored in the GNSS field. These include GNSS signal acquisition, signal detection and classification, Earth observation, precise positioning, and indoor navigation.

In the application of machine learning towards mitigating multipath errors, researchers have typically used supervised machine learning to identify and address signals affected by reflections from the surrounding environment. Classifiers are trained to distinguish signals reaching the receivers as line of sight (LOS), multipath, or nonline of sight (NLOS). Two primary strategies are commonly employed in this context: applying machine learning algorithms to signals obtained directly from the front end of the receivers, such as the in acquisition or tracking loops [8], or applying them after the signals have been processed by the receiver, such as utilizing RINEX (Receiver INdependent EXchange) data [4], [3], [6]. In this context, the authors in [4] compares the classifiers trained based on the data obtained from national marine electronics association (NMEA), RINEX and correlator and claim that classifier based on correlator data improved the metric F1 score by 25 percent compared to the classifier trained by former method.

In particular, [4] reported a 75 % accuracy using support vector machines (SVM) using raytracing simulations to label the data. Their methodology involves creating features, including one derived from the difference between the time derivatives of pseudorange and Doppler shift signals. In our study, we draw inspiration from this approach and create a feature by computing the difference between the time derivatives of pseudorange from code and carrier signals, and we call this "difference in range velocity". Additionally, we introduce another feature, "difference in range acceleration", derived from the difference in double time derivatives of pseudorange from code and carrier signals. Our analysis reveals that, "difference in range accel-

eration" exhibits a good distribution and statistics when examined in relation to the labeled data. Closely related feature was utilised in [10] where in the Authors use second derivative of Pseudorange.

The authors in [3] adopt a similar approach by employing simulations to label data and subsequently training a GRU to categorize signals into two groups: clean or containing Multipath components. Each data point received by the receiver from both GPS and Galileo constellations is flagged, allowing for the exclusion of signals from the corresponding satellite for one second during the estimation of position, velocity, and time (PVT). Their study incorporates five features: pseudorange, ephemerides (satellite position), Doppler shift, Carrier-to-Noise density (C/N0), and elevation. To benchmark the performance, the authors compare results with other machine learning algorithms such as decision trees, K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and claim that GRU yields superior performance. In subsequent sections of this thesis, we also utilize the GRU, albeit with a different approach than the one in the referenced paper. We transform the problem from a multi-class classification into a multi-label-multi-output problem. This modification involves leveraging features from all 31 GPS satellites at any given instant.

The authors in [6] leverage CNN in conjunction with Sparse Auto-Encoder (SAE) for feature extraction. They subsequently utilize classifiers, namely softmax and random forest, to distinguish between Multipath (MP) and clean signals in both static and kinematic data scenarios. Feature extraction is accomplished using Carrier-to-Noise Ratio (CNR) and Pseudorange Multipath (MP) independently through SAE. These features are then combined to determine the presence of Multipath. Their reported accuracy, measured in terms of recall, reaches 80 percent. Also, preprocessing the data plays an important role in machine learning, and in this paper CNR and MP from each satellite are processed in the time domain. The mean within the filter window is subtracted, and the data is subsequently scaled using the maximum and minimum bounds from the entire dataset to normalize it within the range of zero to one. Data labeling is governed by a rule: if the doubled differenced carrier phase residual exceeds 3 sigma, the data point is labeled as Multipath. In this thesis, we also experiment with CNN-based autoencoder for feature extraction but without the sparsity constraint.

The Authors in [8], [9] use SVM and fully connected neural network to classify NLOS and LOS-with-Multipath using smartly constructed feature using the correlator output, and signal strength from the front end of the receiver. They use fish eye camera with Raspberry Pi to automatically label the data. They claim approximately 97 percent of the NLOS signals were correctly classified. The Authors in [5] use 5 CNN layers with 2 fully connected layer to detect NLOS signal in an indoor environment using tracking loop outputs from pseudolite system(PLS). They use features such as correlation value, carrier phase error, carrier frequency, code phase error and code frequency to discriminate against 5 classes namely, no NLOS or multipath transition, NLOS reception occurs, NLOS reception disappears, a multipath ray appears, a multipath ray disappears. They claim over 75.8 percent avaerage accuracy across

all classes.

In conclusion, existing literature demonstrates diverse approaches to address GNSS Multipath errors using machine learning. This thesis contributes by exploring machine learning algorithms applied to the RINEX data obtained at the back end (post-receiver processing), emphasizing new features and adapting existing methodologies for improved performance. The reason behind this choice is the absence of front-end data. And hence the goal is to build upon existing research based on the RINEX data from the receiver and investigate the effectiveness of machine learning algorithms. This thesis doesn't claim to solve the problem but instead takes a first step in identifying suitable solution by using machine learning approach. While some literature suggest using Azimuth, elevation and residual information, which are typically obtained after processing using the IGS (International GNSS Service) data, we exclude these in our study. The reason is that we want to use signals directly output from the receiver.

Note: We've intentionally placed 75 percent of the literature review at the end. This serves to inspire creativity and critical thinking, making the concluding literature review engaging as it compares methodologies from various authors.

3

Hardware

The hardware utilized in Global Navigation Satellite System (GNSS) measurements plays a crucial role in determining the accuracy and reliability of the collected data. In this chapter, we present the equipment employed for our GNSS measurements, as depicted in Figures 3.1.

Figure 3.1a showcases the Septentrio PolaRx5 receiver, kown for its multi-frequency and multi-constellation capabilities. This receiver enables the tracking of signals from various satellite constellations, including GPS, GLONASS, Galileo, BeiDou, and NavIC, ensuring comprehensive coverage and precise positioning. On the other hand, Figure 3.1b presents the Leica AS10 antenna, a compact yet powerful GNSS antenna. Similar to the receiver, the Leica AS10 is also multi-frequency and multi-constellation, capable of tracking signals from multiple satellite systems, including GPS, GLONASS, Galileo, Compass, and SBAS signals.

(a) Septentrio PolaRx5 Receiver **(b)** Leica AS10 Antenna

Figure 3.1: Images of the receiver (a) and antenna (b) used in the GNSS measurements.

4

Measurements

In this chapter, we provide an overview of the measurement location and briefly present the measurements such as the Range, Signal Strength Indicator (SSI) in time and Azimuth-Elevation domain for both the GPS and Galileo constellations. Additionally, we illustrate the density of measurements in the azimuth-elevation domain as perceived by the receiver, emphasizing that the majority of data points are concentrated in low elevations, which increases susceptibility to Multipath errors. Majority of the measurements are concentrated at low elevation because of the geographic location of Sweden and orbit of the satellite.

4.1 Placement and location of the receiver

The receiver is positioned on the top of an excavator, as illustrated in Figure 4.1. Subfigure 4.1a displays an image from Google Maps, pinpointing the location of the excavator with the receiver mounted on it. Notably, towards the northwest direction highlighted in red lies a tree, while to the west highlighted in yellow is the terrain, and to the south stands a building. Subfigure 4.1b offers a clear view of the relative positions of the building and the tree in relation to the excavator, with the terrain also visible. Subfigure 4.1c depicts the placement of the receiver on top of the excavator. Lastly, the figure 4.1d shows the open field located east of the excavator.

On 3rd of September, the GNNS antenna was positioned on an excavator pointing north, marking the start of the measurements. On the 5th of September, between 7:30 AM and 8:00 AM, the excavator-top (only the top part, not the whole body) was rotated towards the east. However, due to the antenna's placement slightly off the center of the excavator-top, its position shifted by a few centimeters during rotation. Careful consideration of this change is essential during postprocessing to prevent error propagation into the residuals.

Continuing the measurements, on the 6th and 7th of September, between 7:30 AM and 8:00 AM each day, the excavator top was rotated again to face south and east, respectively. As a result, we obtain measurements for the antenna pointing toward the north (3rd-5th September), east (5th-6th September), south (6th-7th September), and west (7th-8th September). Given that the excavator's arm obstructs the antenna's view towards the satellites, we can expect multipath and non-line-of-sight (NLOS) signals on different days.

(a) Image from google maps showing the location of the excavator.

(b) Photo of excavator showing the building, tree and the terrain near it.

(c) Photo of the excavator showing the placement of the receiver.

(d) Photo of the excavator showing the environment around it.

Figure 4.1: Placement of the Receiver on the excavator and its surrounding environment. Image (a) provides an overview of the excavator and its surroundings. To the south of the excavator is the building, whereas the terrain is situated to the west highlighted in yellow. Towards the north-west direction, there is a tree (highlighted in red), and to the east lies an open field. Image (b) shows the relative positions of the building and tree in relation to the excavator, with the terrain also visible. Image (c) illustrates the receiver's placement on the top of the excavator. Finally, image (d) shows the open field located to the east of the excavator.

On the 9th of September, around 2:00 PM, a digging scenario was simulated. During this period, there were instances when the antenna was briefly obstructed by the excavator's arm, leading to signal obstruction and multipath effects. On the 10th and 11th of September, the antenna and excavator remained stationary and unrotated.

In the subsequent section, a concise presentation of the measured data obtained from the receiver for GPS and Galileo constellations is provided. The study in this thesis is limited to these two constellations. For simplicity, only the range measurements for channels C1C and L2W for GPS, and C1C and L7Q for Galileo are shown among all the available measurements on different channels.

4.2 GPS Measurements

Figure 4.2 presents the measured range on the y-axis, with its corresponding Signal Strength Indicator (SSI) represented using the color bar. The left figure corresponds to the C1C signal, while the right figure corresponds to L2W, both displaying data from all the tracked GPS satellites on the 4th of September. For better clarity, only data from the 4th of September is depicted. These measurements were recorded at a frequency of 1 second.

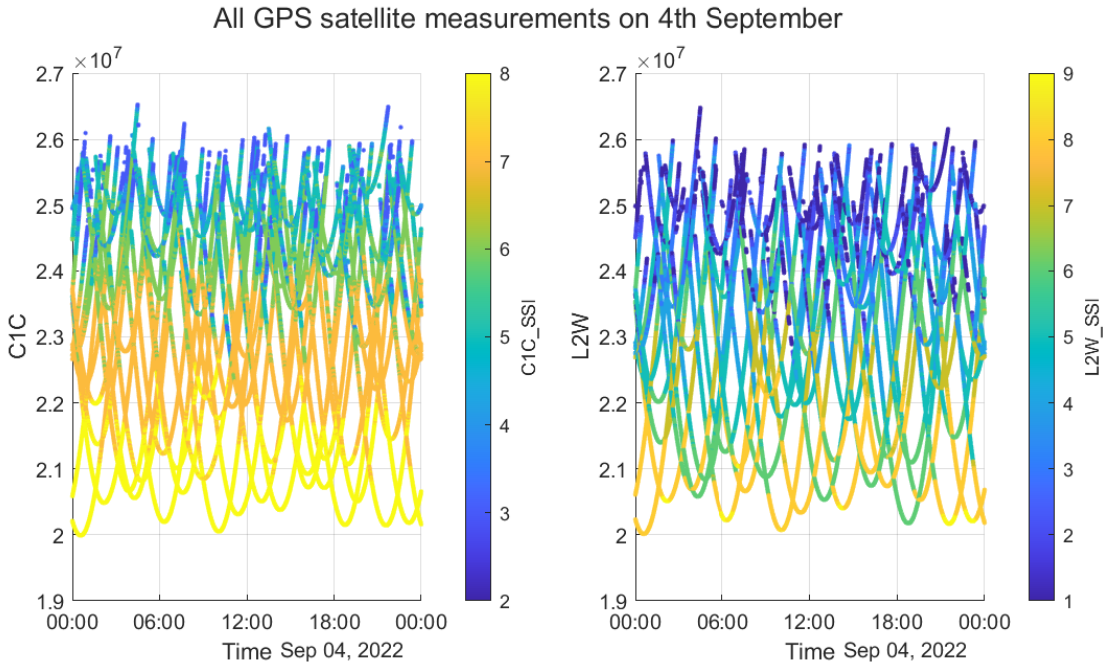


Figure 4.2: Range measurements from all the tracked GPS satellites on C1C (left figure) and L2W (right figure) channels on 4th September. The frequency of the measurements are 1sec. The color bar indicates the signal strength indicator.

The parabolic shape observed in Figure 4.2 is primarily a result of the orbital motion of the satellite around Earth. A brief explanation of the shape of the curve is presented below. The curve reflects the tracking of the satellite from the moment it emerges at a low elevation on the horizon. At this point, the range measurement is relatively high. As the satellite ascends in the sky, reaching its peak elevation at the zenith, the range measurement reaches its minimum value. Following this peak, the satellite starts to descend, and the range measurement gradually increases. This is the reason for the parabolic shape of the curve. Eventually, the satellite goes below the horizon, making it impossible to track its signals. However, the satellite continues its journey around Earth until it emerges again at a low elevation, and the motion repeats. This cyclical pattern results in two parabolic curves per day for a given satellite, encapsulating the satellite's complete tracking cycle as it orbits

the Earth.

In Figure 4.2, the color represents the Signal Strength Indicator (SSI). The curves exhibits noticeable discontinuities, particularly at low SSI levels and high range values. These discontinuities occur primarily when the satellite elevation is low, leading to occasional signal loss in those regions. As a result, the curves show discontinuity during these periods of weakened signal strength. Additionally, there are instances where signals are lost even at higher elevations.

The satellite's position can be obtained from IGS data. Additionally, if we have knowledge of the receiver's position, we can calculate the geometric azimuth and elevation of the satellite as perceived by the receiver. This azimuth-elevation information is shown in polar plot in figure 4.3 and in figure 4.4. with color bar indicating the range and SSI respectively. Notice that the only difference between these two figure is the colorbar. In all the polar plots, the radial axis represents the elevation, and the circumferential axis denotes the azimuth. An elevation of 90 degrees corresponds to the satellite being directly above the receiver (in the zenith), and an elevation of 0 degrees corresponds to the horizon.

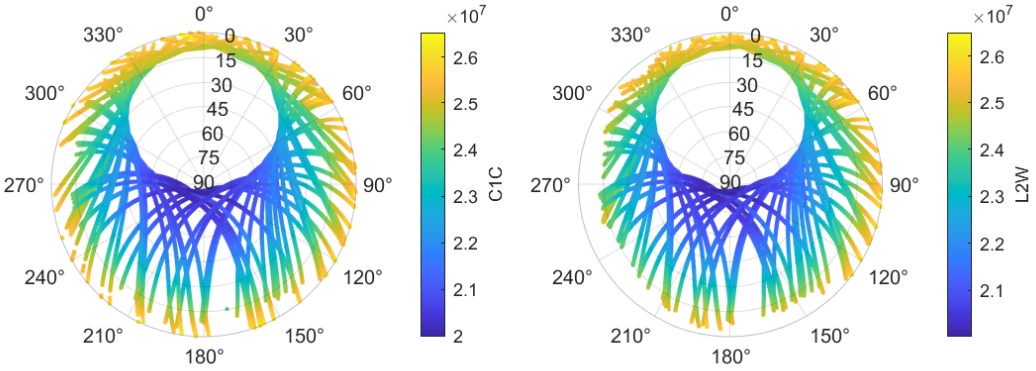


Figure 4.3: Polar plot of satellite trajectories and range measurements, from all the tracked GPS satellites, on all days, on C1C (left) and L2W (right) channels. The frequency of the measurements are 1sec.

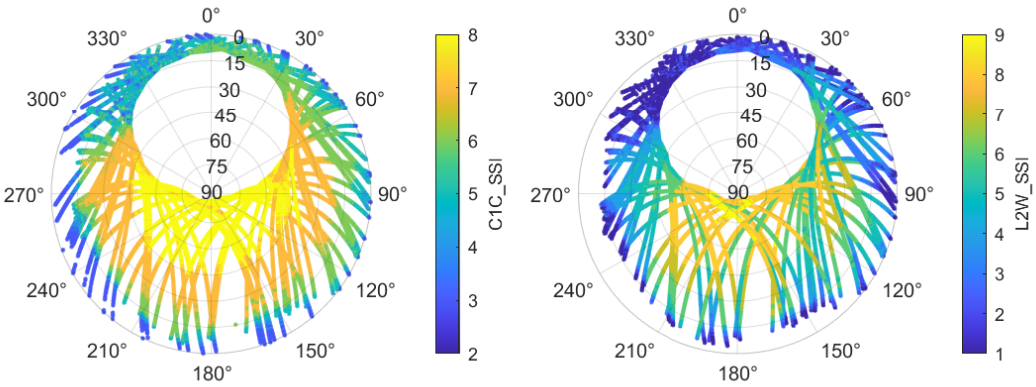


Figure 4.4: Polar plot of satellite trajectories and signal strength indicator(SSI), from all the tracked GPS satellites, on all days, on C1C (left) and L2W (right) channels. The frequency of the measurements are 1sec.

When observing the plots, it becomes evident that high range values occur at low elevations and low Signal Strength Indicator (SSI) levels. In contrast, low range values occur at high elevations and high SSI levels. This conclusion was also derived from the time series plot shown in Figure 4.2. The parabolic shape of the curves, observed in Figure 4.2, is also evident here. Furthermore, the white empty spot at the center of the figure is a consequence of Sweden's geographical location and the satellite's orbit, leading to a lack of visible Satellite in that region. To provide additional insights, a density plot is displayed in Figure 4.5. This plot indicates a higher concentration of data points at low elevations, particularly between azimuth angles of approximately 300 to 60 degrees. It is worth noting that low elevation measurements are more susceptible to multipath effects.

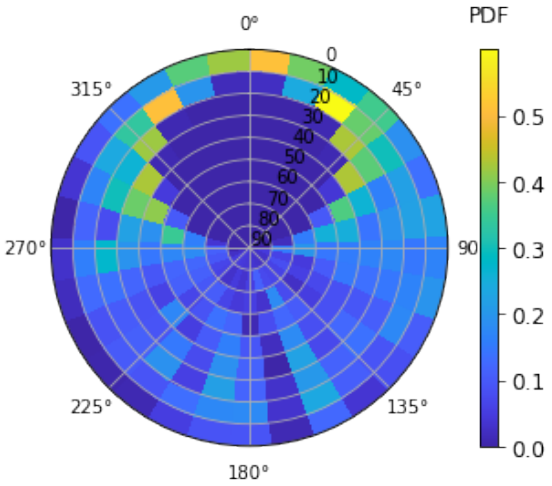


Figure 4.5: Density of measurements at different Azimuth-elevation for GPS constellation. It shows more data are collected at low elevation within approximately 300 to 60 deg Azimuth.

4.3 Galileo Measurements

Figure 4.6 shows the measured range on the y-axis and its corresponding signal strength indicator (SSI) using the color bar. The left figure corresponds to the C1C channel and the right figure corresponds to the L7Q channel, both displaying the data for all the tracked Galileo satellites on the 4th of September. When comparing the range measurements from the Galileo constellation with those of the GPS constellation, the range measurements from Galileo are generally higher. This difference is because of distinct orbits of the GPS and Galileo constellations, with Galileo satellites being further away from the receiver compared to GPS satellites. Despite the differences in range values, measurements from both constellations exhibit a parabolic shape, which arises from the satellites orbital motion around the Earth. The curves display discontinuities, mainly occurring due to weakened SSI at low elevations and occasionally at higher elevations as well.

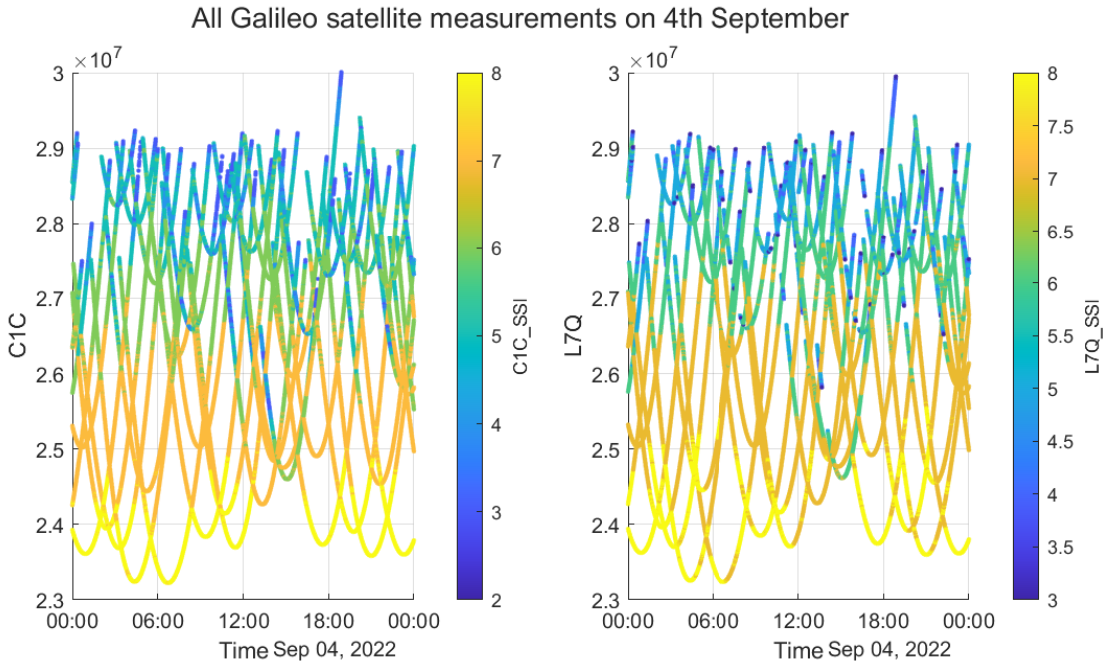


Figure 4.6: Range measurements from all the tracked Galileo satellites on C1C (left figure) and L7Q (right figure) channels from 4th September.The frequency of the measurements are 1sec. The color indicates the signal strength indicator

The satellite trajectory in azimuth-elevation domain from Galileo constellation is shown in figure 4.7 and in figure 4.8, with color bar indicating the range and SSI, respectively. When compared to GPS measurements, the measurements from Galileo constellation fill up the azimuth-elevation space better. In the plot, certain trajectories appear to stand out because of dissimilar range patterns, and these have been identified as satellite IDs 11, 12, and 19.

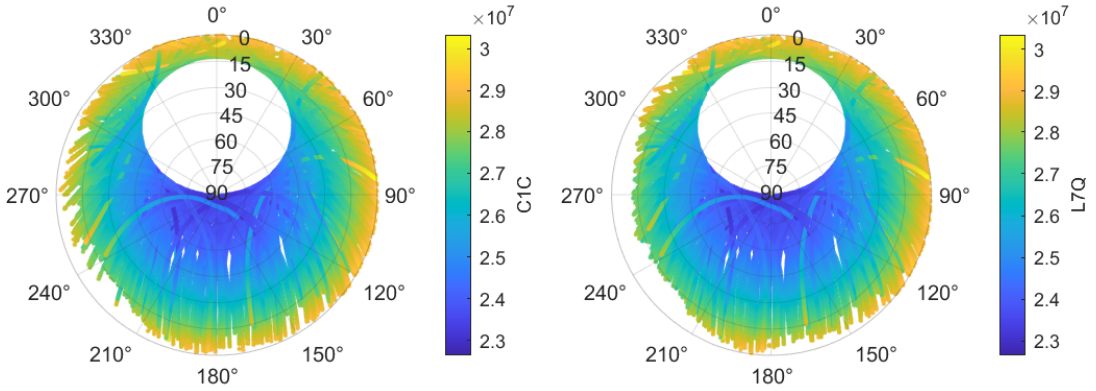


Figure 4.7: Polar plot of satellite trajectories and range measurements, from all the tracked Galileo satellites, on all days, on C1C (left) and L7Q (right) channels. The frequency of the measurements are 1sec. Color indicates the range

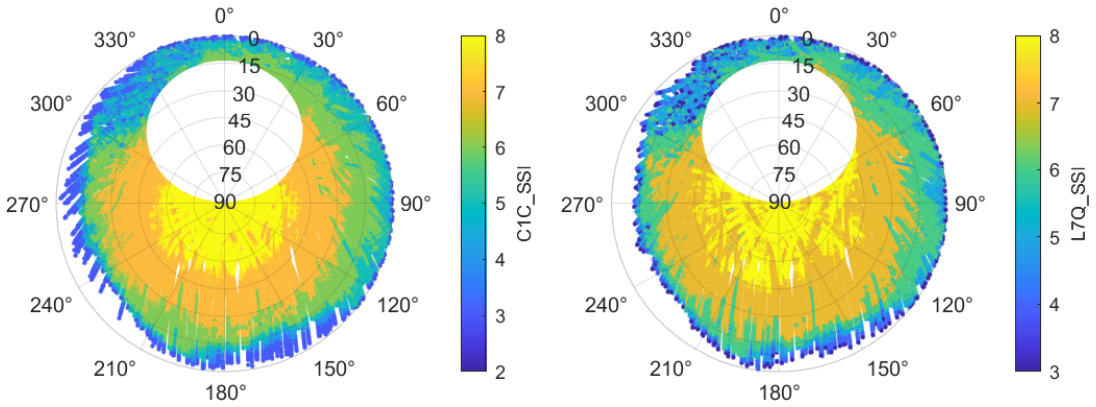


Figure 4.8: Polar plot of satellite trajectories and range measurements, from all the tracked Galileo satellites, on all days, on C1C (left) and L7Q (right) channels. The frequency of the measurements are 1sec. Color indicates SSI

Figure 4.9 shows the density of measurements in the Azimuth-elevation space, indicating a notable concentration of points at low elevations, particularly between azimuths approximately 300 to 60 degrees. This pattern closely resembles the GPS data observed in Figure 4.5, implying that it is a characteristic of the geographic location of Sweden and the orbit of the satellite. Furthermore, it is important to note that low elevation measurements are more susceptible to multipath effects.

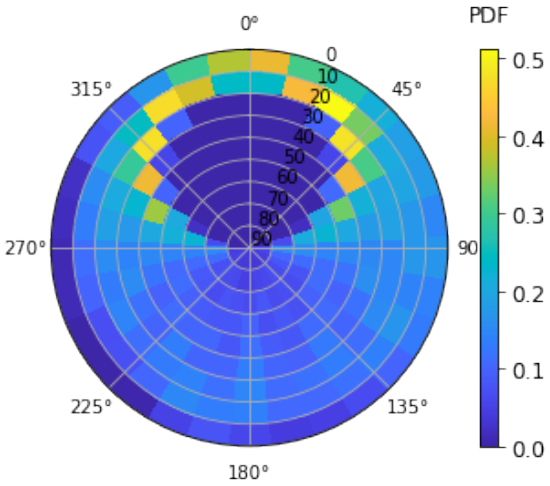


Figure 4.9: Density of measurements at different Azimuth-elevation for Galileo constellation. It shows more data are collected at low elevation within approximately 300 to 60 deg Azimuth

5

Post processing using GipsyX

The measurements obtained from the receiver are processed using GipsyX, a tool developed by the Jet Propulsion Laboratory for applications in positioning, navigation, timing, and earth science [1]. GipsyX employs an Extended Kalman Filter (EKF) and, in conjunction with the Precise Point Positioning (PPP) technique enabled by data from the International GNSS Service (IGS), generates residuals at 30-second intervals. The PPP technique mentioned above utilizes a combination of code and carrier phase measurements to mitigate errors arising from ionospheric and clock effects. The residuals thus obtained from GipsyX represent the disparities between the true and estimated positions. These residuals encompass errors stemming from multipath reflections and measurement noise.

