Portable Doppler Tracking Ground Station

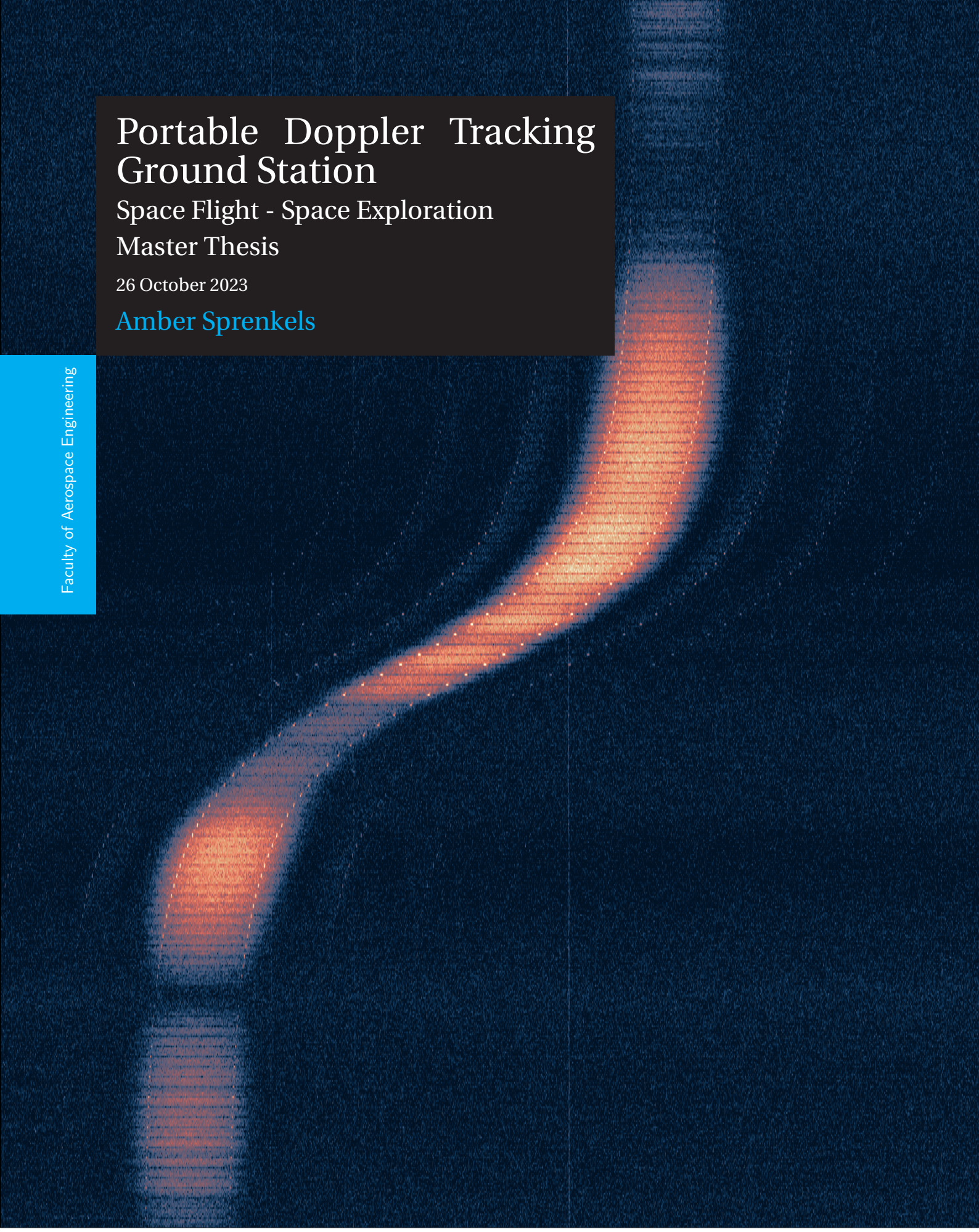
Space Flight - Space Exploration

Master Thesis

26 October 2023

Amber Sprenkels

Faculty of Aerospace Engineering



TU Delft Delft University of Technology

Challenge the future

PORTABLE DOPPLER TRACKING GROUND STATION

SPACE FLIGHT - SPACE EXPLORATION

MASTER THESIS

26 OCTOBER 2023

by

A. E. S. Sprenkels

In partial fulfilment of the requirements for the degree of

Master of Science
in Aerospace Engineering
at Delft University of Technology

To be defended publicly on 26 October 2023

Thesis duration: 04 January 2022 - 26 October 2023

Author:	A. E. S. Sprenkels,	TU Delft (4351584)
Supervisor:	dr. ir. B.C. Root,	TU Delft
Thesis Committee:	dr. ir. E. Mooij,	TU Delft (Chair)
	dr. ir. B.C. Root,	TU Delft (Supervisor)
	dr. ir. R. Saathof,	TU Delft (Exernal member)
	dr. ir. J.G. De Teixeira da Encarnação,	TU Delft (Additional)

Cover image is the spectrogram of Nayif recorded on 1 September 2022 at 12:19 CEST

TU Delft

Preface

A long journey has come to an end. A journey that started in September 2014 when I first arrived at the Faculty of Aerospace Engineering to start my Bachelor in Aerospace Engineering. Now, nine years later, the chapter closes with this thesis being the pinnacle of my work. It is not the end goal that matters, but your journey towards it. All knowledge gained on this journey helped me to complete this final task.

It was a journey not without issues and setbacks. Originally, I started my master thesis externally in October 2019 on the topic of Microsatellites in Very Low Earth Orbit. When the pandemic and lockdown hit, progress stagnated. Unfortunately the supervision structure broke down, which combined with my own personal issues at the time caused that thesis to flop. It took some time to get on the right track again, and I started this thesis in January 2022. There are still some ups and downs, which resulted in this thesis taking longer than I had originally hoped for.

I have always liked the practical application of research. This is what peeked my interest in this thesis topic. I want to thank Bart Root for granting me the possibility to work on this thesis. I greatly appreciated our weekly meetings, where the proper guidance and counselling was given. I also greatly appreciate your understanding for what I am going through whilst doing this thesis, and describing it with the great metaphor of running two marathons simultaneously. In addition to Bart, I also want to extend my gratitude to João De Teixeira da Encarnacao for his insights and feedback and to Martin Søndergaard for helping me unravel the various DopTrack programs and functionalities.

Gathering data for DopTrackBox was a unique experience. With real world experiments things can go wrong, and many things did. These mistakes are taken into account and help you improve. I liked them being real world experiments over just running simulations, as these are somehow more tangible. There is an actual satellite out there, which transmits actual signals. The interactions with hardware, real world satellites, and software resulted in something I had not done before this thesis.

Next to my supervisor and other supporting TU Delft staff I want to thank my parents, family, and friends for their support. Without them this experience would have been a whole lot more difficult. They helped me remain focused and motivated. This chapter has been closed, but the book is not yet finished. There is always more to learn and you are never too old to keep learning. I do not know what the future holds, but I do know that the journey is more important than the goal.

Amber Sprenkels
Nieuw Beijerland, October 2023

Contents

Preface iii
Summary vii
List of Abbreviations & Symbols ix
List of Figures xi
List of Tables xv
1 Introduction 1
2 DopTrackBox Concept 3
2.1 DopTrack Background . . . 3
2.2 DopTrackBox Concept Description . . . 6
2.3 Design Goals . . . 7
2.4 DopTrackBox Requirements . . . 8
2.5 DopTrackBox Hardware Components . . . 11
2.6 DopTrackBox Software . . . 15
2.7 Concept Iterations . . . 20
3 Installation Manual 23
3.1 Hardware Set-up . . . 23
3.2 Software Set-up . . . 25
3.3 DopTrackBox Operation Set-up . . . 27
4 Experiment Set-up 29
4.1 Nayif-1 . . . 30
4.2 Different Experiments . . . 31
4.3 Challenges . . . 32
4.4 Methodology . . . 33
5 Verification and Validation 35
5.1 Hardware Verification . . . 35
5.2 Software Verification . . . 38
5.3 Verification Conclusions . . . 42
6 Results 45
6.1 Data Presentation . . . 45
6.2 Experiment 1: The DopTrackBox and DopTrack Comparison . . . 46
6.3 Experiment 2: DopTrackBox Configuration Changes . . . 59
7 Data Analysis 75
7.1 Shortcomings . . . 75
7.2 DopTrackBox and DopTrack . . . 76
7.3 Effect of a GPS Clock . . . 78
7.4 Effect of a Different SDR . . . 79
7.5 Effect of Manually Pointing . . . 80
7.6 Data Analysis Conclusions . . . 81
8 Improvements 83
8.1 DopTrackBox System Improvements . . . 83

8.2 DopTrackBox Finalised Requirements . . . 86
8.3 Cost Overview . . . 88
8.4 Recommendations . . . 89
8.5 DopTrackBox Improvement Conclusions . . . 90
9 Conclusion **93**
9.1 DopTrackBox Set-up . . . 93
9.2 Research Questions . . . 93
9.3 Research Objective Reflection . . . 95
Bibliography **97**
A Additional Plots & Graphs **99**
A.1 Spectrograms and SNR plots for Experiment 1 . . . 99
A.2 Frequency plots for Experiment 1.3 . . . 109
A.3 Spectrograms and SNR plots for Experiment 2 . . . 110
A.4 Frequency plots for Experiment 2.3 . . . 124
B Program Code **127**
B.1 Scheduler Software . . . 127
B.2 Recorder Software . . . 137
B.3 Processing Software . . . 139

Summary

This thesis analyses the possibility of creating a portable Doppler tracking ground station using COTS components. This solution is called DopTrackBox and is based on TU Delft's DopTrack. It is intended to be a smaller and cheaper solution for satellite Doppler tracking. The main factors deciding whether DopTrackBox is feasible are the quality of the data compared to DopTrack and the development of the hardware and software. The data quality is determined by looking at the SNR and range rate difference, which is the difference between the range rate determined by the TLE orbit and the actual measured data.

Satellite Doppler tracking uses the Doppler shift of radio signals received from the satellite to analyse its location and velocity. Both DopTrackBox and DopTrack use one-way Doppler tracking, where the signal sent by the satellite is processed at the ground station. DopTrackBox' main hardware components are a computer, SDR, GPS clock, antenna, and external SSD. The different experiments will use different combinations of SDRs and GPS clock. At the concept stage of DopTrackBox design goals and requirements were created to guide the development of DopTrackBox. The most important design goal is for DopTrackBox to be independent and portable. DopTrackBox uses the same software as DopTrack, which had to be adopted for the new system. The software can be divided in the scheduler, recorder, and processing. The scheduler looks at when the satellites pass in line of sight of the ground station. It then creates the commands to activate the recorder when a satellite passes. The recorder records the satellite data when it passes. The processing software takes the recorded data and extracts useful information.

To test the data quality of DopTrackBox, a number of different experiments are devised. All these experiments used the data from the Nayif-1 satellite. This satellite was chosen for its reliability and favourable pass times, number of passes, and signal strength. It is no longer possible to obtain more data from this satellite as it re-entered the atmosphere on 18 July 2023. The experiments are divided into two main experiments, with three subexperiments each. Experiment 1 was done at the DopTrack ground station and uses the same omnidirectional antennas as DopTrack. The subexperiments compare the Green and Red-Yellow antennas, the effect of an LNA, and the data from DopTrackBox and DopTrack. Experiment 2 looks at different hardware configurations of DopTrackBox and uses the manually pointed Yagi antenna. Its subexperiments look at the effect of using the GPS clock, using a different SDR, and manually aiming. A methodology was devised to standardise the data acquisition flow to reduce the number of inconsistencies between recordings.

DopTrackBox' original concept was to use a Raspberry Pi 3B computer, but during the verification and validation phase it was found out that the Raspberry Pi is not powerful enough to run the required software. The hardware verification entails running several recordings at specific frequencies for a set time. The Raspberry Pi crashed in the majority of the test runs, thus a different computer had to be used. Due to time constraints a virtual machine running DopTrackBox' software was chosen as the computer. The VM passed the verification without issues. The software was verified by looking at the inputs and outputs and comparing it against the known outputs from DopTrack.

The data was recorded from 29 August 2022 to 16 November 2022. During Experiment 1 DopTrackBox performed on par with DopTrack, with a 0.6 dB or 21% higher mean SNR and a 8.4 m/s or 12% smaller RMS range rate difference. The Red-Yellow antenna yields a 0.9 dB or 27% higher SNR and a 15.6 m/s or 22% lower RMS range rate difference compared to the Green antenna when both are used on DopTrackBox. It is beneficial to use the LNA as the SNR is 0.6 dB or 22% higher and the Doppler shift data acquisition is more consistent. One out of four data sets without LNA contained useful data, compared to all seven data sets with LNA. Experiment 2 showed that manually pointing the antenna has a big impact on the system performance, resulting in a 1.0 dB or 19% lower and less consistent SNR and a 147 m/s or 547% higher RMS range rate difference. Due to imbalanced data sets and the major impact of manually pointing no decisive conclusions can be drawn on the effect of the GPS clock and different SDR.

The largest improvements can be made by reducing the negative effects caused by manually pointing the antenna, as data from Experiment 2 has generally lower SNRs and higher RMS and maximum range rate differences. It is recommended to gather more data to look into the effect of the GPS clock and SDR, whilst limiting the effect of manually pointing. The minimum requirements for the computer can also be investigated to see whether a fully integrated DopTrackBox system is possible. In the future new features for DopTrackBox could be added, like UHF satellite data recordings or using an LNA on the manual antenna. Software improvements have the lowest priority, but making it easier to change ground station details and automating data processing could increase the system's ease of use.

The outcome of this thesis is that DopTrackBox is a viable satellite Doppler tracking ground station. Even though the original concept was not able to function as intended, the current iteration of DopTrackBox can. More research is needed to look into the specific effects of the GPS clock and different SDR.

List of Abbreviations & Symbols

Abbreviation	Definition
COTS	Commercial of the Shelf
DTB	DopTrackBox
DTB-VM	DopTrackBox Virtual Machine
DTGS	DopTrack Ground Station
DTPP	DopTrack Processing Package
LEO	Low Earth Orbit
LNA	Low Noise Amplifier
NORAD ID	North American Aerospace Defence Identification number
PoE	Power over Ethernet
RMS	Root Mean Square
RRD	Range Rate Difference
SDR	Software Defined Radio
SNR	Signal to Noise Ratio
TCA	Time of Closest Approach
TLE	Two-line Element set
UHF	Ultra High Frequency (300 - 1000 MHz)
VHF	Very High Frequency (30 - 300 MHz)
VM	Virtual machine

Symbol	Definition	Unit
a	Semi-major axis	[km]
b_s	Bits per sample	[b]
c	Speed of light	[m/s]
e	Orbit eccentricity	[-]
F_s	File size	[B]
f	Frequency	[Hz]
f_0	Transmitted frequency	[Hz]
r	Orbital radius	[km]
s_r	Sample rate	[Hz]
t	Time	[s]
V	Orbital velocity	[m/s]
V_c	Circular orbit velocity	[m/s]
v_r	Velocity of signal receiver	[m/s]
v_s	Velocity of signal source	[m/s]
Δv	Range rate	[m/s]
θ	True anomaly	[°]
λ	Wavelength	[m]
μ	Mean	[-]
μ	Gravitational parameter	[km^3/s^2]
σ	Standard Deviation	[-]

List of Figures

2.1 Atmospheric opacity for electromagnetic radiation of different wavelengths. [5] 4
2.2 Photograph of the DopTrack Ground Station and DakLab at the Faculty of Aerospace Engineering. 5
2.3 Photograph of the DopTrack antenna setup. The UHF antenna is on the left, the VHF antennas are on the right. The Green VHF antenna is in the back of the photo and the Red-Yellow VHF antenna is in the front of the photo. 6
2.4 Photo of the DopTrackBox computer, external storage, SDR, and GPS clock. 11
2.5 Schematic diagram of all DopTrackBox hardware components. 11
2.6 Photo of the Raspberry Pi 3B computer used in DopTrackBox. 12
2.7 Images of the RSP1A and the RPSduo. [16, 17] 12
2.8 Picture of the GPS clock with the power cable, GPS antenna, and resistance. 13
2.9 Photo of the Yagi antenna used with DopTrackBox with the fully extended VHF prongs. 13
2.10 DopTrackBox Yagi antenna gain pattern. 14
2.11 DopTrackBox scheduler software flow schematic. 15
2.12 Example of a yaml file from DopTrackBox. 16
2.13 DopTrackBox recorder software flow schematic. 18
2.14 DopTrackBox processor software flow schematic for SNR data and spectrograms. 19
2.15 DopTrackBox processor software flow schematic for Doppler shift data. 20
2.16 Schematic diagram DopTrackBox Virtual Machine hardware components. 22

4.1 Main satellite selection matrix. 30
4.2 A photo of the Nayif-1 cubesat. [27] 31
4.3 Panorama taken from the location where the field recordings were made with the cardinal directions. Due to the way the panorama was taken the cardinal directions are not evenly spaced out. The dashed red line indicates an elevation of approximately 15°. 33

5.1 The original signal compared to method 1 and 2 to define the noise in the signal. 40
5.2 Noise as determined by method 1 and 2 with the respective average values to set as the noise floor. 41
5.3 Spectrogram and noise floor for Data set 1 (NAYIF_42017_202209021146). 41

6.1 Data set 1 (NAYIF_42017_202209021146) different SNR values. 45
6.2 Spectrogram and noise floor for Data set 1 (NAYIF_42017_202209021146). 46
6.3 Experiment 1.1 SNR. The top plot shows results from the Green antenna and the bottom plot shows results from the Red-Yellow antenna. 48
6.4 Experiment 1.2 SNR. The top plot shows results from the Green antenna with the LNA on and the bottom plot shows results from the Green antenna with the LNA off. 50
6.5 Experiment 1.3 SNR comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA on (Data sets 1, 2, and 3). Each plot represents its own data set with the corresponding DopTrack reference. 51

6.6 Experiment 1.3 SNR comparison for DopTrackBox (blue) and DopTrack (orange) for Red-Yellow LNA on (Data sets 4, 5, 6, and 7). Each plot represents its own data set with the corresponding DopTrack reference. . . . 52
6.7 Experiment 1.3 SNR comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA off (Data sets 9, 10, and 11). Each plot represents its own data set with the corresponding DopTrack reference. . . . 52
6.8 Experiment 1.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA off (Data set 10). . . . 55
6.9 Experiment 1.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA on (Data sets 1, 2, and 3). Each plot represents its own data set with the corresponding DopTrack reference. . . . 56
6.10 Experiment 1.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for Red-Yellow LNA on (Data sets 4, 5, 6, and 7). Each plot represents its own data set with the corresponding DopTrack reference. . . . 57
6.11 Experiment 1.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA off (Data set 10). . . . 57
6.12 Experiment 2.1 SNR. The top plot shows results from DTB (without GPS) and the bottom plot shows results from DTB Pro (with GPS). . . . 61
6.13 Experiment 2.1 Range Rate Difference. The top plot shows results from DTB (without GPS) and the bottom plot shows results from DTB Pro (with GPS). . . . 63
6.14 Experiment 2.2 SNR. The top plot shows results from DTB (RSPduo) and the bottom plot shows results from DTB Light (RSP1A). . . . 65
6.15 Experiment 2.2 Range Rate Difference. The top plot shows results from DTB (RSPduo) and the bottom plot shows results from DTB Light (RSP1A). . . . 67
6.16 Experiment 2.3 SNR comparison for DopTrackBox (blue) and DopTrack (orange) for Data sets 12, 13, and 14. Each plot represents its own data set with the corresponding DopTrack reference. 68
6.17 Experiment 2.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB configuration (Data sets 12 and 14). Each plot represents its own data set with the corresponding DopTrack reference. . . . 70
6.18 Experiment 2.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Pro configuration (Data sets 15, 16, 17, and 18). Each plot represents its own data set with the corresponding DopTrack reference. . . . 71
6.19 Experiment 2.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Pro configuration (Data sets 19 and 20). Each plot represents its own data set with the corresponding DopTrack reference. . . . 71
6.20 Experiment 2.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Light configuration (Data sets 21, 22, 23, and 24). Each plot represents its own data set with the corresponding DopTrack reference. . . . 72
6.21 Experiment 2.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Light configuration (Data set 25). Each plot represents its own data set with the corresponding DopTrack reference. . . . 72

A.1 Data set 2 (NAYIF_42017_202209021319) different SNR values. . . . 99
A.2 Spectrogram and noise floor for Data set 2 (NAYIF_42017_202209021319). . . . 99
A.3 Data set 3 (NAYIF_42017_20221014114) different SNR values. . . . 100
A.4 Spectrogram and noise floor for Data set 3 (NAYIF_42017_20221014114). . . . 100
A.5 Data set 4 (NAYIF_42017_202208311119) different SNR values. . . . 101
A.6 Spectrogram and noise floor for Data set 4 (NAYIF_42017_202208311119). . . . 101
A.7 Data set 5 (NAYIF_42017_202209011046) different SNR values. . . . 102
A.8 Spectrogram and noise floor for Data set 5 (NAYIF_42017_202209011046). . . . 102
A.9 Data set 6 (NAYIF_42017_202209011219) different SNR values. . . . 103
A.10 Spectrogram and noise floor for Data set 6 (NAYIF_42017_202209011219). . . . 103
A.11 Data set 7 (NAYIF_42017_202209011352) different SNR values. . . . 104

A.12 Spectrogram and noise floor for Data set 7 (NAYIF_42017_202209011352). 104
A.13 Data set 8 (NAYIF_42017_202208291225) different SNR values. 105
A.14 Spectrogram and noise floor for Data set 8 (NAYIF_42017_202208291225). 105
A.15 Data set 9 (NAYIF_42017_202208291359) different SNR values. 106
A.16 Spectrogram and noise floor for Data set 9 (NAYIF_42017_202208291359). 106
A.17 Data set 10 (NAYIF_42017_202208301152) different SNR values. 107
A.18 Spectrogram and noise floor for Data set 10 (NAYIF_42017_202208301152). 107
A.19 Data set 11 (NAYIF_42017_202208301325) different SNR values. 108
A.20 Spectrogram and noise floor for Data set 11 (NAYIF_42017_202208301325). 108
A.21 Experiment 1.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA on (Data sets 1, 2, and 3). Each plot represents its own data set with the corresponding DopTrack reference. 109
A.22 Experiment 1.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for Red-Yellow LNA on (Data sets 4, 5, 6, and 7). Each plot represents its own data set with the corresponding DopTrack reference. 109
A.23 Data set 12 (NAYIF_42017_202210181158) different SNR values. 110
A.24 Spectrogram and noise floor for Data set 12 (NAYIF_42017_202210181158). 110
A.25 Data set 13 (NAYIF_42017_202210181332) different SNR values. 111
A.26 Spectrogram and noise floor for Data set 13 (NAYIF_42017_202210181332). 111
A.27 Data set 14 (NAYIF_42017_202210251229) different SNR values. 112
A.28 Spectrogram and noise floor for Data set 14 (NAYIF_42017_202210251229). 112
A.29 Data set 15 (NAYIF_42017_202210261153) different SNR values. 113
A.30 Spectrogram and noise floor for Data set 15 (NAYIF_42017_202210261153). 113
A.31 Data set 16 (NAYIF_42017_202210281214) different SNR values. 114
A.32 Spectrogram and noise floor for Data set 16 (NAYIF_42017_202210281214). 114
A.33 Data set 17 (NAYIF_42017_202211021121) different SNR values. 115
A.34 Spectrogram and noise floor for Data set 17 (NAYIF_42017_202211021121). 115
A.35 Data set 18 (NAYIF_42017_202211031045) different SNR values. 116
A.36 Spectrogram and noise floor for Data set 18 (NAYIF_42017_202211031045). 116
A.37 Data set 19 (NAYIF_42017_202211041141) different SNR values. 117
A.38 Spectrogram and noise floor for Data set 19 (NAYIF_42017_202211041141). 117
A.39 Data set 20 (NAYIF_42017_202211141014) different SNR values. 118
A.40 Spectrogram and noise floor for Data set 20 (NAYIF_42017_202211141014). 118
A.41 Data set 21 (NAYIF_42017_202211071125) different SNR values. 119
A.42 Spectrogram and noise floor for Data set 21 (NAYIF_42017_202211071125). 119
A.43 Data set 22 (NAYIF_42017_202211101108) different SNR values. 120
A.44 Spectrogram and noise floor for Data set 22 (NAYIF_42017_202211101108). 120
A.45 Data set 23 (NAYIF_42017_202211111032) different SNR values. 121
A.46 Spectrogram and noise floor for Data set 23 (NAYIF_42017_202211111032). 121
A.47 Data set 24 (NAYIF_42017_202211151110) different SNR values. 122
A.48 Spectrogram and noise floor for Data set 24 (NAYIF_42017_202211151110). 122
A.49 Data set 25 (NAYIF_42017_202211161033) different SNR values. 123
A.50 Spectrogram and noise floor for Data set 25 (NAYIF_42017_202211161033). 123
A.51 Experiment 2.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB configuration (Data sets 12 and 14). Each plot represents its own data set with the corresponding DopTrack reference. 124
A.52 Experiment 2.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Pro configuration (Data sets 15, 16, 17, and 18). Each plot represents its own data set with the corresponding DopTrack reference. 124

A.53 Experiment 2.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Pro configuration (Data sets 19 and 20). Each plot represents its own data set with the corresponding DopTrack reference. 124
A.54 Experiment 2.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Light configuration (Data sets 21, 22, 23, and 24). Each plot represents its own data set with the corresponding DopTrack reference. 125
A.55 Experiment 2.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Light configuration (Data set 25). Each plot represents its own data set with the corresponding DopTrack reference. 125

List of Tables

2.1 Different radio spectrum bands following the IEEE 2019 standard. [6] . . . 4
2.2 Initial DopTrackBox concept Hardware and Software. . . . 7
2.3 Different DopTrackBox concepts used for data acquisition. . . . 7
2.4 Overview of Operational Requirements (DTB-OPS) for DopTrackBox. . . . 9
2.5 Overview of Technical Requirements (DTB-TECH) for DopTrackBox. . . . 10
2.6 rec.list details used for DopTrackBox. . . . 16
2.7 DopTrackBox Hardware and Software during recorder software determination. . . . 21

4.1 Experiment 1 subexperiments and subgoals. . . . 32
4.2 Experiment 2 subexperiments and subgoals. . . . 32

5.1 DopTrackBox Hardware and Software overview during radio test verification. . . . 35
5.2 Radio test results with RSPduo on Raspberry Pi . . . 35
5.3 Radio test results on Raspberry Pi with various hardware, sample rates, and number of samples at 145.870MHz . . . 36
5.4 Updated computer hardware specifications . . . 37
5.5 DopTrackBox virtual machine (DTB-VM) initial specifications . . . 37
5.6 DopTrackBox virtual machine (DTB-VM) final specifications . . . 38
5.7 Obsolete and removed DopTrackBox requirements after verification and validation. . . . 42
5.8 Updated DopTrackBox requirements after verification and validation. . . . 43
5.9 New DopTrackBox requirements after verification and validation. . . . 43

6.1 DopTrackBox hardware and software used for data acquisition during Experiment 1. . . . 47
6.2 Overview table of all measurements for Experiment 1. Azimuth column shows the azimuth of the start of recording and azimuth at the end of the recording, along with if the pass was to the east or west of the ground station. . . . 47
6.3 Data used to achieve the goal of Experiment 1.1. . . . 48
6.4 Overview of mean values of noise floor and SNR for Experiment 1.1 Lower noise floor values and higher SNR values are better. . . . 49
6.5 Data used to achieve the goal of Experiment 1.2. . . . 49
6.6 Overview of mean values of noise floor and SNR for Experiment 1.2 Lower noise floor values and higher SNR values are better. . . . 50
6.7 Data used to achieve the goal of Experiment 1.3. . . . 51
6.8 Overview of mean values of noise floor for Experiment 1. Lower values are better. . . . 53
6.9 Overview of mean values of the mean SNR for Experiment 1. Higher values are better. . . . 53
6.10 Frequency data availability of DopTrackBox and DopTrack for Experiment 1.3. . . . 55
6.11 RMS of range rate difference for DopTrackBox and DopTrack reference data for Experiment 1.3. Lower values are better. . . . 58
6.12 Overview of maximum range rate difference (RRD) and the time at which this maximum occurs for Experiment 1.3. . . . 58
6.13 Overview of the TCA, the difference of the TCA between the TLE and measurement, and the slope of the linear fit of the first residual of DopTrackBox and DopTrack for Experiment 1.3. . . . 59

6.14 DopTrackBox hardware and software used for data acquisition during Experiment 2. 59
6.15 Overview table of all measurements for Experiment 2. Azimuth column shows the azimuth of the start of recording, azimuth at maximum elevation, and azimuth at the end of the recording, along with whether the pass was to the east or west of the ground station. 60
6.16 Experiment 2.1 goal and data overview. 61
6.17 Overview of mean values of noise floor and SNR for Experiment 2.1 Lower noise floor values and higher SNR values are better. 62
6.18 Frequency data availability of DopTrackBox and DopTrack for Experiment 2.1. 63
6.19 Overview of DopTrackBox range rate difference (RRD) RMS, time and value of maximum RRD, TCA, the difference of the TCA between the TLE and measurement, and the slope of the linear fit of the first residual for Experiment 2.1. 64
6.20 Experiment 2.2 goal and data overview. 65
6.21 Overview of mean values of noise floor and SNR for Experiment 2.2 Lower noise floor values and higher SNR values are better. 66
6.22 Frequency data availability of DopTrackBox and DopTrack for Experiment 2.2. 66
6.23 Overview of DopTrackBox range rate difference (RRD) RMS, time and value of maximum RRD, TCA, the difference of the TCA between the TLE and measurement, and the slope of the linear fit of the first residual for Experiment 2.2. 67
6.24 Experiment 2.3 goal and data overview. 68
6.25 Overview of mean values of noise floor for Experiment 2.3. Lower values are better. 69
6.26 Overview of mean values of the mean SNR for Experiment 2.3. Higher values are better. 69
6.27 Frequency data availability of DopTrackBox and DopTrack for Experiment 2.3. 70
6.28 RMS of range rate difference for DopTrackBox and DopTrack reference data for Experiment 2.3. Lower values are better. 73
6.29 Overview of maximum range rate difference (RRD) and the time at which this maximum occurs for Experiment 2.3. 73
6.30 Overview of the TCA, the difference of the TCA between the TLE and measurement, and the slope of the linear fit of the first residual of DopTrackBox and DopTrack for Experiment 2.3. . . . 74

7.1 Overview of average performance parameters for the Green DopTrackBox antenna LNA on and LNA off cases. Higher values are better. 76
7.2 Overview of average performance parameters for the Red-Yellow and Green DopTrackBox antenna cases. Higher values are better, except for RMS RRD. 76
7.3 Overview of average performance parameters for Research Question 2.1. Lower values are better, except for SNR and Data quality. 78
7.4 Overview of average performance parameters for Research Question 2.2. Lower values are better, except for SNR. 79
7.5 Overview of average performance parameters for Research Question 2.3. Lower values are better, except for SNR. 80
7.6 Overview of average performance parameters for Research Question 2.4. Lower values are better, except for SNR and Data quality. 81

8.1 Obsolete and removed DopTrackBox requirements. 83
8.2 Updated DopTrackBox requirements. 84
8.3 Requirement pass of DTB-OPS-1. 84
8.4 Requirement pass of DTB-OPS-2. 84
8.5 Requirement pass of DTB-TECH-1. 85
8.6 Requirement pass of DTB-TECH-2. 85
8.7 Requirement pass of DTB-TECH-3. 85
8.8 Requirement pass of DTB-TECH-4. 85
8.9 Requirement pass of DTB-TECH-5. 85
8.10 Requirement pass of DTB-TECH-6. 86

8.11 Requirement pass of DTB-TECH-7. 86
8.12 Final Operational Requirements (DTB-OPS) for DopTrackBox. 87
8.13 Final Technical Requirements (DTB-TECH) for DopTrackBox. 87
8.14 Overview of hardware component prices. 88

1 Introduction

Small satellites are increasing in popularity as their performance keeps improving. They show a major cost reduction over their larger counterparts allowing space to become more accessible for more parties. An important part of having a satellite is communicating with it, and retrieving the data that is created. At the TU Delft research is being done on satellite tracking and communications. One of these projects is the Doppler Tracking experiment, also known as DopTrack. DopTrack receives and records communication signals from Low Earth Orbit (LEO) satellites to determine the satellite's position and velocity. To do so, a ground station is located on the roof and upper floor of the Faculty of Aerospace Engineering in Delft. The ground station can receive both VHF and UHF signals which can be used for Doppler tracking.

This report delves into the feasibility of creating a portable Doppler tracking station with commercial of the shelf (COTS) components. This solution is called DopTrackBox (DTB), and shall work as an in situ ground station for Doppler tracking of LEO satellites with functionalities similar to the DopTrack ground station. It is therefore based on the same technology as the DopTrack station, and uses similar software protocols. DopTrackBox should work as a readily available solution with low cost hardware to bring satellite tracking or communication to a greater audience. These can be research groups in education or small businesses. The main factors that decide the feasibility of the DopTrackBox concept are the quality of the obtained data and the development of the hardware and software. The data generated by DopTrackBox will be broken down into the range rate and the signal to noise ratio to gain insight in the system's performance compared to the DopTrack station.

The research is focused around a central objective with a set of research questions. The main objective of the thesis is:

To analyse DopTrackBox' viability as a Doppler tracking ground station.

To gain insight into the viability of DopTrackBox, three main research questions have been formed:
Research Question 1 is *"Can the proposed DopTrackBox concept be build?"*. This question delves into whether the hardware and software are capable of executing the protocols required for Doppler tracking. If no suitable COTS hardware can be found to reliably yield results, the scope of what DopTrackBox is needs to be revised.

Research Question 2 has been defined as *"How do the Doppler and SNR data of DopTrackBox compare to that of DopTrack?"*. It is important that the obtained data can still be used for satellite tracking. Therefore, the DopTrack station will act as a baseline. Different experiments will be performed to analyse different aspects of DopTrackBox, and to see the effect of differences in hardware configurations.

Finally, Research Question 3 is stated as *"Which improvements can be made to enhance DopTrackBox?"*. These improvements can be ease of use for hardware operation or data quality improvements. They will also show the limits of the current system and will provide steps in ensuring optimal use of DopTrackBox.
All research questions can be divided in subquestions for further clarification of the different aspects that are involved in answering these questions. These will be discussed in their relevant chapters.

The structure of this report is the following: The DopTrackBox concept will be discussed in Chapter 2. This includes the theory behind DopTrack and DopTrackBox concept, the design goals, and the design requirements.
Next, a setup manual will be provided regarding the DopTrackBox hardware and software in Chapter 3. The goal of this manual is to make it easier for others to replicate DopTrackBox and get it up and running.
The next four chapters relate to the experiments that were performed to gather the DopTrackBox data. Chapter 4 delves into the set-up of the experiments, the satellite that is tracked, and the challenges and methodology for the experiments.
Before any real data can be gathered, the system must be verified and validated. This process is described in Chapter 5. This chapter will also provide the necessary information to answer Research Question 1.

Once the system has been verified and validated, the experiments can be executed. The results of all the experiments are presented in Chapter 6. Here all the relevant results are shown which are used in the next chapter for discussion and analysis.
Logically, Chapter 7 uses the results and analyses them to answer the posed research questions. Therefore Research Question 2 will be answered in this chapter.
With all data acquired and the hardware tested, system improvements are discussed in Chapter 8. Research Question 3 will be answered here.
Finally, Chapter 9 concludes the findings of this thesis.

2 | DopTrackBox Concept

This chapter functions as an introduction to DopTrack and DopTrackBox. Some background on Doppler tracking, and thereby the functioning of DopTrack and DopTrackBox, will be provided in Section 2.1. After this has been established, the DopTrackBox design concept itself will also be explicitly explained in Section 2.2. Research Question 1 follows from this section as well, and reasoning behind it will be provided here. To answer this question the experimental data will be needed which is presented in Chapter 6. Naturally, this question can therefore not be answered within this chapter. Now that the concept behind DopTrackBox has been explained the design goals for the system are presented in Section 2.3. The design requirements follow from the design goals and the concepts, and they will therefore be presented in Section 2.4. With the design requirements and goals set, the system can be build. Since the hardware is of key importance, Section 2.5 elaborates upon DopTrackBox' different hardware components. DopTrackBox cannot function without the different software programs that make it all work. Therefore they are explained in Section 2.6. Finally, some concept iterations have been made during this set-up stage of the research after some initial tests. These are shown in Section 2.7.

2.1. DopTrack Background

This section provides some background information on the DopTrack project, which DopTrackBox is a part of. There are three main areas of interest, each of which is discussed in its own section. The basics of the Doppler effect are explained in Section 2.1.1. With this basic knowledge, Doppler tracking can be explained in Section 2.1.2. This forms the base principle on which DopTrack and DopTrackBox are build. The final section, Section 2.1.3, dives into the DopTrack project of the TU Delft.

2.1.1. The Doppler Effect

The Doppler effect is the perceived difference in frequency of a wave signal if there is relative motion between the transmitter and observer [1]. It was first described by C.A. Doppler in 1842. A well known occurrence of this effect for most is the change in pitch of the siren of a passing emergency vehicle. The pitch is higher when the vehicle approaches the observer and it becomes lower when it moves away from the observer. The Doppler effect, or Doppler shift, is not exclusive to sound waves, but is applicable to all waves. In astronomy the Doppler effect is used for example to determine a planet's rotation rate. This can be done actively using radar or passively using visible light. The part of the planetary disk that moves towards the observer sees a positive Doppler shift, and the part that moves away sees a negative shift. This is also known as blueshift and redshift, as blue light has a higher frequency and shorter wavelength and red light has a lower frequency and longer wavelength [2]. Frequency and wavelength are connected to each other via the relation shown in Equation 2.1, where f is the frequency, c the speed of light, and λ the wavelength.

$$f = \frac{c}{\lambda} \tag{2.1}$$

A first order approximation of the Doppler effect can be described by Equation 2.2. Here v_r is the velocity of the receiver, v_s the velocity of the source, and f_0 the transmitted frequency. The velocity of the receiver is added to the speed of light if it moves closer to the source, and subtracted if it moves away. Conversely, the velocity of the source is subtracted from the speed of light if it moves closer to the receiver, and added when it moves away [3].

$$f = \left(\frac{c \pm v_r}{c \mp v_s}\right) f_0 \tag{2.2}$$

In DopTrackBox' case the velocity of the receiver and source are significantly lower than the speed of light. Therefore Equation 2.2 can be simplified and the frequency change can be approximated by Equation 2.3 [4].

$$f = \left(1 + \frac{\Delta v}{c}\right) f_0 \tag{2.3}$$

In Equation 2.3 Δv is the difference between the velocity of the receiver and velocity of the source; $v_r - v_s$, and is called the range rate.

Another effect that ties both into astronomy and Doppler tracking is using the Doppler tracking data from spacecraft to determine the mass of a celestial body. As the Doppler shift can be measured very precisely, it can be used to determine the forces acting on the spacecraft that alter its orbit. This gives insight in the gravitational pull of a body, and thus its mass [2].

2.1.2. Doppler Tracking

The application of the Doppler effect on which DopTrack and DopTrackBox are build is Doppler tracking. This is the tracking of a satellite's location by looking at the Doppler shift of received radio signals. Not all kinds of electromagnetic radiation can be used for Doppler Tracking. Figure 2.1 shows that radio waves are fully led through by the atmosphere, but other parts of the electromagnetic spectrum are fully or partially blocked. Using Equation 2.1, the wavelengths of 1 cm to 10 m can be converted to a frequency of roughly 0.03 through 30 GHz.

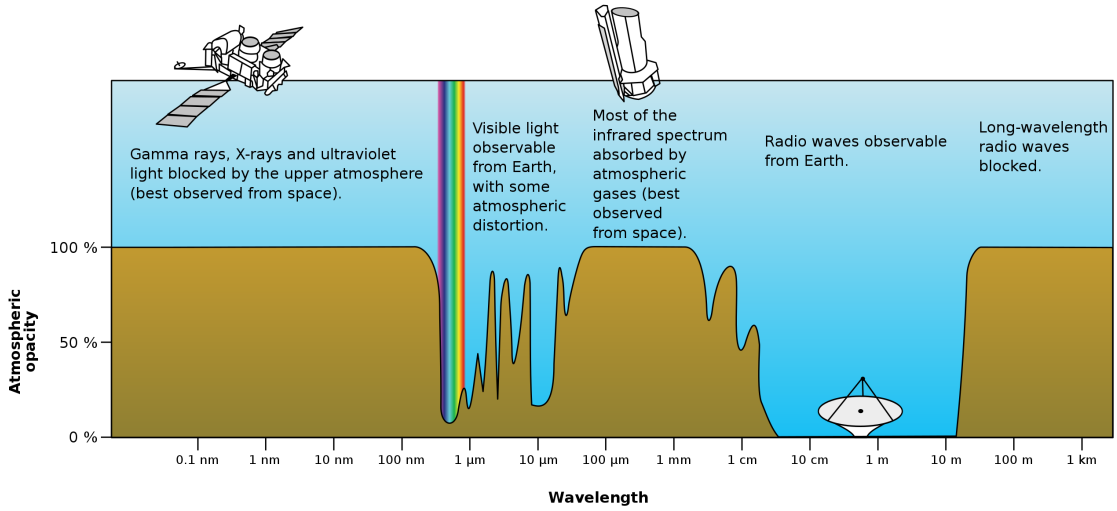


Figure 2.1: Atmospheric opacity for electromagnetic radiation of different wavelengths. [5]

The radio spectrum is divided into different bands. There are several standards dividing the radio spectrum, of which the IEEE and ITU are two. An overview of the different radio bands can be seen in Table 2.1. There is some ambiguity regarding the UHF band, as the ITU designation for UHF goes from 0.3 GHz to 3 GHz which includes the IEEE L band and part of the S band [6]. When UHF in this report is mentioned the IEEE standard is followed, meaning frequencies from 300 MHz to 1 GHz. There is no such ambiguity for VHF, as both the IEEE and ITU designations refer to the same frequency range.

Table 2.1: Different radio spectrum bands following the IEEE 2019 standard. [6]

Band	Frequency Range [Ghz]	Band	Frequency Range [Ghz]
HF	0.003 - 0.03	K_u	12 - 18
VHF	0.03 - 0.3	K	18 - 27
UHF	0.3 - 1	K_a	27 - 40
L	1 - 2	V	40 - 75
S	2 - 4	W	75 - 110
C	4 - 8	mm	110 - 300
X	8 - 12	THz	300 - 1000

From the Doppler shift the range rate can be obtained, which is the spacecraft's line of sight velocity in relation to the ground station. According to *Montenbruck and Gill* [7], the range rate can be determined with an accuracy of 1 mm/s. The range rate of a satellite is negative when the satellite moves towards the ground station, and increases when it moves closer. It is zero when exactly overhead, and increases further when moving away from the ground station. The moment at which the satellite is overhead can be used to determine the transmitted frequency f_0 from Equation 2.3, as at that time Δv is zero. This moment is known as the Time of Closest Approach and can be used to obtain the transmitted frequency from the recorded Doppler data [5]. With the range rate of a satellite known, its orbit can be constructed by using Equation 2.4. This equation is known as the Vis-Viva equation with V the orbital velocity, μ the gravitational parameter of the celestial body, which is 398600.441 km^3/s^2 for Earth [8], r the orbital radius, and a the semi-major axis of the orbit.

$$\frac{V^2}{2} - \frac{\mu}{r} = -\frac{\mu}{2a} \tag{2.4}$$

There are different ways of conducting Doppler tracking measurements. Examples of those are either two-way or one-way Doppler tracking. A one-way Doppler tracking measurement is done by transmitting a signal from either the ground station or the satellite, and using the receiving platform to determine the Doppler shift. For a two-way measurement a signal is transmitted by the ground station to the satellite, who transmits it back to the ground station at a different frequency to avoid signal interference. Since the signal travels both up to the satellite and down to the ground station, the Doppler shift is twice as big as with a one-way measurement. If the signal transmitted by the satellite back to Earth is not send to the original ground station but to another, it is a three-way measurement [7].

2.1.3. DopTrack

DopTrack is a Doppler tracking project at the Faculty of Aerospace Engineering of the TU Delft. Its main purpose is to calculate the range rate of several satellites for educational and research purposes [9]. It started in 2015 with tracking TU Delft's Delfi-C3 satellite. At the time the ground station was located at the highest building of the TU Delft campus, the EWI building. Due to the planned demolition of the building the DopTrack ground station had to move, and as of 2022 it has been relocated to the Aerospace Engineering building where it is located in the DakLab, 'Rooftop laboratory'. A look inside the DakLab can be seen in Figure 2.2. The two monitors on the wall and the monitor on the right are connected to an SDR and one of the VHF antennas to monitor satellite passes. The monitor on the left, which is turned off, can be used to directly interface with DopTrack.

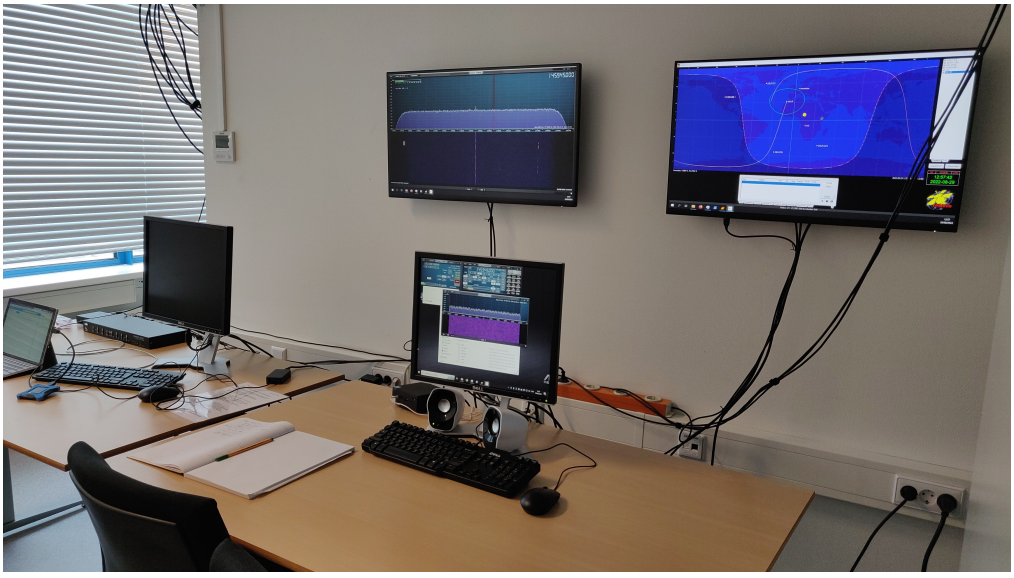


Figure 2.2: Photograph of the DopTrack Ground Station and DakLab at the Faculty of Aerospace Engineering.

DopTrack is a one way Doppler tracking system where it uses the radio signals transmitted by a satellite to determine the Doppler shift. Due to this DopTrack can track any satellite, instead of the satellite requiring specific hardware or to be made specifically compatible with DopTrack. Key satellites tracked by DopTrack are Delfi-C3 (32789), Delfi-N3XT (39428), Nayif (42017), and Funcube (39444). The numbers behind the satellite name are their NORAD IDs, which are unique satellite identification numbers. On the roof are three omnidirectional antennas in total. Two of those are VHF antennas and the other one is for receiving UHF signals. An advantage of these antennas being omnidirectional is that they do not need to be pointed to the satellite whilst it passes. VHF covers the frequency range of 30 to 300 MHz, whereas UHF covers 300 to 1000 MHz. At the time of writing, DopTrack does not track any satellites using the UHF antenna. An overview of the antennas can be seen in Figure 2.3. The smaller antenna on the left is the UHF antenna, and the two bigger ones on the right are the VHF antennas, which have a designated colour code corresponding to the colour of the cable entering the ground station. This colour is there to distinguish between the two VHF antennas. All of them are connected to an LNA to enhance the power of the received signal.

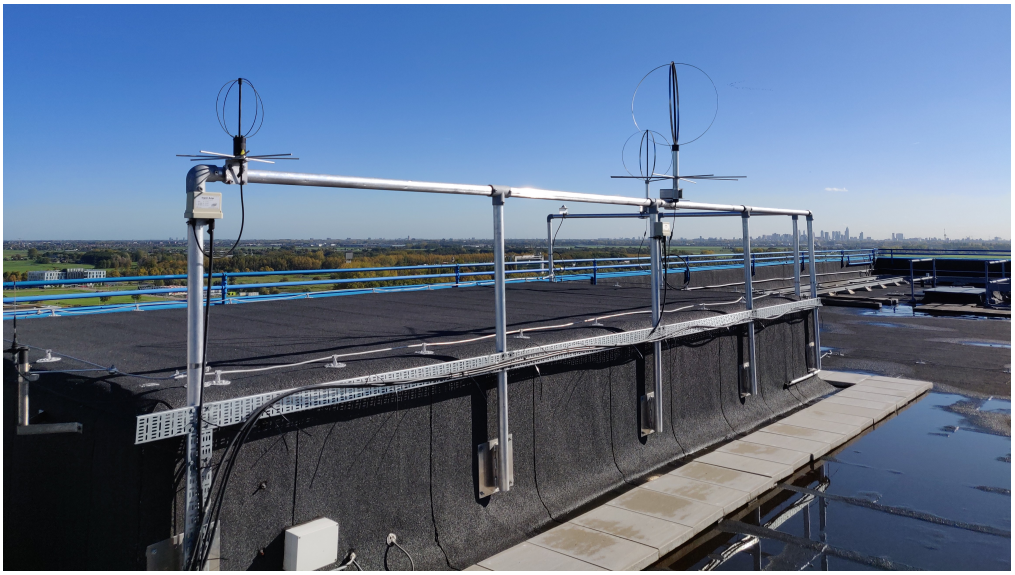


Figure 2.3: Photograph of the DopTrack antenna setup. The UHF antenna is on the left, the VHF antennas are on the right. The Green VHF antenna is in the back of the photo and the Red-Yellow VHF antenna is in the front of the photo.

Since the antennas are located at the roof of the Aerospace Engineering building they have an almost completely unobstructed view of the sky. Therefore a low elevation threshold can be used to record passing satellites. Depending on the satellite, DopTrack automatically chooses the correct antenna for the recording. The signal from the antenna is passed to the dedicated AOR5001DX radio. The radio then sends the signal to the USRP-N210 Software Defined Radio (SDR) to convert the analogue signal to a digital one. A GPS receiver and clock are used to synchronise all the hardware components of DopTrack [5].

The functionalities of DopTrack are still being expanded and actively worked on through the DopTrack Github. All DopTrack's results are available online for anyone to view on the online DopTrack dashboard at *doptrack.tudelft.nl* [10].

2.2. DopTrackBox Concept Description

DopTrackBox is as a light weight and portable satellite ground station based on the DopTrack ground station. Its main function is therefore satellite Doppler tracking, as was explained in Section 2.1. DopTrackBox is a feasibility study to see if lower grade hardware in a portable form factor can obtain adequate results for Doppler tracking. This thesis builds upon a preliminary study conducted as part of the Spaceflight minor, which investigated the feasibility of a specific hardware configuration for DopTrackBox. The hardware and software used during this study acted as the starting point for further development of DopTrackBox. An important finding from this study to keep into account is the temperature sensitivity of the Raspberry Pi, which is why it needs to be equipped with a CPU cooler [11]. DopTrackBox has five main hardware components: the computer, Software Defined Radio (SDR), GPS clock, antenna, and storage. An overview of the initial concept based on the previous study can be seen in Table 2.2.

Table 2.2: Initial DopTrackBox concept Hardware and Software.

Hardware					*Software*	
Computer	**SDR**	**GPS Clock**	**Antenna**	**Storage**	**OS**	**Recorder**
Raspberry Pi 3B	LogiLink VG0002A	None	LogiLink	8 GB SD card	Raspberry Pi OS	GQRX

Of the main components the computer is the heart of the system. One of the key features of DopTrackBox is for it to run on a Raspberry Pi, a readily available COTS component. An SDR is a radio where analogue hardware is replaced by a software or digital implementation [12]. For DopTrackBox there will be different SDRs to test their data quality differences. This has resulted in different DopTrackBox concepts which are shown in Table 2.3.

Table 2.3: Different DopTrackBox concepts used for data acquisition.

Concept	**Hardware Configuration**
DTB Light	SDRplay RSP1A
DTB	SDRplay RSPduo
DTB Pro	SDRplay RSPduo + GPS clock

The LogiLink VG0002A SDR and its accompanying antenna as shown in Table 2.2 will only be used during the setup phase of DopTrackBox and will not be used for data acquisition experiments. From these different concepts follows Research Question 1: *Can the proposed DopTrackBox concepts be build?* To aid in answering this main question two subquestions have been posed. The first of these is "Can the selected hardware combination record the satellite data required for Doppler tracking?" To answer this question, the different concepts have to be build and the required software will have to be installed to run some verification tests. This section will not delve further into this, but Chapter 5 will. The second subquestion is "What are the hardware requirements of DopTrackBox?". This question can be answered once the first subquestion has been answered. This will also tie into the design goals and requirements.

2.3. DESIGN GOALS

In order to define the boundaries within which Research Question 1 must be answered, design goals for DopTrackBox have been established. These goals follow from the intended use case of DopTrackBox and the envisioned hardware and software from Tables 2.2 and 2.3. The design goals mainly focus on the hardware of DopTrackBox and less on the software. This section continues by stating the design goals and elaborating upon them.

2.3.1. INDEPENDENT AND PORTABLE

DopTrackBox shall be fully independent and portable. Independent means that for DopTrackBox to function all the required components shall be included in the DopTrackBox system. DopTrackBox has to be able to operate off the grid whilst gathering data in the field with no access to power or internet. This means that a different power source has to be available for the system, like battery power. For field operations having a portable screen is also required since operating DopTrackBox is easier with visual feedback.

To make DopTrackBox portable, its form factor should be as small as possible without deteriorating its performance. Part of the hardware provided for DopTrackBox is a protective case with internal dimensions of 27 × 15 × 6 cm. The goal is to see if all the DopTrackBox hardware fits in this case, except for the antenna. Comparisons between the hardware in the case and outside the case can be made to see whether there is a significant impact on system performance due to different thermal conditions.
Another factor that ties in with independent operation, but also with the data storage capacity design goal is the fact that having an 8 GB SD card for storage is insufficient for continuous operations. Additional storage is therefore needed, which should be portable and rugged as the risk of losing or corrupting data has to be minimised when operating under potentially harsh conditions in the field.

2.3.2. Minimum Power Consumption
DopTrackBox shall have a minimal power consumption to ensure maximum operational time. Especially when operating in the field and having to rely on battery power it is crucial to choose hardware components that are power efficient. The lower the power consumption of the system, the longer it will last until its batteries have to be recharged. This allows the system to operate longer whilst off the grid which increases DopTrackBox' versatility.

2.3.3. Data Storage Capacity
For DopTrackBox to record multiple satellite passages it shall have additional data storage capacity. For LEO satellites, a pass is around 12 minutes. DopTrack records satellites at 250,000 samples per second, with each sample being a complex float with 32 bits per value. This means a total of 64 bits per sample. The file size can then be calculated by using Equation 2.5, where t is the time of the recording, s_r the sample rate, and b_s the number of bits per sample. This value is divided by 8 to get the file size in bytes instead of bits.

$$F_s = \frac{t s_r b_s}{8} \tag{2.5}$$

Substituting the numbers shows that a recording of 12 minutes with these parameters produces a data file of 1.44 GB. With several recordings per day, and the goal for DopTrackBox to work independently for an extended period of time without offloading the data to a different system. Having removable storage allows for easier data transfers.

2.3.4. Data Transmission
DopTrackBox shall be capable of transmitting the gathered data to DopTrack's servers. This allows the data to be backed up and for it to be more easily accessible. Having the option to transmit the data once DopTrackBox connects to the internet can reduce the total storage capacity that the system needs. This can be done automatically by software or manually. Data transmission between different systems without internet is also a possibility. In that case using removable storage would be recommended.

2.3.5. Thermal Control
DopTrackBox shall be able to gather satellite data without thermal throttling. This design goal has been set up based on previous research on the DopTrackBox hardware [11]. In that preliminary study it was found out that the hardware had a tendency to overheat and thermal throttle when used over extensive periods of time. The goal of making DopTrackBox portable is by using a protective case, which puts the hardware in a small enclosed space which could influence the thermal performance of the hardware. Therefore this design goal has been established to keep this in mind.

2.4. DopTrackBox Requirements
The goals from Section 2.3 have to be turned into requirements for the design that can be easily referenced and checked to see if DopTrackBox fulfils its intended purpose. The requirements can be divided into operational requirements and technical requirements. The former are derived by the operational goals of the system and its intended functionalities. The latter are construed from the operational requirements by translating their intended functionalities into requirements that the software and hardware must achieve. This section has thus been divided into the Operational requirements in Section 2.4.1 and the Technical requirements in Section 2.4.2.

2.4.1. Operational Requirements
First set of requirements are physical requirements for the DopTrackBox system and hardware, and the second set of requirements relate to DopTrackBox' operational use. These sets can be seen in Table 2.4.

The size requirement of *DTB-OPS-1.01* is based on the provided Yagi antenna, which measures 41.5 × 29 × 4.5 cm. This size is for carrying DTB around and not necessarily its size when operating. Requirements *DTB-OPS-1.02* through *DTB-OPS-1.07* state the hardware components needed for DopTrackBox to function.

Table 2.4: Overview of Operational Requirements (DTB-OPS) for DopTrackBox.

Identifier	Requirement
DTB-OPS-1.01	DTB shall be no larger than 0.45 × 0.30 × 0.10 m.
DTB-OPS-1.02	DTB shall include a computer.
DTB-OPS-1.03	DTB shall include an SDR.
DTB-OPS-1.04	DTB shall include a GPS clock.
DTB-OPS-1.05	DTB shall include an antenna.
DTB-OPS-1.06	DTB shall include an external data storage medium.
DTB-OPS-1.07	DTB shall include a battery.
DTB-OPS-2.01	DTB shall be able to record automatically scheduled satellite passes.
DTB-OPS-2.02	DTB shall be able to record unscheduled satellite passes.
DTB-OPS-2.03	DTB shall be able to operate without active internet connection during satellite passes.
DTB-OPS-2.04	DTB shall be able to record satellite passes without crashing.
DTB-OPS-2.05	DTB shall be able to back-up its data to online servers when connected to the internet.
DTB-OPS-2.06	DTB shall be able to operate on battery power for at least 120 minutes.
DTB-OPS-2.07	DTB shall be able to store satellite data of at least 150 passes.
DTB-OPS-2.08	DTB shall be able to record satellite data with a mean SNR of 3.0 dB or higher.
DTB-OPS-2.09	DTB shall be able to record satellite data with a RMS range rate difference of 100 m/s or lower.
DTB-OPS-2.10	The SNR difference between DTB and DT shall be no smaller than -3 dB.
DTB-OPS-2.11	The RMS range rate difference between DTB and DT shall be no larger than 50 m/s.

Requirements *DTB-OPS-2.01* through *DTB-OPS-2.05* were set as operational requirements during the setup phase of the thesis. DopTrackBox should be able to record satellites that have been scheduled, but also have the ability to record unscheduled satellites. The other requirements come from the design goals of Section 2.3; *DTB-OPS-2.03* from the independent operation goal, *DTB-OPS-2.04* based on the thermal control goal, and *DTB-OPS-2.05* on the data transmission goal. For requirement *DTB-OPS-2.06* the operating time of 120 minutes is based on 5 recordings per day each 12 minutes long, with additional 5 minute start and stop time for system setup for a total of 22 minutes per recording. This leaves 10 additional minutes of battery time before it needs to be recharged, which is intended to happen daily. Requirement *DTB-OPS-2.07* is based on the capability to operate DopTrackBox for a longer period of time without other means of offloading the data. Especially when recording data from multiple satellites over the course of one or two weeks it is important to have sufficient storage. With 5 passes per satellite per day, a safe estimate would be to account for 150 passes of storage. The next four requirements, *DTB-OPS-2.08* through *DTB-OPS-2.11* are performance targets for the system to meet. The first two are absolute targets whereas the second two are relative targets with respect to DopTrack. The numbers are based on values from DopTrack.

2.4.2. TECHNICAL REQUIREMENTS

Technical requirements are made by looking at the operational requirements and design goals. They are turned into specifications which DopTrackBox must meet. All technical requirements can be seen in Table 2.5. The first set of technical requirements all relate to the computer, the second set relates to the SDR, the third set to the GPS clock, the fourth set to the antenna, the fifth set to the external storage, and the sixth set to the software. The reasoning for the requirements is presented below the table.

Table 2.5: Overview of Technical Requirements (DTB-TECH) for DopTrackBox.

Identifier	Requirement
DTB-TECH-1.01	The computer shall have at least 3 USB type A ports.
DTB-TECH-1.02	The computer shall have the capability of connecting to the internet.
DTB-TECH-1.03	The computer shall have the means to connect to a screen.
DTB-TECH-1.04	The computer shall have the means to connect to a Human Interface Device for interaction with the system.
DTB-TECH-1.05	The computer shall have a minimum internal storage capacity of 32 GB.
DTB-TECH-1.06	The computer shall be able to run Python 3.9 or newer.
DTB-TECH-1.07	The computer shall be able to process a data stream of 2 MB/s.
DTB-TECH-2.01	The SDR shall have a connection for the antenna.
DTB-TECH-2.02	The SDR shall be connected to the computer via a USB type A connection.
DTB-TECH-2.03	The SDR shall be capable of receiving VHF signals from 30 to 300 MHz.
DTB-TECH-2.04	The SDR shall be capable of receiving UHF signals from 0.3 to 1 GHz.
DTB-TECH-2.05	The SDR shall be capable of sampling at a sample rate of 250 kHz.
DTB-TECH-3.01	The GPS clock shall be physically compatible with the rest of the DTB system.
DTB-TECH-4.01	The antenna shall be capable of receiving VHF signals from 30 to 300 MHz.
DTB-TECH-4.02	The antenna shall be capable of receiving UHF signals from 0.3 to 1 GHz.
DTB-TECH-4.03	The antenna shall be physically compatible with the SDR.
DTB-TECH-5.01	The external storage shall not have any moving parts.
DTB-TECH-5.02	The external storage shall be connected to the computer via a USB type A connection.
DTB-TECH-5.03	The external storage shall have a minimum capacity of 256 GB.
DTB-TECH-6.01	The Operating System shall be based on Linux.
DTB-TECH-6.02	The recorder software shall be compatible with the SDR.

The number and type of USB ports seen in requirement *DTB-TECH-1.01* is based on the number of peripherals that need to be attached for DopTrackBox to function on a most basic level. They are reserved for the external storage, the SDR, and GPS clock. USB type A ports have been chosen since they are the most common type of USB port. The internet connectivity requirement of *DTB-TECH-1.02* ensures that DopTrackBox is capable of receiving the most up to date information for scheduling satellite passes, as well as allowing the system to back-up its information. *DTB-TECH-1.03* and *DTB-TECH-1.04* are put in place to allow users to interact more easily with the system. The internal storage requirement of *DTB-TECH-1.05* is put in place to ensure the system has sufficient storage for the operating system and all the other programs that are needed to run DopTrackBox. The Python version requirement of *DTB-TECH-1.06* is based on the version of Python required to run the data processing software. This software is written with functionality provided by that version of Python or newer in mind. *DTB-TECH-1.07* states a data stream of 2 MB/s. This is based on the recordings of DopTrack, which are done at 250,000 samples per second at 64 bits per sample, resulting in a total data stream of 2 MB/s.

The *DTB-TECH-5.01* requirement ensures the external storage is more rugged, especially whilst travelling or operating DopTrackBox in the field. The minimum storage capacity stated in requirement *DTB-TECH-5.03* is based on the 150 passes seen in requirement *DTB-OPS-2.07* and the basic calculation from Section 2.3.3. 256 GB is the next highest standard data storage capacity from the required minimum of 216 GB.

Regarding the software requirements, *DTB-TECH-6.01* ensures compatibility between DopTrackBox and DopTrack. DopTrack uses a Linux based OS, so therefore programs written for one system can easily be transferred to the other. *DTB-TECH-6.02* puts the SDR over the software, since the SDR is provided for the research. Therefore the software used to record the data should be compatible with it, and not the other way around.

The rest of the requirements are mainly put in place to ensure that all system components are compatible with each other.

2.5. DOPTRACKBOX HARDWARE COMPONENTS

This section shows the different hardware components used in DopTrackBox, and explains the thought behind them. Each different hardware component is explained in its own section. An overview of the DopTrackBox hardware, excluding the antenna, can be seen in Figure 2.4. The Raspberry Pi computer can be seen in the top left. To the right of that is the external storage in blue. The black box is the SDR and below that in grey is the GPS clock. To the right of the GPS clock is a resistance which is put between the clock and the SDR. A schematic diagram of the hardware and how it all connects can be seen in Figure 2.5.