Note: The labeling of measurements using Raytracing simulations will be discussed in the next chapter. However, for the sake of understanding this chapter, it's essential to introduce some terminologies and results here. Through raytracing, the data is categorized into four classes: Line of Sight (LOS), LOS with Multipath (LOS+Multipath), Non-Line of Sight (NLOS), and NLOS with Multipath (NLOS+Multipath).

It is important to mention that the receiver records the original range measurements at 1-second intervals. Consequently, the labels obtained from raytracing are also provided at the same 1-second intervals. However, the residuals which are generated by GipsyX after post-processing the original measurements are at 30-second intervals. Therefore, these two datasets are synchronized, meaning that each data point in the residuals is labeled with corresponding raytracing results and range measurements from the receiver. This can be seen as downsizing of the original measurements at 30 sec sampling rate with the residual information.

In this chapter, we will conduct an analysis of the residuals in relation to the labels obtained from raytracing.

Note: While analysing it was noticed that one of the GPS satellite with SVN ID 76 showed abnormal residuals between 6 to 7 am between Septemebr 7th and 8th, and hence was not considered in the analysis.

5.1 General statistics about the dataset

Figure 5.1 illustrates the total percentage of each labels in the residuals, for each day, for both the GPS (Top subplot) and Galileo (bottom subplot) constellations. The numeric values displayed above each column represents the total number of data points in the residuals generated by GipsyX for the corresponding days. It is worth noting that the first column encompasses more data points as it spans over two days. Conversely, the last column contains the fewest data points due to the measurement being stopped on the 11th of September, contrary to what is inferred from the x-axis. Furthermore, it is evident that there is a higher occurrence of LOS labels when compared to the other labels. This can be attributed to the excavator's location, which is situated in a fairly open space with fewer obstructions, resulting in lower occurrences of NLOS and multipath labels.

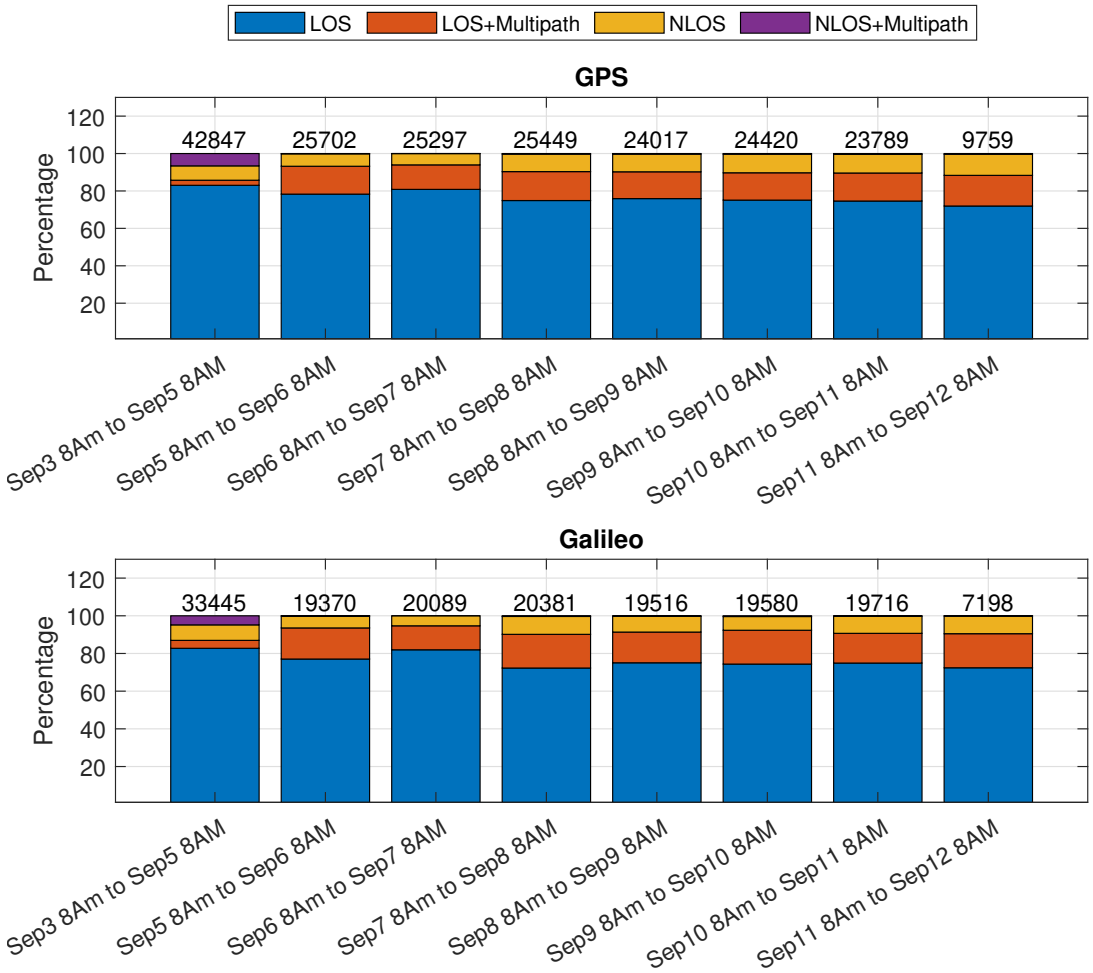


Figure 5.1: Total percentage of labels in residuals for each day, comparing GPS (top) and Galileo (bottom) constellations. Numeric values above each column indicate the total data points in residuals generated by GipsyX for corresponding days.

5.2 Distribution of residuals with respect to raytracing results

Figure 5.2 depicts the residual distribution with respect to labels for the GPS constellation. In each subplot, a red curve representing a Gaussian fit is shown, with the figure clipped to show the 3-sigma region for clarity. It is important to note that higher residuals in the tail of the distribution, exceeding what is shown, are present but are not shown for clarity. In GipsyX, a rule is used to tag and exclude data points exceeding 0.25m as outliers during position estimation. While disregarding these outliers would result in a normal distribution (and better position estimation), we choose to include all residuals, mainly because those linked to Multipath errors often exhibit higher magnitudes. Additionally, it's crucial to recognize that GipsyX generates residuals at 30-second intervals, potentially providing insufficient data points for robust distribution construction, especially for NLOS and NLOS+Multipath categories. Nevertheless, noticeable trends ca be seen in figure 5.2: generally, LOS labels show a lower mean and standard deviation compared to NLOS and Multipath labels. In the next section, we will identify residuals above the standard deviation of the LOS (0.0384) as outliers for improved visual clarity.

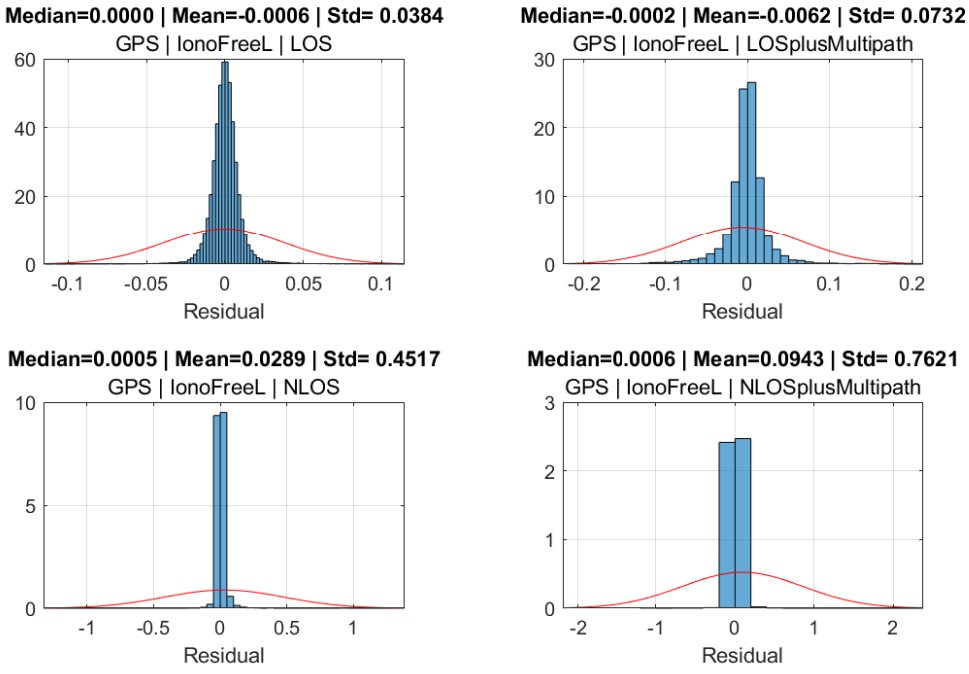


Figure 5.2: Distribution of residuals with respect to labels from raytracing for GPS constellation. In each subplot red curve indicates a Gaussian fit.

Similarly, Figure 5.3 shows the residual distribution with respect to the labels for Galileo constellation. Unlike what we have observed in Figure 5.2 for GPS constellation, here for Galileo constellations we do not discern a clear, consistent trend.

This could be because of the dynamic nature of the Galileo constellation, where the azimuth-elevation of satellites with respect to the receiver changes daily, potentially leading to varying multipath reflections.

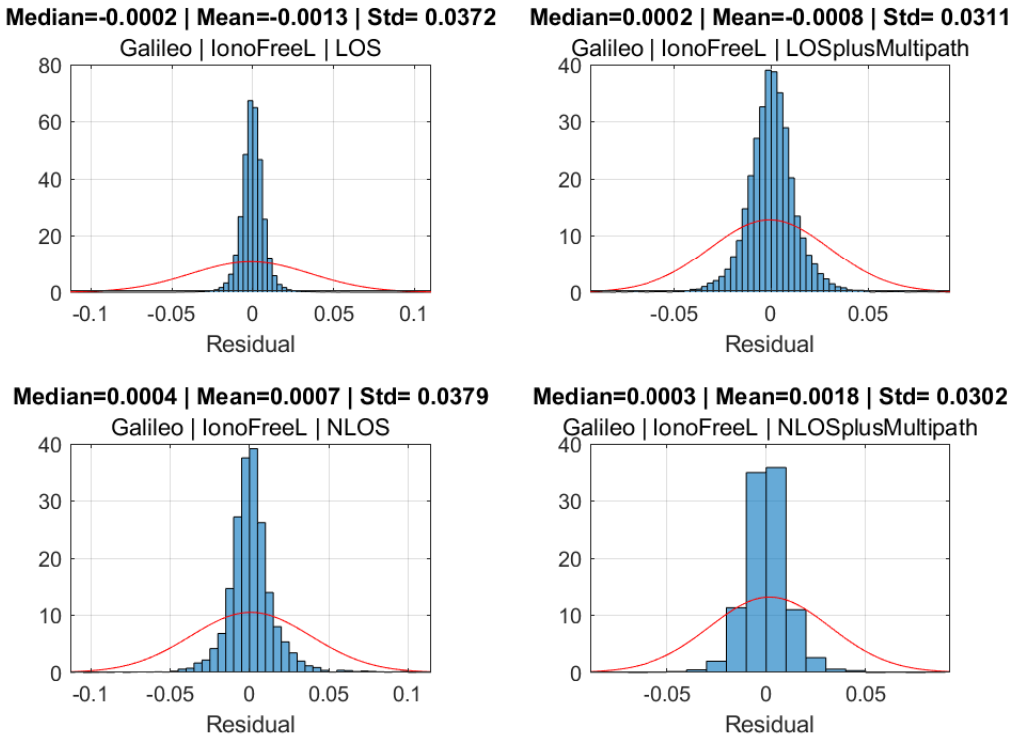


Figure 5.3: Residual distribution with respect to labels from raytracing for Galileo constellation. The red curve in each subplot represents a Gaussian fit.

Figures 5.2 and 5.3 depict the distribution of data points for the entire measurement set. However, considering the excavator's daily rotation which alters the measurement setup, it is pertinent to analyze statistics for each day. Thus, Figure 5.4 presents the daily mean and standard deviation of the residuals concerning the labels for both the GPS and Galileo constellations.

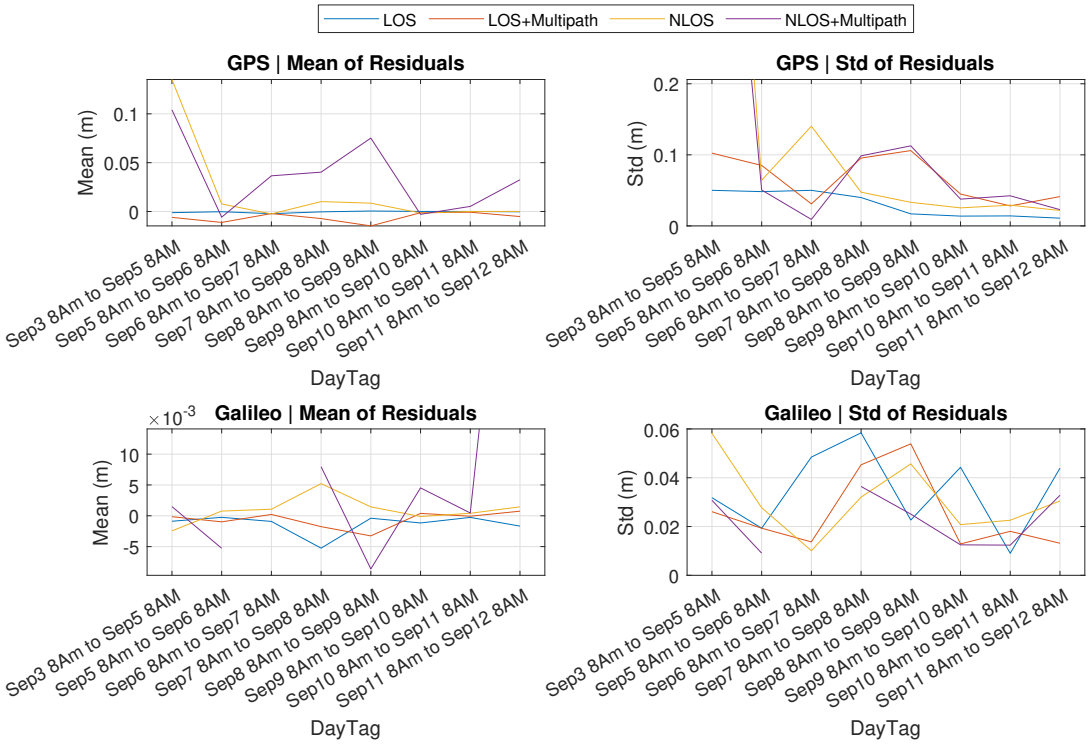


Figure 5.4: Daily summary illustrating the mean and standard deviation of residuals with respect to labels for both GPS and Galileo constellations

5.3 Comparison of Residuals with Ray Tracing Labels

Figure 5.5 shows the comparison of residuals with the labels obtained from ray tracing for both GPS and Galileo constellations. As mentioned in the previous section, residuals above the threshold of 0.0384 are flagged as outliers. The first column of figures displays the residuals without outliers, while the last column exclusively shows the outliers. The middle column represents the labels obtained from ray tracing.

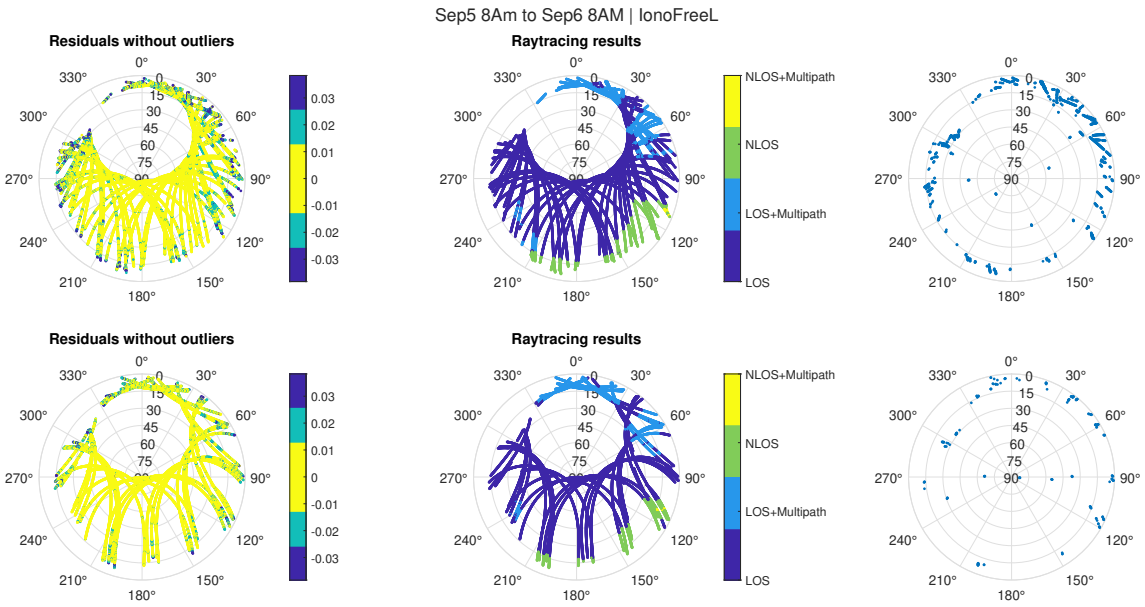


Figure 5.5: Comparison of residuals with ray tracing labels for GPS and Galileo constellations. Fist and second row represents GPS and Galileo constellations, respectively. first column of figures displays the residuals without outliers, while the last column exclusively shows the outliers. The middle column represents the labels obtained from ray tracing.

When comparing the ray tracing labels in the second column with the figures in the first column, a noticeable correlation emerges: regions labeled as NLOS or multipath tend to coincide with high residuals. This relationship persists even when examining the results in the last column, which focuses solely on the outliers. It is worth noting that some outliers occurring in the regions labeled as LOS might be attributed to limitations in the environmental models and the ray tracing simulations. Therefore, future research should prioritize the development of more accurate environmental models to enhance the precision of ray tracing simulations and, consequently, the accuracy of labeling.

6

Ray tracing

In chapter 4, we presented the measurements recorded by the receiver. In order to label these observations, a ray tracing method was employed, which will be discussed in this section.

Ray tracing is a widely used rendering technique that simulates the behavior of light in a virtual environment to create realistic images and is widely used in computer graphics. However, in this thesis we employ Ray tracing from wireless communication tool box within Matlab. Our purpose here is to label each time epoch rather then rendering images. Therefore, we opted for Matlab, despite some limitations (which will be discussed later in this chapter), as it suffices for achieving our thesis goals.

Within Matlab, shooting bouncing rays (SBR) method is used for ray tracing. SBR is a technique used in ray tracing to simulate the behavior of light rays as they bounce off surfaces in a virtual environment. Several rays are launched from the transmitter site and are traced forward to the receiver while taking care of interactions with virtual environment such as reflection, refraction, diffraction and scattering. At the time of writing this thesis, Matlab only supports reflection and do not support corner diffraction, refraction or diffuse scattering. Keeping the simulation time in mind, up to two reflection paths are considered.

The ray tracing functionality within the wireless communication toolbox in Matlab is limited to scenarios where the receiver is within a 500 km great circle radius distance from the transmitter. Consequently, in order to perform ray tracing, the transmitter is placed at a fixed distance of 500 km from the receiver, even though real satellite distances range from 20,000km to 26,000km for GPS constellations (23000km to 30000 km for Galileo constellations). It is important to acknowledge that this approximation introduces a certain level of error in the simulated results, and is not as big as initially thought, and will be thoroughly discussed in later sections of this chapter.

Also, in order to improve the reliability of the simulation results, the transmitter and receiver positions are interchanged, as will be shown in figure 6.2a later.

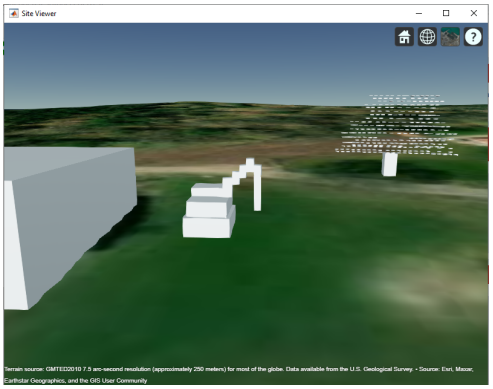
6.1 Environment model

Ray tracing relies on having an accurate representation of the environment. However, in this thesis an approximate virtual environment was created by extracting the base layer from OpenStreetMap. The base layer was further modified in JOSM to incorporate additional details such as buildings, excavators, and trees. To visualize this virtual environment, Matlab's site viewer was utilized, resulting in the rendering shown in figure 6.1b and 6.1d. For comparison, the real environment is shown in figure 6.1a and 6.1c.

(a) Image from google maps showing the location of the excavator.

(b) Environment rendered in Matlab

(c) Photo of excavator showing the building, tree and the terrain near it.

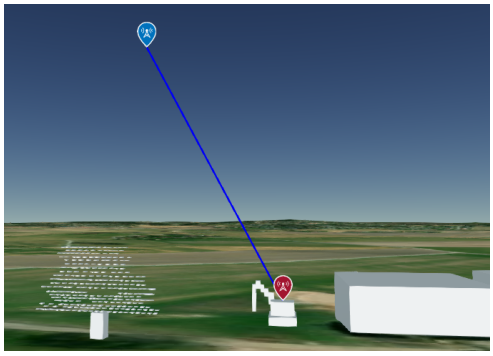


(d) Rendered building, excavator and tree

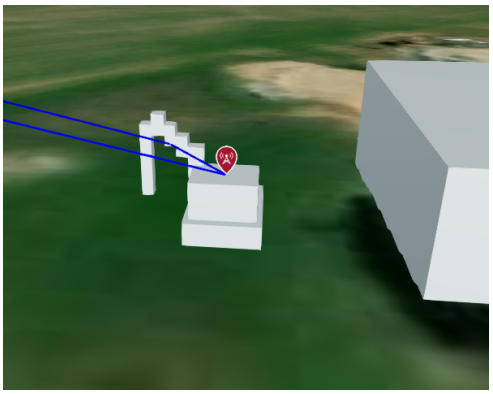
Figure 6.1: Comparison of True and Rendered Environment in Matlab. Image (a) and (b) shows the top-view of the true and rendered environment around the excavator, respectively. Image(c) and (d) shows comparison of the true and rendered building, excavator and tree, respectively

6.2 Labels and classes

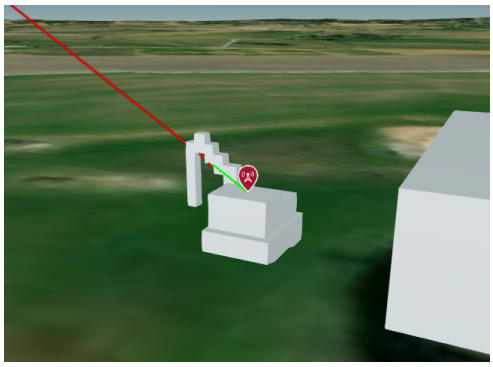
By employing ray tracing, we can accurately determine the paths of the signals and analyze their interactions with the virtual environment. This allows us to label the observed measurements into LOS (line of sight), or NLOS (non line of sight), or Multipath reflection, or combination of them.



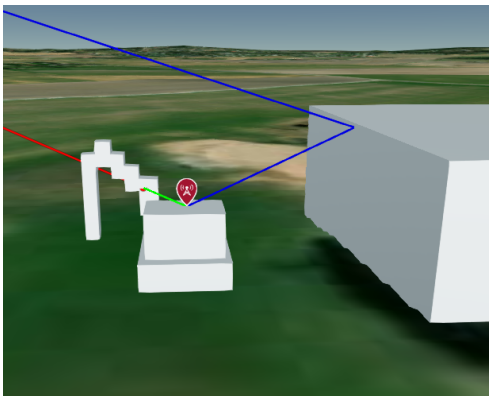
(a) Example of line of site (LOS)



(b) Example of line of site (LOS) plus Multipath reflection from the arm of the excavator



(c) Example of non line of sight (NLOS).Arm of the excavator obstructing the signal from the satellite.



(d) Example of non line of sight (NLOS) plus Multipath reflection from the building.

Figure 6.2: Example of labels generated from ray tracing when the excavator is pointing north. Ideally, the blue icon in images represents the satellite location and the red icon represent the receiver location. But for better simulation performance these are swapped, meaning the blue and the red icon represents the receiver and transmitter, respectively.

- Line of sight (LOS): It represents a scenario where there exists a direct path between the transmitter and receiver without encountering any obstructions, as shown in figure 6.2a. LOS signals experiences minimal attenuation or interference, leading to stronger and more reliable signal. (Note: In the figure Blue icon represents the receiver and the red icon represents the transmitter. The position of transmitter and receiver are interchanged intentionally for more reliable simulation results.)
- Multipath reflection: It represents a scenarios where the signal interacts with the surrounding environment, such as reflection, diffraction, and scattering, before reaching the receiver. These interactions can result in signal degradation. Figure 6.2b shows an example of a satellite signal reflecting on the arm of the excavator before reaching the receiver. It also shows direct LOS signal from the same satellite, which travels unobstructed to the receiver.

- Non line of sight (NLOS): It represents a scenario where the direct line of sight between the transmitter and receiver is obstructed as shown in figure 6.2c. In NLOS conditions, the signal typically interacts with the surrounding environment, such as reflection, diffraction, and scattering, before reaching the receiver. These interactions can result in signal degradation. Figure 6.2d shows an example of NLOS signal reaching the receiver through reflecting on the surface of the building.

The labels of LOS, NLOS, and Multipath can occur in various combinations. Table 6.1 presents the truth table that outlines all the possible combinations of these labels. During the simulation using a virtual environment, only four combinations, highlighted in the table, can occur. However, it is important to note that pure NLOS signals (Class 3) are never captured in the measurements due to signal obstruction. Nevertheless, the presence of recorded observations at the receiver implies the existence of NLOS signals accompanied by multipath reflections. Consequently, it is appropriate to consolidate or group the labels corresponding to Class 3 together with Class 4.

LOS	**NLOS**	**Multipath**	**Comment**
T	T	T	Cannot be LOS and NLOS at the same time
T	T	F	Cannot be LOS and NLOS at the same time
T	F	F	Pure LOS (Class 1)
T	F	T	LOS and multipath reflection (Class 2)
F	T	F	Pure NLOS (Class 3)
F	T	T	NLOS and multipath reflection (class 4)
F	F	T	Either LOS or NLOS must be true
F	F	F	Either LOS or NLOS must be true

Table 6.1: Truth Table for LOS, NLOS, and Multipath

6.3 Discritized space in Azimuth-elevation domain

Performing ray tracing in Matlab presents two challenges. Firstly, in order to label each time epoch, ray tracing simulations must be conducted for both GPS and Galileo constellations. However, this process can be time-consuming, as it requires performing ray tracing for each time epoch and for each constellation separately. Secondly, the ray tracing functionality within the wireless communication toolbox in Matlab has a limitation: it can only handle scenarios where the receiver is within a 500km great circle radius from the transmitter.
To overcome these challenges, a solution is employed. The Azimuth-Elevation space surrounding the receiver is discretized with a step size of 0.5 degrees. Additionally, a fixed distance of 500km from the receiver is used to represent the discretized space. As a result, the discretized space takes on the shape of a dome, as shown in figure. 6.3. Also in order to improve the reliability of ray tracing simulation, the receiver and transmitter locations are swapped, meaning the receiver is placed on the surface of the dome, and the transmitter on the excavator.

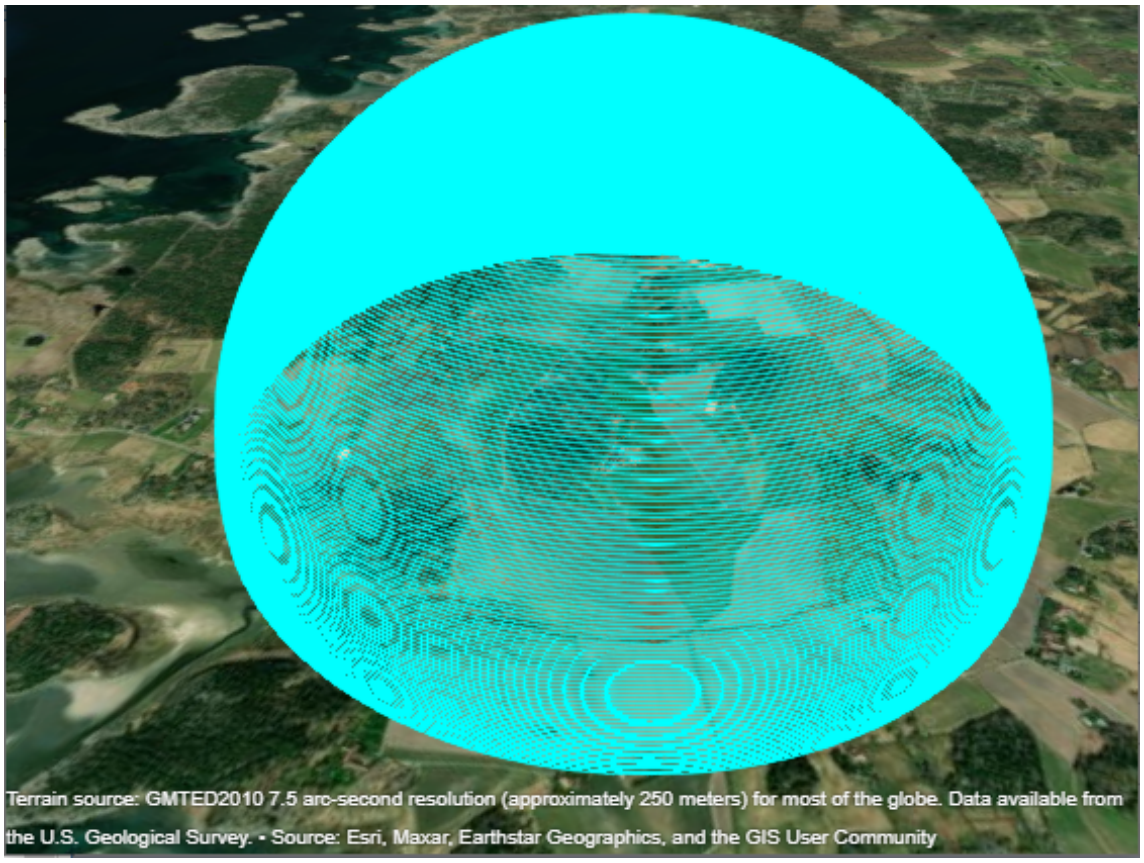
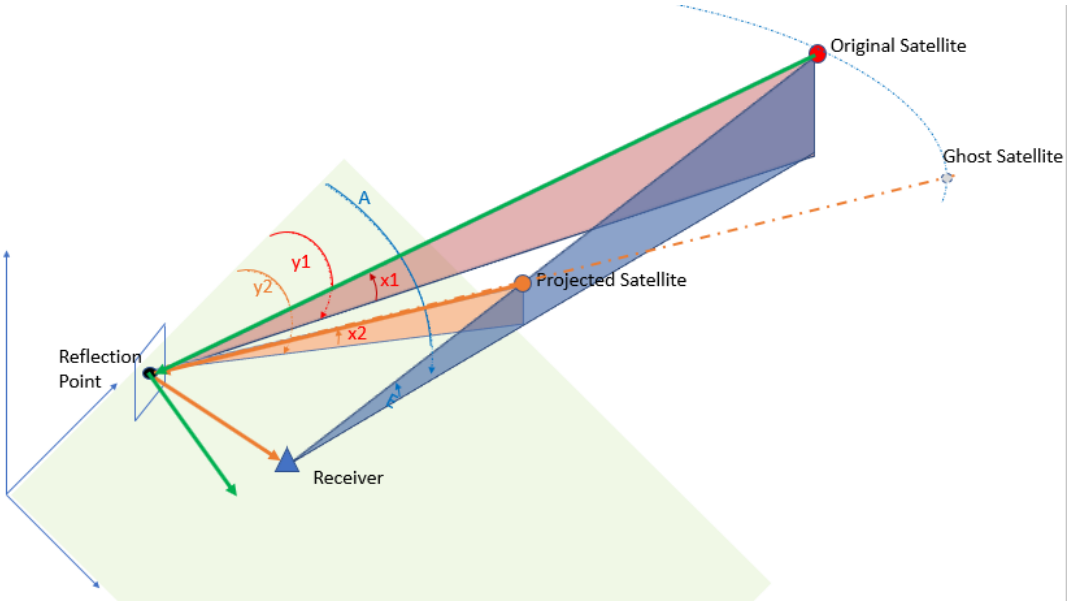


Figure 6.3: Discritised Azimuth-Elevation space around the receiver with a fixed distance of 500km around the receiver. Diagram not to scale.

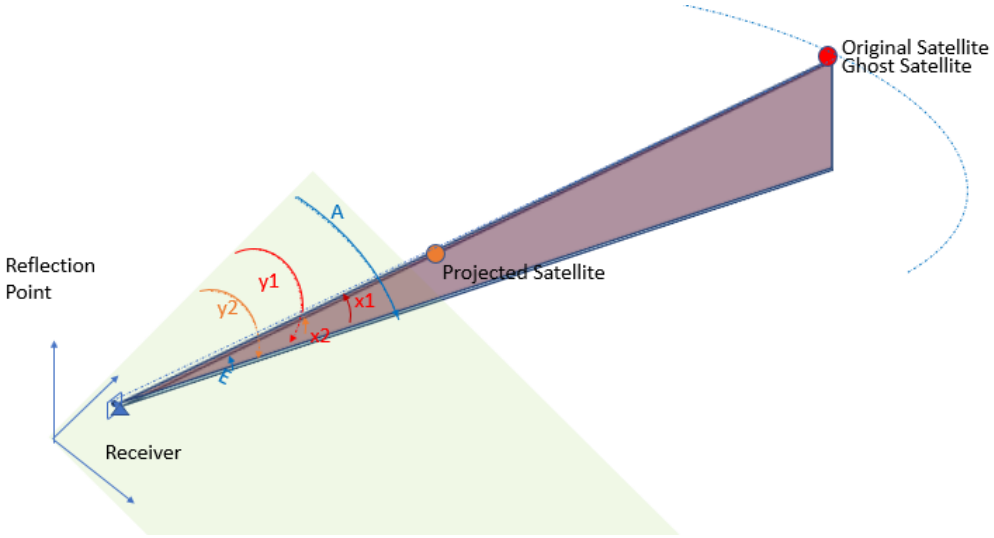
This discretized Azimuth-Elevation space is used for the subsequent ray tracing process. The result is that each each point in the space, characterized by specific Azimuth and Elevation values, is labeled accordingly. This labeling information is then used as a lookup table to enable classification of each time epoch in the recorded observations for both GPS and Galileo constellations.

6.4 Limitation but not as significant as initially thought

The method employed here can be conceptualized as projecting the original satellite position closer to the receiver. Normally, the original satellite positions for the GPS constellation range from 20,000 to 26,000 km, while for the Galileo constellation, they range from 23,000 to 30,000 km. However, due to the limitation mentioned above,in the ray tracing simulation, these satellite positions are projected to a fixed distance of 500 km from the receiver along the line of sight, as depicted in Figure 6.4a.



(a) Schematic illustrating angles in Azimuth and elevation from the projected and original satellite position



(b) A more realistic Schematic when compared to 6.4a

Figure 6.4: Schematic illustrating the initally perceived concept when compared to more realistic schematic

By using the projected satellite positions, the azimuth-elevation data points are labeled during the ray tracing simulation. As a result, a given azimuth-elevation point, as illustrated in Figure 6.4a, would encompass both the line of sight (LOS) and multipath components. However, in reality, the received signal from the original satellite only consists of the LOS component as shown in figure 6.4a. Therefore, this projection does not introduce any approximation in classifying the LOS label. However, initially, it may appear that this approach could result in significant errors when labeling the multipath component. But this is not the case since the diagram is not to scale.

A more realistic representation, depicted in Figure 6.4b, shows that the distance between the reflection point and the receiver is considerably smaller compared to the projected satellite distance of 500 km. Consequently, the differences between azimuth angles A and Y1 (as well as Y1 and Y2) and elevation angles E and X1 (as well as X1 and X2) are very minimal. The difference in the angels computed for one of the GPS satellite is shown in figure 6.5. This clarification highlights that the apparent limitations initially perceived in labeling the multipath component are not significant, and one should consider a realistic scale while imagining this.

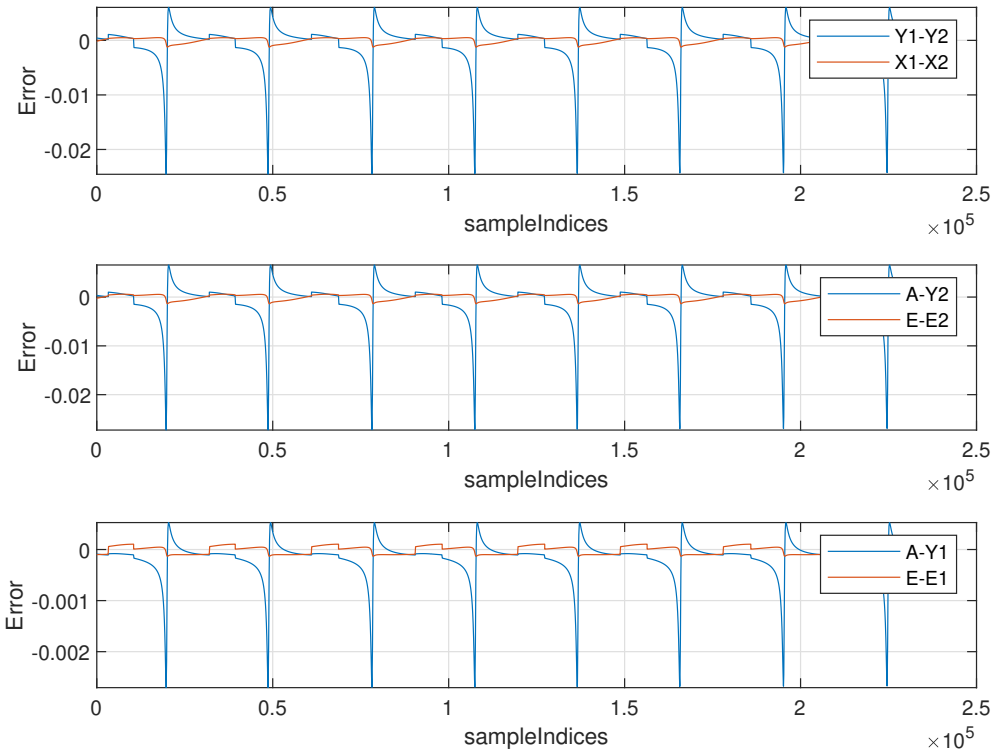
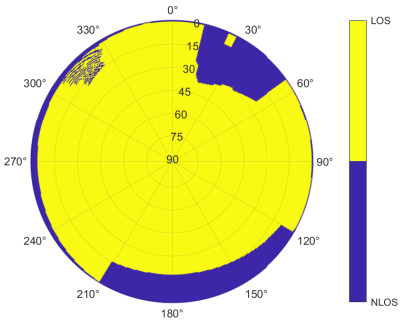


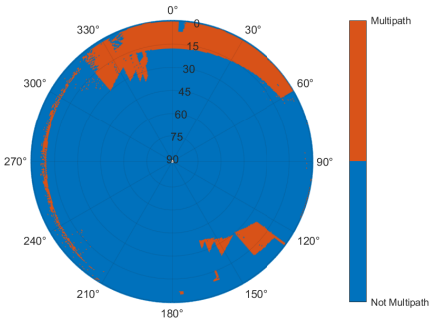
Figure 6.5: Errors introduced due to projecting the satellite closer to over come the limitations in Matlab tool box. It indicates that the apparent limitations initially perceived in labeling the multipath component are not significant.

6.5 Labeling for discretized azimuth-elevation space

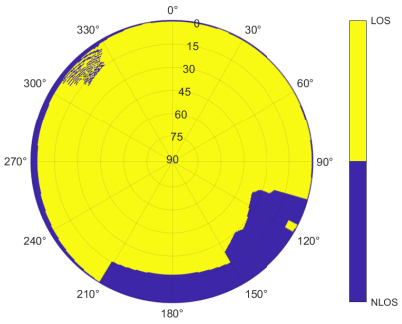
To align with the setup during the measurements, ray tracing is conducted independently for different scenarios where the excavator is oriented towards the North, East, South, and West directions In each of these scenarios, the measurement data points, characterized by their Azimuth and Elevation attributes, are labeled accordingly as LOS, NLOS, or multipath. The corresponding results from ray tracing are shown in Figure 6.6.



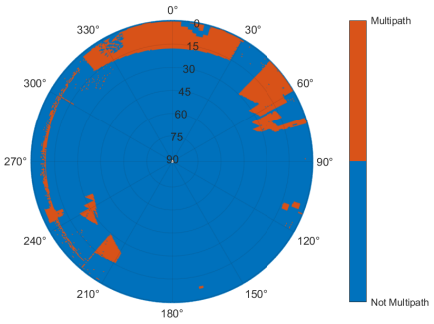
(a) North: Los and NLos



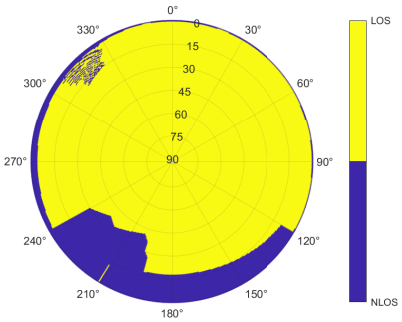
(b) North: Multipath



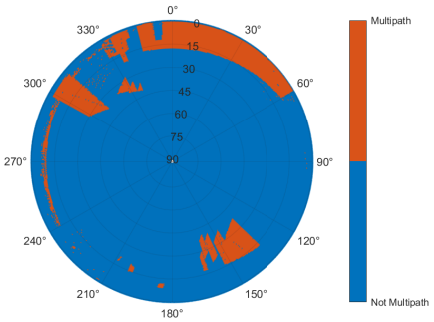
(c) East: Los and NLos



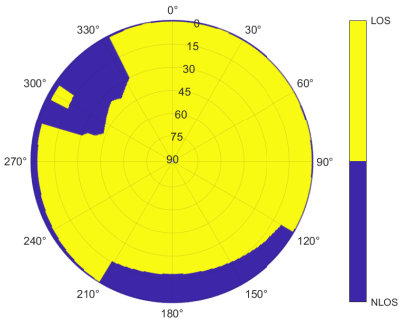
(d) East: Multipath



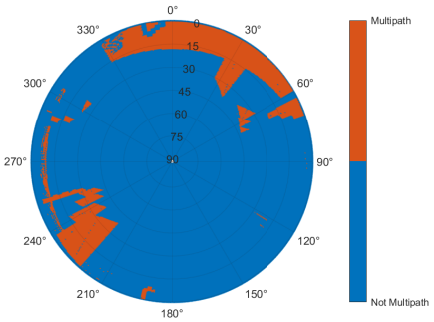
(e) South: Los and NLos



(f) South: Multipath



(g) West: Los and NLos



(h) West: Multipath

Figure 6.6: Results from the raytracing simulation are presented in pairs: Images a and b, c and d, e and f, and g and h correspond to results from the simulations conducted with the excavator pointing north, east, south, and west, respectively. The first columns indicate whether the data point characterized by Azimuth and elevation is Line of Sight (LOS) or Non-Line of Sight (NLOS). Meanwhile, the second columns indicate whether the data point consists of a multipath component. The combination of these two criteria results in the labels LOS, LOS+Multipath, NLOS, and NLOS+Multipath.

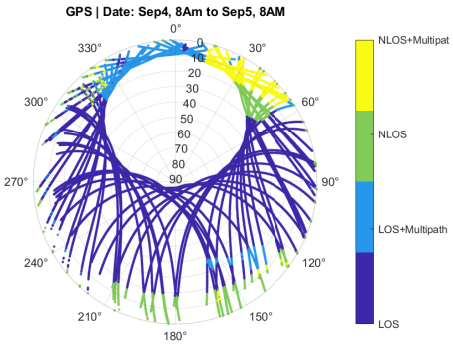
Specifically, Figure 6.6a and 6.6b display the labeling when the excavator was pointing north, while Figure 6.6c and 6.6d showcase the labeling for the east-facing orientation. Similarly, Figure 6.6e and 6.6f demonstrate the labeling in the south-facing scenario, and Figure 6.6g and 6.6h represent the labeling when the excavator was pointing west. The results are stored in the form of a lookup table, enabling the determination of the corresponding label for any given azimuth-elevation pair. The interpolation method employed for lookup is set to utilize the nearest sampled data point, ensuring classification of the measurement points based on their proximity to the recorded samples.

Upon comparing the measurement setup (see Figure 6.1a) with the corresponding results depicted in Figure 6.6a, several notable observations can be made. The NLOS labeling between azimuths 120 deg to 220 deg is a direct consequence of signal obstruction caused by the building. This obstruction is stationary, and as a result, it manifests consistently across all the other sub-figures (figures 6.6c 6.6g 6.6e). Likewise, a similar pattern can be observed for the tree's location, which influences the NLOS labeling between azimuths 300 deg to 330 deg. Additionally, the low elevation NLOS labeling between azimuth 220 deg −0 deg can be attributed to terrain obstruction. Furthermore, the NLOS labeling between azimuths 15 deg to 40 deg is attributed to the excavator's arm, which rotates along the azimuth direction in the remaining sub-figures (figures 6.6c 6.6g 6.6e)

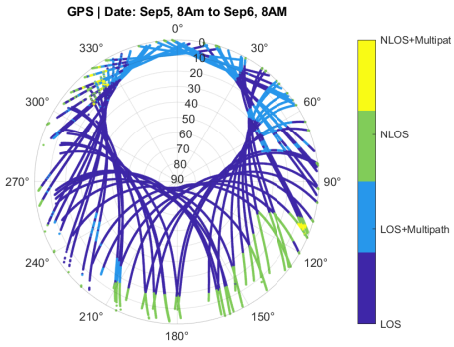
Upon comparing the multipath labeling in Figure 6.6b with the measurement setup illustrated in Figure 6.1a, a distinct pattern becomes apparent between azimuths 330 deg and 40 deg, which persists across the other sub-figures. This recurring pattern can be primarily attributed to the presence of low-elevation multipath reflections originating from the building. It is interesting to observe the building's presence in Figure 6.6a and establish a connection between the multipath label within azimuths 330 deg and 40 deg in figure 6.6b, both of which can be attributed to the building's influence. Additionally the multipath reflection originating from the arm of the excavator can be seen rotating in the azimuth direction in the rest of the sub-figures (figures 6.6d 6.6h 6.6f). It is interesting to observe the low-elevation multipath reflection emerging from the terrain between azimuth 240 deg −0 deg in all the sub-figures (figures 6.6b 6.6d 6.6h 6.6f) .

6.6 Labeling the observations

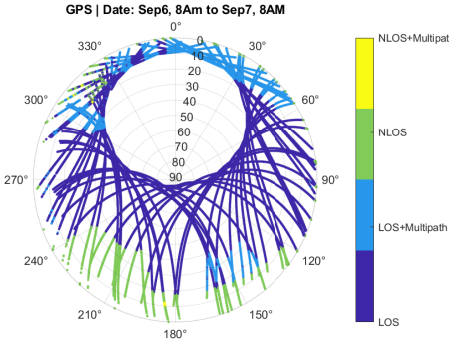
By utilizing the lookup table established in the preceding section, it is now possible to label each time epoch for both the GPS and Galileo constellations in the recorded measurements. This labeling is achieved based on the azimuth and elevation values corresponding to each time epoch. The resulting labels for the GPS constellation are presented in Figure 6.7, while the labels for the Galileo constellations are shown in Figure 6.8.For simplicity, only the results from the 4th to the 7th of September are presented, and the remaining days are omitted since the excavator's orientation remains unchanged, resulting in similar labeling patterns as observed on the 7th of September.



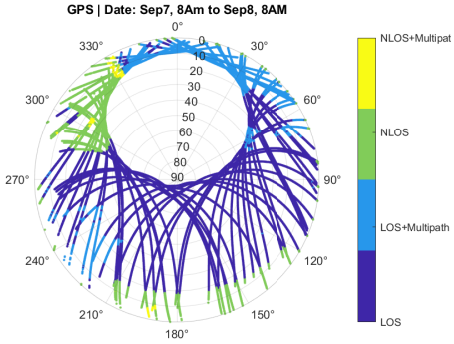
(a) North: Los and NLos



(b) North: Multipath

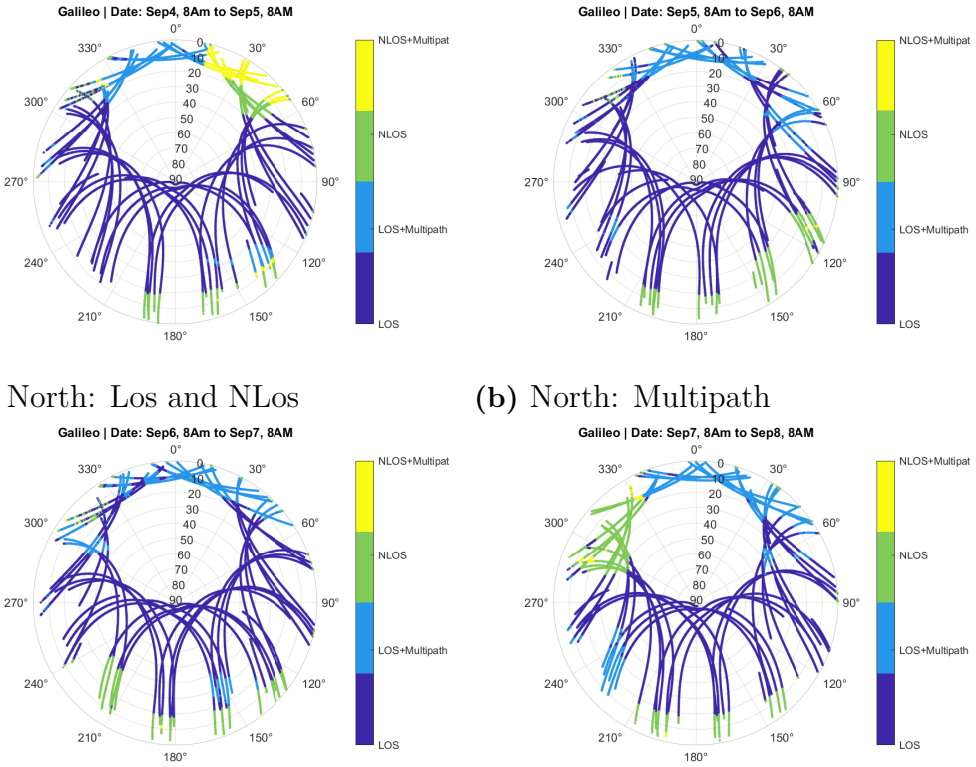


(c) East: Los and NLos



(d) East: Multipath

Figure 6.7: Labels assigned to each time epoch in the recorded measurements for the GPS constellation, determined using the lookup table with azimuth and elevation values. The displayed results span from the 4th to the 7th of September, providing a focused representation as the excavator's orientation remains constant, resulting in similar labeling patterns as observed on the 7th of September.



(a) North: Los and NLos

(b) North: Multipath

(c) East: Los and NLos

(d) East: Multipath

Figure 6.8: Labels assigned to each time epoch in the recorded measurements for the Galileo constellation, utilizing the lookup table based on azimuth and elevation values. The presented results cover the period from the 4th to the 7th of September, simplifying the representation while maintaining consistency due to the unchanged orientation of the excavator

Figure 6.9 illustrates the total percentage of labels for each day for both the GPS and Galileo constellations. The numeric values displayed above each column represent the total number of data points recorded for the corresponding days. It is worth noting that the first column encompasses more data points as it spans over two days. Conversely, the last column contains the fewest data points due to the measurement being stopped on the 11th of September, contrary to what is mentioned on the x-axis. Furthermore, it is evident that there is a higher occurrence of LOS labels when compared to the other labels. This can be attributed to the excavator's location, which is situated in a fairly open space with fewer obstructions, resulting in lower occurrences of NLOS and multipath labels.