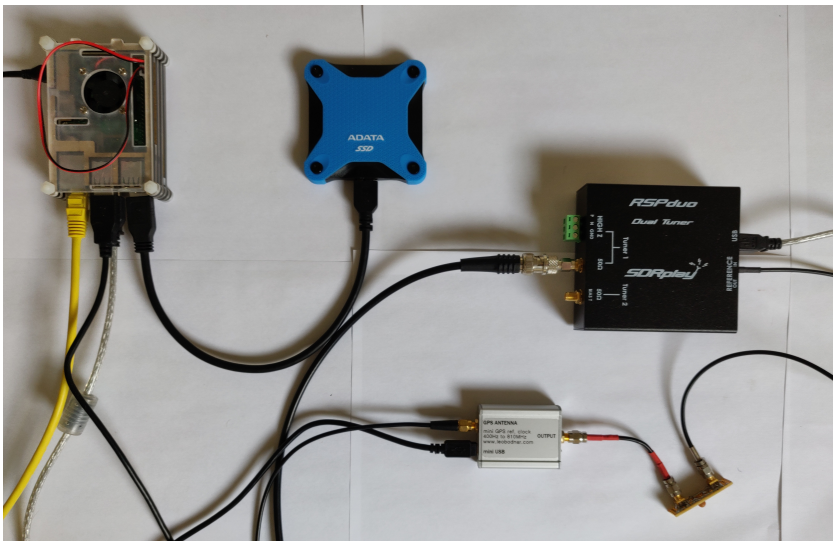


Figure 2.4: Photo of the DopTrackBox computer, external storage, SDR, and GPS clock.

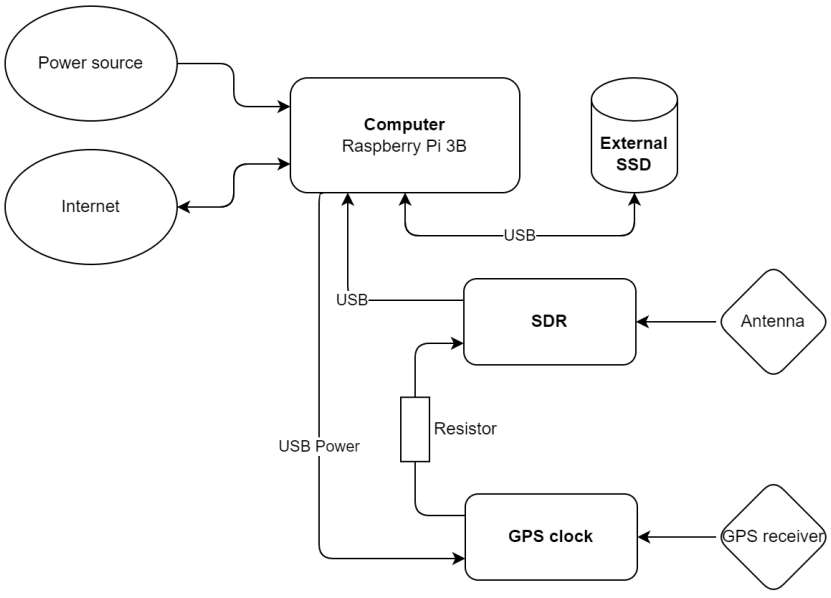


Figure 2.5: Schematic diagram of all DopTrackBox hardware components.

2.5.1. COMPUTER

The computer is the heart of the system. As was shown in Section 2.2, this is a Raspberry Pi 3B. This computer fulfils all the specific technical requirements. The Raspberry Pi has four USB 2.0 ports, an ethernet port and wifi, and an HDMI port [13]. As three of the USB ports will be used to connect the SDR, external storage, and GPS clock, the fourth one can be used for a keyboard. The Raspberry Pi used in DopTrackBox can be seen in Figure 2.6.

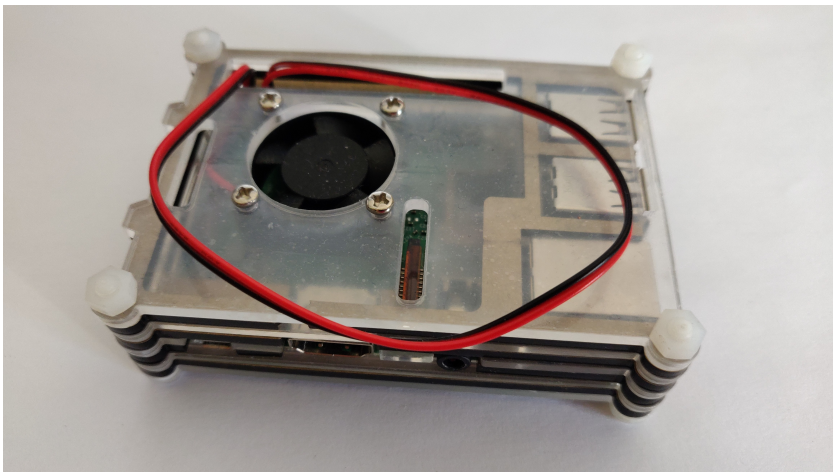
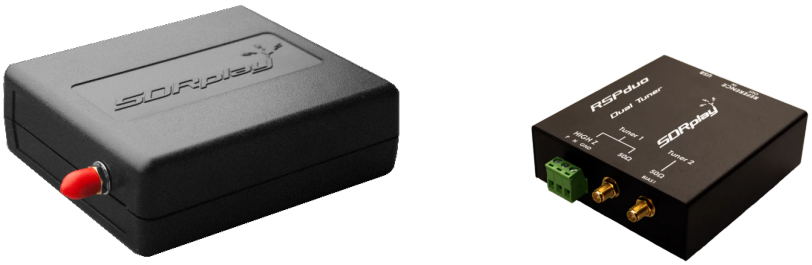


Figure 2.6: Photo of the Raspberry Pi 3B computer used in DopTrackBox.

During the verification phase it was found out that the Raspberry Pi could not handle the load put on the system by the recorder and thus a different computer had to be chosen. This is all explained in more detail in Chapter 5.

2.5.2. Software Defined Radio

Two different SDRs were provided for this thesis: the SDRplay RSP1A and the SDRplay RSPduo. The RSP1A is the more basic of the two and is used in the DTB Light concept from Table 2.3, whereas the RPSduo is more advanced and will be used for the DTB and DTB Pro concepts. Both are capable of covering the VHF and UHF frequency ranges and should therefore meet requirements DTB-TECH-2.03 and DTB-TECH-2.04 [14, 15]. They both have a USB type B port which connects to the computer via a USB type B to A cable, satisfying requirement DTB-TECH-2.02. As can be seen from Figure 2.7, the RSP1A has one connection for an antenna whereas the RSPduo has three of which two can be used simultaneously. Of these two only one will be used at a time in DopTrackBox' case. The RPS1A is lighter than the RSPduo, at 110 g compared to 315 g [14, 15]. However, this weight difference does not have a major impact on DopTrackBox' overall portability.



(a) RSP1A. (b) RSPduo.

Figure 2.7: Images of the RSP1A and the RPSduo. [16, 17]

2.5.3. GPS clock

The GPS clock will be used only in the DTB Pro concept together with the RSPduo. The GPS clock can be seen in Figure 2.8. The clock is connected to an included GPS antenna and power via a mini USB port. The output signal is passed through a resistance before it is send to the SDR. The resistance has been put there based on recommendations made during previous research by *E.J.O. Schrama* to regulate the output power.

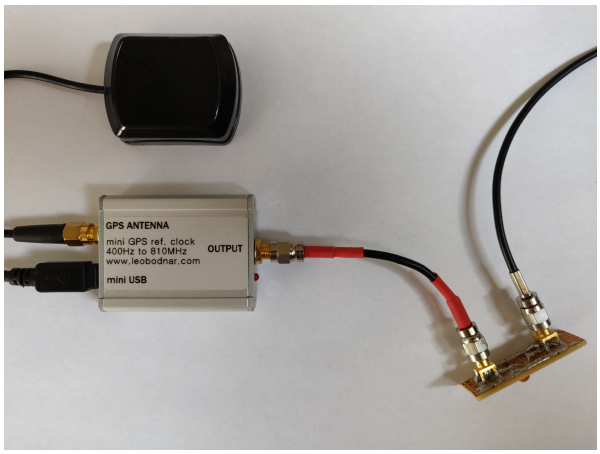


Figure 2.8: Picture of the GPS clock with the power cable, GPS antenna, and resistance.

The clock needs to be configured before it can be used properly. This has to be done using the special configuration software from Leo Bodnar on a Windows PC or MAC, since this software is not available on Linux. In the configuration software output frequency and output drive strength have to be changed to work properly with the RSPduo. The frequency must be set to 24 MHz to work with the RSPduo [15]. The ideal output device strength was found to be 16 mA, which resulted in the most stable recording behaviour. This is also consistent with the value presented in the manual for the GPS clock with the RSPduo provided by *SDR-Kits* and research done by *W.D. Reeve* [18, 19]. To ensure stable operation the GPS clock cannot be connected or disconnected from the RSPduo whilst in operation. Therefore the RSPduo has to be disconnected from the computer before connecting or disconnecting the GPS clock.

2.5.4. ANTENNA

The antenna used for DopTrackBox' portable operations can be seen in Figure 2.9. The outer prongs are used to receive VHF signals whereas the inner prongs are used for UHF. There are extenders that should be attached to the outer prongs when using the antenna to receive VHF signals. There are two connections on the bottom of the antenna, one for VHF and one for UHF. These should be connected to the SDR. Since a Yagi antenna is directional, it has to be pointed towards the satellite whilst tracking.

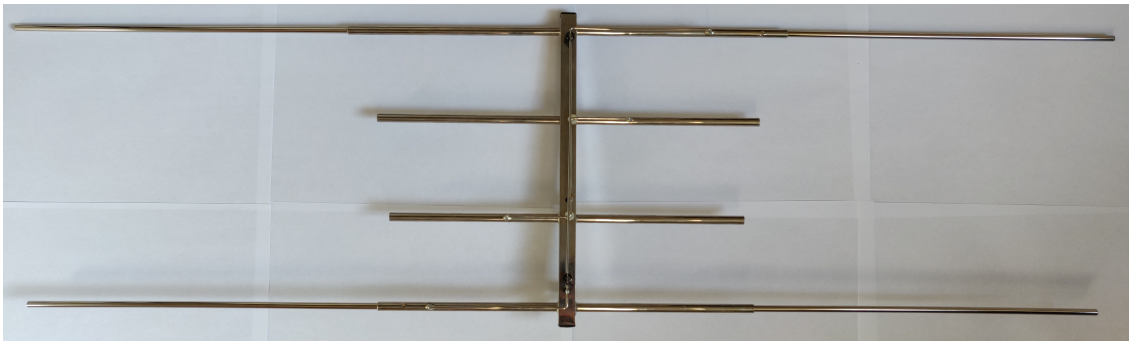


Figure 2.9: Photo of the Yagi antenna used with DopTrackBox with the fully extended VHF prongs.

Due to the gain pattern of the antenna, it does not have to be pointed to the exact location of the satellite. The antenna can be pointed in the general direction of the satellite for it to receive the signal. The specific antenna used is a two-element Yagi antenna. The antenna does not have a reflector, thus a different model has been used to provide some insight into the antenna's gain pattern. This model is represented by a linear array of two elements, with an element length of 1.05 metres and an element spacing of 0.26 metres. The gain pattern has been calculated at the tuning frequency used for the NAYIF satellite; 145.975 MHz. The antenna gain can be seen in Figure 2.10. The radial direction represents the directivity or antenna gain, and the angular direction represents the azimuth.

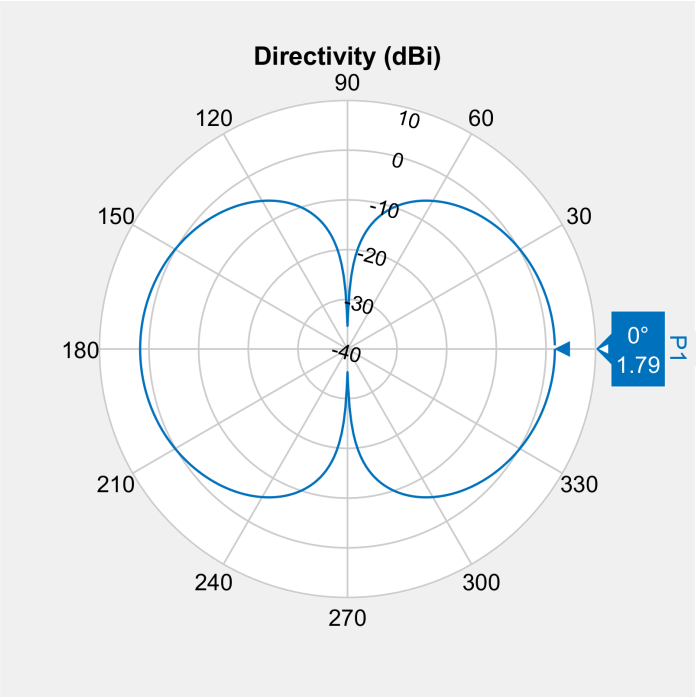


Figure 2.10: DopTrackBox Yagi antenna gain pattern.

Due to the lack of a reflector, the gain pattern is symmetrical. The main lobe and back lobe are therefore equally large. Whether or not the actual antenna has this property has not been tested due to the way the antenna has to be held and is connected to the system. As expected, the gain is the largest at 0° when directly pointing towards the satellite. The antenna's gain at that position is 1.79 dBi. At a beamwidth of 60° the antenna's gain becomes 0 dBi. At this point the received signal's power is no longer increased by the antenna. The half power, or 3dB beamwidth of the antenna model is 80°. The gain at that specific angle is -1.26 dBi. If the antenna is pointed within 30° of where the satellite actually is, there should be no loss in received signal power. This is something which has to be taken into account during data acquisition.

2.5.5. External Storage

The external storage is an item that was not initially provided at the start of the project. Therefore a suitable candidate had to be found that would meet all the requirements set in the previous section. The relevant requirements are DTB-OPS-2.06, DTB-OPS-2.07, DTB-TECH-5.01, DTB-TECH-5.02, and DTB-TECH-5.03. DTB-TECH-5.01 states that the external storage shall not have any moving parts, so this excludes hard drives as options for the external storage. The most logical option is an SSD. Two options were investigated: COTS external SSDs and internal SSDs within an enclosure.

There are 2 main different SSD form factors; the M.2 SSD and the 2.5" SATA SSD. The M.2 SSD is directly placed on the motherboard of a computer and is therefore smaller than the 2.5" SATA SSD. M.2 SSDs can use either the SATA or NVMe protocol, with the latter one allowing for higher data transfer speeds. However, using one or the other does not matter since the bottleneck will be the Raspberry Pi's USB 2.0 data transmission speeds [13, 20]. To maximise DopTrackBox' battery life a power efficient SSD has to be chosen. By comparing a few SSDs it was observed that the 2.5" SATA SSDs had a slightly lower power consumption.

The final step in choosing a suitable candidate is comparing prices. Whilst this is not an official requirement, the aim is to keep the price low to make DopTrackBox as available as possible. The price comparison website *Tweakers.net* [21] was used to compare the prices of 2.5" internal SATA SSDs, M.2 SSDs, SSD enclosures, and COTS external SSDs. This research pointed out that several 2.5" internal SATA SSDs of 256 GB capacity are available from €30 to €40. For the M.2 SSDs not many 256 GB variants were available, so the next biggest size would be 512 GB. These cost around €40 to €50. Both these internal SSDs need an enclosure, available from €15 to €30 where enclosures for the 2.5" SSDs are cheaper than those for the M.2 SSDs. Finally, external COTS SSDs are also not widely available in 256 GB size but the 512 GB ones are available from €70.

The choice was made to select an external COTS SSD, since using an internal one does not provide a huge cost benefit when looking at 512 GB SSDs. More advantages of the external COTS SSD are that it can be smaller than the enclosure for the 2.5" SSD and that there are specifically designed rugged variants which are ideal for the portable nature of DopTrackBox. The specific SSD that was chosen is the ADATA SD600Q, which has a capacity of 512 GB of which 480 GB is usable. It has a power consumption of 4.5 Wh [22].

2.6. DopTrackBox Software

For DopTrackBox to record satellite data a set of programs is used based on the DopTrack software. The largest change between the original DopTrack programs and the DopTrackBox programs is the switch from Python 2 to Python 3. The main software can be divided into three distinct parts: the scheduler, the recorder, and post processing. Each of these will be walked through in its own section from start to finish whilst explaining the functioning of each program. A complete overview of DopTrackBox' code that has been changed from the original DopTrack code, or any new code packages, can be found in Appendix B. All code has been backed-up to the DopTrack Github.

2.6.1. Scheduler

The DopTrackBox Scheduler's objective is to check when satellites that DopTrackBox should track are in view and schedule a recording. The scheduler consists of five different programs. An overview of the scheduler's flow can be seen in Figure 2.11.

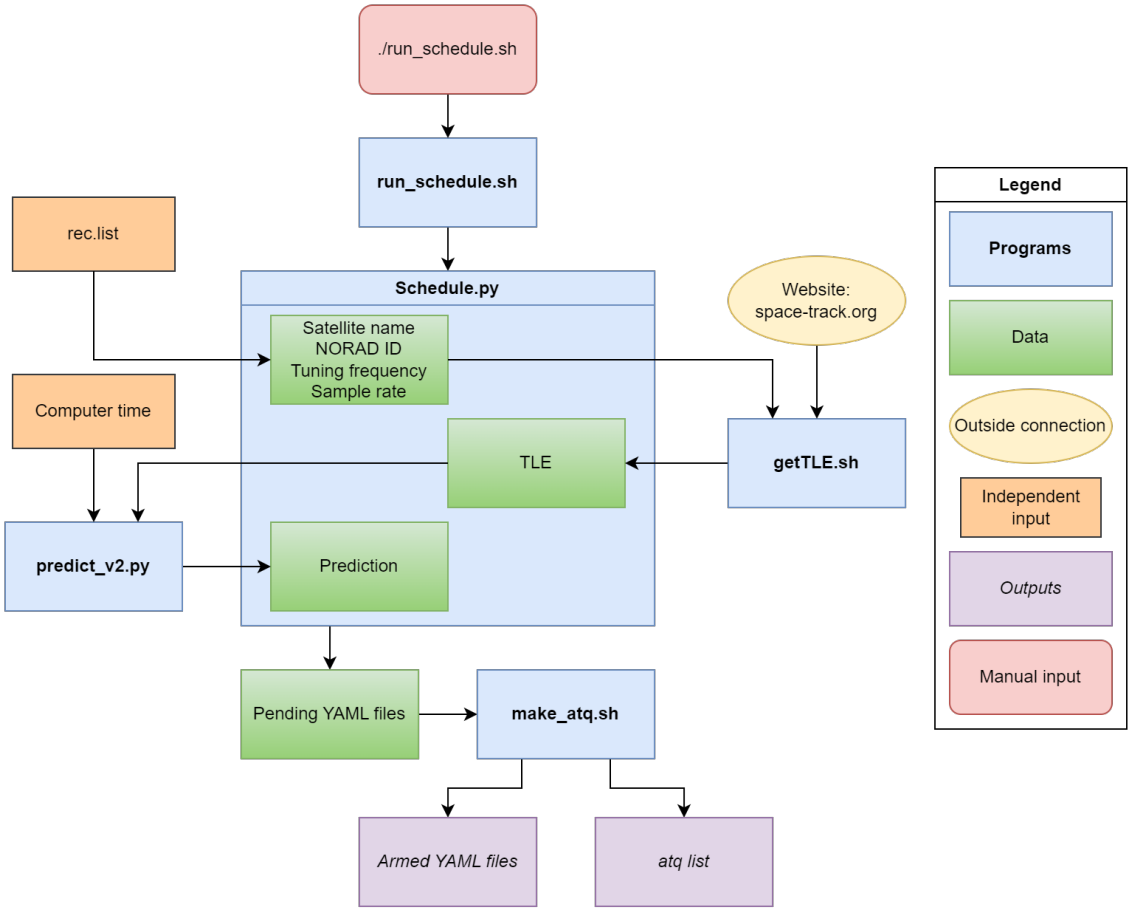


Figure 2.11: DopTrackBox scheduler software flow schematic.

All programs can be seen in a blue box with bold text. There is one prerequisite for the scheduler to work, which is the rec.list file. This is a simple text file containing the satellite names, NORAD IDs, tuning frequencies, and sample rates of every satellite DopTrackBox is tracking. The NORAD ID is a unique satellite identification number, the tuning frequency tells the recorder at which frequency it must record the satellite pass, and the sample rate tells the recorder at how many samples per second it must record. All these values are eventually added to the yaml file at the end of the scheduler. The rec.list used for DopTrackBox can be

seen in Table 2.6. These tuning frequencies are based on the downlink or beacon frequencies of the respective satellite. The sample rate of 250 kHz has been adopted from DopTrack. A variable that is present in the table but not shown as such is the satellite recording priority. The highest priority is given to the satellite on the top of the list, and the lowest to the satellite on the bottom. Another difference is the header which is not present in the rec.list, but has been added here for clarity.

Table 2.6: rec.list details used for DopTrackBox.

Satellite Name	NORAD ID	Tuning Frequency [MHz]	Sample Rate [kHz]
NAYIF	42017	145.975	250
Delfi-C3	32789	145.870	250
Delfi-N3XT	39428	145.870	250
FUNCUBE_1	39444	145.935	250

The outputs of the scheduler are the armed yaml files and the atq list. The yaml files contain information of the satellite pass and recording, and are used in the recorder and post processing. An example yaml file can be seen in Figure 2.12. Most of the details within a yaml file are added during the scheduling phase, and are thus explained below where the flow of the scheduler is explained. The atq list is the automatic trigger to start recording when a satellite pass is happening.

```
Sat:
  Predict:
    EAzimuth: 184
    Elevation: 56
    Length of pass: 780
    SAzimuth: 16
    time used UTC: 20220830085536
    timezone used: 2.0
    used TLE line1: 1 42017U 17008BX  22240.17851576  .00011593  00000-0  32717-3
      0  9992
    used TLE line2: 2 42017  97.2634 305.3280 0004183   9.0198 351.1121 15.36532478308088
  Record:
    Start of recording: 202208301152
    num_sample: 195000000
    sample_rate: 250000
    time pps: ''
    time1 UTC: 2022-08-30 09:52:00.140771
    time2 UTC: 2022-08-30 10:05:03.322436
    time3 LT: 2022-08-30 11:52:00.140748
  State:
    Antenna: 1
    NORADID: '42017'
    Name: NAYIF
    Priority: 1
    Tuning Frequency: 145975000
  Station:
    Height: 0.0
    Lat: 51.99899638888889
    Lon: 4.3734883333333325
    Name: DopTrack
```

Figure 2.12: Example of a yaml file from DopTrackBox.

The scheduler is started by typing in the command *"./run_schedule.sh"*, which calls the run_schedule.sh program. This is a very basic program that makes the directory where the Schedule.py program is located active, and calls that program in Python. Schedule.py is the main program in the scheduler. It starts by defining the directories on the computer where the different data and programs are stored. It then grabs the rec.list and extracts the information from it, including satellite priority. This information is filled in into an empty yaml file, which will be added to during the scheduling process. Using the tuning frequency the program determines which antenna to use. This is not as relevant for DopTrackBox as DopTrackBox only has a single antenna. This is a leftover from the original DopTrack code, but has not been removed as it can be used to trace back if a VHF or UHF antenna has to be used in the yaml file.

Schedule.py then calls for getTLE.sh, using the NORAD ID. This step requires an active internet connection, as getTLE.sh goes to *space-track.org* and uses the provided NORAD ID to get the two-line element set (TLE) of the satellite. The TLE contains the orbit data that is needed for the orbit and pass prediction. The TLE is passed back to Schedule.py, which passes it through to predict_v2.py. This is a program taken from DopTrack with some minor changes to make it work on DopTrackBox hardware, mainly translating the old Python 2

version to Python 3. Due to some inconsistencies the "_v2" suffix was added to distinguish between the original program from DopTrack and the new one for DopTrackBox. The actual prediction loop has not been functionally changed.

Predict_v2.py checks when the satellite is in view of the ground station. It uses the current computer time as a starting point for the calculations together with the ground station's coordinates. In this version of the program the ground station coordinates are hard coded, so when changing the ground station location the coordinates need to be manually updated in the program. The ground station name, latitude, longitude, and altitude are added to the yaml file. After this step, the predictor reads the TLE and adds the contents to the yaml file. With the TLE it starts the prediction loop, where it checks for passes over the ground station in a five day period. This is done by constructing the orbit and checking if the satellite is in view of the ground station. The time when a satellite is in view is stored and passed back to Schedule.py.

When the predicted passes are read by Schedule.py, the first thing that it checks is whether the predicted pass meets the set elevation threshold. The minimum elevation is determined by the conditions at the ground station. The hard coded minimum is 5° for DopTrack or DopTrackBox recordings at the DopTrack ground station. If a different ground station needs a higher elevation minimum, it has to be changed within Schedule.py manually. If the maximum elevation of the predicted pass does not exceed the threshold, the prediction is discarded. Otherwise the date, start time, end time, start azimuth, and end azimuth are extracted. These are used to determine the length of the recording and thus the number of samples in the recording. Finally, both azimuth values, the length of the pass, the maximum elevation, the starting time of the recording, and the number of samples are added to the yaml file.

The yaml file is put in the pending folder and make_atq.sh is called. This program grabs all the pending and currently armed yaml files, and merges them into a single list. It checks if the time of recording of any of the old armed yaml files is after the time of this update. If that is the case, the yaml file is removed. The program then checks for duplicate yaml files of the same pass. If these exist, the oldest version is removed as a new version with a newer TLE can contain updated and more up to date information. The next step is checking for overlapping recordings, as DopTrackBox can only record one satellite pass at a time. This is done by checking the starting time of the recording and the length of the pass. If any overlap is detected, the yaml file of the satellite with the lowest priority from the rec.list is removed. The remaining yaml files are armed for recording. Finally, the program removes the old pending jobs from the atq list and adds the times of recording from the armed yaml files to the atq list.

2.6.2. RECORDER

The DopTrackBox Recorder records the satellite data at the times set in the atq list by the scheduler. It consists of a single Python program and the radio, Rx_tools. Its software flow schematic is visible in Figure 2.13. For the recorder to function, the armed yaml file should be present. The recording automatically begins at the scheduled time via the atq, which calls the Record_DTB.py file with the corresponding yaml file. It is possible to manually start a recording, but this is not the recommended way of recording a satellite pass. To start a manual recording Record_DTB.py must be started with a corresponding yaml file as input parameter. These yaml files are automatically made by the scheduler, so therefore the automatic recording process is more streamlined.

The results from the recorder are a complex binary .32fc satellite data file and the yaml file that corresponds with the recording. This is the same file as the armed yaml file that was used to start the recording with some additional timestamps added. When a recording has been completed, the armed yaml file is removed and only the recorded yaml file remains.

When Record_DTB.py is started it reads the details in the yaml file. It checks the time at the start of the recording and stores them both in UTC and local time. From the yaml file the tuning frequency, sample rate, and number of samples are extracted and used as input parameters for the radio. The program starts the radio, which is rx_tools, with the input parameters from the yaml file. The recording ends automatically when rx_tools has recorded the predetermined number of samples. At this point Record_DTB.py checks for the time at the end of the recording. The armed yaml file is updated with the three recorded times: time1 is the UTC time at the start of the recording, time2 is the UTC time at the end of the recording, and time3 is the local time at the start of the recording.

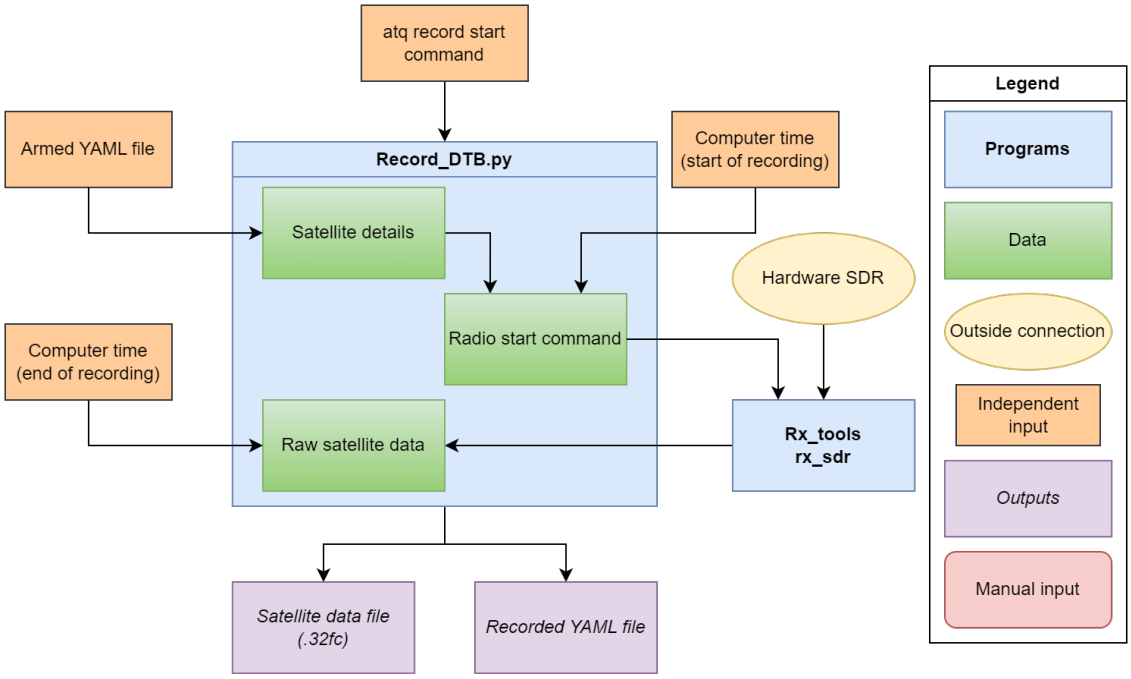


Figure 2.13: DopTrackBox recorder software flow schematic.

2.6.3. Processing

The DopTrackBox Processing software uses the recorded satellite data and extracts useful details from the raw satellite data file. For DopTrackBox the most interesting data is the SNR and Doppler shift data, but a lot more can be extracted from the raw I/Q data. The software flow can be divided into two distinct parts, as the SNR and Doppler shift data extraction are handled by two distinct programs.

Signal, noise, and spectrogram

The flow schematic for the SNR, spectrogram, noise, and signal data can be seen in Figure 2.14.

The curvetool.py program has been newly created for DopTrackBox, therefore it is not as tightly integrated with the previous software packages. It is started manually by typing the command *"Python3 curvetool.py"* in the terminal. Several parameters are hard coded in the program that need to be changed depending on the satellite pass. The most important of those are the satellite file name, without the extension, the location where the file is stored, and the tuning frequency used during the recording. These steps could be automated in the future to more tightly integrate the post processing with the recording. This has not been done during this thesis as the program was still in active development, and changing this would take more time than changing a few parameters for the few satellite data sets that were processed.

The curvetool.py program uses several Python programs that are part of the DopTrack Processing Package (DTPP). Some of these programs were slightly changed to add some additional functionality, but most of them are left untouched. The satellite data is loaded in to curvetool.py by the recording.py program. A function to create a spectrogram is called from the processing.py program, which in turn calls the signals.py program to process the raw I/Q data. A Fast Fourier Transform is performed on the data, and the signal and noise are separated from each other. These are used to calculate the noise floor and signal to noise ratio (SNR). This is done for every row, and when they are all put together they form the spectrogram. The functionality of calculating the signal and noise for each row was not present in the original code, so this has been added in the DopTrackBox version.

This data is passed back to curvetool.py. Here the spectrogram data is plotted as a spectrogram and the processed data is saved as a .dat file for further analysis. The noise floor data is saved and plotted to show how the noise floor behaves throughout the recording. There are three sets of signal data passed back to curvetool.py. Those are the mean, median, and maximum signal per second. These three are all used to create their respective SNR plots and a combined SNR data file. Plot parameters can be adjusted within the code if necessary. The colour bar for the spectrogram shows values from -5 dB to 20 dB. These were selected as they resulted in spectrograms where the signal is clearly visible without losing too much detail. A setting

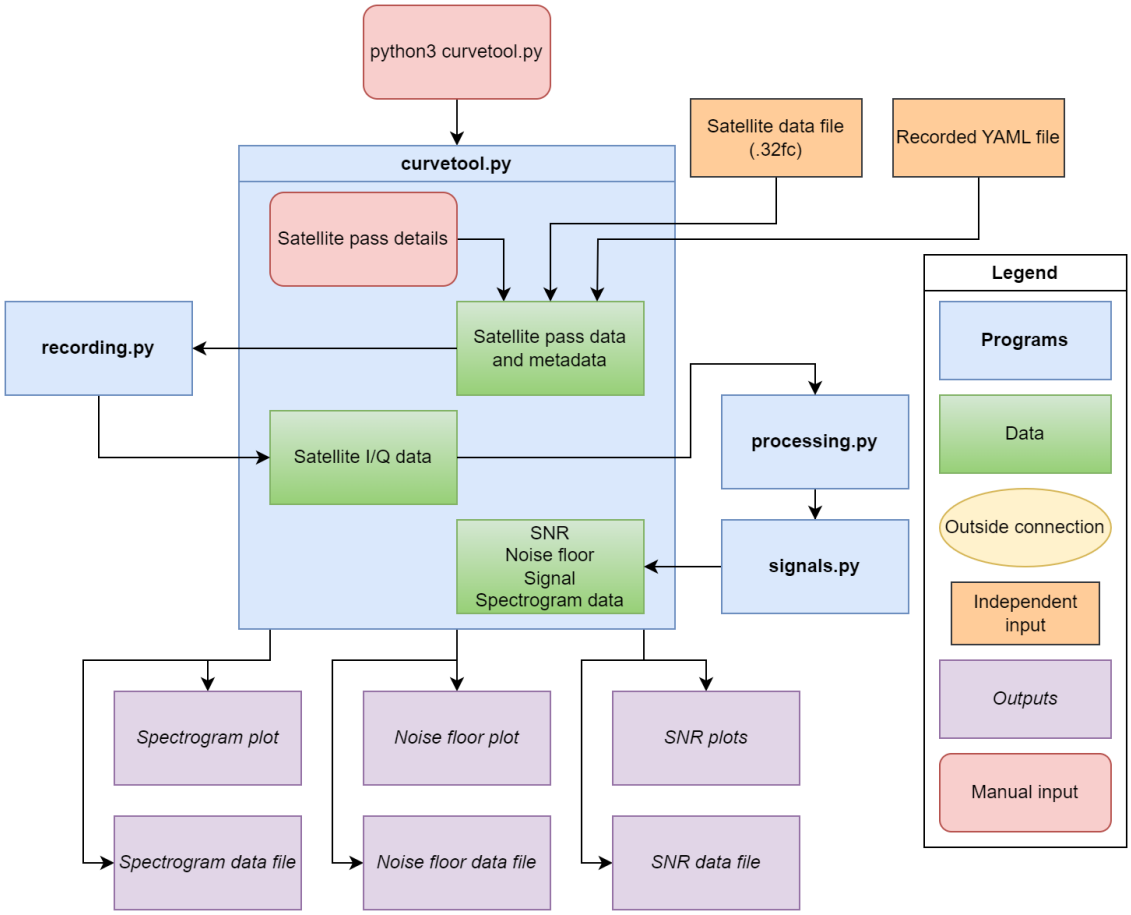


Figure 2.14: DopTrackBox processor software flow schematic for SNR data and spectrograms.

to automatically set the colour bar ranges was used at first, where it selected the minimum and maximum detected values. However, the maximum value is often only present at one spot whilst the rest of the signal is significantly weaker. This resulted in spectrograms where the signal could not easily be distinguished. The program checks if a data file already exists, so that is does not overwrite it when rerunning the same data set when some graph parameters have been changed. This functionality has been but in place since writing the data files takes quite some time, which would slow down the debugging of the program.

DOPPLER SHIFT DATA

The Doppler shift data processing solely relies on the DopTrack Processing Package. The newest version of this package at the time of writing this report is called DopTrack DopTools. The version used with DopTrackBox was obtained on 6 September 2022 and was the most recent version at the time. This package is downloaded from the DopTrack Github and installed on DopTrackBox. This makes it harder to describe step by step what the program does, as the program operates as a unit opposed to a single Python file that uses functions or calls to other files. An overview of the basic software flow can be seen in Figure 2.15.

The DTPP needs to be run and installed in a virtual environment, so when using DTPP this virtual environment must be activated again. There are some configuration files for DTPP stored in the home directory of DopTrackBox. Amongst these are the config.yml and the agenda.csv files. These are used to tell the DTPP which satellite data it needs to process and where it can find this data. The DTPP functions as any other installed program on Linux that operates through the terminal. It can be stared by typing *"doptrack processing process"*. It will then automatically go through every data file in the directory that has been specified within the config.yml file, as long as the satellite has been added to the agenda.csv file.

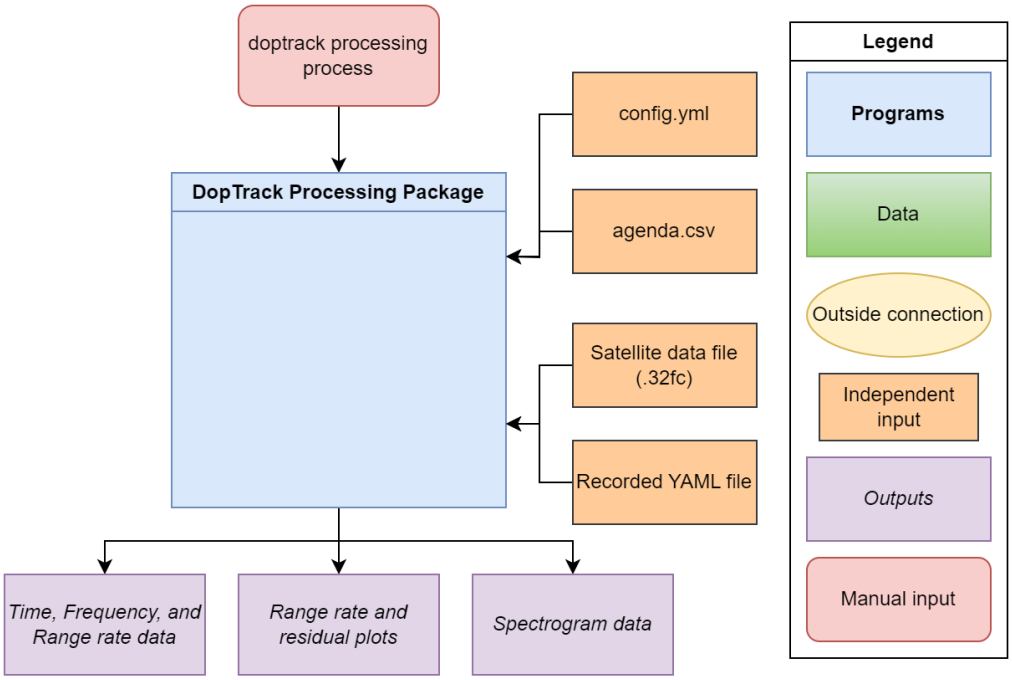


Figure 2.15: DopTrackBox processor software flow schematic for Doppler shift data.

2.7. Concept Iterations

During the initial set-up phase of the hardware and software some iterations were made to improve DopTrackBox. This section dives into these iterations. The initial concept from Table 2.2 based on previous research was changed early on. The preliminary study conducted earlier on did not use the same foundational software code as DopTrack, making the two systems less compatible with each other. It was also based on the GQRX radio software which uses a graphical user interface, making it harder to control the radio via the command line and thus remotely operate DopTrackBox.

2.7.1. DopTrack Parity

To allow for easier remote access and more parity between DopTrackBox and DopTrack the decision was made to change the operating system from Raspberry Pi OS to Ubuntu. Due to the limited storage size of the SD card a light operating system had to be chosen. Because DopTrack works exclusively with a terminal, the same decision was made for DopTrackBox and the most recent Ubuntu Server operating system was chosen. At the time this was Ubuntu Server 21.10. Another advantage of this operating system over Raspberry Pi OS is that Ubuntu Server is a 64 bit operating system instead of 32 bit. Since the change was made from an operating system with a graphical user interface to a command line interface, different radio software had to be chosen due to the difficulty of using GQRX through the terminal.

During this reconfiguration phase it became clear that even with the external storage provided by the SSD, the 8 GB of internal storage would be insufficient for the programs that needed to be installed on DopTrackBox. Therefore the 8 GB SD card was swapped out for a 64 GB one. This new SD card has a 10 MB/s sequential write speed, which is the same as the old one. This meant that an operating system with a graphical user interface could be chosen again instead of strictly command line only OS. Since the set-up of all programs would be the same for a graphical OS and a command line OS, the development on the Ubuntu Server version continued whilst noting all the steps taken. If the switch had to be made between operating systems it would be as easy as following these steps.

Another change at this time was the adoption of the DopTrack code instead of using the proprietary DopTrackBox code. Since the DopTrack code was still written in Python 2 it had to be changed for DopTrackBox. At first it was tried to install a Python environment based on Python 2, but due to the difficulty of getting it to work properly that cause was abandoned. As a result, all the DopTrack Python 2 code was translated to Python 3 which also increases ease of use for other systems in the future. Python 3 is also the currently supported version of Python which future proofs the system over sticking with Python 2.

2.7.2. RECORDER SOFTWARE CHANGES

DopTrack uses recording software specific to its hardware. Therefore DopTrackBox cannot use the same recording software. Several recording software programs were tried out to see which one would work best with DopTrackBox. To find suitable software compatible with the SDR and OS, the SDR's documentation was consulted. For this the RSP1A was used as it functions as a simpler SDR than the RSPduo, but will still be used during the actual testing. An overview of the current hardware and software setup can be seen in Table 2.7.

Table 2.7: DopTrackBox Hardware and Software during recorder software determination.

Hardware					***Software***	
Computer	**SDR**	**GPS Clock**	**Antenna**	**Storage**	**OS**	**Recorder**
Raspberry Pi 3B	RSP1A	None	Yagi	64 GB SD card	Ubuntu Server 21.10	SoapySDR

To properly operate the SDR through the terminal several programs were needed to ensure compatibility. SoapySDR is the most basic of these and was the first recorder program used. This is an API that allows other programs to interact directly with the SDR hardware. For it to work with the RSP1A and RSPduo a compatibility plugin called SoapySDRplay had to be installed. Now it is possible to interact with the SDR directly through the terminal, but SoapySDR cannot record and save any data in a way that is compatible with the DopTrackBox workflow. The radio recorder software recommended for a Raspberry Pi with Ubuntu is CubicSDR. CubicSDR is also a graphical recorder tool, so a compatibility software had to be installed to call CubicSDR from the terminal. To interact with CubicSDR without a graphical user interface, GNUradio was used. However, during the installation of GNUradio there seemed to be an incompatibility with the software which resulted in several errors. Because of this, GNUradio could not run on DopTrackBox and an alternative had to be found for both GNUradio and CubicSDR. This lead to rx_tools, which contains several software tools for receiving data from SDRs. Rx_tools is based on librtlsdr for RTL-SDR devices, but it uses SoapySDR to support other SDRs as well [23]. Within rx_tools is rx_sdr which can be used to get the I/Q data that is needed for DopTrackBox.

2.7.3. DOPTRACKBOX VIRTUAL MACHINE

During verification and validation it became clear that the Raspberry Pi was not capable of handling the load put on the system by the incoming data stream and the processing of that data. The verification process that led to this conclusion will be discussed in more detail in Section 5.1. Therefore the DopTrackBox software was installed on a virtual machine running Ubuntu. Since the host machine has a build in screen, keyboard, and battery an operating system with a graphical user interface was chosen instead of sticking with the terminal only nature of Ubuntu Server. This change increased DopTrackBox' flexibility and ease of use. The DopTrackBox Virtual Machine (DTB-VM) is the final version of DopTrackBox used for the research in this thesis. The change from a Raspberry Pi 3B to a virtual machine meant that other changes in the hardware configuration had to be made as well due to the hardware limits of the host machine. The biggest limiting factor is the host machine's single USB port, albeit a higher speed USB port, compared to the four on the Raspberry Pi. To overcome this, recordings are stored locally first before being transferred to the external storage and the GPS clock is connected to a separate battery. An overview of this new hardware configuration can be seen in Figure 2.16.

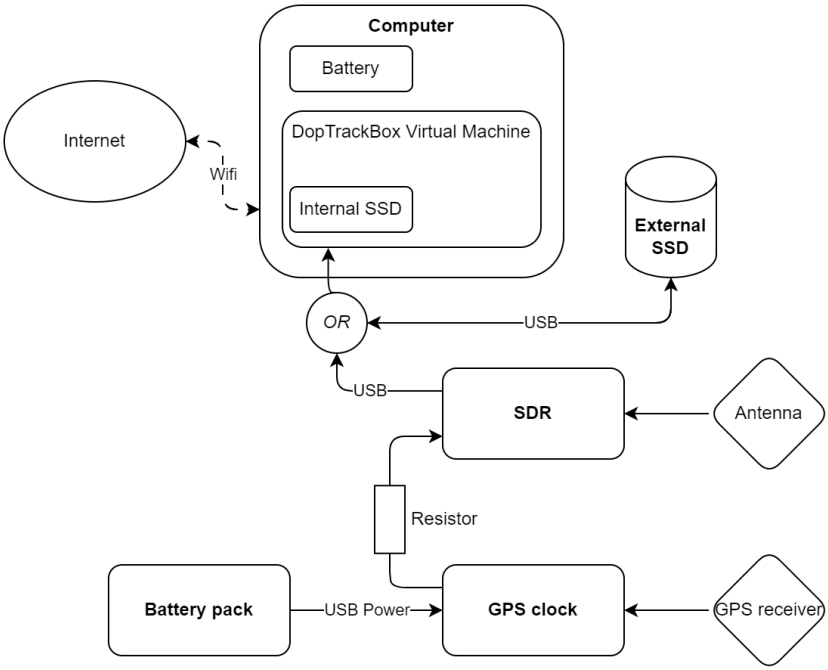


Figure 2.16: Schematic diagram DopTrackBox Virtual Machine hardware components.

3 | Installation Manual

One of the main intentions behind DopTrackBox is that it can easily be used by others. This is achieved by using commercial off the shelf (COTS) components and by writing software that is available on GitHub. A large part of the thesis was dedicated to getting the hardware and software working. This process is documented in this chapter so that it can be reproduced easily. The initial hardware set-up and installation process is explained in Section 3.1. This is the hardware that was discussed in Sections 2.5 and 2.7. The software installation procedure is discussed in Section 3.2. This is mainly the underlying software enabling DopTrackBox to perform its intended functions. With both the hardware and software up and running, Section 3.3 describes how to set up DopTrackBox for recording and operational purposes.

DopTrack is in active development, so the software that is available on GitHub can change over time. The software installation flow in this chapter was written for the then up to date version of the software. The initial software used for DopTrackBox was cloned from the DopTrack GitHub in March 2022. After the switch to the DTB-VM system, the software was cloned on 8 July 2022 with the most recent updates at the time. The post processing DopTrack software was cloned on 6 September 2022. When applicable, links to the download pages are provided in the footnotes.

3.1. Hardware Set-up

This section describes how to use the DopTrackBox hardware. It is subdivided into two sections. First, Section 3.1.1 describes the initial set-up procedure for the Raspberry Pi. Even though the Raspberry Pi was not used for the DopTrackBox experiments, its set-up process is included here in case a more powerful Raspberry Pi or equivalent system could be used in the future. After that, Section 3.1.2 elaborates upon the initial set-up procedure for the DopTrackBox Virtual Machine.

3.1.1. Initial Raspberry Pi Set-up

The first step in getting the Raspberry Pi up and running is installing the operating system. For this a second computer with an active internet connection is needed to install the operating system on the SD card. This computer also needs to have an SD or microSD card slot, or an extension dongle that provides these. For the next steps in the set-up process a keyboard, ethernet cable, and monitor are needed. The installation steps are the following:

Raspberry Pi initial installation

1. Download and install the Raspberry Pi Imager[1].
2. Insert the SD card into the PC.
3. Open the Raspberry Pi Imager and select the SD card.
4. Select the desired operating system: Ubuntu Server 21.10 LTS, or another 64 bit equivalent compatible with the hardware.
    - *Optional: In the advanced options configure the wifi network and SSH.*
5. Install the operating system.
6. Remove the SD card from the PC and insert it in the Raspberry Pi.
7. Connect a keyboard, ethernet cable, and monitor to the Raspberry Pi.
8. Connect the power cable to the Raspberry Pi to turn it on.
9. Log in to the Raspberry Pi with the standard username and password: "ubuntu".
10. Change the password.
11. Fetch and install the latest updates:
    i. Type "Sudo apt update".
    ii. Type "Sudo apt upgrade".
    iii. Confirm the update installation when asked to do so.

[1] Available on: https://www.raspberrypi.com/software/ [Accessed 13 July 2023].

The Raspberry Pi should be set up now to install the other software. To access the Raspberry Pi remotely, do the following steps. This is recommended if using a secondary keyboard or monitor is difficult. Since the development was largely done on a Windows computer, the steps below are based on the workflow on Windows 10.

REMOTE ACCESS

1. Check the IP address of the Raspberry Pi.
2. Open a terminal on the PC you want to connect to the Raspberry Pi, Windows PowerShell is recommended.
3. Type: "ssh ubuntu@<*IP address*>".
4. Log in with the selected password.
5. When asked to confirm the connection, type "yes".

Of the other hardware components, only the GPS clock needs to be configured before it can be used. The procedure for this was described in Section 2.5.3.

3.1.2. INITIAL DTB-VM SET-UP

The initial hardware set-up of the DopTrackBox Virtual Machine entails the installation of the VM software and the operating system. The only requirement for this is the host machine with an active internet connection.

1. Download and install VirtualBox[2]. DTB-VM uses version 6.0.24 for the experiments, but a newer version should function similarly.
2. Download and install the VirtualBox Extension Pack compatible with the chosen version of VirtualBox.
3. Download the Ubuntu operating system image[3]. DTB-VM uses Ubuntu 22.04 LTS.
4. Open VirtualBox and click on 'New' to make a new VM.
5. Name the VM "DopTrackBox" and select the operating system 'Linux' and version 'Ubuntu (64-bit)'.
6. Change the available RAM to 3072 MB.
7. Create a new virtual hard drive of 32 GB, using the setting VDI.
8. Start the virtual machine and select the downloaded Ubuntu operating image for installation.
9. Select 'try or install Ubuntu'.
10. Select 'Install Ubuntu' and choose the language and keyboard options of choice.
11. Select 'Minimal installation'.
12. Select 'Erase disk and install Ubuntu'. The disk created for the VM should be empty so no data should be lost.
13. Install Ubuntu.
14. Select the device's location, create login details, and use the appearance settings of choice.
15. Install the latest updates using the software updater or the terminal. For the terminal do the following:
    i. Type "Sudo apt update".
    ii. Type "Sudo apt upgrade".
    iii. Confirm the update installation when asked to do so.
16. After installing the updates shut down the VM.
17. In VirtualBox select the VM and go to 'Settings'.
18. Go to 'USB' and enable the USB 3.0 (xHCI) Controller.
19. Go to 'System' and change the number of Processors to 2.
20. Change the scaling in 'Monitor' settings if needed in VirtualBox combined with the display settings within Ubuntu to better fit the host monitor.

After these steps the virtual machine should be set up for further installation of the DopTrackBox software. The VirtualBox Extension Pack is needed to enable USB passthrough in the used version of VirtualBox. This allows the VM to see and interact with USB devices connected to the host, which is needed for the external SSD and SDR.

As was discussed in Section 3.1.1, the GPS clock needs to be configured as well before it can be used.

[2]Available on: https://www.virtualbox.org/wiki/Downloads [Accessed 13 July 2023].
[3]Available on: https://ubuntu.com/download/desktop [Accessed 13 July 2023].

3.2. SOFTWARE SET-UP

There are various software programs required for the functioning of DopTrackBox. This section describes how to install them for DopTrackBox. As DopTrackBox runs Ubuntu, users familiar with the operating system will find it easy to install all the required programs. Nevertheless, a full step by step guide is provided here. The focus in this section is mainly on the DTB-VM system, as that was the main system used for the remainder of the thesis. For clarity, every program is described in its own subsection.

3.2.1. PYTHON 3

Python is used for many of the calculations and other programs of DopTrackBox. Version 3.9 or newer is required for compatibility with other DopTrackBox programs. From the terminal, type the following:

1. Type: “sudo apt install python3-pip”.
2. Confirm the installation when asked to do so.
3. The installation can be verified by typing “python3 –version" and “pip list”.
4. Install the required packages by typing “sudo pip install <package name>”.
   Required packages are:
   i. numpy
   ii. matplotlib
   iii. pytz
   iv. requests
   v. tzlocal
   vi. sgp4
   vii. psutil
   viii. cmcrameri

3.2.2. SOAPYSDR

SoapySDR is an important API allowing other programs to interact with the SDR hardware. It consists of the main program and several packages for it to work properly with the SDRplay RSP1A and RSPduo.

1. Get dependencies: “sudo apt-get install cmake g++ libpython3-dev python3-numpy swig”.
2. Get SoapySDR[4] by typing the following commands in the terminal:
   i. “git clone https://github.com/pothosware/SoapySDR.git”
   ii. “cd SoapySDR”
   iii. “mkdir build”
   iv. “cd build”
   v. “cmake ..”
   vi. “make -j4”
   vii. “sudo make install”
   viii. “sudo ldconfig”
   ix. “SoapySDRUtil –info”
3. Go back to the main installation folder.
4. Get SDRPlay driver binaries by typing the following commands in the terminal:
   i. “mkdir SDRPlayAPI”
   ii. “cd SDRPlayAPI”
   iii. “wget https://www.sdrplay.com/software/SDRplay_RSP_API-Linux-3.07.1.run”
   iv. “chmod 755 ./SDRplay_RSP_API-Linux-3.07.1.run”
   v. “./SDRplay_RSP_API-Linux-3.07.1.run”
   vi. “sudo reboot”

If the ’wget’ option does not work, the file can be manually downloaded from the SDRplay website[5]. Here the RSPduo and Linux/x86 Ubuntu should be chosen. Then under ’API’, the API 3.07 should be downloaded and manually moved to the correct folder. After the system has been rebooted, continue with the following steps:

[4] Available on: https://github.com/pothosware/SoapySDR [Accessed 13 July 2023].
[5] Available on: https://www.sdrplay.com/downloads/ [Accessed 13 July 2023].

5. Go back to the main installation folder.
6. Get SoapySDRPlay plugin[6] by typing the following commands in the terminal:
   i. “git clone https://github.com/pothosware/SoapySDRPlay.git”
   ii. “cd SoapySDRPlay”
   iii. “mkdir build”
   iv. “cd build”
   v. “cmake ..”
   vi. “make”
   vii. “sudo make install”

Check if the installation went successfully by connecting the SDR to the computer and typing "SoapySDRUtil –probe". If everything went well, the probe overview should appear. During the initial set-up and installation of DopTrackBox an error occurred, stating that a file was missing. This can be solved by copying a file that is located in a different folder to the requested folder [24]:

7. “sudo cp SoapySDRPlay/build/libsdrPlaySupport.so /usr/local/lib/SoapySDR/modules0.8 /libsdrPlaySupport.so

After this step, using "SoapySDRUtil –probe" allowed the system to successfully see the SDR.

3.2.3. RX_TOOLS

RX_tools is the software that is used for recording the satellite data [23]. Details surrounding this software were discussed in Section 2.7. The installation procedure can be executed by typing the following commands in the terminal:

1. "git clone https://github.com/rxseger/rx_tools"
2. "cmake ."
3. "make"

3.2.4. DOPTRACK SOFTWARE

The special software that was discussed in Section 2.6 can be found on the DopTrack GitHub. At the time of installing all the different tools and packages were part of a single repository. At the time of writing they have been split into repositories GroundControl and doptools, formerly Process. The procedure described in this section assumes the split repositories as they were at the time of writing. The only difference with the repository at the time of installing is the first ’clone’ command cloning the entire repository containing both GroundControl and doptools. The installation of the DopTrack software is as follows:

1. "sudo apt install at curl"
2. "git clone https://github.com/DopTrack/GroundControl"
3. Go back to the main installation folder.
4. "git clone https://github.com/DopTrack/doptools"
5. "python3 -m venv venv"
6. "source venv/bin/activate"
7. "pip install ."

The GroundControl folder contains the files for the DopTrackBox Scheduler and Recorder software from Section 2.6, whereas doptools contains the DopTrack Processing Package.

3.2.5. OTHER PROGRAMS AND TWEAKS

There is one more small program that is required for DopTrackBox to function properly. In addition, some terminal aliases should be added to streamline the interaction with the system by linking short aliases to frequently used commands. To finalise the DopTrackBox software set-up, the following is needed:

1. "sudo apt install mailutils"
2. Open "~/.bashrc" with a text editor.
3. Add the following aliases:
   i. “alias atc=’atq | sort -k 6n -k 3M -k 4n -k 5 -k 7 -k 1’”
   ii. “alias prb=’SoapySDRUtil –probe’”

[6] Available on: https://github.com/pothosware/SoapySDRPlay3 [Accessed 13 July 2023].

The mailutils software allows the system to send messages to control different programs. The first alias is used to sort the atq chronologically, so it is easier to see which satellite recordings are scheduled at what time. The second alias is a shorthand for the SoapySDR probe, which is useful to see if the SDR is properly connected to the system for recording.

3.3. DopTrackBox Operation Set-up

Once the initial hardware and software setup has been completed, DopTrackBox is ready for use. This section describes how to use DopTrackBox for recording satellite data. The focus lies on the DTB-VM system, as that was the actual system used for data acquisition during this thesis.

Before turning on the VM different hardware components can be connected. The way these components are connected was visualised in Figure 2.16. The antenna can be connected to the SDR, and if applicable for the measurement the GPS clock can be connected through the resistance with the SDR. For the GPS clock to function, the GPS receiver and the battery pack should be connected. It is important to connect the GPS clock before connecting the SDR to the computer. The host machine and the VM can be turned on now, if not on already. When recording data it was found out that some issues could arise when the VM was not shut down after the SDR was disconnected and reconnected for new recordings. Therefore it is recommended to fully shut down the VM after all recordings have finished.

With the VM operational the SDR can be connected to the USB port of the computer. The USB device is not visible to the VM, but only to the host. To connect a USB device to the VM, from within the VM software window go to 'Devices', 'USB', and select the device that needs to be connected to the VM. This should connect the SDR to the VM. The host should produce an audio cue similar to when other USB devices get disconnected, and the VM should produce an audio cue that a device has been connected. To ensure that the SDR is working properly, use the "prb" command in the terminal. If the SDR has been connected properly, it should show the SDR results in the terminal. If not, it should say that no SDR is present.

It is possible to monitor if everything goes correctly by opening a different terminal window and typing "top". During the recordings, the utility 'SoapySDRUtil' should be present and near the top of the list. As long as this program is present, DopTrackBox is recording data. Due to the limitation of a single USB port, DopTrackBox writes the data to the internal SSD. Once all recordings are done for the day, the SDR can be disconnected following the same procedure used to connect a USB device to the VM. Once it has successfully been disconnected, it can be removed from the USB port. The external SSD can now be connected and passed through to the VM. The data is now able to be backed up to the external SSD. Once the external SSD is no longer needed, it needs to be unmounted first to avoid data loss. This can be done by right-clicking the drive in the menu on the left-hand side of the screen. Once a message within the VM pops up notifying of the successful unmount, it can be disconnected from the VM. The drive should then pop up on the host machine. Again, to ensure no data is lost, disconnect the SSD following the proper methods for the respective host OS.

4 | Experiment Set-up

Various experiments will be performed to gather data to answer the research questions. This chapter presents the set up of the experiments and the challenges that arise. Before the data acquisition can begin the hardware and software need to be verified to work as intended. To verify the software however, it is needed to acquire data to use in verification. Therefore the experiments' methodology is laid out before the verification is presented in Chapter 5. The key factors to determine the system's performance are the frequency data and Signal to Noise ratio (SNR) of the recorded signals. The frequency of the measurements will be compared to that of the satellite's two-line element set (TLE) to see the timing error. From the frequency data, the range rate can be calculated and thus the difference between observed and TLE range rate. From the range rate and the frequency, other variables such as the Time of Closest Approach (TCA) and the linear drift can be calculated. This is the intended goal of DopTrack and DopTrackBox; to use Doppler tracking data for satellite tracking. The frequency data will be generally referred to as 'Doppler shift data' in this thesis. SNR tells you if the received data can be extracted or if it gets drowned out too much by the noise. The minimum margin is generally set at 3 dB for the link to close [25, 26]. This is more important when data transmission from satellites is a key factor in a mission, but this is generally not a requirement for Doppler tracking. Therefore this functions more to see what DopTrackBox is capable of in addition to Doppler tracking. The experiments will reflect these two types of data measurements.

To aid in answering Research Question 2, four subquestions have been posed which follow from the different experiments:
Research Question 2.1 is "What is the difference between the SNR and Doppler shift data of DopTrackBox and DopTrack?" To answer this question an as fair as possible comparison between the two systems has been made to analyse the impact of the hardware differences.

Research Question 2.2 is "What is the effect of including or excluding a GPS clock on the SNR and Doppler shift data?". The hypothesis is that including a GPS clock will reduce the observed range rate difference between the measured data and the TLEs and thus result in a smaller difference between the TCA of the measurement and TCA of the TLE. This subquestion aims to test that hypothesis and also functions to compare the DTB with the DTB Pro configuration from Table 2.3.

Research Question 2.3 is "What is the effect of using a different SDR for DopTrackBox on the SNR and Doppler shift data?". This ties into the comparison of the DTB Light and DTB configurations from Table 2.3. The current hypothesis is that DTB will have a higher SNR and a lower range rate difference since it uses higher grade hardware. This subquestion aims to find out whether or not DopTrackBox is still able to perform its intended mission with lower end hardware.

Research Question 2.4 is "What is the effect of manually aiming for DopTrackBox on the SNR and Doppler shift data?". This subquestion is similar to the first, but here the Yagi antenna will be used for DopTrackBox instead of DopTrack station's omnidirectional antennas.

This chapter continues by first selecting four different satellites as potential candidates for the experiments. These are Nayif-1, Delfi-C3, Delfi-N3XT, and FunCube. The main reason to select these four is that DopTrack is also tracking these satellites. Therefore there will be sufficient data to compare between DopTrackBox and DopTrack for both the research questions and the validation process. Of these four satellites, Nayif-1 is chosen as the main satellite. The reasoning behind this is explained in more detail in Section 4.1, along with some background on the satellite. To answer the four subquestions of Research Question 2 different experiments will be conducted. All these experiments will be presented and elaborated upon in Section 4.2. To conduct these experiments it is important to identify the challenges beforehand to minimise their impact. The identification of these challenges is handled in Section 4.3. Finally, the methodology of conducting the experiments is explained in Section 4.4. It is paramount to follow the same methodology each time to minimise the differences between data recordings so that the differences are caused by the data and not by human interactions.

4.1. NAYIF-1

This section is split in two subsections. The first subsection explains why Nayif-1, from now on referred to as just 'Nayif', is chosen as the main satellite for the experiments over Delfi-C3, Delfi-N3XT, and FunCube. The second subsection provides some background on Nayif.

4.1.1. MAIN SATELLITE

There are several criteria that were taken into account to decide which satellite to focus on for this thesis. A crucial factor was previous experience with recording the satellites with DopTrack so that comparison data was readily available, which is why the four earlier mentioned satellites were selected. The other criteria for the trade-off to select the main satellite are given here. Per criterion an explanation will be given.

- **Satellite reliability:** The odds of the satellite transmitting a signal during a pass. When setting up the system to record data, the satellite must be on and be transmitting data. If this is not the case, the measurement will be for nought. The reliability is based on previous experience with tracking the satellites on DopTrack.
- **Satellite pass time:** The local time of day when the satellite passes. If the satellite passes very early or late on the day measurements can possibly not be done due to inaccessibility of the DopTrack ground station.
- **Number of passes:** The number of satellite passes within line of sight of the ground station per day. The more passes, the more data acquisition possibilities.
- **Signal strength:** The strength of the satellite signal. The stronger the signal, the easier it is to pick up.

These criteria are combined in a trade-off matrix which can be seen in Figure 4.1.

DopTrackBox Satellites for Tracking		Criteria				
		Satellite Reliability	Satellite Pass Time	Number of Passes	Signal Strength	Comments
Satellite	Nayif	60% Excellent No reliability issues reported	15% Excellent Two passes during the late morning or early afternoon	5% Good Generally four passes each day	20% Excellent Strong	
	Delfi-C3	60% Unacceptable No battery onboard so it is always off during eclipse	15% Good Two passes in the morning	5% Good Generally four passes each day	20% Acceptable Weak	
	Delfi-N3XT	60% Unacceptable At the time there were some reliability issues	15% Excellent Two passes during the early afternoon	5% Good Generally four passes each day	20% Good Moderate	
	FunCube	60% Excellent No reliability issues reported	15% Acceptable Two passes late in the afternoon	5% Excellent Generally five passes each day	20% Excellent Strong	

Excellent: system enabler	Acceptable: minor system design drivers
Good: no system design driver	Unacceptable: major system design drivers

Figure 4.1: Main satellite selection matrix.

Some observations can be made from this figure. The unreliability of Delfi-C3 and Delfi-N3XT add additional uncertainty to the data acquisition which is not ideal when trying to reliably get data to answer the research questions. This effectively eliminates these two satellites as main targets. When comparing Nayif to FunCube, it can be seen that FunCube passes over the ground station later during the afternoon whereas Nayif passes in the late morning or early afternoon. This results in a more favourable outcome for Nayif, as during the late afternoon accessibility to the ground station can be a more limiting factor. Therefore, Nayif is chosen as the main candidate to perform the experiments on. Data from the other satellites will be recorded too, following the recording and scheduling logic of DopTrackBox as was presented in Section 2.6.1.