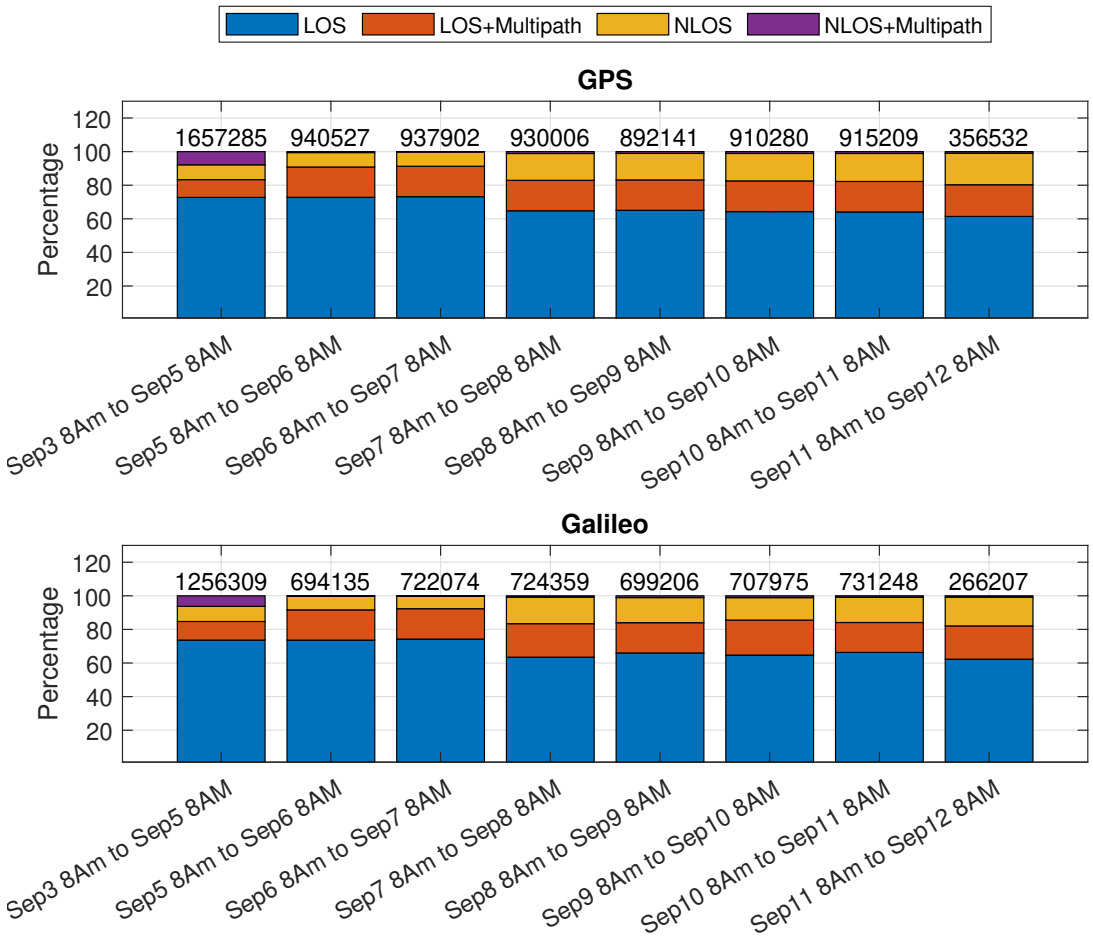


Figure 6.9: Total percentage of labels in residuals for each day, comparing GPS (top) and Galileo (bottom) constellations. Numeric values above each column indicate the total data points in residuals generated by GipsyX for corresponding days.

As mentioned before, it is important to note that pure NLOS signals (Class 3) are never captured in the measurements due to signal obstruction. Nevertheless, the presence of recorded observations at the receiver implies the existence of NLOS signals accompanied by multipath reflections. Consequently, it is appropriate to consolidate or group the labels corresponding to Class 3 together with Class 4.

7

Feature analysis and engineering with respect to Classes

Feature analysis refers to the process of studying and evaluating the importance and relevance of different input features (also known as variables or attributes or predictors) in a dataset. The primary objective of feature analysis is to analyze the relationships between individual features and the target variable (Classes) in order to gain insights into their impact on predictive performance.

On the other hand Feature engineering, in the context of machine learning, involves leveraging insights derived from feature analysis to transform the existing features into more relevant representation, to improve the performance of the model. The objective is to enhance the model's performance by creating new features or modifying existing ones based on a deeper understanding of the underlying data patterns. In this chapter, we will define some of the features that will be used later for machine learning.

The recorded measurements for C1C, L1C, C2W, and L2W are utilized to analyse the features for the GPS constellation. Similarly, the recorded measurements for L1C, C1C, L7Q, and C7Q are used to analyse the features for the Galileo constellation. Due to brevity, this report only presents the corresponding features from L1C and C1C measurements for both GPS and Galileo constellations.

7.1 Range-velocity

Definition 7.1.1 (Range-Velocity)**.** Range-velocity v is defined as the time derivative of the pseudorange ρ.

The left panel in Figure 7.1 displays the Range-velocity (blue curve) on the left axis, and elevation (orange curve) on the right axis, for C1C channel, for both GPS and Galileo. The top panel represents GPS satellite ID 3 and the bottom panel represents Galileo satellite ID 3, across all recorded days. The right panel in Figure 7.1 presents the corresponding histogram of the Range-velocity data from the left panel. This histogram provides an overview of the distribution of range velocities for both GPS and Galileo constellations. Similar plot for L1C channel shows similar behaviour and is not shown here for brevity.

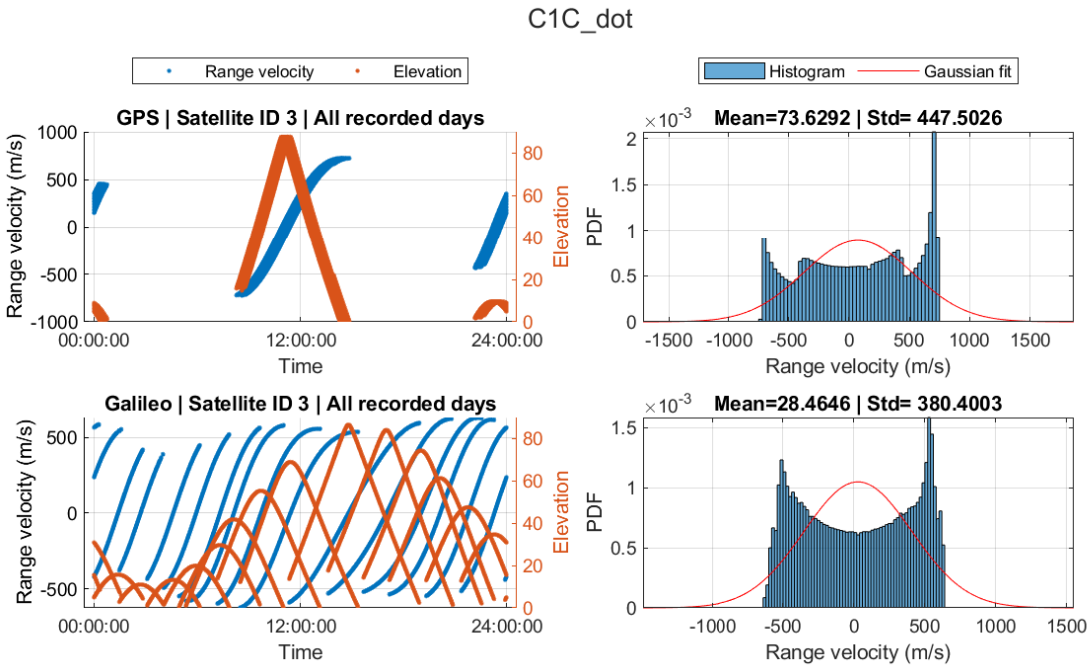


Figure 7.1: Range-velocity and histogram for the **C1C** channel in the GPS constellation (top panel) and the Galileo constellation (bottom panel). In the left set of figures, the blue curve represents Range-velocity on the left axis, while the orange curve represents the elevation on the right axis. The right set of figures corresponds to the histogram of Range-velocity from the left panel.

Observing the plots in the left panel, we notice that the Range-velocity for both GPS and Galileo satellites follows an S-shaped curve (indicated by the blue scattered points). For GPS, the satellite consistently appears in approximately the same azimuth-elevation region every day, shifted in time by 4 min (TBD), resulting in a cluster of S-shaped curves when analyzed in the time domain. In contrast, for Galileo, the same satellite appears at different azimuth-elevation positions and is shifted in the time domain. To further analyze the data, the elevation information is also plotted in the same figure (orange scattered points). This reveals that higher range velocities occur at lower elevations, while zero Range-velocity occurs when the satellite is at its zenith.

7.2 Range-acceleration

Definition 7.2.1 (Range-Acceleration)**.** Range-acceleration a is defined as the time derivative of the Range-velocity v.

Figure 7.2 displays the Range-acceleration for C1C channel for both GPS and Galileo, with the top panels representing GPS satellite ID 3 and the bottom panel representing Galileo satellite ID 3, across all recorded days. The corresponding histogram is presented in the right panel.

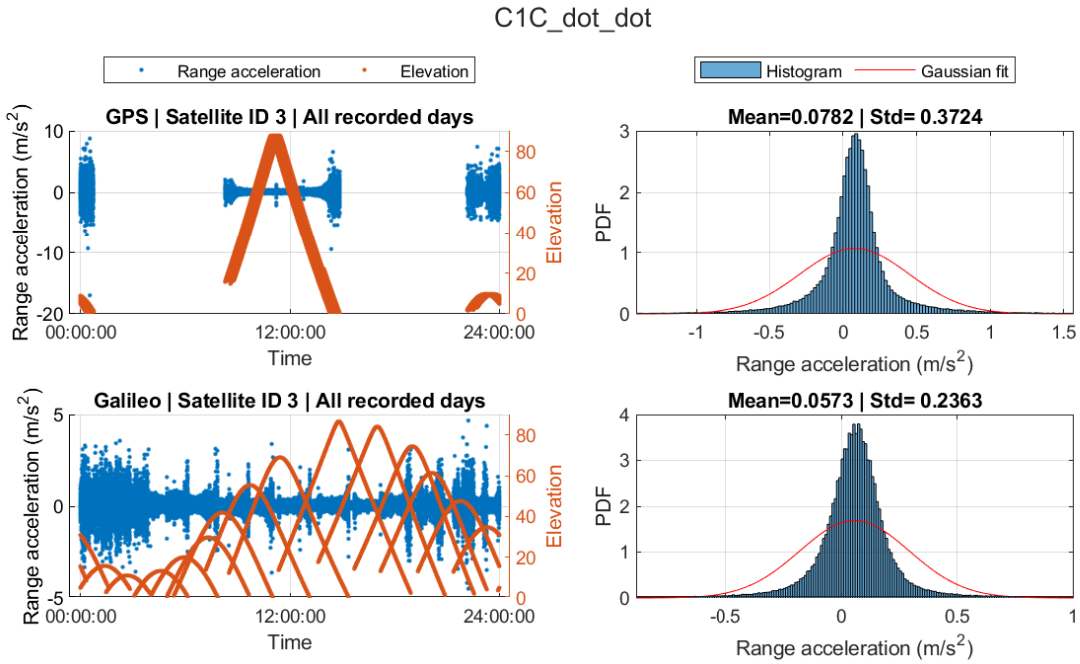


Figure 7.2: Range-acceleration and histogram for the **C1C** channel in the GPS constellation (top panel) and the Galileo constellation (bottom panel). In the left set of figures, the blue curve represents Range-acceleration on the left axis, while the orange curve represents the elevation on the right axis. The right set of figures corresponds to the histogram of Range-acceleration from the left panel.

Upon observing the plots in the left panel, we can notice that the spread in Range-acceleration is higher at low elevations compared to high elevations, for both GPS and Galileo. This is more pronounced for C1C signal than L1C signal (Ref: figure 7.3). As mentioned previously in Chapter 4, figures 4.5 and 4.9 show that the density of low elevation points is significantly higher. In addition to this, we also now know, from figure 7.2 and 7.3, that the standard deviation is higher at low elevation. This suggests that localization performance may be affected in these regions. The histogram in the right panel corresponds to the Range-acceleration from the left panel, which reveals a non-zero mean value. (Note: This mean value disappears when we take differencing between range accelerations, which will be presented later.)

A similar plot for L1C signal is presented in figure 7.3 . The effect of elevation on the spread of the range acceleration is less apparent for L1C signal when compared to C1C signal. Moreover, the distribution looks very close to a Gaussian distribution, as indicated by the red curve. Another noticeable thing is that the mean of range acceleration for L1C and C1C are very similar and this mean disappears when we take the difference which will be presented in the next section.

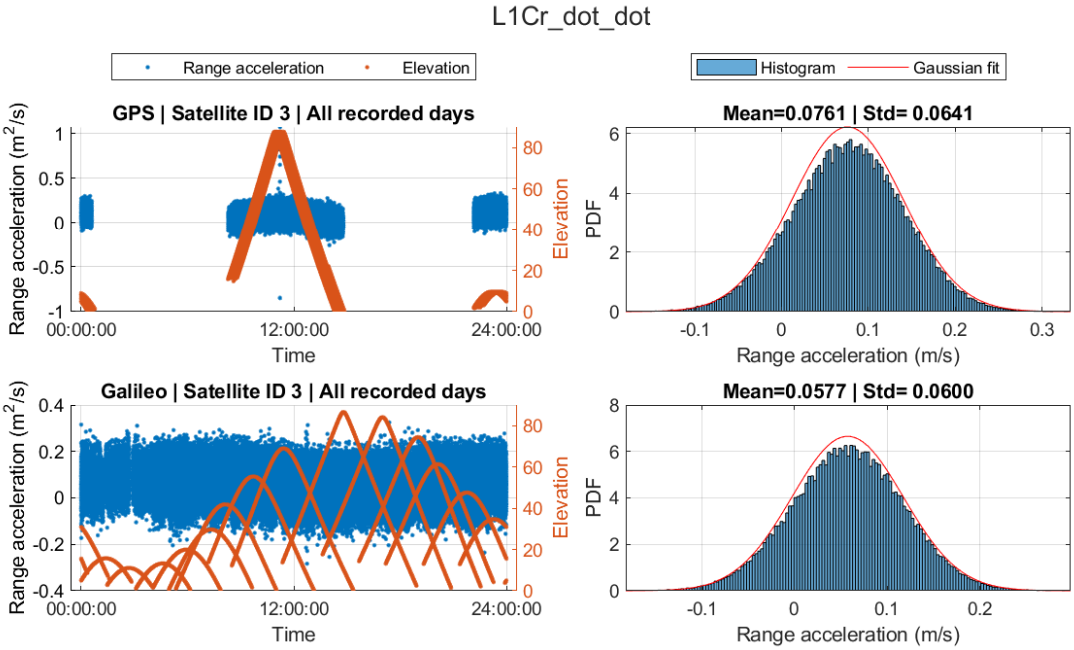


Figure 7.3: Range-acceleration and histogram for the **L1C** channel in the GPS constellation (top panel) and the Galileo constellation (bottom panel). In the left set of figures, the blue curve represents Range-acceleration on the left axis, while the orange curve represents the elevation on the right axis. The right set of figures corresponds to the histogram of Range-acceleration from the left panel.

7.3 Differencing of range-velocity between code and carrier signals

In the previous sections we introduced range velocity and range acceleration for a single satellite. In this section we will look at the differencing of range velocity, and we will consider all the days and all satellites together.

The range-velocities derived from L1C (v_{L1C}^{GPS}) and C1C (v_{C1C}^{GPS}) measurements in the GPS constellation are subtracted from each other. This process is here referred to as the differencing of range-velocities in this thesis. A similar differencing procedure is also applied to the Galileo constellation, and the resulting signals for both GPS and Galileo constellations are listed below:

1. $v_{L1C\_C1C}^{GPS}$: Differencing between range-velocities from L1C v_{L1C}^{GPS} and C1C v_{C1C}^{GPS} for GPS.
2. v_{L2W_C2W} : Differencing between range-velocities from L2W v_{L2W} and C2W v_{C2W} for GPS
3. $v_{L1C\_C1C}^{Galileo}$: Differencing between range-velocities from L1C $v_{L1C}^{Galileo}$ and C1C $v_{C1C}^{Galileo}$ for Galileo.
4. v_{L7Q_C7Q} : Differencing between range-velocities from L7Q v_{L7Q} and C7Q v_{C7Q}

for Galileo

Figure 7.4 presents an example of differencing of the range velocities for both GPS and Galileo constellations. The top panel of figures corresponds to GPS, while the bottom panel corresponds to Galileo. The first column of figures represents histogram of range velocity from L1C, the second column corresponds to histogram of range velocity from C1C, and the last column corresponds to the histogram of differencing of the first two columns. The histogram of the differenced data reveals a dual mode, and after differencing, the mean value becomes approximately zero. The effect of differencing where the mean vanishes, is more pronounced in differencing of range acceleration and has a better distribution, which is explained in the the next section

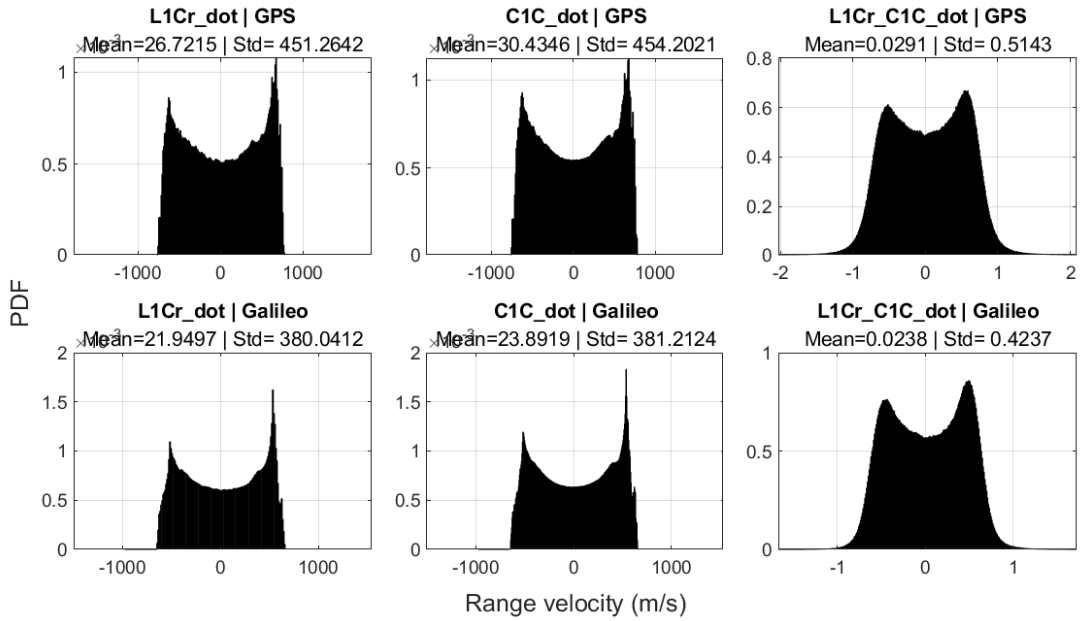


Figure 7.4: An illustration showing the effect of differencing **Range-velocity** on the distribution. The top and bottom panels of figures correspond to the distribution for the GPS and Galileo constellations, respectively. The first and second columns of figures represent the distribution of Range-velocity on the L1C and C1C channels, respectively. The last column represents the distribution after differencing these two Range-velocity.

7.4 Differencing of range-acceleration between code and carrier signals

A similar differencing is performed for range acceleration and the list of signals for both GPS and Galileo constellations calculated are

1. $a^{GPS}_{L1C\_C1C}$: Differencing between range-acceleration from L1C a^{GPS}_{L1C} and C1C a^{GPS}_{C1C} for GPS.

2. a_{L2W_C2W} : Differencing between range-acceleration from L2W a_{L2W} and C2W a_{C2W} for GPS
3. $a^{Galileo}_{L1C\_C1C}$: Differencing between range-acceleration from L1C $a^{Galileo}_{L1C}$ and C1C $a^{Galileo}_{C1C}$ for Galileo.
4. a_{L7Q_C7Q} : Differencing between range-acceleration from L7Q a_{L7Q} and C7Q a_{C7Q} for Galileo

An example of histogram of differencing of range acceleration for L1C and C1C for both GPS and Galileo constellations are shown in figure 7.5. The top panel of figures corresponds to GPS, while the bottom panel corresponds to Galileo. The first column of figures represents histogram of range acceleration from L1C, the second column corresponds to histogram of range acceleration from C1C, and the last column corresponds to the histogram of differencing of the first two columns. After differencing, the mean value becomes approximately zero.

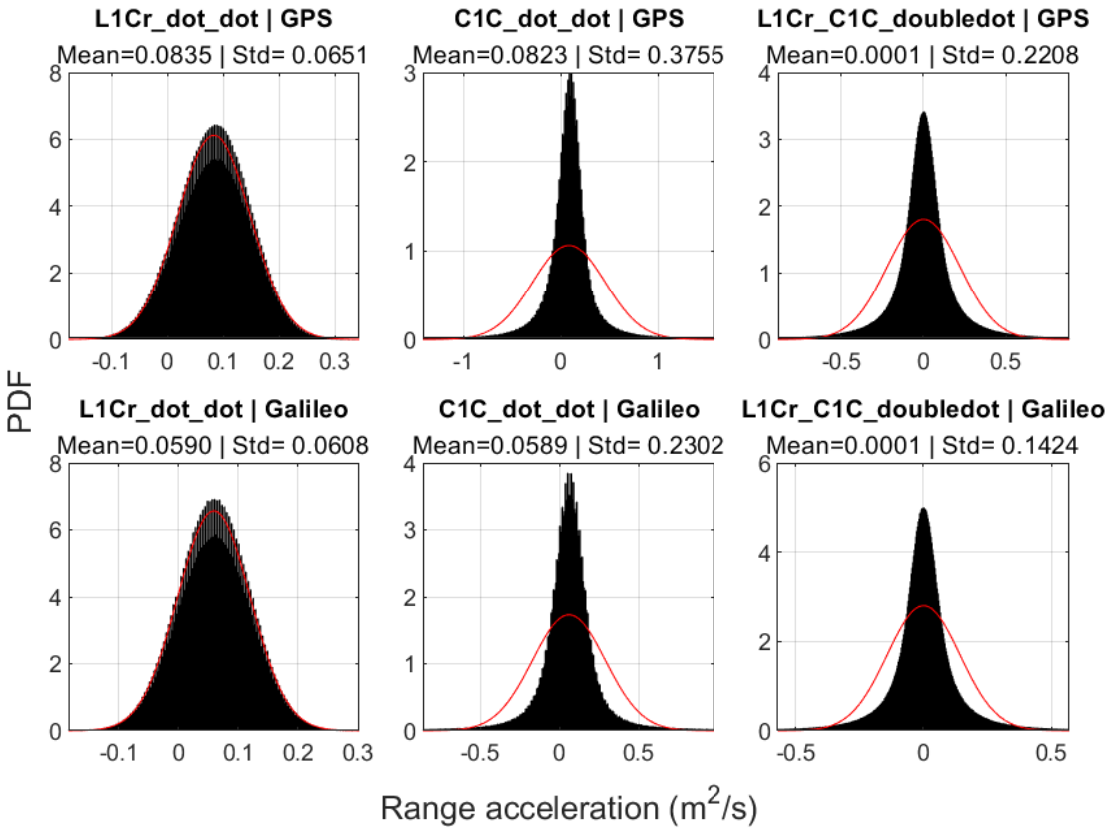
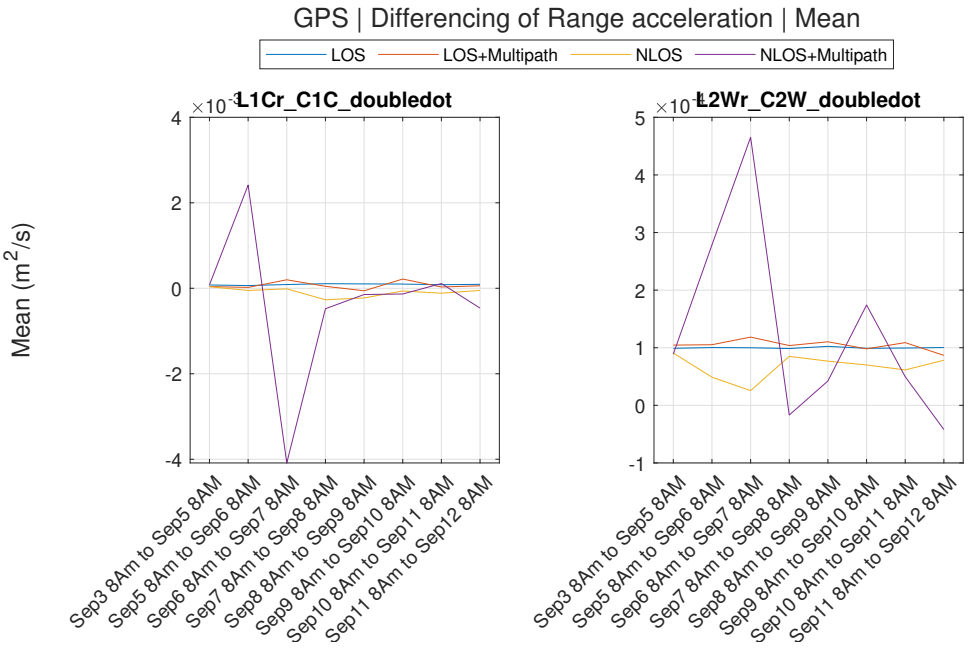


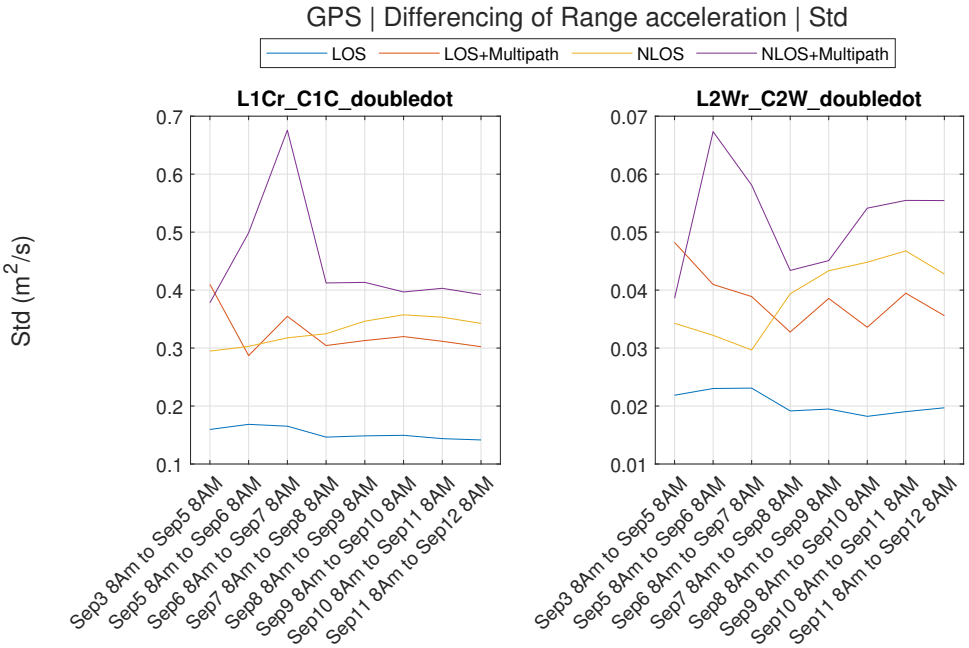
Figure 7.5: An illustration showing the effect of differencing **Range-acceleration** on the distribution.The top and bottom panels of figures correspond to the distribution for the GPS and Galileo constellations, respectively. The first and second columns of figures represent the distribution of Range-acceleration on the L1C and C1C channels, respectively. The last column represents the distribution after differencing these two Range-acceleration.

In the previous figures, we analyzed the distribution of features for all recorded

days combined. However, since the measurement setup varies for different days, it is more appropriate to analyze them separately for each day. Additionally, we can leverage the results from ray tracing and analyze the distribution for each class. A summary of statistics for the differencing of range accelerations for GPS and Galileo on different days, categorized by classes, is presented in Figures 7.6 and 7.7, respectively. Observing the figures, we can notice a similar pattern for both GPS and Galileo constellations. Both show a mean value close to zero (see top panel of the figures 7.6 and 7.7), and the standard deviation (see Bottom panel of the figures 7.6 and 7.7) for LOS is the lowest, followed by classes with multipath components.

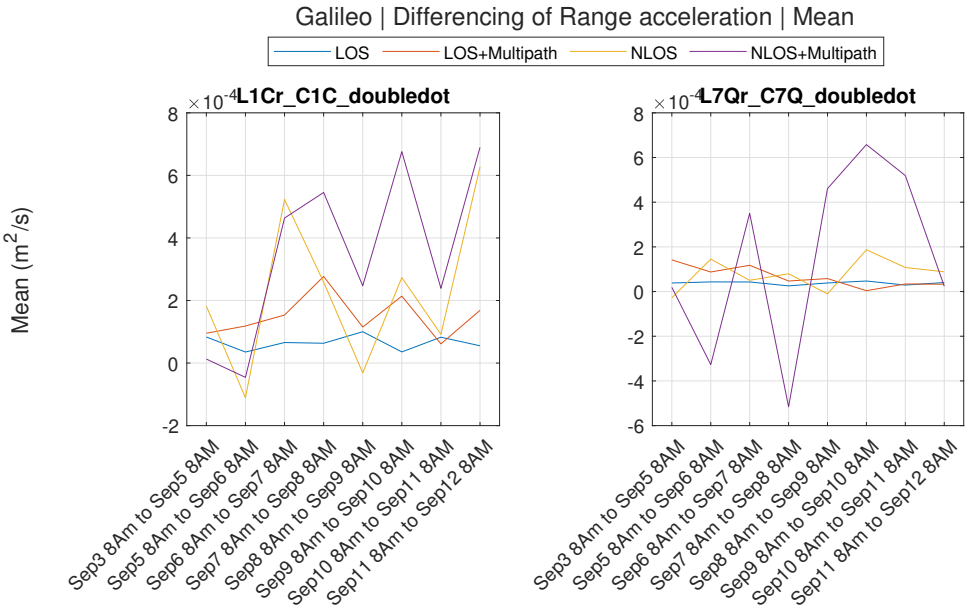


(a) Mean of the differencing of Range-acceleration.

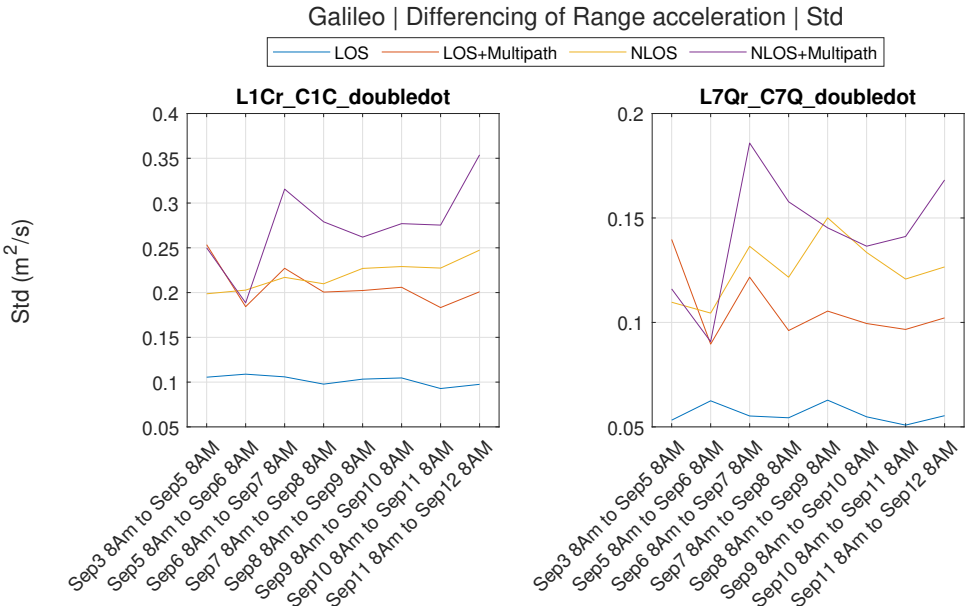


(b) Standard deviation of the differencing of Range-acceleration.

Figure 7.6: GPS: Summary of statistics of differencing of Range-acceleration. Panel a and b shows the mean and standard deviation for each day for different class. In each panel the left and right side figure is for differencing of range acceleration between L1C-C1C and L2W-C2W, respectively.



(a) Mean of the differencing of Range-acceleration.



(b) Standard deviation of the differencing of Range-acceleration.

Figure 7.7: Galileo: Summary of statistics of differencing of Range-acceleration. Panel a and b shows the mean and standard deviation for each day for different class. In each panel the left and right side figure is for differencing of range acceleration between L1C-C1C and L7Q-C7Q, respectively.

To complement the statistics presented above, we provide an example of the differencing of range acceleration for the GPS constellation for a single day in a polar plot, as shown in Figure 7.8. In this matrix figure, the first and second columns represent the range acceleration of L2W and C2W, respectively, while the third column represents the differencing of the first two columns.

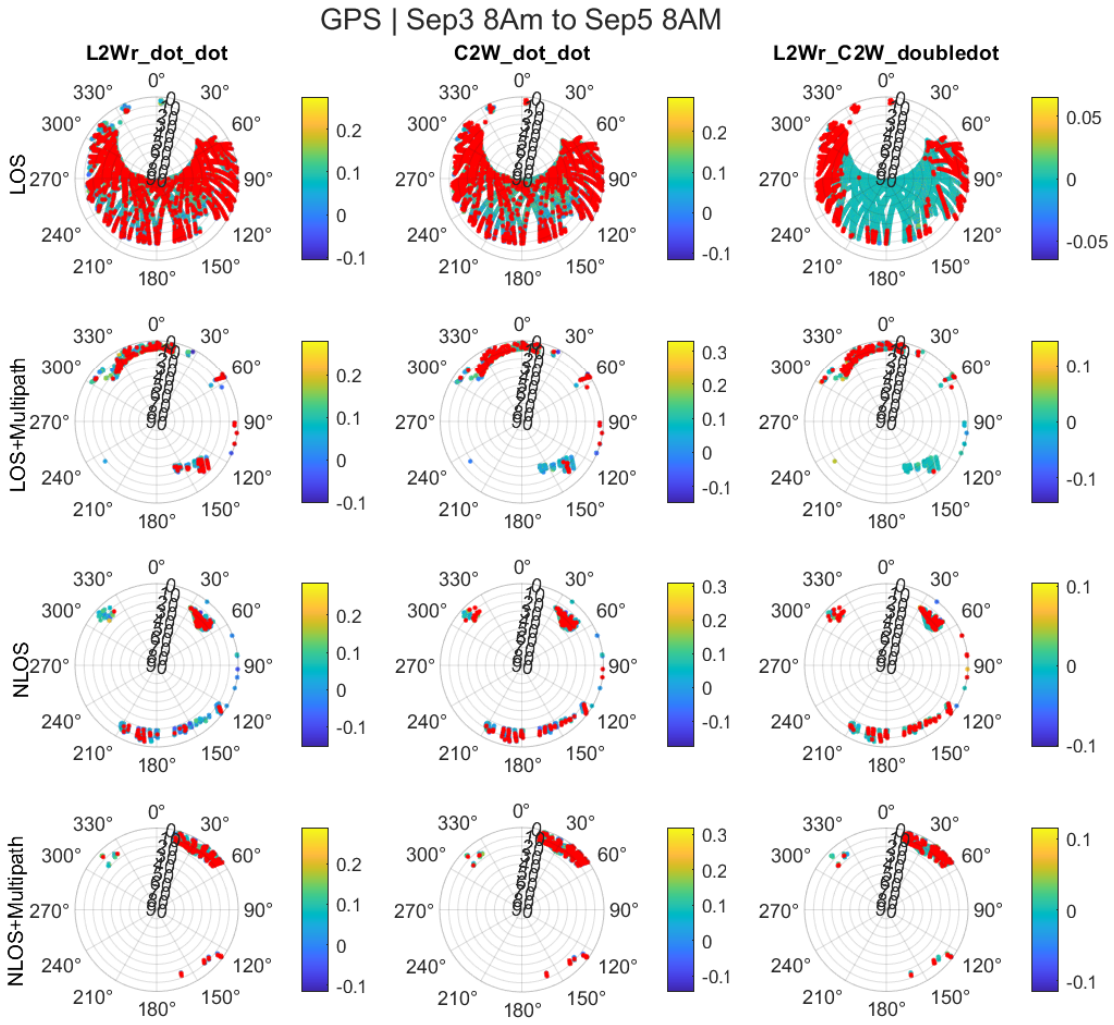


Figure 7.8: An illustration of the effect of differencing of Range-Acceleration in polar plot. Each row corresponds to different classes. The first and second column corresponds to Range-acceleration on L2W and C2W channel respectively for different classes. The last column corresponds the differencing of the first two columns. The colorbar in each plot corresponds to 3-Sigma region. The red marker in each plot corresponds to data points beyond 3-Sigma region.

In each subfigure, the color bar represents the 3-sigma region, which is a relevant choice considering the distribution of range acceleration closely resembles a Gaussian distribution. The red marker are for those points that are beyond the 3-sigma region. One striking observation is apparent in the subfigure representing the differenced range acceleration for the LOS label (located in the first row, third column). In this case, we can observe that data points outside the 3-sigma region are primarily concentrated at low elevations. This indicates that the differencing process effectively emphasizes the influence of low elevations on the range acceleration. Also

note that, similar differencing for other signals and constellations is presented in the appendix. However, the distinct differences observed after the differencing process are more pronounced for GPS and for the L2W and C2W signals, in comparison to other signals and constellations.

7.5 Indicator variable

In the previous section we saw that most of the data points that are outside the 3 sigma region of the 'differencing of range-acceleration of L1C and C1C', occur at low elevation. In order to capture this information and feed it to the machine learning model, we create a indicator variable which takes on a value 1 if the data point is outside the 3-sigma region else 0 ($\mathbb{1}_{\{L\ddot{1}C-C\ddot{1}C>3\sigma\}}$). This is done to help the machine learning algorithm to facilitate more accurate classifications.

7.6 Signal strength indicator

Signal strength indicator(SSI) indicates the the strength of satellite signals received by a receiver. The SSI provides an indication of how strong or weak the received satellite signals are. A higher SSI typically suggests a stronger and more reliable signal, while a lower SSI may indicate weaker signal strength, potentially leading to issues like signal dropout or reduced accuracy in positioning. In the original measurements signal dropouts are indicated with NaN values which are replaced with 0 during post processing as shown in figure 7.9

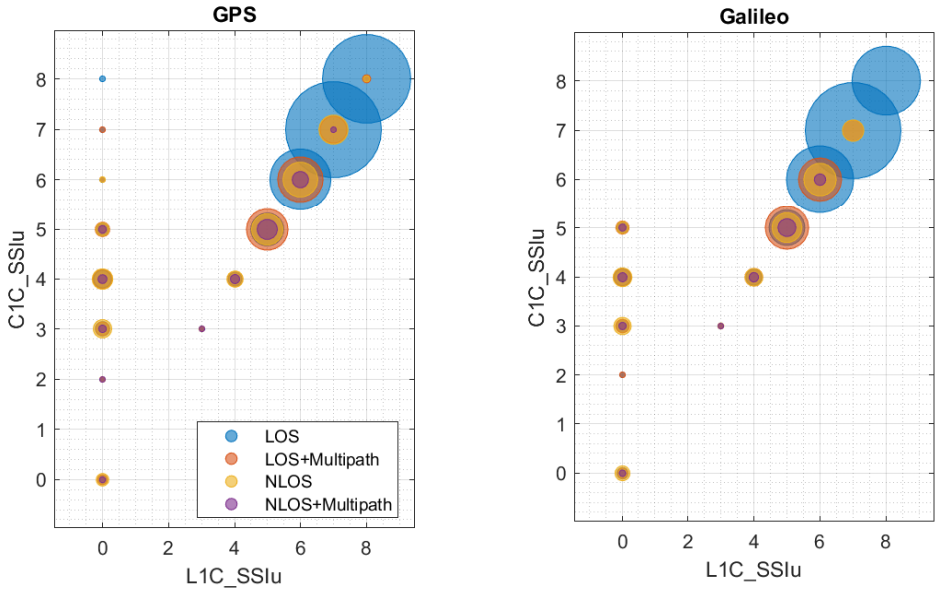


Figure 7.9: SSI of L1C channel on the x-axis and C1C channel on the y-axis for GPS (left figure) and Galileo (right figure), respectively. The size of the bubbles represents the total number of data points in each class.

The data presented in Figure 7.9 reveals generally robust Signal Strength Indicators (SSI), with the majority of signals registering above 5 SSI. This is due to the fact that the the data is captured in an open field environment. The figure also illustrates a trend where LOS signals tend to exhibit higher SSI values, while NLOS signals are associated with comparatively lower SSIs. An interesting detail is observed on the y-axis, indicating instances of signal dropout on the L1C signal despite maintaining a fairly good SSI on the C1C signal.

7.7 Difference of Signal strength indicator

The "Difference of Signal Strength Indicator" refers to the variation in SSI at time t, relative to the same signal at the previous time instance $t-1$. This metric provides insights into how the Signal Strength Indicator (SSI) changes from one data point to the next. This is shown in Figure 7.10 which indicates a well-defined measurements and is because the measurements were done in an open field environment.

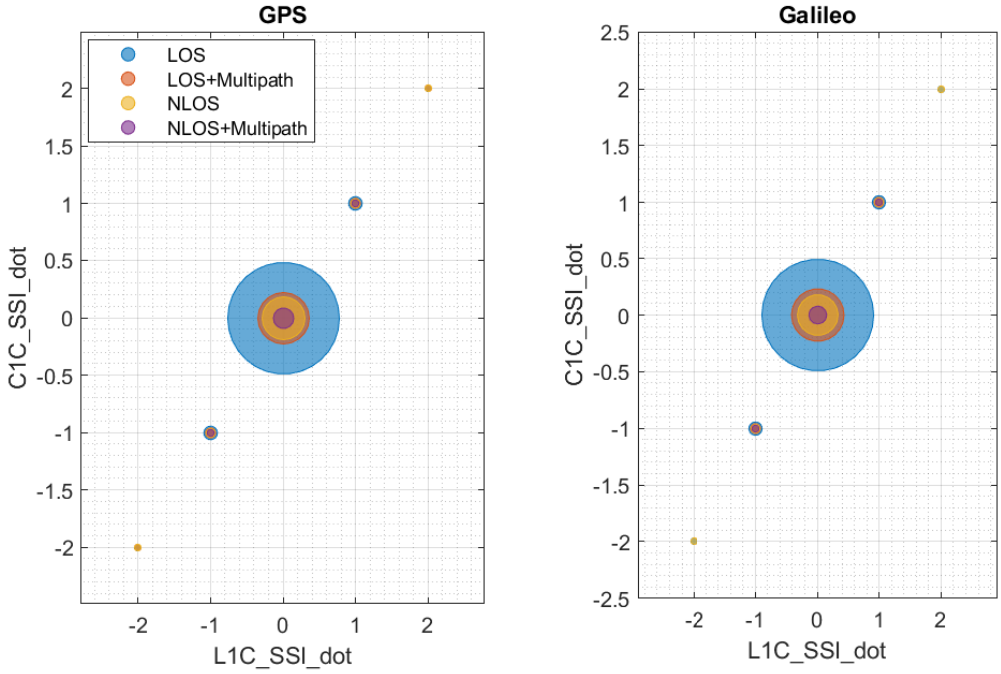


Figure 7.10: The difference of SSI of L1C channel on the x-axis and C1C channel on the y-axis for GPS (left figure) and Galileo (right figure), respectively. The size of the bubbles represents the total number of data points in each class.

8

Support vector machine

Support Vector Machine (SVM) is a powerful and versatile machine learning algorithm used for classification tasks. It is primarily designed as a binary classifier algorithm, capable of distinguishing between two classes. It is a supervised learning method that analyzes and sorts data into distinct classes based on labeled training data. The core idea behind SVM is to find the optimal hyperplane that best separates data into different classes. This hyperplane aims to maximize the margin (the distance between the hyperplane and the nearest data points of each class), ensuring a clear separation. It has gained popularity because of its effectiveness in handling high-dimensional data, robustness in the presence of noisy data, and the ability to capture complex patterns using different kernel functions.

8.1 Dataset

As mentioned before, the original measurements, and consequently the raytracing results, are captured at 1-second intervals over 11-day period, providing a wealth of data for supervised training. However, processing such a large amount of data can be time-consuming. To expedite the training process, this thesis leverages the results generated by GipsyX, which are computed at 30-second intervals. These GipsyX results are then synchronized with the raytracing outcomes. This synchronization involves assigning each data point in the residuals, obtained from GipsyX, with corresponding raytracing results and also include the range measurements from the original dataset. In essence, this synchronization procedure downsamples the original measurements from a 1-second sampling rate to a 30-second sampling rate and includes range measurement's from all channels along with residuals.

Initially, SVM models were trained using both the original and downsampled datasets. It was observed that training with the original dataset incurred significant time costs with only marginal improvements in model performance. Consequently, subsequent experiments, such as feature addition or removal, were performed exclusively with the downsampled dataset.

The data from the GPS and Galileo constellations are normalized independently using z-score normalisation and then concatenated vertically. Figure 8.1 illustrates the total number of data points in each class within the downsampled dataset. Each bar represents the count of data points for each class on specific days. The numerical values displayed above each bar denote the total count of data points in each class.

It is appropriate to re-iterate here that NLOS Class and NLOSplusMultipath shown in the previous chapter are consolidated into one class called NLOS (class3).

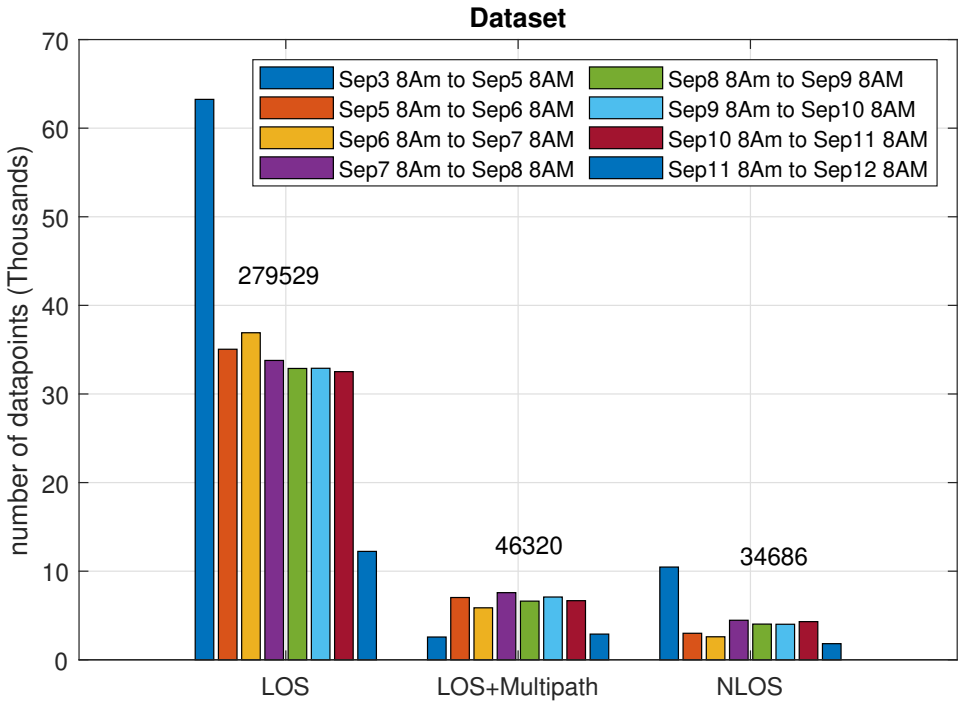


Figure 8.1: Total number of data points in each class across different days within the downsampled dataset.

8.1.1 Cleaning the Data set

In the original measurements obtained from the receiver, there are instances where certain time epochs contain NaN values. This typically occurs when the signal strength is exceptionally low or, in some cases, insufficient for proper measurement. Such data points associated with NaN values are removed from the downsampled dataset. The statistics after cleaning the data is shown in figure 8.1

8.1.2 Handling imbalanced Data set

As previously illustrated in figure 8.1, the down sampled data set is inherently imbalanced, meaning that one class has significantly more instances than the other class. The class with the majority of instances (LOS in this case) is referred to as the majority class, and the class with the fewer instances (NLOS+Multipath) is called the minority class. Machine learning algorithms tends to perform poorly on imbalanced dataset, in that they become biased towards majority class. Addressing imbalanced datasets often involves techniques like oversampling the minority class,

undersampling the majority class, or using more advanced methods like Synthetic Minority Over-sampling Technique (SMOTE).

In this thesis we have used undersampling of the majority classes, where in the portion of data points from each day, belonging to the majority classes, are randomly removed. Additionally, we strategically applied undersampling to the 'LOS+Multipath' data on selected days, ensuring that we avoid excessive undersampling on days with already low numbers of data points. Figure 8.2 shows the balanced data set from the point of view of total number of points in each class (and not each day).

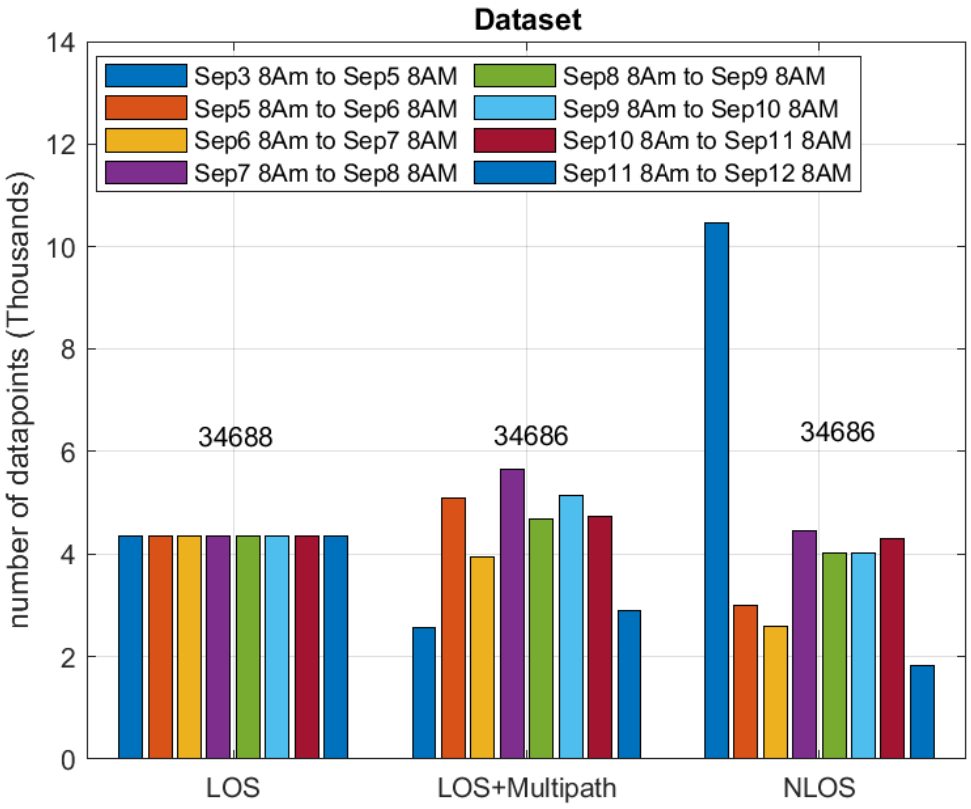


Figure 8.2: Total number of data points in each class across different days in the balanced dataset

8.2 Feature selection

Based on the insights obtained from the previous chapter specific features have been selected. The features used for the support vector machine are outlined in Table 8.1.

Features	Description
SSI_{C1C}	Signal strength indicator of C1C signal
ΔSSI_{C1C}	Differencing of C1C SSI
SSI_{L1C}	Signal strength indicator of L1C signal
ΔSSI_{L1C}	Differencing of L1C SSI
ρ_{C1C}	Range on C1C signal
$\dot{\rho}_{C1C}$	Range velocity on C1C
$\ddot{\rho}_{C1C}$	Range acceleration on C1C
ρ_{L1C}	Range on L1C signal
$\dot{\rho}_{L1C}$	Range velocity on L1C
$\ddot{\rho}_{L1C}$	Range acceleration on L1C
$\rho_{L1C} - \rho_{C1C}$	Difference of L1C and C1C
$\dot{\rho}_{L1C} - \dot{\rho}_{C1C}$	Difference of range velocities of L1C and C1C
$\ddot{\rho}_{L1C} - \ddot{\rho}_{C1C}$	Difference of range acceleration of L1C and C1C
$\mathbb{1}_{\{L\ddot{1}C - C\ddot{1}C > 3\sigma\}}$	Indicator variable for difference of range acceleration beyond the 3 sigma region

Table 8.1: Features used for the SVM.