4.1.2. SATELLITE BACKGROUND

Nayif-1 is a 1U CubeSat from the United Arab Emirates. Its primary objective is to give Emirati students hands-on knowledge on systems engineering related to spaceflight. The secondary objective is to enable amateur radio communications with the satellite using low-end hardware. Nayif also has a new attitude determination and control system, which will have its maiden flight on Nayif [27, 28]. This secondary objective aligns perfectly with this thesis, as Nayif enables the experiments to take place. Nayif was launched on 15 February 2017 and currently has an orbit perigee altitude of 396.0 km, an apogee altitude of 400.9 km, an inclination of 97.2°, and an orbital period of 92.4 minutes [29]. A photo of the satellite can be seen in Figure 4.2. The four antennas can be seen on the bottom of the satellite. The exact polarisation of the signal will have to be found out during the experiment phase.

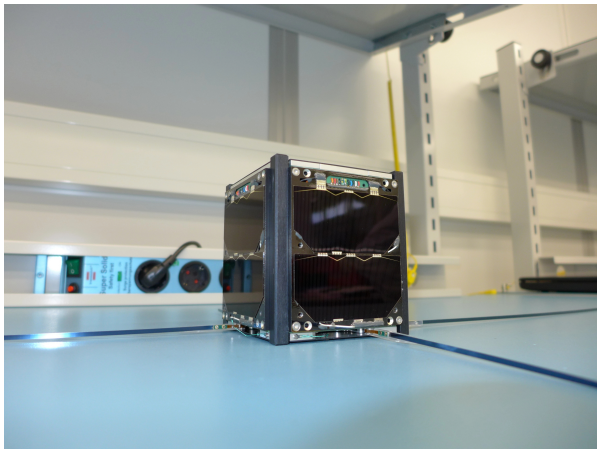


Figure 4.2: A photo of the Nayif-1 cubesat. [27]

During the phase finalising this thesis the Nayif-1 CubeSat has re-entered the Earth's atmosphere on 18 July 2023. This means that Nayif is no longer operational and can therefore no longer be used for any future Doppler tracking experiments. The increased solar activity has led to a rapid decrease in altitude for the satellite, resulting in its re-entry 6 years and 5 months after its launch [30].

4.2. DIFFERENT EXPERIMENTS

To analyse the performance of different parts of DopTrackBox, different kind of experiments will be performed. These will mainly involve different hardware set-ups to see the effect of using different hardware. The first set of experiments eliminate the effect of the directional Yagi antenna. The directionality and polarisation of the antenna mean that the antenna has to be pointed towards the satellite in the right orientation to receive an optimal signal. This can have a large impact on the obtained data if the antenna is not pointed correctly. Therefore, the first set of experiments will be conducted at the DopTrack ground station where the DopTrack omnidirectional antennas can be used. Another benefit of conducting the experiments at the ground station is that the obtained data can be compared between the two systems, as they can record the same satellite pass from the same location simultaneously. This enables the performance comparison to be as fair as possible, as the antennas are the same but the radio and SDR differ.

There are two antennas at the DopTrack ground station. The difference between these antennas can be investigated as well by doing some experiments on one antenna, and other experiments on the other. One of these antennas is labelled with a green tag and the other one with a red-yellow tag. The effect of including

and excluding the Low Noise Amplifier (LNA) on the signal will also be investigated. The first experiment can therefore be subdivided into three subexperiments, which are all mentioned in Table 4.1. These experiments will help answer Research Question 2.1; "What is the difference between the SNR and Doppler shift data of DopTrackBox and DopTrack?"

Table 4.1: Experiment 1 subexperiments and subgoals.

	Goal
Experiment 1.1	Compare data gathered with DopTrackBox from the Green antenna and the Red-Yellow antenna.
Experiment 1.2	Compare data gathered with DopTrackBox with the LNA enabled and the LNA disabled.
Experiment 1.3	Compare data gathered from DopTrackBox and DopTrack.

To see the difference between various hardware configurations of DopTrackBox itself, a second set of experiments will be conducted where different hardware is used. These experiments are not bound to the DopTrack ground station, and will therefore use the Yagi antenna in the field. They will each answer the remaining three subquestions of Research Question 2 and will therefore also provide insight in the advantages and disadvantages of the several proposed DopTrackBox concepts from Table 2.3. The main components that will be changed between the various experiment runs are the radios and the GPS clock. This effectively means that experiments will be conducted with the RSPduo with GPS clock, the RSPduo without GPS clock, and the RSP1A. The second experiment is also subdivided into three subexperiments. These can be seen in Table 4.2.

Table 4.2: Experiment 2 subexperiments and subgoals.

	Goal
Experiment 2.1	Compare data gathered with the DTB and DTB Pro configurations to answer Research Question 2.2.
Experiment 2.2	Compare data gathered with the DTB and DTB Light configurations to answer Research Question 2.3.
Experiment 2.3	Compare data gathered with DTB whilst manually aiming and DopTrack to answer Research Question 2.4.

4.3. CHALLENGES

There are various challenges in conducting the experiments and comparing the results obtained from them. Identifying them beforehand can help streamline the experiments and pinpoint potential difficulties. The hardware requires power, so when conducting experiments in the field this could be an issue. This is especially relevant for Experiment 2. Power can either be provided via a Power over Ethernet (PoE) solution or via a portable battery. When using PoE, the added benefit is that DopTrackBox will have an active internet connection to sync its time. A downside of PoE is that a PoE supported ethernet cable must be provided, which can be difficult when in the field. The chosen location for Experiment 2 provides access to power within 15 metres so both PoE and a battery would be viable. However, the same can not be said in general for a random location.

Currently, the DopTrackBox hardware does not include a screen. Therefore it can be difficult to operate the system when using a remote ssh connection is not possible, i.e. when there is no active internet connection. Again, this is most relevant for Experiment 2 as Experiment 1 will be conducted at the DopTrack ground station with access to power and internet. Adding a small screen to the setup would be the easiest solution, but also the most costly.

Another challenge arises from the comparison of the different satellite passes. For experiments 1.3 and 2.3 direct comparisons can be made as the satellite is tracked on two different systems. For the other experiments this can not be done as DopTrackBox is unable to record the same satellite using multiple radios. Factors that can cause differences between various satellite passes are the distance between the satellite and the ground station, the maximum elevation, weather conditions, and the points where the satellite ascends

above and descends below the horizon. This final factor is especially crucial to note for experiments 2.1, 2.2, and 2.3. Experiments done at the DopTrack ground station have an almost unobstructed view to the entire sky as they are done from the roof of the TU Delft Aerospace Engineering building. The experiments done in the field have different obstructions limiting the line of sight to the satellite, and thus the minimum elevation. It is important to keep this in mind when comparing the data from different satellite passes. The field location which was used for all field experiments can be seen in Figure 4.3. The coordinates of the location are not provided for privacy reasons. They can be requested by contacting the author.



Figure 4.3: Panorama taken from the location where the field recordings were made with the cardinal directions. Due to the way the panorama was taken the cardinal directions are not evenly spaced out. The dashed red line indicates an elevation of approximately 15°.

From this figure it can be seen that there are more buildings in the eastern side than the western side. Therefore, satellites passing in the east will have more obstructions than satellites passing in the west, potentially influencing the signal's quality. Generally, there are more objects obstructing the view at lower elevations compared to the DopTrack ground station. Setting a higher minimum elevation for the software to use to predict recordable satellite passes can circumvent this.

4.4. METHODOLOGY

All the recordings for the same main experiment will follow the same methodology. This methodology is laid out in this section. The difference between the two experiments is the use of the handheld Yagi antenna. Therefore the final steps of the methodology involving pointing the antenna can be ignored for the first set of experiments.

4.4.1. GENERAL METHODOLOGY

The first step is connecting all the hardware components to the computer, making sure the correct SDR and antenna are used. After everything has been connected the computer can be turned on. Once the computer is operational the execution queue has to be checked to ensure the intended recordings are scheduled. If this has been done, DopTrackBox is ready to record the satellite pass. If the recordings are not scheduled, the scheduling software has to be executed. When it is time to start the recording, the system automatically engages the recording software. To minimise system strain during the recordings no other tasks should be performed on the computer.

After the recording has been finished, the recording should be processed with the different software tools. Once this has been done, the recording and the processed data have to be moved from internal storage to the external storage to ensure sufficient storage is available for further recordings.

4.4.2. MANUALLY POINTING

If an antenna is used that has to be pointed, the point and time where the satellite appears above and disappears below the horizon need to be noted, as well as the point and time of maximum elevation. Due to the gain pattern of the Yagi antenna the satellite does not need to be followed precisely, as was discussed in Section 2.5. There it was found out that being within 30° of the actual position of the satellite's azimuth results in no received signal power loss. The total time of the pass will be split in five equal sections, as will the satellite arc. This ensures the satellite will be within the antenna's beamwidth where its gain is optimal. During the first section the antenna will be pointed towards the point where the satellite rises above the horizon. For the second section, the antenna will be pointed between the rise point and the apex. During the third section the antenna is pointed at the point of maximum elevation. Then the antenna is pointed between the set point and the apex for the fourth section, and finally at the point where the satellite disappears below the horizon for the fifth section.

For the first few experiments whilst manually pointing, the orientation of the antenna will be switched every two minutes between horizontal and vertical. The goal of this is to find out whether or not the signal is polarised, and in which way if any. If the signal is polarised, the antenna needs to be pointed in a specific direction as the Yagi antenna cannot receive signals that are not polarised in the correct direction. After it has been found out if the signal is polarised, the optimal orientation strategy can be devised.

5 | Verification and Validation

Before any experiments can be conducted, the hardware and software have to be verified. This follows the set up presented in Chapter 4. With the verification and validation process, it is also possible to answer Research Question 1 as described in Chapters 2 and 3. The first step is to see whether the hardware works as intended. This is explained in more detail in Section 5.1. After the hardware has been verified, the software that is used for the recording and data analysis can be verified. This order has been chosen to eliminate hardware errors before looking at potential software issues. To verify the software real data will be used following the data acquisition guidelines. The software verification is presented in Section 5.2. The conclusions drawn from the verification process regarding Research Question 1 will be discussed in Section 5.3.

5.1. Hardware Verification

The first step is verifying the hardware with some general tests. This section is divided in two subsections. First, the original concept's verification process and the outcome is presented in Section 5.1.1. As some issues arose, changes to the hardware had to be made. The updated hardware is discussed in Section 5.1.2.

5.1.1. Initial DopTrackBox concept

Data at radio station frequencies was recorded to see whether the recordings yielded the expected result. The system overview can be seen in Table 5.1.

Table 5.1: DopTrackBox Hardware and Software overview during radio test verification.

Hardware					*Software*	
Computer	**SDR**	**GPS Clock**	**Antenna**	**Storage**	**OS**	**Recorder**
Raspberry Pi 3B	RSPduo	None	Yagi	External SSD	Ubuntu Server 21.10	rx_tools

For all tests the system was accessed via an SSH connection over the local network. During these test runs inconsistent behaviour was observed. Tests were done with different frequencies, sample rates, and total number of recorded samples. An overview of these tests can be seen in Table 5.2. The command line prompt for these tests is the following, with the variables used as presented in Table 5.2:

```
./rx_sdr -f [VARIABLE] -F CF32 -I CF32 -s [VARIABLE] -n [VARIABLE] -a 'Tuner 1 50 ohm' /
    media/external/testrun230622-[n].dat
```

Table 5.2: Radio test results with RSPduo on Raspberry Pi

Test name	**Frequency [MHz]**	**Sample rate [kHz]**	**Number of Samples**	**Status**
radiotest2.wav	102.7	250	2.5M	Success
testrun210622-1.dat	100	125	1.25M	Success
testrun230622-1.dat	102.7	125	1.25M	Crash
testrun230622-2.dat	100	125	1.25M	Success
testrun230622-3.dat	145.87	125	1.25M	Crash
testrun230622-4.dat	102	125	1.25M	Crash
testrun230622-5.dat	100	125	1.25M	Success
testrun230622-6.dat	145.87	250	2.5M	Crash
testrun230622-7.dat	110	125	1.25M	Crash
testrun230622-8.dat	102.7	125	1.25M	Success
testrun230622-9.dat	102.7	125	1.25M	Crash
testrun230622-10.dat	102.7	125	1.25M	Crash
testrun230622-11.dat	102.7	125	1.25M	Crash
testrun230622-12.dat	100	125	1.25M	Crash

The crashes all generated the same message: *'sync read failed. -1 Short write, samples lost, exiting!'*, except for testrun230622-6.dat, which resulted in the message *'sync read failed. -1 Segmentation fault (core dumped)'*

The 'sync read failed' message indicates that the system cannot handle the load [31]. This might be caused by the CPU being not powerful enough or by a bandwidth limit on the data bus. The recorded data by the RSPduo enters the system via a USB 2.0 port, which is then written to the external SSD which is connected to another USB 2.0 port. This, combined with the SSH connection, could be too much for the system to handle.

To see if this could be resolved by using different hardware configurations, more tests were conducted. To limit the potential strain on the Raspberry Pi, all these tests were conducted locally on the machine instead of via an SSH connection. The command line prompt for these tests follows the following input scheme:

```
./rx_sdr -f 145.870M -F CF32 -I CF32 -s [VARIABLE] -n [VARIABLE] -a 'Tuner 1 50 ohm' -b
    4194304 [filename]
```

The specific input values and results can be seen in Table 5.3. The output in that table refers to where the data was written to; either local on the SD card or on the external SSD. Tests 02, 03, and 08 should have lasted 10 seconds, but they were completed in a shorter time than expected.

Table 5.3: Radio test results on Raspberry Pi with various hardware, sample rates, and number of samples at 145.870MHz

Test name	**Sample rate [kHz]**	**Number of Samples**	**SDR**	**Output**	**Status**	**Note**
testrun270622-...-01.dat	250	2.5M	RSPduo	Local	Crash	-
testrun270622-...-02.dat	125	1.25M	RSPduo	Local	Success	Short runtime
testrun270622-...-03.dat	125	1.25M	RSPduo	SSD	Success	Short runtime
testrun270622-...-04.dat	125	1.25M	RSPduo	SSD	Crash	No '-b' tag
testrun270622-...-05.dat	125	12.5M	RSPduo	SSD	Crash	-
testrun270622-...-06.dat	125	12.5M	RSPduo	Local	Crash	-
testrun270622-...-07.dat	250	2.5M	RSP1A	Local	Crash	-
testrun270622-...-08.dat	125	1.25M	RSP1A	Local	Success	Short runtime
testrun270622-...-09.dat	125	1.25M	RSP1A	SSD	Success	-
testrun270622-...-10.dat	125	12.5M	RSP1A	Local	Crash	-
testrun270622-...-11.dat	125	12.5M	RSP1A	SSD	Crash	-

As can be observed from Table 5.3, the tests with a sample rate of 125 kHz and a total number of samples of 1.25M were successful, both when storing on the local SD card and on the external SSD. All the tests of 100 second duration crashed. The crash message was the same as with the previous tests: *'sync read failed. -1 Short write, samples lost, exiting!'*. This means that a short recording of 10 seconds is achievable, but 100 seconds is not. As all satellite passes are longer than 100 seconds, the conclusion can be drawn that the Raspberry Pi is not powerful enough to conduct the experiments. Therefore, a different solution must be found to replace the Raspberry Pi as the computer for DopTrackBox.

5.1.2. UPDATED DOPTRACKBOX CONCEPT

To replace the Raspberry Pi a new computer had to be found. Finding and ordering a good replacement would take time, so the decision was made to use a personal laptop for the remainder of this thesis to run DopTrackBox. The laptop's specifications can be found in Table 5.4 for further reference. There are several advantages of using this computer over the Raspberry Pi. The first is that the Surface has an internal battery which removes the need for external power, either with PoE or an external battery pack. Secondly, the Surface has a build in screen which solves the issue of the Raspberry Pi currently not having one. This makes it easier to monitor the recordings whilst they are happening. The disadvantage of using the Surface over the Raspberry Pi is the higher price of the hardware.

Table 5.4: Updated computer hardware specifications

Model	Surface Pro (2017)
CPU	Intel i7-7660U (2 cores, 4 threads @ 2.50 GHz)
GPU	Intel Iris Plus 640
RAM	8 GB
Storage	256 GB SSD
Ports	1 USB 3.0 1 Mini display port
Display	Build in 2736 × 1824 pixels
OS	Windows 10 Pro (21H2)

To ensure compatibility between DopTrackBox and DopTrack, and to ensure backwards compatibility with the previously developed software the operating system has to be Linux based. Since this laptop is still in active use, the operating system could not be overwritten and the choice was made to run DopTrackBox within a Virtual Machine (VM). A very important point is that the VM should support USB passthrough since the SDR is connected to the computer via USB.

The first option that was investigated is Hyper-V, since it was a natively available solution. After installing the software it became clear that enabling USB passthrough was more difficult than expected. Hyper-V does not natively support USB passthrough, so several workarounds were tried. These were trying to let the VM directly communicate with the USB device and enabling Hyper-V 'enhanced session' mode. The former did not work as the system could not interact with the SDR and the latter did not work as Linux VMs do not support enhanced session mode for USB passthrough [32].
Hyper-V was abandoned and a switch was made to Windows Subsystems for Linux, as it is another natively available solution. After some investigation the same USB passthrough issues were prevalent here.
The third option was Oracle VM VirtualBox. An advantage of VirtualBox over Hyper-V and Windows Subsystems for Linux is the ability to run a Linux OS with a GUI. Since the laptop has sufficient overhead there is no need to use a terminal only OS. To enable USB passthrough in VirtualBox an extension pack was downloaded and installed, after which the SDR could be detected by the software. VirtualBox allows the user to allocate system resources through its interface. The resources DopTrackBox has access to, as well as other information on the virtual machine can be found in Table 5.5.

Table 5.5: DopTrackBox virtual machine (DTB-VM) initial specifications

VM software	Oracle VM VirtualBox V6.0.24 + Extension Pack V6.0.24
CPU	1 thread allocated @ 2.50 GHz
GPU	16 MB allocated
RAM	2 GB allocated
Storage	32 GB allocated
OS	Ubuntu 22.04 LTS

The combination of the hardware and software of Tables 5.4 and 5.5 together has been given the name DopTrackBox VirtualMachine, or DTB-VM. Since the host machine only has a single USB port which needs to be used for the SDR, the data has to be stored locally before it can be moved to the external SSD. This new hardware setup was verified using the same procedure as the Raspberry Pi. The same tests from Table 5.3 were done, but all using local storage since the sole USB port is used for the SDR. As all tests were successful, the hardware was deemed verified and ready to acquire actual data.

Not everything could be foreseen in advance, since new hardware and real world experiments are involved. During some of the experiments more insight was gained to streamline the data acquisition process. These will be explained in more detail in Chapter 8 to ensure future research building on these findings take them into account. When multiple tasks are running on the VM or host at the same time, the recorder can crash due to lack of computer resources. To mitigate this the number of allocated threads for the VM as well as the allocated RAM were increased from 1 to 2 threads and 2 to 3 GB. The final VM specifications are visualised in Table 5.6.

Table 5.6: DopTrackBox virtual machine (DTB-VM) final specifications

VM software	Oracle VM VirtualBox V6.0.24 + Extension Pack V6.0.24
CPU	2 threads allocated @ 2.50 GHz
GPU	16 MB allocated
RAM	3 GB allocated
Storage	32 GB allocated
OS	Ubuntu 22.04 LTS

A final adjustment to the host machine was made to ensure it kept recording in the field. The standard sleep timer of host machine was set to 5 minutes when on battery power, which meant that 5 minutes into the recording the host machine would go to sleep which cut the recording short. Therefore the sleep timer was increased to 15 minutes. This is long enough for all the recordings to complete. Finally, the WiFi on the host machine was disabled during recordings in the field. The WiFi signal strength at the recording location is very weak, causing the system to continuously try to reconnect to the WiFi using valuable system resources in the process. Disabling the WiFi prevented recording issues.

When using the GPS antenna for the recordings with the DTB Pro configuration an additional external battery was used. The GPS antenna required power via USB, and since the host machine only has 1 USB port which is used by the antenna power had to come from a different source.

5.2. SOFTWARE VERIFICATION

Most of the software that is used in DopTrackBox is directly taken from DopTrack with some small changes made to accommodate the hardware differences. Since the software from DopTrack is verified to work as intended, a closer look has to be taken on the changes. As the software objectives can be divided into three distinct goals, the verification was also done separately for them. The scheduler verification is discussed in Section 5.2.1. Then the recording software is verified in Section 5.2.2. Finally, all the post processing software is verified in Section 5.2.3. This section will not dive into details on how the software functions, as that was discussed in Section 2.6.

5.2.1. SCHEDULER VERIFICATION

The scheduler takes the satellite list and checks when they are in view to enable recordings. The scheduler consists of the following code packages, all adopted from DopTrack:

- run_schedule.sh
- Schedule.py
- getTLE.sh
- predict_v2.py
- make_atq.sh

At the time all DopTrack code was written in Python 2, so every program had to be manually updated to Python 3 as DopTrackBox uses Python 3.10. This included changing the syntax to support Python 3 and changing the called program from Python 2 to Python 3. All the indentations in the code had to be fixed manually too, as there were inconsistencies causing errors. Next to this the references to the storage locations had to be updated to reflect the new hardware.

The verification and validation process here consisted of running the software using various parameters for input satellites and minimum elevation values. The output of the scheduler could be compared against online results on *N2YO.com* for Nayif [33], Delfi-C3 [34], and Delfi-N3XT [35], as N2YO shows a 10 day prediction of when the satellite pass starts. In general, the time of the scheduler should be 1 minute before the actual starting time to ensure nothing of the pass is missed in case of software inaccuracies. Some indentation mistakes were found in make_atq.sh, which resulted in some parts of various 'if-statements' to be part of other 'if-statements' or of other branches of the same 'if-statement'. This caused some parts of the code to be skipped, which resulted in some recordings to erroneously not show up when they were supposed to. After identifying the issue, the indentations were fixed and no further issues were spotted. These mistakes were caused by the transference of code from DopTrack to DopTrackBox and the required adaptations that had to be made to accommodate the code.

5.2.2. RECORDER VERIFICATION

The recorder records the satellite passes as scheduled by the scheduler. The recorder consists only of the Record_DTB.py program, which is a modified version of the original DopTrack Record.py file.

The main changes are the update from Python 2 to Python 3 code and the code line changes representing the different recording software and hardware. The way the recorder has been verified and validated was by creating dummy satellite passings to see if the recorder picked them up and started the recording process without issue. As this is the part of the software that is most closely related to the hardware part of the verification was done in tandem with the hardware verification; as both need to function to obtain data.

5.2.3. PROCESSING VERIFICATION

The data processor uses the recorded data and analyses it to create spectrograms of the passings, and extract SNR and Doppler shift data. The post processing software consists of the following programs, which are all based op their DopTrack counterparts:

- curvetool.py
- signals.py
- io.py
- DopTrack processing package

The DopTrack processing package is available on the DopTrack GitHub and uses the exact same code without changes. The other three programs are modified to accommodate DopTrackBox, with curvetool.py being so heavily modified that it can be regarded as a new program. The main factor for verification and validation is the generation of the spectrogram, as the data represented within is then further processed and analysed to show the relevant information. As each satellite pass and every hardware configuration is different, exact comparisons cannot be made. The first steps are looking at if the processor finds the data and if the spectrogram shown behaves as expected. The frequency of the signal should decrease over time, per the Doppler effect.

Initially no signal could be discovered so the tuning frequency variable in curvetool.py was changed to see if the signal was translated. A frequency addition of 70,000 Hz was needed to get the signal fully into view. The spectrogram was mirrored as it showed the frequency increasing over time. To see if the issue lied in the recording software or the processing software, the data was transferred to the DopTrack systems and put through a Matlab program to create the spectrogram. Here the spectrogram was created correctly, so the recorder is functioning as intended and the data is therefore correct. As all the data is flipped on DopTrackBox, it was thought that the different operating systems could be the cause of the issue. Different operating systems can interpret binary data differently, causing this effect. To mitigate this issue the spectrogram plots were flipped so that they are presented as they should be. This also explains why the 70,000 Hz shift is needed, as the tuning frequency of DopTrackBox differs with 35,000 Hz from that of DopTrack which becomes 70,000 when mirrored. For the analysis of DopTrack reference data, the image flip and the frequency shift are removed to create spectrograms similar to those created by DopTrack itself.

Some minor changes to signals.py have been made to enable the SNR calculations. Instead of outputting the signal power, the software was changed to output SNR. Every row of the signal, which represents 1 second, is analysed separately. The Signal to Noise ratio can be found by finding the signal value and dividing it by the noise. To do this, the noise floor is found for each second by which the signal is divided to get the SNR on every point. This process costs compute time and should therefore be sufficiently accurate whilst still being quick. To do this different methods were used and compared with each other to find the option providing sufficient accuracy for its performance. For every second in the signal, the mean and standard deviation were calculated using Equations 5.1 and 5.2.

$$\mu = \frac{1}{N}\left(\sum_{i=1}^{N} x_i\right) \tag{5.1}$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \tag{5.2}$$

These were used to include or exclude certain parts of the data. The main options that were investigated were using $\mu \pm \sigma$ and $\mu \pm 3\sigma$ with several iterations until convergence is found. These iterations cut off the signal using the $\mu \pm 3\sigma$ threshold and calculated the average of the remainder of the data. This was done until convergence was found for the average of the remainder, where the difference between the currently obtained average and the previous average was smaller than 0.01% or if more than 50 iteration cycles were performed.

The noise is defined as any value within the specific bounds set by the method and the signal is anything above $\mu+3\sigma$. During verification it was found out that parts of the signal could be cut off when there are high spikes in signal strength. Therefore the signal definition was changed to $\mu+2\sigma$. This means that according to the processor there is always a signal, even when there is none. Therefore the SNR will never be 0, but will always have a minimum value of around 2.4 dB. Figure 5.1 shows the effect of using method 1 ($\mu \pm \sigma$) and method 2 ($\mu \pm 3\sigma$ with iterations for convergence) on the unprocessed data, which is the raw data before any signal or noise processing. The same signal can be seen in Figure 5.2, but now lines have been drawn to show the average for both methods, as well as the average for the original unprocessed data. From Figure 5.2 it can be observed that the mean values for the processed data are nearly the same: being 3.0964 for method 1 and 3.1165 for method 2, resulting in a 0.65% difference between the two methods. This implies that the added complexity of method 2 does not necessarily result in more accurate results for the noise floor calculations. Method 2 takes around 4 times the compute time on hardware that is faster than DopTrackBox, so on DopTrackBox' lower end hardware the compute time difference is expected to be even bigger. Therefore the decision was made to use method 1 for further processing. The code used for this was changed from Matlab code to Python code and added to signals.py.

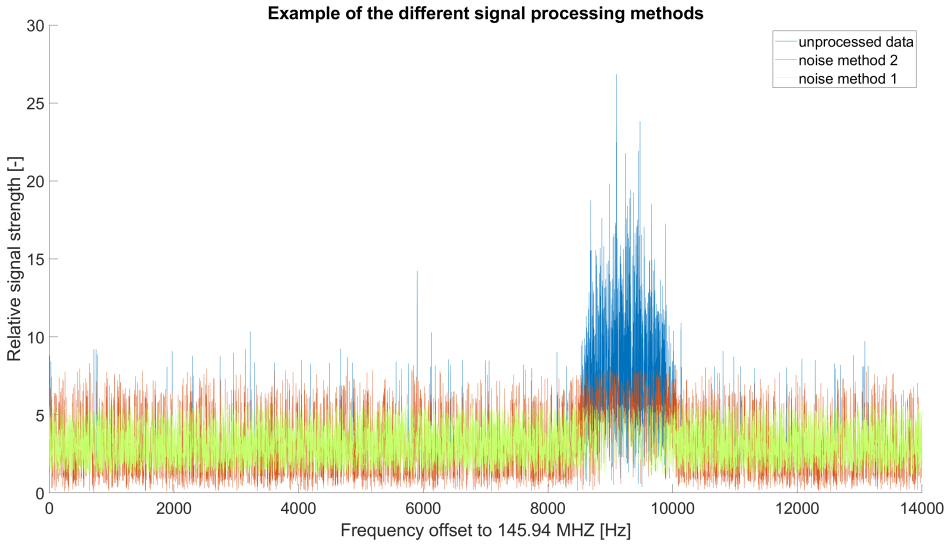


Figure 5.1: The original signal compared to method 1 and 2 to define the noise in the signal.

One of the main outputs is the spectrogram, as can be seen in Figure 5.3. This is a spectrogram of one of the acquired data sets from DopTrackBox, together with its noise floor data. The spectrogram x-axis shows the frequency of the signal with respect to a reference value; 145.94 MHz, which is based on the tuning frequency of Nayif. The y-axis shows the time progression of the recording. The colour of the spectrogram correspond to the various SNR values. The noise floor plot on the right provides additional information on the signal, as the noise floor is used as the noise value to calculate the signal to noise ratio. The spectrogram is made up out of a number of bins with a dt of 1 second, so the total number of bins is equal to the length of the recording. Each bin is then passed through the SNR processor as mentioned earlier in this section, where the noise and signal are isolated. The full data is divided by the noise floor, which is the average value of all noise values in that bin. This produces the SNR which can be seen in the spectrogram. It is also clearly visible that there is a lack of signal at the start and end of the spectrogram, which corresponds to the 60 second grace period before and after each of the satellite passings. The spikes in noise strength can also be clearly seen on the spectrogram, as the signal itself is drowned out there by the higher noise floor.

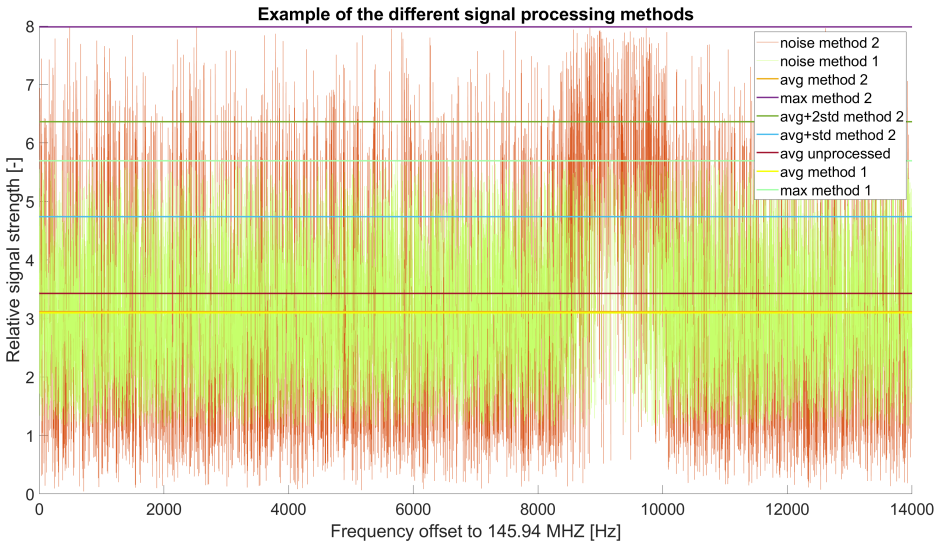
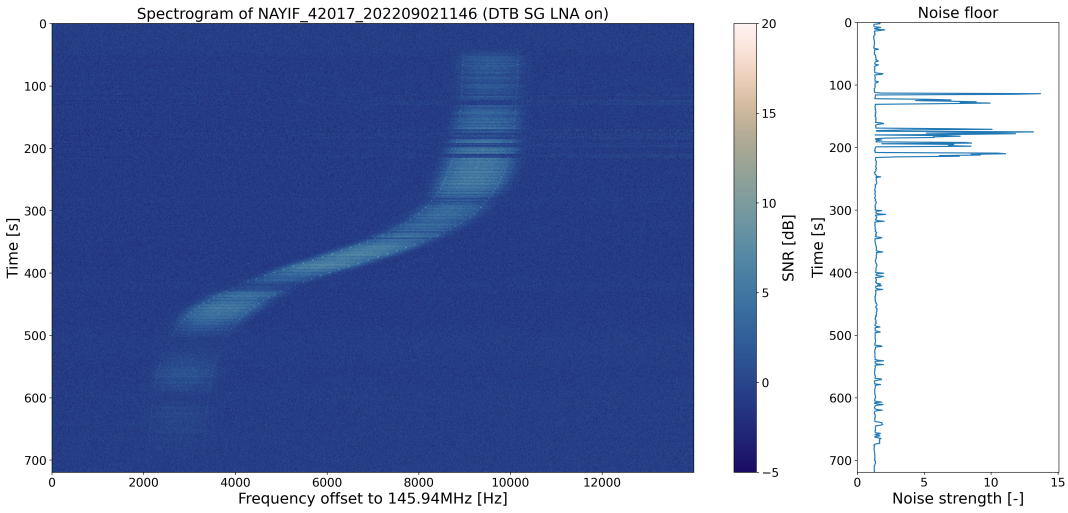


Figure 5.2: Noise as determined by method 1 and 2 with the respective average values to set as the noise floor.



(a) Spectrogram

(b) Noise floor

Figure 5.3: Spectrogram and noise floor for Data set 1 (NAYIF_42017_202209021146).

The final programs that had to be verified were the DopTrack processing package. During this thesis a newer version was published on the DopTrack Github, but the decision was made to stick with the older version as the newer version could break other dependencies. The verification process was done in joint venture with the developer of the DopTrack processing package, M. Søndergaard. The DopTrack processing package could not find any data points within the recorded DopTrackBox data. This was caused by the mirrored nature of the DopTrackBox data with respect to what the program expects. It was found out that within the io.py file a correction was made for the program to interpret DopTrack's data correctly, which results in DopTrackBox' data being mirrored. Removing this correction factor from the io.py file allowed the DopTrack processing package to successfully analyse DopTrackBox' data.

5.3. VERIFICATION CONCLUSIONS

With all the information gathered in this section Research Question 1, *Can the proposed DopTrackBox concept be build?*, can be answered.

The original concept, as shown in Section 2.2, was put through the verification process. There were two subquestions to this main research question: 'Can the selected hardware combination record satellite data required for Doppler tracking?' and 'What are the hardware requirements of DopTrackBox?'.

From Section 5.1 it became clear that this original concept is in fact unable to fulfil the intended functions, as it failed in the key objective of data acquisition. Therefore the selected hardware combination cannot record the satellite data that is required for Doppler tracking. The answer to the first subquestion is therefore no. This also means that the initially proposed DopTrackBox concept, using a Raspberry Pi 3B as the main computer, cannot be build. However, this does not mean that the idea of DopTrackBox cannot be pursued. DopTrackBox is supposed to be a portable Doppler tracking ground station, which is still feasible as was proven after upgrading the DopTrackBox hardware from using a Raspberry Pi to DTB-VM.

The second subquestion about the hardware requirements can help shed a light on what is needed for DopTrackBox to function. Since the original concept with the Raspberry Pi did not work, but the DTB-VM does work the question becomes what is the least powerful computer that can still run all the required DopTrackBox software? Due to time and hardware constraints this thesis could not investigate this further, so this will be included as a potential recommendation for further research in Section 8.4. Something can be said about the hardware that was used for the experiments, as DTB-VM was able to successfully gather the required data. Since the computer is the most critical part of the system, the hardware requirements revolve mostly around the computer.

Table 5.6 showed the computer hardware specifications. These can be used as verified minimum system requirements. Due to the system running in virtual environment some additional overhead might have been needed, resulting in lower minimum requirements when the DopTrackBox software is run natively on a machine. Having a sufficiently fast CPU is required to handle the data stream from the recording. The issue of a lack of system resources that resulted in the increased specifications from the initial concept could be attributed to the system running in a virtual environment. Here the host machine uses those resources too. Having at least 32 GB of storage is recommended to have sufficient space to install the operating system and all the other software required to run DopTrackBox with some spare space to store some data if no external storage is at hand. Another recommendation regarding the computer hardware is to have at least 3 USB ports. One for the antenna, one for the external storage drive, and one for power for the GPS antenna. Additional ports could be used for peripherals like a mouse or keyboard if those are not connected in another way. Having a screen included with DopTrackBox is also a nice improvement over the first concept, as it makes it easier to operate and monitor the system. Other required hardware components not linked to the computer, are an SDR where both the RSPduo and RSP1A are options, a GPS clock, and an antenna.

These findings also affect the requirements from Section 2.4. Due to the change from dedicated hardware to a virtual machine, some requirements can be removed or have to be changed. Table 5.7 shows a requirement that is no longer needed, as the current system has a build in screen.

Table 5.7: Obsolete and removed DopTrackBox requirements after verification and validation.

Identifier	**Requirement**
DTB-TECH-1.03	The computer shall have the means to connect to a screen.

One of the requirements was updated, which can be seen in Table 5.8. For DopTrackBox to function, technically only a single USB port is needed. This does mean that some functionality that was intended to work simultaneously over multiple USB connections cannot be delivered. For example, DopTrackBox cannot directly store the recorded data on the external storage device. It is still recommended to have at least 3 USB ports available for greater flexibility, but the technical minimum has been set to 1 with this requirement update.

A new set of requirements is added to regulate the virtual machine. These have been designated as *DTB-TECH-7* and can be seen in Table 5.9.

Table 5.8: Updated DopTrackBox requirements after verification and validation.

Identifier	Requirement
DTB-TECH-1.01.1	The computer shall have at least 1 USB type A port. *Was 'shall have at least 3 USB type A ports'.*

Table 5.9: New DopTrackBox requirements after verification and validation.

Identifier	Requirement
DTB-TECH-7.01	The VM shall be capable of USB passthrough.
DTB-TECH-7.02	The VM shall be capable of running a Linux based Operating System.
DTB-TECH-7.03	The VM shall have a minimum allocated storage capacity of 32 GB.
DTB-TECH-7.04	The VM shall have a minimum allocated RAM capacity of 3 GB.

Some of the requirements for the VM overlap with those of the computer, as the computer is both the host machine and the VM in this regard. These requirements here are specifically tied to a VM, so a DopTrackBox system that does not use a VM does not need these requirements. All of the requirements in Table 5.9 are based on the findings presented in this chapter.

A full requirement pass will be conducted in Section 8.1, which will also include the changes presented in this section.

In conclusion, the full answer to Research Question 1 is thus; 'No, the proposed concept cannot be build, but a different, more powerful, DopTrackBox concept can be build'.

6 | Results

The data gathered from the experiments as shown in Section 4.2 is presented in this chapter. The chapter is divided into the two main experiments and an explanation on how the data is presented. This explanation provides insight into the choices made and is therefore placed at the beginning of this chapter in Section 6.1. The chapter is further divided into the two experiments, where Experiment 1 is discussed in Section 6.2 and Experiment 2 in Section 6.3. For each experiment an overview is presented to show which data is used to answer the question. All the experiments were done with the DTB-VM hardware and software, changing the hardware part that is investigated in each experiment. Each data set has been given its own unique identifier for easier reference and comparison, along with a subidentifier within that experiment. The graphs shown in this section provide the data that is used for the analysis of Chapter 7. Additional data that was collected as part of the measurements, but not directly used for the analysis can be found in Appendix A.

6.1. Data Presentation

This section provides additional insight into the way the data in this chapter is presented. The data obtained from the various experiments can be divided into two categories: SNR, and Doppler shift and range rate. The former will be discussed in Section 6.1.1 and the latter in Section 6.1.2.

6.1.1. Signal to Noise Ratio data presentation

This section looks at how the SNR is obtained from the results and presented in this chapter. The methodology behind the SNR calculations, along with the implications, were discussed during the software explanation and verification in Sections 2.6.3 and 5.2. For the SNR calculations three different values were obtained. Those being the mean, median, and maximum SNR. An overview of the relation between these three values can be seen in Figure 6.1, where the mean, median, and maximum SNR values for Data set 1 are visible. The mean data is shown in blue, the median data in orange, and the maximum data in its own graph in green. This data is obtained by calculating the respective value for every second over the entire frequency width. To interpret the SNR data, the noise floor data can be used. With both the SNR and noise floor data, the signal can be calculated by multiplying the values. The corresponding noise floor plot can be seen in Figure 6.2. To save space in the report, the other graphs with the mean, median, and maximum SNR and the noise floor plots values are placed in Appendix A.

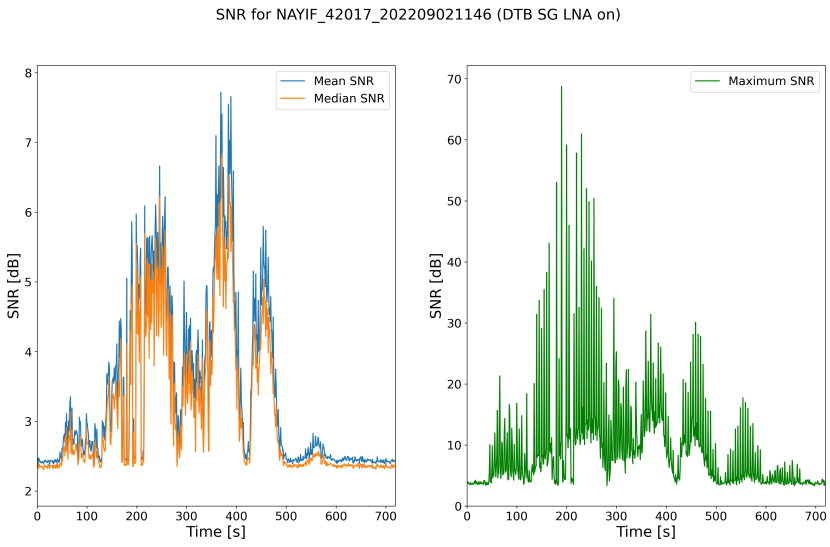
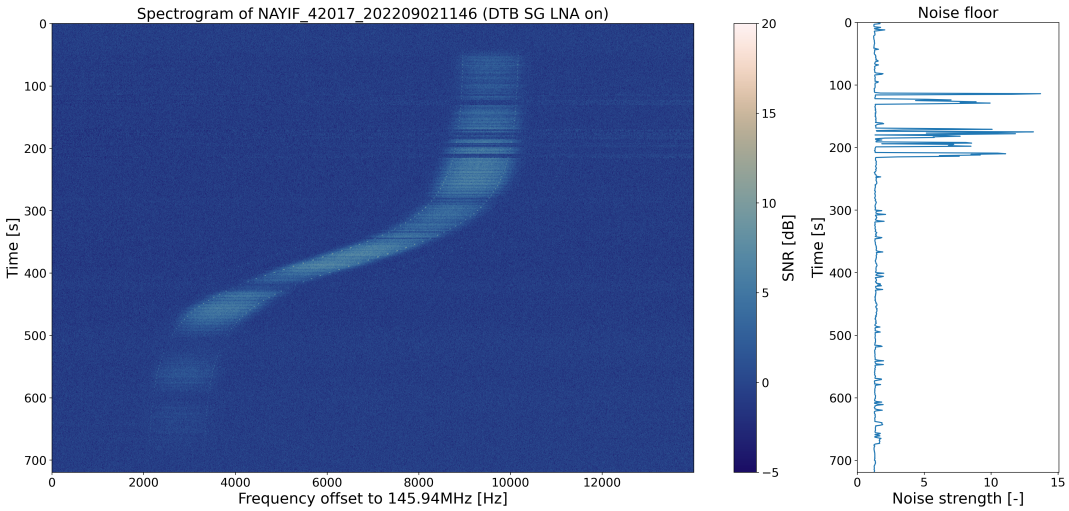


Figure 6.1: Data set 1 (NAYIF_42017_202209021146) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure 6.2: Spectrogram and noise floor for Data set 1 (NAYIF_42017_202209021146).

As was described in Section 5.2, the signal part of the data is defined as any values above $\mu+2\sigma$ and the noise anything between $\mu \pm \sigma$. For the noise floor the average value is used, but for the signal either the average, median, or maximum for every second is noted. It was decided to use the mean for the result plots from Chapter 6 as the values above the threshold should all be weighted to get an idea of the signal and thus the SNR. When using the median, outliers are less taken into account compared to the mean. Since these outliers are high signal spikes, they do not need to be ignored. Another argument of using the mean for the signal is that the mean is also used for the noise floor. The maximum says something about the outlier peaks, but says less about the entire signal when trying to receive data. Any further graphs showing the SNR in this chapter will thus contain the mean SNR for every second in a data set.

6.1.2. DOPPLER SHIFT AND RANGE RATE DATA PRESENTATION

The Doppler shift and range rate data presentation is discussed in this section. This is the Doppler and range rate data on which the system's Doppler tracking capabilities are defined. The methodology for the Doppler shift data acquisition was discussed in Sections 2.6.3 and 5.2. The DopTrack Processing Package (DTPP) produces two main output csv files. One has six different variables which are shown for every data point. These variables are the time, frequency, range rate, range rate according to the TLE, range rate first residual, and range rate second residual. The second file is a summary file, containing additional details of the pass. The details of specific interest are the Time of Closest Approach (TCA) and the first residual fit slope of the frequency data. The TCA determines when the satellite is the closest to the ground station according to the data, which can be compared against the value of the TLE. The first residual fit slope of the frequency data grants insight into the frequency drift of the SDR.

There are two main types of graphs obtained from the Doppler shift and range rate data. These show the frequency and range rate first residuals over time. The range rate first residual is the difference between the measured range rate and the TLE range rate, and are therefore labelled as 'range rate difference' in all the graphs. The TLE range rate is obtained by projecting the satellite's orbit using the TLE parameters. The range rate difference compares thus the measured data and the therefrom derived range rate with the theoretical range rate. The range rate difference graphs provide the most insightful information and will therefore be the most prevalent in this chapter. Additional graphs showing the frequency throughout the data set can be found in Appendix A.

6.2. EXPERIMENT 1: THE DOPTRACKBOX AND DOPTRACK COMPARISON

The first set of experiments look into the differences between DopTrackBox and DopTrack. These experiments all took place at the DopTrack ground station. Table 6.1 shows the configuration of DopTrackBox for these experiments. The antenna varies per experiment as was presented in Table 4.1. Here DTGS Green is the green labelled VHF antenna of the DopTrack ground station and DTGS Red-Yellow the red-yellow labelled VHF antenna of the DopTrack ground station. The GPS clock could not be used due to the lack of GPS signal

inside the ground station. For Experiment 1.1 and 1.2 only the SNR data is used, as they compare different antennas with the same DopTrackBox hardware. Experiment 1.3 uses both the SNR and Doppler shift data, as all the data that is used in Experiments 1.1 and 1.2 is also used in Experiment 1.3.

Table 6.1: DopTrackBox hardware and software used for data acquisition during Experiment 1.

Hardware					***Software***	
Computer	**SDR**	**GPS Clock**	**Antenna**	**Storage**	**OS**	**Recorder**
DTB-VM	RSPduo	None	DTGS Green DTGS Red-Yellow	Internal	Ubuntu 22.04 LTS	rx_tools

These experiments were conducted from 29 August 2022 to 2 September 2022. To see the difference between the different antennas, if there is any, the antenna that is used to gather the data was changed during the week. Initially on Monday and Tuesday, the Green VHF antenna was used. This was switched to the Yellow-Red VHF antenna on Wednesday and Thursday. In addition to the change in antenna during the week, the DC power supply was switched to 0.62A on Tuesday after the experiments were conducted. This increases the signal strength, but also increases the noise. Finally, on Friday the antenna was switched back to the Green VHF antenna to gather data on the initial antenna with the increased power. One more measurement was done on 14 October 2022 using the Green antenna to get more data.

An overview of the recordings can be seen in Table 6.2. The details were obtained from the yaml files at the time of writing the report. This was done since they were not noted at the time when the recordings took place, which was done for Experiment 2. Since there are multiple recordings per day, the starting time of the recording is noted as well. The horizontal lines below the Data set identifiers are used to visually split the 11 data sets in the three data categories of Experiment 1. Those are the DTB Station Green antenna with LNA on, the DTB Station Red-Yellow antenna with LNA on, and the DTB Station Green antenna with LNA off.
As can be seen in Table 6.2, one of the recordings failed. The exact cause of this is unknown, but the most likely reason is the usage of the host machine causing the system resources for the VM to become limited. As the VM could no longer process the incoming data stream, it resulted in a crash causing the data to be not fully recorded.

Table 6.2: Overview table of all measurements for Experiment 1. Azimuth column shows the azimuth of the start of recording and azimuth at the end of the recording, along with if the pass was to the east or west of the ground station.

Experiment 1 Measurements					
Identifier	**Date**	**Starting Time**	**Start - End Azimuth [°]**	**Maximum Elevation [°]**	**Status**
Data set 1	2-9-2022	11:46	17 - 181 (E)	46	Success
Data set 2	2-9-2022	13:19	359 - 236 (W)	16	Success
Data set 3	14-10-2022	11:14	25 - 160 (E)	20	Success
Data set 4	31-8-2022	11:19	24 - 164 (E)	23	Success
Fail_5.1	31-8-2022	12:52	4 - 220 (E)	29	Failure
	Recording crash. Probable cause multitasking on host machine.				
Data set 5	1-9-2022	10:46	33 - 141 (E)	11	Success
Data set 6	1-9-2022	12:19	10 - 200 (E)	68	Success
Data set 7	1-9-2022	13:52	350 - 257 (W)	8	Success
Data set 8	29-8-2022	12:25	350 - 262 (W)	7	Success
Data set 9	29-8-2022	13:59	349 - 263 (W)	7	Success
Data set 10	30-8-2022	11:52	16 - 184 (E)	56	Success
Data set 11	30-8-2022	13:25	356 - 240 (W)	14	Success

This section has three subsections, each corresponding to one of the subexperiments. Experiment 1.1 on different VHF antennas can be found in Section 6.2.1. Experiment 1.2 on the effect of the LNA is presented in Section 6.2.2. Experiment 1.3, comparing DopTrackBox and DopTrack, is located in Section 6.2.3.

6.2.1. EXPERIMENT 1.1: DIFFERENT VHF ANTENNAS

The goal of Experiment 1.1 is to "compare the data gathered with DopTrackBox from the Green antenna and the Red-Yellow antenna". This does not tie into a specific research question, but it can be used to gain insight into the differences between the two antennas for further DopTrack operations. An overview of the data that was used can be seen in Table 6.3.

Table 6.3: Data used to achieve the goal of Experiment 1.1.

Experiment 1.1		
Goal	**Data 1 (DTB Station Green)**	**Data 2 (DTB Station Red-Yellow)**
Compare the data gathered with DopTrackBox from the Green antenna and the Red-Yellow antenna.	• Data set 1 - Green 1 • Data set 2 - Green 2 • Data set 3 - Green 3	• Data set 4 - Red-Yellow 1 • Data set 5 - Red-Yellow 2 • Data set 6 - Red-Yellow 3 • Data set 7 - Red-Yellow 4

The results from Experiment 1.1 can be seen in Figure 6.3. The top plot shows the data from the Green antenna and the bottom plot shows the data from the Red-Yellow antenna. Each line represents a different data set.

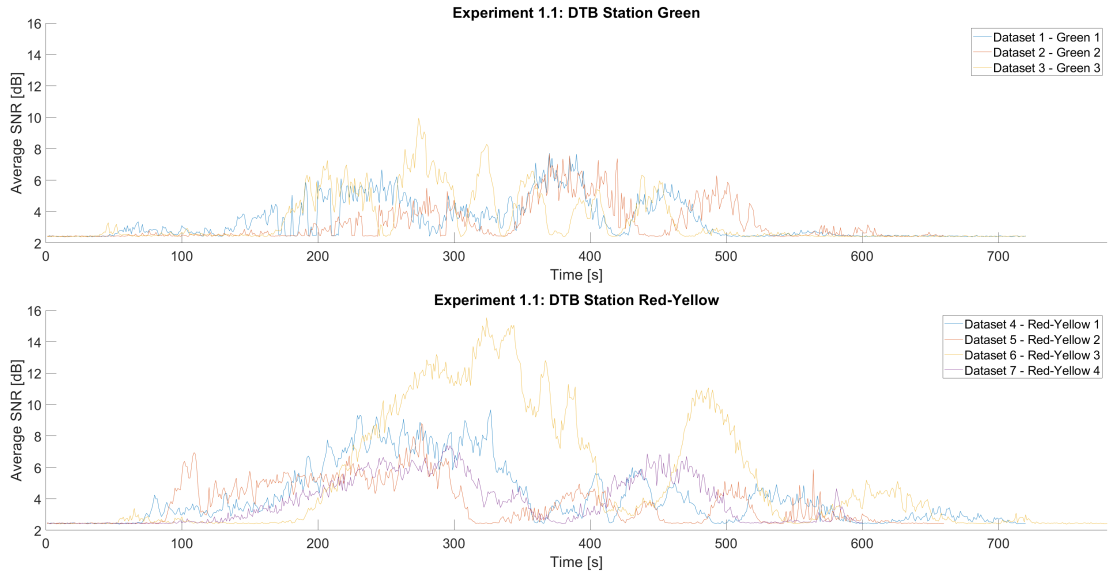


Figure 6.3: Experiment 1.1 SNR. The top plot shows results from the Green antenna and the bottom plot shows results from the Red-Yellow antenna.

What can be observed from Figure 6.3 is the Red-Yellow antenna yielding a higher SNR than the Green antenna. The maximum average SNR for the Green antenna belongs to Data set 3 and is roughly 10 dB, whereas the maximum average SNR for the Red-Yellow antenna is around 15.5 dB and belongs to Data set 6. Cross referencing these passes with Table 6.2 shows that Data set 6 belongs to the highest maximum elevation pass at 68°. The highest maximum elevation pass for the Green antenna is Data set 1 at 46°. The maximum average SNR value for that specific data set is around 8 dB. This means that elevation is not the sole factor determining the obtained SNR.

For the other data sets from the Red-Yellow antenna, it can be observed that the mean of the graph is also higher, which implies that the Red-Yellow antenna yields more consistent results. This observation is backed by the numbers in Table 6.4. Data sets 3 and 4 can be closely compared as they have a similar maximum elevation of 20° and 23° respectively, and they follow a similar path with a starting azimuth of 25° and 24° and an ending azimuth of 160° and 164°. The mean SNR for Data set 3 is 3.418, whereas Data set 4 has a mean SNR of 4.149.

Looking at the noise floor data, which can be found in Appendix A.1 and the second column of Table 6.4, the noise floor for the Red-Yellow antenna data is between 2.6 and 3.2, whereas the Green antenna data's noise floor is between 0.9 and 2.3. Combining this with the higher SNR values, it implies that the signal itself is also stronger as the signal can be obtained by multiplying the noise floor and the SNR. This is also supported by the mean values of the mean SNR. From the third column it can be seen that the values for Data sets 4 through 7, which correspond with the Red-Yellow antenna, have the highest mean SNR; between 3.7 and 5.2 dB. Data sets 1, 2, and 3, corresponding to the Green antenna with the LNA, have the second highest; between 3.2 and 3.4 dB.

From these findings it can be concluded that for receiving data the Red-Yellow antenna is superior to the Green antenna. The locations of the antennas were presented in Figure 2.3. The Green antenna is further away and thus has a slightly longer cable than the Red-Yellow antenna. This could explain slightly lower signal values, but this does not explain why the noise floor is lower. There is therefore most likely a different cause for the disparity between the Green and Red-Yellow antennas.

Table 6.4: Overview of mean values of noise floor and SNR for Experiment 1.1 Lower noise floor values and higher SNR values are better.

Data set Identifier	**DopTrackBox Mean Noise floor [-]**	**DopTrackBox Mean SNR [dB]**
1 - Green LNA on 1	1.718	3.375
2 - Green LNA on 2	2.331	3.216
3 - Green LNA on 3	0.924	3.418
4 - Red-Yellow LNA on 1	3.234	4.149
5 - Red-Yellow LNA on 2	2.869	3.698
6 - Red-Yellow LNA on 3	3.161	5.248
7 - Red-Yellow LNA on 4	2.593	3.822

6.2.2. EXPERIMENT 1.2: EFFECT OF THE LNA

Experiment 1.2's goal is to "compare the data gathered with DopTrackBox with the LNA enabled and the LNA disabled". Just as with Experiment 1.1, there is no specific tie to a research question, but it does provide additional insight into the effect of the LNA on DopTrackBox. An overview of the data for this experiment can be seen in Table 6.5.

Table 6.5: Data used to achieve the goal of Experiment 1.2.

Experiment 1.2		
Goal	**Data 1 (LNA on)**	**Data 2 (LNA off)**
Compare the data gathered with DopTrackBox with the LNA enabled and the LNA disabled.	• Data set 1 - LNA on 1 • Data set 2 - LNA on 2 • Data set 3 - LNA on 3	• Data set 8 - LNA off 1 • Data set 9 - LNA off 2 • Data set 10 - LNA off 3 • Data set 11 - LNA off 4

The results from Experiment 1.2 are shown in Figure 6.4. The top plot shows the data with the LNA on and the bottom plot shows the data with the LNA off.

A clear difference between the top and bottom plots of Figure 6.4 is visible. Data with the LNA on has higher SNR values: peaks to 10 dB for Data set 3 and other maximum values around 7 dB for Data sets 1 and 2, compared to peaks of 6 dB for Data set 11 and other maximum values of 4.5 dB for Data sets 8, 9, and 10. This is to be expected, as the LNA is used to enhance the signal.
Another observation is the lower SNR mean of the plots without LNA, which can be traced back to Table 6.6 where the values are presented. As was stated before, the Green antenna with LNA has mean SNR values between 3.2 and 3.4 dB. Without the LNA, these values are between 2.5 and 3.0 dB. This can be seen in Figure 6.4 by the flatter graphs in the bottom plot.
The effect of elevation could also be more prevalent for the measurements without LNA, as both Data sets 8 and 9 have a maximum elevation of 7°. Data set 11, with the highest peak and mean SNR of the four data sets, has an elevation of 14°; which is lower than that of Data set 10 which has a maximum elevation of 56°.



Figure 6.4: Experiment 1.2 SNR. The top plot shows results from the Green antenna with the LNA on and the bottom plot shows results from the Green antenna with the LNA off.

Data set 10 has the highest maximum elevation of all data sets part of this experiment, but with a mean SNR of 2.7 dB it still lacks behind all data sets with LNA.

The noise floor for the data without the LNA is lower, as can be seen in Table 6.6 and in Appendix A.1. Noise floor values without LNA are between 0.3 and 0.4, where the values with LNA are between 0.9 and 2.3. This is to be expected, as the LNA introduces additional noise into the system.

Table 6.6: Overview of mean values of noise floor and SNR for Experiment 1.2 Lower noise floor values and higher SNR values are better.

Data set Identifier	**DopTrackBox Mean Noise floor[-]**	**DopTrackBox Mean SNR [dB]**
1 - Green LNA on 1	1.718	3.375
2 - Green LNA on 2	2.331	3.216
3 - Green LNA on 3	0.924	3.418
8 - Green LNA off 1	0.347	2.546
9 - Green LNA off 2	0.272	2.641
10 - Green LNA off 3	0.386	2.692
11 - Green LNA off 4	0.361	3.032

From the data of Experiment 1.2 it can be concluded that having the LNA enabled is favourable when looking at SNR. The added noise is greatly offset by the increase in signal strength and thus SNR.

6.2.3. EXPERIMENT 1.3: DOPTRACKBOX AND DOPTRACK

Experiment 1.3 looks at the difference between DopTrackBox and DopTrack, with its goal to "compare the data gathered from DopTrackBox and DopTrack". This ties into Research Question 2.1, the details of which are discussed in full in Section 7.2. The data used for this experiment can be seen in Table 6.7. There was no reference data from DopTrack for Data set 8, so therefore that specific data set could not be used for this experiment. This section is further split into a subsection on SNR data and one on Doppler shift and range rate data.

EXPERIMENT 1.3: SNR

The SNR results for Experiment 1.3 are split up into three different figures. Each figure contains a number of plots. Within each plot there is a direct comparison between DopTrack and DopTrackBox for the same data set, with DopTrackBox data plotted in blue and DopTrack data plotted in orange.

Table 6.7: Data used to achieve the goal of Experiment 1.3.

Experiment 1.3		
Goal	**Data 1 (DopTrackBox)**	**Data 2 (DopTrack)**
Compare the data gathered from DopTrackBox and DopTrack.	• Data set 1 - Green LNA on 1 • Data set 2 - Green LNA on 2 • Data set 3 - Green LNA on 3 • Data set 4 - Red-Yellow LNA on 1 • Data set 5 - Red-Yellow LNA on 2 • Data set 6 - Red-Yellow LNA on 3 • Data set 7 - Red-Yellow LNA on 4 • Data set 9 - Green LNA off 2 • Data set 10 - Green LNA off 3 • Data set 11 - Green LNA off 4	• Reference DopTrack 1 • Reference DopTrack 2 • Reference DopTrack 3 • Reference DopTrack 4 • Reference DopTrack 5 • Reference DopTrack 6 • Reference DopTrack 7 • Reference DopTrack 9 • Reference DopTrack 10 • Reference DopTrack 11

Figure 6.5 shows the SNR results for the Green antenna, Figure 6.6 shows the SNR results for the Red-Yellow antenna, and Figure 6.7 shows the SNR results with the LNA off.

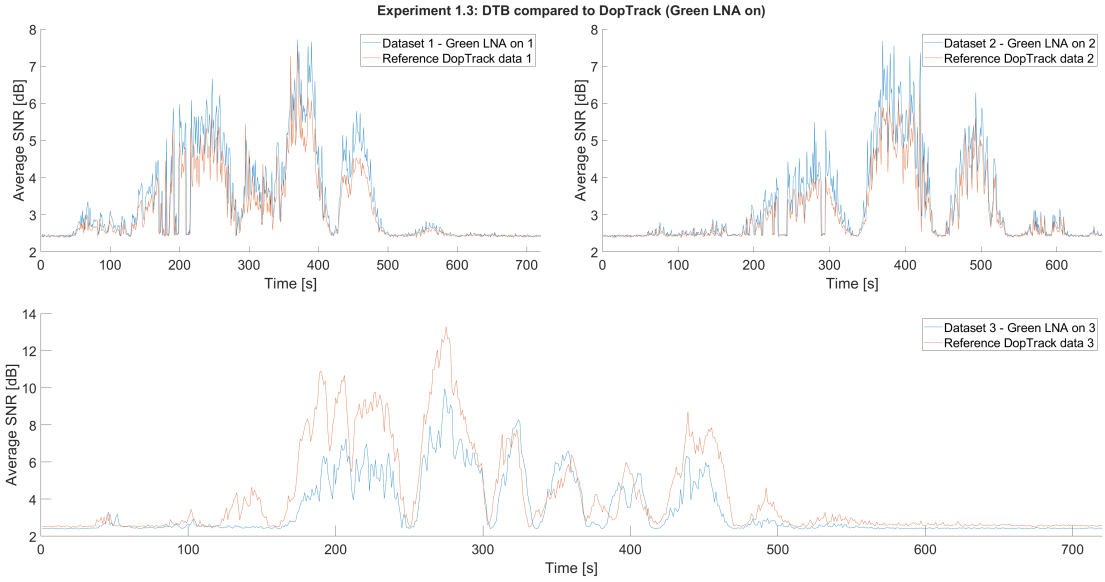


Figure 6.5: Experiment 1.3 SNR comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA on (Data sets 1, 2, and 3). Each plot represents its own data set with the corresponding DopTrack reference.

From the plots in Figure 6.5, Data set 3 stands out. Data sets 1 and 2 have a generally higher SNR for DopTrackBox compared to DopTrack. The mean over the entire measurement for Data set 1 is 0.250 dB higher for DopTrackBox and for Data set 2 the mean is 0.214 dB higher. For Data set 3 the opposite is true, with DopTrackBox' mean SNR being 0.777 dB lower compared to that of DopTrack. Information from DopTrack's yaml files showed DopTrack using the Green antenna for Data sets 1 and 2, and the Red-Yellow antenna for Data set 3. This would explain the change in behaviour seen in the graphs. Data sets 1 and 2 were recorded in September 2022 along with the other data, but Data set 3 was recorded later in October 2022. During this time DopTrack switched its default antenna from Green to Red-Yellow. These numbers are also summarised in Table 6.9. Additionally, it can be seen that both DopTrackBox and DopTrack follow the same pattern with rising and decreasing SNR. Data set 2 for example shows an increase in SNR around the 200, 340, and 450 second mark and a decrease around the 290, 400, and 500 second mark. These patterns are visible in eastern and western passes, and with both antennas.

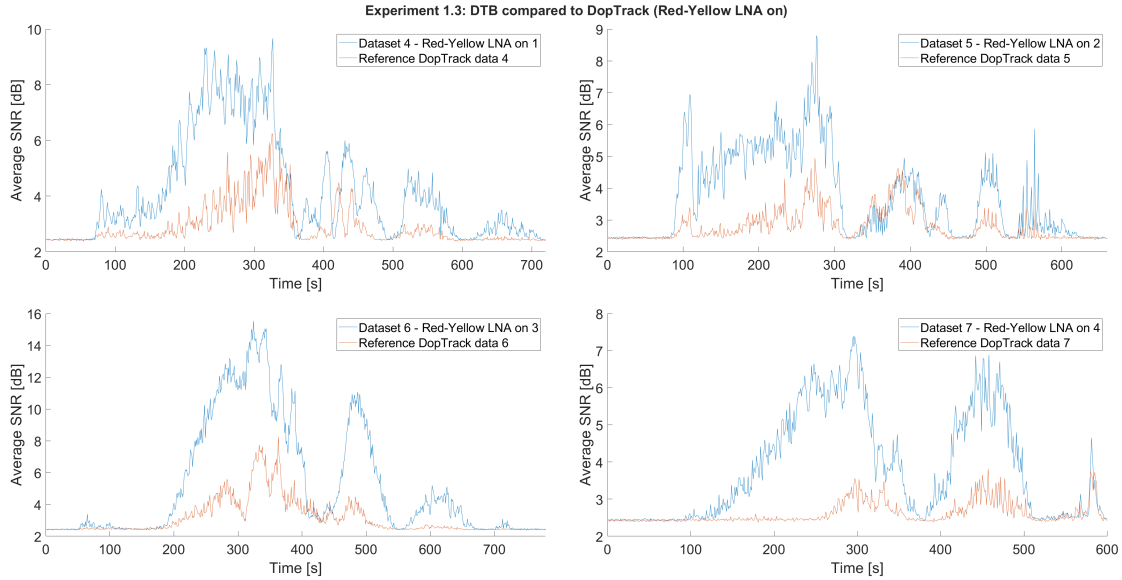


Figure 6.6: Experiment 1.3 SNR comparison for DopTrackBox (blue) and DopTrack (orange) for Red-Yellow LNA on (Data sets 4, 5, 6, and 7). Each plot represents its own data set with the corresponding DopTrack reference.