8.3 SVM for multiclass classification

The Support Vector Machine (SVM) is primarily designed as a binary classifier algorithm, capable of distinguishing between two classes. However, when dealing with multiclass classification in MATLAB, the 'fitcecoc' function is used. This function implements an Error Correcting Output Codes (ECOC) model, which transforms the multiclass classification problem into a set of binary classification problems.

In this algorithm we use 'One-Versus-One' method where in $\frac{k(k-1)}{2}$ binary classifier is used to predict the class for a given sample x_i. All this handled within matlab fitcecoc function. The RBF kernel is used for training a support vector machine. The coding matrix used is shown in table 8.2, where f_1, f_2, and f_3 serve as three distinct binary classifiers. The entries within the matrix indicate the positive and negative classes against which each binary classifier is trained. The algorithm then assigns a observation x_i to a class K that minimizes the aggregation of the losses for the 3 binary learners.

Table 8.2: Coding Matrix for One-Versus-One Classification

Class	f_1	f_2	f_3
Los	1	1	0
LosPlusMultipath	-1	0	1
Nlos	0	-1	-1

8.4 k-Fold cross validation

K-fold cross-validation, with K=5, is employed to evaluate the performance and generalization capability of the SVM model. The process is iterated 20 times, with each iteration involving the generation of a new balanced dataset and the application of stratified K-Fold cross-validation. The Radial Basis Function (RBF) kernel is utilized as the kernel function. Figure 8.3 illustrates the average hinge loss versus K-folds for each of the 20 iterations. The loss are consistent across K-Folds and different iterations which suggests a robust generalisation of the model and dataset.

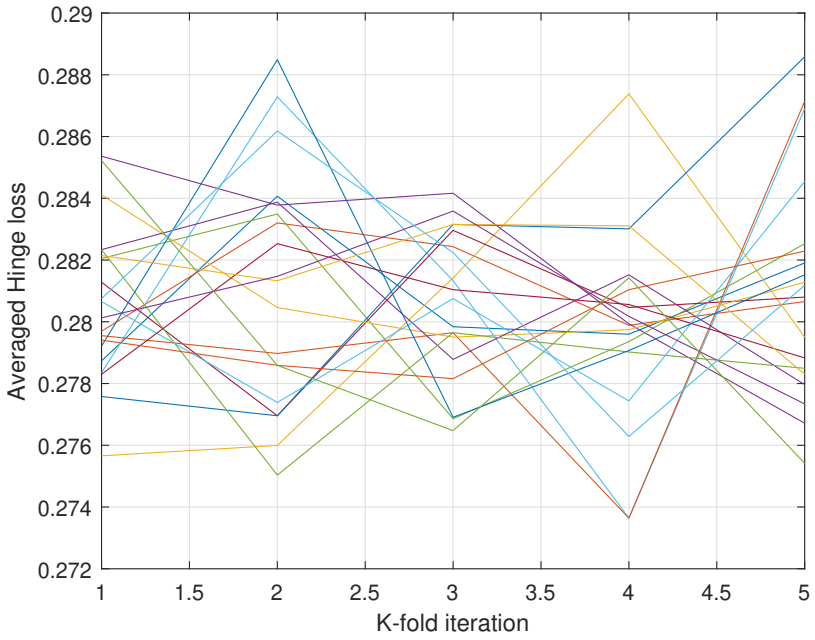


Figure 8.3: Averaged hinge loss across K-Fold where K=5 for 20 different iterations.

8.5 Result

The balanced dataset, is split into stratified training and test data, with a train-test split ratio of 0.8:0.2. After this the data is shuffled. Following the training, classes are predicted on the test data. The confusion matrix for both the training and test data is depicted in Figure 8.4.

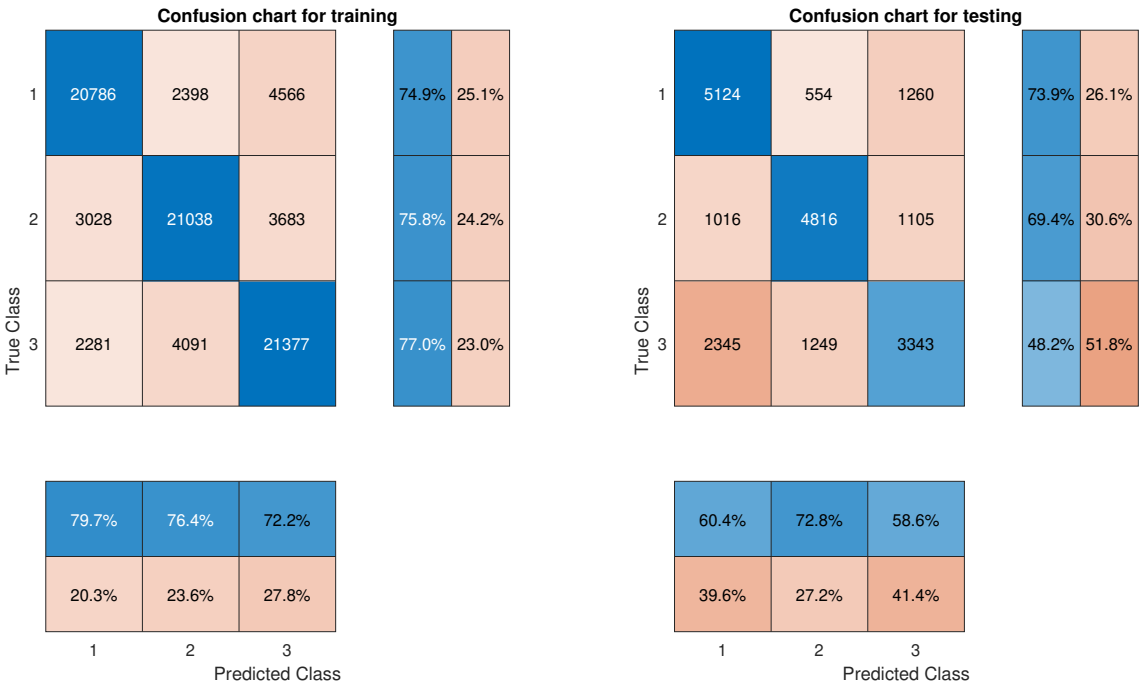


Figure 8.4: Confusion chart for training and test dataset.

Each confusion chart presented in figure 8.4 comprises three components:

1. **Confusion Matrix:** Located in the center, this matrix represents the classification results. Rows correspond to true classes, and columns correspond to predicted classes. Diagonal cells indicate correct classifications, while off-diagonal cells represent misclassifications.

2. **Row-Normalized Row Summary:** Positioned to the right of the confusion matrix, this summary presents the percentages of correctly and incorrectly classified observations for each **true class**. For example, in the confusion chart for training, 74.9 % of the 'true' class 1 data points were correctly predicted, while 25.1 % of class 1 data points where wrongly predicted. It can also be thought of as class wise recall.

3. **Column-Normalized Column Summary:** Displayed at the bottom, this summary showcases the percentages of correctly and incorrectly classified observations for each **predicted class**. For example, in the confusion chart for training, 79.7 % of all data points 'predicted' as class 1 were correct, whereas 20.3 % those prediction were incorrect. It can also be thought of as class wise precision.

Overall the testing performance is lower than on the training dataset especially for class3 which could suggest over-fitting. To complement this result, the prediction on training and test data are presented in polar plot in figure 8.5

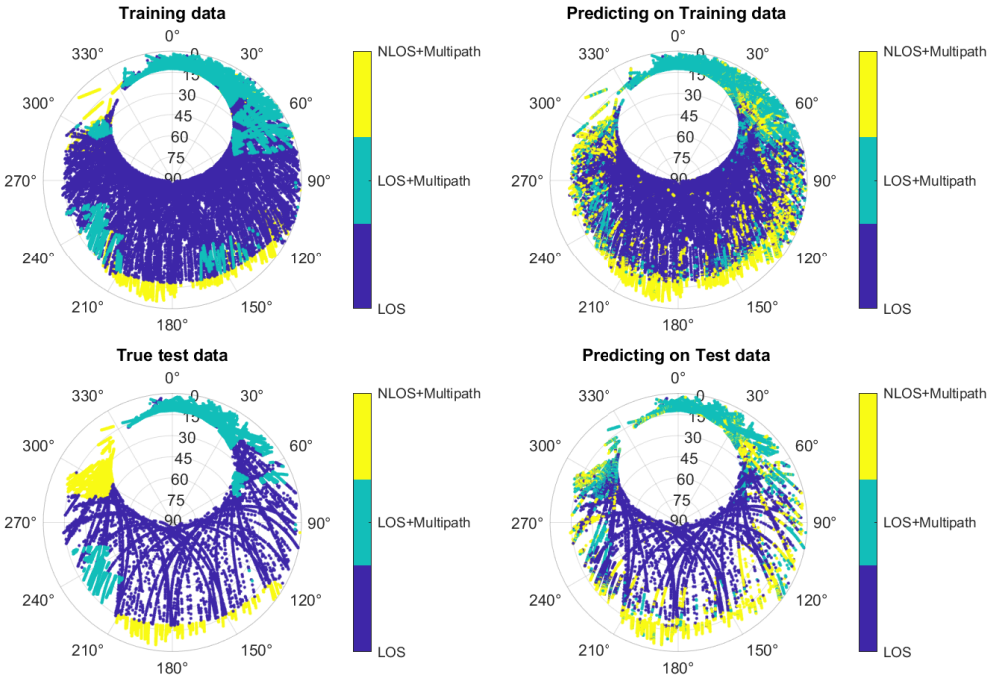


Figure 8.5: True and predicted classes for training (top row) and for testing (bottom row) in polar plot.

9

CNN based Auto encoders with SVM for Multi-class classification

Autoencoders are a type of artificial neural network architecture designed for unsupervised learning, particularly in the field of feature learning and data compression. The primary objective of an autoencoder is to learn a compact and efficient representation, or encoding, of input data. It does so by training the network to reconstruct its own input. In this thesis, we will use Autoencoders to learn a set of new features using 14 features which we used in the previous chapter.

The architecture of an autoencoder consists of an encoder and a decoder. The encoder transforms the input data into a compressed representation, which captures the essential features of the input. The decoder then reconstructs the original input from this compressed representation. The network is trained by minimizing the difference between the input and the reconstructed output, typically using a loss function such as the mean squared error.

After the training phase, only the encoder part of the Autoencoder is integrated with a classification layer, which is then re-trained for classification. In this second stage of training, only the parameters associated with the classification layer are fine-tuned to minimize the classification loss.

In this chapter, we will model an Autoencoder using 1D Convolutional Neural Network (1D-CNN), complemented by a Support Vector Machine (SVM) for classification purposes. Given the CNN's ability to learn from neighboring data points, we conduct three distinct simulations, each varying in the method of sorting the data before being fed into the Autoencoder, which will be detailed in the coming section.

9.1 Dataset and features

The preprocessing steps, which include cleaning, z-score normalization, concatenation of data from GPS and Galileo constellations, and handling of imbalanced datasets, remain consistent with the procedures outlined in the previous chapter. Furthermore, we conduct three simulations employing the same structure of autoencoders and Support Vector Machine (SVM), with the only variation being the manner in which the data is sorted before being fed into the Autoencoder.
In the first simulation, the data are sorted according to time, and in the second sim-

ulation, the data is sorted with satellites first and then time to break ties. Lastly, in the third simulation, the data are sorted according to classes. The way the data are sorted leads to different results, primarily due to the 1D-convolution layer used in the Autoencoder. This distinction will become more apparent as we detail the layers in the Autoencoder in the next section.

To continue the discussion, in the first simulation where data is sorted by time, the filter assimilates features from different satellites that are visible at that time instant. Since the chosen filter size is 7, it can learn from features of up to 7 visible satellites available at that specific time. If there are fewer visible satellites than 7, then it encounters the features from the satellites that are visible from the next time instance. In the second simulation, where the data is sorted first by satellite and then by time, the filter encounters features from only a single satellite. In the third simulation where the data is sorted by classes, the filter encounters features of the same class. This difference in the features seen by the filter influences the learned representations and, consequently, the subsequent predictions from the Support Vector Machine (SVM).

9.2 Autoencoder structure

The table 9.1 provides a description of the layers in an autoencoder model that is used in this chapter. Since the 1-D convolution layer operates in a time dimension, the manner in which the data are sorted before being fed into the encoder impacts the learned features. This is especially true due to the fact that CNNs are particularly well-suited for tasks where the relationship between neighboring data points is crucial. This, in turn, influences the classification results from the SVM layer, as will be demonstrated in the results section.

Layer	Type	Description	Details
1	Seq_input	Sequence Input	Sequence input with 14 dimensions
2	Conv1d	1-D Convolution	64 filters of size 7 with stride 1 and padding 'same'
3	Trans_Conv1d_1	1-D Transposed Convolution	64 filters of size 7 with stride 1, and cropping 'same'
4	Trans_Conv1d_2	1-D Transposed Convolution	14 filters of size 7 with stride 1, and cropping 'same'
5	Reg_Output	Regression Output	Mean-squared-error

Table 9.1: Description of Layers in the Autoencoder.

9.3 Svm for classification

After training the autoencoder, the decoder part is discarded, and the encoder part is integrated with the SVM for the final training phase. During this stage, the tuning

of parameters is done exclusively on those related to the SVM. The SVM employed for multiclass classification remains consistent with the one detailed in the previous chapter. It utilizes the One-versus-One method, the coding matrix, and the Radial Basis Function (RBF) kernel, as described in the previous chapter.

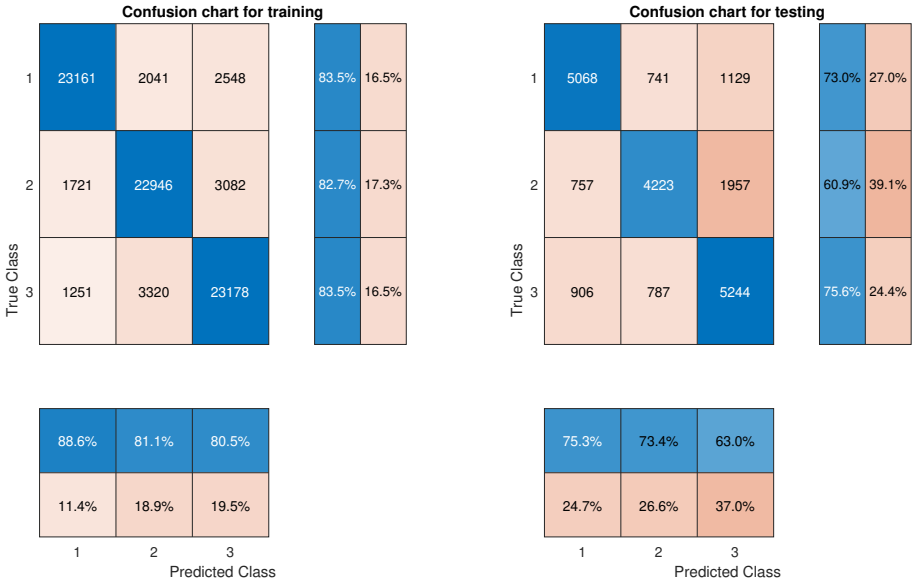
9.4 Results

The training progress of the Autoencoder for all the simulation is shown in Table 9.2. Adam solver is used and the training is limited to 300 epochs. The weights that give the lowest validation loss are saved as the final weights.

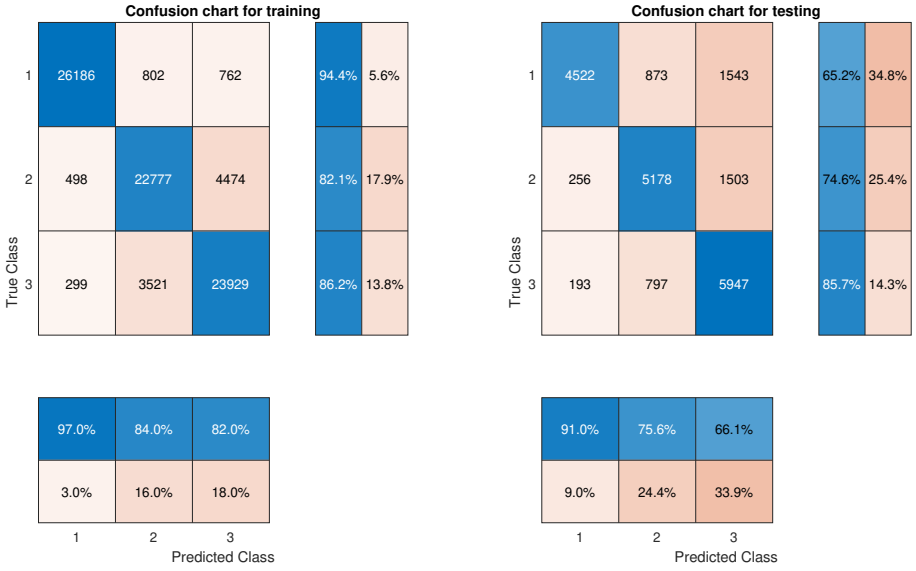
Epoch	**Iteration**	**Mini-batch RMSE**			**Validation RMSE**		
		Sim1	**Sim2**	**Sim3**	**Sim1**	**Sim2**	**Sim3**
1	1	5.75	6.10	5.74	5.48	9.05	8.07
25	25	0.99	1.64	2.32	0.97	1.72	1.93
50	50	0.34	0.68	0.72	0.33	0.67	0.66
75	75	0.16	0.40	0.38	0.16	0.40	0.35
100	100	0.11	0.31	0.28	0.11	0.31	0.27
125	125	0.09	0.26	0.23	0.09	0.27	0.22
150	155	0.07	0.23	0.19	0.08	0.23	0.18
175	175	0.06	0.60	0.17	0.07	0.63	0.16
200	200	0.06	0.22	0.15	0.07	0.24	0.14
225	225	0.05	0.17	0.13	0.06	0.18	0.13
250	250	**0.05**	0.15	0.12	**0.06**	0.16	0.11
275	275	0.13	0.14	0.11	0.18	0.15	0.10
300	300	0.44	**0.13**	**0.10**	0.62	**0.14**	**0.10**

Table 9.2: Training Progress Metrics

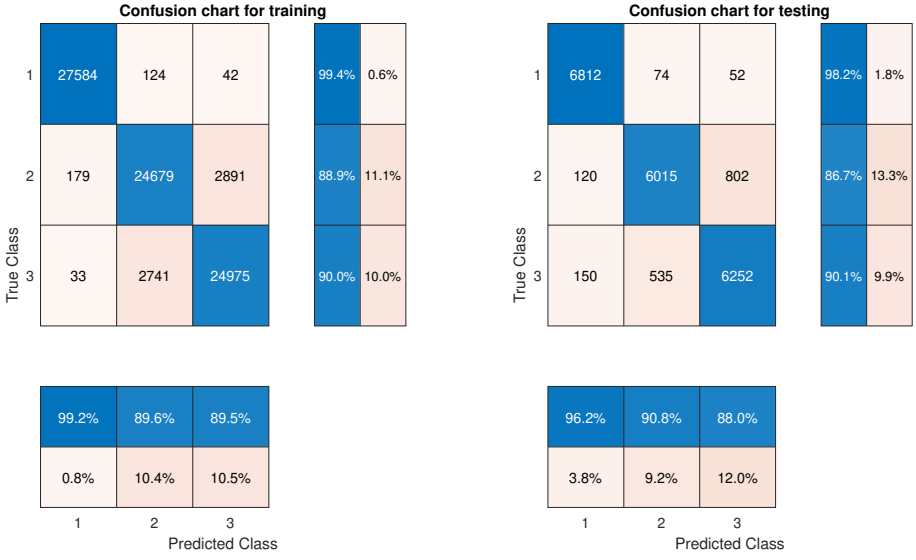
After training the Autoencoder, the extracted features from the encoder(Layer2) are input into the One-Versus-One SVM multiclass classifier. The results are illustrated using a confusion chart and is shown Figure 9.1.



(a) Sorted based on time



(b) Sorted by satellite and then time



(c) sorted by class

Figure 9.1: Confusion chart for simulation with different neurons in the bottleneck layer.

In the initial simulation, where data is sorted by time, we achieve an average of approximately 85% for both class-wise recall and precision during training. However, it struggles to generalize effectively on the test dataset. In the subsequent simulation, where data is sorted by satellite first and then time, we observe improvement in metrics, particularly for the LOS class (class 1). Nevertheless, it still faces challenges in generalization on the test set.

In contrast, in the third simulation, where data is sorted by classes, we achieve approximately more than 90% average recall and precision during both training and testing. This notable improvement can be attributed to the synergistic relationship between the sorting methodology and the CNN autoencoder. The improvement observed in the third simulation can be largely attributed to CNNs ability to capture patterns when the negihobouring data points share similarities, similar to the challenges encountered in image recognition tasks where CNNs excel. By sorting the data by classes, we effectively provide the CNN autoencoder with a structured input that matches its strengths. However, in practice, we often don't know the classes beforehand, which limits its usefulness.

Additional simulations similar to those detailed in this chapter were also conducted and are outlined in Appendices B and C. In Appendix B, a dense layer is used instead of a CNN layer in the Autoencoder, and we explore different number of neurons in the bottle neck layer. In Appendix C, we use the same Autoencoder structure with CNN as detailed in this chapter but using only 3 features instead of 14.

9.5 Conclusion

This chapter presented three simulations using the same model, but with distinct data sorting methodologies before inputting it into the model. In the first simulation, where data was sorted based on time, the model accessed features from multiple satellites concurrently. The filter size set to 7 enabled the model to learn from up to 7 different visible satellites simultaneously at a given time instant. If fewer satellites were visible, the model incorporated features from the next time instant's visible satellites.

In contrast, the second simulation involved sorting the data first by satellite and then by time to resolve ties. In this scenario, the model had access to features from only one satellite at any given time. The model's learning process was constrained to information from a single satellite.

In the third simulation, where the data was sorted by classes, the filter had access to similar features from different satellites but of the same class at a given time instant. This favored the CNN since it excels when neighboring data points share relationships. As a result, this simulation outperformed the results of the other simulations. Also, the confusion chart of the first and second simulation indicates more overfitting when compared to the results from the third simulation .

In the Subsequent chapter, we will extend our exploration by constructing a model that considers features from all 31 GPS satellites at a given time instant. This model aims to predict the classes for each of the 31 satellites at a given time instant which transforms the problem at hand from Multi-class classification to Multi-Label-Multi-Output classification. Before jumping into Multi-Label-Multi-Output classification, we provide a brief overview of different classification problem that are discussed in this thesis and how they differ.

10

Overview of different classification problems

In this chapter we provide a brief overview of the different classification problems that are discussed in this thesis. In particular we attempt to provide some mathematical definitions and examples for four different classification problems,

1. Binary classification
2. Multi class classification
3. Multi label classification
4. Multi-label-multi-output classification

10.1 Binary classification

Definition 10.1.1 (Binary classification)**.** Given a labeled training dataset $\mathcal{D} = \{(x_i, y_i)\}$, where each $x_i \in \mathbb{R}^d$ represents a feature vector in d-dimensional real space and each y_i takes values in one of the two classes, i.e., $y_i \in \{C_1, C_2\}$, the task is to learn a function $f : \mathbb{R}^d \times \mathbb{R}^p \to \{C_1, C_2\}$, where $\mathbb{R}^p$ is a parameter space.

In a binary classification problem, the task is to distinguish between two classes based on input data. Table 10.1 illustrates an example of typical data for a binary classification problem. Given the input features x_1 to x_d, the goal is to predict whether the data point belongs to class C_1 or C_2. For instance, we can use features x_1 to x_{14} to predict, for example, whether the features correspond to Line of Sight (LOS) or not.

Table 10.1: Typical input data for Binary Classification problem

Time	x_1	x_2	.	.	x_d	**y**
1	2.5	1.0	.	.	3.1	C_1
2	1.8	0.5	.	.	2.5	C_2
.	.	.	.	.	.	.
.	.	.	.	.	.	.
N-1	2.8	1.5	.	.	5.4	C_2
N	2.8	1.5	.	.	5.4	C_1

10.2 Multi class classification

Definition 10.2.1 (Multiclass Classification)**.** Given a labeled training dataset $\mathcal{D} = \{(x_i, y_i)\}$, where each $x_i \in \mathbb{R}^d$ represents a feature vector in d-dimensional real space and each y_i takes only one value in a set of multiple classes, i.e., $y_i \in \{C_1, C_2, \ldots, C_k\}$, the task is to learn a function $f : \mathbb{R}^d \times \mathbb{R}^p \to \{C_1, C_2, \ldots, C_k\}$ where $\mathbb{R}^p$ is a parameter space.

In a Multi class classification problem, the task is to distinguish between more than two classes based on input data. Table 10.2 illustrates an example of typical data for multi class classification problem. It is similar to 10.1 with only difference being that there are more than two classes. This is the type of classification problem we have encountered in this thesis so far.

Table 10.2: Example of Multiclass Classification

Time	x_1	x_2	.	.	x_d	**y**
1	2.5	1.0	.	.	3.1	C_2
2	1.8	0.5	.	.	2.5	C_1
.	.	.	.	.	.	.
.	.	.	.	.	.	.
N-1	2.8	1.5	.	.	5.4	C_k
N	2.8	1.5	.	.	5.4	C_3

10.3 Multi label classification

Definition 10.3.1 (Multilabel Classification)**.** Given a labeled training dataset $\mathcal{D} = \{(x_i, y_i)\}$, where each $x_i \in \mathbb{R}^d$ represents a feature vector in d-dimensional real space and each $y_i \in \mathbb{R}^m$ represents the label vector in m-dimensional real space, i.e., $y_i = (L_1, L_2, \ldots, L_m)$, with each $L_i \in \{C_1, C_2\}$, the task is to learn a function $f : \mathbb{R}^d \times \mathbb{R}^p \to \mathbb{R}^m$ where $\mathbb{R}^p$ is a parameter space.

In binary and multi-class classification problems, we utilize the input features x_1 to x_d to predict a single class (out of two or more classes). However, in multi-label classification problems, multiple classes can be associated with the input features x_1 to x_d at a single time step. Table 10.3 provides an example of typical dataset for multi-label classification, where the task could be to predict the satellites visible at a given time. For example, in the first time step, only a subset of satellites may be visible, such as y_2, y_{30}, y_{11}.

Table 10.3: Example of Multiclass Classification

Time	x_1	x_2	.	.	x_d	**y**
1	2.5	1.0	.	.	3.1	L_2, L_{30}, L_{11}
2	1.8	0.5	.	.	2.5	L_2, L_{30}
.	.	.	.	.	.	.
.	.	.	.	.	.	.
N-1	2.8	1.5	.	.	5.4	L_1, L_{15}
N	2.8	1.5	.	.	5.4	$L_{10}, L_{19}, L_{14}, L_{25}$

10.4 Multi-Label Multi-Output Classification

Definition 10.4.1 (Multi-Label Multi-Output Classification)**.** Given a labeled training dataset $\mathcal{D} = \{(x_i, y_i)\}$, where each $x_i \in \mathbb{R}^d$ represents a feature vector in d-dimensional real space and each $y_i \in \mathbb{R}^m$ represents the label vector in m-dimensional real space, i.e., $y_i = (L_1, L_2, \ldots, L_m)$, with each L_i takes one values from the set of multiple classes, i.e, $L_i \in \{C_1, C_2, \ldots, C_k\}$, the task is to learn a function $f : \mathbb{R}^d \times \mathbb{R}^p \to \mathbb{R}^m$ where $\mathbb{R}^p$ is a parameter space.

Table 10.4 illustrates an example of typical dataset for Multi-Label-Multi-Output Classification problem. It is similar to 10.3 with only difference being that there are more than one output and each output can belong to one of the multiple classes. This is the type of classification problem we will encounter in the next chapter.

Table 10.4: Example of Multi-Label Multi-Output Classification

Time	x_1	x_2	.	.	x_d	y_1	y_2	.	.	y_m
1	2.5	1.0	.	.	3.1	C_k	C_1	.	.	C_1
2	1.8	0.5	.	.	2.5	C_3	C_2	.	.	C_k
.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.
N-1	2.8	1.5	.	.	5.4	C_4	C_1	.	.	C_k
N	2.8	1.5	.	.	5.4	C_1	C_3	.	.	C_2

10.5 Conclusion

Table 10.5 provides an overview of the different types of classification problem commonly encountered in machine learning. It categorizes these problems according to the number of classes and the number of outputs. In binary classification, the goal is to classify data into one of two classes. Examples include identifying whether the satellite signal is LOS or not. On the other hand, multiple class classification

involves predicting a single class from multiple options. This can include tasks like what we dealt with in the previous chapter where the satellite signals are classified into various categories such as LOS, multipath, or NLOS. Multi-label classification deals with scenarios where each input can be associated with multiple classes simultaneously. For instance the received signals in the receiver can be tagged as belonging to multiple satellites. Lastly, multi-label-multi-output classification extends this concept further by allowing each input to have multiple associated outputs, each belonging to a different class. For instance based on the signals received there can be multiple satellites visible and each satellite signal can be classified into multiple classes. This type of problem will be dealt in the next chapter. This table serves as a useful reference for understanding the distinctions between these different classification problem types.

Table 10.5: Overview of classification problems

		Number of class	
		Binary	**Multiple**
Number of Output	**One**	Binary Classification	Multi class classification
	Multiple	Multi label classification	Multi-label-multi-output classification

11

Multi-Label-Multi-output classification using DNN with GRU units

In Chapter 9, we saw how different data sorting methods influenced the learned representation by the autoencoder. Sorting the data by time exposed the filter to features from satellites visible at specific time instances (occasionally including features from satellites visible in the next time instant if fewer satellites were visible than the filter size). On the other hand, sorting the data by satellite (and then by time to resolve ties) exposed the filter to features from a single satellite, potentially capturing time-dependent information from that satellite. In this chapter, we aim to leverage the advantages of both approaches. To accomplish this, we need to restructure our data to consolidate features from all the satellites at a given time instance. We will explore this data sorting method and use a deep neural network (DNN) with a Gated Recurrent Unit (GRU). GRU, a type of Recurrent Neural Network (RNN), is particularly adept at handling sequence data such as time series or text. We believe it is more suitable for this type of restructured data.

The chapter starts with an important preprocessing step, where the data is restructured to consolidate features from 31 different GPS satellites at given time instant. This Restructuring of the data transforms the problem from Multi-class classification to Multi-Label-Multi-Output classification problem. In this context, "Multi-Label" signifies that 31 GPS satellites, and each satellite can belong to one of the four classes, constituting the "Multi-Output" aspect of the classification task, thus leading to the nomenclature Multi-Label-Multi-Output.

Furthermore, we use Error Correcting Output Code (ECOC) approach, where the output classes are encoded using a 7-bit length binary code. In typical ECOC approach, the one would train 7 binary classifiers corresponding to the length of the binary code. However, in our approach, we will expand the output neurons, so the first seven neurons correspond to the 7-bit code for the first satellite, and this pattern continues for the rest of the 31 GPS satellite. This expansion will increase the neurons in the output layer from 31 to 31x7.

Since the data is inherently imbalanced we employ heuristic approaches to try mitigate the impact of imbalance, leading to the creation of a training and test split, which is explained in the next section

11.1 Data Processing

As in previous chapter, we begin with downsampled data, utilizing the 14 features outlined in Table 8.1 from the GPS constellation exclusively. It's important to mention that, unlike the previous chapter, we refrain from using Galileo data in this chapter. This decision is made to streamline the prediction process in the classification layer by reducing the number of satellites we need to account for.

11.1.1 Restructuring and cleaning the data

The first step involves restructuring the data to consolidate features from 31 different GPS satellites at any given instant. This results in a feature matrix with dimensions 23133×432, where each row represents a specific time instance, and each column corresponds to features (14 features from 31 satellites each). Simultaneously, a class label matrix of size 23133×31 is created. This restructuring introduces NaN values for time instance where no satellite signal is available (or the satellite is below the horizon). Therefore, cleaning the data becomes imperative.

During the cleaning stage, we substitute NaN values with zeros. This approach is logical for the SSI feature, as 'zero' values imply poor signal strength. While replacing NaN with zeros for other features may not always be ideal, but we opt for this simple solution as our initial step. Regarding class labels, NaN values are substituted with Class 4, which signifies satellites beyond the horizon or unavailable signals. This introduction of Class 4 is an unavoidable consequence of restructuring the data. It's worth reiterating that in this chapter we introduce Class 4 which is a direct outcome of the data restructuring process.

11.1.2 Error-Correcting Output Codes (ECOC) encoding

Error-Correcting Output Codes (ECOC) is a machine learning technique utilized for addressing multi-class classification problems. It accomplishes this by converting the original multi-class (output) problem into several binary classification tasks. In this method, each class is assigned a unique binary code of a specific length. We adopt the approach proposed by XXX to devise the binary code, resulting in an encoding matrix depicted in Table 11.1.

Table 11.1: Code book used to encode each classes

	f_1	f_2	f_3	f_4	f_5	f_6	f_6
Class 1	1	1	1	1	1	1	1
Class 2	0	0	0	0	1	1	1
Class 3	0	0	1	1	0	0	1
Class 4	0	1	0	1	0	1	0

Following the standard procedure, one would typically train a binary classifier for each bit in the code (7 binary classifier in this case). However, in this chapter, we take a different approach by increasing the number of neurons in the output layer from 31 to 217 (31x7). In this arrangement, the first 7 neurons in the output layer correspond to the 7-bit length code of the first satellite, and this pattern continues for the remaining satellites. The fundamental idea is to train the neural network to produce the code, and then decode it using a minimum distance approach.

11.2 Imbalance in dataset

In Multi-label-Multi-Output classification, dataset imbalance can be viewed from two perspectives: imbalance in the frequency of individual classes, and imbalance in the frequency of label sets. Figure 11.1 illustrates the imbalance in the frequency of individual classes, depicting the total count of classes present in the restructured dataset. Note: we will come back to the red curve later in this chapter. It's important to note that Class 3 comprises NLOS and NLOS+Multipath, while Class 4 represents satellites beyond the horizon. Due to data restructuring, Class 4 becomes the majority class compared to Class 1 in the previous chapter.

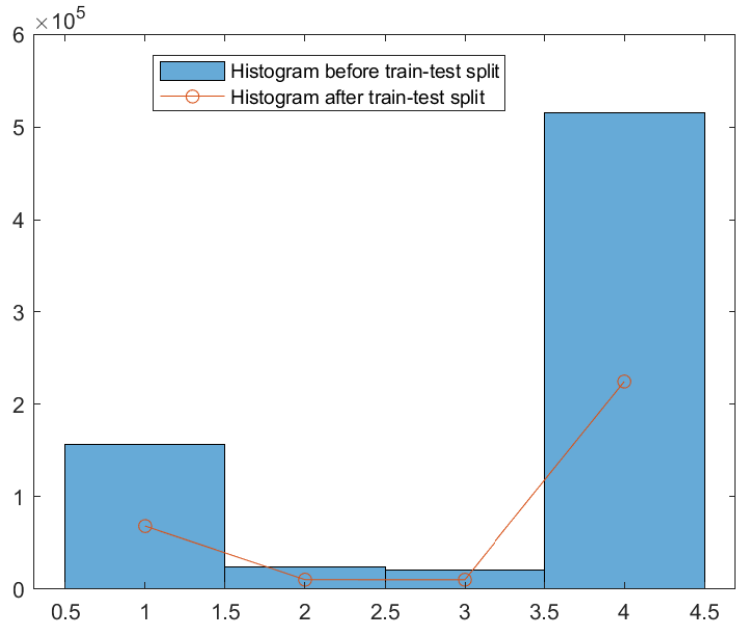


Figure 11.1: Total number of data points in each class. Note: Class 3 consists of NLOS and NLOS+Multipath and class 4 represents satellite beyond horizon.

Figure 11.2 highlights the imbalance in the frequency of label sets within the dataset, featuring a total of 1034 unique label sets. In the first plot, the count of each unique label set is presented in descending order, revealing a distribution where certain

label sets appear only once, while others occur more than 100 times. Note: we will come back to the red curve later in this chapter. The second plot demonstrates the proportion of classes within each unique label set. This imbalance in the frequency of classes and label sets, complicates the task of achieving a balanced dataset for training and testing. In the next section we try to address this issue.

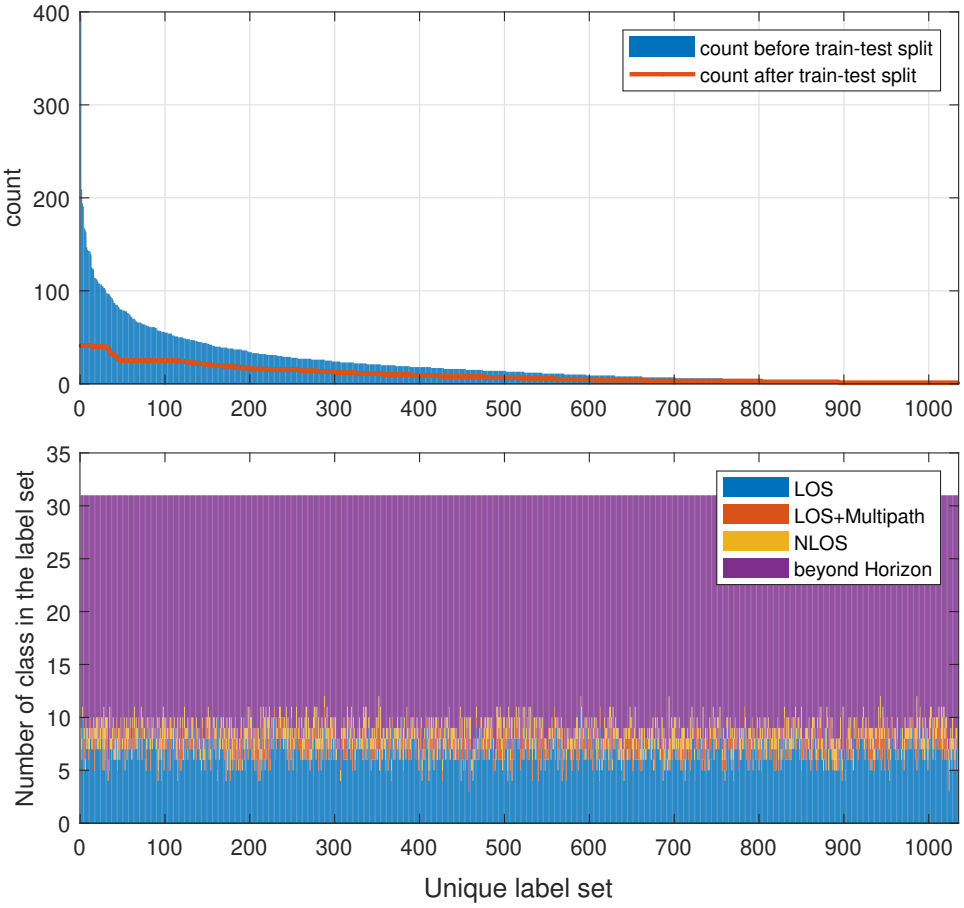


Figure 11.2: Total number of data points in each class. Note: Class 3 consists of NLOS and NLOS+Multipath and class 4 represents satellite beyond horizon.

11.2.1 Heuristic approach to handle imbalance dataset

Figure 11.3 shows a 3D-bubble chart, where each axis represents the total number of classes in a unique label set (excluding Class 4, which is not shown), with the size of the bubble representing the total number of examples for that unique label set in the dataset. For instance, for the data point shown on the rightmost side in figure 11.3 signifies a unique label set with 60 such examples (size=60) in the dataset, featuring 7 LOS satellites (X=7), 3 NLOS satellites (Z=3), and the remaining 21 satellites beyond the horizon (summing up to 31, i.e., 7+3+21). Similarly, a data point shown

on the leftmost side in figure 11.3 represents a unique label set with 20 such examples in the dataset, comprising 5 LOS satellites, 3 LOS+Multipath satellites, 4 NLOS satellites, and 19 satellites beyond the horizon. Note, the concentric bubbles at each data point represents data point with different number of class4 but same number of other classes. It appears there due to the fact that we cannot depict the 4th axis separately.

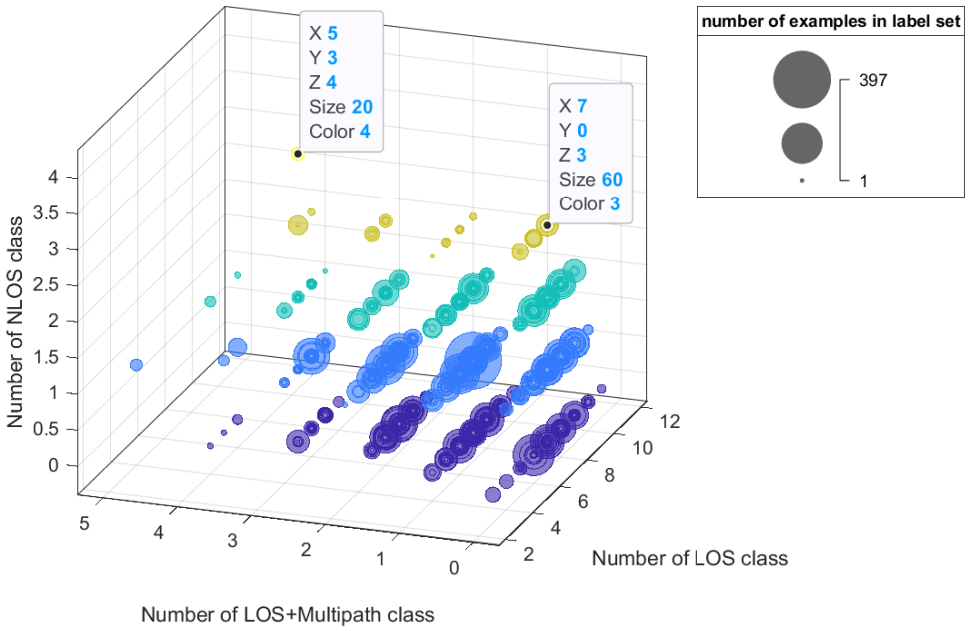


Figure 11.3: 3D-bubble chart showcasing proportion of classes in different label sets.

To address the imbalance in the dataset, we employ a distinct Train-Test split ratio, considering two aspects of imbalance:

Imbalance in Frequency of Label Sets:

- To tackle this, we set a dynamic train-test ratio ensuring that the maximum number of examples in each label set is not more than 50 in the training set. This strategy focuses on data points close to 0 on the x-axis in the first subplot of Figure 11.2.
- For label sets with less than 6 examples, we set the train-test ratio to 50:50. This targets data points close to the rightmost part on the x-axis in the first subplot of Figure 11.2.

Imbalance in Frequency of Classes:

- To address the imbalance in the frequency of classes, we use train-test ratio of 30:80 for label sets containing all LOS classes and zero instances of class 2 and class 3. This would be the bottom-right axis in Figure 11.3. (Note: Although class 4 is the majority class we cannot completely get rid of it since it was introduced due to the restructuring of the data.

So we instead focus on the next majority class.)
- However, this strategy alone proved insufficient, leading us to adjust our approach. We then targeted the label set closest to the bottom-right axis, setting a train-test ratio of 50:50.

Rest of the label sets:
- Test-train split ratio is set to 80:20 for the rest of the label sets.

It is important to mention that achieving a balanced stratified Train-Test split is difficult for Multi-Label-Multi-Output classification. However for this thesis we use the above mentioned naive-heuristic approach. The red curve in figure 11.1 and 11.2 represents the data after the test-train split. Drawbacks are that we do not get a perfect stratified train-test split, however this is considered OK for this thesis.

11.2.2 Standardisation of the data

After splitting the data into training and testing data set, the training data is standardised using z-score standardisation. The mean and standard deviation thus obtained is then used to standardise the testing data set.

11.3 GRU neural network structure

We use DNN with GRU, comprising of two hidden layers and one output layer. In the first hidden layer, the number of units is set to double the number of features. And in the second hidden layer, the number of units aligns with the feature size. The number of units in the output layer corresponds to the 7-bit code length for each of the 31 GPS satellites. The network structure is provided in Table 11.2.

Layer	Type	Description	Details
1	Input Layer	Sequence Input	Sequence input with 434 dimensions
2	Gru1	GRU	GRU with 868 hidden units
3	Gru2	GRU	GRU with 434 hidden units
4	FC1	Fully Connected	Fully connected layer with 217 units
5	sigLayer	Sigmoid	Sigmoid
6	crossEntropyLayer	Regression Output	Regression Output

Table 11.2: Description of Layers used in DNN.

11.4 Result

The confusion chart in Figure 11.4 shows how well the network learned to perform on the training and testing dataset. It's particularly good at predicting Class 4, which came about during data restructuring. This is helpful because we can easily deal with introduced data (or noise) instances. Although there are indications of overfitting, the network demonstrated robust performance, achieving over 85% in

both class-wise recall and precision metrics, and easily over 96% on an average for LOS classes, highlighting the efficacy of our proposed methodology.

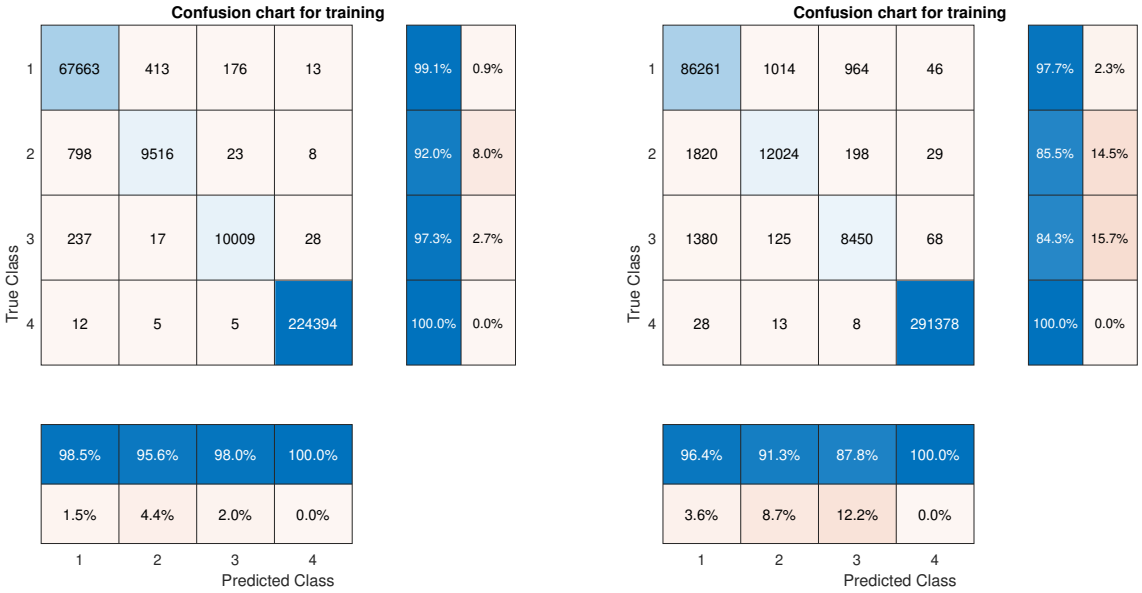


Figure 11.4: Confusion chart showing the results from DNN with GRU units.

12
Conclusion

In this thesis, our primary focus has been to use machine learning algorithms to tackle one of the main challenges in localization using GNSS called the Multipath errors. Our approach involved collecting data using a receiver mounted on an excavator and then using ray tracing simulations to annotate the data for supervised machine learning. Through the deployment of various machine learning algorithms, including Support Vector Machine, Autoencoders, 1D Convolution Neural Network, and Gated Recurrent Unit, our aim was to classify the signals into LOS, mulitpath and NLOS.

The chapter on "Ray Tracing" outlines the methodology used to label observations recorded by the GNSS receiver using Matlab's wireless communication toolbox. Ray tracing in Matlab employs the shooting-bouncing-rays (SBR) method to simulate signal behavior. Despite limitations such as a distance between the transmitter and receiver should not be more than 500 km great circle radius, and the absence of phenomena like diffraction, ray tracing proves effective in categorizing signals into Line of Sight (LOS), Non-Line of Sight (NLOS), and Multipath classes. The virtual environment for ray tracing was set up using the data from Open Street Map (OSM) and modified for detail (such as adding building, excavator and tree). Computational challenges involved in labeling each time epoch for both GPS and Galileo constellations are mitigated through discretization of the azimuth-elevation space around the receiver, and then performing raytrcing enabling efficient simulation and labeling.

The thesis then delves into the next chapter where we conduct feature analysis and engineering with respect to the labels obtained from ray tracing simulations to gain insights into any correlation between the labels and features. Feature analysis involves assessing the significance of various input features in relation to the target variable. Meanwhile, feature engineering utilizes insights from feature analysis to enhance the model's performance by transforming existing features or creating new ones. In this context of feature engineering, Indicator variables and signal strength indicators are introduced to augment the existing features. Inspired by the literature, we create a feature called range acceleration, which is the time derivative of the range measurement. Additionally, the difference of range accelerations between code and carrier signals showed a nice distribution with mean approximately zero and a standard deviation showing correlations when analyzed with respect to the labels. In particular, LOS labels showed a lower standard deviation when compared to NLOS counterparts.

To begin applying machine learning algorithms, we started with something classical and simple - the Support Vector Machine (SVM). The data set in hand was inherently imbalanced, meaning that some classes were over-represented than others, with LOS being the majority class and NLOS being the minority class, due to measurements being done in an open field. Techniques to handle imbalanced datasets, such as under-sampling the majority class, were employed. Since SVM is typically a binary classification algorithm and our data is multi class classification, we utilized the One-Versus-One (OVO) approach within MATLAB's 'fitcecoc' function, enabling us to extend SVM's binary classification capability to multi class scenarios effectively. Through k-Fold cross-validation, we demonstrated the model's consistent performance and generalization capability across different folds and iterations. The results showed approximately 75 % class-wise recall and precision metric. It's crucial to acknowledge potential overfitting, particularly evident in the lower performance on the test dataset compared to the training data. This observation suggests the need for further exploration to enhance model robustness and prevent overfitting. Overall, SVM served as a base to start with in order to classify LOS, Multipath, and NLOS signals, offering a foundation to explore and benchmark for other machine learning algorithms.

We then used CNN-based autoencoders in the hope of learning new features. The encoder was then coupled with SVM for multi class classification. Through the process of training autoencoders, the model learns to extract compact and efficient representations of input data, which are subsequently utilized for classification purposes. In this chapter, we explored three distinct simulations, each employing the same model architecture but varying in the methodology of sorting the input data before feeding it into the autoencoder. These sorting methods - based on time, based on satellites and then time, and based on classes - influenced the features learned by the autoencoder and consequently impacted the classification performance. In the simulation where we sorted the data with time, the filter is exposed to features from all the available satellites at that given instance. If fewer satellites were visible, the model incorporated features from the next time instants visible satellites. Whereas in the simulation where we sorted the data with satellite first, the filter was exposed to features from one particular satellite. This influenced the features learned by the autoencoder and consequently impacted the classification performance. In the simulation where data was sorted by classes, showed exteremley good performance. Sorting based on classes allowed the autoencoder to learn features from data points with similar characteristics, leveraging the CNN's capability to capture relationships between neighboring data points effectively. As a result, this simulation outperformed others in terms of classification accuracy. However, in practice, we cannot use this since the classes are not known first hand.

To leverage the benefits of sorting data by time and satellite, we transitioned from a 'multi class' to a 'multi-label-multi-output' classification problem. This meant inputting features from all 31 GPS satellites to predict classes for each satellite at given time instant, enabling us to harness information from all available satellites

simultaneously. The chapter begins with a crucial preprocessing step, consolidating features from the entire GPS satellite set. While this transformation introduces new challenges such as imbalance in label frequency and imbalance in frequency of label-sets, we addressed them using heuristic approaches. Although achieving a perfect stratified training-test split proves challenging, our employed strategies sufficiently balances the dataset for the task at hand. Moreover, we adopted an Error Correcting Output Code (ECOC) approach, expanding the output layer to accommodate binary codes corresponding to a class. The training process utilized a deep neural network with GRU units, which yielded promising results. The network successfully identified all synthetic classes that were introduced during the process of restructuring the data. Although there were indications of overfitting, the network demonstrated robust performance, achieving over 85 % in both class-wise recall and precision metrics, and easily over 96% on an average for LOS classes, highlighting the efficacy of our proposed methodology.

Bibliography

[1] Willy Bertiger et al. "GipsyX/RTGx, a new tool set for space geodetic operations and research". In: *Advances in Space Research* 66.3 (2020), pp. 469–489. ISSN: 0273-1177. DOI: https://doi.org/10.1016/j.asr.2020.04.015. URL: https://www.sciencedirect.com/science/article/pii/S0273117720302532.

[2] Daniel Egea-Roca et al. "GNSS User Technology: State-of-the-Art and Future Trends". In: *IEEE Access* 10 (2022), pp. 39939–39968. DOI: 10.1109/ACCESS.2022.3165594.