Figure 6.6 shows a consistent behaviour. DopTrackBox shows higher maximum and mean SNR values than DopTrack. Maximum values are 9.7 compared to 6.2 dB for Data set 4, 8.7 compared to 4.9 dB for Data set 5, 15.5 compared to 8.3 dB for Data set 6, and 7.4 compared to 3.8 dB for Data set 7. The mean values are also consistently higher, with the smallest mean difference for Data set 5 at 0.907 dB and the largest difference at 2.124 dB for Data set 6. The difference between DopTrackBox and DopTrack for Data set 6 is the largest difference for all data sets in Experiment 1. These findings fall in line with the finding from Data set 3; the Red-Yellow antenna yields higher mean and maximum SNR values compared to the Green antenna. Data set 6 also has the highest maximum elevation of all passes at 68°, which could explain the high SNR values.

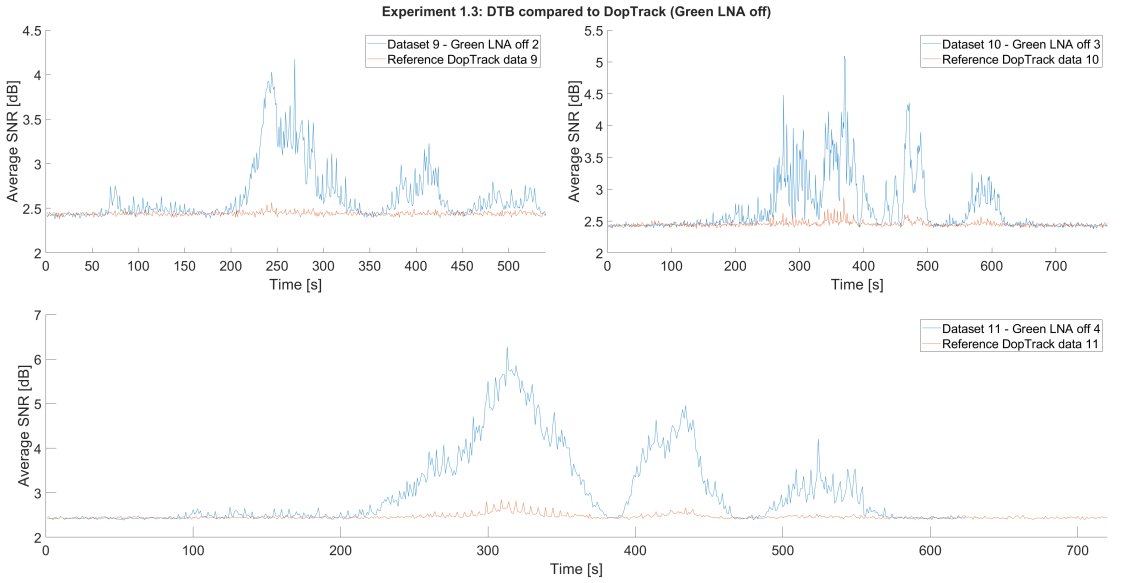


Figure 6.7: Experiment 1.3 SNR comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA off (Data sets 9, 10, and 11). Each plot represents its own data set with the corresponding DopTrack reference.

DopTrack's data in Figure 6.7 is more or less a flat line, whereas peaks are visible in DopTrackBox' data. The difference between DopTrackBox' and DopTrack's mean SNR values for these measurements, as also visible in Table 6.9, are in the same order of magnitude as those of the Green antenna with LNA. These differences are between 0.202 and 0.572 dB in DopTrackBox' favour. The mean values for DopTrack are all very close to the 2.4 dB, being 2.439, 2.451, and 2.460 dB, which is the value of the SNR when no actual signal is present due to the way the signal and noise are processed. This implies that DopTrackBox is capable of receiving signal data, whereas DopTrack has more noise.

Something not visible in Figures 6.5, 6.6, and 6.7 are the noise floor values. These are obtained from the plots in Appendix A.1 and are presented in Table 6.8. The mean SNR values mentioned in this section have all been tabulated in Table 6.9 to provide a clear overview. The second column shows which antenna DopTrack used to gather the data, fourth column shows the DopTrack reference data, and the fifth column shows the difference between DopTrackBox and DopTrack.

Table 6.8: Overview of mean values of noise floor for Experiment 1. Lower values are better.

Data set Identifier	**DopTrackBox Mean Noise floor [-]**	**DopTrack Mean Noise floor [-]**	**DopTrackBox and DopTrack difference [-]**	**Relative difference [-]**
1	1.718	0.034	1.683	50.0
2	2.331	0.045	2.286	51.9
3	0.924	0.038	0.885	24.0
4	3.234	0.091	3.143	35.6
5	2.869	0.088	2.781	32.6
6	3.161	0.057	3.104	55.8
7	2.593	0.086	2.507	30.1
8	0.347	No data	No data	No data
9	0.272	0.011	0.261	25.6
10	0.386	0.011	0.375	35.6
11	0.361	0.011	0.350	33.6

Table 6.9: Overview of mean values of the mean SNR for Experiment 1. Higher values are better.

Data set Identifier	**DopTrack Antenna**	**DopTrackBox Mean SNR [dB]**	**DopTrack Mean SNR [dB]**	**DopTrackBox and DopTrack difference [dB]**
1 - Green LNA on 1	G	3.375	3.125	0.250
2 - Green LNA on 2	G	3.216	3.002	0.214
3 - Green LNA on 3	RY	3.418	4.195	-0.777
4 - Red-Yellow LNA on 1	G	4.149	2.904	1.245
5 - Red-Yellow LNA on 2	G	3.698	2.791	0.907
6 - Red-Yellow LNA on 3	G	5.248	3.124	2.124
7 - Red-Yellow LNA on 4	G	3.822	2.616	1.206
8 - Green LNA off 1	G	2.546	No data	No data
9 - Green LNA off 2	G	2.641	2.439	0.202
10 - Green LNA off 3	G	2.692	2.451	0.241
11 - Green LNA off 4	G	3.032	2.460	0.572

A few observations can be made regarding the different antenna setups when comparing them to DopTrack. When looking at the noise floor in Table 6.8, DopTrack has a lower noise floor in all data sets. For Data set 3 the relative difference is the smallest as DopTrackBox' noise floor is only 24.0 times higher than DopTrack's one, whereas the greatest relative difference is 55.8 times with Data set 6. The relative difference is obtained by dividing the noise floor value of DopTrackBox by the value of DopTrack.
Even though the noise floor for DopTrack is lower across all data sets, the obtained mean SNR is slightly higher for DopTrackBox as can be seen in the final column of Table 6.9. These values vary between 0.202 dB for Data set 9 to 2.124 dB for Data set 6. The only exception is Data set 3. Here DopTrack outperforms DopTrackBox regarding mean SNR, with a 0.777 dB deficit. The most likely cause is the difference in antenna,

as DopTrackBox uses the Green antenna whereas DopTrack uses the Red-Yellow one. This is in line with the findings where DopTrackBox uses the Red-Yellow antenna, and DopTrack uses the Green antenna.

When inspecting DopTrack's yaml files it was found that the ground station coordinates were different for measurements made before 31 August 2022. This only impacts the measurements with the LNA off. These use the coordinates of the old ground station location, which was located at the EWI building at the TU Delft campus. EWI and the LR building, where the ground station is now, are approximately 1 kilometre apart so this should not majorly impact DopTrack's data as the location is mainly used to help schedule the satellite passes. With this 1 kilometre difference, the 60 additional seconds at the front and back of the recordings are more than sufficient to account for the different coordinates.

The minimum margin is generally set at 3 dB for the link to close [25, 26]. For proper data transfer, the margin should be exceeding 3 dB at all times. At first glance, all the DopTrackBox measurements with the LNA on meet this requirement, having a minimum mean SNR of 3.216 dB. However, it should be noted that the mean over the entire measurement as seen in Table 6.9 does not imply the SNR is at least 3 dB at any time. The mean also includes the parts of the data where there is no signal yet. This is due to the way the recorder has been set up, to take an additional minute before and after the pass to ensure the entire pass is captured. Looking at the SNR plots of Figures 6.5, 6.6, and 6.7 the main parts of the recordings do exceed this threshold. The elevation effect is also visible, as passes with a higher elevation show higher SNR values than those with a lower elevation. This effect seems to be the greatest in Figure 6.6 for the Red-Yellow antenna, where Data set 6 has the highest elevation and SNR, followed in order by Data sets 4, 5, and 7.

A feature that was discussed with Figure 6.5 was the pattern of increasing and decreasing SNR seen in both DopTrackBox' and DopTrack's data. This pattern is also visible in Figures 6.6 and 6.7. A decrease in SNR happens around the 300 to 400 second time range. This is both visible in satellite passes from the east and west. The noise floor data does not show a spike in noise levels around the same time, so the signal strength is lower whilst the noise is not higher. The exact reasoning for this is unknown. It could be interference caused by something, or an innate feature of the satellite itself as the measurements occurred during different times at different days.

EXPERIMENT 1.3: DOPPLER SHIFT

Next to the SNR data for Experiment 1.3, there is also the Doppler shift data. This data has been processed by the DopTrack Processing Package (DTPP) which uses a Fast Fourier Transform on the raw IQ data. From the data, the signal is extracted which is comprised of several data points. Generally speaking, the more data points the better the quality of the signal as it could be better extracted by the software. An overview of the data points extracted from both the DopTrackBox and the DopTrack reference data can be seen in Table 6.10.

The column with maximum data points in set shows the number of data points that are in the data set that could contain data. Since the bin size, dt, is 0.5 seconds for the DTPP, the number of bins or data points is equal to twice the length of the pass of the recording. 60 seconds are added to the start and end of the recording as a buffer, so these 120 seconds have to be subtracted from the length of the pass to obtain the time in which data could be present. This time is then divided by dt to find the maximum number of data points in the data set. The quality value is the percentage of the number of data points in a data set with respect to the maximum.

For Data set 9 the DTPP could not find any data points in either the DopTrackBox, nor the reference data. For Data set 11, no data points could be extracted by the DTPP from the DopTrackBox data. Therefore, these data sets have been left out of the Doppler shift and range rate data along with Data set 8 where there is no DopTrack reference data.

When comparing the data quality for DopTrackBox and DopTrack as seen in Table 6.10, a few notable observations can be made. The DopTrackBox Green antenna with LNA show small differences between the two systems from -1.4% for Data set 3 to 1.0% for Data set 1, where a negative percentage indicates DopTrackBox having a lower data quality than DopTrack.
The Red-Yellow antenna shows DopTrackBox having a favourable data quality, varying from 4.4% for Data set 6 to 11.3% for Data set 7. These differences are larger than the difference seen for Data set 3, where DopTrack uses the Red-Yellow antenna and DopTrackBox the Green antenna. This implies that the effect of using the Red-Yellow antenna is greater for DopTrackBox than for DopTrack.

Table 6.10: Frequency data availability of DopTrackBox and DopTrack for Experiment 1.3.

Experiment 1.3					
Identifier	**Maximum Data points in set**	**DopTrackBox**		**DopTrack**	
		Data points	**Quality**	**Data points**	**Quality**
1 - Green LNA on 1	1200	295	24.6%	283	23.6%
2 - Green LNA on 2	1080	263	24.4%	270	25.0%
3 - Green LNA on 3	1200	325	27.1%	342	28.5%
4 - Red-Yellow LNA on 1	1200	381	31.8%	321	26.8%
5 - Red-Yellow LNA on 2	1080	330	30.6%	273	25.3%
6 - Red-Yellow LNA on 3	1320	355	26.9%	297	22.5%
7 - Red-Yellow LNA on 4	960	320	33.3%	212	22.1%
8 - Green LNA off 1	840	0	0.0%	No recording	
9 - Green LNA off 2	840	0	0.0%	0	0.0%
10 - Green LNA off 3	1320	300	22.7%	122	9.2%
11 - Green LNA off 4	1200	0	0.0%	113	9.4%

Of the four data sets with the LNA off, only Data set 10 could be used. This could point to the signal being too weak for the DTPP to extract or there being no actual Doppler shift in the measured signal. Data set 10 does show a significant increase in data quality over DopTrack, with a 13.5% increase. However, since it is the only measurement without the LNA no further conclusions can be drawn as it could be an anomaly. From this it can be concluded that the LNA is beneficial in increasing the data quality for Doppler tracking.

Table 6.10 suggests there is no strong connection between data quality and maximum elevation of the pass. Data sets 1 and 6 have the highest elevation of their category at 46° and 68° respectively, but do not have the highest data quality. These would be Data set 3, with a 20° maximum elevation, and Data set 7 with only a 8° elevation.

There are two different plots for the Doppler shift data. One shows the change of frequency over time for both the DopTrackBox and DopTrack reference data. The other shows the range rate difference of both DopTrackBox and the reference, along with the Root Mean Square (RMS) of that data. The range rate difference is obtained by subtracting the range rate determined by the TLE from the actual measured range rate. The RMS is used to quantify the difference between DopTrackBox and DopTrack. Since the plots with the range rate difference are more interesting to draw conclusions from, only one frequency plot is shown here in this chapter. The other frequency plots can be found in Appendix A.2.

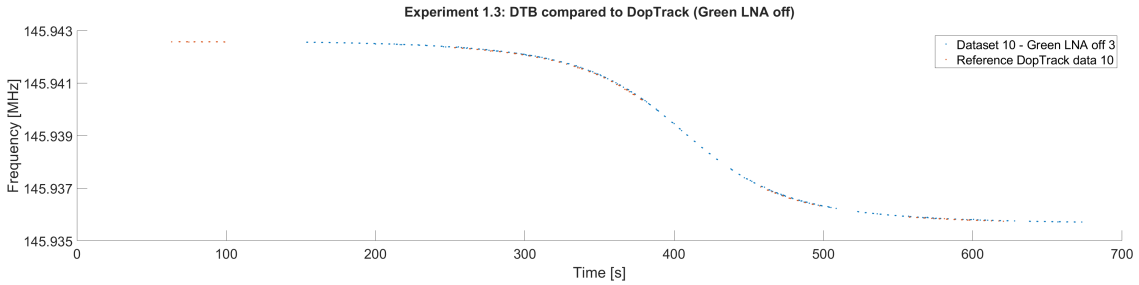


Figure 6.8: Experiment 1.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA off (Data set 10).

Figure 6.8 shows how the frequency changes over time for Data set 10. Both DopTrackBox' and DopTrack's data follow the same pattern, but the S-shape is more clearly distinguishable from DopTrackBox' data. The shape of the curve can be used to identify differences in linear drift.

There are three figures with range rate difference data. Each plot shows a direct comparison between DopTrackBox' and DopTrack's range rate difference for the same data set. The DopTrackBox data is plotted in blue and the DopTrack data is plotted in orange. Next to the data, a line with the RMS value for the range rate difference has been plotted in each plot. This provides a simple metric to analyse overall performance. It should be noted that these data sets are not continuous. The plots only show a data point when there is actual data present, which can result in slight differences between the DopTrack and DopTrackBox data. Figure 6.9 shows the range rate difference for the Green antenna, Figure 6.10 shows the range rate difference for the Red-Yellow antenna, and Figure 6.11 shows the range rate difference with the LNA off.

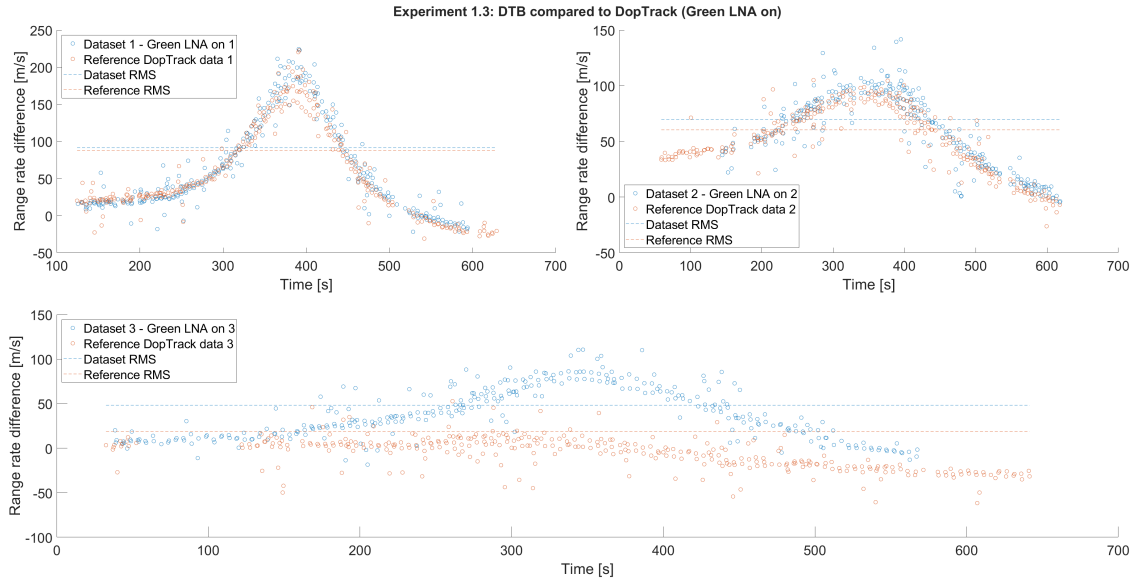


Figure 6.9: Experiment 1.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA on (Data sets 1, 2, and 3). Each plot represents its own data set with the corresponding DopTrack reference.

Just as with the SNR data, in Figure 6.9, Data set 3 catches the eye. The DopTrackBox data sets all follow a similar pattern, but the DopTrack reference for Data set 3 is less bell-shaped. This bell shape curve implies that the difference is small at first, increases towards the middle of the data set, and decreases again towards the end. This could be attributed to the fact that the satellite's relative velocity with respect to the ground station increases as the satellite gets closer to the ground station and decreases as the satellite gets further away again. Next to the shape, other interesting factors within the plots are the absolute maximum range rate difference, the time when that maximum occurs, and the RMS of the range rate.
The RMS range rate difference is the largest for Data set 3 at 29.7 m/s and the smallest for Data set 1 at 3.7 m/s. This makes sense when looking at the plots. The time of when the maximum range rate difference occurs is around the same for Data set 1 at 391.0 and 391.5 seconds, for Data set 2 they slightly differ which can be seen by the additional dots above the curve. The timings are 395.0 and 272.5 seconds. For Data set 3 they are vastly different, at 347.0 seconds for DopTrackBox and 607.0 seconds for DopTrack. Something that could explain this, is again the different antenna that is used for DopTrack's data in Data set 3. The RMS values are all summarised in Table 6.11 and the times of the maximum range rate difference errors can be seen in Table 6.12.

There is more consistency regarding the shapes between DopTrackBox and DopTrack in Figure 6.10. What can be observed here is that the DopTrackBox range rate differences are lower than those of DopTrack, which is also reflected in the RMS value of the data. The RMS range rate difference is the largest for Data set 6 at -39.4 m/s and the smallest for Data set 7 at -11.8 m/s. Negative values indicate DopTrackBox' RMS range rate difference being smaller than that of DopTrack. Another interesting observation is that the absolute error is the largest for Data set 6 at 230.9 m/s, whereas it is the smallest for Data set 5 at 72.7 m/s. This could be an effect of the elevation of the pass, since Data set 6 has a maximum elevation of 68° and Data set 5 has a maximum elevation of 11°. In that sense, the expectation would be that Data set 7, with a maximum elevation of 7° would have an even smaller range rate error, but this is not the case at 113.5 m/s.

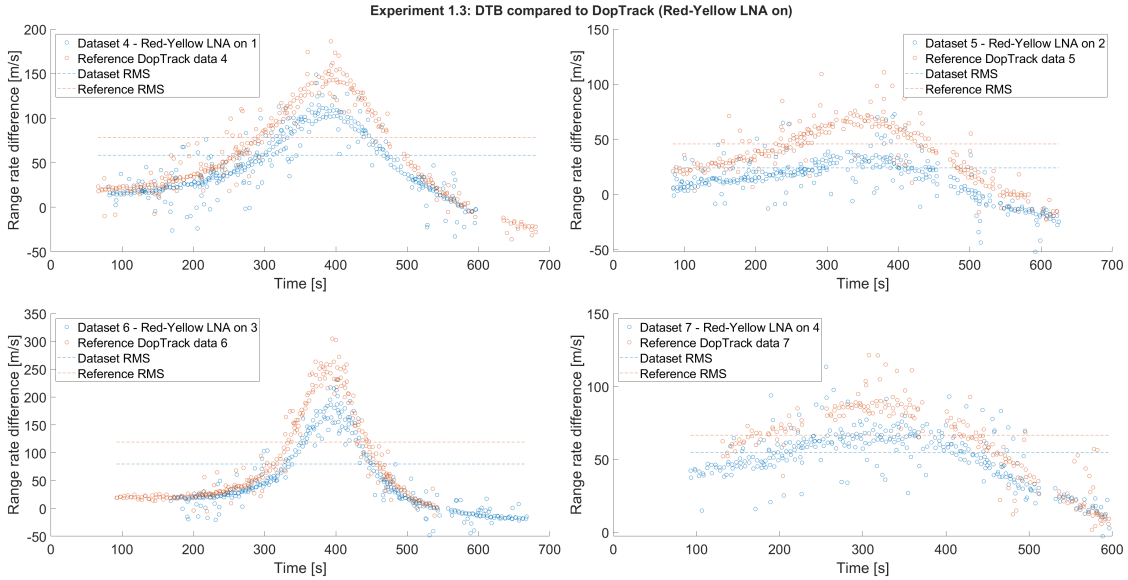


Figure 6.10: Experiment 1.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for Red-Yellow LNA on (Data sets 4, 5, 6, and 7). Each plot represents its own data set with the corresponding DopTrack reference.

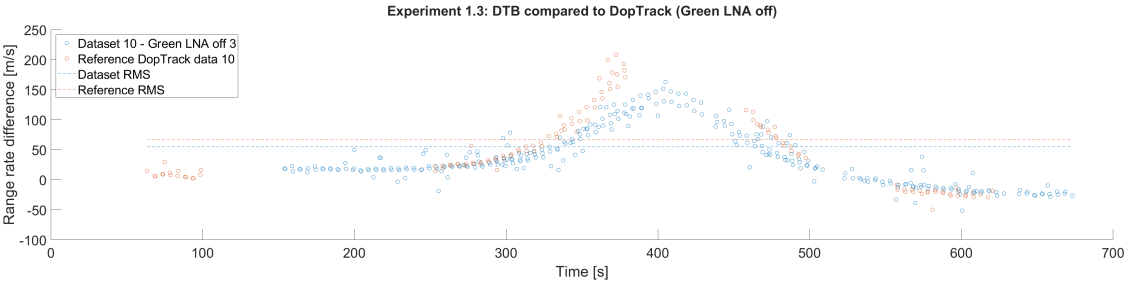


Figure 6.11: Experiment 1.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA off (Data set 10).

Figure 6.11 shows the sole data set without LNA. The reduction in available data points for DopTrack is clearly visible, at only 122 data points for DopTrack against 300 for DopTrackBox. The peak of the bell curve is not visible in DopTrack's data. The RMS range rate difference between DopTrackBox and DopTrack is -17.3 m/s, suggesting DopTrackBox having a range rate closer to the theoretical orbit.

The values of the range rate difference RMS from Figures 6.9, 6.10, and 6.11 are all summarised in Table 6.11. The second column shows the RMS value of the range rate difference for DopTrackBox and the third column shows the RMS value of the range rate difference for the DopTrack reference data. These are all RMS values of the difference between the measured data and the TLE. The fourth column shows the difference between the second and third column, thus comparing the range rate difference of DopTrackBox with that of DopTrack. This difference is therefore a comparison of the two systems. A value below 0 means DopTrackBox has a lower RMS range rate difference than DopTrack. DopTrack uses the Green antenna for all data sets, except Data set 3. The LNA setting is the same between DopTrackBox and DopTrack.

Table 6.11: RMS of range rate difference for DopTrackBox and DopTrack reference data for Experiment 1.3. Lower values are better.

Data set Identifier	DopTrackBox Range rate difference RMS [m/s]	DopTrack Range rate difference RMS [m/s]	DopTrackBox and DopTrack difference [m/s]
1	91.43	87.70	3.726
2	69.57	60.36	9.210
3	48.28	18.57	29.711
4	58.39	78.14	-19.749
5	23.97	45.76	-21.794
6	79.73	119.13	-39.398
7	54.72	66.53	-11.806
10	58.95	76.25	-17.297

From Table 6.11 it can be seen that Data sets 1, 2, and 3 have a positive DopTrackBox and DopTrack difference, whereas the opposite is true for the other data sets. Data sets 1, 2, and 3 correspond to the measurements from the Green antenna with the LNA enabled, with the DopTrack reference data using the Green antenna except for Data set 3 as could be seen in Table 6.9. This would suggest that when using the LNA DopTrackBox' range rate measurements are inferior to those of DopTrack when using the Green antenna for both systems, as the differences are 3.7 and 9.2 m/s. Data set 3 sees the difference between DopTrackBox and DopTrack increase to 29.7 m/s. For the data sets where DopTrackBox uses the Red-Yellow antenna; 4, 5, 6, and 7, the range rate difference for DopTrackBox is lower than for the reference data; anywhere between -11.8 and -39.4 m/s. The same is true for Data set 10, which is from the Green antenna without the LNA and has a difference between the systems of -17.3 m/s.

This is in line with an observation from the SNR measurements, where the Red-Yellow antenna yielded higher SNRs than the Green antenna. The range rate difference findings solidify this theory, as DopTrackBox has lower range rate differences when using the Red-Yellow antenna and the difference increases between systems when DopTrack uses the Red-Yellow antenna where DopTrackBox uses the Green antenna.
Cross-referencing Table 6.11 with the Figures 6.9, 6.10, and 6.11 shows that a lower range rate RMS results in a flatter data set, as can be seen in DopTrack data 3 and Data set 5.

The maximum values of the range rate difference plots can be seen in Table 6.12. Here the time at which the maximum occurs, along with its value is presented for both the DopTrackBox and DopTrack reference. These time values can be compared against the TCA to see whether the highest range rate difference coincides with the TCA. Range Rate Difference has been abbreviated to RRD in this table.

Table 6.12: Overview of maximum range rate difference (RRD) and the time at which this maximum occurs for Experiment 1.3.

Data set Identifier	DopTrackBox Time of max RRD [s]	DopTrackBox Value of max RRD [m/s]	DopTrack Time of max RRD [s]	DopTrack Value of max RRD [m/s]
1	391.0	224.35	391.5	222.74
2	395.0	141.62	272.5	105.02
3	347.0	110.53	607.0	-61.64
4	372.5	148.83	393.0	186.17
5	394.5	72.67	380.0	110.68
6	402.0	230.92	395.0	304.84
7	256.0	113.47	307.5	121.38
10	405.0	162.65	372.5	208.06

The data points are not spread out equally, which could result in the absolute maximum not being recorded for either the DopTrackBox or DopTrack data sets. Table 6.12 shows that the maximum values and RMS values follow the same trend: when their absolute value is higher, the RMS is also higher. The time of when the maximum range rate difference occurs is within 10% of each other for most data sets, but not for Data sets 2, 3, and 7. Looking back at their respective plots, it can be observed that Data set 3 shows a flatter data set for DopTrack, resulting in its maximum not being at the same point in time. Data sets 2 and 7 shows more

scattering around their peak values, resulting in the observed difference.

These times of when the maximum range rate difference occurs can be compared against the time of closest approach (TCA), which can be found in Table 6.13. This table shows when the TCA is within each data set and shows the ΔTCA, which is the difference between the measured TCA and the TCA according to the TLE. A negative number indicates the TCA of the measurement occurring before the TLE TCA. The final value indicated is the slope of the linear fit of the first residual, which grants information on the frequency drift of the system.

Table 6.13: Overview of the TCA, the difference of the TCA between the TLE and measurement, and the slope of the linear fit of the first residual of DopTrackBox and DopTrack for Experiment 1.3.

Data set Identifier	**DopTrackBox**			**DopTrack**		
	TCA [s]	**ΔTCA [s]**	**Slope first residual [m/s²]**	**TCA [s]**	**ΔTCA [s]**	**Slope first residual [m/s²]**
1	383.0	-2.08	0.035	383.0	-1.95	-0.011
2	358.0	-2.29	-0.152	358.0	-2.10	-0.075
3	355.0	-1.60	0.032	354.0	-0.04	-0.073
4	394.0	-1.95	0.035	393.0	-2.63	-0.002
5	372.0	-0.87	-0.052	371.0	-1.90	-0.059
6	391.0	-1.66	-0.111	390.0	-2.29	0.221
7	342.0	-2.15	-0.053	341.0	-2.78	-0.136
10	406.0	-1.39	-0.092	406.0	-2.17	-0.054

All the ΔTCA values for both DopTrackBox and DopTrack are negative, meaning that the measured TCA occurred before the predicted TLE TCA. There are also some minor differences between TCA values for DopTrackBox and DopTrack for Data sets 3, 4, 5, 6, and 7. It should be noted that the TCA values are rounded, since the information used to obtain the TCA is rounded to the nearest second. The ΔTCA values are not rounded and contain the actual difference between the TLE TCA and the measured TCA.
The ΔTCA values are the smallest for Data set 5 for DopTrackBox and DopTrack data 3, at -0.87s and -0.04s. These data sets also have the smallest absolute maximum range rate difference and the smallest RMS range rate difference.
When comparing the slope of the linear fit of the first residual for DopTrackBox and DopTrack, it can be seen that the slopes for DopTrackBox and DopTrack are generally in the same order of magnitude. This suggests that the linear drift for the two systems is comparable.

6.3. EXPERIMENT 2: DOPTRACKBOX CONFIGURATION CHANGES

The second set of experiments is about the different DopTrackBox configurations as presented in Table 2.3; DopTrackBox, DopTrackBox Pro, and DopTrackBox Light. The hardware configuration for each of these experiments can be seen in Table 6.14, where DopTrackBox and DopTrackBox Pro use the RSPduo, DopTrackBox Light uses the RSP1A, and only DopTrackBox Pro uses the GPS receiver. All of these experiments use the manually pointed Yagi antenna following the strategy as described in Section 4.4. During the data acquisition it was found out that Nayif's signal does not seem to be polarised, so the orientation of the antenna does not matter. When holding the antenna vertically it was observed that the signal's power was lower in the spectrogram. A cause for this can be the fact that holding the Yagi antenna vertically requires a better grip on the antenna which could cause more interference when comparing it to the horizontal position. When held horizontally the antenna can rest on the hand holding the antenna, resulting in a decreased contact area with the wielder. All experiments in this set use both the SNR and Doppler shift data.

Table 6.14: DopTrackBox hardware and software used for data acquisition during Experiment 2.

Hardware					***Software***	
Computer	**SDR**	**GPS Clock**	**Antenna**	**Storage**	**OS**	**Recorder**
DTB-VM	RSPduo RSP1A	Yes for some	Yagi	Internal	Ubuntu 22.04 LTS	rx_tools

These experiments are also conducted at different locations. The majority of them were conducted at the field location, but some of them were conducted at the roof of the Aerospace Engineering building at the TU Delft. As they had to be manually pointed and there are more buildings obstructing the line of sight to the satellite at the field location, the minimum elevation requirement for the satellite in the scheduler software was increased from 5° to 30°. This resulted in generally one measurement possibility on each day. Therefore the data acquisition for these experiments took longer than those for the first set, starting on 14 October 2022 and ending on 16 November 2022. These measurements also had more things that could go wrong compared to the measurements at the DopTrack ground station, resulting in various crashes and errors. Since line of sight from the measurement point is not equal in all directions, these details have been noted as well. A full overview can be seen in Table 6.15. If the measurement crashed, a reason for the crash has been provided. For measurements Fail_12.1, Data set 12, and Data set 13 no azimuth at maximum elevation was written down at the time of the measurement, so therefore no value is given there. Horizontal lines below the data sets are used to separate the data sets into the three subsets: DTB, DTB Pro, and DTB Light.

Table 6.15: Overview table of all measurements for Experiment 2. Azimuth column shows the azimuth of the start of recording, azimuth at maximum elevation, and azimuth at the end of the recording, along with whether the pass was to the east or west of the ground station.

Experiment 2 Measurements					
Identifier	**Date**	**Start - Max - End Azimuth [°]**	**Maximum Elevation [°]**	**Weather**	**Status**
Fail_12.1	14-10-2022	5 - ... - 217 (W)	33	Sunny	Failure
	Host machine went to sleep, prematurely ending the recording.				
Data set 12	18-10-2022	15 - ... - 188 (E)	65	Sunny	Success
Data set 13	18-10-2022	355 - ... - 245 (W)	12	Sunny	Success
Data set 14	25-10-2022	12 - 274 - 199 (W)	76	Partly cloudy Partial Solar Eclipse	Success
Data set 15	26-10-2022	16 - 100 - 186 (E)	59	Cloudy	Success
Data set 16	28-10-2022	12 - 274 - 199 (W)	76	Partly cloudy	Success
Fail_17.1	30-10-2022	8 - 285 - 212 (W)	42	Partly cloudy	Failure
	System ran out of RAM. VM RAM increased from 2 GB to 3 GB.				
Data set 17	2-11-2022	10 - 296 - 203 (E)	64	Sunny	Success
Data set 18	3-11-2022	18 - 99 - 180 (E)	46	Cloudy	Success
Data set 19	4-11-2022	7 - 293 - 215 (W)	37	Cloudy	Success
Data set 20	14-11-2022	24 - 95 - 162 (E)	23	Sunny	Success
Data set 21	7-11-2022	10 - 286 - 204 (W)	49	Light rain	Success
Fail_22.1	8-11-2022	17 - 105 - 183 (E)	51	Cloudy	Failure
	Linux automatic suspend. VM CPU increased from 1 to 2 threads.				
Data set 22	10-11-2022	13 - 190 - 195 (-)	86	Light clouds	Success
Data set 23	11-11-2022	21 - 98 - 173 (E)	33	Cloudy	Success
Fail_24.1	12-11-2022	9 - 286 - 207 (W)	52	Light clouds	Failure
	xHCI host not responding.				
Fail_24.2	13-11-2022	17 - 105 - 184 (E)	55	Sunny	Failure
	xHCI host not responding.				
Data set 24	15-11-2022	13 - 318 - 196 (-)	86	Sunny	Success
Data set 25	16-11-2022	21 - 100 - 173 (E)	33	Sunny	Success

A total of five attempted data acquisitions failed. A short explanation per case is provided here. Fail_12.1 was caused by the host machine going to sleep due to inactivity on the system. To save battery, the system automatically goes into sleep mode. This was not anticipated as this was the first recording with the hand held antenna, and thus the first time this could occur.
Fail_17.1 occurred due to the VM running out of RAM. The exact reason for this is unknown as no other tasks were performed on the VM or host machine at the time. To mitigate the issue, the amount of RAM was increased to 3 GB.
Fail_22.1 was caused by an OS error, with the exact reason for this being unknown.

Fail_24.1 and Fail_24.2 have the same cause, namely the xHCI host not responding. This means that the system can no longer communicate with the USB device, being the SDR. To mitigate this issue, it was ensured that the host machine was shut down and rebooted before the measurements took place.

This section is further divided into three subsections, corresponding with the subexperiments. Section 6.3.1 contains Experiment 2.1 on the effect of a GPS clock. Section 6.3.2 revolves around Experiment 2.2 which looks at the effect of using a different SDR. Section 6.3.3 is where Experiment 2.3 is explained and looks into the effect of manually pointing the antenna.

6.3.1. EXPERIMENT 2.1: EFFECT OF THE GPS CLOCK

Experiment 2.1 looks at the effect of including or excluding a GPS, so effectively it compares the DTB and DTB Pro variants of DopTrackBox. This is directly linked to Research Question 2.2, which will be discussed in Section 7.3. This section is further split into a subsection on SNR data and a subsection on Doppler shift data. An overview of the data used for this experiment can be seen in Table 6.16.

Table 6.16: Experiment 2.1 goal and data overview.

Experiment 2.1		
Goal	**Data 1 (DTB)**	**Data 2 (DTB Pro)**
Compare the data gathered with the DTB and DTB Pro configurations to see the effect of a GPS on SNR and Doppler shift data.	• Data set 12 - DTB 1 • Data set 13 - DTB 2 • Data set 14 - DTB 3	• Data set 15 - DTB Pro 1 • Data set 16 - DTB Pro 2 • Data set 17 - DTB Pro 3 • Data set 18 - DTB Pro 4 • Data set 19 - DTB Pro 5 • Data set 20 - DTB Pro 6

Of the data sets mentioned in Table 6.16, Data sets 12, 13, and 20 were obtained on the roof of LR at the DopTrack ground station, and the others at the field location. Due to this, the elevation plays a greater part compared to Experiment 1 as there are more obstacles blocking line of sight at the field location.

EXPERIMENT 2.1: SNR

The SNR results from Experiment 2.1 can be seen in Figure 6.12. The top plot shows the results for DTB and the bottom plot for DTB Pro. Each line represents a different data set.

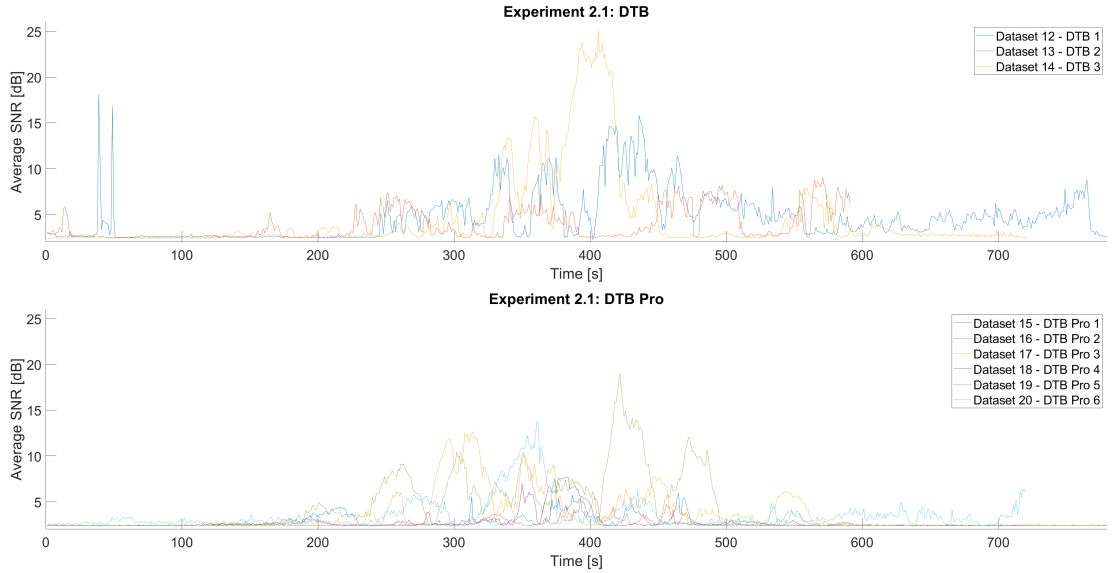


Figure 6.12: Experiment 2.1 SNR. The top plot shows results from DTB (without GPS) and the bottom plot shows results from DTB Pro (with GPS).

Comparing the DTB and DTB Pro data from Figure 6.12 results in the following findings. First the number of data sets should be addressed, since the DTB Pro data contains double the number of data sets. This could lead to biased results due to favourable data from one part of the experiment being compared to more average data from the other part. This will be discussed in more detail in Section 7.1. Elevation and azimuth values also play a role as eastern passes have more obstacles obscuring line of sight than western passes. This was shown in Figure 4.3.

Data set 14 has the highest SNR peak at 25 dB and second highest mean SNR at 4.6 dB. This is a western pass with the highest maximum elevation of 76°. Data set 16, which is also a western pass at the same elevation as Data set 14, but with the DTB Pro hardware shows significantly lower SNR values. Here the peak SNR is only 5.4 dB with the mean at 2.6 dB. Data set 12 has the highest mean SNR at 4.7 dB, which could be explained by it being recorded at the DopTrack roof with a high maximum elevation of 65°. There being no obstacles obstructing the view has a positive effect on the overall SNR quality, which is reflected in the mean value. The best performing data from DTB Pro is Data set 19, which only has an elevation of 37° on a western pass. This data set has a peak SNR of 19.0 dB and a mean SNR of 4.2 dB. The data suggests that a direct correlation between elevation and SNR does therefore not seem to exist. All this data was gathered whilst manually pointing, which can have a big influence on the data itself.

The noise floor data of this experiment is present in Appendix A.3 and can be seen in Table 6.17. This data shows the noise floor being lower for DTB, between 0.27 and 0.67, compared to DTB Pro, which has values between 0.36 and 0.96. Higher noise values can be expected for DTB Pro, as the introduction of the GPS clock to the system is an additional component that can be a source adding to the overall system noise. When looking at the mean SNR, three out of the six data sets from DTB Pro do not meet the 3 dB threshold, whereas all data sets from DTB pass it. The findings from this section point to it being detrimental for the SNR to add a GPS clock.

Table 6.17: Overview of mean values of noise floor and SNR for Experiment 2.1 Lower noise floor values and higher SNR values are better.

Data set Identifier	**DopTrackBox Mean Noise floor [-]**	**DopTrackBox Mean SNR [dB]**
12 - DTB 1	0.453	4.721
13 - DTB 2	0.665	3.812
14 - DTB 3	0.270	4.573
15 - DTB Pro 1	0.963	2.739
16 - DTB Pro 2	0.901	2.600
17 - DTB Pro 3	0.362	3.718
18 - DTB Pro 4	0.762	2.787
19 - DTB Pro 5	0.682	4.213
20 - DTB Pro 6	0.495	3.565

EXPERIMENT 2.1: DOPPLER SHIFT

The Doppler shift data part of Experiment 2.1 can be seen in Table 6.18. There is no frequency data available for Data set 13, which leaves only Data sets 12 and 14 for the DTB side of the comparison. The format of this table is the same as that of Table 6.10. The maximum number of data points is determined by the pass length, minus 120 seconds to account for the additional recording buffer. This value is then divided by the bin size, 0.5. The data quality is the percentage of data points in a data set with respect to the maximum.

There does not seem to be a major difference in data quality between the DTB and DTB Pro configurations. The only noteworthy sample being Data set 16 at 11.3%. There does not seem to be a specific reason for this when looking at the pass characteristics; it being a western pass with a maximum elevation of 76°. The SNR data for this data set also performed subpar. The signal was thus not received very well by DopTrackBox which caused the DTPP to have trouble extracting useful information.

The range rate difference for Experiment 2.1 is presented in Figure 6.13. The range rate difference is the difference between the measured range rate and the TLE range rate. The top plot contains the DTB data and the bottom plot contains the DTB Pro data. The RMS values have not been shown in this figure to improve legibility. Since the RMS value is a single value for each data set, they have been placed into Table 6.19 at the end of this section.

Table 6.18: Frequency data availability of DopTrackBox and DopTrack for Experiment 2.1.

Experiment 2.1			
Identifier	**Maximum Data points in set**	**DopTrackBox**	
		Data points	**Quality**
12 – DTB 1	1320	261	19.8%
13 – DTB 2	960	0	0.0%
14 – DTB 3	1200	325	27.1%
15 – DTB Pro 1	1320	245	18.6%
16 – DTB Pro 2	1320	149	11.3%
17 – DTB Pro 3	1200	292	24.3%
18 – DTB Pro 4	1200	215	17.9%
19 – DTB Pro 5	1320	308	23.3%
20 – DTB Pro 6	1200	279	23.3%

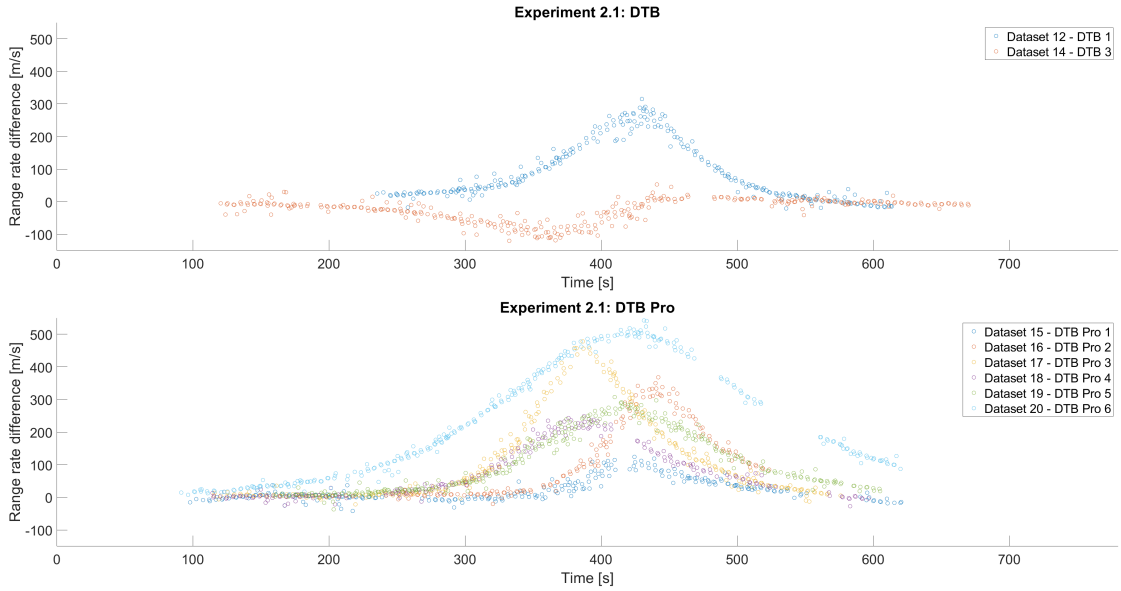


Figure 6.13: Experiment 2.1 Range Rate Difference. The top plot shows results from DTB (without GPS) and the bottom plot shows results from DTB Pro (with GPS).

As was stated in the SNR part of this section, the data imbalance must be taken into account. Since Data set 13 does not contain any frequency data, this skews the balance even more. The available data seems to indicate not including a GPS is beneficial as it results in lower range rate differences between the measured range rate and the TLE range rate. The maximum range rate difference value for Data set 12 is 315 m/s, whereas Data sets 16, 17, and 20 have a higher range rate difference and Data set 19 is on par. Data set 20 consistently has the highest range rate difference and also has the highest maximum range rate difference at 542 m/s. Another observation regarding the DTB Pro data sets is that the second highest maximum range rate difference belongs to Data set 17 at 478 m/s, but over all its RMS is lower than that of Data set 16; 170 m/s compared to 172 m/s. Data set 16 has the third highest maximum peak at 368 m/s, suggesting that across the board Data set 17 performed better than Data set 16. Many of the key values from Figure 6.13 can also be found in Table 6.19. Data set 14 is the only data set with a concave shape. This suggests the measured data for a given time is a lower range rate value compared to the TLE, something that will be discussed in more detail in Section 6.3.3.

A reason for this could be a bias due to two favourable measurements without the GPS compared to six measurements with the GPS painting a more complete picture. To see if more clarity can be provided, data from Experiment 1 could be used. The main disadvantage with this approach is the bigger difference between the

systems and the omission of one key aspect of Experiment 2, being the manually pointed measurements with the Yagi antenna. When comparing the results from Table 6.11 with Table 6.19, RMS values from Experiment 1.3 are between 24 and 91 m/s whereas those of the DTB configuration in Experiment 2.1 are 44 and 131 m/s. This means that Data set 14 falls within the bounds set by Experiment 1.3, but Data set 12 does not. When looking at the DTB Pro configuration, the RMS range rate difference is between 42 m/s and 292 m/s, with only Data set 15 falling within the range seen in Experiment 1.3. However, as the GPS clock is added to the DTB Pro another difference is added to the system as a whole. This makes it difficult to see which difference is causing the observed results.

Another factor is the line of sight and maximum elevation, which was discussed in the SNR part of this section as well. Data sets 12, 14, 16, and 17 all have a maximum elevation over 60°, with two eastbound and two westbound passes. Nevertheless, the lowest RMS range rate belongs to Data set 15; an eastbound pass with a maximum elevation of 59° with a RMS range rate difference of 42 m/s. As was visible in Figure 4.3, eastbound passes have more obstructions than westbound passes. Data sets 13 and 20 suggest elevation can play a role, as their maximum elevation is 12° and 23° respectively. These data sets were both obtained at the LR faculty roof. The fact that no data could be extracted from Data set 13, and that the RMS range rate difference for Data set 20 is the highest of all data sets within this experiment at 292 m/s could point to low elevation passes in urban environments being more susceptible to interference from the ground.

Table 6.19 contains various details regarding range rate difference, abbreviated as RRD in the table, TCA, and the slope of the linear fit of the first residual for Experiment 2.1. The value and time of the maximum range rate difference are obtained from the available data points, and can therefore not exactly match up with the absolute maximum value if no data point exists for that value. The RMS and maximum range rate difference values have been mentioned earlier in this section.

Table 6.19: Overview of DopTrackBox range rate difference (RRD) RMS, time and value of maximum RRD, TCA, the difference of the TCA between the TLE and measurement, and the slope of the linear fit of the first residual for Experiment 2.1.

Data set Identifier	**RRD RMS [m/s]**	**Time of max RRD [s]**	**Value of max RRD [m/s]**	**TCA [s]**	**ΔTCA [s]**	**Slope first residual [m/s²]**
12	131.26	430.0	315.32	421.0	-2.39	-0.090
14	44.40	332.5	-119.99	378.0	0.80	0.057
15	41.93	424.5	124.10	402.0	-0.80	0.103
16	172.45	442.0	367.93	426.0	-2.42	1.029
17	170.45	386.5	477.84	373.0	-3.55	0.289
18	111.52	387.5	248.80	371.0	-2.45	0.250
19	144.78	410.0	307.36	402.0	-3.33	0.326
20	291.55	431.5	542.23	399.0	-9.87	0.593

The ΔTCA values are all negative, except for Data set 14. This is also the only data set with a concave shape, which is to be expected. The hypothesis of the GPS clock enabling better timings does not seem to hold up, as the ΔTCA is smaller than Data set 12's -2.39 s for only Data set 15. The slope of the linear fit of the first residual also is greater for DTB Pro compared to DTB, which implies the GPS clock increasing the linear drift of the SDR.

6.3.2. EXPERIMENT 2.2: EFFECT OF THE SDR

Experiment 2.2 is about using a different SDR, so the difference between the RSPduo in the DTB configuration and the RSP1A in the DTB Light configuration is investigated. This experiment is linked to Research Question 2.3, which will be discussed and answered in Section 7.4. This section is also split into a subsection on SNR and a subsection on Doppler shift. The data overview of this experiment can be seen in Table 6.20.

Data sets 12 and 13 were obtained on the roof of LR at the DopTrack ground station. All other data sets were obtained at the field location.

Table 6.20: Experiment 2.2 goal and data overview.

Experiment 2.2		
Goal	**Data 1 (DTB)**	**Data 2 (DTB Light)**
Compare the data gathered with the DTB and DTB Light configurations to see the effect of a different SDR on SNR and Doppler shift data.	• Data set 12 - DTB 1 • Data set 13 - DTB 2 • Data set 14 - DTB 3	• Data set 21 - DTB Light 1 • Data set 22 - DTB Light 2 • Data set 23 - DTB Light 3 • Data set 24 - DTB Light 4 • Data set 25 - DTB Light 5

EXPERIMENT 2.2: SNR

The SNR results from Experiment 2.2 can be seen in Figure 6.14. The top plot shows the results for DTB and the bottom plot for DTB Light. Each line represents a different data set.

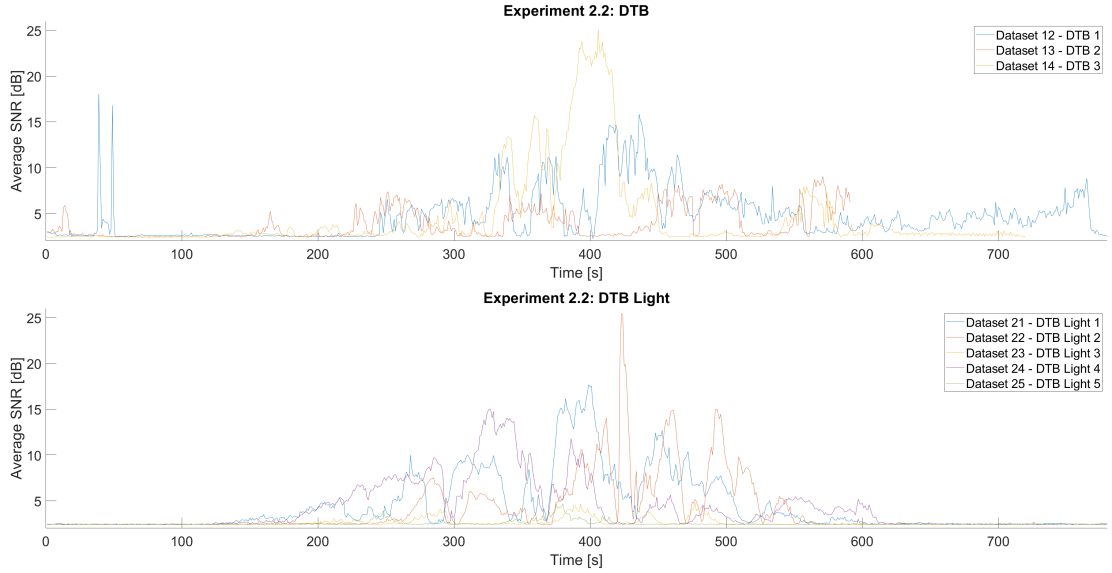


Figure 6.14: Experiment 2.2 SNR. The top plot shows results from DTB (RSPduo) and the bottom plot shows results from DTB Light (RSP1A).

It is not immediately clear from Figure 6.14 which of the two systems produces higher or more consistent SNR values. Peak SNR values are on par; 25 dB for Data set 14 and 25.4 dB for Data set 22. However, consistent performance is more important than peak SNR. Data sets 23 and 25 have a mean SNR value below 3 dB, of 2.6 dB and 2.5 dB respectively, and show no significant peaks. This can be attributed to their pass characteristics, with a maximum elevation of 33° and a nearly identical pass progression in the east as can be seen in Table 6.15. Data sets 22 and 24 have the highest maximum elevation of all data sets at 86°. The satellite passing overhead brought some challenges with pointing and holding the antenna in the most optimal direction, which can explain the behaviour of the SNR data in Figure 6.14.

The mean noise floor data for this experiment and the mean SNR values can be found in Table 6.21. The obtained noise floor values for DTB Light are lower than those for DTB for some cases, with the minimum noise floor belonging to Data set 24 at 0.228 compared to 0.270 for Data set 14. However, the highest noise floor value can also be found in Data set 23 at 0.719. This can be explained by the pass characteristics, as done earlier in this section. The mean SNR values are also close together, with the lower values of Data sets 23 and 25 being covered by their unfavourable pass characteristics. This results in it not being very clear what the effect of the SDR is on the obtained SNR.

Table 6.21: Overview of mean values of noise floor and SNR for Experiment 2.2 Lower noise floor values and higher SNR values are better.

Data set Identifier	DopTrackBox Mean Noise floor [-]	DopTrackBox Mean SNR [dB]
12 - DTB 1	0.453	4.721
13 - DTB 2	0.665	3.812
14 - DTB 3	0.270	4.573
21 - DTB Light 1	0.238	4.460
22 - DTB Light 2	0.311	3.955
23 - DTB Light 3	0.719	2.610
24 - DTB Light 4	0.228	4.187
25 - DTB Light 5	0.652	2.527

EXPERIMENT 2.2: DOPPLER SHIFT

Experiment 2.2's Doppler shift data is presented in Table 6.22, which follows the same build-up as previous tables. Just as with Experiment 2.1, Data set 13 cannot be used for the range rate difference data.

The data quality of Data sets 23 and 25 at 14.6% and 12.0% respectively are lower than those of the DTB data sets, whereas the other three are within their bounds. Data sets 23 and 25 both follow a similar pass path, being an eastern pass with a maximum elevation of 33°. Therefore these passes are more likely to be subjected by buildings and terrain blocking line of sight, which could explain their lower quality.

Table 6.22: Frequency data availability of DopTrackBox and DopTrack for Experiment 2.2.

Experiment 2.2			
Identifier	Maximum Data points in set	DopTrackBox	
		Data points	Quality
12 – DTB 1	1320	261	19.8%
13 – DTB 2	960	0	0.0%
14 – DTB 3	1200	325	27.1%
21 – DTB Light 1	1320	348	26.4%
22 – DTB Light 2	1320	285	21.6%
23 – DTB Light 3	1200	175	14.6%
24 – DTB Light 4	1320	328	24.8%
25 – DTB Light 5	1320	158	12.0%

The range rate difference between the measured range rate and the TLE range rate for Experiment 2.2 is shown in Figure 6.15. The top plot contains the DTB data and the bottom plot contains the DTB Light data. The RMS values have been omitted to improve the plot's legibility. These values can be found in Table 6.23 at the end of this section.

The data imbalance has to be taken into account here as well, as two data sets are compared against five. The DTB Light data has data sets with a much higher maximum and RMS range rate differences, notably Data set 25 at 683 m/s RMS and 1097 m/s maximum. The elevation and azimuth values are similar to those of Data set 23 with a 33° maximum elevation, which is the best performing data set of Experiment 2.2 at a 42 m/s RMS range rate difference. This points to the orbit and the pass parameters therefore being unlikely factors. Table 6.22 showed these two data sets having a similar data quality. This data suggests that a low elevation pass does not necessarily translate to worse Doppler tracking data. The disparity between the two data sets could be caused by human error, since both measurements were done whilst manually pointing. What does stand out regarding Data set 25 is the lack of data points after the 500 second mark, which can partially help explain the high RMS range rate difference.

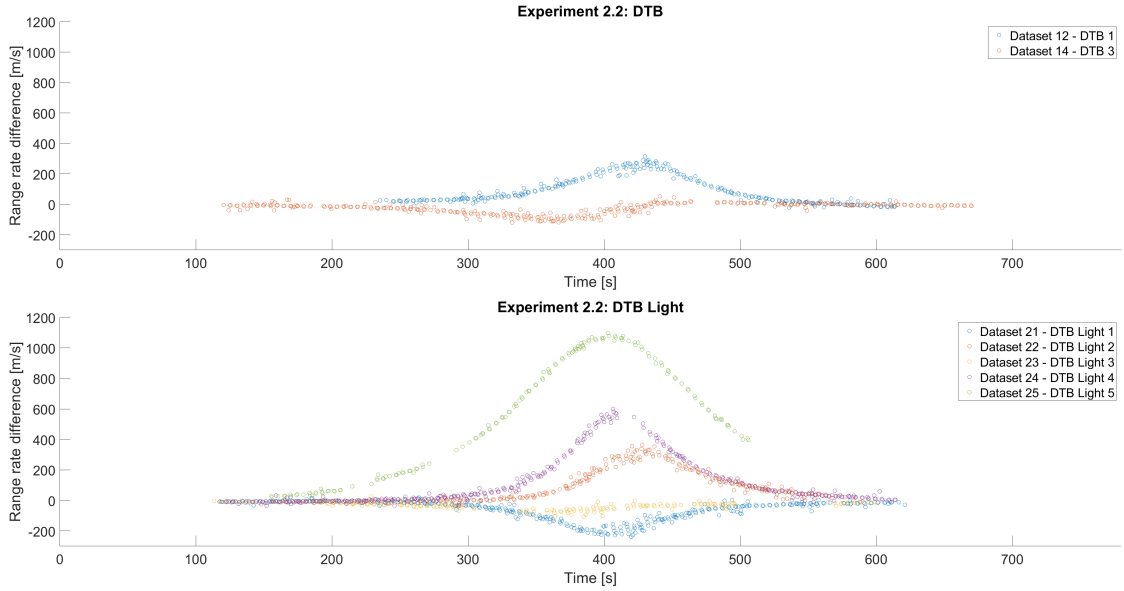


Figure 6.15: Experiment 2.2 Range Rate Difference. The top plot shows results from DTB (RSPduo) and the bottom plot shows results from DTB Light (RSP1A).

Three data sets have a concave shape. Those being Data set 14 for DTB and Data sets 21 and 23 for DTB Light. This is confirmed by the ΔTCA value of Table 6.23 being positive. The large maximum range rate difference seen in Data set 25 is accompanied by the largest ΔTCA of -15.27 s. Other details present in Table 6.23 include the range rate difference (RRD) RMS, maximum values, TCA, and slope of the linear fit of the first residual. Of the ΔTCA values for DTB Light those of Data sets 21, 22, and 23 are within the bounds of the DTB data when looking at their absolute value.

Table 6.23: Overview of DopTrackBox range rate difference (RRD) RMS, time and value of maximum RRD, TCA, the difference of the TCA between the TLE and measurement, and the slope of the linear fit of the first residual for Experiment 2.2.

Data set Identifier	**RRD RMS [m/s]**	**Time of max RRD [s]**	**Value of max RRD [m/s]**	**TCA [s]**	**ΔTCA [s]**	**Slope first residual [m/s²]**
12	131.26	430.0	315.32	421.0	-2.39	-0.090
14	44.40	332.5	-119.99	378.0	0.80	0.057
21	92.57	420.0	-238.24	403.0	1.87	-0.131
22	140.80	428.0	364.90	415.0	-2.36	0.436
23	42.18	385.0	-107.20	377.0	0.99	-0.025
24	199.60	406.5	600.19	400.0	-4.34	0.289
25	683.20	403.0	1096.92	392.0	-15.27	2.755

The data in Table 6.23 shows that DTB Light is capable of obtaining a similar RMS range rate compared to DTB, as is seen in Data sets 21, 22, and 23. These values are between 42 and 141 m/s. The slope of the linear fit of the first residual is larger for DTB Light than for DTB, which could indicate a different linear drift for the RSP1A compared to the RSPduo used in DTB. However, more data would be needed to definitively prove this.

6.3.3. EXPERIMENT 2.3: MANUALLY POINTING

Experiment 2.3 compares the data from DopTrackBox with the same data from DopTrack, just as with Experiment 1.3. But now the focus is on the effect of manually pointing the data. This experiment is linked to Research Question 2.4, which will be elaborated upon in Section 7.5. The data initially used for this experiment can be seen in Table 6.24.

Table 6.24: Experiment 2.3 goal and data overview.

Experiment 2.3		
Goal	**Data 1 (DopTrackBox)**	**Data 2 (DopTrack)**
Compare the data gathered with DopTrackBox whilst manually aiming with DopTrack to see the effect of manually aiming on SNR and Doppler shift data.	• Data set 12 - DTB 1 • Data set 13 - DTB 2 • Data set 14 - DTB 3	• Reference DopTrack 12 • Reference DopTrack 13 • Reference DopTrack 14

Data sets 12 and 13 were obtained on the roof of LR at the DopTrack ground station. Data set 14 was obtained at the field location.

EXPERIMENT 2.3: SNR

Figure 6.16 shows the results for the differences in SNR between manually pointing DopTrackBox and DopTrack. Each plot shows a direct comparison between the same data set as measured by DopTrack and DopTrackBox. The DopTrackBox data is plotted in blue and the DopTrack data is plotted in orange. According to the DopTrack's yaml files, DopTrack used the Red-Yellow antenna as is visible in the second column of Table 6.26.

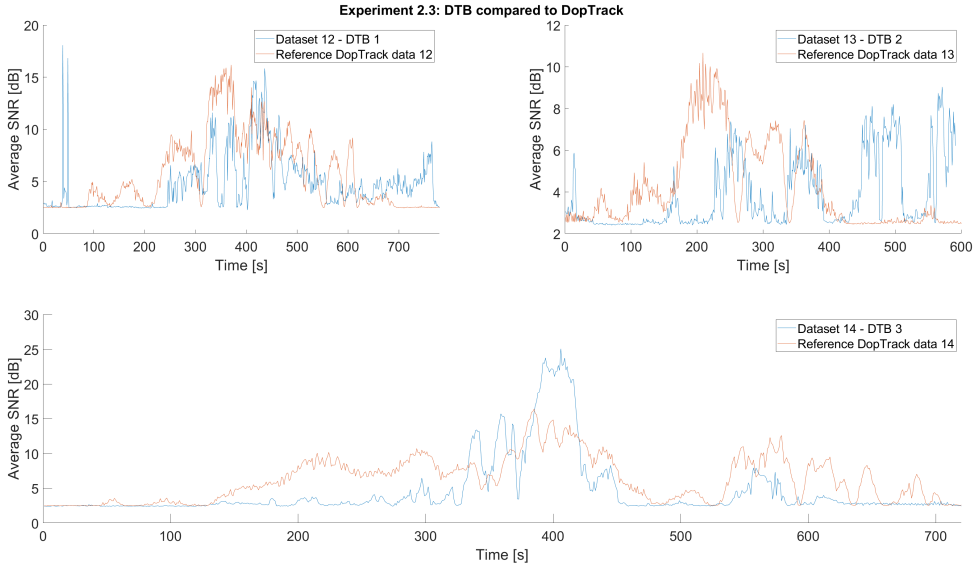


Figure 6.16: Experiment 2.3 SNR comparison for DopTrackBox (blue) and DopTrack (orange) for Data sets 12, 13, and 14. Each plot represents its own data set with the corresponding DopTrack reference.

Looking at Figure 6.16, it can be seen that Data set 12 more or less follows the same pattern as the DopTrack data. There are two SNR spikes visible at the beginning of Data set 12, which are most likely caused by interference since this behaviour has not been observed anywhere else. Ignoring those peaks, the peak values are around 16 dB for both DopTrackBox and DopTrack. Data set 13 also shows some strange peaks and valleys in the SNR graph. Looking at the specific spectrogram plot, Figure A.26 in Appendix A.3, a signal seems to be present next to the satellite's signal. This could be the cause of the spikes in SNR. As the Yagi antenna is directional, the gain of the antenna is greater than that of a omnidirectional antenna. This could explain why the DopTrackBox data picks up this signal and why it is less visible in DopTrack's data. This specific satellite pass was at a lower maximum elevation of 12°. Even though no obstacles blocked line of sight with the satellite from the LR roof, the antenna could have picked up signals from the ground. Data set 14's peak value is higher for DopTrackBox at 25 dB compared to DopTrack's 16 dB, which could be due to the nature of the directional Yagi antenna providing an increased gain.

When looking at the noise floor values from Table 6.25 it can be seen that DopTrack has a lower noise floor. DopTrackBox' noise floor is between 5.3 and 12.0 times higher in the three data sets. A factor causing a higher noise floor can be the human interactions with the system, of which the antenna is the main component.

Table 6.25: Overview of mean values of noise floor for Experiment 2.3. Lower values are better.

Data set Identifier	DopTrackBox Mean Noise floor [-]	DopTrack Mean Noise floor [-]	DopTrackBox and DopTrack difference [-]	Relative difference [-]
12	0.453	0.053	0.400	8.5
13	0.665	0.055	0.610	12.0
14	0.270	0.051	0.218	5.3

The mean SNR data from Table 6.26 show higher mean SNRs across all data sets. The smallest difference at 0.372 dB lower for Data set 13 can most likely be attributed to the strange behaviour seen in the plot, and not necessarily to the signal itself. Data set 14 has a mean SNR value 1.628 dB lower than the DopTrack data, and has thereby the largest difference. Even Data set 12, which looks similar has a lower SNR mean of 0.982 dB. These three data sets show that even though higher peak values are obtainable, consistency with a handheld antenna is lower.

Table 6.26: Overview of mean values of the mean SNR for Experiment 2.3. Higher values are better.

Data set Identifier	DopTrack Antenna	DopTrackBox Mean SNR [dB]	DopTrack Mean SNR [dB]	DopTrackBox and DopTrack difference [dB]
12 - DTB 1	RY	4.721	5.703	-0.982
13 - DTB 2	RY	3.812	4.185	-0.372
14 - DTB 3	RY	4.573	6.201	-1.628

EXPERIMENT 2.3: DOPPLER SHIFT

Finally, the effect of manually pointing on the Doppler shift data as part of Experiment 2.3 is presented. The original intention was to compare the DTB data, with the RPSduo without GPS, against the DopTrack reference. This would be an identical hardware setup compared to Experiment 1, whilst changing only the antenna. Data sets 12 and 13 were recorded at the DopTrack ground station to make the comparison between DopTrackBox and DopTrack as small as possible. Since Data set 13 contains no useful data for Doppler shift and range rate analysis, only two usable data sets would remain. Because of this, it was decided to include the DTB Pro and Light data for the comparison of manually pointing versus DopTrack's omnidirectional antennas. This can therefore also function as an extension of Experiments 2.1 and 2.2, where the additional comparison is made between the DopTrackBox concepts and the DopTrack ground station. The DTB Pro and Light data was recorded at a different location than the DopTrack ground station, which can result in some slight differences between the data sets. The complete overview of data used for this experiment is visible in Table 6.27, together with the respective DopTrack reference.

DopTrack's reference data for Data set 13 is not used, since there is no DopTrackBox data to compare with. One of the DopTrack data sets that does not completely align with its DopTrackBox counterpart is Data set 20. The DopTrack data set 20 has a starting time 1 minute later than the DopTrackBox data set, which is most likely caused by an older version of the yaml file from the scheduler software on one of the two systems.

When comparing the data quality of DopTrackBox to that of DopTrack from Table 6.27, it is clear that DopTrack has a higher overall quality. For each individual data set, DopTrack's data quality is higher than that of DopTrackBox'. DopTrackBox' data quality varies from 11.3% to 27.1%, whereas DopTrack's varies from 23.9% to 38.9%. One aspect all DopTrackBox measurements have in common is their usage of the manually aimed Yagi antenna, which is the most likely reason for the drop in data quality as it yields less consistent results. Comparing DopTrackBox' values to those of Experiment 1.3 from Table 6.10, where the variance is between 22.7% and 33.3%, lower lows are observable in Experiment 2.3. Other data sets are on par with performance seen from the Green antenna with DopTrackBox. When comparing DopTrack's data quality, an increase in quality is visible between Experiments 1.3 and 2.3, which could be attributed to the switch from the Green to the Red-Yellow antenna for DopTrack.

Table 6.27: Frequency data availability of DopTrackBox and DopTrack for Experiment 2.3.

Experiment 2.3					
Identifier	**Maximum Data points in set**	**DopTrackBox**		**DopTrack**	
		Data points	**Quality**	**Data points**	**Quality**
12 – DTB 1	1320	261	19.8%	409	31.0%
13 – DTB 2	960	0	0.0%	322	33.5%
14 – DTB 3	1200	325	27.1%	410	34.2%
15 – DTB Pro 1	1320	245	18.6%	463	35.1%
16 – DTB Pro 2	1320	149	11.3%	412	31.2%
17 – DTB Pro 3	1200	292	24.3%	414	34.5%
18 – DTB Pro 4	1200	215	17.9%	415	34.6%
19 – DTB Pro 5	1320	308	23.3%	416	31.5%
20 – DTB Pro 6	1200	279	23.3%	423	35.3%
21 – DTB Light 1	1320	348	26.4%	489	37.0%
22 – DTB Light 2	1320	285	21.6%	315	23.9%
23 – DTB Light 3	1200	175	14.6%	467	38.9%
24 – DTB Light 4	1320	328	24.8%	397	30.1%
25 – DTB Light 5	1320	158	12.0%	424	32.1%

There are five figures with range rate data corresponding to Experiment 2.3. Each plot shows a direct comparison between DopTrackBox' and DopTrack's range rate difference for the same data set. The DopTrackBox data is plotted in blue and the DopTrack data is plotted in orange. A line with the RMS value for the range rate difference has been plotted in each plot. The plots only show a data point when there is actual data present, which can result in slight differences between the DopTrack and DopTrackBox data.

Figure 6.17 shows the range rate difference for the DTB configuration, the RSPduo without GPS. Due to there being six different data sets with the DTB Pro configuration, the RSPduo with GPS, these data sets have been placed in two figures. Figure 6.18 contains Data sets 15 through 18 and Figure 6.19 contains Data sets 19 and 20. The same has been done for the five data sets for the DTB Light configuration, which uses the RSP1A. These can be found in Figure 6.20 for Data sets 21 through 24, and in Figure 6.21 for Data set 25. Additional graphs with the frequency plots are available in Appendix A.4.

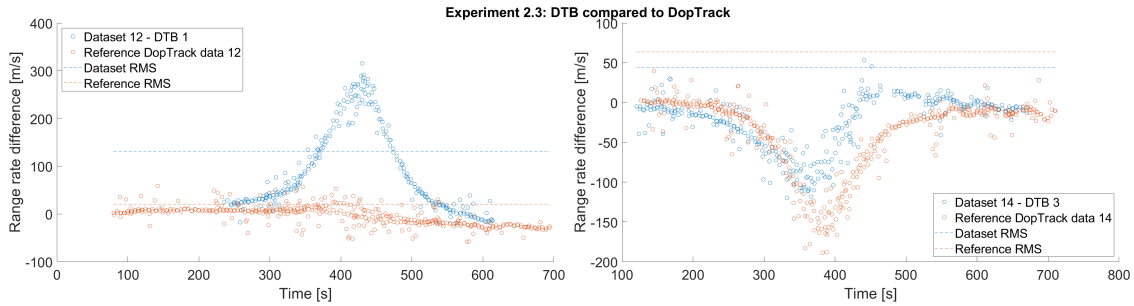


Figure 6.17: Experiment 2.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB configuration (Data sets 12 and 14). Each plot represents its own data set with the corresponding DopTrack reference.

In Figure 6.17 both data sets show interesting features compared to DopTrack's data. The maximum range rate difference for Data set 12 is around 250 m/s greater than that of DopTrack, with a difference in RMS range rate of 111 m/s. DopTrack's data is a lot flatter too. Data set 14 is the only data set of Experiment 2 where DopTrackBox' RMS range rate difference is smaller than DopTrack's, with a 19 m/s difference. It starts out with a larger difference than DopTrack's until around 300 seconds when it has a smaller difference, resulting in the overall RMS being smaller. Both DopTrackBox and DopTrack have a concave bell shape, but the DopTrackBox data seems to have its maximum absolute difference 50 seconds earlier. An exact reason for this is unknown. There is a difference in the TCA and ΔTCA values for DopTrackBox and DopTrack. DopTrackBox' TCA is at 378 s with a ΔTCA of 0.80 s, whereas DopTrack's TCA is at 380 s with a ΔTCA of 1.49

s. This is a slight difference and should therefore not cause the 50 second gap between maximum values as seen in the plot. The RMS values are always positive, as a negative number squared is positive.

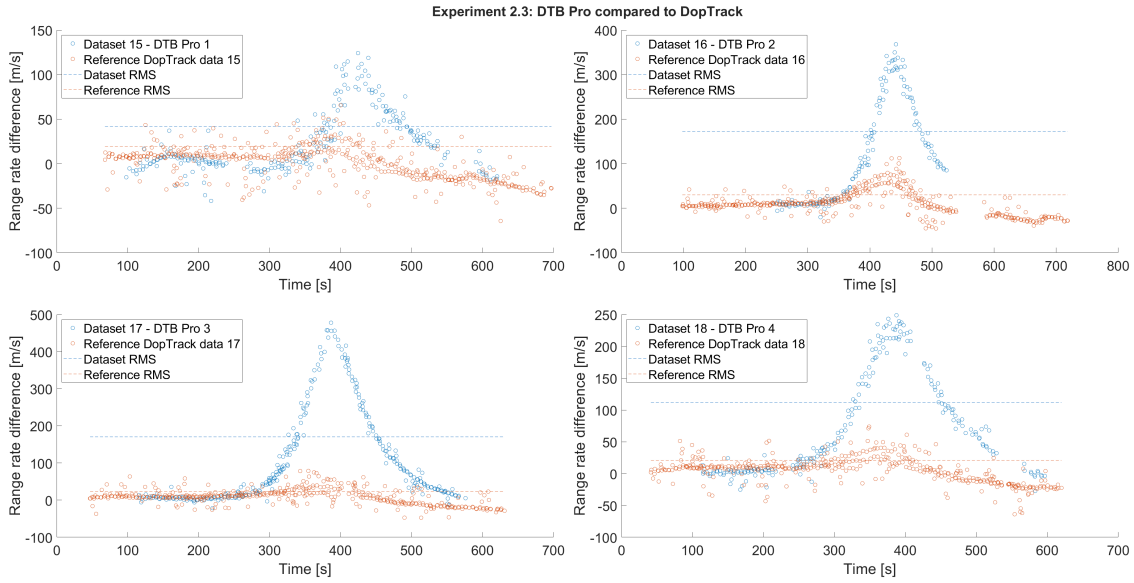


Figure 6.18: Experiment 2.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Pro configuration (Data sets 15, 16, 17, and 18). Each plot represents its own data set with the corresponding DopTrack reference.

The first four DTB Pro data sets can be found in Figure 6.18. In all four cases, DopTrackBox has a larger range rate variance than DopTrack. The low number of data points in Data set 16, at only 149 points and a 11.3% quality rating, are all centred around the middle of the graph. All four DopTrackBox data sets follow the DopTrack data until it starts to diverge around 250 to 350 seconds. This causes the larger difference in RMS range rate, which varies from 22.8 m/s for Data set 15 to 147.4 m/s for Data set 17. The time of the maximum range rate difference value is within 10% of the DopTrack time value, suggesting a similar pattern. These larger differences can also be found in the TCA data of Table 6.30, where the difference between the TCA of DopTrackBox and DopTrack is smallest for Data set 15 at 4 seconds and largest for Data sets 16 and 17 at 6 seconds. The ΔTCA values for DopTrackBox are also 5 to 10 times larger than those of DopTrack.

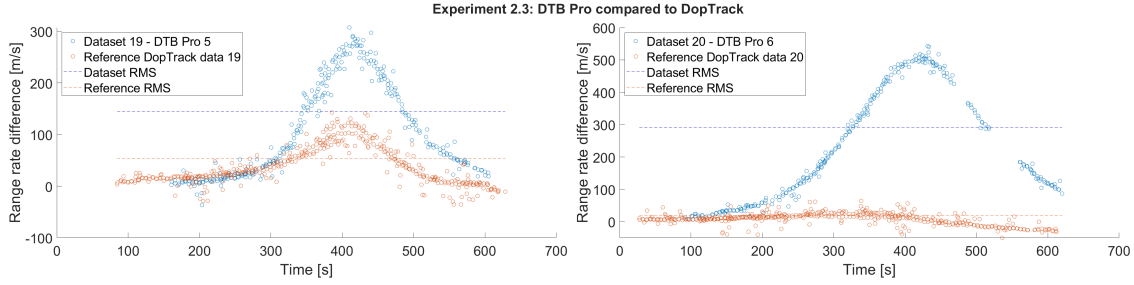


Figure 6.19: Experiment 2.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Pro configuration (Data sets 19 and 20). Each plot represents its own data set with the corresponding DopTrack reference.

Figure 6.19 contains the last two DTB Pro data sets. Data set 19 shows DopTrack's second largest maximum range rate difference at 143 m/s. This results in a RMS range rate difference of 91 m/s. Data set 20 was recorded at the DopTrack ground station. The RMS range rate difference between DopTrackBox and DopTrack is 271 m/s, which is the second highest range rate difference between the two systems. When looking at the respective TCA for DopTrackBox and DopTrack, there is a 53 second difference between the systems. The ΔTCA for DopTrackBox is -9.9 s, which could help explain the large range rate difference..

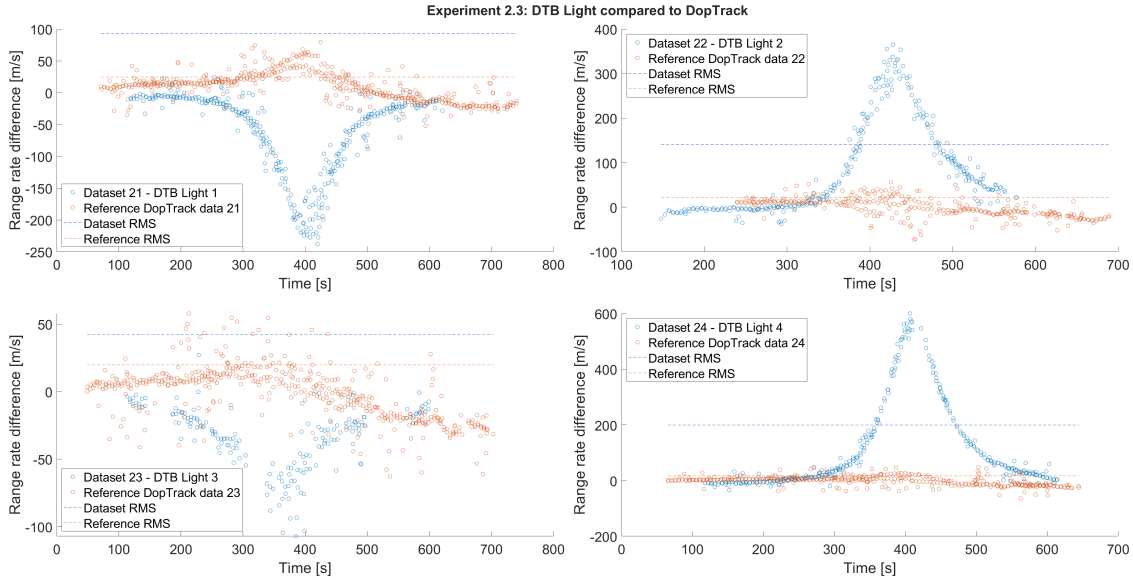


Figure 6.20: Experiment 2.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Light configuration (Data sets 21, 22, 23, and 24). Each plot represents its own data set with the corresponding DopTrack reference.

The first four of the five total DTB Light data sets are shown in Figure 6.20. Something unique to Data sets 21 and 23 is their shape as they do not follow the same pattern as DopTrack's data. In all other data sets the DopTrackBox and DopTrack reference data follow the same pattern; both a convex or concave bell shape. This is not the case for these two aforementioned data sets. Here the DopTrackBox data is concave, whereas the DopTrack reference is convex. The pass parameters of these two data sets from Table 6.15 do not seem to have a common factor between them. Their TCA occurs within 2 and 3 seconds respectively of DopTrack's, and the ΔTCA of these two data sets is not strange either. No specific explanation can be provided to pinpoint the cause of the data to behave this way.

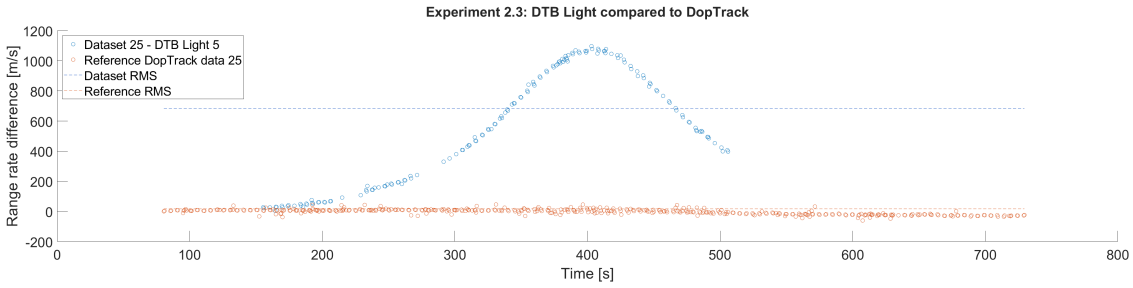


Figure 6.21: Experiment 2.3 Range Rate Difference comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Light configuration (Data set 25). Each plot represents its own data set with the corresponding DopTrack reference.

The fifth DTB Light data set can be seen in Figure 6.21. This data set stands out since it has the largest RMS range rate difference between the two systems of 683 m/s and the largest maximum value of 1097 m/s. The ΔTCA for this specific data set is also the largest of all data sets at -15.27 seconds.

From all the range rate difference measurements part of Experiment 2.3 the value of both DopTrackBox' and DopTrack's RMS are shown in Table 6.28. The second column shows the DopTrackBox data, the third DopTrack's data, and the fourth is the difference between the two. A value below 0 indicates that DopTrackBox has a lower RMS range rate difference than DopTrack. It should be noted that all DopTrack data was obtained using the Red-Yellow DopTrack antenna with the LNA on. DopTrackBox employed no LNA and all data was obtained whilst manually pointing the antenna. All the RMS values are obtained from the difference between the measured range rate and the TLE range rate.

Table 6.28: RMS of range rate difference for DopTrackBox and DopTrack reference data for Experiment 2.3. Lower values are better.

Data set Identifier	DopTrackBox Range rate difference RMS [m/s]	DopTrack Range rate difference RMS [m/s]	DopTrackBox and DopTrack difference [m/s]
12	131.26	20.35	110.908
14	44.40	63.66	-19.261
15	41.93	19.09	22.837
16	172.45	29.83	142.620
17	170.45	23.04	147.404
18	111.52	21.12	90.401
19	144.78	53.40	91.383
20	291.55	19.94	271.604
21	92.57	24.57	68.003
22	140.80	21.44	119.362
23	42.18	19.79	22.388
24	199.60	16.71	182.886
25	683.20	17.31	665.891

Table 6.28 shows that only Data set 14 has a lower RMS range rate than DopTrack. This suggests that in general, DopTrackBox has a higher RMS range rate difference compared to DopTrack. For Experiment 1.3 the differences varied between -39.4 m/s and 29.7 m/s. Data sets 14, 15, and 23 fall within that range. These three data sets have different pass parameters, with elevations of 76°, 59°, and 33° and one western pass and two eastern passes. These three passes are also all part of a different experiment subset. The most likely cause for the increase in RMS between DopTrackBox and DopTrack is the change to a manually pointed antenna, as this is something all data sets have in common.

Next to the RMS range rate values, the maximum range rate difference values and times for Experiment 2.3 have also been noted. They can be seen in Table 6.29, both for DopTrackBox and DopTrack. Range rate difference has been abbreviated to RRD in the table.

Table 6.29: Overview of maximum range rate difference (RRD) and the time at which this maximum occurs for Experiment 2.3.

Data set Identifier	DopTrackBox Time of max RRD [s]	DopTrackBox Value of max RRD [m/s]	DopTrack Time of max RRD [s]	DopTrack Value of max RRD [m/s]
12	430.0	315.32	142.0	58.51
14	332.5	-119.99	382.5	-188.84
15	424.5	124.10	400.0	65.56
16	442.0	367.93	447.5	112.42
17	386.5	477.84	354.5	78.75
18	387.5	248.80	349.0	73.82
19	410.0	307.36	385.5	142.77
20	431.5	542.23	338.5	63.96
21	420.0	-238.24	424.5	79.26
22	428.0	364.90	453.0	-73.27
23	385.0	-107.20	536.5	-62.87
24	406.5	600.19	562.5	-64.81
25	403.0	1096.92	607.5	-60.22

Some of the data sets have a large difference between the time of the maximum range rate difference for DopTrackBox and DopTrack. These are Data set 12, with a 288 second difference, Data set 20, with a 93 second difference, Data set 23, with a 151.5 second difference, Data set 24, with a 156 second difference, and Data set 25 with a 204.5 second difference. In some plots, like those of Data sets 24 and 25 in Figures 6.20 and 6.21, there is no clear curvature visible in the DopTrack data. This effect could appear larger due to the axis scale of the y axis, which is needed to represent the DopTrackBox data correctly. For the other data sets a curve in DopTrack's data is visible, but the maximum does not align with the peak of the curve.

These points can be compared against the TCA seen in Table 6.30. This table contains the TCA, ΔTCA, and slope of the linear fit of the first residual for both DopTrackBox and DopTrack. It shows that the times at which the maximum range rate difference is observed for DopTrack do not align with the TCA for that data set in the case of DopTrack data 12, 23, 24, and 25. This implies that the maximum range rate error does not occur when the satellite is the closest to the ground station, and thus when the change in relative velocity is the greatest. DopTrackBox does not show this behaviour.

Table 6.30: Overview of the TCA, the difference of the TCA between the TLE and measurement, and the slope of the linear fit of the first residual of DopTrackBox and DopTrack for Experiment 2.3.

Data set Identifier	**DopTrackBox**			**DopTrack**		
	TCA [s]	**ΔTCA [s]**	**Slope first residual [m/s²]**	**TCA [s]**	**ΔTCA [s]**	**Slope first residual [m/s²]**
12	421.0	-2.39	-0.090	422.0	0.03	-0.076
14	378.0	0.80	0.057	380.0	1.49	0.008
15	402.0	-0.80	0.103	406.0	-0.16	-0.057
16	426.0	-2.42	1.029	432.0	-0.49	-0.042
17	373.0	-3.55	0.289	379.0	-0.34	-0.043
18	371.0	-2.45	0.250	376.0	-0.32	-0.052
19	402.0	-3.33	0.326	408.0	-1.35	0.016
20	399.0	-9.87	0.593	346.0	-0.49	-0.043
21	403.0	1.87	-0.131	405.0	-0.50	-0.058
22	415.0	-2.36	0.436	421.0	-0.12	-0.103
23	377.0	0.99	-0.025	380.0	-0.18	-0.061
24	400.0	-4.34	0.289	408.0	-0.05	-0.047
25	392.0	-15.27	2.755	412.0	-0.09	-0.064

The ΔTCA values for Data sets 14, 21, and 23 are positive which is reflected by their plots as they show a concave bell shape. A positive ΔTCA means the TCA occurred after the predicted TLE TCA. Something that can be observed by looking at the ΔTCA value and the plots, is that a large ΔTCA has large impact on range rate difference. Data sets with a bigger ΔTCA, like Data set 25, have a larger RMS and maximum range rate difference. A reason for this can be the relative velocity changing the quickest around the TCA, resulting in a larger range rate difference between the TLE and measured data. When comparing the DopTrackBox ΔTCA for Experiment 2.3 with those of Experiment 1.3, the ΔTCA for Experiment 2.3 shows higher numbers. Experiment 1.3's ΔTCA varied between 0.87 s and 2.29 s, whereas those for Experiment 2.3 varied between 0.80 s and 15.27 s. The sign has been omitted, as it is the magnitude which is important as it indicates the difference whereas the sign indicates whether it happened earlier or later compared to the TLE TCA.

When comparing the ΔTCA of DopTrackBox with those of DopTrack in Table 6.30, only for Data set 14 is the magnitude of the ΔTCA smaller for DopTrackBox. Therefore, DopTrackBox whilst manually pointing the antenna has a generally larger ΔTCA compared to DopTrack.

The slope of the linear fit of the first residual can provide some insight into the linear drift of each of the DopTrackBox configurations. Comparing the slope for DopTrackBox with DopTrack, DopTrackBox' slope is approximately one order of magnitude greater than those of DopTrack. Comments on the different DopTrackBox configurations were made in the respective sections. Since there are only two DTB data sets, those being Data sets 12 and 14, there is not enough data to determine whether the antenna has a great impact on the slope of the linear fit of the first residual.

7 | Data Analysis

The purpose of this chapter is to analyse and discuss the data presented in Chapter 6 in order to answer Research Question 2; *How do the Doppler and SNR data of DopTrackBox compare to that of DopTrack?* There are four subquestions which can all be answered with the acquired data once it has been analysed.

To streamline this analysis, this chapter has been divided into several sections. The first thing to be discussed are the shortcomings of the available results in Section 7.1. As the results are used to draw the conclusions, it is important to acknowledge their shortcomings so they can be taken into account. With the data described in Chapter 6 and the shortcomings mentioned, the data can be analysed and interpreted to answer the four subquestions. These are answered in their own respective section.
The first subquestion where the difference between DopTrackBox and DopTrack is compared is addressed in Section 7.2. The second subquestion on the effect of a GPS clock on DopTrackBox' data is answered in Section 7.3. The third subquestion looking at different SDR hardware is discussed in Section 7.4. The fourth subquestion regarding manually pointing is explicated in Section 7.5. Finally, all the findings from the data analysis are concluded in Section 7.6, where Research Question 2 will be answered.

7.1. Shortcomings

There are several shortcomings that need to be addressed to properly analyse and discuss the obtained results. These shortcomings will be used as points of improvement and recommendations for further development of DopTrackBox, which will be discussed in Chapter 8. Some of these shortcomings can be mitigated by broadening the data sets included for an experiment.

One of the main points to consider is the imbalance of the data sets that are compared. Experiment 2.1 has three different data sets for DTB and six for DTB Pro. Due to there being less data, conclusions could be drawn based on the few data sets that would not align with the findings were there more data. This is also true in a more general sense. A more complete overview can be seen if there is more data, as patterns that emerge from a few data sets can form due to randomness. This randomness decreases when the number of different data sets increases. Another experiment where this was the case was Experiment 2.3, where there originally were only two data sets comparing the Doppler shift data between DopTrackBox and DopTrack. Since this is not enough to draw meaningful conclusions, all data sets part of Experiment 2 were included to be compared against DopTrack's Doppler shift data. Therefore, next to only DTB data, DTB Pro and DTB Light data are used. These configurations have an additional hardware difference compared to DTB; the GPS clock for DTB Pro and the different SDR for DTB Light. This additional difference has to be taken into account for the analysis of Experiment 2.3.

The experiments have been set up in such a way that the differences between the recordings of the same experiment are minimised. However, no two recordings are the same. This is true for both the satellite's orbit during the pass and the human operation of the system. By following the methodology outlined in Section 4.4 these effects are minimised, but cannot be fully eliminated. Notes were taken after each recording to specify if any circumstances occurred that could impact the data acquisition. Even though great care was taken, behaviour seen in the data cannot be fully attributed to a single event as all events during a measurement determine the data's characteristics.

Something not necessarily a shortcoming, but something of note, is when comparing different data sets with each other, different passes are compared. Each pass has its own characteristics regarding visibility and distance to the ground station. These differences in the different data sets must be taken into account when comparing them. Data sets from DopTrackBox and DopTrack from the same satellite pass have the identical pass characteristics, whereby these effects can be eliminated.

7.2. DOPTRACKBOX AND DOPTRACK

Research Question 2.1 "What is the difference between the SNR and Doppler shift data of DopTrackBox and DopTrack?" is answered in this section. The answers are obtained from Experiment 1.3 and its analysis provided in Section 6.2.3.

Looking at the SNR data, a clear observation is the increased values when using the Red-Yellow antenna compared to the Green antenna for both DopTrackBox and DopTrack. The mean SNR of all data sets using the Red-Yellow antenna is 4.229 dB, whereas it is 3.336 dB for the data sets using the Green antenna with LNA enabled. This is a 0.893 dB or 26.8% increase for the Red-Yellow antenna on DopTrackBox. There is only a single DopTrack data set using the Red-Yellow antenna, but its 4.195 dB SNR is 1.070 dB higher than the highest SNR using the Green antenna.
It is also advantageous to use an LNA when available. The Green antenna without LNA has a mean SNR of 2.728 dB, which is 0.608 dB lower compared to the Green antenna with LNA; enabling the LNA results in a 22.3% increase in SNR. The effect of the LNA is more impactful on DopTrack than on DopTrackBox, where using the LNA shows a 19.5% increase from 2.450 dB without LNA to 2.927 dB with LNA on the Green antenna. This is a smaller percentage increase than DopTrackBox', but the DopTrackBox SNR data shows some parts exceeding the 3 dB threshold whereas this is not the case for DopTrack. In that regard, DopTrackBox should be capable of receiving data for decoding as most of the actual signal exceeds the 3 dB threshold.
If actual data acquisition were to be one of the objectives of a system based on DopTrackBox, the system should be changed such that the SNR always exceeds 3 dB. For this purpose, DopTrackBox seems to exceed or be on par with DopTrack. When looking at all data, DopTrackBox has a higher SNR in 9 out of 10 cases resulting in an average SNR difference between the systems of 0.618 dB in DopTrackBox' favour.

The findings on the effect of the LNA on DopTrackBox are summarised in Table 7.1. There are several tables similar to this one within this chapter, each following the same layout and principle. These tables contain the average values for all relevant data sets of a specific variable. These values are presented in the second and third columns. The fourth column shows the difference between the second and third column. The fifth column expresses this difference in a percentage value. Positive difference and percentage difference values indicate the value in the second column being greater, whereas negative values indicate the opposite.

Table 7.1: Overview of average performance parameters for the Green DopTrackBox antenna LNA on and LNA off cases. Higher values are better.

Variable Average	LNA on	LNA off	Difference	Percentage Difference
SNR [dB]	3.336	2.728	0.608	22.3%
Data availability	100%	25%	75%	300.0%

Doppler Tracking and thus range rate measurements are the main objective of DopTrackBox and DopTrack. The average data quality using the Green antenna is 25.4 %, whereas it is 30.7% for the Red-Yellow antenna. Data quality is defined as the number of available data points with respect to the theoretical maximum number of data points in a data set. When looking at the average RMS range rate difference for the different antennas, the Green antenna with LNA on yields 69.8 m/s and the Red-Yellow antenna results in 54.2 m/s. These findings fall in line with those of the SNR data, where the Red-Yellow antenna produced better results than the Green antenna.
With all specifics regarding the different antennas on DopTrackBox mentioned, an overview can be presented. Table 7.2 contains the average values for various variables from DopTrackBox' measurements on the Red-Yellow and Green antennas. The abbreviation for range rate difference (RRD) has been used.

Table 7.2: Overview of average performance parameters for the Red-Yellow and Green DopTrackBox antenna cases. Higher values are better, except for RMS RRD.

Variable Average	Red-Yellow	Green	Difference	Percentage Difference
SNR [dB]	4.229	3.336	0.893	26.8%
Data quality	30.7%	25.4%	5.3%	20.9%
RMS RRD [m/s]	54.2	69.8	-15.6	-22.3%

From the acquired data it is clear that DopTrackBox can perform Doppler tracking. Especially when DopTrackBox is using the Red-Yellow antenna, the range rate difference RMS is smaller than that of DopTrack, with an average 23.2 m/s smaller RMS range rate difference. This indicates that DopTrackBox can provide range rate data suggesting an orbit closer to the expected TLE orbit than DopTrack in some cases. All these results were obtained by using the same processing software used for DopTrack with slight modifications to ensure DopTrackBox' compatibility. When both systems are using the Green antenna, DopTrack's data has a lower RMS difference than DopTrackBox'. This scenario includes Data sets 1 and 2, where the average between the two is 6.5 m/s in DopTrack's favour. Data set 3 uses the Red-Yellow antenna for DopTrack, and has a 29.7 m/s lower RMS range rate difference than DopTrackBox. When combining these three data sets, as all three use the Green antenna in DopTrackBox' case, DopTrackBox has an average RMS range rate difference 14.2 m/s greater than DopTrack. However, this does not invalidate the fact that DopTrackBox is still capable of performing Doppler tracking.
As with the SNR data, it is advantageous to use an LNA. Only one of four data sets yielded usable data without the LNA enabled, compared to all seven data sets with LNA yielding usable data. The maximum range rate differences from Table 6.12 also suggest parity between the two systems. The maximum range rate difference for DopTrackBox using the Green antenna with LNA is 158.8 m/s on average, whereas it is 129.8 m/s in DopTrack's case. The absolute value of the maximum is used, as it indicates the difference between the TLE and measured data. In case of the Red-Yellow antenna, the average for DopTrackBox is 141.5 m/s and 180.8 m/s for DopTrack. The RMS provides more insight over the entire measurement compared to the maximum value.
The same can be said regarding the ΔTCA. The average over the three Green antenna measurements results in being 0.63 s larger than DopTrack's, whereas the opposite is true for the Red-Yellow antenna which is 0.74 s smaller for DopTrackBox compared to DopTrack. This means that when using the Red-Yellow antenna, DopTrackBox' measured TCA is closer to the TLE TCA and the opposite is true for the Green antenna. This is another indication for DopTrackBox' ability to perform on par with DopTrack.
The final metric to look at is the frequency drift of both systems. This is measured by the slope of the linear fit of the first residual, being the range rate's derivative. For the Green antenna, DopTrackBox has a slope value of 0.073 m/s², compared to DopTrack's 0.053 m/s². In case of the Red-Yellow antenna DopTrackBox has a slope value of 0.063 m/s² and DopTrack has a 0.10 m/s² slope. This points towards there not being a larger frequency drift in DopTrackBox' case.

From this it can be concluded that DopTrackBox and DopTrack perform very similar. There are some factors that could explain the results obtained from DopTrack. DopTrack is a larger system with a discrete radio. Inspection of the DopTrack system showed there were some delays caused by radio induced lag, which can deteriorate performance. The cables connecting DopTrack to the antennas are also physically longer, but the increase in length is not that significant being only 3 to 4 metres. The differences between the two systems are smaller than initially expected. DopTrackBox is clearly capable of completing the same tasks as DopTrack. In some cases, like Data set 3, DopTrack outperforms DopTrackBox on the SNR front being 0.777 dB higher. On average, DopTrackBox has a 0.618 dB higher SNR than DopTrack. For the Doppler shift data, DopTrackBox and DopTrack also perform similarly. For the Red-Yellow antenna, DopTrackBox has a 23.2 m/s smaller RMS range rate difference compared to DopTrack, whereas with the Green antenna DopTrackBox has a 14.2 m/s larger RMS range rate difference than DopTrack. The ΔTCA is 0.63 s larger for DopTrackBox using the Green antenna compared to DopTrack, but is 0.74 s smaller using the Red-Yellow antenna.
Table 7.3 presents the average values of DopTrackBox and DopTrack mentioned in this section. The values in these tables have been rounded, but the differences are calculated using the unrounded numbers. Some of the variables have been split in results obtained with the Red-Yellow (RY) antenna, the Green (G) antenna, and all data; just as has been done in this section. For the ΔTCA and the linear slope the absolute values were used to calculate the average. This has been done to prevent positive and negative values from cancelling each other out. Therefore, these values represent the average error. The percentage values indicate the overall average SNR of DopTrackBox being 21.2% higher compared to DopTrack and the average RMS range rate difference being 12.2% lower.
In conclusion, the differences between the SNR and Doppler shift data of DopTrackBox and DopTrack are not very big and seem to perform on a similar level.

Table 7.3: Overview of average performance parameters for Research Question 2.1. Lower values are better, except for SNR and Data quality.

Variable Average	DopTrackBox	DopTrack	Difference	Percentage Difference
SNR [dB]	3.529	2.911	0.618	21.2%
Data quality	22.1%	19.2%	2.9%	15.0%
RMS RRD (RY) [m/s]	54.2	77.4	-23.2	-30.0%
RMS RRD (G) [m/s]	69.8	55.5	14.2	25.6%
RMS RRD [m/s]	60.6	69.1	-8.4	-12.2%
Absolute ΔTCA (RY) [s]	1.66	2.40	-0.74	-30.9%
Absolute ΔTCA (G) [s]	1.99	1.36	0.63	46.2%
Absolute ΔTCA [s]	1.75	1.98	-0.23	-11.8%
Absolute Linear slope (RY) [m/s²]	0.063	0.104	-0.042	-39.8%
Absolute Linear slope (G) [m/s²]	0.073	0.053	0.020	38.1%
Absolute Linear slope [m/s²]	0.070	0.079	-0.009	-10.8%

7.3. EFFECT OF A GPS CLOCK

This section answers Research Question 2.2: "What is the effect of including or excluding a GPS clock on the SNR and Doppler shift data?". This follows from the results of Experiment 2.1 and the analysis in Section 6.3.1. Thereby it compares the DTB and DTB Pro configurations as seen in Table 2.3.

When looking specifically at SNR data there does not seem to be any benefit from the included GPS clock. The data would even suggest the SNR being lower when the GPS clock is included compared to excluded, as the average DTB SNR is 4.4 dB and the average DTB Pro SNR is 3.3 dB. Pass parameters do not seem to play a major role, as the data sets with the on paper worst parameters do not have the worst SNR. These would be Data sets 19 and 20, with a maximum elevation of 37° and 23°. This aligns with the expectation of more accurate timings not necessarily improving signal quality.
For Doppler shift data, the timing benefits of a GPS clock should help to decrease timing errors. However, the data from Experiment 2.1 does not make this effect clearly visible. The ΔTCA belonging to the DTB data is 1.6 s on average, whereas it is 3.7 s for the DTB Pro data with the GPS clock. For these averages the absolute value was used. When looking at the RMS range rate difference, the DTB data has an average of 87.8 m/s compared to DTB Pro's 155.5 m/s. The Doppler shift data obtained from Experiment 2.1 suggests it is beneficial to not include a GPS clock. This could be caused by data set imbalance, where two favourable data sets without GPS clock are compared to six more average data sets with a GPS clock.

Broadening the number of data sets without a GPS by including data from Experiment 1 does not change this observation. The RMS range rate difference values from Table 6.11 are all between 24.0 and 91.4 m/s and have an average of 60.6 m/s. This is 27.2 m/s lower than the average of the RMS range rate difference data of DTB for Experiment 2.1. This would thus reinforce the findings suggesting the GPS clock is not beneficial, as the RMS range rate difference with GPS is 67.6 m/s higher than without GPS. Therefore being 94.8 m/s higher than the RMS range rate difference of Experiment 1. The maximum range rate differences for Experiment 1 are between 72.7 and 230.9 m/s with an average of 150.6 m/s. The Experiment 2.1 DTB Pro's data has maximum range rate differences between 124.1 and 542.2 m/s with an average of 344.7 m/s. Looking at these results, both the RMS and the maximum range rate differences corresponding to the DTB Pro configuration are higher than Experiment 1. It should be noted that including this data is also less fair. There are more differences between the two tested systems than just the omission of a GPS clock. The most important difference is the manually pointing of the antenna.

It is difficult to definitively answer the subquestion posed in this section due to the imbalance between the data sets. The data suggests there being no benefit on the SNR when the GPS clock is included, as the average SNR over the respective data sets is 1.1 dB lower. When looking at the Doppler shift data, the effect of including the GPS seems to be an increase in the range rate difference between the measured orbit and the TLE predicted orbit. Data sets 12 and 14, without GPS, have a RMS range rate difference of 131.3 and 44.4 m/s respectively, whereas the data sets with GPS vary from 41.9 and 111.5 m/s for Data sets 15 and 18, up to 291.6 m/s for Data set 20. Data sets 15 and 18 are on par with the data sets without GPS. When looking

at the averages, the RMS range rate difference increases by 67.6 m/s when the GPS is included. Data set 15 shows the possibility of attaining a lower RMS range rate difference compared to the data sets without the GPS, but this result could not be replicated across the other data sets. However, as was already discussed in Section 6.3.1 this could be caused by two favourable data sets without GPS clock and six more average data sets with a GPS clock. The time difference between the TLE TCA and measured TCA is 2.1 s greater for DTB Pro. Finally, for the linear drift the average slope value for DTB is 0.074 m/s². The DTB Pro configuration has an average slope of 0.432 m/s², suggesting adding the GPS increases the system's linear drift. Table 7.4 contains an overview of the average values for the SNR, RMS range rate difference, ΔTCA, and linear slope. The obtained data points to the inclusion of a GPS clock being detrimental for SNR and Doppler tracking. The obtained SNR was decreased by 25.1% and the obtained RMS range rate difference by 77.0%.

Table 7.4: Overview of average performance parameters for Research Question 2.2. Lower values are better, except for SNR.

Variable Average	**DTB Pro (With GPS)**	**DTB (No GPS)**	**Difference**	**Percentage Difference**
SNR [dB]	3.270	4.369	-1.098	-25.1%
RMS RRD [m/s]	155.4	87.8	67.6	77.0%
Absolute ΔTCA [s]	3.74	1.60	2.14	134.0%
Absolute Linear slope [m/s²]	0.432	0.074	0.358	487.0%

7.4. EFFECT OF A DIFFERENT SDR

Research Question 2.3 "What is the effect of using a different SDR for DopTrackBox on the SNR and Doppler shift data?" is discussed and answered here. The answer to this subquestion is obtained from the results of Experiment 2.2 and its analysis in Section 6.3.2. Looking at the original concepts from Table 2.3, this question looks at the DTB and DTB Light configurations.

Comparing the data from the RSPduo and RSP1A shows that there does not seem to be a big difference regarding SNR. The average SNR with the RSPduo is 4.4 dB, compared to 3.5 dB for the RSP1A. This includes Data sets 23 and 25, which have the least favourable pass parameters being eastbound passes with an elevation of 33°. Excluding those would put the average SNR for the RSP1A at 4.2 dB. This suggests that the different SDRs seem to perform on par under similar conditions.

There are bigger differences visible when looking at the Doppler shift data. Just as with Research Question 2.2 in Section 7.3, these differences could be aggravated by the imbalanced data sets in the comparison. The average RMS range rate difference using the RSPduo is 87.8 m/s. For the RSP1A this is 231.7 m/s, but the RMS range rate difference of Data set 25 on its own is 683.2 m/s which greatly impacts the average. The Doppler shift data from the RSP1A shows that equal results are possible, as Data sets 21 and 23 have a RMS range rate difference of 92.6 m/s and 42.2 m/s respectively. These values fall within the bounds seen by the data from the RSPduo, which are 131.3 and 44.4 m/s.

Data sets 24 and 25 have a maximum range rate difference value exceeding Data set 12's maximum by 284.9 m/s and 781.6 m/s. These are the highest range rate difference maximum values seen throughout all data sets. These extreme maximum values are paired with a high ΔTCA, at -4.34 s and -15.27 s. To calculate the average, the absolute value is used otherwise positive and negative ΔTCA values will cancel each other out. This brings the average for the RSP1A to 5.0 s, compared to the RSPduo's 1.6 s. These larger time differences result in greater range rate differences.

The linear drift of the two SDR's differ almost by a factor of 10; being 0.074 m/s² for the RSPduo and 0.727 m/s² for the RSP1A on average over all measurements. These are again the averages of the absolute values. These numbers suggest the RSP1A having a larger linear drift than the RSPduo.

Just as with Research Question 2.2, this subquestion is not easily answered. When changing the SDR, the difference in SNR data is only 0.2 dB when two outlier data sets are excluded. With these two data sets included the difference increases to 0.8 dB in the RSPduo's favour, which is a 18.8% increase of the RSPduo over the RSP1A. The Doppler shift data suggests that similar performance between the different SDRs is possible, but more outliers are visible with the RSP1A. This is shown by Data sets 21 and 23, with their RMS range rate differences of 92.6 m/s and 42.2 m/s, and Data set 25 with its RMS range rate difference of 683.2 m/s. The exact cause for this is hard to pin point, but seeing that similar results are possible it can be concluded that both

SDRs and thus both DopTrackBox concepts are viable. Data set 23 has a 2.22 m/s RMS range rate difference smaller than Data set 14 from the RSPduo. The timing differences on the RSP1A were observed to be 3.4 s greater on average and the linear drift of the RSP1A was observed to be 0.653 m/s² higher.
An overview of the average values that were used in this this section and the differences between the RSP1A and RSPduo are visible in Table 7.5. These include SNR, RMS range rate difference, ΔTCA, and linear slope. The effect of using a different SDR is hard to pinpoint. The data suggests using the RSP1A decreases SNR and increases range rate difference, which is mainly caused by outliers. Therefore, it seems likely that the RSP1A yields less consistent results.

Table 7.5: Overview of average performance parameters for Research Question 2.3. Lower values are better, except for SNR.

Variable Average	**DTB Light (RSP1A)**	**DTB (RSPduo)**	**Difference**	**Percentage Difference**
SNR [dB]	3.548	4.369	-0.821	-18.8%
RMS RRD [m/s]	231.7	87.8	143.8	163.8%
Absolute ΔTCA [s]	4.97	1.60	3.37	211.2%
Absolute Linear slope [m/s²]	0.727	0.074	0.654	888.9%

7.5. EFFECT OF MANUALLY POINTING

Research Question 2.4, "What is the effect of manually aiming for DopTrackBox on the SNR and Doppler shift data?", is answered in this section. The data for this subquestion is obtained from Experiment 2.3 and the analysis in Section 6.3.3. The inclusion of additional data from Experiments 2.1 and 2.2 has helped to form a better supported answer.

Looking at the effect of manually pointing on the SNR, it can be observed that it is more difficult to obtain a consistent SNR over the entire measurement. This is seen by the difference between the mean SNR of DopTrackBox and DopTrack, which ranges from 0.372 dB to 1.628 dB lower. This has to do with the inconsistencies that arise from the manual operation of a directional antenna. Under perfect conditions, a directional antenna should provide increased results as its gain is higher than an omnidirectional antenna. It should also pick up less noise from the surroundings as it can only record signals at which it is aimed. The tighter the antenna beam, the higher the gain factor but also the more accurate the pointing needs to be.

When comparing DopTrackBox' data quality to DopTrack's using Table 6.27, on average DopTrackBox has a 14.1% lower data quality. This includes all data from Experiment 2, therefore also including data from DTB Pro and DTB Light. This has been done since having only two data sets is insufficient to draw meaningful conclusions from.

The Doppler shift and range rate difference data show a clear pattern. Especially when comparing the results from Experiment 2 with those of Experiment 1, a clear degradation of DopTrackBox' performance can be seen. The data sets with the most equal comparison between Experiments 1 and 2 are Data sets 12 and 14. Data set 12 has a maximum range rate difference of 315.3 m/s with an RMS of 131.3 m/s. The worst performing data set of Experiment 1 is Data set 1, which has a maximum range rate difference of 224.4 m/s and an RMS of 91.4 m/s. Data set 14 is the only manually pointed data set where DopTrackBox has a lower range rate difference RMS and maximum compared to DopTrack. Compared to its DopTrack reference, Data set 14 has a 19.3 m/s lower RMS and a 68.9 m/s lower maximum. When taking the average of all data sets within Experiment 1 and 2, the average of the RMS range rate difference is 113.7 m/s higher for Experiment 2 compared to Experiment 1.

All the other data shows DopTrackBox having a higher range rate difference RMS compared to DopTrack. Tallying up the average difference between the systems results in an average 147.4 m/s higher RMS range rate difference when using DopTrackBox. It is unlikely that these effects are merely caused by the change in hardware. A much more likely case seems to be the inconsistencies that arise when manually pointing the antenna.

The timing difference between the TLE TCA and measured TCA are greater on DopTrackBox. The average absolute value of the ΔTCA for DopTrackBox is 3.9 s, compared to DopTrack's 0.4 s. There is no specific DopTrackBox configuration where the ΔTCA stands out across the measurements, but a few notable outliers can be seen in Data sets 20, 24, and 25. Their corresponding ΔTCA is -9.87 s, -4.34 s, and -15.27 s. When

this is compared to the average of the absolute values for Experiment 1, being 1.7 s, Experiment 2.3 using the handheld antenna sees an increase of 2.2 s.

Similar observations can be made by looking at the linear drift of DopTrackBox. However, it is difficult to pinpoint whether these effects are caused by the antenna or by other hardware changes between the different DopTrackBox configurations. Nevertheless, compared to DopTrack a 0.44 m/s² increase in linear drift is visible on DopTrackBox.

Regarding elevation, the following observations can be made. There seem to be no direct links between SNR and elevation. Data set 14 has a maximum elevation of 76° and a mean SNR of 4.6 dB, whereas Data set 16 with the same elevation has a mean SNR of 2.6 dB. These two data sets are of different DopTrackBox configurations, but Data set 19 which is using the same configuration as Data set 16 has a maximum elevation of 37° and has a mean SNR of 4.2 dB. These are all westbound passes. Data sets 23 and 25, obtained with DTB Light, are eastbound passes with a maximum elevation of 33° and a mean SNR of 2.6 dB and 2.5 dB.
Regarding range rate difference, Data set 13 had a maximum elevation of 12° and no frequency data could be extracted. Data set 20 had the highest RMS range rate difference of all DTB Pro data at 292 m/s, with a maximum elevation of 23°. These passes were obtained at the DopTrack ground station roof. When looking at Data sets 23 and 25 again, two passes at a maximum elevation of 33°, Data set 23 has a RMS range rate difference of 42 m/s. However, Data set 25 a RMS range rate difference of 683 m/s.

These findings imply that elevation is not the major factor deciding the data's performance. When widely different results are obtained at the same elevation, something else must be behind this. These differences are also seen between data sets using the same DTB configuration. Therefore, the most likely cause is the human operation of the manually pointed antenna. This is the only factor that can differ between measurements within the same hardware configuration.

The effects of manually aiming for DopTrackBox are the following: the SNR of the data is less consistent when compared to using an omnidirectional antenna with LNA on. This is mainly seen by the mean SNR being 0.372 dB to 1.628 dB lower for DopTrackBox compared to DopTrack, whereas this was not visible during Experiment 1.3. This results in an average decrease of 0.994 dB. The Doppler shift data from DopTrackBox has on average a 147.4 m/s higher RMS range rate difference compared to DopTrack. When comparing the data from this experiment with Experiment 1.3, the average range rate difference RMS of Experiment 2 is 113.7 m/s higher. The timing between the measured TCA and TLE TCA is also 3.5 s greater compared to DopTrack or 2.2 s greater compared to Experiment 1. The averages of these variables comparing DopTrackBox whilst manually aiming with DopTrack are placed in Table 7.6 for easy reference.
A likely cause for the decrease in SNR and increase in Doppler tracking parameters are the additional human interactions with the system due to the antenna's nature. These interactions can cause inconsistencies and increase the room for error, which could explain the results from this experiment.

Table 7.6: Overview of average performance parameters for Research Question 2.4. Lower values are better, except for SNR and Data quality.

Variable Average	**DopTrackBox**	**DopTrack**	**Difference**	**Percentage Difference**
SNR [dB]	4.369	5.363	-0.994	-18.5%
Data quality	18.9%	33.1%	-14.1%	-42.8%
RMS RRD [m/s]	174.4	26.9	147.4	547.2%
Absolute ΔTCA [s]	3.88	0.43	3.45	802.9%
Absolute Linear slope [m/s²]	0.490	0.050	0.440	880.4%

7.6. DATA ANALYSIS CONCLUSIONS

All the information from the previous sections can be used to answer Research Question 2; *How do the Doppler and SNR data of DopTrackBox compare to that of DopTrack?*.

From Research Question 2.1 and Section 7.2 it can be concluded that the DopTrackBox hardware is perfectly capable of performing satellite Doppler tracking. The experiments part of this subquestion provide the closest comparison between the two systems, as they are done at the same location using the same antennas. The obtained RMS range rate differences between the measured data and the TLE orbits are 23.2 m/s smaller than DopTrack on average when using the Red-Yellow antenna. For the Green antenna the RMS range rate

difference is 14.2 m/s larger on average than DopTrack's. Overall, this results in the RMS range rate difference being 8.4 m/s smaller in DopTrackBox' case. A similar difference can be seen in the ΔTCA, which is 0.74 s smaller for the Red-Yellow antenna and 0.63 s larger for the Green antenna compared to DopTrack. The observed linear drift of the Red-Yellow antenna is 0.037 m/s² lower, and for the Green antenna 0.020 m/s² higher. The SNR difference between the two systems also points in DopTrackBox' favour, being on average 0.618 dB higher than DopTrack.

The other three subquestions all involve data acquired whilst manually pointing and individually looking at certain aspects of the hardware. These are the GPS clock, the SDR, and the manually pointed antenna. Of these three the effect of the antenna seems to have the greatest impact on the quality of the data, as was discussed in Section 7.5. The SNR data is less consistent, resulting in a mean SNR being between 0.372 and 1.628 dB lower compared to DopTrack. Over the three data sets, this results in an average decrease of 0.994 dB with respect to DopTrack. The Doppler shift data consists on average of 14.1% fewer data points when comparing all data acquired whilst manually pointing with DopTrack. The RMS range rate differences themselves are on average 147.4 m/s greater compared to DopTrack and 113.7 m/s greater with respect to DopTrackBox using the omnidirectional antennas. The ΔTCA is on average 3.5 s larger compared to DopTrack. Data sets with a larger ΔTCA also have a larger range rate difference maximum.
The effect of the satellite's elevation seems to be second to the effects caused by manually pointing. Of two data sets with a maximum elevation of 76° the first has a mean SNR of 4.6 dB whereas the second has a mean SNR of 2.6 dB. Data set 19 with a maximum elevation of 37° has a mean SNR of 4.2 dB, which points to higher elevations not necessarily resulting to a higher SNR. For the range rate difference, lower maximum elevations of 12° and 23° resulted in no data at all and an RMS range rate difference of 292 m/s. Two data sets, both with a maximum elevation of 33° and almost identical passes, show an RMS range rate difference between the two of 641 m/s. This again points to inconsistencies being caused by a different factor than elevation.

The effect of using a GPS clock is not very clear. There is no improvement visible on the SNR data, as the average SNR decreases by 1.1 dB when comparing the data sets with GPS to those without GPS. Only two of the six data sets attain similar range rate differences as the two data sets without the GPS clock, being 41.9 m/s and 111.5 m/s. Looking at the overall average, a 67.6 m/s increase in RMS range rate difference is observed when including the GPS clock. The goal of the GPS clock is to aid in providing better timings, but the ΔTCA is 2.1 s greater on average with the GPS clock. An increase in the system's linear drift is also visible when using the GPS clock, resulting in a 0.358 m/s² increase.
It is important to reiterate again that a cause for this could be the asymmetrical nature of the data sets. Two favourable data sets without GPS could have been compared against six more average data sets with GPS. The obtained data points to the GPS clock not improving the performance of DopTrackBox.

A similar conclusion can be drawn from using a different SDR. Switching the SDR resulted in a 0.2 dB lower SNR, when two outliers are excluded, or a 0.8 dB lower SNR with those outliers included. The Doppler shift data has the same difficulty of an asymmetrical comparison as the GPS clock data. The data showed an outlier with a RMS range rate difference of 683.2 m/s, and a data set with a maximum range rate difference of 600.2 m/s. There are also two data sets with their RMS range rate difference at 92.6 and 42.2 m/s, being comparable to those using the RSPduo of 131.3 and 44.4 m/s. On average, the RSP1A has a RMS range rate difference of 143.8 m/s higher than the RSPduo. The timing difference and linear drift for the RSP1A are also greater than the RSPduo's, at 3.4 s and 0.654 m/s² respectively. This could point to the RSP1A performing slightly worse than the RSPduo, but similarly performing data sets are possible.

Experiments 1.1 and 1.2 are not part of any research question, but their conclusions provide some insight in the functioning of DopTrack and DopTrackBox. As DopTrack has two antennas, the differences between the two were observed. From the data it could be concluded that the Red-Yellow antenna provided better results than the Green antenna, both for the SNR and the range rate difference. The SNR on the Red-Yellow antenna was on average 0.893 dB higher than with the Green antenna and the RMS range rate difference was 15.6 m/s lower. It was also found that enabling the LNA is beneficial for data acquisition. The LNA resulted in more consistency for the Doppler shift data as only 25% of data sets without LNA yielded usable data, compared to 100% of data sets with LNA. Using the LNA also resulted in a 0.608 dB higher SNR.

The answer to Research Question 2 can be described as the following. Depending on the exact configuration of DopTrackBox, DopTrackBox' data is on par with DopTrack. The factor causing the biggest difference can be attributed to manually pointing the antenna, which increases measured range rate differences, ΔTCA, and linear drift, thus decreasing the system's performance. From this it can be concluded that DopTrackBox is capable of satellite Doppler tracking, just as the bigger DopTrack system.

8 | Improvements

With the data analysed and the second research question answered, a good understanding of DopTrackBox has been formed. This understanding of the system can be used to pinpoint areas of improvement, which is the main goal of this chapter. Along this the third and final research question will be discussed here: *Which improvements can be made to enhance DopTrackBox?*

The improvements for DopTrackBox will be discussed in Section 8.1. These will be based on the experience gained with DopTrackBox, the results from the first two research questions, and the system requirements. After the system improvements have been discussed, the finalised list of DopTrackBox requirements is presented in Section 8.2. These can be used for future reference. Thereafter, an overview of the cost of DopTrackBox is provided in Section 8.3. Next to improvements regarding the system, there is also room to reflect on improvements and recommendations for future research into DopTrackBox. These recommendations are given in Section 8.4. Finally, the findings from this chapter will be concluded in Section 8.5. This section will also provide the answer to Research Question 3.

8.1. DopTrackBox System Improvements

DopTrackBox has been developed during this thesis from a concept to a fully working system. There are still ways to improve the system and to streamline it even more. An important way to see how the system performed is by looking back at the requirements for DopTrackBox set at the beginning of this thesis. This will be done in Section 8.1.1. This compliance check can be used as a basis to look at future system improvements along with the other insights gained. Section 8.1.2 dives into this.

8.1.1. Requirement Updates and Compliance

The requirements were initially presented in Section 2.4 and amended in Section 5.3. Requirements that were amended in Section 5.3 are presented here with an asterisk. Some of the requirements are no longer needed due to changes made to the system. These have been listed in Table 8.1. The two OPS requirements have been removed due to a change in philosophy. With the tight integration between the different DopTrackBox software systems it is impractical to record unscheduled satellites, as it is more difficult to process the data at a later point in time. Technically DopTrackBox is still capable of recording unscheduled passes, but this feature has been removed from the base functionality of DopTrackBox. The online data back-ups are also still technically possible to do manually, but due to changes in the greater DopTrack support structure and the still early phase of DopTrackBox this has not been officially supported. The three TECH requirements have been removed due to a change in scope. *DTB-TECH-1.03* is no longer needed as a build in screen is available. It is still recommended to have a screen for future versions of DopTrackBox. *DTB-TECH-2.04* and *DTB-TECH-4.02* both relate to receiving UHF signals. The analysis of UHF signals was originally part of the proposed research questions, but this was scrapped to limit the scope of the research and due to the lack of DopTrack comparison data. These requirements can be re-used for future studies into UHF performance.

Table 8.1: Obsolete and removed DopTrackBox requirements.

Identifier	Requirement
DTB-OPS-2.02	DTB shall be able to record unscheduled satellite passes.
DTB-OPS-2.05	DTB shall be able to back-up its data to online servers when connected to the internet.
DTB-TECH-1.03*	The computer shall have the means to connect to a screen.
DTB-TECH-2.04	The SDR shall be capable of receiving UHF signals from 0.3 to 1 GHz.
DTB-TECH-4.02	The antenna shall be capable of receiving UHF signals from 0.3 to 1 GHz.

Other requirements were updated due to findings from the verification process and the real world experiences with the system. An updated requirement has a numerical suffix added to the original identifier, showing the iteration of the requirement. These changed requirements are listed in Table 8.2. The change itself is included in italics.

Table 8.2: Updated DopTrackBox requirements.

Identifier	Requirement
DTB-OPS-1.07.1	DTB shall include a system to receive electrical power. *Was 'shall include a battery'.*
DTB-OPS-2.06.1	DTB shall be able to operate on battery power for at least 60 minutes. *Changed 120 minutes to 60 minutes.*
DTB-TECH-1.01.1*	The computer shall have at least 1 USB type A port. *Was 'shall have at least 3 USB type A ports'.*

Requirement *DTB-OPS-1.07.1* has been updated to more broadly imply the need of a source of power for the system, as a battery is not necessarily a requirement on its own but a means of meeting this requirement. *DTB-OPS-2.06.1* was changed to 60 minutes to reflect changes made to the system. The main change is the switch from the Raspberry Pi to a Virtual Machine.

Beyond the newly added requirements in Section 5.3 regarding the Virtual Machine, there are no other newly added requirements. These requirements can be seen in Table 5.9 and will not be repeated here directly. They will be checked for compliance together with the other requirements and can therefore be seen in Table 8.11.

The remaining requirements will be checked for compliance. Each set of requirements has its own table. Here the identifier, description, and status are presented. The status symbol meanings are the following: The "✓" status means the requirement is fully met. The "!" status means a requirement is partially met or cannot be verified for compliance. The "×" status means the requirement has been failed. If a requirement has the "!" or "×" status, an explanation will be provided below the table as for why this status has been given.

Table 8.3: Requirement pass of DTB-OPS-1.

Identifier	Requirement	Status
DTB-OPS-1.01	DTB shall be no larger than 0.45 × 0.30 × 0.10 m.	✓
DTB-OPS-1.02	DTB shall include a computer.	✓
DTB-OPS-1.03	DTB shall include an SDR.	✓
DTB-OPS-1.04	DTB shall include a GPS clock.	✓
DTB-OPS-1.05	DTB shall include an antenna.	✓
DTB-OPS-1.06	DTB shall include an external data storage medium.	✓
DTB-OPS-1.07.1	DTB shall include a system to receive electrical power.	✓

All requirements in Table 8.3 have been met.

Table 8.4: Requirement pass of DTB-OPS-2.

Identifier	Requirement	Status
DTB-OPS-2.01	DTB shall be able to record automatically scheduled satellite passes.	✓
DTB-OPS-2.03	DTB shall be able to operate without active internet connection during satellite passes.	✓
DTB-OPS-2.04	DTB shall be able to record satellite passes without crashing.	✓
DTB-OPS-2.06.1	DTB shall be able to operate on battery power for at least 60 minutes.	✓
DTB-OPS-2.07	DTB shall be able to store satellite data of at least 150 passes.	✓
DTB-OPS-2.08	DTB shall be able to record satellite data with a mean SNR of 3.0 dB or higher.	!
DTB-OPS-2.09	DTB shall be able to record satellite data with a RMS range rate difference of 100 m/s or lower.	!
DTB-OPS-2.10	The SNR difference between DTB and DT shall be no smaller than -3 dB.	✓
DTB-OPS-2.11	The RMS range rate difference between DTB and DT shall be no larger than 50 m/s.	!

Requirements *DTB-OPS-2.08*, *DTB-OPS-2.09*, and *DTB-OPS-2.11* all have been partially met, as can be seen in Table 8.4. The reason for this can be found in Chapter 7. For *DTB-OPS-2.08* most recordings the SNR threshold of 3.0 dB has been passed, but for some it has not. These are the recordings from Experiment 1 with the LNA off and some DTB Pro and DTB Light recordings. As the majority passed the threshold it was shown that it is possible to meet this requirement, but not under all circumstances. *DTB-OPS-2.09* has been met for all Experiment 1 recordings, but only for four out of 13 recordings for Experiment 2. Due to this, requirement *DTB-OPS-2.11* has only been met for some recordings of Experiment 2 but has been met for all recordings of Experiment 1.

Table 8.5: Requirement pass of DTB-TECH-1.

Identifier	Requirement	Status
DTB-TECH-1.01.1	The computer shall have at least 1 USB type A port.	✓
DTB-TECH-1.02	The computer shall have the capability of connecting to the internet.	✓
DTB-TECH-1.04	The computer shall have the means to connect to a Human Interface Device for interaction with the system.	✓
DTB-TECH-1.05	The computer shall have a minimum internal storage capacity of 32 GB.	✓
DTB-TECH-1.06	The computer shall be able to run Python 3.9 or newer.	✓
DTB-TECH-1.07	The computer shall be able to process a data stream of 2 MB/s.	✓

It is worth repeating that requirement *DTB-TECH-1.01.1* from Table 8.5 is the technical minimum, but the recommended number of USB ports is still three for more flexible operation of DopTrackBox. This allows all the relevant components to be connected to the system at all times instead of swapping them out.

Table 8.6: Requirement pass of DTB-TECH-2.

Identifier	Requirement	Status
DTB-TECH-2.01	The SDR shall have a connection for the antenna.	✓
DTB-TECH-2.02	The SDR shall be connected to the computer via a USB type A connection.	✓
DTB-TECH-2.03	The SDR shall be capable of receiving VHF signals from 30 to 300 MHz.	✓
DTB-TECH-2.05	The SDR shall be capable of sampling at a sample rate of 250 kHz.	✓

Table 8.7: Requirement pass of DTB-TECH-3.

Identifier	Requirement	Status
DTB-TECH-3.01	The GPS clock shall be physically compatible with the rest of the DTB system.	✓

Table 8.8: Requirement pass of DTB-TECH-4.

Identifier	Requirement	Status
DTB-TECH-4.01	The antenna shall be capable of receiving VHF signals from 30 to 300 MHz.	✓
DTB-TECH-4.03	The antenna shall be physically compatible with the SDR.	✓

Table 8.9: Requirement pass of DTB-TECH-5.

Identifier	Requirement	Status
DTB-TECH-5.01	The external storage shall not have any moving parts.	✓
DTB-TECH-5.02	The external storage shall be connected to the computer via a USB type A connection.	✓
DTB-TECH-5.03	The external storage shall have a minimum capacity of 256 GB.	✓

Table 8.10: Requirement pass of DTB-TECH-6.

Identifier	Requirement	Status
DTB-TECH-6.01	The Operating System shall be based on Linux.	✓
DTB-TECH-6.02	The recorder software shall be compatible with the SDR.	✓

Table 8.11: Requirement pass of DTB-TECH-7.

Identifier	Requirement	Status
DTB-TECH-7.01	The VM shall be capable of USB passthrough.	✓
DTB-TECH-7.02	The VM shall be capable of running a Linux based Operating System.	✓
DTB-TECH-7.03	The VM shall have a minimum allocated storage capacity of 32 GB.	✓
DTB-TECH-7.04	The VM shall have a minimum allocated RAM capacity of 3 GB.	✓

All requirements from Tables 8.5, 8.6, 8.7, 8.8, 8.9, 8.10, and 8.11 have passed.

8.1.2. FUTURE IMPROVEMENTS

When looking at the requirement compliance checks, the updated DopTrackBox system used during the thesis meets most of the high level requirements. This does not mean that there is no room for improvements. Some of these improvements will also tie into the recommendations for further research into DopTrackBox, which are discussed in more detail in Section 8.4.

One of the largest changes to how DopTrackBox functioned during this thesis and how it could work in the future is the change back to a dedicated hardware computer. Even though a Raspberry Pi 3B is not powerful enough this does not exclude other computers, maybe even those of similar size, from working with DopTrackBox. An important feature to keep in mind is the number of available USB ports, such that all peripherals can be connected. The change to a virtual machine was one out of necessity, so going back to the original design goal could improve DopTrackBox by having a better integrated hardware system. Due to the standalone nature of the virtual machine a tightly integrated hardware package could not be created. This is the 'Box' reference in DopTrackBox' name. More tightly integrating DopTrackBox in this way would also improve the portability of the system to make it easier to do in field measurements. Some of the challenges and issues that arose from using the virtual machine could be avoided by switching back to dedicated hardware.

When looking at the results and the requirements tied to the results, there is room for improvements. Especially when looking at Experiment 2 and the data obtained from manually pointing. This configuration has generally performed worse with a lower SNR and a higher range rate difference. Since manually pointing in the field is one of the intended functions of DopTrackBox, two ways of improving this feature have been established.
The first is to change the directional Yagi antenna for a portable omnidirectional antenna with LNA. DopTrackBox had a lower SNR and a lower data availability for Experiment 1 without the LNA, so having one is highly advised. The LNA would be another component that needs to be powered, which could drive DopTrackBox' future design to have a larger battery. Something which could be ignored during this thesis due to the change from the Raspberry Pi to a laptop with a Virtual Machine.
The second option is to try to reduce the interference caused by manually pointing the antenna. During the thesis this was done manually whilst holding the antenna. Putting it on a standard would decrease the surface area of the antenna which is in contact with something holding it. The antenna could also be pointed more steadily and more accurately when on a standard, especially when a build in compass and inclinometer are present. This would also have the advantage of not needing an LNA, but the standard itself would potentially decrease the portability of the system. This could be further improved by making the antenna automatically track the passing satellite, but this would greatly increase the system's complexity.

8.2. DOPTRACKBOX FINALISED REQUIREMENTS

This section contains the DopTrackBox requirements that have passed the requirement updates in Sections 5.3 and 8.1.1, as well as the compliance check within the latter section. The specific reasoning behind the requirements were initially presented in Section 2.4, with the reasoning for the changes or newly added re-

quirements provided in the aforementioned sections. This will therefore not be repeated here. This section merely functions as an overview for future reference into DopTrackBox' requirements for further development.

The requirements have been split into operational requirements and technical requirements. The operational requirements can be seen in Table 8.12. The technical requirements are shown in Table 8.13.

Table 8.12: Final Operational Requirements (DTB-OPS) for DopTrackBox.

Identifier	Requirement
DTB-OPS-1.01	DTB shall be no larger than 0.45 × 0.30 × 0.10 m.
DTB-OPS-1.02	DTB shall include a computer.
DTB-OPS-1.03	DTB shall include an SDR.
DTB-OPS-1.04	DTB shall include a GPS clock.
DTB-OPS-1.05	DTB shall include an antenna.
DTB-OPS-1.06	DTB shall include an external data storage medium.
DTB-OPS-1.07.1	DTB shall include a system to receive electrical power.
DTB-OPS-2.01	DTB shall be able to record automatically scheduled satellite passes.
DTB-OPS-2.03	DTB shall be able to operate without active internet connection during satellite passes.
DTB-OPS-2.04	DTB shall be able to record satellite passes without crashing.
DTB-OPS-2.06.1	DTB shall be able to operate on battery power for at least 60 minutes.
DTB-OPS-2.07	DTB shall be able to store satellite data of at least 150 passes.
DTB-OPS-2.08	DTB shall be able to record satellite data with a mean SNR of 3.0 dB or higher.
DTB-OPS-2.09	DTB shall be able to record satellite data with a RMS range rate difference of 100 m/s or lower.
DTB-OPS-2.10	The SNR difference between DTB and DT shall be no smaller than -3 dB.
DTB-OPS-2.11	The RMS range rate difference between DTB and DT shall be no larger than 50 m/s.

Table 8.13: Final Technical Requirements (DTB-TECH) for DopTrackBox.

Identifier	Requirement
DTB-TECH-1.01.1	The computer shall have at least 1 USB type A port.
DTB-TECH-1.02	The computer shall have the capability of connecting to the internet.
DTB-TECH-1.04	The computer shall have the means to connect to a Human Interface Device for interaction with the system.
DTB-TECH-1.05	The computer shall have a minimum internal storage capacity of 32 GB.
DTB-TECH-1.06	The computer shall be able to run Python 3.9 or newer.
DTB-TECH-1.07	The computer shall be able to process a data stream of 2 MB/s.
DTB-TECH-2.01	The SDR shall have a connection for the antenna.
DTB-TECH-2.02	The SDR shall be connected to the computer via a USB type A connection.
DTB-TECH-2.03	The SDR shall be capable of receiving VHF signals from 30 to 300 MHz.
DTB-TECH-2.05	The SDR shall be capable of sampling at a sample rate of 250 kHz.
DTB-TECH-3.01	The GPS clock shall be physically compatible with the rest of the DTB system.
DTB-TECH-4.01	The antenna shall be capable of receiving VHF signals from 30 to 300 MHz.
DTB-TECH-4.03	The antenna shall be physically compatible with the SDR.
DTB-TECH-5.01	The external storage shall not have any moving parts.
DTB-TECH-5.02	The external storage shall be connected to the computer via a USB type A connection.
DTB-TECH-5.03	The external storage shall have a minimum capacity of 256 GB.
DTB-TECH-6.01	The Operating System shall be based on Linux.
DTB-TECH-6.02	The recorder software shall be compatible with the SDR.
DTB-TECH-7.01	The VM shall be capable of USB passthrough.
DTB-TECH-7.02	The VM shall be capable of running a Linux based Operating System.
DTB-TECH-7.03	The VM shall have a minimum allocated storage capacity of 32 GB.
DTB-TECH-7.04	The VM shall have a minimum allocated RAM capacity of 3 GB.

8.3. COST OVERVIEW

One of the original design ideas for DopTrackBox is for it to be as widely available as possible. For this it was established to use COTS components. Another factor to increase availability is to reduce the cost of making DopTrackBox. This section provides a preliminary cost overview of all the major components used in the current version of DopTrackBox.

Most parts of DopTrackBox will be used in any configuration of the system. The large variable is the computer itself, since the current version of DopTrackBox uses a VM. The key components are the SDR, antenna, GPS clock, and external SSD. For these components it is easy to find a price.
The configuration used to determine the price is DopTrackBox Pro, using the RSPduo SDR and GPS clock. The external SSD is the same for all configurations and is around €70 as was discussed in Section 2.5. The price of the SDR is obtained from SDRPlay's website. Since no VAT is included on the website, the price has been increased by 21% to reflect the VAT of the Netherlands. The RSPduo costs €310 whereas the RSP1A is available for €133 [36]. The exact antenna could not be found, but similar handheld VHF and UHF antennas are available online around the €100 price point. The price for the GPS clock can be found on the Leo Bodnar website [37]. Here it is for sale for £110 or €138 at the time of writing. The GPS clock is only used for the DTB Pro configuration. To provide power to the GPS clock an additional battery was used due to the limited amount of USB ports on the host computer. This is a standard USB power bank, which are available from around €25.

The host computer used is more expensive than a computer has to be to fulfil DopTrackBox' intended functions. It would therefore skew the system's price. Nevertheless, it has been provided here for completeness. The computer, as specified in Table 5.4, can no longer be bought as new but costed around €1800 at the time of purchase. A complete overview of all component prices is presented in Table 8.14.

Table 8.14: Overview of hardware component prices.

Component	**Price (€)**
Computer	1800
External SSD	70
SDR	310
Antenna	100
GPS Clock	138
Battery	25

When tallying up the numbers of Table 8.14, the total system price of the DopTrackBox system used during this thesis would be €2443, or €643 without the computer. When using the DTB Light configuration with only the RSP1A, the price of the system would be €303 excluding the computer.

The original idea of using the Raspberry Pi 3B would put the total system price at €693, as the Raspberry Pi 3B was sold for €50 [13]. Future testing should point out which computer can be used for DopTrackBox. The newer Raspberry Pi 4B is available from €45 for the 1 GB RAM version or €89 for the 8 GB RAM version [38]. However, it cannot be said for certain that the Raspberry Pi 4B is capable of working as DopTrackBox' computer. The new Raspberry Pi 5, which was unveiled on 28 September 2023 and is expected to launch at the end of October 2023, can also be a substitute. The Raspberry Pi 5 is available from €70 for 4 GB RAM or €93 for 8 GB RAM [38, 39]. Small desktop PCs with specifications similar to those of the computer used during this thesis can be found around the €250 price point. It should be noted that these do not have build in screens, batteries, and peripherals. Laptops with similar specifications should also be available around the €500 price point. A dedicated machine for DopTrackBox should have lower required specifications than a machine running DopTrackBox in a VM.

It is hard to tell what the exact price of DopTrackBox will be. DopTrackBox can be run as a VM on any available hardware with sufficient specifications, or an older device can be transformed into a dedicated computer for DopTrackBox. Further research should point out if a dedicated small computer can be used as part of DopTrackBox.

8.4. RECOMMENDATIONS

This thesis has achieved many of the original goals set out for it to achieve. During the setup and experimentation phases some of the initial goals were cut back to reduce the scope of this thesis. As is common in research, new questions arise when old ones are answered. Some other questions posed by this thesis need more data to provide a definitive answer for. This section provides recommendations for future research on DopTrackBox.

The recommendations can roughly be divided into three categories. The first category is deepening the work of this thesis by obtaining more of the same data to help provide clearer answers and is presented in Section 8.4.1. This category would receive the highest priority as these recommendations help to contextualise the answers provided within this thesis. The second category is broadening the work done by looking at DopTrackBox' functionality extensions and can be found in Section 8.4.2. Recommendations within this category are aimed at possible further research using this thesis as a groundwork. The third category encompasses software improvements to increase the functionality of DopTrackBox in that way. These can also be applied to DopTrack and have less of an academic priority. These are outlined in Section 8.4.3. With all the recommendations a plan for the continuation of research into DopTrackBox can be made. This can be found at the end of this section in Section 8.4.4.

8.4.1. DEEPENING RESEARCH

From the conclusion of Chapter 7 some recommendations can be made. A few of the effects of changing DopTrackBox' hardware are unclear, mainly the inclusion of the GPS clock and the different SDR. To gain a completer picture, more data is needed to eliminate the bias that can occur from having smaller numbers of data sets. These are more apple to apple experiments, where only a single variable is changed between them to get a good idea on the variable's effect. As Doppler tracking is the main focus of DopTrackBox, this should primarily be additional Doppler shift data. This would include range rate differences, ΔTCA, and the linear drift of the system. It is also important to have balanced data sets, with an approximately equal amount of data on either side of the comparison. Since these are fundamental questions regarding the functioning of DopTrackBox at this time, future research should acquire more data using the same processes as described in this thesis. This could help clear up the uncertainty surrounding the effect of these pieces of hardware. A way to get the most even data sets for comparison would be to have multiple DopTrackBox systems with a single difference between them. This would mean that a single satellite pass can be recorded by a system using the DTB Pro, DTB, DTB Light, and DopTrack system configurations. This is more labour intensive, especially when manually aiming, as each system needs to be operated by a human operator. Unless the same antenna input can be fed to all systems at once.

Research Question 1.2 'What are the hardware requirements of DopTrackBox?' was answered, but a follow-up question can be created. This was also partly discussed in Section 8.3 and would encompass the minimum required computer specifications to successfully run DopTrackBox. Finding this out can help clear up the uncertainty surrounding the price of DopTrackBox and what kind of computer hardware is required to run DopTrackBox. This would also shed a light on the viability of using a more powerful Raspberry Pi as the computer for DopTrackBox.

Another possibility to create parity between DopTrackBox and DopTrack for comparison's sake is using a manually aimed, handheld antenna for DopTrack. This provides a more even comparison between the data collected by DopTrackBox and DopTrack whilst manually pointing the antenna. The same challenges for the comparison of different data sets would still be in place.

8.4.2. BROADENING FUNCTIONALITY

One of the original goals was to look at DopTrackBox' UHF recording Doppler shift performance. This goal was scrapped due to time constraints and scope limitations. It is still a valid point for further research, which can be compared against DopTrack as DopTrack is also starting to gather UHF satellite data from Delfi-PQ. Knowing whether DopTrackBox is capable of Doppler tracking using UHF signals can further solidify DopTrackBox' position as a portable Doppler Tracking ground station.

An effect that was noticed during the discussion of DopTrackBox' results is the disparity between the use of an LNA on DopTrack's data whilst DopTrackBox uses none. An interesting idea is to look into the effect of including an LNA on DopTrackBox' manually pointed data to see the effects on the Doppler shift data.
Something similar to the previous recommendation is to use different antennas. Especially those with different gain patterns to see what the effect of different antennas is and whether there is an ideal antenna for

different scenarios. A next step in this could be to let the antenna automatically track the satellite whilst it is passing. However, this would greatly increase the system's complexity.

It is also recommended to look at the possibility of tightly integrating the entire DopTrackBox system in a small portable package. This was another original goal which was abandoned when the switch was made from the Raspberry Pi to the VM. Going back to a small dedicated computer should make this possible again. Having the entire system tightly integrated could cause new challenges when different systems interfere with each other or when heat dissipation might become an issue.

8.4.3. SOFTWARE IMPROVEMENTS

There are also some recommendations regarding software improvements. These do not necessarily expand upon DopTrackBox' functionality, but could increase ease of use. Since DopTrackBox uses the same software as DopTrack, these recommendations can also be used to improve DopTrack.

A recommended improvement in the scheduling software is the inclusion an input for ground station parameters. When only a single ground station is used, as is the case with DopTrack, this is not needed. However, since DopTrackBox is portable having an easy solution to change the ground station coordinates and elevation constraints can be a big quality of life improvement.

A smaller software improvement is the automation of the data processing after the recording. Currently, the processing has to be manually initiated on DopTrackBox. Having it occur automatically reduces the user interaction with the system. If there are several recordings in close succession new recordings should be prioritised. If the system is powerful enough to handle both a recording and data processing at the same time, these tasks could be executed simultaneously. However, this of course depends on the computer of DopTrackBox.

8.4.4. RESEARCH CONTINUITY

This section describes which steps should be taken to advance the research into DopTrackBox, which corresponds roughly with the order of Section 8.4. First, the effects of the GPS clock and SDR should be further investigated as this will provide insight in which components should be used for DopTrackBox in the future. Once this has been done, several paths can be chosen depending on whether DopTrackBox' current functionality is sufficient. The recommendation is to look into DopTrackBox' computer and find the minimum required specifications as this also helps with future research into the portability of the system.

After this step, the functionality of DopTrackBox can be broadened by looking at the suggestions in Section 8.4.2. Looking at UHF performance was one of the original goals, as well as tightly integrating the system. The effect of using an LNA on the handheld antenna and looking at the effects of different antennas could be reserved for later.

The recommendations regarding software improvements have the lowest priority, but can also be done during DopTrack development as both systems use the same software. The improvements made for DopTrack can then trickle down to DopTrackBox.

8.5. DOPTRACKBOX IMPROVEMENT CONCLUSIONS

All the improvements and recommendations from this chapter can be used to answer Research Question 3: *Which improvements can be made to enhance DopTrackBox?*

All but a few requirements set for DopTrackBox have been passed. Those that have not were related to performance metrics, where only some of the acquired data met the threshold. The performance improvements for DopTrackBox should thus address these requirements. Most of the acquired data that did not meet the set threshold was part of Experiment 2, where pointing manually was identified as the largest culprit. This can be avoided by either using an omnidirectional antenna with LNA or by reducing the interference caused by manually pointing. The latter can be done by using a standard to more steadily point the antenna.

From the acquired data it is unclear what the effect of the GPS clock and different SDR are. This is mainly caused by the effect of manually aiming the antenna, which introduced a lot of variance. When looking at the price of these subsystems, they can be quite significant. Therefore it is advised to continue research for DopTrackBox by gathering more data to see whether a definitive effect can be discovered. This can be used to see if the full DTB Pro setup is required, which costs €643 without computer, or if the simpler DTB Light would suffice. The DTB Light system only costs €303 in comparison. The computer's price is still an

unknown, as the exact minimum specifications for DopTrackBox are not known. Further research could shed a light on this, which would help to determine the full system's price.

After these initial research continuation suggestions aimed to deepen the research of this thesis, the focus can be shifted to broadening DopTrackBox' functionality. Examples are looking at UHF signals, using an LNA on DopTrackBox' manually pointed antenna to see what the effects are, and looking at possibilities of tightly integrating DopTrackBox as a complete system. There are also some software improvements that can be implemented in the future, which would also benefit DopTrack. These are the inclusion of an input for ground station parameters in the scheduling software, as DopTrackBox is a portable system. This would make it easier to move locations and easily change the parameters without having to go into the code to change the hardcoded variables. Another small improvement is to automatically enable DopTrackBox to process the data once it has finished recording it.

To answer Research Question 3, several improvements can be made. The negative effect on the data whilst manually pointing can be reduced by either using an omnidirectional antenna with LNA or by possibly stabilising the antenna. Some quality of life improvements for DopTrackBox' software are the ability to easily change ground station parameters and to automatically start data processing after data acquisition.

9 | Conclusion

This report aimed *To analyse DopTrackBox' viability as a Doppler tracking ground station.*

To achieve this objective, three research questions were formed with their various subquestions. This chapter provides the overall conclusion of the thesis and is broken down into three sections. First, Section 9.1 concludes the process behind the setup of DopTrackBox and the data acquisition. Second, Section 9.2 provides the conclusion of all the research questions and reflects on the thesis' objective. Third, Section 9.3 reflects upon the research objective using all the previous findings of this chapter.

9.1. DopTrackBox Set-up

Different versions of DopTrackBox were tested to see what the effect of changing a specific hardware component is. Throughout the research, the hardware used for DopTrackBox was changed to reflect newly gained insights. The main subsystem that had to be changed in this way was the computer, as the original Raspberry Pi 3B is not powerful enough to run DopTrackBox' software. Two sets of data were used as performance metrics, those being Signal to Noise ratio (SNR) and range rate difference. The range rate difference was obtained by comparing the measured data with the two-line element set (TLE) predicted orbit. From the range rate difference, other characteristics can be derived. The Time of Closest Approach (TCA) and linear slope of the range rate residual are two of these characteristics which were used during data analysis. The TCA of the measurement is compared against the TCA of the TLE to gain insight into the system's timing. The slope of the range rate residual provides information into the system's linear frequency drift.

The satellite used for this thesis was Nayif-1 due to its favourable pass timings, number of passes, signal strength, and reliability. As of finishing this thesis, Nayif is no longer operational. As the virtual machine's host computer only has a single USB port, only the SDR and antenna could be connected. When all data was recorded, the SSD was connected to offload the data. When recording data with a directional antenna the signal's polarisation has to be established. This is done by changing the orientation of the antenna from horizontal to vertical every two minutes. If the signal is polarised, the signal's strength should be significantly higher when the antenna is aligned correctly. After the first two recordings of Nayif data it was found out that Nayif's data was not polarised and the antenna could be held in the most comfortable way to reduce human interference.

Experiments 1.1 and 1.2 are not part of any research question, but have provided insight into DopTrack's antennas. These experiments used both the Green and Red-Yellow antenna and looked at the effect of using the LNA.

The data showed that using the LNA is favourable. The average SNR without LNA is 2.728 dB, whereas enabling the LNA increased the SNR by 0.608 dB for the same antenna. Using the LNA also increases the reliability to extract Doppler shift data, as only one out of four data sets without LNA contains useful data compared to all seven data sets with LNA.
The Red-Yellow antenna also performed better than the Green antenna, yielding a higher average SNR of 4.229 dB compared to 3.336 dB and lower range rate difference; 54.2 m/s compared to 69.8 m/s.

9.2. Research Questions

This section provides a comprehensive overview of the conclusions of the three research questions. Each research question is discussed in its own section. Research Question 1 and its two subquestions can be found in Section 9.2.1. The conclusion to Research Question 2 along with the respective four subquestions is presented in Section 9.2.2. The conclusion of Research Question 3 is provided in Section 9.2.3.

9.2.1. CAN THE PROPOSED DOPTRACKBOX CONCEPT BE BUILD?

The answer to Research Question 1 was first provided in Section 5.3. To aid in answering Research Question 1, two subquestions were created.

Research Question 1.1 is stated as *Can the selected hardware combination record satellite data required for Doppler tracking?* At this point in time, the original DopTrackBox configuration with the Raspberry Pi 3B was still in use. During the verification and validation phase of DopTrackBox' hardware it was found out that the Raspberry Pi cannot record satellite data without crashing and causing data loss. Various different configurations using the Raspberry Pi were used, but none managed to fully record a satellite pass.

Research Question 1.2 is *What are the hardware requirements of DopTrackBox?* As the Raspberry Pi cannot be used, this question can grant insight in what kind of computational power is needed to run DopTrackBox. Due to scope and time limitations, the more interesting question into the minimum hardware requirements could not be answered. However, it can be stated that the virtual machine with the provided computational resources was able to fulfil the task of being the computer for DopTrackBox. The full details can be found in Table 5.6, but the most noteworthy are the 2 CPU threads @ 2.50 GHz, 3 GB of RAM, and 32 GB of storage. These hardware requirements can be used as the verified minimum requirements needed to run DopTrackBox, but these might not be the absolute minimum requirements. Next to the computer, other required hardware components are the SDR, antenna, the GPS clock, and an external battery.

The originally proposed DopTrackBox concept can thus not be build, with the main bottleneck being the Raspberry Pi. Even though this initial concept is not feasible, it has been proven that a different combination of hardware is capable of functioning as DopTrackBox.

9.2.2. HOW DO THE DOPPLER AND SNR DATA OF DOPTRACKBOX COMPARE TO THAT OF DOPTRACK?

Research Question 2 was discussed in full in various sections of Chapter 7. This is the largest research question, with four subquestions.

Research Question 2.1, *What is the difference between the SNR and Doppler shift data of DopTrackBox and DopTrack?*, used data from Experiment 1.3 by comparing omnidirectional antenna data from both systems. It was discussed in full in Section 7.2. During this experiment it was found out that DopTrackBox performed on par with DopTrack. The average SNR over all data sets is 3.529 dB for DopTrackBox, which is 0.618 dB higher than DopTrack's. The RMS range rate differences vary per antenna. On the Red-Yellow antenna the average is 54.2 m/s, 23.2 m/s lower than DopTrack's. When using the Green antenna, DopTrackBox has an average RMS range rate difference of 69.8 m/s. This is 14.2 m/s higher compared to DopTrack.
The SNR and RMS range rate differences of the two systems are closer together than initially expected. Since DopTrackBox uses cheaper hardware, greater differences between the systems were expected. There is thus no big difference when DopTrackBox is using the omnidirectional antennas. Therefore, DopTrackBox is perfectly capable of performing satellite Doppler tracking.

Research Question 2.2 is defined as *What is the effect of including or excluding a GPS clock on the SNR and Doppler shift data?*. It compared DopTrackBox' DTB and DTB Pro configurations. The full analysis was provided in Section 7.3. The effect of using a GPS is unclear, which is partly caused by the lacking number of data sets to draw a decisive conclusion. No benefit was found in the SNR data for including a GPS clock over excluding one. The average SNR of all data sets with GPS clock is 3.270 dB, which is 1.098 dB lower than the average SNR without GPS clock. The Doppler shift data suggested that using a GPS clock would increase the range rate difference by 67.6 m/s. This is the average of the various data sets. Without GPS clock the RMS range rate difference is measured to be 87.8 m/s. This seemingly large increase could be caused by data set imbalance. One of the key features of including the GPS clock is to improve the system's timing. However, when looking at the ΔTCA an increase of 2.1 s on average is visible when the GPS clock is included. Without GPS clock the ΔTCA is 1.6 s. A similar effect is visible with the linear drift, where the slope with GPS is 0.432 m/s^2. This is 0.358 m/s^2 higher than the configuration without GPS.
The conclusion drawn from the data is thus that the effect of including a GPS clock is a seeming increase in range rate difference, ΔTCA, and linear drift, whilst decreasing SNR.

Research Question 2.3, *What is the effect of using a different SDR for DopTrackBox on the SNR and Doppler shift data?*, was discussed in Section 7.4 and compared DopTrackBox' DTB and DTB Light configurations. The SNR was 0.2 dB lower with the RSP1A on average when two outliers are excluded due to their pass parameters. With these included the difference becomes 0.8 dB. There are bigger differences when looking at

the Doppler shift data, but data set imbalance could play a role here as well. The average RMS range rate difference using the RSP1A is 231.7 m/s, which is 143.8 m/s higher than using the RSPduo. This does include an outlier value of 683.2 m/s, but when comparing this pass to others there is no reason to exclude it. Similar observations can be made regarding the ΔTCA and linear drift. With the RSP1A the ΔTCA is 5.0 s, which is a 3.4 s increase, and the linear drift is 0.727 m/s², resulting in a 0.654 m/s² increase compared to the RSPduo. There are also data sets obtained with the RSP1A which have range rate differences in line with observations from the RSPduo, but there are more outliers as well.
The conclusion drawn from the available data is that similar performance between different SDRs, the RSPduo and RSP1A, is possible, but there are larger outliers with the RSP1A. The exact effect of using a different SDR is not clearly visible from the available data, but it does point to higher range rate differences.

Research Question 2.4 is *What is the effect of manually aiming for DopTrackBox on the SNR and Doppler shift data?*. The full contents of this question can be found in Section 7.5. The effect of manually aiming is significant. The data quality, defined as the number of data points in a data set with respect to the theoretic maximum, is 14.1% lower for DopTrackBox compared to DopTrack. DopTrack's average data quality is 33.1% for this experiment. DopTrackBox had a data quality during Experiment 1 of 22.1%, which is 3.2% higher than the manually aimed data of Experiment 2.
Compared to DopTrack, the SNR is on average 0.994 dB lower whilst manually aiming; resulting in an average 4.369 dB SNR. The RMS range rate difference is on average 147.4 m/s greater compared to DopTrack. This results in the average RMS range rate difference of 174.4 m/s for DopTrackBox whilst manually aiming.
The data from Experiment 2 also points to manually aiming being the cause for the observed data inconsistencies. Two different data sets with equal pass parameters and DopTrackBox configuration show a RMS range rate difference of 641 m/s. These were both at a maximum elevation of 33°. A pass at 12° maximum elevation resulted in a data set without any useful Doppler shift data, whereas a pass with an elevation of 23° had a higher than average 292 m/s RMS range rate difference. This points to lower elevations increasing the likelihood of less consistent Doppler shift data.
This can be confirmed when comparing data from Experiment 2 with that of Experiment 1. The average range rate difference is 113.7 m/s greater in Experiment 2's case.

DopTrackBox' system performance is both defined by the SNR data and the Doppler shift data. These can be comparable when the systems are the most equal, as was shown by Research Question 2.1 with Experiment 1. The largest factor which deteriorates data quality for DopTrackBox seems to be manually aiming. This is also the most likely factor that introduced inconsistencies resulting in the variance of the data for Research Questions 2.2 and 2.3. The findings of Research Question 2 reinforce DopTrackBox' ability to successfully perform satellite Doppler tracking.

9.2.3. WHICH IMPROVEMENTS CAN BE MADE TO ENHANCE DOPTRACKBOX?

Research Question 3 was answered in Section 8.5 and has no specific subquestions.

For further development of DopTrackBox it was advised to first get a better understanding of the effects of the GPS clock and a different SDR. To gain this understanding more data is needed, which will need to be obtained in future research. To return to the original idea of a portable Doppler tracking box, a different computer will need to be found. This ties into finding the minimum system requirements for DopTrackBox. The entire system could then be completely integrated. Effects expanding DopTrackBox' functionality can also be investigated. Examples are UHF satellite data or using an LNA on the manually pointed DopTrackBox antenna.

To improve DopTrackBox, the negative effect on the data whilst manually pointing needs to be reduced. This can be done by using an omnidirectional antenna with LNA or otherwise stabilising the antenna. Regarding software improvements, DopTrackBox could be changed to more easily change ground station parameters and automatically start processing the data after recording it.

9.3. RESEARCH OBJECTIVE REFLECTION

The objective, *To analyse DopTrackBox' viability as a Doppler tracking ground station*, has been completed. The original concept of a portable, fully integrated system with a Raspberry Pi was not viable, but DopTrackBox in its current iteration is fully capable of satellite Doppler tracking. More research is needed to look into the specific effects of using a GPS clock and using a different SDR. In the future, the findings from this thesis can be used to go back to the original vision and create the fully integrated DopTrackBox.

Bibliography

[1] P.G. Newman and G.S. Rozycki. The History of Ultrasound. *Surgical Clinics of North America*, 78(2):179–195, 1998.

[2] J.J. Lissauer and I. de Pater. *Fundamental Planetary Science*. Cambridge University Press, New York, 2013.

[3] D.C. Giancoli. *Physics for Scientists & Engineers with Modern Physics*. Pearson, Harlow, 2014.

[4] E.J.O. Schrama. Lecture notes on Planetary sciences and Satellite Orbit Determination. Faculty of Aerospace, Astrodynamics and Satellite missions, Delft University of Technology, January 2018.

[5] B.C. Root. *Satellite Tracking Practical Manual*. TU Delft, Delft, v1 edition.

[6] Radar Systems Panel. IEEE Standard Letter Designations for Radar-Frequency Bands. *IEEE Std 521-2019 (Revision of IEEE Std 521-2002)*, pages 1–15, 2020.

[7] O. Montenbruck and E. Gill. *Satellite Orbits*. Springer, Berlin, 2001.

[8] J. R. Wertz, H. F. Meissinger, L. K. Newman, and G. Smit. *Orbit & Constellation Design & Management*. Microcosm Press and Springer, New York, 2009.

[9] TU Delft Planetary Exploration. Facilities. Available online: https://www.delftplanetary.nl/facilities/. [Accessed 12 July 2023].

[10] TU Delft. DopTrack Dashboard. Available online: http://doptrack.tudelft.nl/. [Accessed 12 July 2023].

[11] W. de Leeuw. DopTrackBox assignment. Spaceflight Minor, TU Delft.

[12] M. Dillinger, K. Madani, and N. Alonistioti. *Software defined radio: architectures, systems, and functions*. Wiley, Hoboken, NJ, 2003.

[13] Raspberry Pi. Raspberry Pi 3 Model B. Available online: https://www.raspberrypi.com/products/raspberry-pi-3-model-b. [Accessed 24 January 2022].

[14] SDRplay. *Radio Spectrum Processor 1A*. SDRplay Ltd, Hampshire, v2.2 edition, August 2020.

[15] SDRplay. *RSPduo*. SDRplay Ltd, Hampshire, v0.9 edition, June 2020.

[16] SDRplay. RSP1A. Available online: https://www.sdrplay.com/rsp1a/. [Accessed 3 July 2023].

[17] SDRplay. RSPduo. Available online: https://www.sdrplay.com/rspduo/. [Accessed 3 July 2023].

[18] SDR-Kits. *Using SDR-Kits External Clock-in Cable with the SDRPlay*. SDR-Kits, Melksham, Wilts.

[19] W.D. Reeve. Using the SDRPlay SDR Receivers with an External Frequency Reference. November 2020.

[20] Giorgos. Waar let je op bij het kopen van een M.2 SSD? Available online: https://www.coolblue.nl/advies/waar-op-letten-m2-ssd.html. [Accessed 30 June 2023].

[21] Tweakers. Tweakers. Available online: https://www.tweakers.net. [Accessed 24 January 2022].

[22] ADATA. SD600Q externe Solid State Drive. Available online: https://www.adata.com/nl/specification/603?tab=specification. [Accessed 6 July 2023].

[23] rxseger. rx_tools. Available online: https://github.com/rxseger/rx_tools. [Accessed 8 June 2022].

[24] AsciiWolf. libsdrPlaySupport.so cannot find libsdrplay_api.so.3.07 #44. Available online: https://github.com/pothosware/SoapySDRPlay3/issues/44. [Accessed 1 March 2022].

[25] W. J. Larson and J. R. Wertz. *Space Mission Analysis and Design, 3rd ed.* Microcosm Press and Kluwer Academic Publishers, Dordrecht, 2005.

[26] K. Cheung. The role of margin in link design and optimization. In *2015 IEEE Aerospace Conference*, pages 1–11, 2015.

[27] AMSAT-UK. Nayif-1. Available online: https://amsat-uk.org/satellites/comms/nayif-1/. [Accessed 8 April 2023].

[28] ISISpace. Nayif-1. Available online: https://www.isispace.nl/projects/nayif-1/. [Accessed 8 April 2023].

[29] N2YO. Nayif (EO88). Available online: https://www.n2yo.com/satellite/?s=42017. [Accessed 8 April 2023].

[30] David G0MRF (AMSAT-UK). EO-88 (Nayif-1) Re-enters. Available online: https://amsat-uk.org/2023/07/19/eo-88-nayif-1-re-enters/. [Accessed 9 August 2023].

[31] bclswl0827 (Seunghun Lee). Failed to activate stream #68. Available online: https://github.com/rxseger/rx_tools/issues/68. [Accessed 20 February 2023].

[32] June Castillote. Enabling Hyper-V USB passthrough to Access a USB Storage. Available online: https://adamtheautomator.com/hyper-v-usb-passthrough/#Enabling_Server-Side_Hyper-V_USB_Passthrough_Access. [Accessed 27 September 2023].

[33] N2YO. 10-Day Prediction Nayif (EO88). Available online: https://www.n2yo.com/passes/?s=42017&a=1. [Accessed 12 July 2022].

[34] N2YO. 10-Day Prediction Delfi C3. Available online: https://www.n2yo.com/passes/?s=32789&a=1. [Accessed 12 July 2022].

[35] N2YO. 10-Day Prediction Delfi-N3XT. Available online: https://www.n2yo.com/passes/?s=39428&a=1. [Accessed 12 July 2022].

[36] SDRplay. Purchase. Available online: https://www.sdrplay.com/purchase/. [Accessed 6 September 2023].

[37] Leo Bodnar Electronics. Mini Precision GPS Reference Clock. Available online: http://www.leobodnar.com/shop/index.php?main_page=product_info&cPath=107&products_id=301&zenid=0883eb421be99784354455e43d41f491. [Accessed 6 September 2023].

[38] KIWI Electronics. Raspberry Pi Boards. Available online: https://www.kiwi-electronics.com/en/raspberry-pi-boards-cases-addons-and-accessories-59/raspberry-pi-boards-363. [Accessed 9 October 2023].

[39] Eben Upton. Introducing: Raspberry Pi 5! Available online: https://www.raspberrypi.com/news/introducing-raspberry-pi-5/. [Accessed 9 October 2023].

A | Additional Plots & Graphs

This appendix contains additional data plots of satellite data that was produced by DopTrackBox. This data was used to construct the plots from Chapter 6, but has not been presented there to save space.

A.1. Spectrograms and SNR plots for Experiment 1

This section contains the spectrogram, noise floor, and SNR plots of the data used in Experiment 1 from Section 6.2 that are not present elsewhere in the report.

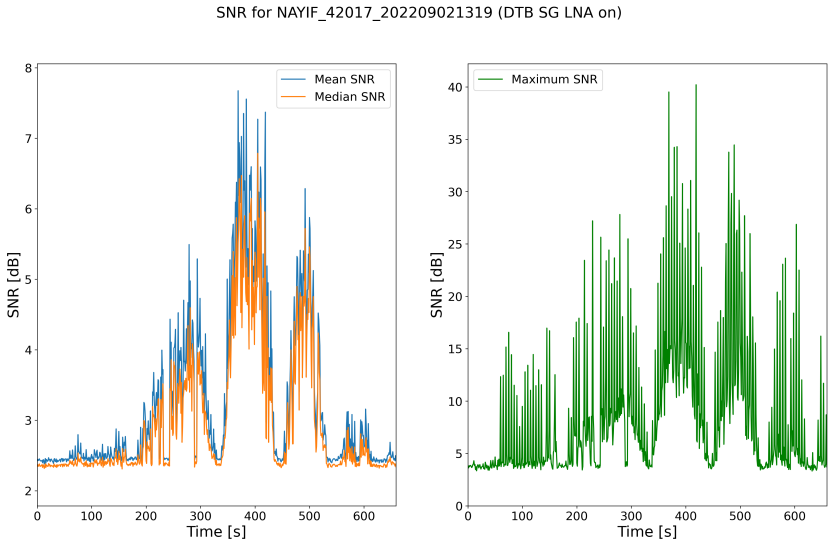
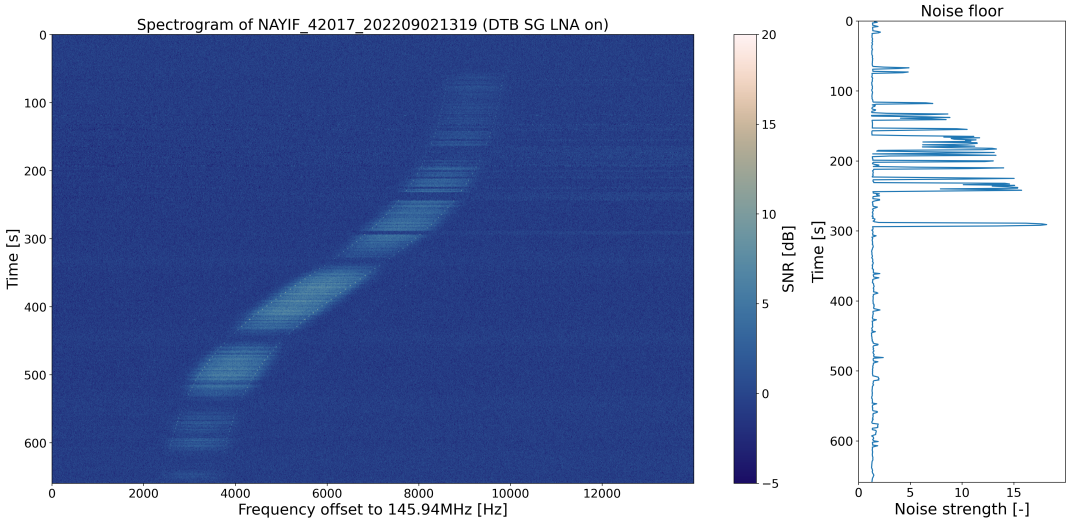


Figure A.1: Data set 2 (NAYIF_42017_202209021319) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.2: Spectrogram and noise floor for Data set 2 (NAYIF_42017_202209021319).

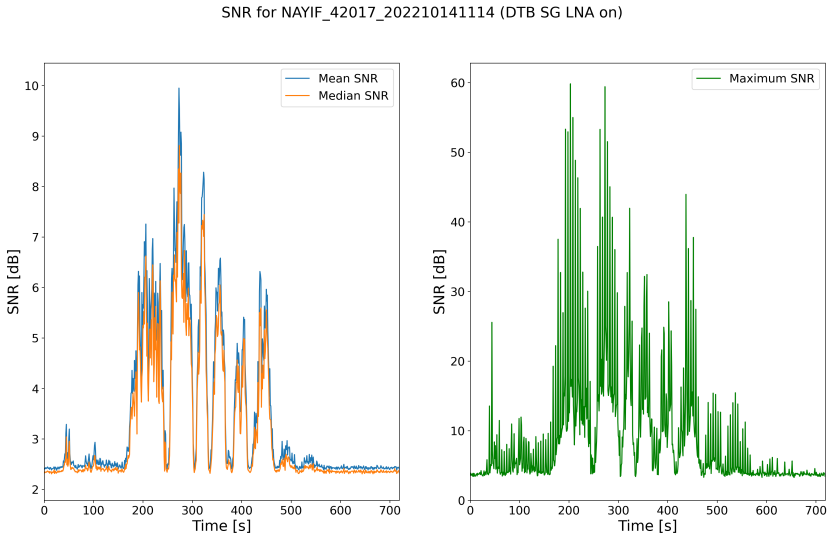
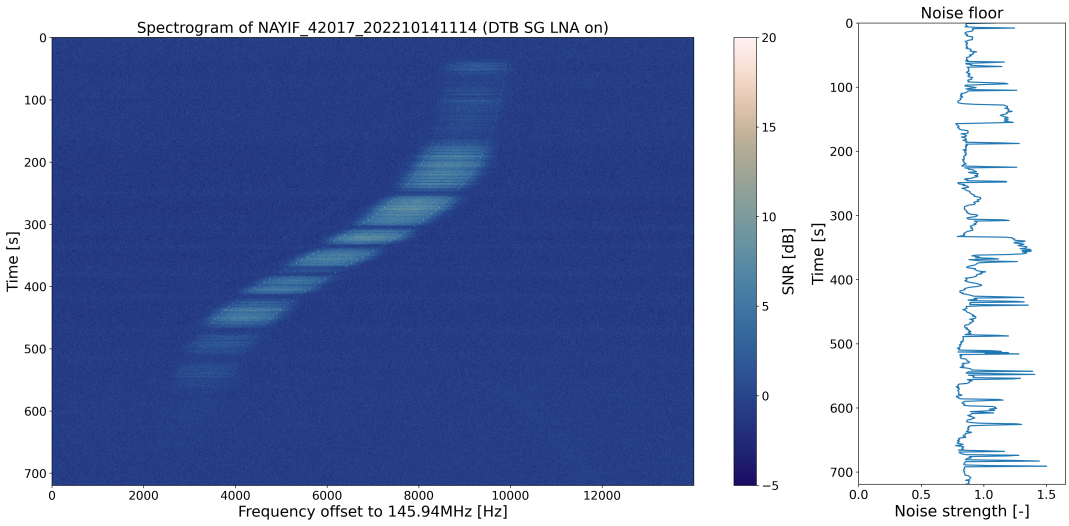


Figure A.3: Data set 3 (NAYIF_42017_20221014114) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.4: Spectrogram and noise floor for Data set 3 (NAYIF_42017_20221014114).

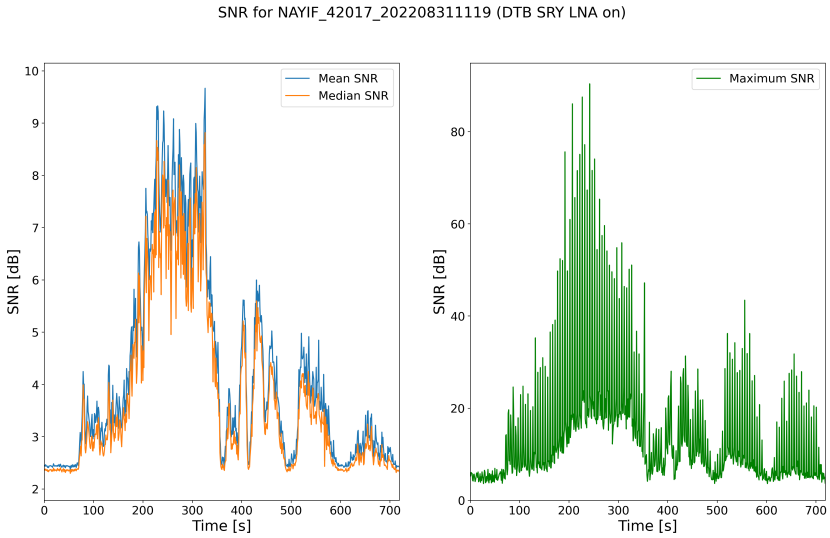
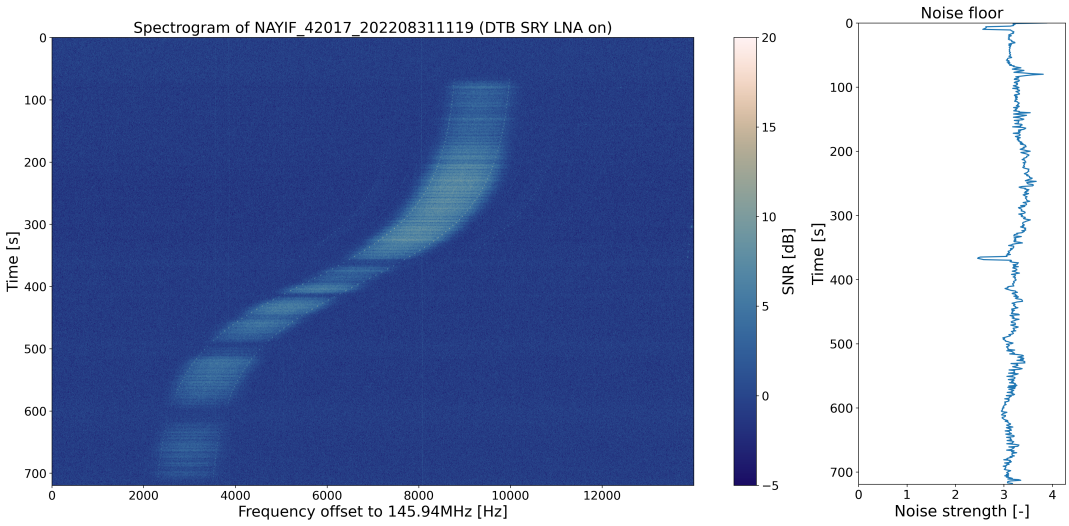


Figure A.5: Data set 4 (NAYIF_42017_202208311119) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.6: Spectrogram and noise floor for Data set 4 (NAYIF_42017_202208311119).

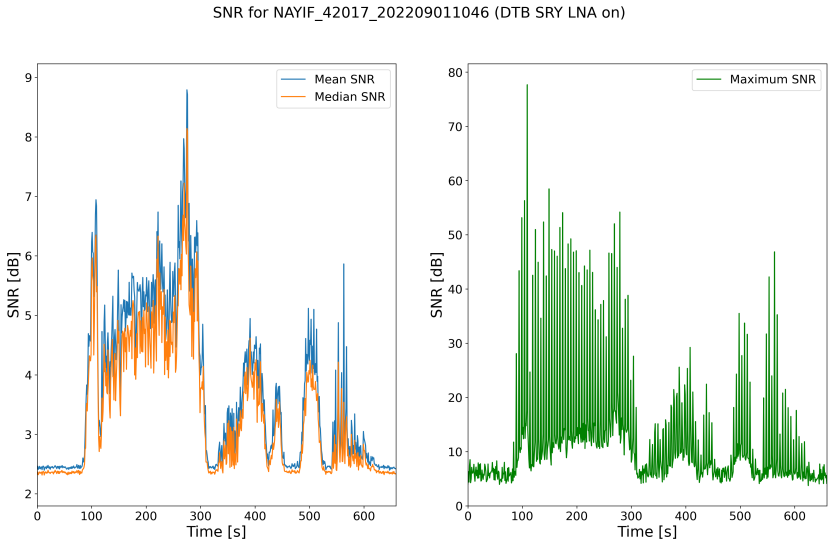
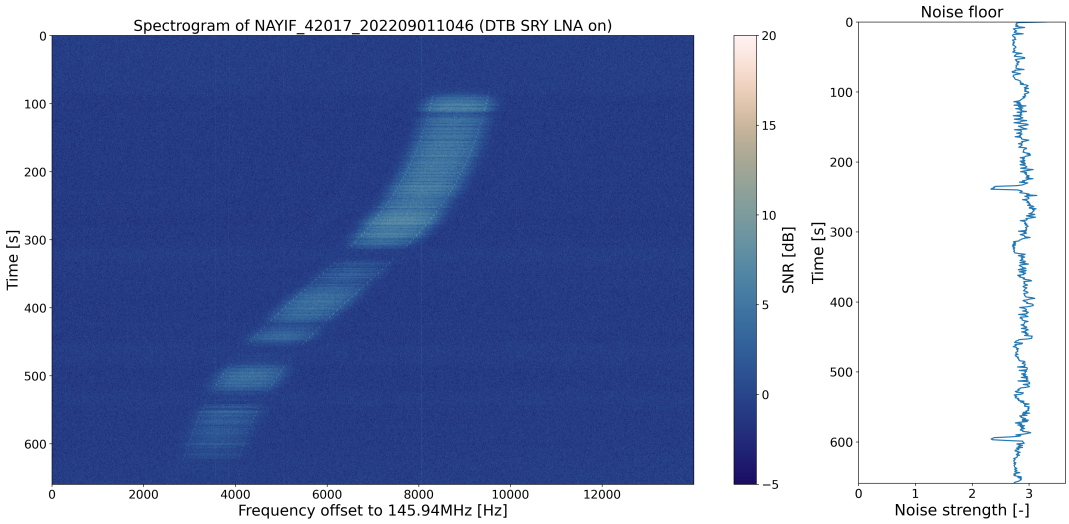


Figure A.7: Data set 5 (NAYIF_42017_202209011046) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.8: Spectrogram and noise floor for Data set 5 (NAYIF_42017_202209011046).

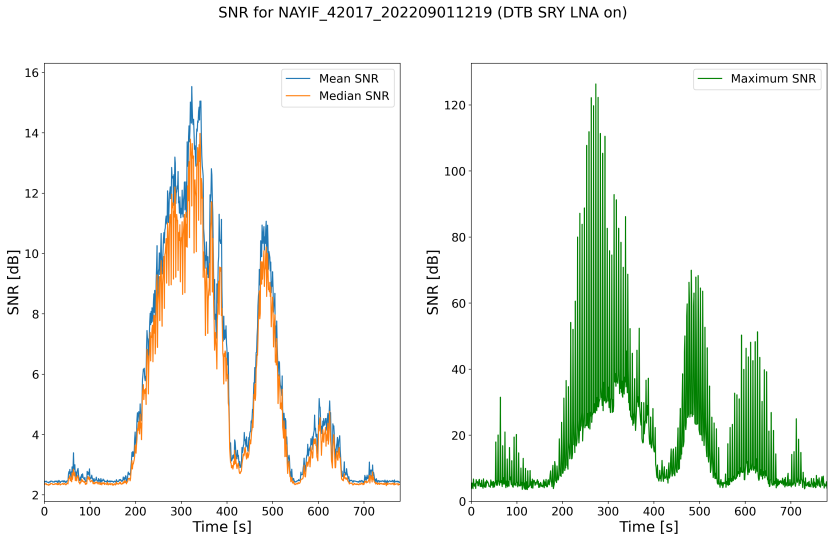
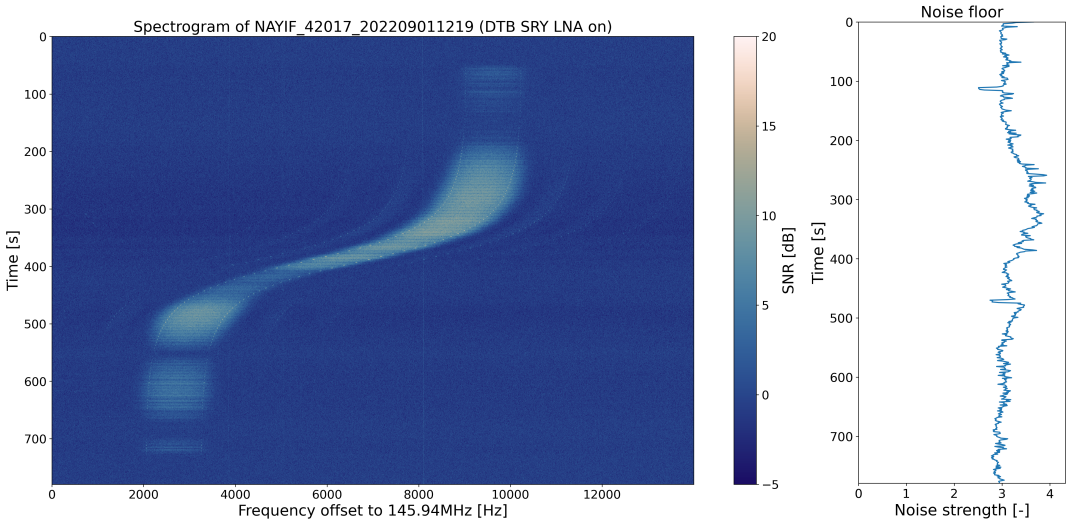


Figure A.9: Data set 6 (NAYIF_42017_202209011219) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.10: Spectrogram and noise floor for Data set 6 (NAYIF_42017_202209011219).

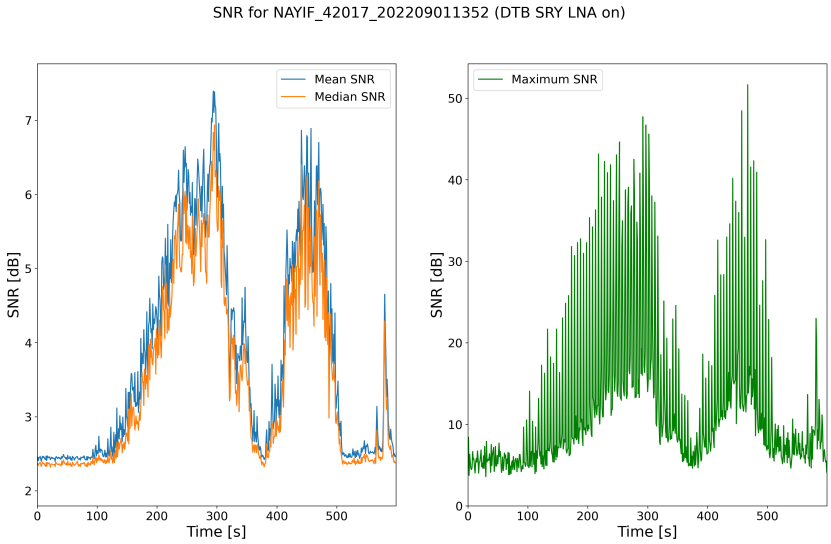
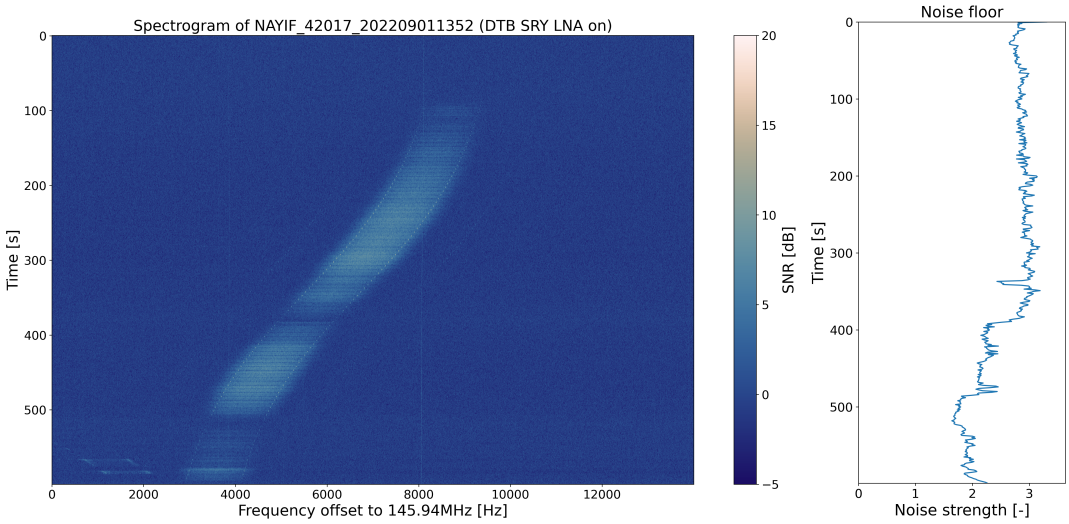


Figure A.11: Data set 7 (NAYIF_42017_202209011352) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.12: Spectrogram and noise floor for Data set 7 (NAYIF_42017_202209011352).

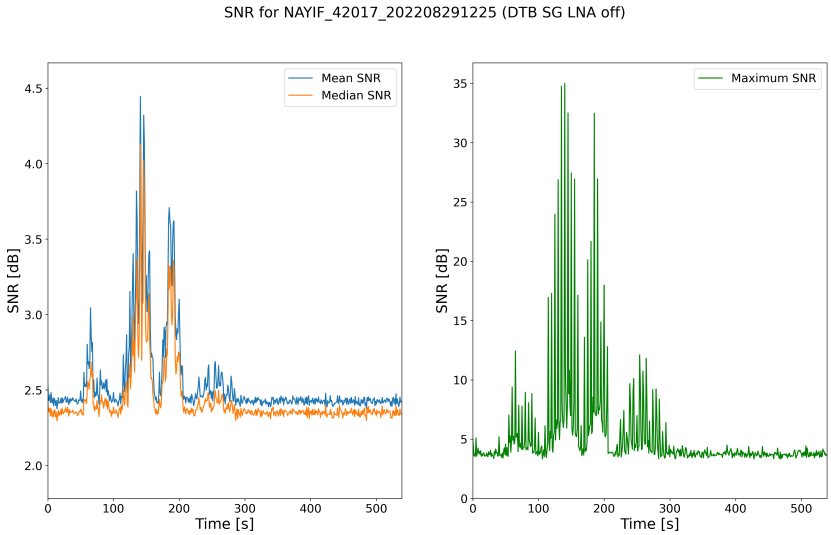
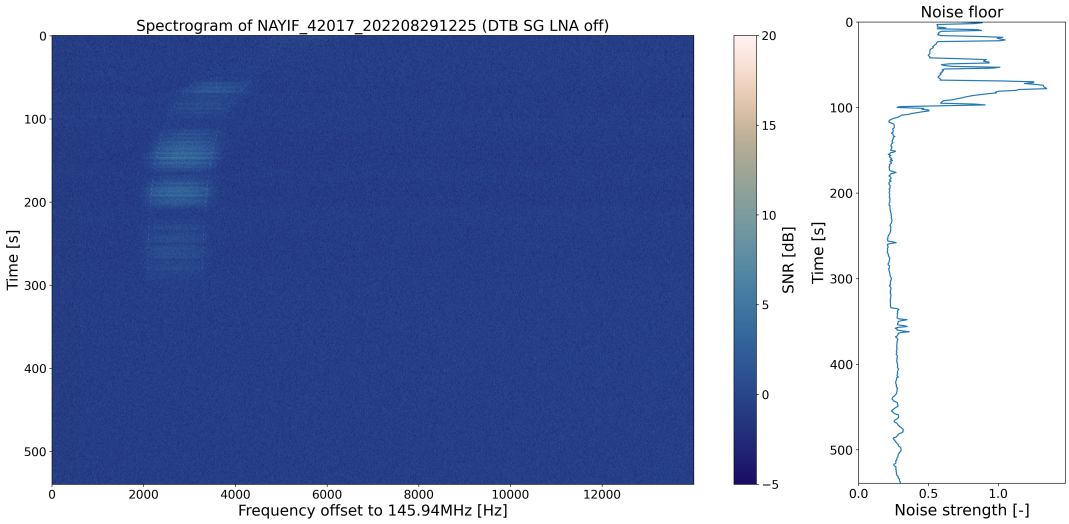


Figure A.13: Data set 8 (NAYIF_42017_202208291225) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.14: Spectrogram and noise floor for Data set 8 (NAYIF_42017_202208291225).

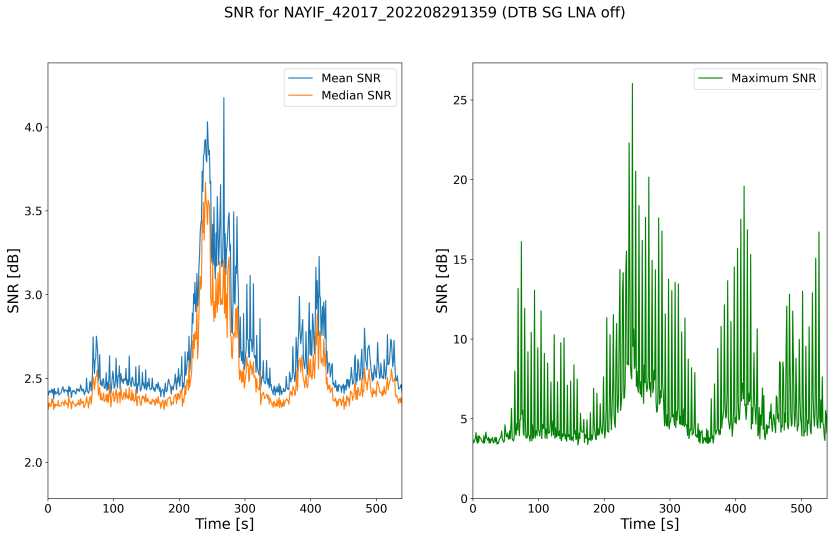
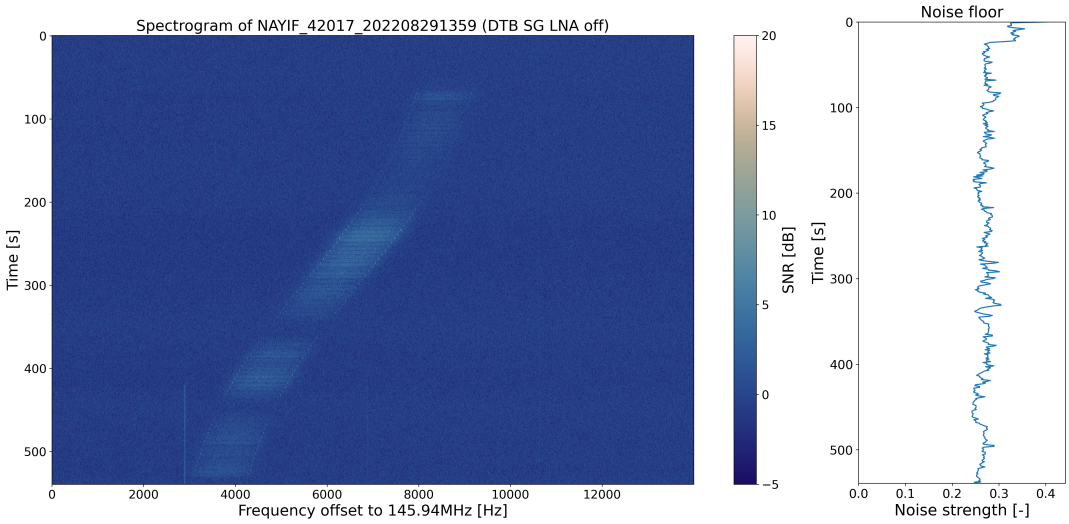


Figure A.15: Data set 9 (NAYIF_42017_202208291359) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.16: Spectrogram and noise floor for Data set 9 (NAYIF_42017_202208291359).

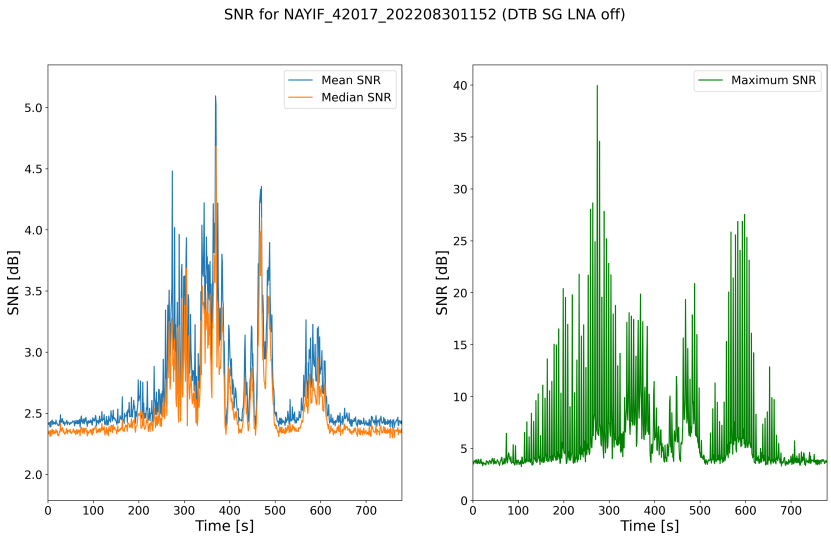
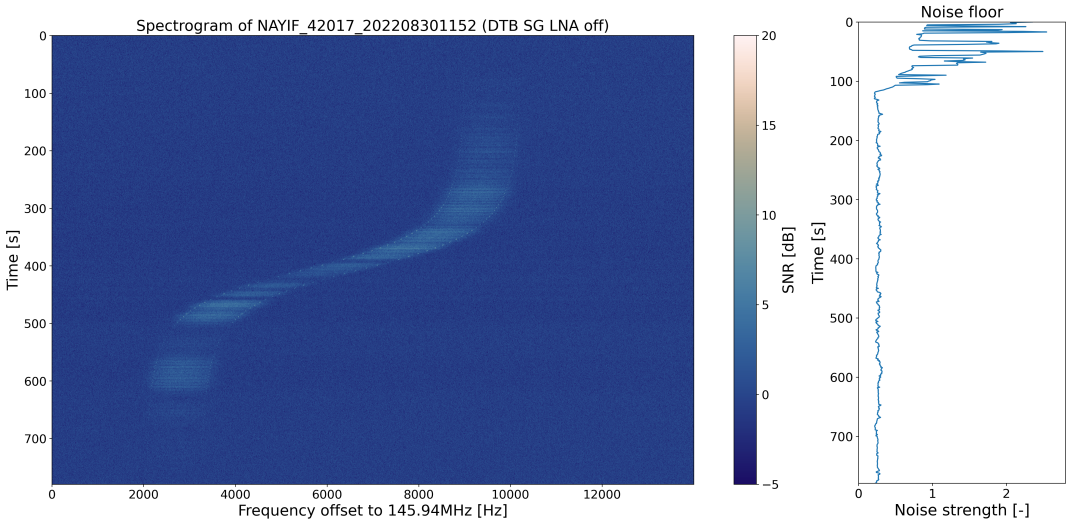


Figure A.17: Data set 10 (NAYIF_42017_202208301152) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.18: Spectrogram and noise floor for Data set 10 (NAYIF_42017_202208301152).

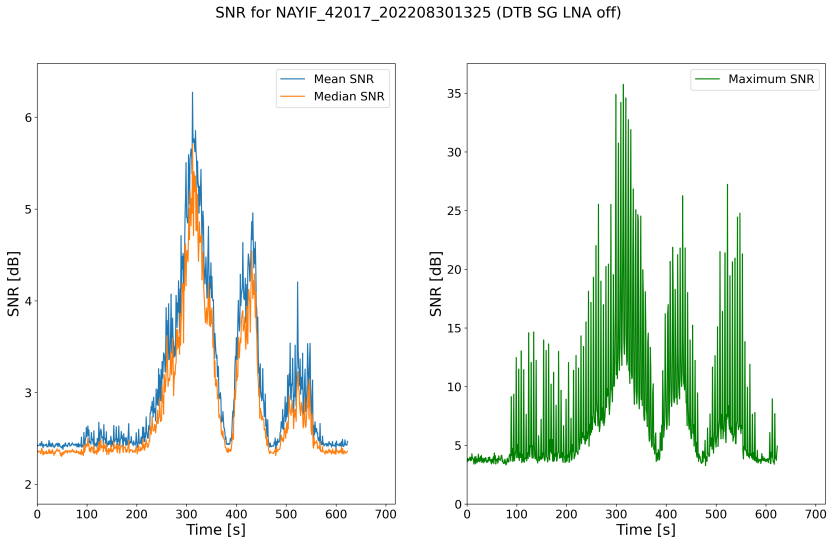
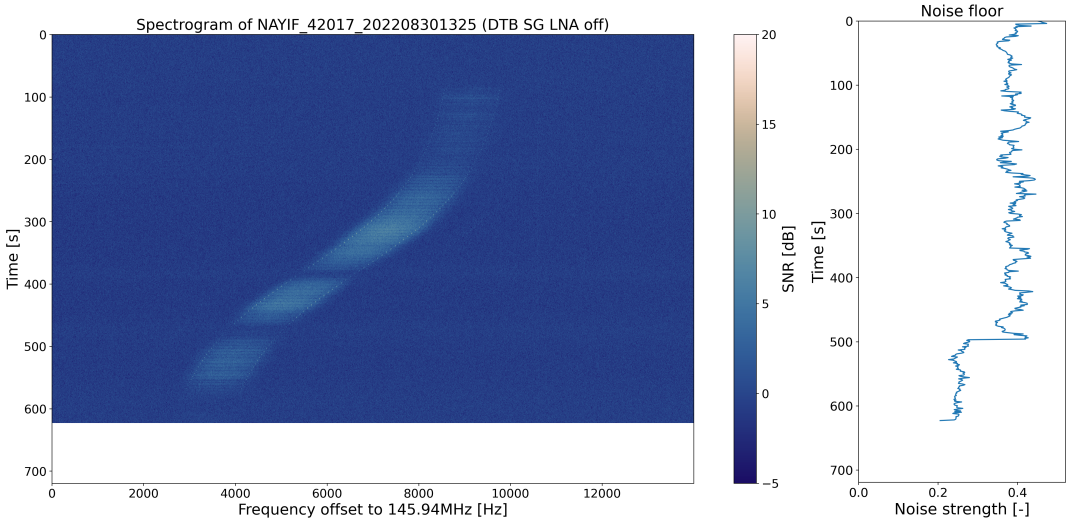


Figure A.19: Data set 11 (NAYIF_42017_202208301325) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.20: Spectrogram and noise floor for Data set 11 (NAYIF_42017_202208301325).

A.2. FREQUENCY PLOTS FOR EXPERIMENT 1.3

This section contains the additional frequency plots part of Experiment 1.3 from Section 6.2.3.

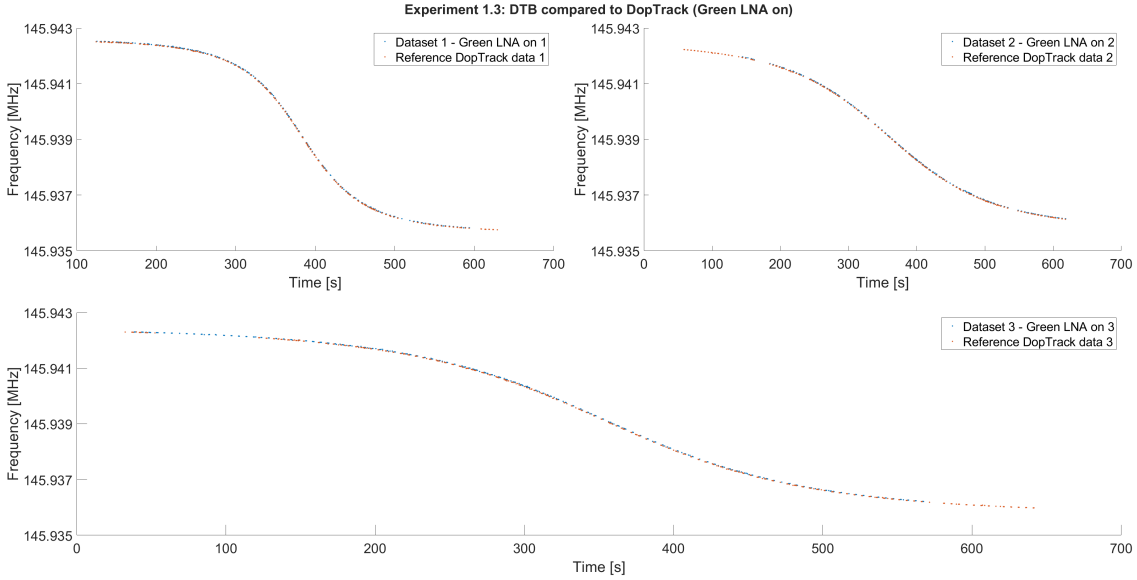


Figure A.21: Experiment 1.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for Green LNA on (Data sets 1, 2, and 3). Each plot represents its own data set with the corresponding DopTrack reference.

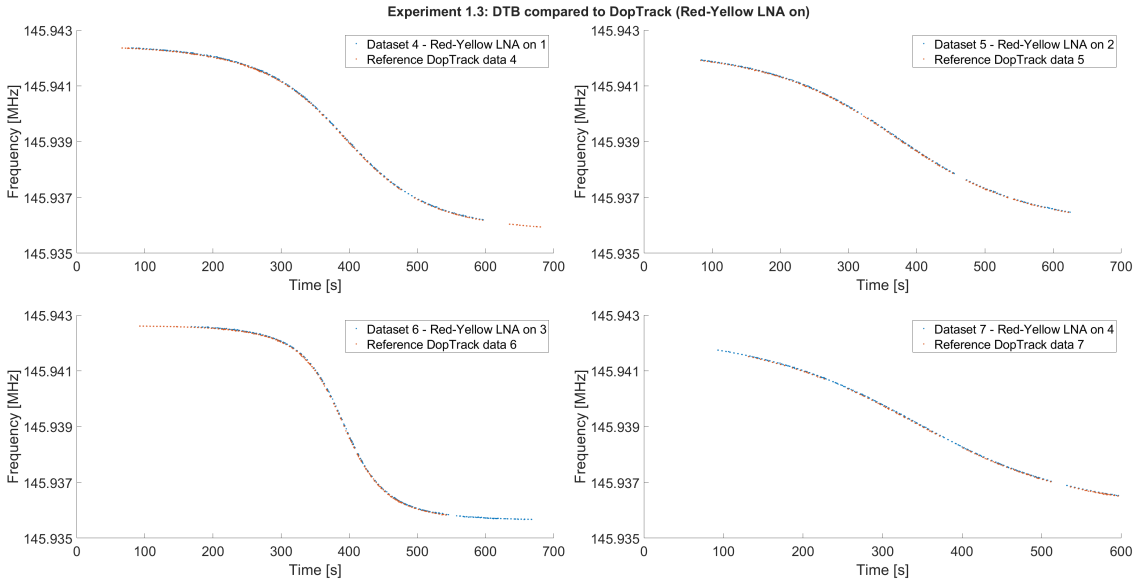


Figure A.22: Experiment 1.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for Red-Yellow LNA on (Data sets 4, 5, 6, and 7). Each plot represents its own data set with the corresponding DopTrack reference.

A.3. SPECTROGRAMS AND SNR PLOTS FOR EXPERIMENT 2

This section contains the spectrogram, noise floor, and SNR plots of the data used in Experiment 2 from Section 6.3 that are not present elsewhere in the report.

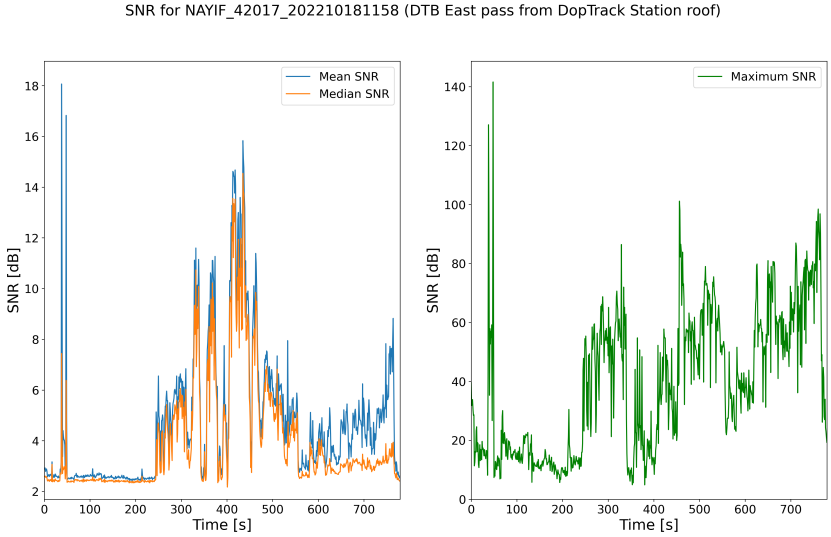
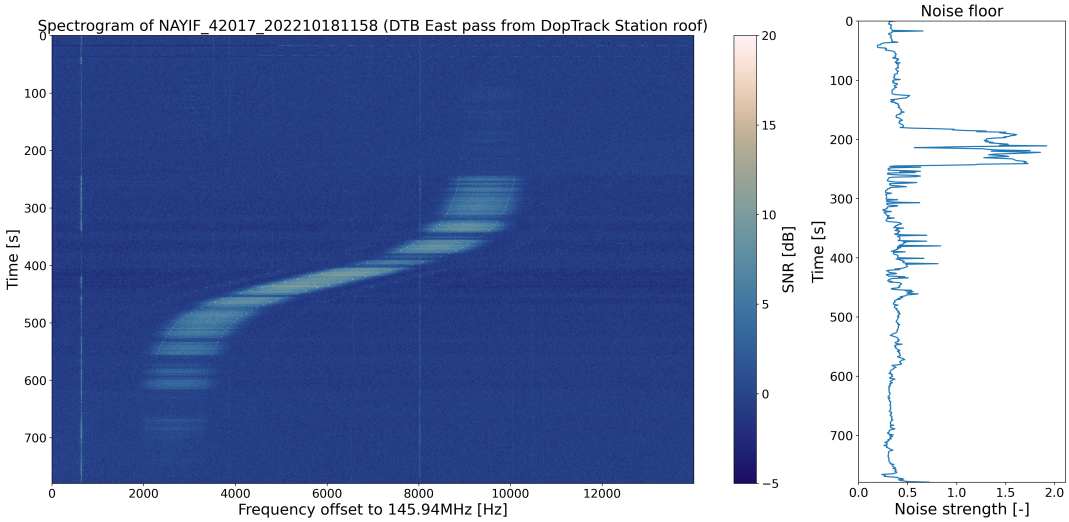


Figure A.23: Data set 12 (NAYIF_42017_202210181158) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.24: Spectrogram and noise floor for Data set 12 (NAYIF_42017_202210181158).

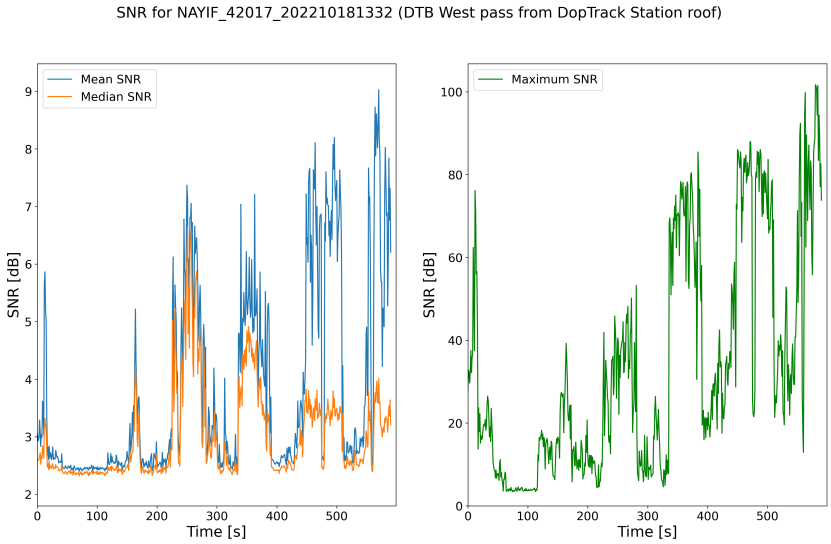
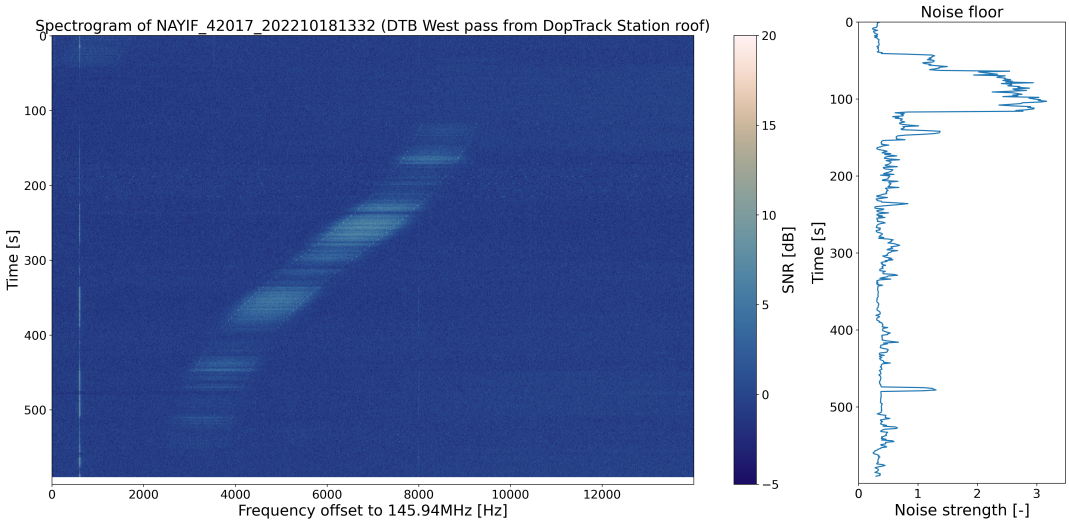


Figure A.25: Data set 13 (NAYIF_42017_202210181332) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.26: Spectrogram and noise floor for Data set 13 (NAYIF_42017_202210181332).

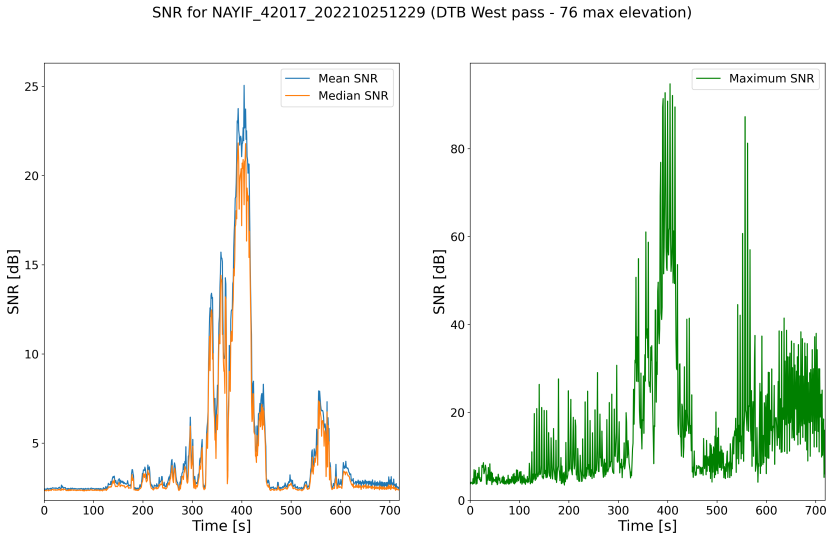
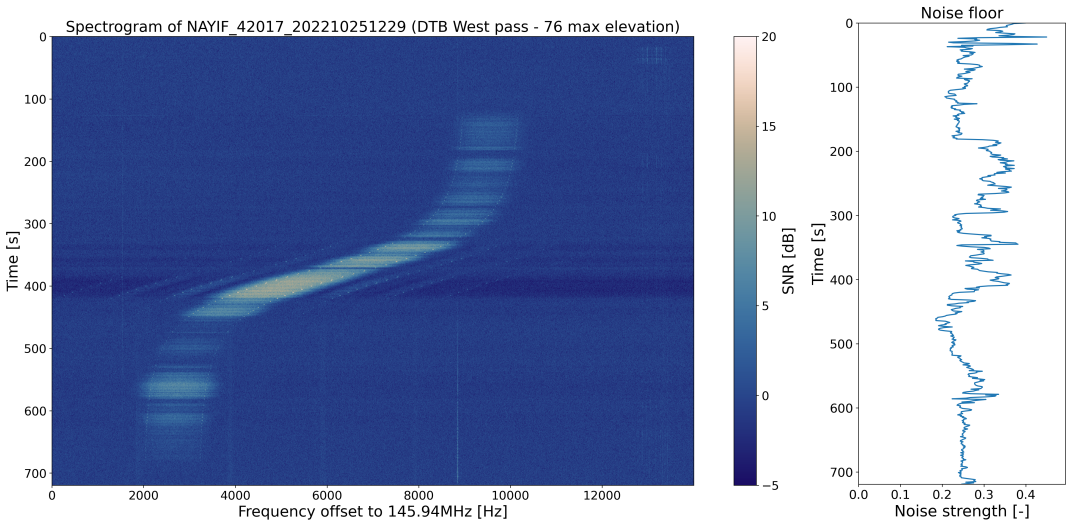


Figure A.27: Data set 14 (NAYIF_42017_202210251229) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.28: Spectrogram and noise floor for Data set 14 (NAYIF_42017_202210251229).

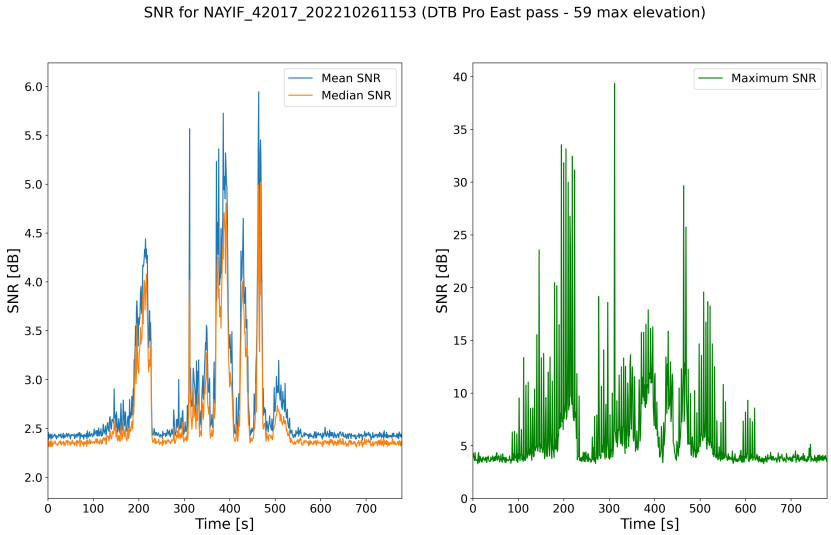
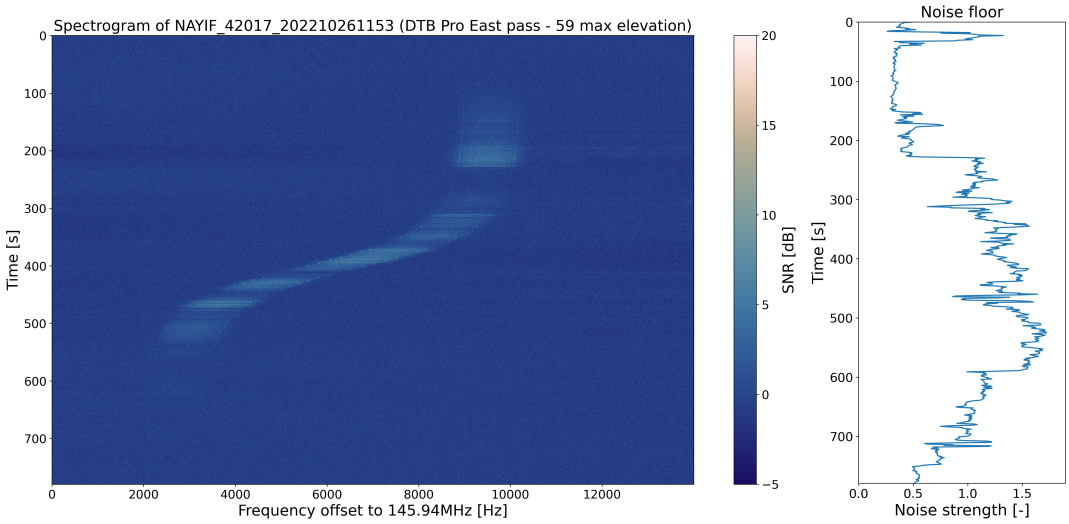


Figure A.29: Data set 15 (NAYIF_42017_202210261153) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.30: Spectrogram and noise floor for Data set 15 (NAYIF_42017_202210261153).

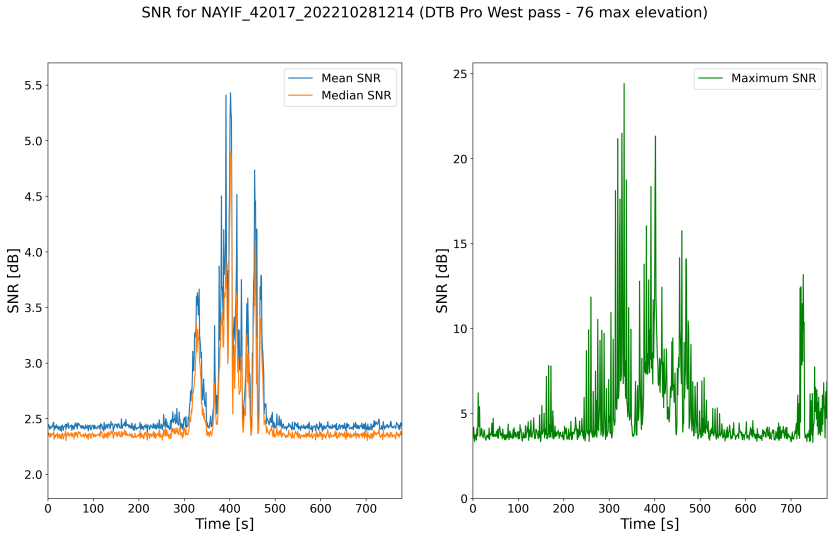
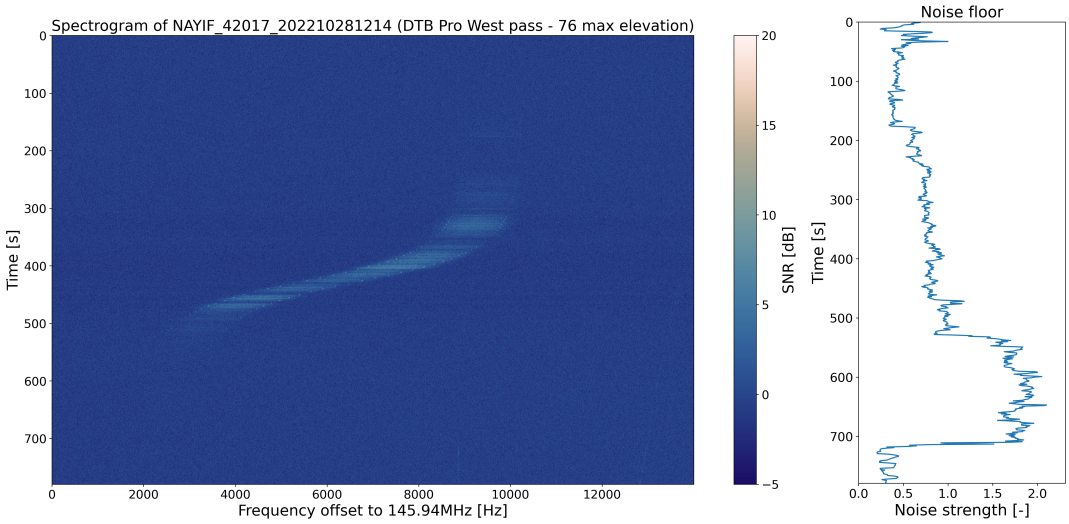


Figure A.31: Data set 16 (NAYIF_42017_202210281214) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.32: Spectrogram and noise floor for Data set 16 (NAYIF_42017_202210281214).

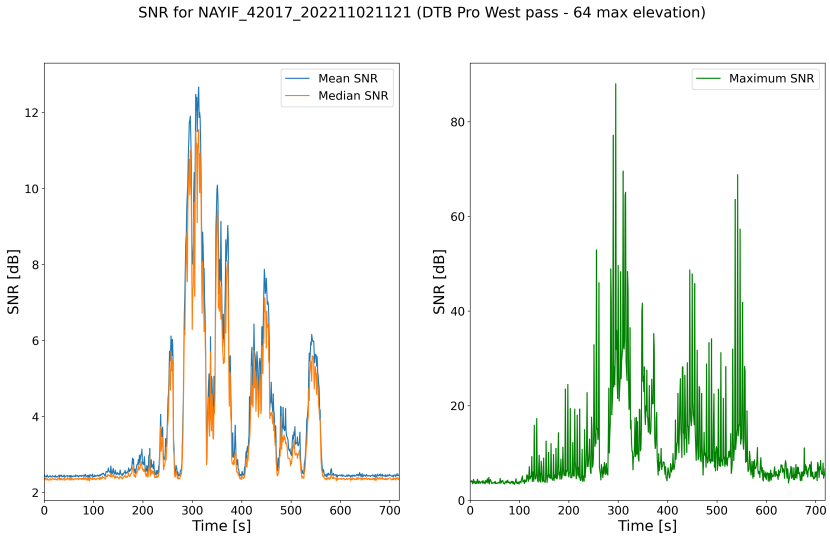
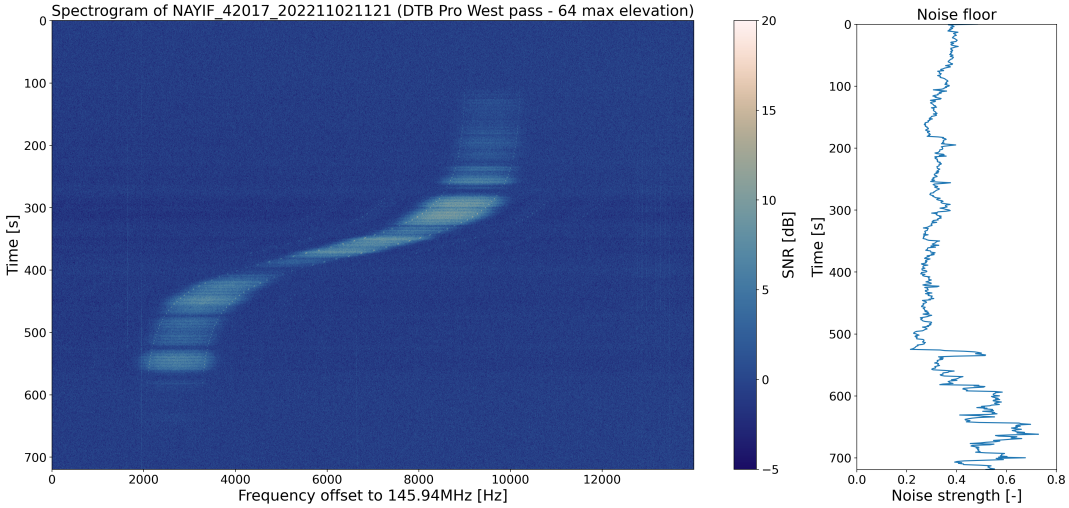


Figure A.33: Data set 17 (NAYIF_42017_202211021121) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.34: Spectrogram and noise floor for Data set 17 (NAYIF_42017_202211021121).

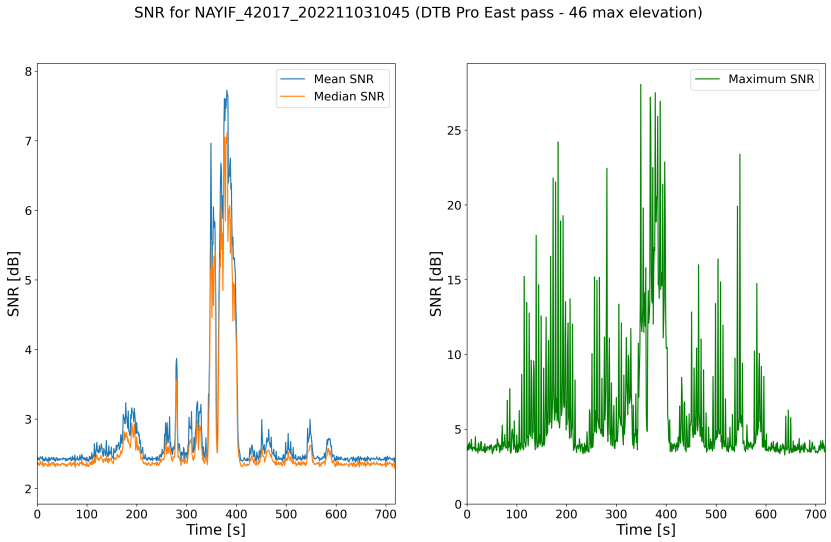
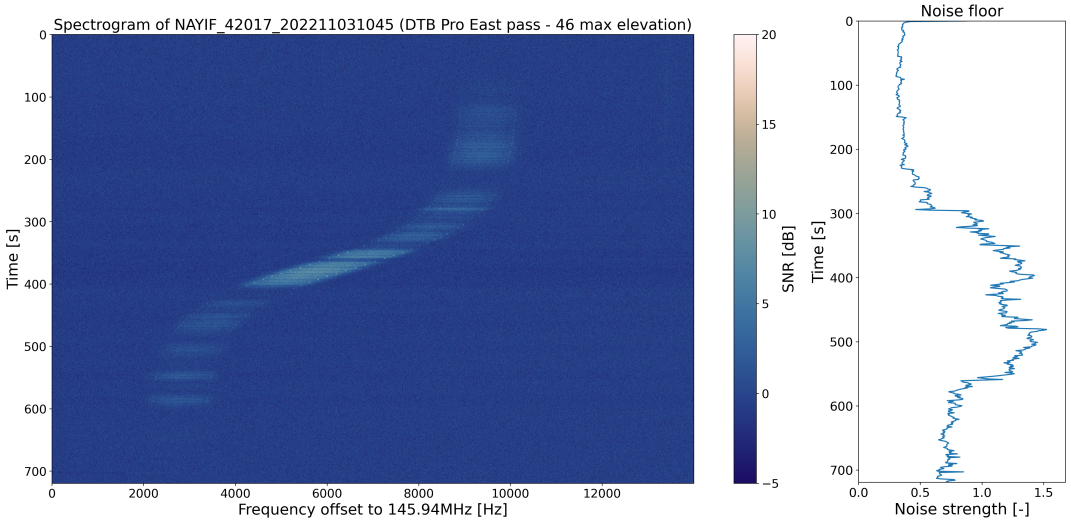


Figure A.35: Data set 18 (NAYIF_42017_202211031045) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.36: Spectrogram and noise floor for Data set 18 (NAYIF_42017_202211031045).

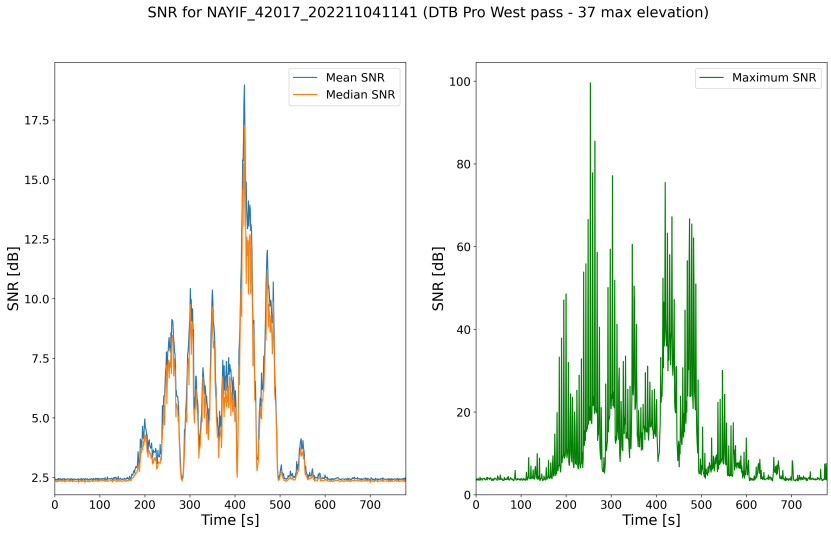
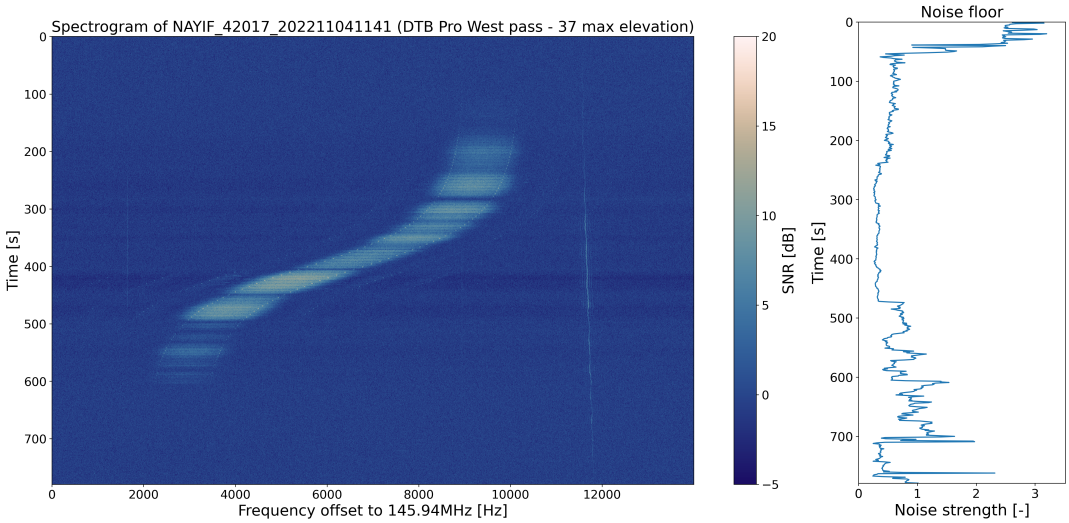


Figure A.37: Data set 19 (NAYIF_42017_202211041141) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.38: Spectrogram and noise floor for Data set 19 (NAYIF_42017_202211041141).

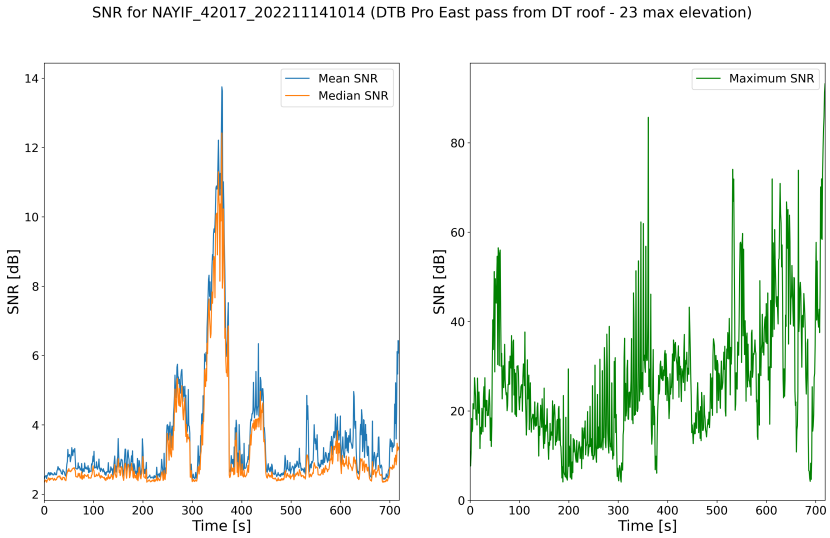
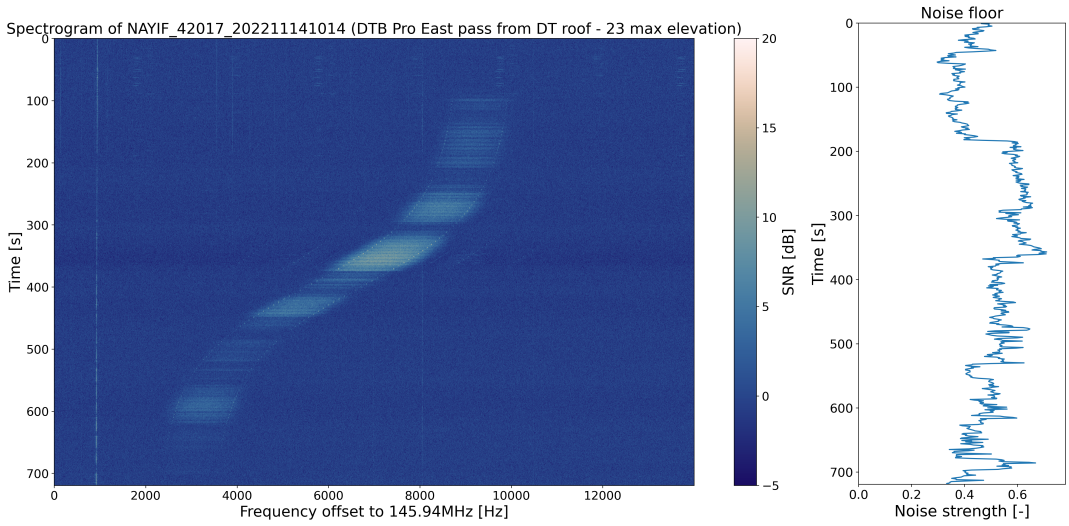


Figure A.39: Data set 20 (NAYIF_42017_202211141014) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.40: Spectrogram and noise floor for Data set 20 (NAYIF_42017_202211141014).

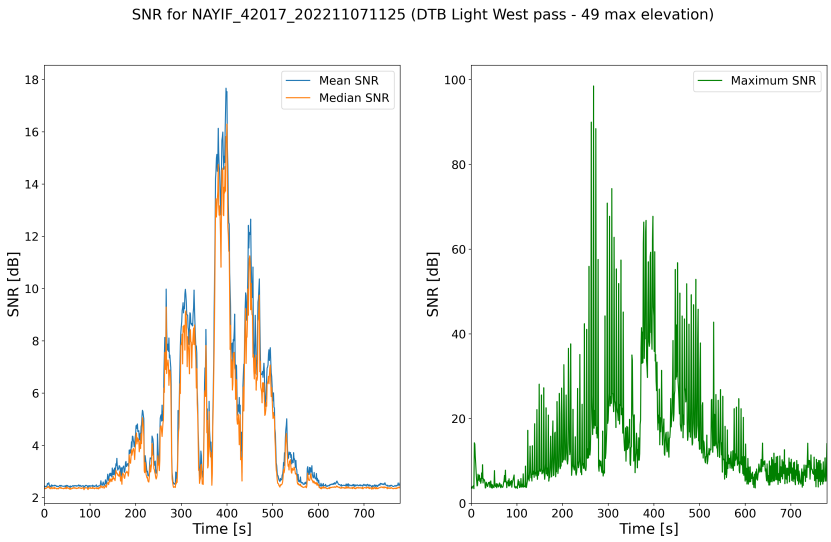
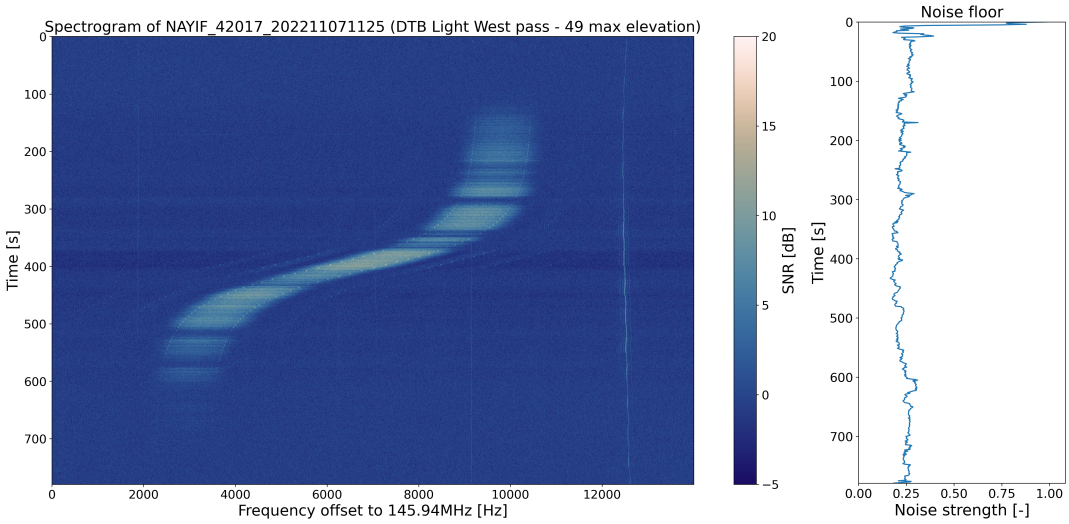


Figure A.41: Data set 21 (NAYIF_42017_202211071125) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.42: Spectrogram and noise floor for Data set 21 (NAYIF_42017_202211071125).

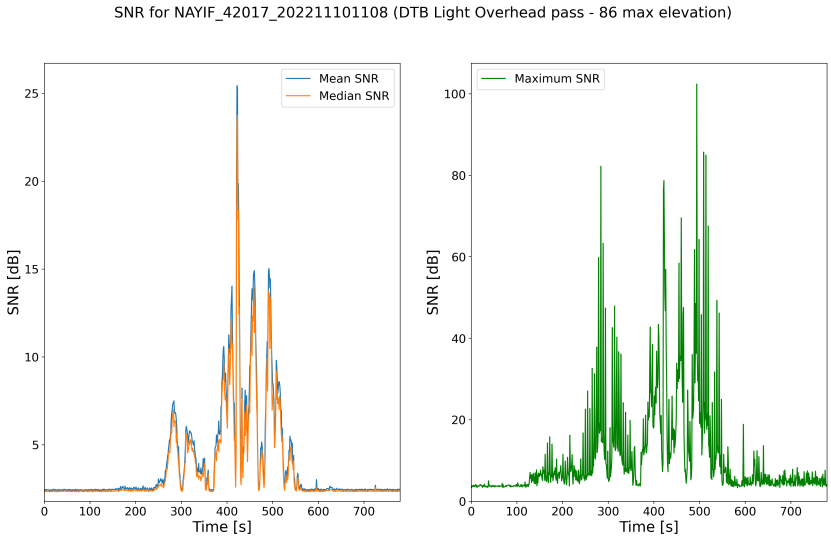
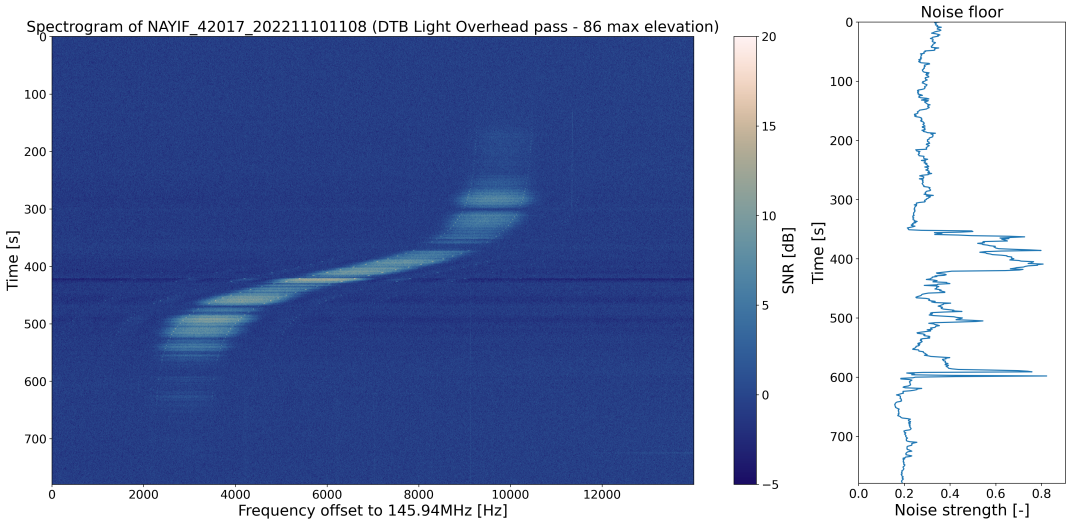


Figure A.43: Data set 22 (NAYIF_42017_202211101108) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.44: Spectrogram and noise floor for Data set 22 (NAYIF_42017_202211101108).

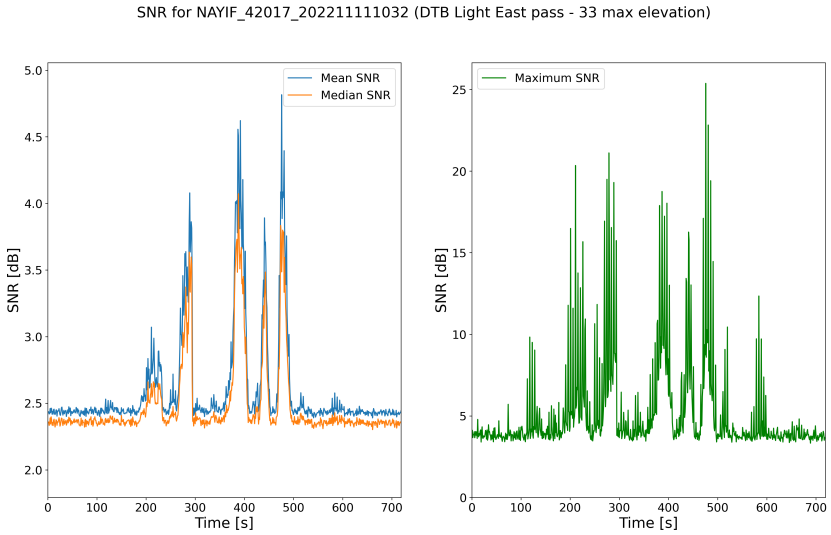
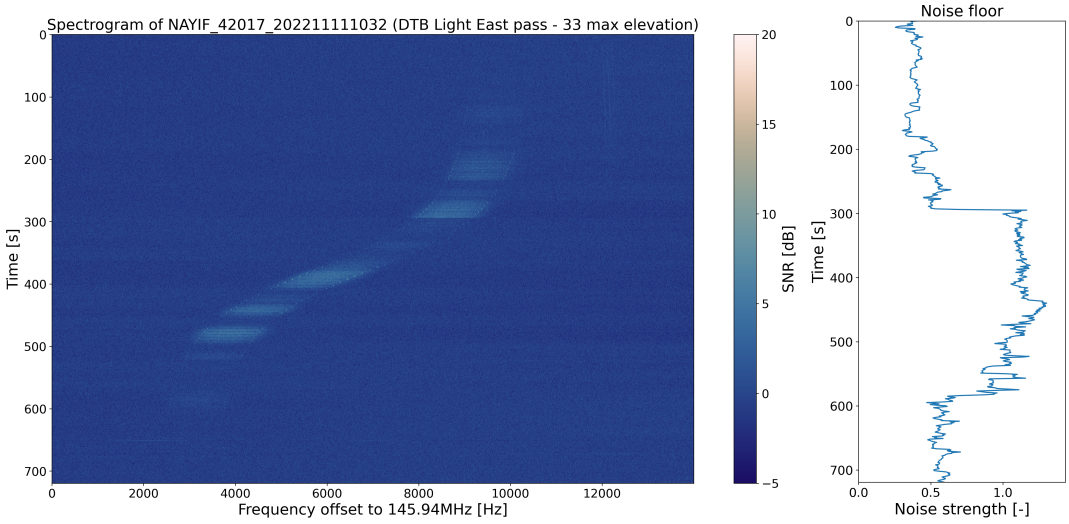


Figure A.45: Data set 23 (NAYIF_42017_202211111032) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.46: Spectrogram and noise floor for Data set 23 (NAYIF_42017_202211111032).

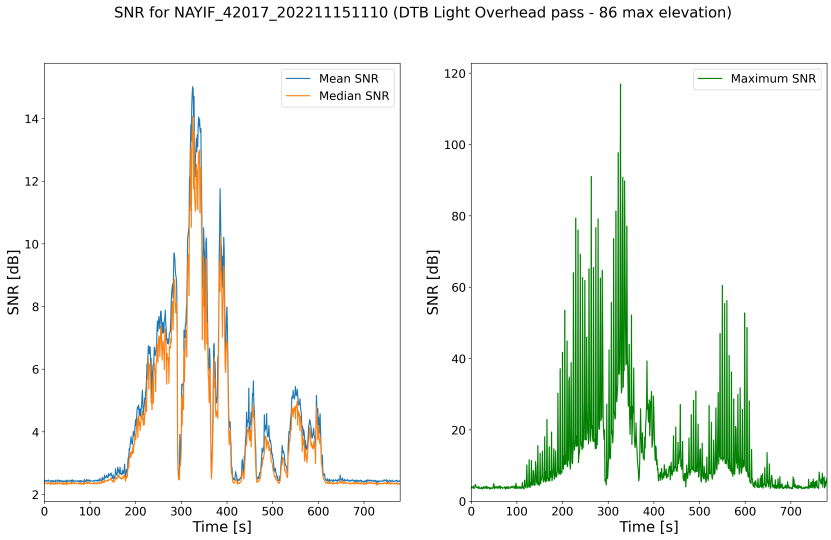
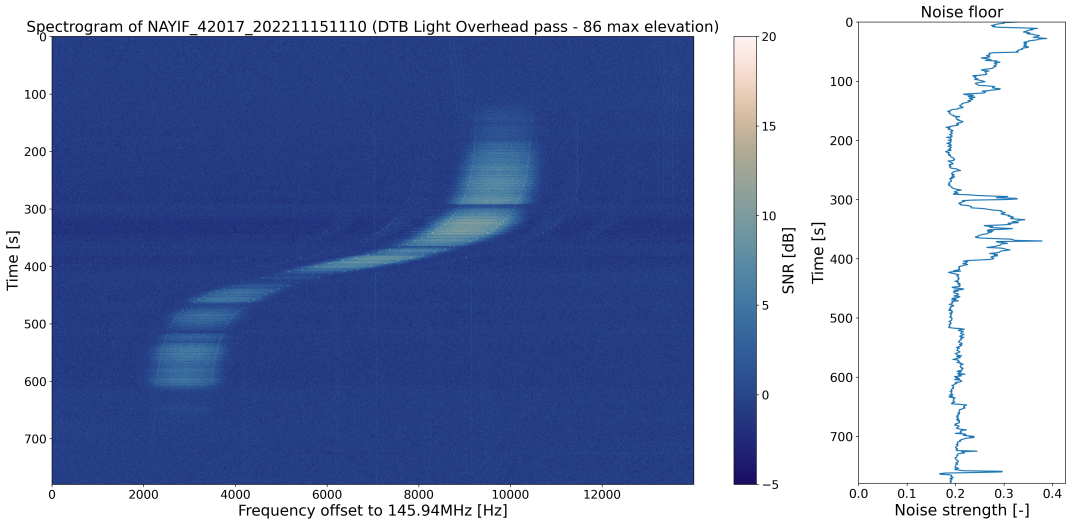


Figure A.47: Data set 24 (NAYIF_42017_202211151110) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.48: Spectrogram and noise floor for Data set 24 (NAYIF_42017_202211151110).

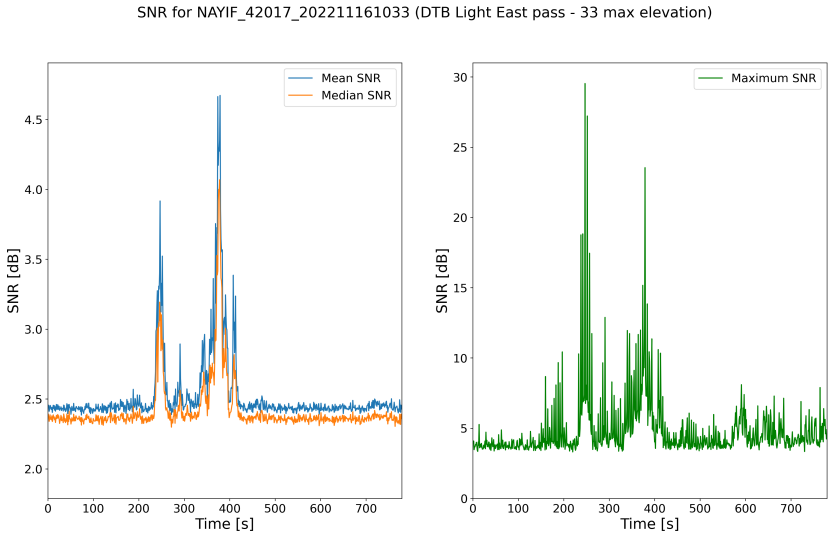
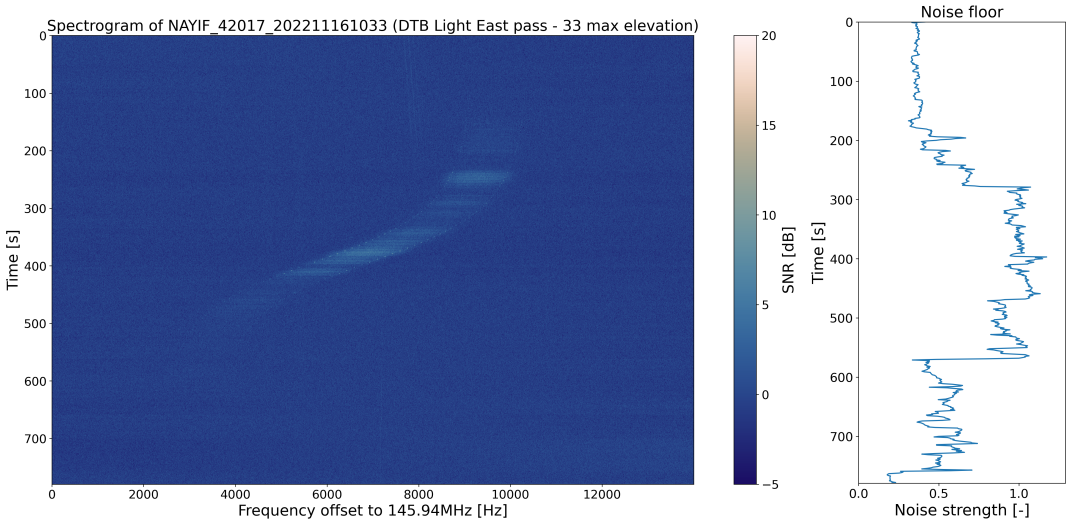


Figure A.49: Data set 25 (NAYIF_42017_202211161033) different SNR values.



(a) Spectrogram

(b) Noise floor

Figure A.50: Spectrogram and noise floor for Data set 25 (NAYIF_42017_202211161033).

A.4. FREQUENCY PLOTS FOR EXPERIMENT 2.3

This section contains the additional frequency plots part of Experiment 2.3 from Section 6.3.3.

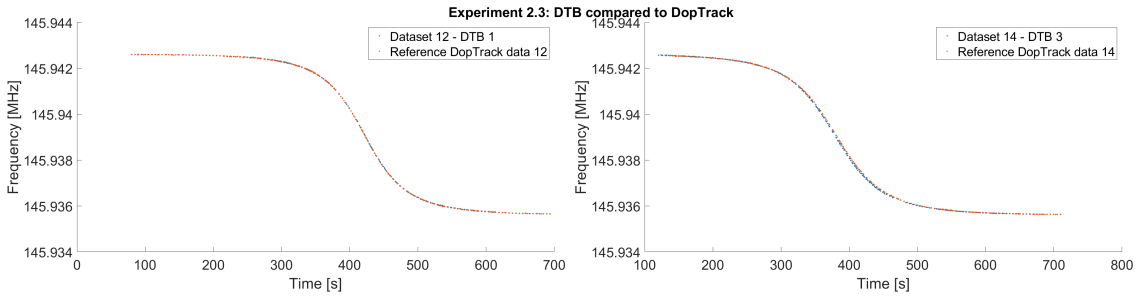


Figure A.51: Experiment 2.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB configuration (Data sets 12 and 14). Each plot represents its own data set with the corresponding DopTrack reference.

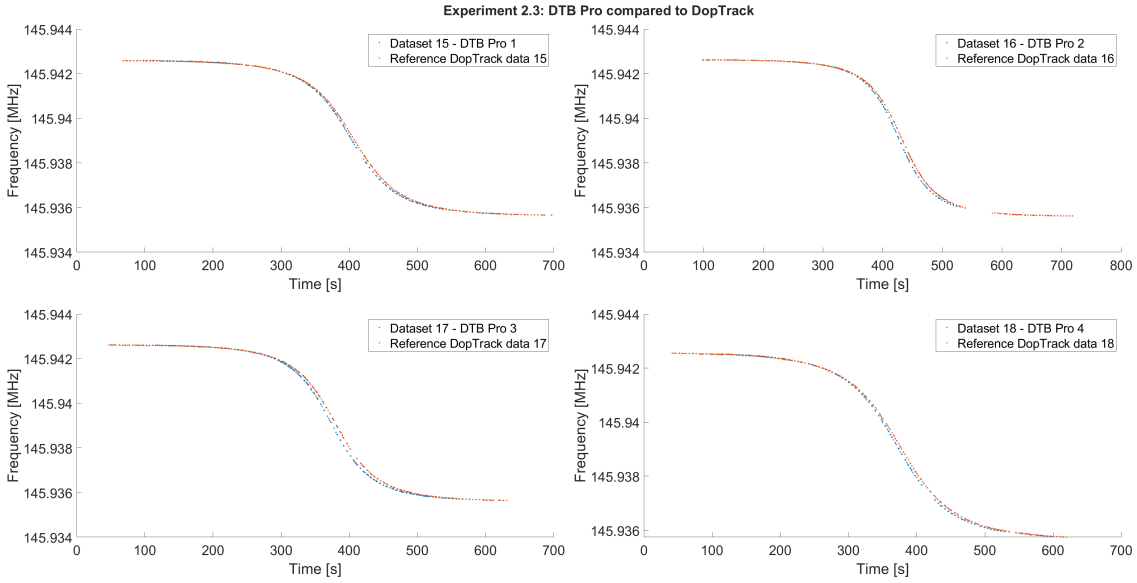


Figure A.52: Experiment 2.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Pro configuration (Data sets 15, 16, 17, and 18). Each plot represents its own data set with the corresponding DopTrack reference.

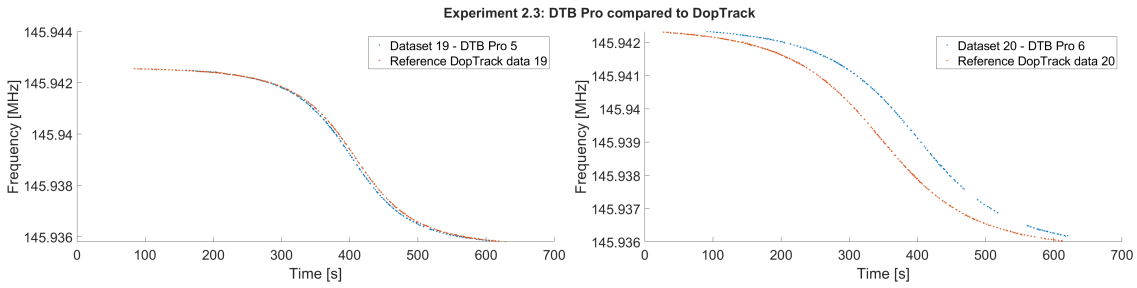


Figure A.53: Experiment 2.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Pro configuration (Data sets 19 and 20). Each plot represents its own data set with the corresponding DopTrack reference.

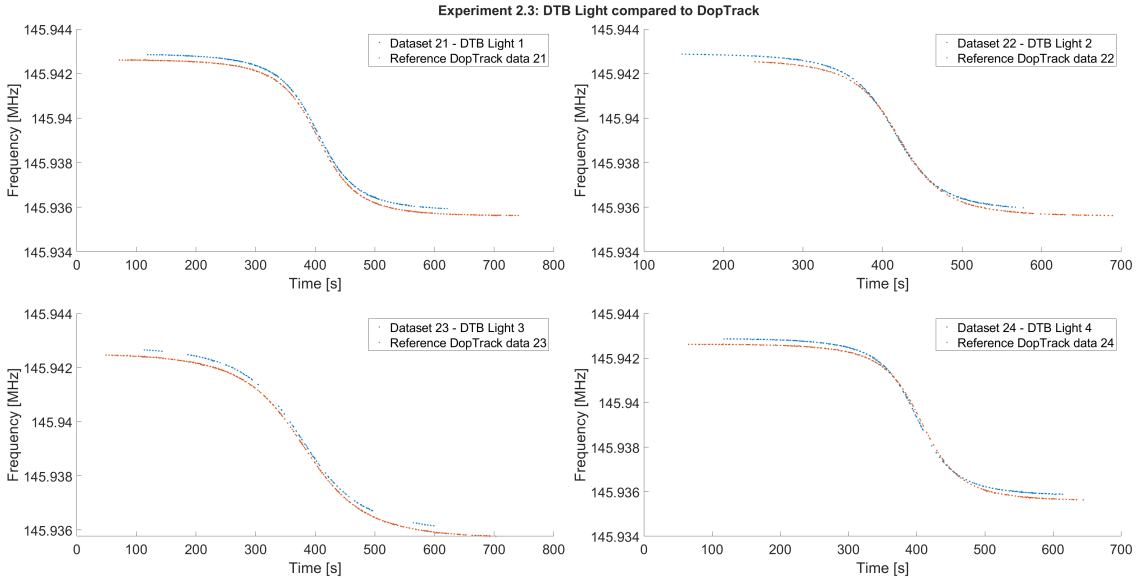


Figure A.54: Experiment 2.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Light configuration (Data sets 21, 22, 23, and 24). Each plot represents its own data set with the corresponding DopTrack reference.

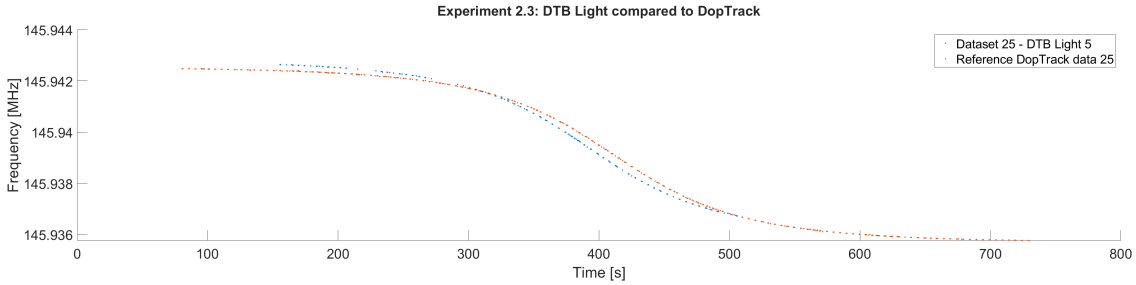


Figure A.55: Experiment 2.3 Frequency change over time comparison for DopTrackBox (blue) and DopTrack (orange) for the DTB Light configuration (Data set 25). Each plot represents its own data set with the corresponding DopTrack reference.

B | Program Code

This appendix contains the major DopTrackBox program code that was altered from the original DopTrack software or added specifically for DopTrackBox. It is split up in three sections. Section B.1 contains the scheduler software, Section B.2 contains the recorder software, and Section B.3 contains the processing software. All program code is also backed-up on the DopTrack Github.

B.1. Scheduler Software

Software part of the scheduler package discussed in Section 2.6.1 can be found here.

B.1.1. schedule.py

```
#!/usr/bin/python
#
# This python program reads the rec.list and set ups the meta-files and arms them for
    recording
#
# Development log:
# - 14-12-2015, Bart Root: initial development
# - 10-05-2022, Amber Sprenkels: updated for DopTrackBox and python3.x
#
#-------------------------------------------------
import yaml
import Record_DTB
import predict_v2
import os
import subprocess
import datetime
import createYAMLfile

# set global
HOME = '/home/amber/DTB/'
LOC_PEN = '/home/amber/DTB/Recordings/REC_PENDING/'
LOC_ARM = '/home/amber/DTB/Recordings/REC_ARMED/'
LOC_REC = '/home/amber/DTB/Recordings/data'
LOC_RUN = '/home/amber/DTB/DopTrack/GroundControl/Automation-DopTrackBox/'

priority = 0
os.chdir(LOC_RUN)

# make mother metafile
createYAMLfile.make()

# read rec list and loop over all satellite entries
rec_list = HOME + 'rec.list'
with open(rec_list) as f:
  for line in f:
  # priority is in sequency of read first
    priority = priority + 1
    line = line.strip()
    columns = line.split()

    # set satellite values for later
    NORADID = columns[1]
    name = columns[0]
    freq = int(columns[2])
    samp_rate = int(columns[3])

    # create mother meta file for this particular satellite
    with open('empty.yml', 'r') as metaf:
      metam = yaml.load(metaf)
    metaf.close()

```

```
51      # fill in mothe meta file
52      metam['Sat']['State']['Name'] = columns[0]
53      metam['Sat']['State']['NORADID'] = columns[1]
54      metam['Sat']['State']['Tuning Frequency'] = int(columns[2])
55      metam['Sat']['Record']['sample_rate'] = int(columns[3])
56      metam['Sat']['State']['Priority'] = priority
57
58      # Determine Antenna
59      if 30000000 < freq <= 300000000:
60        # VHF antenna range
61        metam['Sat']['State']['Antenna'] = 1
62      elif 300000000 < freq < 1000000000:
63        #UHF antenna range
64        metam['Sat']['State']['Antenna'] = 3
65      else:
66        # default: VHF antenna range
67        metam['Sat']['State']['Antenna'] = 1
68
69      #read TLE
70      if NORADID[0:1] == 'PL':
71        # Don't get TLE, because using a pre-specified pre-launch TLE
72        print ("Pre-Launch TLE")
73      else:
74        # get TLE from the space-track website
75        subprocess.call(['./getTLE',str(NORADID)])
76
77      # make prediction
78      metam = predict_v2.predict(metam)
79
80      # make the metafiles
81      tnow = datetime.datetime.now()
82      year = tnow.year
83
84      # input file
85      fname = 'prediction_' + str(NORADID) + '.txt'
86      fin = open(fname,'r')
87
88      # Start reading the prediction file and construct the metafile
89      for fline in fin.readlines():
90        elevation = int(fline[25:27])
91
92        # check if elevation is above a certain treshold
93        if elevation > 30 : #5 :
94          # get the acquired variables from the line
95          day = int(fline[3:5])
96          month = int(fline[6:8])
97          bhour = int(fline[9:11])
98          bminute = int(fline[12:14])
99
100         ehour = int(fline[30:32])
101         eminute = int(fline[33:35])
102         SAzimuth = int(fline[15:18])
103         EAzimuth = int(fline[36:39])
104         # do the calculations
105         if bminute == 0 :
106           bminute = 59
107           if bhour == 0 :
108             bhour = 23
109           else :
110             bhour = bhour - 1
111         else :
112           bminute = bminute - 1
113         if eminute == 59 :
114           eminute = 0
115           if ehour == 23 :
116             ehour = 0
117           else :
118             ehour = ehour + 1
119         else :
120           eminute = eminute + 1
121
122         if ehour == 24 :
```

```
123           ehour = 0
124         if bhour == 24 :
125           bhour = 0
126
127         # determine the length of recording
128         tb = datetime.datetime(year,month,day,bhour,bminute)
129         te = datetime.datetime(year,month,day,ehour,eminute)
130         lofp = te - tb
131         lofp = lofp.seconds
132
133         # determine number of samples
134         num_samp = int(lofp)*int(samp_rate)
135
136         # start of recording
137         start_rec = int(str(year) + str(month).zfill(2) + str(day).zfill(2) + str(bhour)
    .zfill(2) + str(bminute).zfill(2))
138
139         #  make daughter metafile
140         meta = metam
141         # fill in the meta fill
142         meta['Sat']['Predict']['EAzimuth'] = EAzimuth
143         meta['Sat']['Predict']['SAzimuth'] = SAzimuth
144         meta['Sat']['Predict']['Length of pass'] = int(lofp)
145         meta['Sat']['Predict']['Elevation'] = elevation
146
147         meta['Sat']['Record']['Start of recording'] = start_rec
148         meta['Sat']['Record']['num_sample'] = num_samp
149
150         # Stored the metafile in the pending direcory
151         metaname = LOC_PEN + str(name) + '_' + str(NORADID) + '_' + str(start_rec) + '.
    yml'
152         with open(metaname, 'w+') as outfile:
153           outfile.write( yaml.dump(meta, default_flow_style=False) )
154         outfile.close()
155       else:
156         # do nothing
157         elevation = elevation
158
159
160 print ("End of for-loop!")
161 fin.close()
162 # close rec list
163 f.close()
164
165 # check the pending recordings for arm
166 subprocess.call(['./make_atq.sh'])
167
168 # Log list armed recordings
169
170 # end of program
```

B.1.2. PREDICT_V2.PY

```
1  #!/usr/bin/python
2  #
3  # this program will read a TLE file and produce a prediction of the satellite
4  #
5  # Written by Bart Root, TUDelft, 24-Aug-2015
6  #
7  # Dependent functions:
8  #
9  # - earth_gravity.py
10 # - io.py
11 #
12 # Change log:
13 #
14 # Initial developement: Bart Root - 24-Aug-2015
15 # Update for DopTrackBox and python3.x: Amber Sprenkels - 10-05-2022
16 #
17 #---------------------- start of routine ---------------------------------------
18
19 # import libaries
```

```
20 import numpy as np
21 import os.path
22 import sys
23 import datetime
24 import math
25 from sgp4.coordconv3d import *
26 from sgp4.geo import WGS84
27 from sgp4.sidereal import *
28 from sgp4.earth_gravity import wgs84
29 from sgp4.io import twoline2rv
30 import time as gmttime
31 import yaml
32
33 def predict(meta):
34
35     # load meta file variable
36     name = meta['Sat']['State']['Name']
37     NORADID = meta['Sat']['State']['NORADID']
38
39     # Set global variables
40     pi = 3.1415926535897
41     Re = 6378136.00 # radius of Earth in meters
42     min_part = 15   # interval between SGP4 propagation
43     fullPeriod = 3*24*60*60//min_part # 5 day period of prediction
44
45     # Get starttime and start date
46     TT = datetime.datetime.utcnow()
47     gmtoff = int(-gmttime.timezone)/3600 + int(gmttime.localtime().tm_isdst)
48
49     # get time format correct
50     TT_stamp = TT.strftime("%Y%m%d%H%M%S")
51     meta['Sat']['Predict']['time used UTC'] = int(TT_stamp)
52     meta['Sat']['Predict']['timezone used'] = gmtoff
53
54     # station coordinates [station: DopTrack]
55     station_lat = 51+59/60+56.387/60/60
56     station_lon = 4+22/60+24.558/60/60
57     station_h = 0#130.85
58     # station coordinates [station: DopTrackBox]
59     station_lat = [REDACTED]
60     station_lon = [REDACTED]
61     station_h = [REDACTED]
62
63
64     # store station coordinates
65     #meta['Sat']['Station']['Name'] = 'DopTrack'
66     meta['Sat']['Station']['Name'] = 'DopTrackBox'
67     meta['Sat']['Station']['Lat'] = float(station_lat)
68     meta['Sat']['Station']['Lon'] = float(station_lon)
69     meta['Sat']['Station']['Height'] = float(station_h)
70
71     # Initializing
72     print ("Start of reading TLE file...")
73
74     # --------------------input TLE-------------------------------------
75     fname = 'TLE_' + NORADID + '.txt'
76     # check if TLE file is present
77     if os.path.isfile(fname):
78         f = open(fname,'r')
79     else:
80         sys.exit('Error: TLE.txt file is not present!')
81
82     # continue program if TLE file is present
83     line1 = f.readline()
84     line2 = f.readline()
85     f.close()
86
87     # store in the meta file
88     meta['Sat']['Predict']['used TLE line1'] = str(line1[:69]).rstrip()
89     meta['Sat']['Predict']['used TLE line2'] = str(line2[:69]).rstrip()
90     # --------------------input TLE-------------------------------------
91
```

```
92      # Construct satellite variable
93      satellite = twoline2rv(line1, line2, wgs84)
94      print ("Start 5-day prediction at: %s" % (TT))
95
96      # scenario variables
97      Pass = 0
98      pass_hor_minus1 = 0
99      inview = 0
100     scenario = 0
101     k = 1
102     start = 0
103     predictfile = 'prediction_' + str(NORADID) + '.txt'
104     f3 = open(predictfile,'w')
105     lst = []
106
107     # start of the prediction loop
108     for it in range(1,fullPeriod):
109         # add time
110         time = TT + datetime.timedelta(0,15*(it-1))
111
112         year = time.year
113         month = time.month
114         day = time.day
115         hour = time.hour
116         minute = time.minute
117         sec = time.second
118
119         pos,vel = satellite.propagate(year, month, day, hour, minute, sec)
120
121         # Get Julian date
122         jdTT = JulianDate.fromDatetime(time)
123         jdut1 = JulianDate.__float__(jdTT)
124
125         # Get Greenwich Apparent Siderial Time
126         tut1= ( jdut1 - 2451545.0 ) / 36525.0
127
128         temp = -6.2e-6 * tut1 * tut1 * tut1 + 0.093104 * tut1 * tut1 + (876600.0 *
    3600.0 + 8640184.812866) * tut1 + 67310.54841
129
130         # 360/86400 = 1/240, to deg, to rad
131
132         dumy2 = divmod( temp*pi/180.0/240.0,2*pi )
133         temp = dumy2[1]
134
135         # ------------------------ check quadrants --------------------
136         if ( temp < 0.0 ):
137             temp = temp + 2*pi
138
139         gst = temp
140         cgast = math.cos(gst)
141         sgast = math.sin(gst)
142
143         # Transformation of coordinates of satellite
144         x =  pos[0] * cgast * 1000 + pos[1] * sgast * 1000
145         y = -pos[0] * sgast * 1000 + pos[1] * cgast * 1000
146         z =  pos[2] * 1000
147
148         # Transformation to Latitude, Longitude, and altitude
149         #Calculate lon
150         lon = math.atan2(y, x)
151         #Initialize the variables to calculate lat and alt
152         alt = 0
153         N = WGS84.a
154         p = sqrt(x**2 + y**2)
155         lat = 0
156         previousLat = 90
157         #Iterate until tolerance is reached
158         while abs(lat - previousLat) >= 1e-9:
159             previousLat = lat
160             sinLat = z / (N * (1 - WGS84.e**2) + alt)
161             lat = math.atan((z + WGS84.e**2 * N * sinLat) / p)
162             N = WGS84.a / sqrt(1 - (WGS84.e * sinLat)**2)
```

```
163             alt = p / math.cos(lat) - N
164 
165         lon = lon/pi*180
166         lat = lat/pi*180
167 
168     # get station coordinates into ecef
169         vlat = station_lat/180*pi
170         vlon = station_lon/180*pi
171         valt = station_h
172         #Calculate length of the normal to the ellipsoid
173         N = WGS84.a / sqrt(1 - (WGS84.e * math.sin(vlat))**2)
174         #Calculate ecef coordinates
175         vx = (N + valt) * math.cos(vlat) * math.cos(vlon)
176         vy = (N + valt) * math.cos(vlat) * math.sin(vlon)
177         vz = (N * (1 - WGS84.e**2) + valt) * math.sin(vlat)
178 
179     # check if satellite is in view of the station
180         cosgamma = Re/ (Re + alt)
181 
182     # Calculate the satellite gamma angle
183         satgamma = np.inner([vx,vy,vz],[x,y,z]) / math.sqrt(np.inner([x,y,z],[x,y,z])) /
       math.sqrt(np.inner([vx,vy,vz],[vx,vy,vz]))
184 
185     #check if satellite is inside view of horizon
186         if satgamma>cosgamma:
187             pass_hor = 1
188         else:
189             pass_hor = 0
190 
191     # get elevation and azimuth values
192         aer = geodetic2aer(lat, lon, alt, vlat/pi*180, vlon/pi*180, valt, ell=
      EarthEllipsoid(),deg=True)
193 
194     # Quantify the passes
195         hourp = int(hour+gmtoff)
196         if hourp == 24:
197             hourp = 00
198         if hourp == 25:
199             hourp = 1
200 
201     # check scenario for the first epoch
202         if TT == time:
203             if pass_hor == 1:
204                 Pass = 1
205                 inview = 1
206                 start = 1
207 
208               # initialize dynamic array
209                 lst.append(aer[1])
210 
211               # satellite is in view
212                 scenario = 1-pass_hor
213                 # update logfile
214                 #f3.write("%02i %02i-%02i %02i:%02i %3i | " % (int(Pass),int(day),int(
      month),int(hour+gmtoff),int(minute),int(aer[0])))
215                 k = k+1
216             else:
217                 scenario = 0-pass_hor
218         else:
219             scenario = pass_hor_minus1 - pass_hor
220 
221         # Scenario: out of view = 1, in view = -1, no change = 0
222         if scenario == 1:
223             # out of view: pass ends
224             inview = 0
225 
226             # Find elevation of TCA
227             if start == 1:
228                 start = 0;
229             else:
230                 # search for maximum Elevation in pass
231                 max_EL = max(lst)
```

```
232                 # update logfile
233                 f3.write("%03i %02i | " % (int(aer[0]),int(max_EL)))
234                 k = k+1;
235 
236                 lst = []
237 
238                 # update logfile
239                 f3.write("%02i:%02i %03i |\n" % (int(hourp),int(minute),int(AZ_pre)))
240                 k=k+1
241 
242         elif scenario == -1:
243             # inview: pass begins
244             Pass = Pass + 1
245             inview = 1
246             lst = []
247             lst.append(aer[1])
248 
249             # update logfile
250             f3.write("%02i %02i-%02i %02i:%02i %3i | " % (int(Pass),int(day),int(month),
    int(hourp),int(minute),int(aer[0])))
251             k = k+1
252         elif scenario == 0:
253             #no change
254             lst.append(aer[1])
255         else:
256             sys.exit('Error: Undefined scenario at pass check!')
257 
258         # Needed for the following scenario: out of view
259         AZ_pre = aer[0]
260         EL_pre = aer[1]
261         jdut1_pre = jdut1
262         pass_hor_minus1 = pass_hor
263 
264     f3.close()
265 
266     if inview == 1:
267         readFile = open(predictfile)
268         lines = readFile.readlines()
269         readFile.close()
270         w = open(predictfile,'w')
271         w.writelines([item for item in lines[:-1]])
272         w.close()
273     # return the updated meta file
274     print ("End of Program!")
275     return meta;
276 
277 if __name__ == "__main__":
278    main(meta)
```

B.1.3. MAKE_ATQ.SH

```
1  #!/bin/bash
2  #
3  # This script will check the pending meta files and arm them according to a priority
       setting
4  #
5  # Development log:
6  #     - 15-11-2015, Bart Root: Initial development
7  #     - 10-05-2022, Amber Sprenkels: Update for DopTrackBox and python 3.0
8  #
9  #-------------------------------------------------------------------------------
10 
11 LOC_PEN='/home/amber/DTB/Recordings/REC_PENDING/'
12 LOC_ARM='/home/amber/DTB/Recordings/REC_ARMED/'
13 LOC_REC='/home/amber/DTB/Recordings/data/'
14 max_prio=$(cat /home/amber/DTB/rec.list | wc -l)
15 
16 # get all the meta files
17 ls $LOC_PEN | grep ".yml" > pending.list
18 ls $LOC_ARM | grep ".yml" > armed.list
19 grep -vf pending.list armed.list > old.list
20 grep -vf armed.list pending.list > new.list
```

```
grep -f pending.list armed.list > double.list

# add all the new metafiles to the armed list
numl=$(cat new.list | wc -l)
echo "Amount of new recordings: $numl"

for i in `seq 1 $numl`;
do
        line=$(tail -n+$i new.list | head -n1)
        mv $LOC_PEN$line $LOC_ARM$line
done

# with old recordings check if time of recording is after time of this update and remove past recordings
ntime=$(date +"%Y%m%d%H%M")
numl=$(cat old.list | wc -l)
echo "Amount of old recordings: $numl"
for i in `seq 1 $numl`;
do
        line=$(tail -n+$i old.list | head -n1)
  otime=$(cat $LOC_ARM$line | grep "Start of recording" | awk '{print $4}')
        # if old file has start of recording less then previous TLE update
        if [ "$ntime" -gt "$otime" ]; then
            # old file is removed
            echo "Old file is removed: $LOC_ARM$line"
            rm $LOC_ARM$line
        fi
done


# with doubles check the time of TLE update and remove oldest
numl=$(cat double.list | wc -l)
echo "Amount of double recordings: $numl"

for i in `seq 1 $numl`;
do
        line=$(tail -n+$i double.list | head -n1)
        ntime=$(cat $LOC_PEN$line | grep "time used UTC" | awk '{print $4}')
        otime=$(cat $LOC_ARM$line | grep "time used UTC" | awk '{print $4}')
        # if new file has improved TLE time
        if [ "$ntime" -gt "$otime" ]; then
            echo "Updated double record to newer (TLE) version: $line"
      mv $LOC_PEN$line $LOC_ARM$line
  elif [ "$ntime" -eq "$otime" ]; then
            echo "Updated double record to newer (TLE) version: $line"
      mv $LOC_PEN$line $LOC_ARM$line
        else
            rm $LOC_PEN$line
        fi
done


for p in `seq 1 $max_prio`
do
        # update armed.list
  rm armed.list
  ls $LOC_ARM | grep ".yml" > armed.list
  echo "Priority check: $p"

  # check for priority
  numl=$(cat armed.list | wc -l)

  for i in `seq 1 $numl`;
  do
          record=$(tail -n+$i armed.list | head -n1)
    if [ -f $LOC_ARM$record ];then
            prio=$(cat $LOC_ARM$record | grep "Priority" | awk '{print $2}')
                  if [ "$prio" -eq "$p" ]; then
              start_rec=$(cat $LOC_ARM$record | grep "Start of recording" | awk '{print $4}')
              year=$(echo $start_rec | cut -c1-4)
              month=$(echo $start_rec | cut -c5-6)
```

```
91              day=$(echo $start_rec | cut -c7-8)
92              hour=$(echo $start_rec | cut -c9-10)
93              minute=$(echo $start_rec | cut -c11-12)
94              lofp=$(cat $LOC_ARM$record | grep "Length of pass" | awk '{print $4}')
95              end_rec=$(date -d "${year}-${month}-${day} ${hour}:${minute} $lofp seconds
    " +%Y%m%d%H%M)
96              # make selection that overlaps the selected recording
97              for a in `seq 1 $numl`
98              do
99                      rec_test=$(tail -n+$a armed.list | head -n1)
100                     if ! [ $rec_test == $record ]; then
101                         # not the same record, then check if overlapping
102             if [ -f $LOC_ARM$rec_test ];then
103                         start_test=$(cat $LOC_ARM$rec_test | grep "Start of
    recording" | awk '{print $4}')
104             year=$(echo $start_test | cut -c1-4)
105                                 month=$(echo $start_test | cut -c5-6)
106                                 day=$(echo $start_test | cut -c7-8)
107                                 hour=$(echo $start_test | cut -c9-10)
108                                 minute=$(echo $start_test | cut -c11-12)
109                                 lofp=$(cat $LOC_ARM$rec_test | grep "Length of
    pass" | awk '{print $4}')
110                                 end_test=$(date -d "${year}-${month}-${day} ${hour
    }:${minute} $lofp seconds" +%Y%m%d%H%M)
111             prio_test=$(cat $LOC_ARM$rec_test | grep "Priority" | awk '{print $2}')
112             # check if start time is overlapping
113                         if  [ "$start_rec" -le "$start_test" -a "$start_test" -le "
    $end_rec" ]; then
114                             # recordings are overlapping! Check priority and remove
    lowest priority
115               if [ "$prio_test" -gt "$prio" ]; then
116                 echo "Overlapping file is removed: $LOC_ARM$rec_test"
117                                 rm $LOC_ARM$rec_test
118                                                         elif [ "$prio_test" -eq
    "$prio" ]; then
119                 # similar priority recordings, but off with one or two minutes
120                 # check the newest TLE propagation
121                 ntime=$(cat $LOC_ARM$rec_test | grep "time used UTC" | awk '{print $4}
    ')
122                       otime=$(cat $LOC_ARM$record | grep "time used UTC" | awk '{print
     $4}')
123                       # if new file has improved TLE time
124                       if [ "$ntime" -gt "$otime" ]; then
125                     # don't do anything. In the following i loop the record will be
    removed
126                     otime=$otime
127                             else
128                         echo "Old file is removed: $LOC_ARM$rec_test"
129                           rm $LOC_ARM$rec_test
130                       fi
131               fi
132                         fi
133             if [ -f $LOC_ARM$rec_test ];then
134               # check if end time is overlapping
135                                                       if [ "$start_rec" -le "
    $end_test" -a "$end_test" -le "$end_rec" ]; then
136                 # recordings are overlapping! Check priority and remove lowest
    priority
137                                                             if [ "$prio_test" -gt
    "$prio" ]; then
138                                                                     echo "
    Overlapping file is removed: $LOC_ARM$rec_test"
139                                                                     rm
    $LOC_ARM$rec_test
140                                                             elif [ "$prio_test" -
    eq "$prio" ]; then
141                                                                     # similar
    priority recordings, but off with one or two minutes
142                                                                     # check the
    newest TLE propagation
143                                                                     ntime=$(cat
    $LOC_ARM$rec_test | grep "time used UTC" | awk '{print $4}')
```

```
144                                                
```

```
numj=$(cat armed.list | wc -l)
if [ $numj > 0 ]; then
  # now remove all pending at jobs, such that the new jobs are refreshed
  for j in `atq | awk '{print $1}'`; do atrm $j;done
fi

# update the atq list
for t in `seq 1 $numj`
do
  # set the new Recording of Doptrack
        line=$(tail -n+$t armed.list | head -n1)
        Stime=$(cat $LOC_ARM$line | grep "Start of recording" | awk '{print $4}')
        #echo "python Record.py -i $line at time: $Stime"
        #echo "python Record.py -i $line" | at -t $Stime
  echo "python3 Record_DTB.py -i $line" | at -t $Stime
done

rm pending.list
rm armed.list
rm old.list
rm new.list
rm double.list
```

B.2. RECORDER SOFTWARE

Software part of the recorder package discussed in Section 2.6.2 can be found here.

B.2.1. RECORD_DTB.PY

```
#!/usr/bin/pytho::
#
# This function extracts data from the meta-file and starts a recording using the USRP
#
# Development log:
#
# - Bart Root, 14-12-2015: Initial development
#       - Amber Sprenkels, 29-08-2022: DopTrackBox adaptations & Python 3 updates
#
#####################################################

# import libraries

from __future__ import print_function

import sys, getopt
import os
import os.path
import yaml
import subprocess
import datetime
import time
import pmt

# initialisation of global variables
LOC_ARM = '/home/amber/DTB/Recordings/REC_ARMED/'
LOC_REC = '/home/amber/DTB/Recordings/data/'
LOC_ERR = '/home/amber/DTB/Recordings/REC_ERROR/'
LOC_RAD = '/home/amber/DTB/rx_tools/'

def get_time():
    t_str = datetime.now()
    return t_str

def main(argv):
    inputfile = ''
    try:
        opts, args = getopt.getopt(argv, "hi:", ["ifile="])
    except getopt.GetoptError:
        print('test.py -i <inputfile>')
        sys.exit(2)
    for opt, arg in opts:
```

```
43          if opt == '-h':
44              print('test.py -i <inputfile>')
45              sys.exit()
46          elif opt in ("-i", "--ifile"):
47              inputfile = arg
48
49      # load meta-file
50      inputstr = LOC_ARM + str(inputfile)
51      with open(inputstr, 'r') as metaf:
52          meta = yaml.load(metaf)
53
54      # set the parameters for the recording
55      NORADID = meta['Sat']['State']['NORADID']
56      name = meta['Sat']['State']['Name']
57      freq = meta['Sat']['State']['Tuning Frequency']
58      antenna = meta['Sat']['State']['Antenna']
59      samp_rate = meta['Sat']['Record']['sample_rate']
60      num_samp = meta['Sat']['Record']['num_sample']
61      STime = meta['Sat']['Record']['Start of recording']
62
63      # set recording
64      filename = str(name) + '_' + str(NORADID) + '_' + str(STime)
65      filepath = str(LOC_REC) + filename + '.32fc'
66      start_rec_cmd = ' -f ' + str(freq) + ' -F CF32 -I CF32 -s ' + str(samp_rate) + ' -n
        ' + str(num_samp) + ' -a "Tuner 1 50 ohm" ' + str(filepath)
67      time3 = datetime.datetime.now()
68      # get time1
69      time1 = datetime.datetime.utcnow()
70
71      # start recording
72      time.sleep(1)  # wait 1 sec to allow next pps to occur and be set before recording
73      BACK = os.getcwd()
74      os.chdir(LOC_RAD)
75      #initiate recording
76      run_cmd = LOC_RAD + 'rx_sdr' + str(start_rec_cmd)
77      print (run_cmd)
78      subprocess.run([run_cmd],shell=True)
79      #subprocess.call([run_cmd])
80      #subprocess.call(['./rx_sdr', str(start_rec_cmd)])
81      os.chdir(BACK)
82
83      # get time 2
84      time2 = datetime.datetime.utcnow()
85
86      # fill in the rest of meta file
87      meta['Sat']['Record']['time1 UTC'] = time1
88      meta['Sat']['Record']['time2 UTC'] = time2
89      meta['Sat']['Record']['time3 LT'] = time3
90      #meta['Sat']['Record']['time pps'] = time_pps
91
92      # put data meta file in recording directory
93      data_out = LOC_REC + filename + '.32fc'
94      if os.path.isfile(data_out):
95          meta_out = LOC_REC + filename + '.yml'
96          with open(meta_out, 'w') as outfile:
97              outfile.write(yaml.dump(meta, default_flow_style=False))
98          outfile.close()
99          os.remove(inputstr)
100
101     else:
102         meta_err = LOC_ERR + filename + '.yml'
103         with open(meta_err, 'w') as outfile:
104             outfile.write(yaml.dump(meta, default_flow_style=False))
105         outfile.close()
106         os.remove(inputstr)
107
108 if __name__ == "__main__":
109     main(sys.argv[1:])
```

B.3. PROCESSING SOFTWARE

Software part of the processing package discussed in Section 2.6.3 can be found here.

B.3.1. CURVETOOL.PY

```
from doptrack.recording import Recording
import doptrack.processing as processing
import doptrack.signals as signals
from cmcrameri import cm
from os.path import exists
import matplotlib.pyplot as plt
import numpy as np

print ('Starting curvetool')

### inputs
# Data input settings
LOC = '/media/amber/ADATA SD600Q/data/Experiment_1/'
#LOC = '/media/amber/ADATA SD600Q/data/Experiment_2/DopTrack_Reference/'
#LOC = '/home/amber/DTB/Recordings/data/'
FILE = 'NAYIF_42017_202209011219'
#FILE = 'Nayif-1_42017_202210251229'
INFO = '(DTB Light East pass - 33 max elevation)'
#FILE = 'Delfi-N3XT_39428_202211131322'
TUNE = 145940000  #Nayif
#TUNE = 145870000  #Delfi Next

# Plot and processed data output settings
LOCi = '/media/amber/ADATA SD600Q/data/Experiment_2/'
PLOT_LOC = LOCi + 'Plots/'
PROCESS_LOC = LOCi + 'Processed/'

# Colourbar settings. Set colour_auto to 1 for TRUE and 0 for FALSE. min/max only work if auto is false!
colours = cm.lapaz
colour_auto = 0
colour_min = -5
colour_max = 20

params = {'axes.labelsize': 20,'axes.titlesize': 20, 'xtick.labelsize': 16, 'ytick.labelsize': 16, 'figure.titlesize': 20, 'legend.fontsize': 16}
plt.rcParams.update(params)

# Noise calculation settings. Set noise_calc to TRUE to enable noise post processing calculations or to FALSE to disable them.
# Choose the frequency and time points for which you want specific signal strength plots. If set to a negative value, the plot is disabled.
noise_calc = 0
frequency_point = 2000
time_point = 420

########################
### assemble
########################
print ('Loading dataset...')
META = LOC + FILE +'.yml'
DATA = LOC + FILE +'.32fc'
FREQ = TUNE + 70000
#FREQ = TUNE
#FREQ = TUNE - 50000

### code
rec = Recording.load(metafile=META , datafile=DATA)
print ('Dataset',FILE,'loaded...')
print ('Dataset information provided:',INFO)
print ('Creating spectrogram...')

"""
The ‘dt‘ value is the desired time resolution of the spectrogram. The ‘center_frequency‘ is the estimated frequency of the satellite signal.

This function zooms in on the area just around the estimated satellite signal.
```

```
If you are interested in the full spectrogram you should use the
‘doptrack.signals.create_spectrogram()‘ function instead.
But be warned that the full spectrogram takes quite a lot of memory to create,
and the plotting can be really slow.
"""
# dt = 1 has the correct timescale on the y axis
#spec = signals.create_spectrogram(data=rec, sampling_rate=250000, dt=1)
spec = processing.create_spectrogram_product(recording=rec, dt=1, center_frequency=FREQ)

fig, ax = plt.subplots()
# Converting the SNR data in spec.image to dB
img = ax.imshow(np.fliplr(10*np.log10(spec.image)), aspect='auto', cmap=colours) #use
    for DTB
#img = ax.imshow(10*np.log10(spec.image), aspect='auto', cmap=colours) #use for DopTrack
cbar = fig.colorbar(img, ax=ax)

if colour_auto==1:
  colour_max = 'auto'
  print ('Using automatic colour bounds...')
else:
  img.set_clim(colour_min,colour_max)
  print ('Using set colour bounds from', colour_min, 'to', colour_max,'...')

# Axis labels
#x_text = 'Frequency 'Frequency offset to ' + str(TUNE/1e6) + 'MHz [Hz]'
plt.xlabel('Frequency offset to ' + str(TUNE/1e6) + 'MHz [Hz]')
plt.ylabel('Time [s]')
cbar.set_label('SNR [dB]')
plt.title('Spectrogram of ' + FILE + ' ' + INFO)

# Plotting and automatic saving
scale = 2.5
fig.set_size_inches(19.2, 10.8)
fig.savefig(PLOT_LOC + 'Spectrogram ' + FILE + '_cmax='+str(colour_max), dpi=scale*100,
    bbox_inches = 'tight')
plt.show()

# Original spectrogram plot (no colourbar and custom colours)
#spec.plot(clim=(0.1,1))

print ('Spectrogram plotted...')
print ('Calculating noise floor...')

NOISE = FILE +'_NOISE.dat'
rewrite_noise = exists(PROCESS_LOC + NOISE)

noise_floor = np.transpose(spec.noise)

if rewrite_noise==0:
  print ('Creating noise floor data file...')
  np.savetxt(PROCESS_LOC + NOISE, spec.noise)
  print ('Noise floor data written to file',NOISE)
  print (' ')
else:
  print ('Noise floor data already exists and has therefore not been updated')
  print (' ')

fig = plt.figure()
plt.plot(spec.noise,spec.time)
#plt.axis([0,3,np.max(spec.time),0])
plt.axis([0,np.nanmax(spec.noise)*1.1,np.max(spec.time),0])
#plt.axis([0,np.max(spec.noise)*1.1,np.max(spec.time),0])
#plt.axis([np.min(spec.noise)*1.1,np.max(spec.noise)*1.1,np.max(spec.time),0])

#plt.title('Noise floor for ' + FILE + ' ' + INFO, wrap=True) #, fontsize=22)
plt.title('Noise floor') # smaller title for quarter size
plt.xlabel('Noise strength [-]') #, fontsize=22)
plt.ylabel('Time [s]') #, fontsize=22)

#fig.set_size_inches(19.2, 10.8) #default size
fig.set_size_inches(4.8, 10.8)  #quarter size
```

```
133 fig.savefig(PLOT_LOC + 'Spectrogram ' + FILE + ' Noise floor', dpi=scale*100,
        bbox_inches = 'tight')
134 plt.show()
135
136 print ('Extracting signal data...')
137 s_avg = spec.signal_average
138 s_max = spec.signal_maximum
139 s_med = spec.signal_median
140
141 print ('Calculating Signal to Noise ratios...')
142 SNR = FILE +'_SNR.dat'
143 rewrite_snr = exists(PROCESS_LOC + SNR)
144
145 SNR_avg = [i/j for i,j in zip(s_avg,spec.noise)]
146 SNR_max = [i/j for i,j in zip(s_max,spec.noise)]
147 SNR_med = [i/j for i,j in zip(s_med,spec.noise)]
148
149 if rewrite_snr==0:
150   print ('Creating SNR data file...')
151   SNR_out = np.column_stack((SNR_avg,SNR_max,SNR_med))
152   #SNR_out = np.hstack((SNR_avg,SNR_max))
153   #SNR_out = np.hstack((SNR_out,SNR_med))
154   np.savetxt(PROCESS_LOC + SNR, SNR_out)
155   print ('SNR data written to file',SNR)
156   print (' ')
157 else:
158   print ('SNR data already exists and has therefore not been updated')
159   print (' ')
160
161 fig=plt.figure()
162 fig, axs = plt.subplots(1,2)
163 fig.suptitle('SNR for ' + FILE + ' ' + INFO)
164
165 axs[0].plot(spec.time, SNR_avg, label='Mean SNR')
166 axs[0].plot(spec.time, SNR_med, label='Median SNR')
167 axs[0].axis([0,np.max(spec.time),np.nanmin(SNR_avg)*0.75,np.nanmax(SNR_avg)*1.05])
168 #axs[0].axis([0,np.max(spec.time),np.min(SNR_avg)*0.75,np.max(SNR_avg)*1.05])
169 #axs[0].axis([0,np.max(spec.time),2,7])
170
171 axs[1].plot(spec.time, SNR_max,'g',label='Maximum SNR')
172 axs[1].axis([0,np.max(spec.time),0,np.nanmax(SNR_max)*1.05])
173 #axs[1].axis([0,np.max(spec.time),0,40])
174
175 axs[0].set_xlabel('Time [s]')
176 axs[0].set_ylabel('SNR [dB]')
177 axs[0].legend()
178 axs[1].set_xlabel('Time [s]')
179 axs[1].set_ylabel('SNR [dB]')
180 axs[1].legend()
181
182
183 fig.set_size_inches(19.2, 10.8)
184 fig.savefig(PLOT_LOC + 'Spectrogram ' + FILE + ' SNR', dpi=scale*100, bbox_inches = '
        tight')
185 plt.show()
186
187 PROCESS = FILE +'.dat'
188 rewrite = exists(PROCESS_LOC + PROCESS)
189
190 if rewrite==0:
191   print ('Creating output data file...')
192   np.savetxt(PROCESS_LOC + PROCESS, spec.image)
193   print ('Processed spectrogram data written to file',PROCESS)
194   print (' ')
195 else:
196   print ('Processed spectrogram data file already exists and has therefore not been
        updated')
197   print (' ')
198
199 # Noise data plots
200 if noise_calc==1:
201   print ('Post processing noise data...')
```

```
  if frequency_point<0:
    print ('Calculations for a specific frequency through time have been disabled')
  else:
    abs_freq = (TUNE+frequency_point)/1e6
    freq_point = frequency_point - 1
    time_data = spec.image[:,freq_point]
    time_avg = np.average(time_data)
    time_avg_complete = np.ones(time_data.size)*time_avg
    print ('Average noise value at f= ',abs_freq,'MHz:', time_avg)

    plt.plot(time_data)
    plt.plot(time_avg_complete, 'r--')
    plt.axis([0, time_data.size, 0, np.max(time_data)*1.1])

    # Axis labels
    plt.title('Noise over time at ' + str(abs_freq) + ' MHz for ' + FILE + ' ' + INFO)
    plt.xlabel('Time [s]')
    plt.ylabel('Noise strenght [?]')
    plt.show()

  if time_point<0:
    print ('Calculations for a specific time over all frequencies are disabled')
  else:
    ti_point = time_point - 1
    freq_data = spec.image[ti_point,:]
    freq_avg = np.average(freq_data)
    freq_avg_complete = np.ones(freq_data.size)*freq_avg
    print ('Average noise value at t= ', time_point, 'seconds:',freq_avg)

    plt.plot(freq_data)
    plt.plot(freq_avg_complete, 'r--')
    plt.axis([0, freq_data.size, 0, np.max(freq_data)*1.1])

    #Axis labels
    plt.title('Signal strength at ' + str(time_point) + ' seconds for ' + FILE + ' ' +
    INFO)
    plt.xlabel('Frequency offset to ' + str(TUNE/1e6) + 'MHz [Hz]')
    plt.ylabel('Signal strength [?]')
    plt.show()
else:
  print ('Noise data calculations have been set to disabled in the program settings by
    the user')

print ('Program finished')
```

B.3.2. PROCESSING.PY

```
from dataclasses import dataclass
from datetime import datetime, timedelta
from pathlib import Path
from typing import Optional, Callable

import dacite
import numpy as np
from scipy.optimize import brentq, bisect
from scipy.stats import linregress

import doptrack.io
from doptrack import constants as constants
from doptrack.astro import GroundStation, TLESatellite
from doptrack.recording import Metadata, Recording
from doptrack.signals import extract_s_curve, Spectrogram, create_spectrogram
from doptrack.utils import ArrayComparisonMixin, FilePath

@dataclass(frozen=True, eq=False)
class SpectrogramProduct(ArrayComparisonMixin, Spectrogram):
    meta: Metadata
    epoch: datetime

    def save(self, metafile: FilePath, datafile: FilePath) -> None:
        """Save the date to files."""
        doptrack.io.write_metadata_to_yml(Path(metafile), self)
```

```
26          doptrack.io.write_arraydata_to_npz(Path(datafile), self)
27
28      @classmethod
29      def load(cls, metafile: FilePath, datafile: FilePath) -> 'SpectrogramProduct':
30          """Load the date from files."""
31          metadict = doptrack.io.read_metadata_from_yml(Path(metafile))
32          arraydict = doptrack.io.read_arraydata_from_npz(Path(datafile))
33          spetrogram_metadata = metadict.pop('spectrogram')
34          return dacite.from_dict(
35              cls, dict(meta=metadict, **spetrogram_metadata, **arraydict),
36          )
37
38
39  def create_spectrogram_product(recording: Recording, dt: float, center_frequency: int =
        0) -> SpectrogramProduct:
40      """
41      Creates a spectrogram from chunked recording data.
42
43      The default spectrogram window width should be wide enough to
44      show the whole S-curve for LEO satellites.
45      """
46      spectrogram = create_spectrogram(
47          data=recording.load_data(dt),
48          sampling_rate=recording.sampling_rate,
49          dt=dt,
50          mask_width=14000,  # Should be wide enough to show the whole S-curve for LEO
        satellites.
51          mask_center=center_frequency - recording.tuning_frequency,
52      )
53
54      return SpectrogramProduct(
55          meta=recording.meta,
56          epoch=recording.time_start,
57          dt=dt,
58          df=spectrogram.df,
59          image=spectrogram.image.astype(np.float16),
60          time=spectrogram.time,
61          frequency=spectrogram.frequency + recording.tuning_frequency,
62          noise=spectrogram.noise,
63          signal_average=spectrogram.signal_average,
64          signal_maximum=spectrogram.signal_maximum,
65          signal_median=spectrogram.signal_median,
66      )
67
68
69  @dataclass(frozen=True, eq=False)
70  class TrackingProduct(ArrayComparisonMixin):
71      meta: Metadata
72      epoch: datetime
73      time: np.ndarray
74      frequency: np.ndarray
75      rangerate: np.ndarray
76      tca: datetime
77      tca_error: float
78      fca: float
79      fca_error: float
80
81      def save(self, metafile: FilePath, datafile: FilePath) -> None:
82          doptrack.io.write_metadata_to_yml(Path(metafile), self)
83          doptrack.io.write_arraydata_to_csv(Path(datafile), self)
84
85      @classmethod
86      def load(cls, metafile: FilePath, datafile: FilePath) -> 'TrackingProduct':
87          metadict = doptrack.io.read_metadata_from_yml(Path(metafile))
88          arraydict = doptrack.io.read_arraydata_from_csv(Path(datafile))
89          tracking_metadata = metadict.pop('tracking')
90          return dacite.from_dict(
91              cls, dict(meta=metadict, **tracking_metadata, **arraydict),
92          )
93
94
95  def create_tracking_product(spectrogram: SpectrogramProduct, sidelobe_distance: int) ->
```

```
    TrackingProduct:
    s_curve = extract_s_curve(spectrogram, sidelobe_distance=sidelobe_distance)
    rangerate = (1 - (s_curve.frequency / s_curve.fca)) * constants.c
    return TrackingProduct(
        meta=spectrogram.meta,
        epoch=spectrogram.epoch,
        time=s_curve.time,
        frequency=s_curve.frequency,
        rangerate=rangerate,
        tca=spectrogram.epoch + timedelta(seconds=s_curve.tca),
        tca_error=float(s_curve.tca_error),  # Call float() since yaml does not like the np.float type
        fca=float(s_curve.fca),
        fca_error=float(s_curve.fca_error),
    )


@dataclass(frozen=True, eq=False)
class ValidationProduct(ArrayComparisonMixin):
    meta: Metadata
    epoch: datetime
    time: np.ndarray
    frequency: np.ndarray
    rangerate: np.ndarray
    rangerate_tle: np.ndarray
    first_residual: np.ndarray
    second_residual: np.ndarray

    tca: datetime
    tca_error: float
    tca_tle: datetime
    dtca: float

    fca: float
    fca_error: float

    max_elevation: float
    time_since_eclipse: Optional[float]  # Optional since some satellites might never be in eclipse.
    first_residual_fit_slope: float
    second_residual_rmse: float

    def save(self, metafile: FilePath, datafile: FilePath) -> None:
        doptrack.io.write_metadata_to_yml(Path(metafile), self)
        doptrack.io.write_arraydata_to_csv(Path(datafile), self)

    @classmethod
    def load(cls, metafile: FilePath, datafile: FilePath) -> 'ValidationProduct':
        metadict = doptrack.io.read_metadata_from_yml(Path(metafile))
        arraydict = doptrack.io.read_arraydata_from_csv(Path(datafile))
        validation_metadata = metadict.pop('validation')
        return dacite.from_dict(
            cls, dict(meta=metadict, **validation_metadata, **arraydict),
        )


def create_validation_product(data: TrackingProduct, station: GroundStation, satellite: TLESatellite) -> ValidationProduct:
    times = [data.epoch + timedelta(seconds=t) for t in data.time]
    rangerate_tle = np.array([station.rangerate(t, satellite) for t in times])
    first_residual = data.rangerate - rangerate_tle
    fit = linregress(data.time, first_residual)
    second_residual = first_residual - (fit.slope * data.time + fit.intercept)

    second_residual_rmse = np.sqrt(
        sum(second_residual ** 2) / len(second_residual)
    )

    def _calculate_rangerate(t: float) -> float:
        time = data.epoch + timedelta(seconds=t)
        return station.rangerate(time, satellite)

```

```
    tca_tle_relative: float = brentq(_calculate_rangerate, 0, data.time[-1] + 100)
    tca_tle = data.epoch + timedelta(seconds=tca_tle_relative)
    max_elevation = station.aer(tca_tle, satellite).elevation
    dtca = (data.tca - tca_tle).total_seconds()

    time_since_eclipse = _find_time_since_last_eclipse(satellite, tca_tle)

    return ValidationProduct(
        meta=data.meta,
        epoch=data.epoch,
        tca=data.tca,
        tca_error=data.tca_error,
        fca=data.fca,
        fca_error=data.fca_error,
        time=data.time,
        frequency=data.frequency,
        rangerate=data.rangerate,
        rangerate_tle=rangerate_tle,
        first_residual=first_residual,
        second_residual=second_residual,
        max_elevation=max_elevation,
        tca_tle=tca_tle,
        dtca=dtca,
        time_since_eclipse=time_since_eclipse,
        first_residual_fit_slope=float(fit.slope),
        second_residual_rmse=float(second_residual_rmse),
    )


def _find_time_since_last_eclipse(satellite: TLESatellite, tca: datetime) -> Optional[
    int]:
    """Accurate only to 1 second."""

    if satellite.is_in_eclipse(tca):
        return 0

    increment = 15 * 60  # Needs to be shorter than the duration a satellite is in
    eclipse
    time_a = 0
    while time_a > -100 * 60:
        time_a, time_b = time_a - increment, time_a
        if satellite.is_in_eclipse(tca + timedelta(seconds=time_a)):
            break
    else:
        # If the satellite is in a sun-synchronous orbit it might never be in eclipse.
        return None

    time = _boolean_bisect(
        lambda t: satellite.is_in_eclipse(tca + timedelta(seconds=t)),
        time_a,
        time_b,
        xtol=0.5,
    )
    return round(time)


def _boolean_bisect(func: Callable[[float], bool], a: float, b: float, xtol: float) ->
    float:
    def transfer(x: float) -> int:
        if func(x):
            return 1
        else:
            return -1

    result: float = bisect(transfer, a, b, xtol=xtol)
    return result
```

B.3.3. SIGNALS.PY

```
import logging
from dataclasses import dataclass
from typing import Optional, Iterable, Callable, Union
```

```
4
5  import matplotlib
6  import numpy as np
7  from matplotlib import pyplot as plt
8  from numpy.fft import fftshift, fftfreq, fft
9  from scipy.optimize import curve_fit
10 from scipy.signal import fftconvolve
11 from scipy.stats import linregress
12 from sklearn.cluster import DBSCAN
13
14 from doptrack.base import DoptrackError
15
16 logger = logging.getLogger(__name__)
17
18
19 @dataclass(frozen=True)
20 class Spectrogram:
21     image: np.ndarray
22     time: np.ndarray
23     frequency: np.ndarray
24     dt: float
25     df: float
26     noise: np.ndarray
27     signal_average: np.ndarray
28     signal_maximum: np.ndarray
29     signal_median: np.ndarray
30
31     def __post_init__(self):
32         assert len(self.image.shape)
33         assert len(self.time.shape) == 1 and len(self.time) == self.image.shape[0]
34         assert len(self.frequency.shape) == 1 and len(self.frequency) == self.image.
       shape[1]
35
36     def plot(self, ax: Optional[matplotlib.axes.Axes] = None, **kwargs):
37         """
38         Plot the spectrogram using matplotlib.
39
40         Parameters
41         ----------
42         ax :
43             The axes on which to plot the spectrogram. If no axes are given then a new
       plot will be made.
44         **kwargs :
45             Additional settings passed on to imshow.
46         """
47         extent = (self.frequency[0], self.frequency[-1], self.time[-1], self.time[0])
48         clim = (0.01, 0.2)
49         is_standalone_plot = ax is None
50         if is_standalone_plot:
51             fig, ax = plt.subplots()
52         options = dict(cmap="viridis", aspect="auto", extent=extent, clim=clim)
53         options.update(**kwargs)
54         ax.imshow(self.image.astype(np.float64), **options)
55         if is_standalone_plot:
56             plt.show()
57
58
59 def create_spectrogram(
60         data: Iterable[np.ndarray],
61         sampling_rate: int,
62         dt: float,
63         mask_width: Optional[int] = None,
64         mask_center: int = 0
65 ) -> Spectrogram:
66     """
67     Creates a spectrogram from array data.
68
69     The default spectrogram window width should be wide enough to
70     show the whole S-curve for LEO satellites.
71
72
73     """
```

```
74      df = 1/dt
75      nfft = int(dt * sampling_rate)  # Use optimal value of nfft
76      frequency = fftshift(fftfreq(nfft, 1/sampling_rate))
77
78      if mask_width:
79          lower = mask_center - mask_width / 2
80          upper = mask_center + mask_width / 2
81          # Exclude upper limit from mask to give nicer mask length, e.g. 2800 instead of
        2801 when including upper limit.
82          mask, = np.nonzero((lower <= frequency) & (frequency < upper))
83          frequency = frequency[mask]
84      else:
85          mask = None
86
87      rows = []
88      noise = []
89      signal_average = []
90      signal_maximum = []
91      signal_median = []
92      for i, chunk in enumerate(data):
93          row = abs(fftshift(fft(chunk, nfft)))
94          row = row[mask] if mask is not None else row
95          # noise calculation using standard deviation
96          mean = np.average(row)
97          deviation = np.std(row, dtype=np.float64)
98          signal_limit = mean + 2*deviation
99          noise_floor = []
100         signal_line = []
101         for entry in row:
102             if entry < mean+deviation and entry > mean-deviation:
103                 noise_floor.append(entry)
104             if entry >= signal_limit:
105                 signal_line.append(entry)
106         noise.append(np.average(noise_floor))
107         signal_average.append(np.average(signal_line))
108         signal_maximum.append(np.max(signal_line,initial=0))
109         signal_median.append(np.median(signal_line))
110         row = row/np.average(noise_floor)
111         # noise calculation using only the mean
112         #noise.append(np.average(row))
113         #row = row/np.average(row)
114         rows.append(row)
115     image = np.array(rows)
116     time = np.arange(image.shape[0]) * dt
117
118      #plt.plot(noise,time)
119      #plt.show()
120     return Spectrogram(
121         image=image,
122         time=time,
123         frequency=frequency,
124         dt=dt,
125         df=df,
126         noise=noise,
127         signal_average=signal_average,
128         signal_maximum=signal_maximum,
129         signal_median=signal_median,
130         )
131
132
133 class SignalNotFound(DoptrackError):
134     """Raised when the extraction algorithm is unable to find a satellite signal (S-
        curve) in the spectrogram."""
135     pass
136
137
138 @dataclass
139 class Signal:
140     """A discrete-time signal."""
141     time: np.ndarray
142     frequency: np.ndarray
143
```

```
144
145 @dataclass
146 class SCurve(Signal):
147     """A discrete-time signal representing an S-curve."""
148     tca: float
149     tca_error: float
150     fca: float
151     fca_error: float
152
153
154 @dataclass
155 class Fit:
156     model: Callable
157     coeffs: np.ndarray
158     covar: np.ndarray
159
160     def __call__(self, x: Union[float, np.ndarray]) -> Union[float, np.ndarray]:
161         return self.model(x, *self.coeffs)
162
163
164 def s_curve_model(
165         t: float, tca: float, fca: float, a1: float, a2: float, b1: float, b2: float
166 ) -> Union[float, np.ndarray]:
167     """A function that fits well to S-curves.
168
169     Neither arctan nor tanh work perfectly by themselves, but the combination
170     of both fits almost perfectly to an S-curve. However, the extra degrees of
171     freedom result in this model being less robust than a simple arctan or tanh model,
172     and it requires a better initial guess and relatively few outliers in order to
    converge.
173
174     We assume that both functions are zero at tca. This reduces the degrees of freedom
175     during the fitting procedure. This should make the fitting slightly more robust,
176     and it is a more accurate representation of the physical reality.
177     It also makes it trivial to determine both estimated tca and estimated fca from
178     the model parameters.
179     """
180     return a1 * np.arctan((t - tca) / b1) + a2 * np.tanh((t - tca) / b2) + fca
181
182
183 def s_curve_model_robust(x: float, tca: float, fca: float, a: float, b: float) -> Union[
    float, np.ndarray]:
184     """A function that fits robustly to S-curves."""
185     return a * np.arctan((x - tca) / b) + fca
186
187
188 def fit_s_curve(time: np.ndarray, frequency: np.ndarray):
189     p0 = [400, 7000, -2000, -1000, 100, 100]
190     try:
191         coeffs, covar = curve_fit(s_curve_model, time, frequency, p0=p0, loss='soft_l1',
     method='trf')
192     except RuntimeError:
193         raise SignalNotFound('Non-robust fitting of S-curve failed')
194     return Fit(s_curve_model, coeffs, covar)
195
196
197 def fit_robust_s_curve(time: np.ndarray, frequency: np.ndarray):
198     p0 = [400, 7000, -2000, 100]
199     try:
200         coeffs, covar = curve_fit(s_curve_model_robust, time, frequency, p0=p0, loss='
    soft_l1', method='trf')
201     except RuntimeError:
202         raise SignalNotFound('Robust fitting of S-curve failed.')
203     return Fit(s_curve_model_robust, coeffs, covar)
204
205
206 def extract_s_curve(spectrogram: Spectrogram, sidelobe_distance: int, plot=False) ->
    SCurve:
207     image = spectrogram.image / np.median(spectrogram.image, axis=0)
208     dt, df = spectrogram.dt, spectrogram.df
209
210     left_window = create_signal_window(sidelobe_distance=sidelobe_distance, peak=-
```

```
        sidelobe_distance / 2, df=df)
211     center_window = create_signal_window(sidelobe_distance=sidelobe_distance, peak=0, df
        =df)
212     right_window = create_signal_window(sidelobe_distance=sidelobe_distance, peak=
        sidelobe_distance / 2, df=df)
213
214     left_sidelobe = fftconvolve(image, left_window, mode='same')
215     center_lobe = fftconvolve(image, center_window, mode='same')
216     right_sidelobe = fftconvolve(image, right_window, mode='same')
217     convolution = left_sidelobe * center_lobe * right_sidelobe
218
219     signal = get_initial_signal_from_image(convolution, dt=dt, df=df)
220
221     if plot:
222         fig, ax = plt.subplots()
223         fig.suptitle('Initial data points')
224         spectrogram.plot(ax)
225         ax.scatter(signal.frequency + spectrogram.frequency[0], signal.time, 5, color='r
        ')
226         fig.show()
227
228     # INITIAL FILTERING
229     signal = remove_outliers_by_signal_clustering(signal, dt, plot=plot)
230     fit = fit_robust_s_curve(signal.time, signal.frequency)
231     signal = remove_outliers_by_residual_limiting(signal, fit, limit=sidelobe_distance /
         2, plot=plot)
232
233     # INITIAL VALIDATION
234     tca, fca = fit.coeffs[0], fit.coeffs[1]
235     tca_bounds = (spectrogram.time[0], spectrogram.time[-1])
236     if not tca_bounds[0] < tca < tca_bounds[1]:
237         raise SignalNotFound(f'The signal has an invalid tca: {tca_bounds[0]} < {tca} <
        {tca_bounds[1]}')
238     fca_bounds = (0, image.shape[1] * df)
239     if not fca_bounds[0] < fca < fca_bounds[1]:
240         raise SignalNotFound(f'The signal has an invalid fca: {fca_bounds[0]} < {fca} <
        {fca_bounds[1]}')
241     if slope := fit.coeffs[2] > 0:
242         raise SignalNotFound(f'The signal fit must have a negative slope: {slope} < 0')
243
244     # FINAL FILTERING
245     fit = fit_s_curve(signal.time, signal.frequency)
246     signal = remove_outliers_by_residual_clustering(signal, fit, dt=dt, plot=plot)
247
248     # FINAL FITTING
249     # Perform a final fit to get the best possible estimate of TCA and FCA
250     fit = fit_s_curve(signal.time, signal.frequency)
251     tca, fca = fit.coeffs[:2]
252     errors = np.sqrt(np.diag(fit.covar))
253     tca_error, fca_error = errors[0], errors[1]
254     if tca_error > 0.5:
255         # If TCA error is too high we will try and use a more robust fit as a last
        resort
256         fit = fit_robust_s_curve(signal.time, signal.frequency)
257         tca, fca = fit.coeffs[:2]
258         errors = np.sqrt(np.diag(fit.covar))
259         tca_error, fca_error = errors[0], errors[1]
260
261     # Move from relative frequency back to absolute frequency.
262     # Until this point we use relative frequency since fitting is more robust.
263     signal = Signal(signal.time, signal.frequency + spectrogram.frequency[0])
264     fca += float(spectrogram.frequency[0])
265
266     logger.debug(f'Estimated TCA: {tca:.3f} (+-) {tca_error:.3f}')
267     logger.debug(f'Estimated FCA: {fca:.3f} (+-) {fca_error:.3f}')
268
269     # FINAL VALIDATION
270     if len(signal.time) < 50:
271         raise SignalNotFound('The signal has too few data points')
272     if not (signal.time[0] < tca < signal.time[-1]):
273         raise SignalNotFound(
274             f'The estimated TCA is not within the range of the extracted signal: '
```

```
275             f'{signal.time[0]} < {tca} < {signal.time[-1]}')
276     if tca_error > 0.5:
277         # If the TCA error is too high it is a sign of one of two things:
278         # 1) An incorrect cluster wasn't filtered out properly.
279         # 2) Too few points on one side of TCA.
280         # In both cases the data is bad. Testing showed 0.5 to be a good cutoff value.
281         raise SignalNotFound(f'The TCA error is too high: {tca_error} > 0.5')
282 
283     if plot:
284         fig, ax = plt.subplots()
285         fig.suptitle('Final extracted S-curve')
286         spectrogram.plot(ax)
287         ax.scatter(signal.frequency, signal.time, 5, color='r')
288         ax.plot(fit(signal.time) + spectrogram.frequency[0], signal.time, color='lime',
    label='fit')
289         ax.legend()
290         fig.show()
291 
292     return SCurve(
293         time=signal.time, frequency=signal.frequency, tca=tca, tca_error=tca_error, fca=
    fca, fca_error=fca_error
294     )
295 
296 
297 def gaussian(n: np.ndarray, shift: float, std: float) -> np.ndarray:
298     return np.exp(-0.5 * ((n - shift) / std) ** 2)
299 
300 
301 def create_signal_window(sidelobe_distance: int, peak: float, df: float):
302     sidelobe_distance_pixels = sidelobe_distance / df
303     signal_std_pixels = 5
304     peak_pixels = peak / df
305     x = np.arange(int(sidelobe_distance_pixels * 1.5 - 1))
306     window = gaussian(x, shift=len(x) / 2 + peak_pixels, std=signal_std_pixels)
307     return window.reshape((1, len(window)))
308 
309 
310 def get_initial_signal_from_image(image: np.ndarray, dt: float, df: float) -> Signal:
311     pixel_frequency = np.nanargmax(image, axis=1)
312     pixel_time = np.arange(image.shape[0])
313     non_zero = pixel_frequency != 0
314     time = pixel_time[non_zero] * dt
315     frequency = pixel_frequency[non_zero] * df
316     return Signal(time=time, frequency=frequency)
317 
318 
319 def remove_outliers_by_signal_clustering(signal: Signal, dt: float, plot=False) ->
    Signal:
320     normalized_frequency = signal.frequency / 15
321     data = np.dstack((signal.time, normalized_frequency))[0]
322     clustering = DBSCAN(eps=15 * np.sqrt(dt), min_samples=10).fit(data)
323     labels = clustering.labels_
324     logger.debug(f'Found {len(set(labels))} clusters during signal clustering: {set(
    labels)}')
325 
326     if plot:
327         fig, ax = plt.subplots()
328         fig.suptitle('Clusters found during signal clustering')
329         ax.invert_yaxis()
330         for label in set(labels):
331             ax.plot(signal.frequency[labels == label], signal.time[labels == label], '.'
    , label=label)
332             ax.legend()
333         fig.show()
334 
335     # We know that the complete S-curve, as well as each individual segment,
336     # must have a negative slope. Because of this we take all clusters with
337     # positive or near zero slope and designate them as outliers.
338     # This removes a lot of smaller invalid clusters and makes the fitting
339     # process much more likely to succeed.
340     for label in set(labels):
341         x, y = signal.time[labels == label], signal.frequency[labels == label]
```

```
342         slope = linregress(x, y).slope
343         if slope > -0.2:
344             labels[labels == label] = -1
345 
346     outliers = labels == -1
347     not_outliers = labels != -1
348     if outliers.all():
349         raise SignalNotFound('No signal points after signal clustering')
350     time, frequency = signal.time[not_outliers], signal.frequency[not_outliers]
351     time_outliers, frequency_outliers = signal.time[outliers], signal.frequency[outliers
    ]
352     logger.debug(f'Found {len(time)} valid points and {len(time_outliers)} outliers
    during signal clustering.')
353 
354     if plot:
355         fig, ax = plt.subplots()
356         fig.suptitle('Result of signal clustering and outlier detection')
357         ax.invert_yaxis()
358         ax.plot(frequency, time, '.', label='valid points')
359         ax.plot(frequency_outliers, time_outliers, '.', label='outliers')
360         ax.legend()
361         fig.show()
362 
363     return Signal(time=time, frequency=frequency)
364 
365 
366 def remove_outliers_by_residual_clustering(signal: Signal, fit: Callable, dt: float,
    plot=False) -> Signal:
367     residual = signal.frequency - fit(signal.time)
368     data = np.dstack((signal.time, residual))[0]
369     clustering = DBSCAN(eps=20 * np.sqrt(dt), min_samples=10).fit(data)
370     labels = clustering.labels_
371     logger.debug(f'Found {len(set(labels))} clusters during residual clustering: {set(
    labels)}')
372 
373     if plot:
374         fig, ax = plt.subplots()
375         fig.suptitle('Clusters found during residual clustering')
376         for label in set(labels):
377             ax.plot(signal.time[labels == label], residual[labels == label], '.', label=
    label)
378             ax.legend()
379         fig.show()
380 
381     # If the cluster is located too far from the fitting line we tag it as outliers.
382     # Testing showed that a limit value of 10 seemed approriate.
383     for label in set(labels):
384         cluster = residual[labels == label]
385         if abs(np.mean(cluster)) > 10:
386             labels[labels == label] = -1
387 
388     outliers = labels == -1
389     not_outliers = labels != -1
390     if outliers.all():
391         raise SignalNotFound('No signal points after residual clustering')
392     logger.debug(f'Found {sum(not_outliers)} valid points and {sum(outliers)} outliers
    during residual clustering.')
393 
394     if plot:
395         fig, ax = plt.subplots()
396         fig.suptitle('Result of residual clustering and outlier detection')
397         ax.plot(signal.time[not_outliers], residual[not_outliers], '.', label='valid
    points')
398         ax.plot(signal.time[outliers], residual[outliers], '.', label='outliers')
399         ax.legend()
400         fig.show()
401 
402     return Signal(time=signal.time[not_outliers], frequency=signal.frequency[
    not_outliers])
403 
404 
405 def remove_outliers_by_residual_limiting(signal: Signal, fit: Callable, limit: float,
```

```
    plot=False) -> Signal:
406     residual = signal.frequency - fit(signal.time)
407     outliers = abs(residual) > limit
408     not_outliers = abs(residual) <= limit
409     if outliers.all():
410         raise SignalNotFound('No signal points after residual limiting')
411     logger.debug(f'Found {sum(not_outliers)} valid points and {sum(outliers)} outliers
        during residual limiting.')
412 
413     if plot:
414         fig, ax = plt.subplots()
415         fig.suptitle('Result of residual limiting')
416         ax.invert_yaxis()
417         ax.plot(fit(signal.time), signal.time, 'r', label='fit')
418         ax.plot(fit(signal.time) - limit, signal.time, 'y', label='limits')
419         ax.plot(fit(signal.time) + limit, signal.time, 'y')
420         ax.scatter(signal.frequency[not_outliers], signal.time[not_outliers], 5, 'b',
        label='valid points')
421         ax.scatter(signal.frequency[outliers], signal.time[outliers], 5, 'r', label='
        outliers')
422         ax.legend()
423         fig.show()
424     return Signal(signal.time[not_outliers], signal.frequency[not_outliers])
```