[3] Patrick Geragersian et al. "Multipath Detection from GNSS Observables Using Gated Recurrent Unit". In: *2022 IEEE/AIAA 41st Digital Avionics Systems Conference (DASC)*. 2022, pp. 1–7. DOI: 10.1109/DASC55683.2022.9925850.

[4] Li-Ta Hsu. "GNSS multipath detection using a machine learning approach". In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. 2017, pp. 1–6. DOI: 10.1109/ITSC.2017.8317700.

[5] Qing Liu, Zhigang Huang, and Jinling Wang. "Indoor non-line-of-sight and multipath detection using deep learning approach". In: *GPS Solutions* 23 (2019), pp. 1–14. URL: https://api.semanticscholar.org/CorpusID:182463376.

[6] Yiming Quan et al. "Convolutional Neural Network Based Multipath Detection Method for Static and Kinematic GPS High Precision Positioning". In: *Remote Sensing* 10.12 (2018). ISSN: 2072-4292. DOI: 10.3390/rs10122052. URL: https://www.mdpi.com/2072-4292/10/12/2052.

[7] Akpojoto Siemuri et al. "Machine Learning Utilization in GNSSUse Cases, Challenges and Future Applications". In: *2021 International Conference on Localization and GNSS (ICL-GNSS)*. 2021, pp. 1–6. DOI: 10.1109/ICL-GNSS51451.2021.9452295.

[8] Taro Suzuki and Yoshiharu Amano. "NLOS Multipath Classification of GNSS Signal Correlation Output Using Machine Learning". In: *Sensors* 21 (Apr. 2021), p. 2503. DOI: 10.3390/s21072503.

[9] Taro Suzuki, Yusuke Nakano, and Yoshiharu Amano. "NLOS Multipath Detection by Using Machine Learning in Urban Environments". In: Nov. 2017, pp. 3958–3967. DOI: 10.33012/2017.15291.

[10] Roi Yozevitch, Boaz Ben-Moshe, and Ayal Weissman. "A Robust GNSS LOS/NLOS Signal Classifier: GNSS Shadow Matching Algorithms". In: *Navigation* 63 (Dec. 2016), pp. 429–442. DOI: 10.1002/navi.166.

A

Appendix A

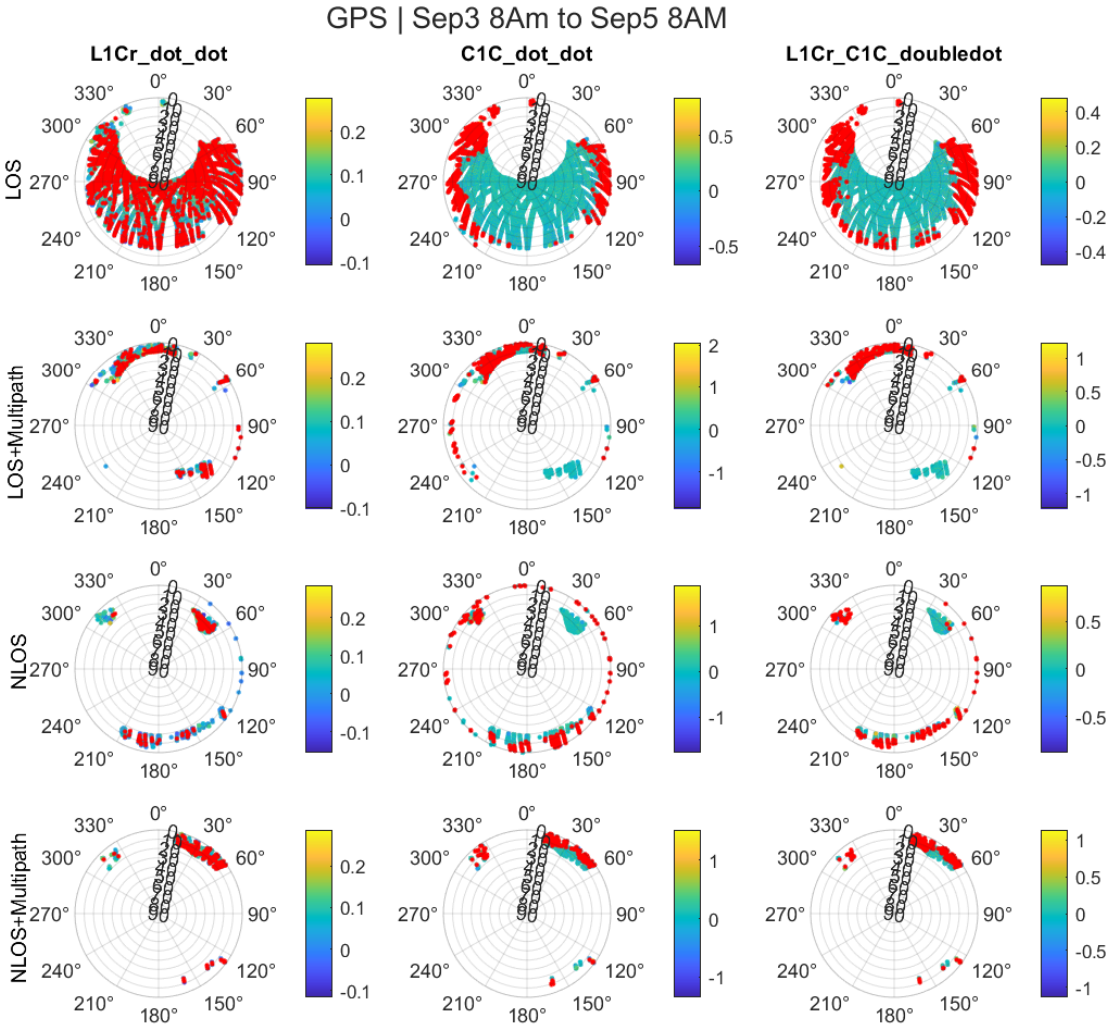


Figure A.1: GPS: An illustration of the effect of differencing of Range-Acceleration in polar plot. Each row corresponds to different classes. The first and second column corresponds to Range-acceleration on L1C and C1C channel respectively for different classes. The last column corresponds the differencing of the first two columns. The colorbar in each plot corresponds to 3-Sigma region. The red marker in each plot corresponds to data points beyond 3-Sigma region

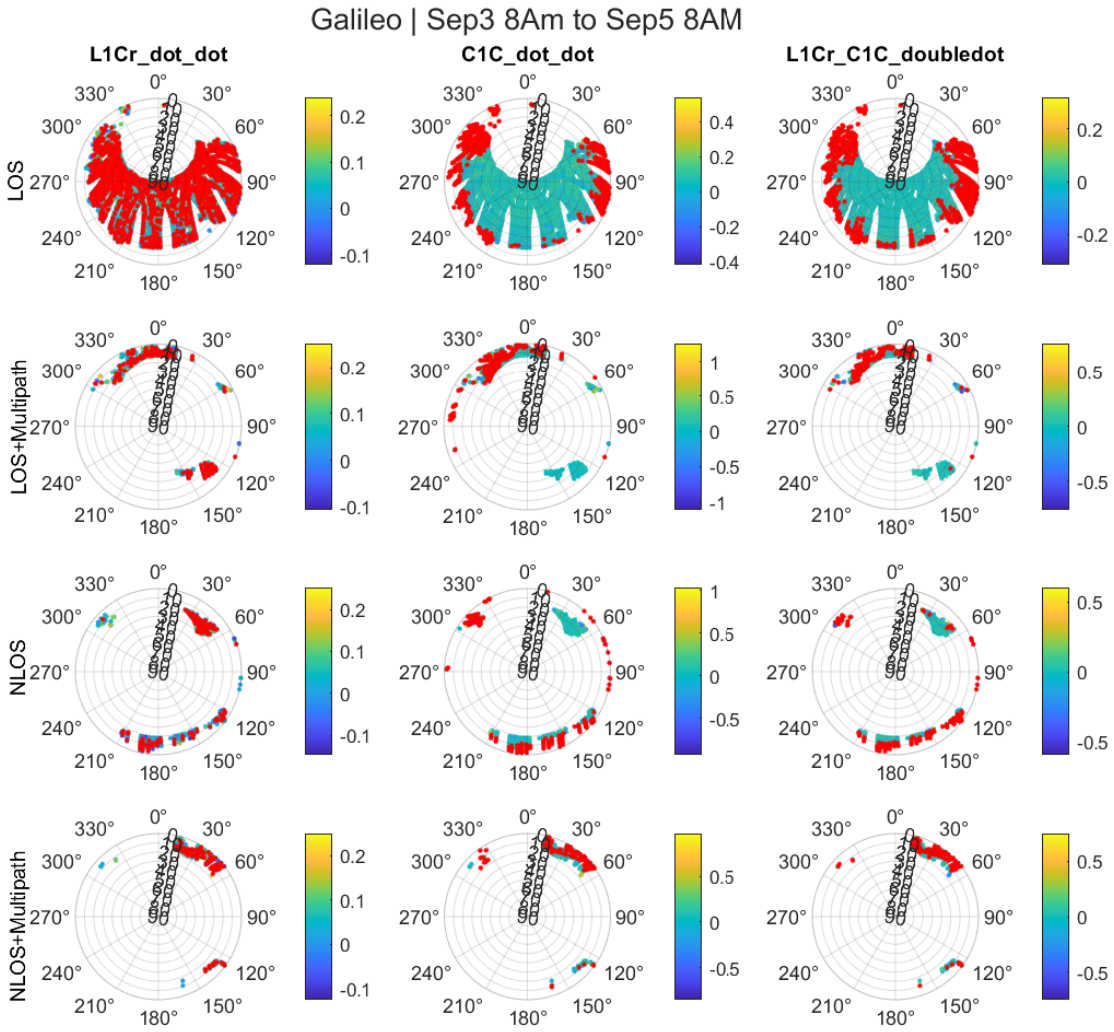


Figure A.2: Galileo: An illustration of the effect of differencing of Range-Acceleration in polar plot. Each row corresponds to different classes. The first and second column corresponds to Range-acceleration on L1C and C1C channel respectively for different classes. The last column corresponds the differencing of the first two columns. The colorbar in each plot corresponds to 3-Sigma region. The red marker in each plot corresponds to data points beyond 3-Sigma region

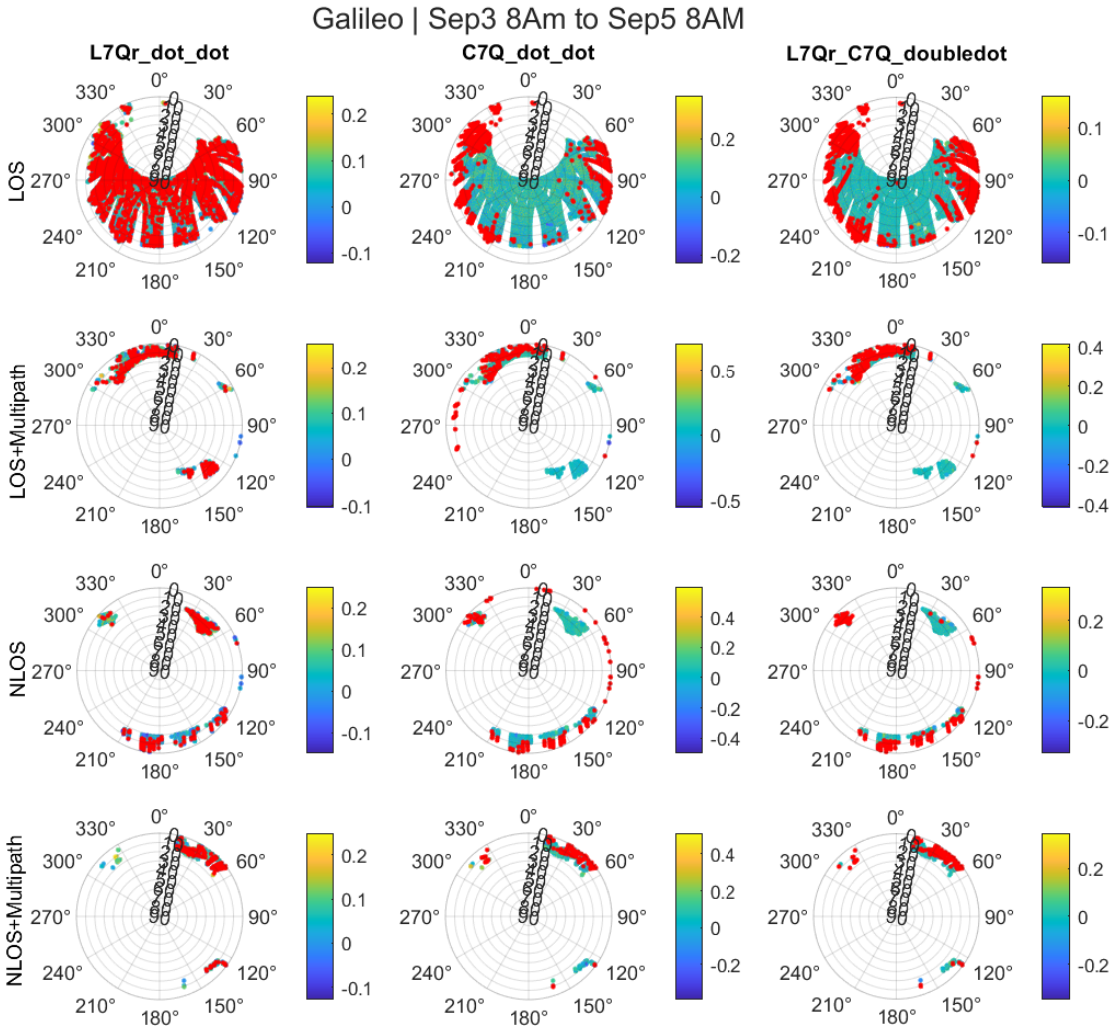


Figure A.3: Galileo: An illustration of the effect of differencing of Range-Acceleration in polar plot. Each row corresponds to different classes. The first and second column corresponds to Range-acceleration on L7Q and C7Q channel respectively for different classes. The last column corresponds the differencing of the first two columns. The colorbar in each plot corresponds to 3-Sigma region. The red marker in each plot corresponds to data points beyond 3-Sigma region

B

Appendix B

B.1 Dense layer based Autoencoders with SVM for Multi-class classification

In this chapter, we will model an Autoencoder using a fully connected layer (or dense layer), complemented by Support Vector Machine (SVM) for classification purposes. We will use this model to perform three different simulations. The difference among these simulations lies in the number of neurons in the bottleneck layer.

B.2 Dataset and features

The preprocessing steps, which include cleaning, z-score normalization, concatenation of data from both GPS and Galileo constellations, and handling imbalanced datasets, remain consistent with the procedures outlined in chapter 8. Also data is shuffled before it is fed to the Autoencoder. The number of features used is also consistent with the chapter 8.

B.3 Autoencoder structure

The table B.1 provides a description of the layers in an Autoencoder model, highlighting the type of each layer, a brief description, and specific details related to the layer's configuration. Three different simulations is performed with different neurons in the bottleneck layer (Layer 3). The number of neurons for each simulation is set at 14, 7, and 3, respectively. This will make the Autoencoder learn different compact representation of the original features, which will then be used for classification using SVM. The solver used is adam with learning rate of 0.01.

B.4 Svm for classification

After training the autoencoder, the decoder part is discarded (layer4 and 5), and the learnt representation from the encoder part (layer3) is integrated with the SVM for the final training phase. During this stage, the tuning of parameters is exclusively done on those related to the SVM. The SVM employed for multiclass classification remains consistent with the one detailed in the chapter 8. It utilizes the One-versus-

Layer	Type	Description	Details
1	SeqInput	Sequence Input	Sequence input with 14 dimensions
2	FC1	Fully Connected	fully connected layer with 14 neurons
3	FC2	Fully Connected	fully connected layer with 3 neurons
4	FC3	Fully Connected	fully connected layer with 14 neurons
5	RegOutput	Regression Output	Mean-squared-error

Table B.1: Description of Layers in the Autoencoder.

One method, the coding matrix, and the Radial Basis Function (RBF) kernel, as described in the chapter 8.

B.5 Results

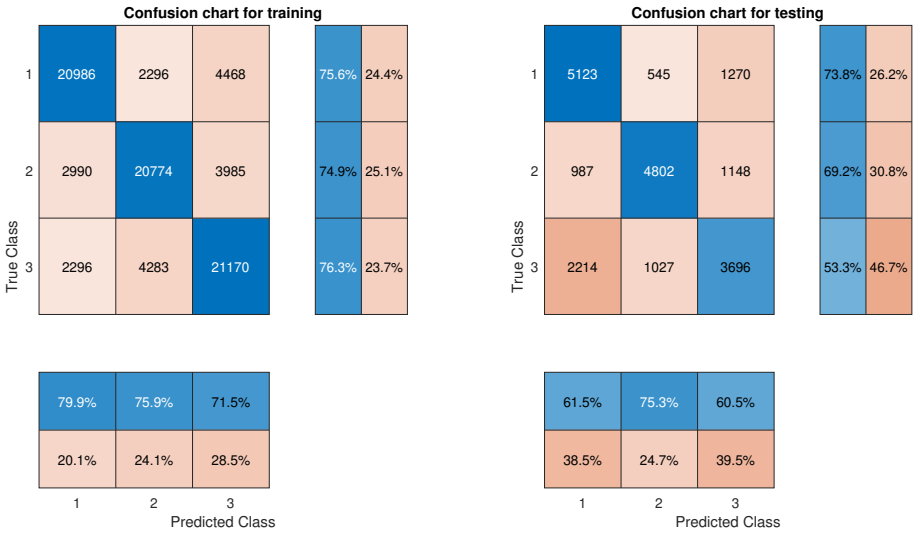
The training metrics for the Autoencoder in three different simulations are presented in Table B.2. In all simulations, the validation RMSE reaches a low value, approximately 0.2, except for the Autoencoder with 3 neurons in the bottleneck layer. This discrepancy is likely attributed to over-compression of features, preventing the model to learn a compressed representation which hinders it from effectively reconstructing the original features.

Epoch	Iteration	Mini-batch RMSE			Validation RMSE		
		Sim1(14)	Sim2(7)	Sim3(3)	Sim1(14)	Sim2(7)	Sim3(3)
1	1	6.52	5.55	5.06	5.47	4.73	4.49
25	25	1.48	2.30	3.02	1.43	2.19	2.89
50	50	0.66	1.24	2.44	0.66	1.22	2.38
75	75	0.44	0.97	2.27	0.46	0.96	2.24
100	100	0.31	0.78	2.23	0.31	0.79	2.21
125	125	0.23	0.65	2.21	0.22	0.67	2.19
150	155	0.21	0.56	2.20	0.19	0.59	2.19
175	175	0.20	0.47	2.19	0.18	0.50	2.19
200	200	0.20	0.39	2.19	0.18	0.40	2.18
225	225	0.19	0.32	2.19	0.18	0.32	2.18
250	250	0.19	0.26	2.19	0.17	0.26	2.18
275	275	0.18	0.23	2.19	0.17	0.21	2.18
300	300	0.18	0.21	2.19	0.17	0.19	2.18

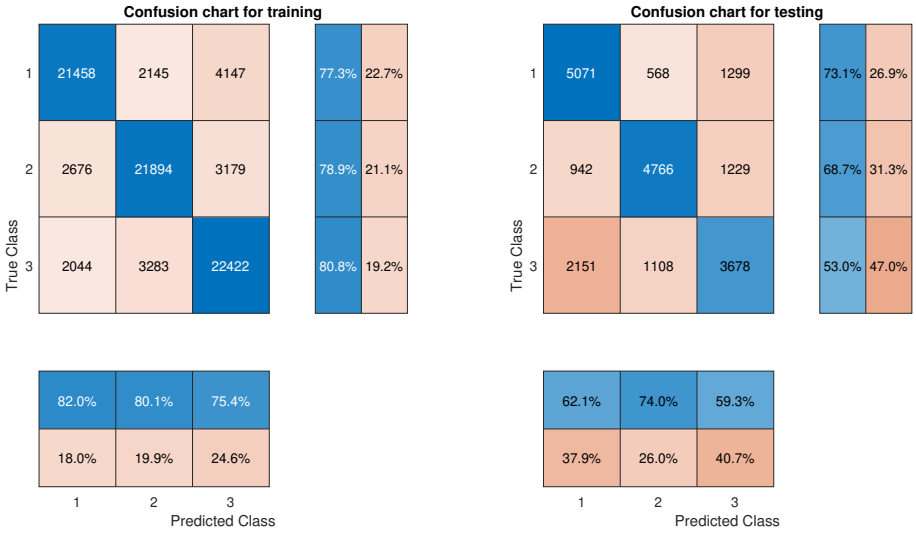
Table B.2: Training Progress Metrics for all three Autoencoders, with sim1, sim2 and sim3 corresponds to training with 14, 7 and 3 neurons in the bottleneck layer.

The confusion charts for the three simulations are illustrated in Figure B.1, with subfigures (a), (b), and (c) representing the confusion charts for simulations with 14, 7, and 3 neurons in the bottleneck layer of the Autoencoder, respectively. Across

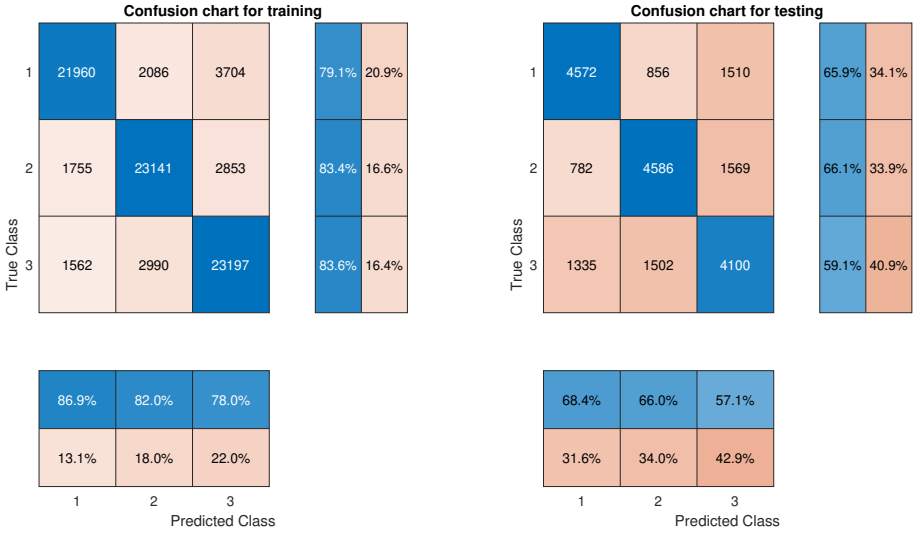
all three figures, there is evident overfitting, as the testing performance consistently lags behind the training performance.



(a) 14 neurons in the bottleneck layer



(b) 7 neurons in the bottleneck layer



(c) 3 neurons in the bottleneck layer

Figure B.1: Confusion chart for predicting the classes from the learned representation of the Autoencoder with varying neurons in the bottleneck layer.

C

Appendix C

Apart from the above simulation an additional simulation was also performed, the difference being that in contrast chapter 8 where 14 features were utilized, the Autoencoder in this context employed only 3 features, as detailed in Table C.1. Specifically, SSI and 'difference in range-Acceleration' is chosen due to its favorable statistics and distribution. The motivation for selecting these features is due to the insights gained in previous chapter in figure 7.7b and figure 7.9.

Features	Description
$C1C_{SSI}$	Signal strength indicator of C1C signal
$L1C_{SSI}$	Signal strength indicator of L1C signal
$L\ddot{1}C - C\ddot{1}C$	Difference of range acceleration of L1C and C1C

Table C.1: Features used in this chapter for training the Autencoder with 1D CNN.

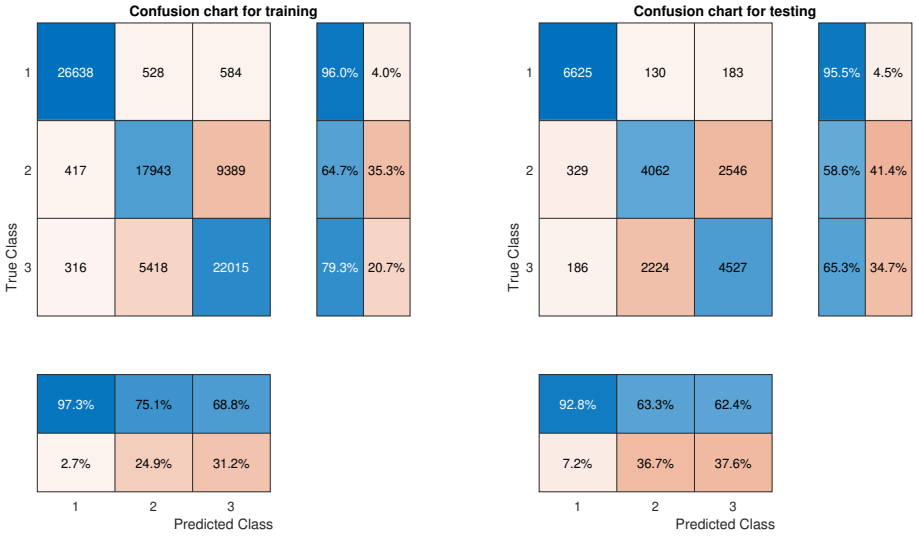
The training progress of the Autoencoder is depicted in Table C.2. Again, Adam solver is used and the training is limited to 150 epochs. The confusion chart obtained after training the SVM is illustrated in Figure C.1c. The overall performance on the test dataset is lower compared to the training dataset for all classes. This discrepancy could indicate potential overfitting.

Epoch	**Iteration**	**Mini-batch RMSE**			**Validation RMSE**		
		Sim1	**Sim2**	**Sim3**	**Sim1**	**Sim2**	**Sim3**
1	1	2.38	2.19	2.65	4.22	8.73	5.91
25	25	0.47	1.05	0.74	0.68	1.14	0.70
50	50	0.23	0.26	0.20	0.20	0.32	0.15
75	75	0.05	0.05	0.05	0.06	0.06	0.04
100	100	0.02	**0.02**	0.01	0.02	**0.02**	0.01
125	125	0.006	0.05	0.009	0.02	0.06	0.007
150	155	**0.005**	0.29	**0.007**	**0.02**	0.07	**0.006**

Table C.2: Training Progress Metrics

(a) Sorted based on time

(b) Sorted by satellite and then time



(c) sorted by class

Figure C.1: Confusion chart for simulation using 3 features and an Autoencoder with 1D CNN.
X

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY