Hochschule Wismar
Fakultät für Ingenieurwissenschaften
Bereich Elektrotechnik und Informatik

Master-Thesis

Investigation of Non – Gaussian Characteristics in SIMONe Germany Meteor System

Submitted on : 27 October, 2021

By : Ms. Reshma Mary Roy

Mat. Nr. : 348076

Born on April 16, 1995

in Kerala, India

Supervisors : Prof. Dr.-Ing. habil. Andreas Ahrens

Prof. Dr. Jorge L. Chau

M.Sc. Matthias Clahsen

Abstract

SIMONe is a multistatic radar system, conceived and developed at IAP, used for the detection of meteor trails and deduction of winds from their Doppler velocities. Meteors ablate in the atmosphere at around 90 km; the generated plasma is able to scatter/reflect radio waves. These reflections are detected, decoded and parameters like distance signal strength, Doppler shift, decay time, and duration are derived. These parameters are used to deduce atmospheric neutral winds at these altitudes. In standard analysis, some detections are rejected from further processing to smooth the output (cleaning procedure).

In this thesis, an investigation of these rejected values (outliers) have be done. This was motivated the recent findings at IAP of large geophysical vertical velocities previously not expected

In this work, outliers have been classified into geophysical and non – geophysical based on their characteristics. The data was take from the SIMONe Germany system, namely selected radar raw data and pre-analyzed parameter data, both in HDF5 format. As a part of the analysis, a GUI have been developed to facilitate the verification of the physical events. The classified events have been stored in HDF5 files and the analysis confirms the presence of possible good meteor events in the previously rejected detection. The analysis was performed on data for three months.

Abstrakt

SIMONe ist ein am IAP konzipiertes und entwickeltes multistatisches Radarsystem, das zur Erkennung von Meteorspuren und zur Ableitung von Winden aus deren Doppler-Geschwindigkeiten eingesetzt wird. Meteore verglühen in der Atmosphäre in etwa 90 km Höhe; das dabei entstehende Plasma ist in der Lage, Radiowellen zu streuen/reflektieren. Diese Reflexionen werden detektiert und dekodiert, und es werden Parameter wie Entfernungssignalstärke, Dopplerverschiebung, Abklingzeit und Dauer abgeleitet. Diese Parameter werden verwendet, um auf die atmosphärischen Neutralwinde in diesen Höhen

zu schließen. Bei der Standardanalyse werden einige Erkennungen von der weiteren Verarbeitung ausgeschlossen, um das Ergebnis zu glätten (Reinigungsverfahren).

In dieser Arbeit wurde eine Untersuchung dieser zurückgewiesenen Werte (Ausreißer) durchgeführt. Dies wurde durch die jüngsten Entdeckungen am IAP motiviert, bei denen große geophysikalische Vertikalgeschwindigkeiten festgestellt wurden, die zuvor nicht erwartet worden waren.

In dieser Arbeit wurden die Ausreißer anhand ihrer Merkmale in geophysikalische und nicht-geophysikalische Werte unterteilt. Die Daten wurden aus dem SIMONe Deutschland System entnommen, nämlich ausgewählte Radar-Rohdaten und voranalysierte Parameterdaten, beide im HDF5 Format. Als Teil der Analyse wurde ein GUI entwickelt, um die Überprüfung der physikalischen Ereignisse zu erleichtern. Die klassifizierten Ereignisse wurden in HDF5-Dateien gespeichert und die Analyse bestätigt das Vorhandensein möglicher guter Meteorereignisse in der zuvor abgelehnten Detektion. Die Analyse wurde mit Daten aus drei Monaten durchgeführt.

Contents

1 Introduction

Radar systems has a very important application in the field of atmospheric research. These are commonly known as weather observation radars and are used for the observation of the lower atmosphere. Another category of radar used for the observation of the upper atmosphere are partial reflection radars, used for the altitude of 50-95km or the ionosondes used for the investigation of ionosphere.

Another type of radar system, that is used for the investigation of the upper atmosphere, are the specular meteor radars (SMRs). When meteors enter the atmosphere, they produce plasma trails and the SMRs measure the reflected electromagnetic wave from the trails when observed perpendicular to them. The trails are typically observed between 70-110km. This is used for the estimation of mean winds in the upper atmosphere from the drift velocity of the plasma trails. This thesis focuses on the non-Gaussian characteristics in the meteor radar system, Spread Spectrum Interferometric Multistatic meteor radar Observing Network (SIMONe) ([4]), Germany.

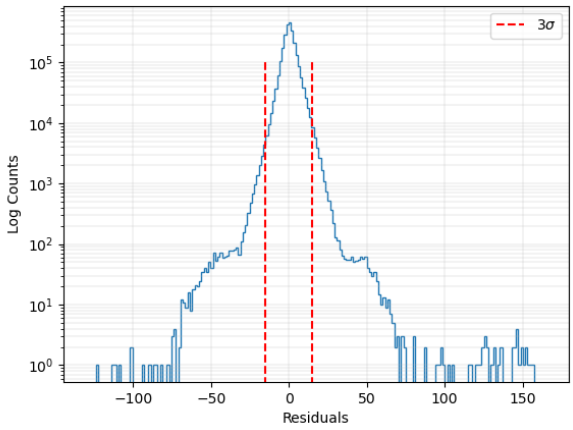

Figure 1: Non – Gaussian Characteristics in SIMONe Germany

A typical Gaussian distribution ([16]) is a continuous probability distribution that has symmetrical sides and is known as a bell curve, due to its resemblance to that of a bell. A common empirical rule that allow as to identify a Gaussian

distribution is the three-sigma rule, also known as the 68-95-99.7 rule. According to this rule 68% data falls within -1 and 1 standard deviation, 95% of data falls between -2 and 2 standard deviation and 99.7% of the data falls between the -3 and 3 standard deviation.

As shown in Figure 1, the distribution of the data obtained from the SMR is non-Gaussian and in the standard procedure the values after the 3 sigma are considered outliers. This thesis particularly focuses on these rejected outliers. This is motivated by the recently detected extreme event ([3]) at IAP.

Extreme events ([3]) are rare but occur in geo-physical flows. Example for such events are tornadoes in the atmosphere or, rogue waves in the ocean. Such kind of extreme events could also occur in the mesosphere and the outer space, however these regions are not thoroughly observed. The coldest region on the earth is the polar summer mesosphere, here mesospheric ice clouds are formed. They produce polar mesospheric summer echoes (PMSE) which are used to study the polar summer mesospheric dynamics.

The extreme vertical draft that motivated this thesis occurred on July 16, 2016 in the mesosphere over northern Norway and had vertical velocities reaching as high as $\pm 50 ms^{-1.}$ These values are greater than 5 times the standard deviation of two months of similar observations. The particular event occurred between 4:25 and 5:00 UT and outside this interval, the value vertical velocity was in the expected range ([3]).

Although, an extreme vertical velocity has been observed, it is difficult to infer about their occurrence ([3]) at this point as the high velocities have been ignored. For the previous observations conducted at IAP, using MAARSY the Nyquist velocity used was around ±30ms-1. This means that the previous events could have been filtered out as outliers or ignored.

2 Multistatic Multi-frequency Agile Radar for Investigation of the Atmosphere

SMRs has been widely used for the observation of horizontal winds in the mesosphere and the lower thermosphere (MLT). When meteors enter the atmosphere, they form a diffusing plasma trail, which last for a fraction of seconds. These plasma trails drift with the atmospheric wind. SMRs measures the line of sight velocity by measuring the backscattered electromagnetic wave. The observation geometry determines the reflection point and is known as the specular point. Mean atmospheric wind is calculated by detecting meteor trails in different directions per time and altitude bins ([6]).

A typical backscatter SMR consists of a receiving interferometer with 5 antennas and a single transmitter antenna. These antennas are usually arranged such that they have a spacing of 2 and 2.5 λ. Mean winds are calculated by averaging all meteor detection for a particular altitude at a time bin (2 to 4 km) of around the 30-120 minutes. Hence, the produced wind vector depends on the number of counts and the detection geometry ([6]).

In order to improve the detection and the wind calculation, IAP conceived a new approach, called Multi-static multi-frequency Agile Investigations of the Atmosphere (MMARIA) ([6]). Unlike the traditional systems, MMARIA operates in bistatic mode, by adding receivers to existing transmitters, or adding transmitters to existing receivers, or a mixed of those configurations.

IAP has two radar systems working with MMARIA concept: a) Pulsed Radar System and b) SIMONe.

2.1 Pulsed Radar Systems

The MMARIA ([6]) pulsed-configuration has receiving and transmitting stations that are distant from each other. The antennas at these stations were circularly polarized to avoid any blind spots. Since, both the stations are distant from each other; the stations need to be synchronized. To achieve this Rubidium crystals

disciplined with GPS were used. This setup acts as a master clock having a frequency of 10MHz.

An experiment was conducted to check the feasibility of the of the new concept using standard SMRs called SKiYMET ([1]) and a receiver only station at the Kühlungsborn. The experiment used different frequencies, as well as different pulse repetition frequencies (PRF) to optimize ambiguous detections at both the stations ([6]) . Moreover, due to the bistatic geometry, the velocity is measured along the Bragg wavelength and the Bragg wavelength for a bistatic geometry is higher than that of a monostatic configuration. In this particular experiment, it was also observed that the altitudinal coverage also increased in MMARIA and hence the overall detections increased by around 70%.

2.2 Spread Spectrum Interferometric Multistate Meteor Radar Observing Network

SIMONe ([4]) is a special implementation of MMARIA, which is much cheaper and robust when compared to the initial MMARIA concept. Coded continuous waves, multiple input multiple output and compressed sensing are used in this implementation.

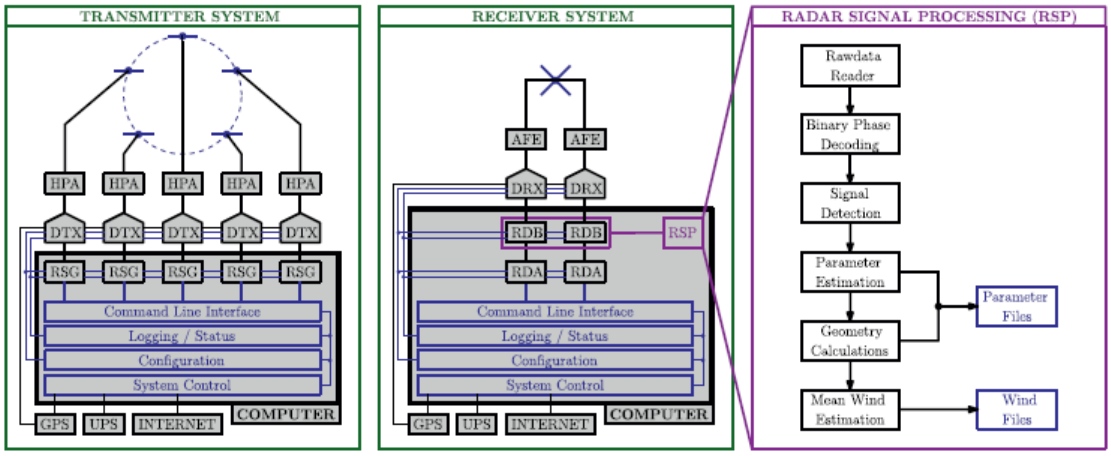

Figure 2: SIMONe Architecture ([4])

At the transmitting side ([4]), multiple antennas are used in an interferometric configuration and the configuration has a minimum of five antennas, each of them

fed by an independent transmitter. Coded continuous waves are used in every transmitter and the code used are pseudorandom binary codes. These seeds of the random number need to be chosen carefully to avoid interstation interference as well as to minimize the cross correlation between them.

At the receiving end ([4]), one or more antennas are used in interferometer configuration, based on which implementation of MISO (Multiple Input Single Output), SIMO (Single Input Multiple Output) or MIMO (Multiple Input Multiple Output) are possible. When a single antenna is used, MISO configuration can be used and can be used for the estimation of angle of departure (AOD). When multiple antennas are used, either a SIMO or MIMO is possible, with SIMO used for the angle of arrival estimation (AOA) or MIMO can be used for the estimation of both AOA and AOD simultaneously.

Figure 2 shows the major components of the SIMONe system ([4]), the transmitter, receiver and the radar signal processing. On the transmission side, a 450W CW power amplifier (HPA) is used for each antenna and a digital transmitting unit (DTX) creates the driver signal for the HPA as a low power phase modulated CW signal by a software-defined radio (SDR). A radar signal generator (RSG) inside the transmitter computer controls the DTX with the modulating signal information, which receives instructions such as waveform, code, baud rate, period, amplitude and phase from the user. The main computer is supplied with an uninterrupted power supply and is remotely connected to the internet. The multistatic synchronization and the timing information for the computer are provided by a GPS receiver unit, which provides a 10MHz reference clock and one pulse per unit (PPS) signal to the DTX.

On reception ([4]), an analog front end (AFE) filters and amplifies the signal received from the antennae. These signals are fed into a digital receiver (DRX), which is an USRP-N200 with a basic receiver daughter board. With the help of the DRX the received signals are down sampled and separated into phase and magnitude components and decimated. This received data is then stored into the raw data buffer (RDB) and the raw data archive (RDA). Then the received complex signal are coherently referenced to the UNIX time standard using the MIT Haystack Observatory's Digital RF coherent RF data package

(https://github.com/MITHaystack/digital_rf). The data is then stored in Hierarchical Data Format version 5 (HDF5). For real time processing and monitoring the data is stored in RDB in ring buffer configuration for 1hr. Similarly, data is stored for 14 days in RDA for offline routine events or any other special events. Similar to the transmitter, the receiver is also connected to the internet and a GPS receiver unit is used to provide a 10MHz clock signal.

The receiver and the transmitter ([4]) computer have an Ubuntu distribution and has a flexible internet connection. Python computing language is used in the RSP modules that are run on the receiving computer. The signals are the decoded using compressed sensing approach; combinations of matched filters, inverse filters and least square fitting approach are used for the decoding. Then an incoherent combination of the decoded signals is done for storing them. The complex signals are combined coherently in order to detect the echoes. After this, Doppler shift, correlation time and amplitudes are estimated using auto and cross correlation, and the interferometric phases using the fitting approach. The echoes are selected for further processing only if it is a good fit. The parameters of the identified events are sent to the central server via internet. Means winds are estimated, visualized and stored on site for monitoring and quality and control purposes.

3 Non – Gaussian Distribution in SIMONe

As mentioned previously, applications of radars include detection of specular echoes for MLT wind calculations, detections of airplanes and other low – latitude geophysical echoes. This chapter primarily focuses on the calculation of mean winds and the residuals as well as the non- Gaussian characteristics observed.

3.1 Mean Wind Calculations

The phase of the voltage at a receiving station ([4]) due to the meteor echo detected by a transmitter p is given by:

$$ø_{mp} = - (k_i - k_u).ut – k_i.R_p – k_s.R_m \quad (3.1)$$

where R_p is the vector pointing towards the meteor from transmitter p and R_m the vector towards the meteor from the receiver m; $k_i = kR_p / |R_p|$ and $k_s = kR_m / |R_m|$ the incident wave vectors and $k = 2\pi / \lambda$, λ is the radar wavelength and $u = u[u,v,w]$, the velocity with which the meteor tail drifts. The u, v, w components represents the zonal, meridional and the vertical winds respectively and is assumed to be positive in the direction of east, north and up.

Then, the voltages due to the transmitters p and q at the receiver m is cross-correlated including the time lag, that is $V_{mp}(t+\tau) * V_{mq}(t)$. The phase of this calculation is given by the equation ([4]):

$$ø_{mp}(\tau) = -(k_i - k_s).u\tau – k_i.R_p – k_s.R_m + k_i.R_q + k_s.R_m$$

$$= k_B.u\tau + k_i.\Delta r_{pq} \quad (3.2)$$

where k_B is the Bragg vector, Δr_{pq} is the vector positioning the antenna to a common reference.

From equation 3.2, it can be inferred that the value k_i can be obtained by cross – correlating the voltages at a time lag of zero. The combination of beamforming and least square fitting technique can be used for estimating the overall solution.

Further information on this can be found in ([7]). Moreover, the Doppler information f_d, can be calculated by performing auto-correlation at different time lags.

In general, the mean winds are calculated by the binning time and altitude with a certain resolution. The mean winds ([4]) are calculated using the following equation;

$$k_{Bi}.u_0 = 2\pi f_{di} \qquad (3.3)$$

where k_{Bi} and fd_i represents the Bragg vector and the Doppler frequency for a particular detection. The solution is then found out using doubly iterated weighted least square fitting approach. In the first fitting, Doppler uncertainties are used as weights while with second fitting detections with differences more than three times the standard deviations are not considered. In this thesis, these neglected detections with values greater than three times the standard deviation are analyzed to identify detections with geophysical meaning.

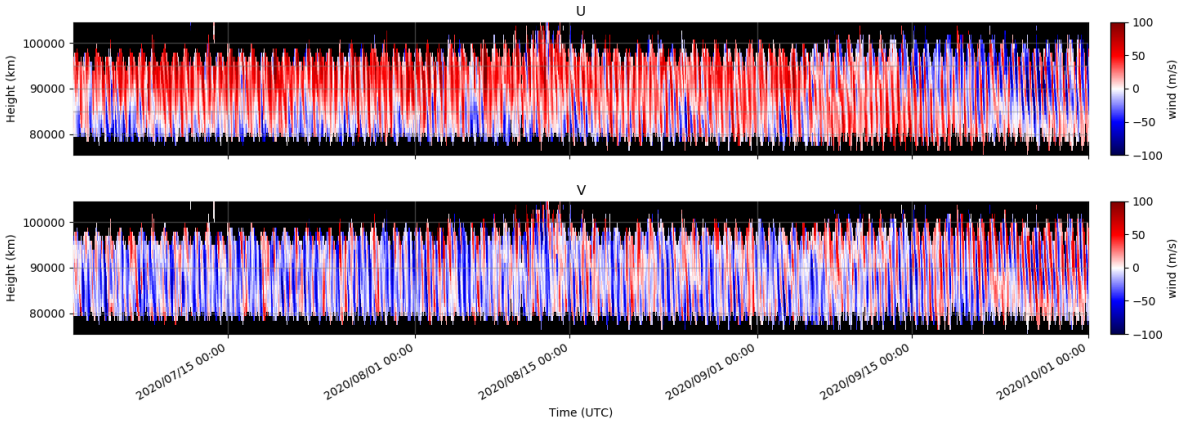

Figure 3: Mean wind for three months

Source: Provided by M. Clahsen on request

Figure 3, depicts the calculated mean winds for the duration of the three months analyzed in this thesis. The first and the second represents the u and v components of the winds respectively.

3.2 Residual Distribution

For the further analyses, residual values are calculated using the detected mean winds and the calculated Doppler. The detection of SIMONe are generally recorded in three files in HDF5 format, namely the detection file, the parameter file and the wind file. For the calculation of residuals, the required variables could be obtained from these files.

The first step is to calculate the Doppler velocities (f_{di}'), using the pre-calculated the Bragg vector and the mean winds, which is calculated as follows:

$$k_{Bi}.u_0 = 2\pi f_{di}'$$

$$f_{di}' = k_{Bi}.u_0 / 2\pi \quad (3.4)$$

Then, the detected dopplers are then subtracted from the calculated once to obtain the residuals.

$$\text{Residual, } \Delta f_{di} = f_{di} - f_{di}' \quad (3.5)$$

The flowchart below will help understand the algorithm used for this calculation better.

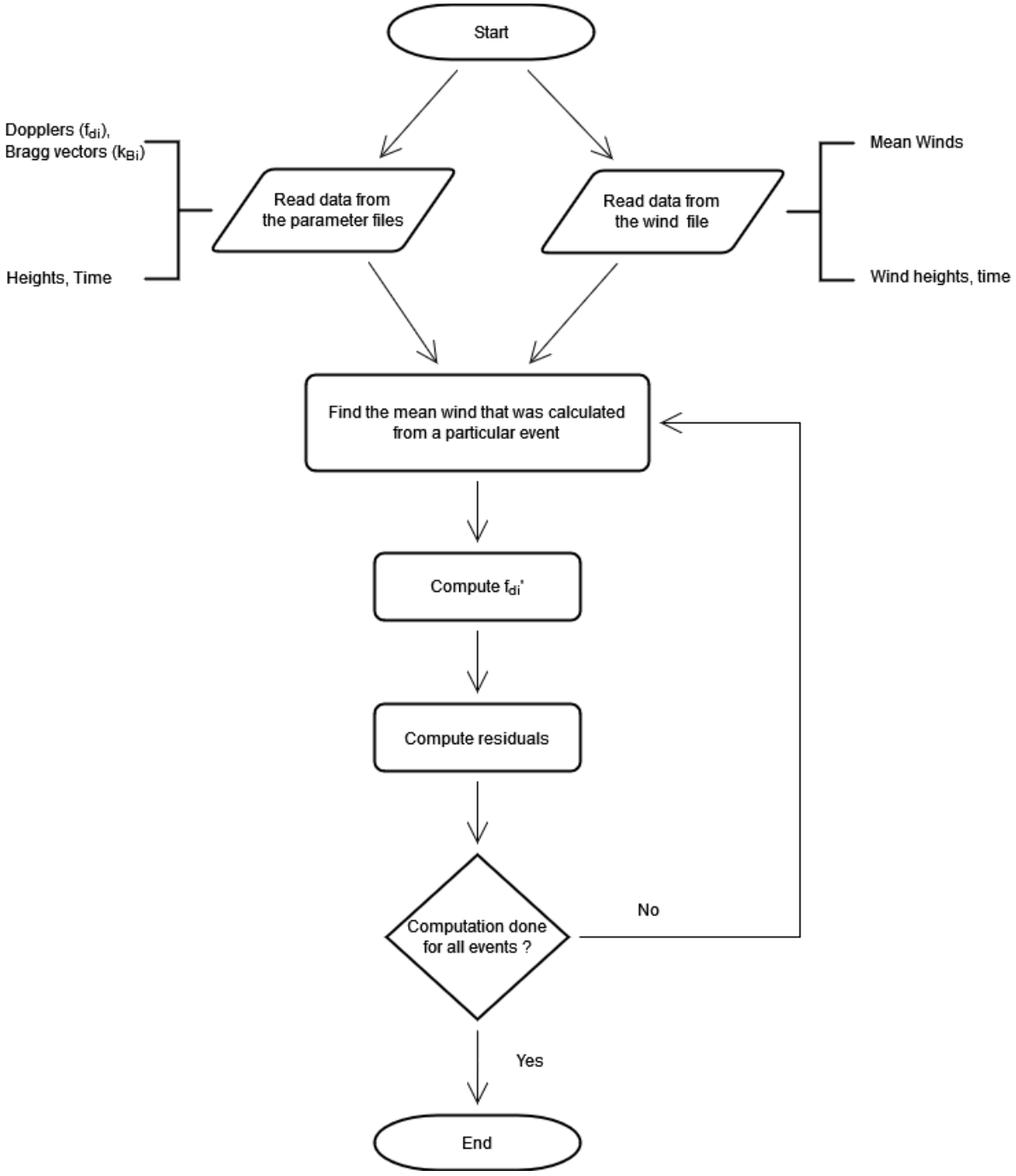

Figure 4: Flowchart for Residual Calculation

As mentioned, the detections are saved into three HDF5 files, the parameter files contains information such the height, Doppler, latitude, longitude, time etc. and the wind file contains information such as the wind heights, wind hours and the mean winds. Mean winds are saved in a three dimensional matrix where the dimensions are based on height, time and the wind components. The initial step is to identify the wind, which was calculated from a particular event. This is

achieved by acquiring the wind, closest to the height of the detected event at a particular time.

Once the wind is identified, the Doppler (f_{di}') is calculated using equation 3.4. In some cases, due to poor detections winds could not be calculated and is stored as not a number (NaN) and thus the corresponding calculated Doppler is also saved as a NaN. Eventually, the residuals are then calculated using equation 3.5.

These residuals are calculated for a month, the residual values along with other parameters including height and direction cosine (dcos) are stored into an HDF5 file (Parameters_<month>_2020.h5) to ease the further analysis and also to reduce the computation time.

3.3 Kurtosis Analysis

In probability theory and statistics, kurtosis ([14]) is a measure of outliers in a given distribution; hence, it essentially represents the shape of a distribution. The concept of kurtosis, developed by Karl Pearson is the fourth order moment of a distribution; where moments are the statistical parameters of a distribution. The first order moment represents the mean, the second order moment represents the variance, the third moment represents the skewness and the fourth moment the kurtosis.

A distribution with kurtosis value of 3, is defined as a normal distribution or a Gaussian distribution ([16]) and a distribution with higher kurtosis value corresponds to non-Gaussian distributions.

Kurtosis ([14]) of a distribution is defined as:

$$\mathrm{Kurt}[X] = \mathrm{E}\left[\left(\frac{x-\mu}{\sigma}\right)^4\right] = \frac{\mathrm{E}[(X-\mu)^4]}{(\mathrm{E}[(X-\mu)^2)^2]} = \frac{\mu_4}{\sigma^4} \qquad (3.6)$$

where μ_4 is the fourth central moment and σ is the standard deviation.

Techniques for finding kurtosis is provided by many software packages including Matlab, IDL and python. In this case, the analysis is done using the function provided by the scipy library ([15]) in python. The function can be used to

calculate kurtosis using Fisher or Pearson definition. If Pearson definition is used, 3 represents a normal distribution and in case of Fisher definition 0 represents normal distribution, which is also known as the excess kurtosis.

Excess kurtosis = kurtosis – 3 (3.7)

For the analysis in this work, Pearson's definition of kurtosis is used and is much greater than one and also this value varies with the applied signal to noise ratio (SNR) threshold. This confirms the non – Gaussian nature of the data, analysis of which will be explained further in the following chapter.

4 Analysis of Three Sigma Values

The focus of this work is to analyze the events with the residual values greater than 3σ. The analysis is performed on 3 months of data obtained from a SIMONe Germany radar system.

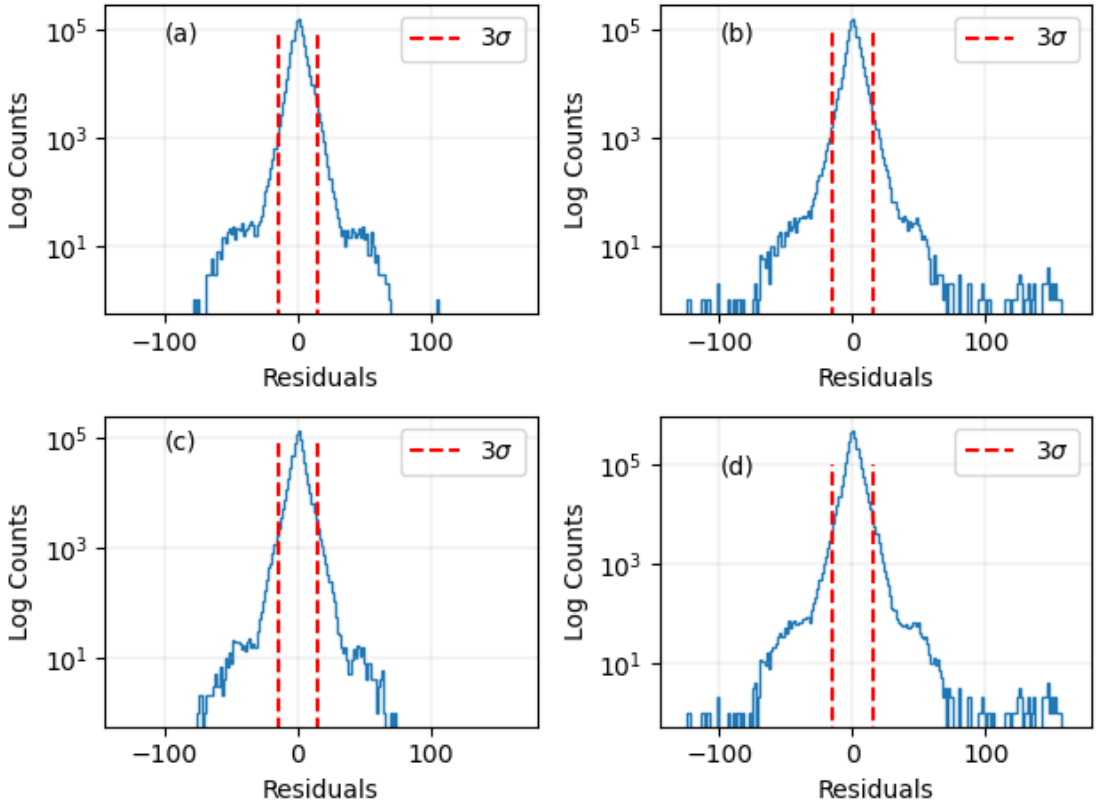

Figure 5: Residual Distribution

Figure 5 shows the residual distribution of (a) July, (b) August, (c) September and (d) overall residual distribution for all three months. The red line represents the three times standard deviation. The events outside the red line are considered as outliers, which are further analyzed.

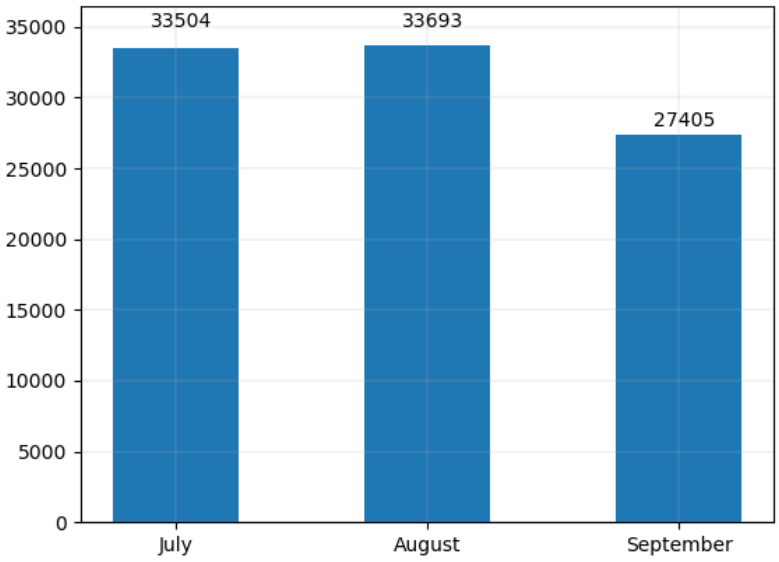

Figure 6: Average Monthly Detections

Figure 6 shows the monthly average detection for July, August and September. The count was obtained by averaging the daily counts of a particular month.

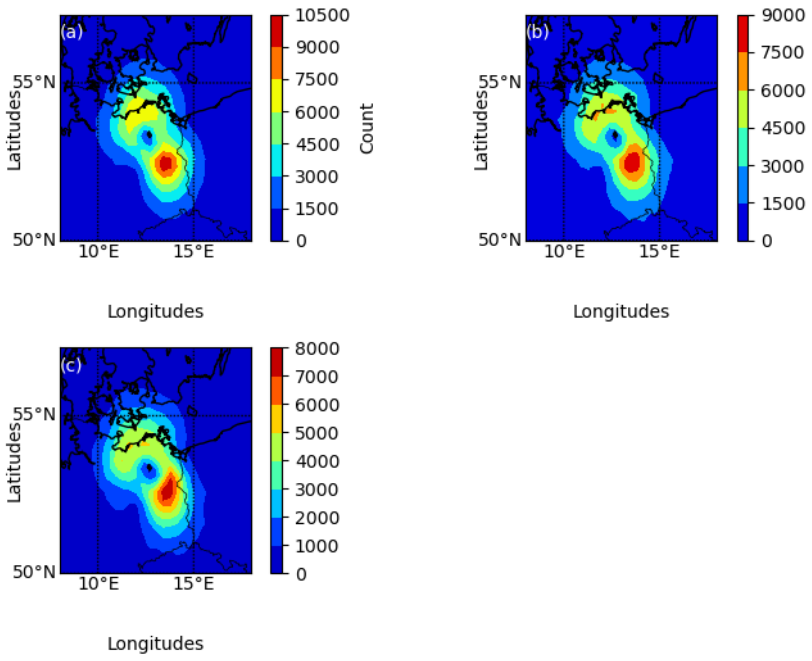

Figure 7: 2D Histogram of Detections on Latitude vs Longitude Axes

Figure 7 depicts the number detections at different latitude and longitude values for the months of (a) July, (b) August and (c) September. It can be observed that

the detections are more at the receiver (53.33°N, 13.071°E) than at the transmitter location (54.119° N, 11.77° E).

The overview of the analysis is depicted in the below.

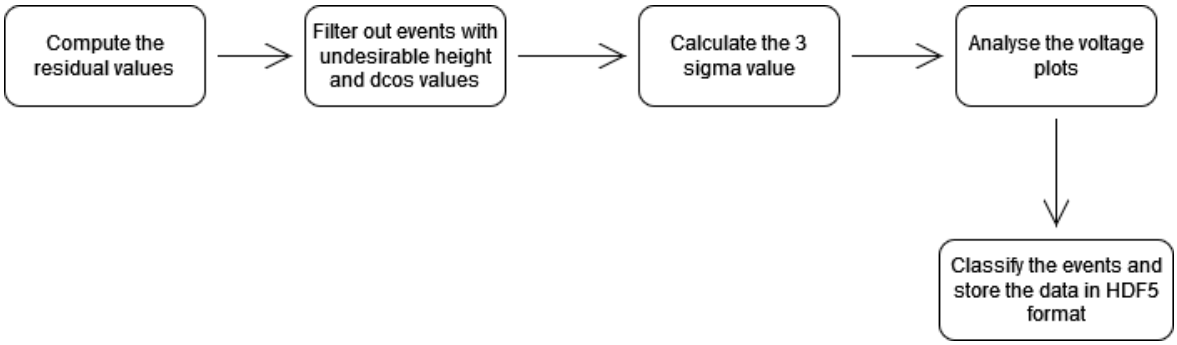

Figure 8: Overview of the Analysis

Initially, the residuals are computed the detailed steps for this is explained in the previous chapter. An additional filter based on their height and direction cosine values is done. Once this filtering, is done the remaining events are the ones that more likely to be an actual meteor. Once, we have the whole new distribution the standard deviation of the distribution is calculated and the events with the values greater than 3 times the standard deviation are identified for further analysis, which in the routine process are cleaned as outliers. Based on the voltage plot, these events are classified as geophysical and non – geophysical.

4.1 Classification Criteria

The outliers are analyzed based on their height, direction cosine and their voltage profile, each of them will be explained separately in this section.

4.1.1 Height

Meteors ([5]) enter the atmosphere at a very high velocity. When they enter the denser parts of the atmosphere, heating occurs rapidly. The heat development becomes so high at altitudes of approximately 70 – 120 km resulting in the ionization of the material, which are then later removed by friction and the heat of

the meteor as well as the surrounding air. Due to ionization a plasma trail is produced in the meteor flight path, which contains metal ions, gas ions as well as electrons. The plasma formed produces a tail, and the tail moves with the speed of the background atmosphere and thus can be used to study the atmosphere movements such as winds.

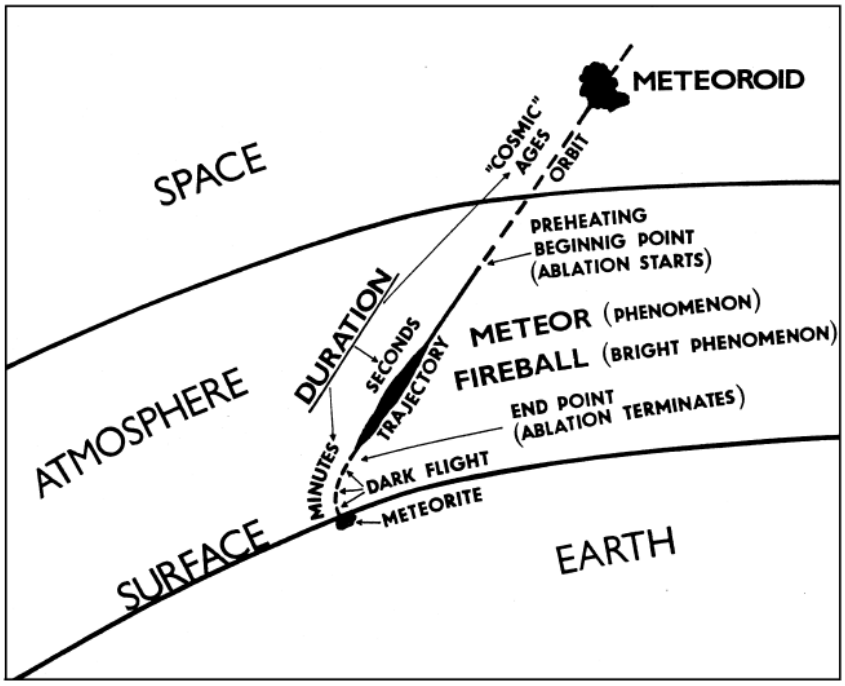

Figure 9: Meteor Interacting with the Atmosphere ([5])

The height limits for the classification was set to be approx. 70 km to 125 km, keeping a tolerance of 5km.

4.1.2 Direction Cosines

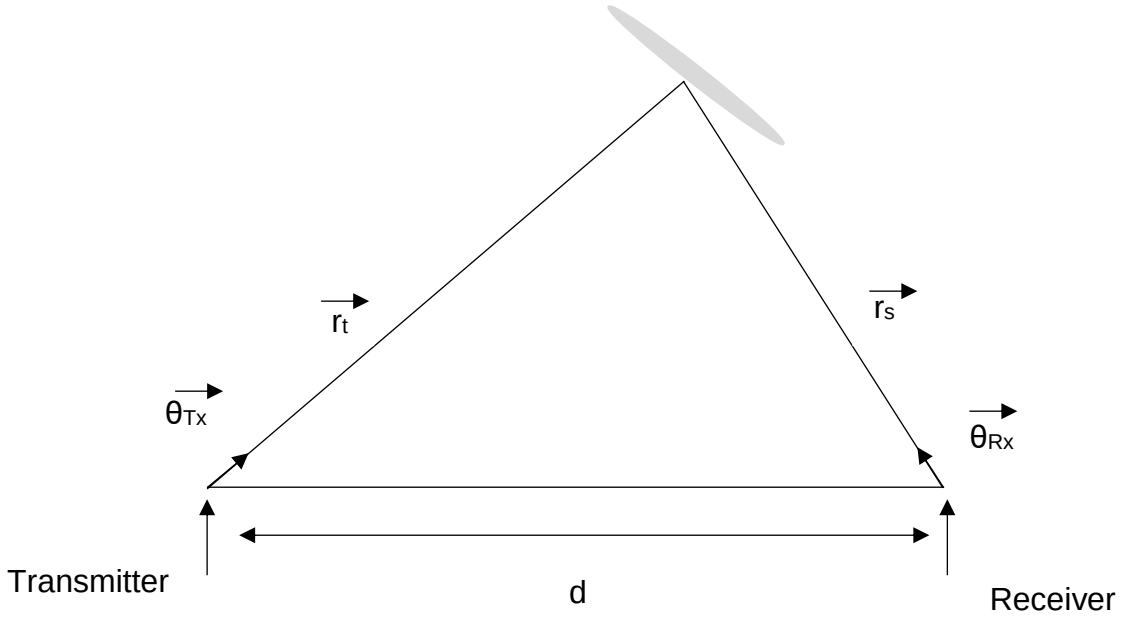

Figure 10: Illustration of Bistatic Position Determination

Here $\vec{\theta}$ is the vector containing directional cosines ([9]) pointing towards the meteor. This can be stated as:

$$\theta = \begin{bmatrix} \theta_x \\ \theta_y \\ \theta_z \end{bmatrix} \tag{4.1}$$

$$\theta_z = \sqrt{1 - \theta_x^2 - \theta_y^2} \tag{4.2}$$

The 2D illustration can be depicted as follows:

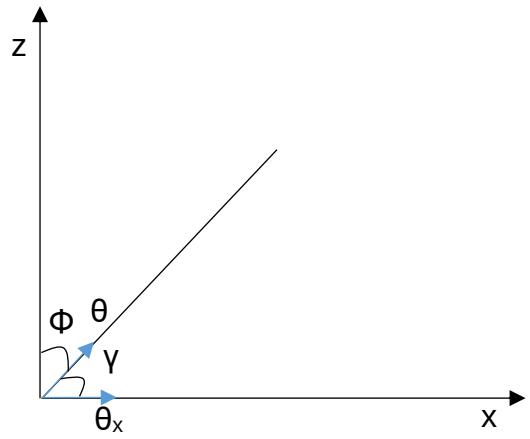

Figure 11: Geometry of the Angle-of-Arrival Determination

In the above illustration, Φ represents the angle from the zenith and γ the elevation angle.

Thus,

$$\Phi = \sin^{-1}\sqrt{\theta_x^2 + \theta_y^2} \tag{4.3}$$

That implies,

$$0 \leq \sqrt{\theta_x^2 + \theta_y^2} \leq 1 \tag{4.4}$$

Moreover, a previous research ([2]) suggest that when a zenith angle greater than 65° increases the height ambiguity. Hence, the events with |dcos| greater than 0.9 automatically are automatically rejected by the program. Thus, events with elevation angle less than 25° are not considered.

4.1.3 Voltage Profile

When the radar receives a radar echo, it is always Doppler shifted due to the projected background wind. A phase shift could also be introduced due to the position of the receiver with respect to the reference. Thus, the baseband model ([9]) of the signal is given by:

$$x(t) = A.\, e^{-\propto t}.\, e^{\varphi_0}.\, e^{2\pi f_d t}.\, e^{\varphi_c} \quad (4.5)$$

Here, A is the maximum amplitude of the signal, f_d the Doppler frequency, α the decay rate, φ_0 the zero phase of the impinging signal at the reference point and φ_c is the constant phase offset of the signal of one sensor depended of position of the sensor and the angle of arrival relative to the reference point of the receiver.

In this thesis only the effect of the Doppler frequency and the decay rate are considered.

Thus, the equation 4.5 can be simplified as:

$$x(t) = A.\, e^{-\propto t}.\, e^{\omega t} \quad (4.6)$$

where $\omega = 2\pi f_d$

From the equation, it can be understood that for a good meteor event it would be possible to observe an exponentially decreasing velocity profile. The voltage plot of all the events that meets the criteria mentioned in the previous sections are manually checked identify good events.

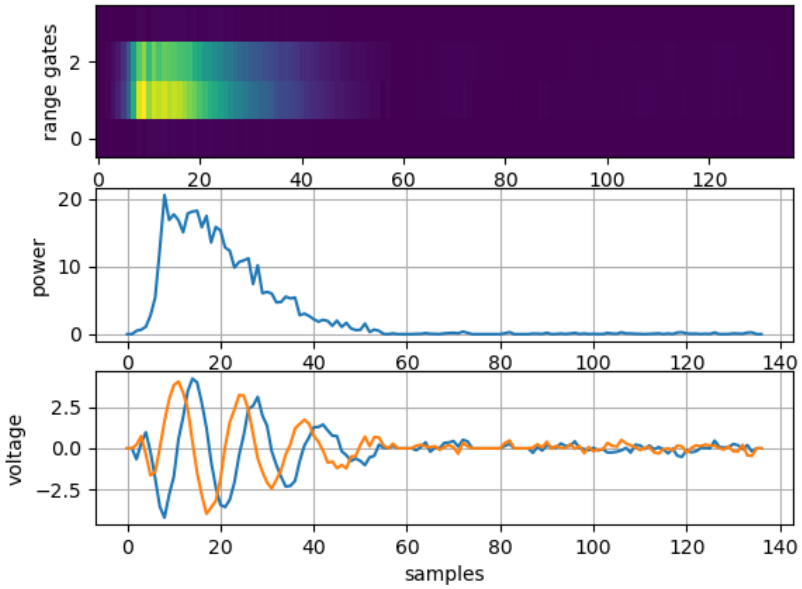

Figure 12: Example for a Good Meteor Detection

Figure 12, gives an example for a good under dense meteor echo. The voltage plot shows a clear exponentially decreasing signal with a velocity profile.

Similarly, a power plot that decreases exponentially can also be observed.

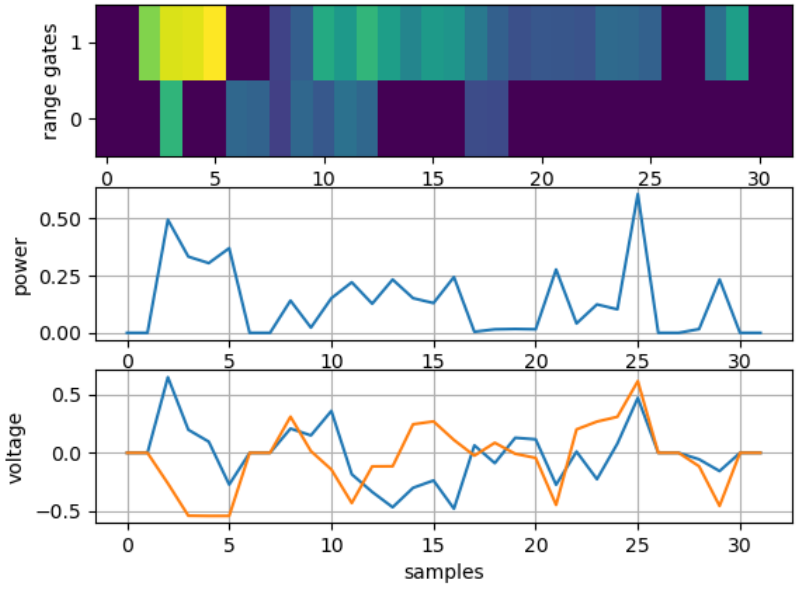

Figure 13: Example for a Non – Meteor Detection

An example for a non – geophysical detection can be observed in Figure 13. The plot shows no velocity profile.

4.2 Graphical User Interface (GUI)

For the analysis, a graphical user interface was developed in Python. The standard package tkinter provides a GUI toolkit. The GUI was developed to ease the manual classification process and to verify or to show the events within certain constrains.

While working with tkinter it is necessary to create a new window, and the appearance of this window depends on the operating system (OS) in which code is run. For this work, the code was done on a Windows OS.

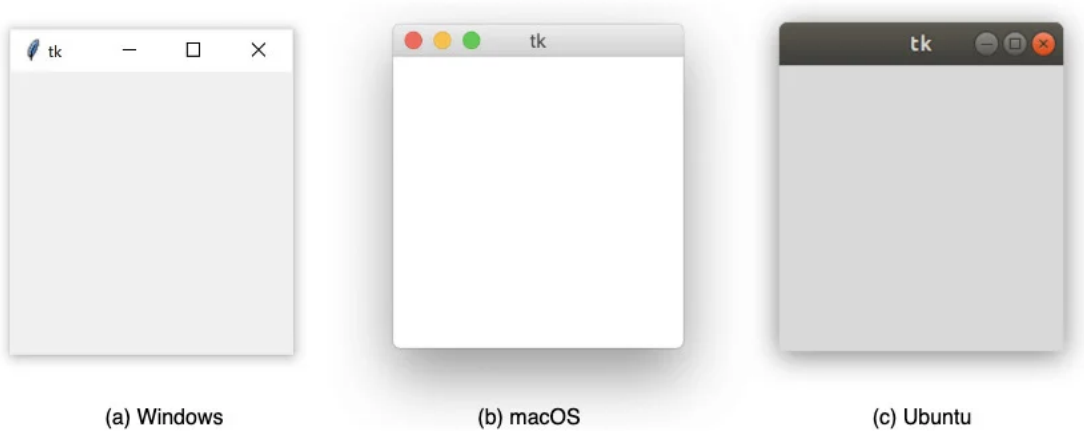

Figure 14: Appearance of the Tkinter GUI in Different OS [13]

The tkinter package provides many widgets such as labels, combo boxes, radio buttons and many more. It is essential to keep in mind to run the code for these widgets within the .mainloop() command as this commands the python to run the tkinter event loop. This makes the program listen to the external inputs such as the keypresses, button clicks and so on.

4.2.1 Tkinter Widgets

Below are the working of the tkinter functions used for the development of the GUI implemented in this project:

1. **Frame** ([10]): Frame is a mostly used for arranging the other widgets. It helps in positioning of other widgets.
   Syntax for this widget is:
   w = Frame (master, option, …)

   Master represents the parent window and options are mainly the various formatting options for the frame including the size, background color etc.

2. **Radiobutton** ([10]): This widget provides a multiple choice while at the same time lets the user to choose only one option from a pre-defined list of options. To realize the functionality of the radiobuttons, this has to be linked to a single variable and each option must have a unique value.

Syntax for radiobutton is as follows:
r = Radiobutton (master, option, …)

Here, master represents the frame or the window insid which the button has to created and options are the various formatting options.

3. **Label** ([10]): The widget is used for placing text or an image on to the parent window. The label can also be updated continuously with a program.

   Syntax of a label is as follows:
   l = Label (master, option, …)

   Similar to a radiobutton, the master represents the parent window and the options are the formatting options as well as the procedure that has to be called each time user chooses an option.

4. **Button** ([10]): This widget is used for displaying a button in the GUI. Usually, a function or a method that runs automatically on each button click is attached to button. A small text could also be displayed on the button as an indication for its function.

   Syntax of a button is as follows:
   b = Button (master, text, command, option, …)

   The master and the option is similar to that of the previous widgets. Text is the one that has to be displayed on the button and the command represents the function attached to the button.

5. **Combobox** ([10]): Combobox is designed to allow the user to select a single value or option from a pre-defined list. It shows or takes only one value out of the inbuilt list. It is also possible to set a default value if in case the user does not choose an option. This widget is available in ttk module

of the tkinter library and is more modern.

Syntax for combobox is as follows:
c = Combobox (master, option, …)

The master and the option is similar to that of the other widgets. However, the command for binding or attaching a function is different from other widgets. The bind() function is used to attach the combobox to the call back function. The syntax for this is given below:

c.bind ("<<ComboboxSelected>>", callbackfunc)

Here, callbackfunc is the name of the pre-defined function that has to be executed automatically when the user makes any change in application.

6. **Notebook** ([11]): Notebook widget allows you to navigate through different pages while clicking on tabs. When we click on a tab, essentially a child pane is displayed, which is mostly a frame.

   The syntax for creating a notebook using the ttk.Notebook class is as follows:
   n = ttk.Notebook (container, options, …)

   Container is usually the root window and options are formatting options such as height, width.

   This class has many methods to manage the tabs efficiently, one of them is the add() method. The method can be used as follows:

   n.add (child, **kwargs)

   The child is a widget that has to be added to the notebook, which is in common language is the tab and kwargs include the options to add text

and image to the tabs.

4.2.2 Embedding Matplotlib Graphs in the GUI

The matplotlib library provides additional functionality to be embedded into a tkinter application. To activate this functionality ([12]), we need to import FigureCanvasTkAgg, NavigationToolbar2Tk and Figure modules from matplotlib.backends.backend_tkagg.

The first step is to create a container to hold all the plots using Figure module mentioning their size. Next, step is to create a matplotlib graph using FigureCanvasTkAgg. This creates the axes for the graph.

The syntax is as follows:

c = FigureCanvasTkAgg (master, f)

where master is the window or the frame and f represents the container for the plots created using matplotlib.figure.

Once a graph is created, it is necessary to add the traditional matplotlib toolbar to the GUI for convenience. This can be done using NavigationToolbar2Tk module.

Syntax of NavigationToolbar2Tk is:

t = NavigationToolbar2Tk(c, frame)

Here c is the previously created matplotlib window and frame represents the frame in which the toolbar has to be attached.

4.2.3 GUI for Outlier Verification

It was intended that with the developed GUI it should be possible to classify the events as well as to cross verify the events based on their height, time and dcos. Hence, both this functionalities are implemented in the same GUI in different tabs.

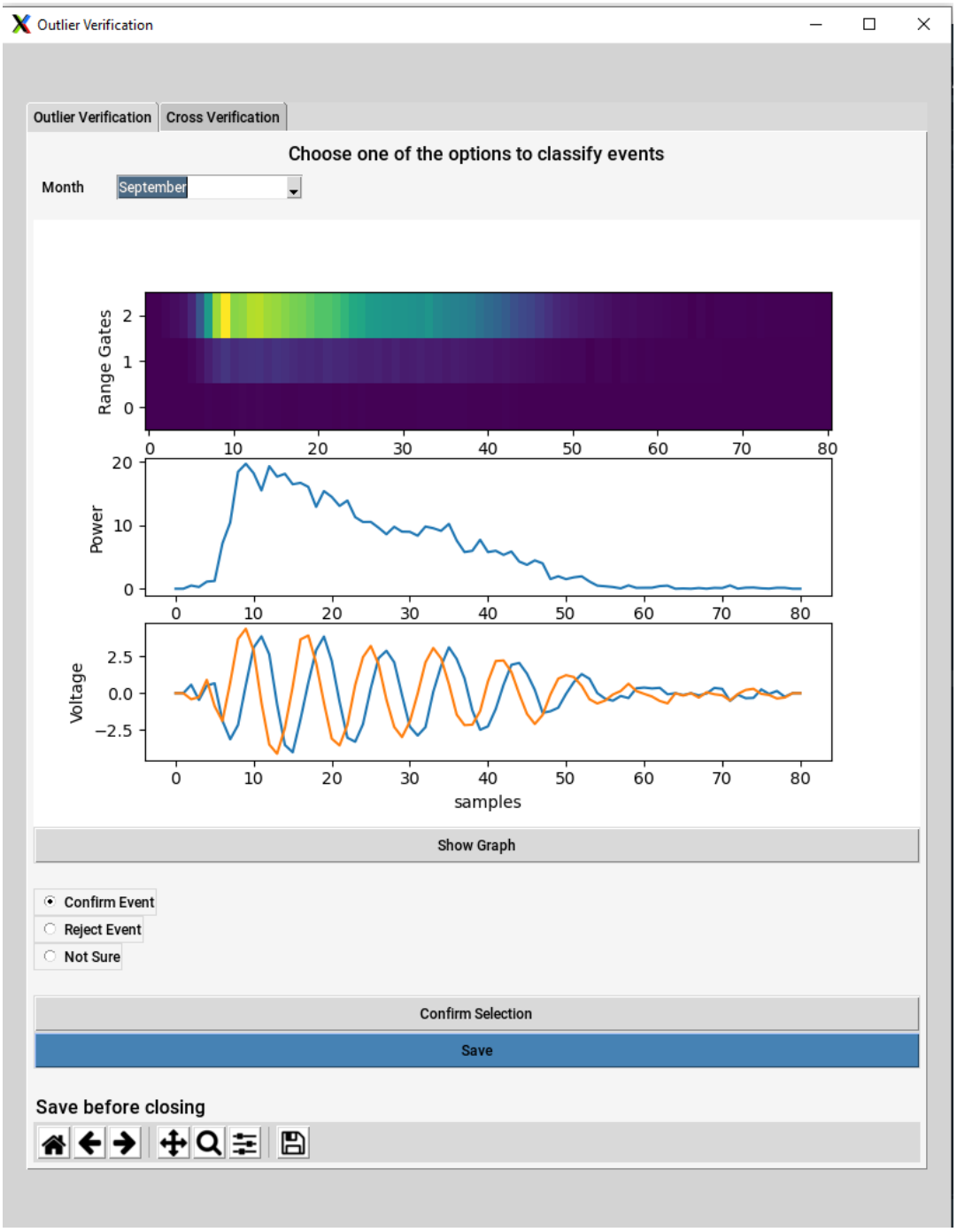

Figure 15: Developed GUI for the Classification of Events

Figure 15 depicts the developed GUI. The first tab or the first child pane of the

GUI is used for the classification of the events. This is used for accepting or rejecting the events by looking into their voltage plots. The function and usage of various widgets are explained below.

1. **Month**: A combobox widget is used to allow the user to select a month, which the user needs to analyze. A pre-defined option for months are given inside the program, from which the user is allowed to choose from. Once the user selects a month, the program tries to access data from the relevant HDF5 file.

2. **Show Graph**: When the show graph button is clicked, the program plots the graph on to the pre-defined canvas. This button calls the function that accesses the data corresponding to an event from the detection file and thereby plotting it into the canvas. The canvas was previously programmed to show three plots as subplots.

3. **Radiobuttons (Confirm event, Reject event, Not sure)**: The radiobutton allows the user to make a choice, by selecting any of given options. The functions bound to these radiobutton raises a flag based on the option chosen. This flag is then used for further processing.

4. **Confirm Selection**: This is an application of the button widget provided by the tkinter library. The bound function checks the value of the flag raised and appends the event id to the corresponding list. Once the appending is done, the function takes the next event id and the access data from the detection file and a new graph is plotted on to the canvas.

5. **Save**: This is also a tkinter button widget. This allows the user to save the work done so far to a HDF5 file. The saved HDF5 file contains the event id of the events that are not yet classified and the event id of the classified events are saved into their corresponding category. To maintain consistency in the file name and the file names of the HDF5 files automatically chosen by the program.

6. **Toolbar**: The GUI also has the navigation toolbar, which has all the functionalities of a matplotlib toolbar.

The overall idea of the programming can be better understood with the help of a flowchart.

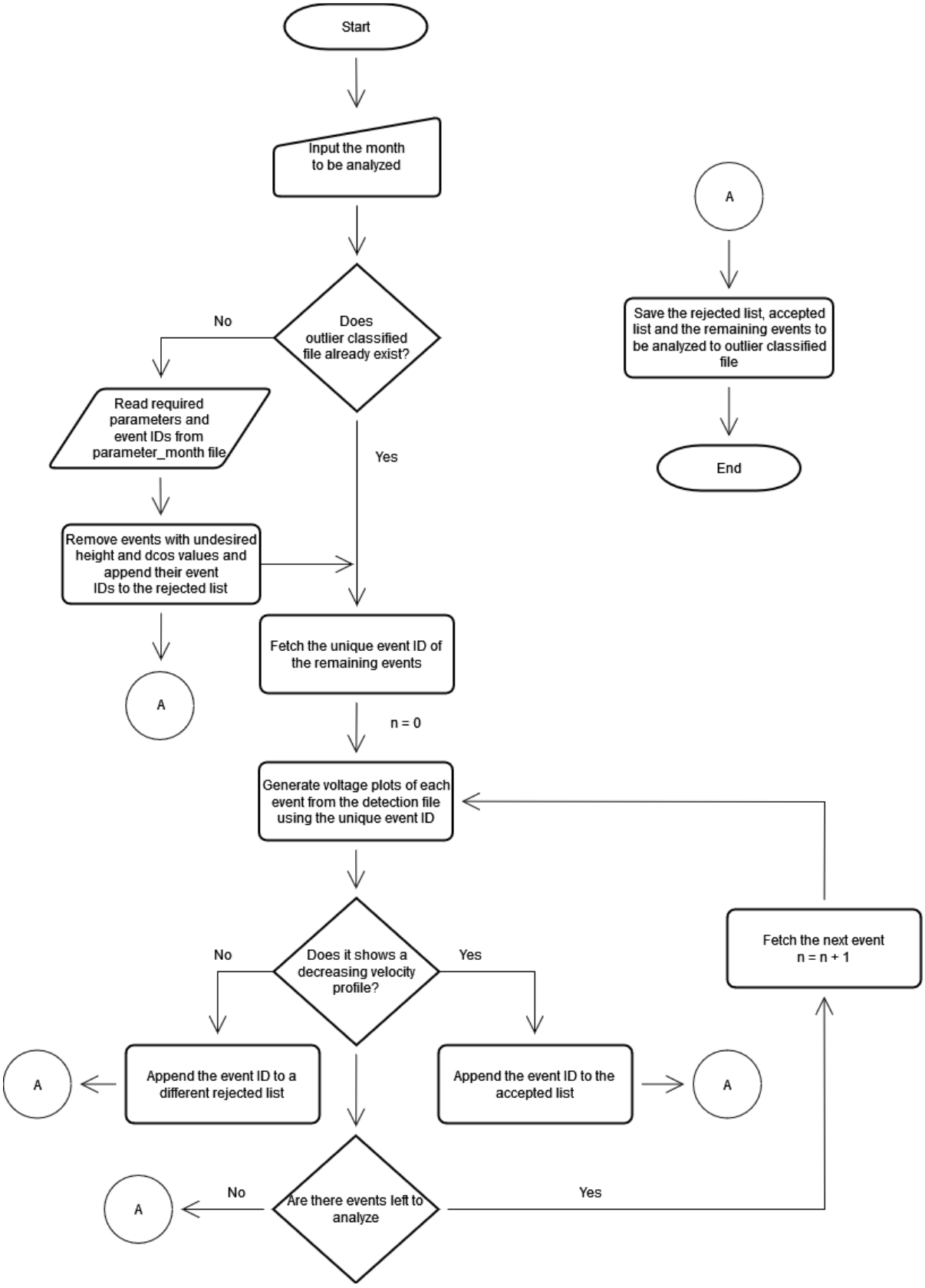

Figure 16: Flowchart Representing the Analysis

The analysis starts when the user chooses the month that needs to be analyzed.

Once the month is chosen, it is necessary to check whether this month was previously analyzed. If it is previously analyzed the program fetches the outlier_classified_<month>_2020.h5 file and then loads corresponding parameters. If the month was not previously analyzed the Parameters_<month>_2020.h5 file is accessed. The program loads the event id along with other parameters, then the outliers with undesired height and dcos values are removed and their event id are appended to an another list, which is then stored into the HDF5 when the user clicks the save button.

As previously mentioned, each event has a unique event id, which is used to fetch their details. When the user presses the *Show Graph* button the program chooses the first event id (n = 0) from the list of events that is yet to be analyzed. The program then tried to identify the parameter file, which the event id is present, since the parameter files are saved on a daily data not as monthly. Once the parameter file is identified, corresponding detection file is accessed to generate voltage plots.

Then, the user can confirm or reject event based on the generated voltage plot. The *Not Sure* option was used to append events that could not be clearly judged. The events in this list was later appended to confirmed or rejected cases after consultation. The radio button just generates a flag, which is later processed when the user presses the *Confirm Selection* button. Once this button is pressed the event id is appended to the corresponding list and also fetches the next event id (n + 1). The loop continues until the whole events are classified or till when the user presses the *Save* button.

When the user clicks the *Save* button, the attached function first converts the event id into bytes208, to save into an HDF5 file. The user need not choose a file name, the file name is already programmed to be as outlier_classified_<month>_2020.h5. The event id are stored into HDF5. The event id are stored into 5 groups; evnt_id_outlr, evnts_accpt, evnts_ns, evnts_rjct_velocity, evnts_ht_dcos which contains the event ids that are not analyzed, accepted events, not sure events, events rejected based on the voltage plot, events rejected based on their height and dcos values respectively. It is always necessary to press the *Save* to button to update the changed into the file.

4.2.4 GUI for Cross Verification

The second child pane of the GUI is used to cross verify or re check the events which were already classified, that is, to look at events that occurred during a specific time of a day, at a particular height etc.

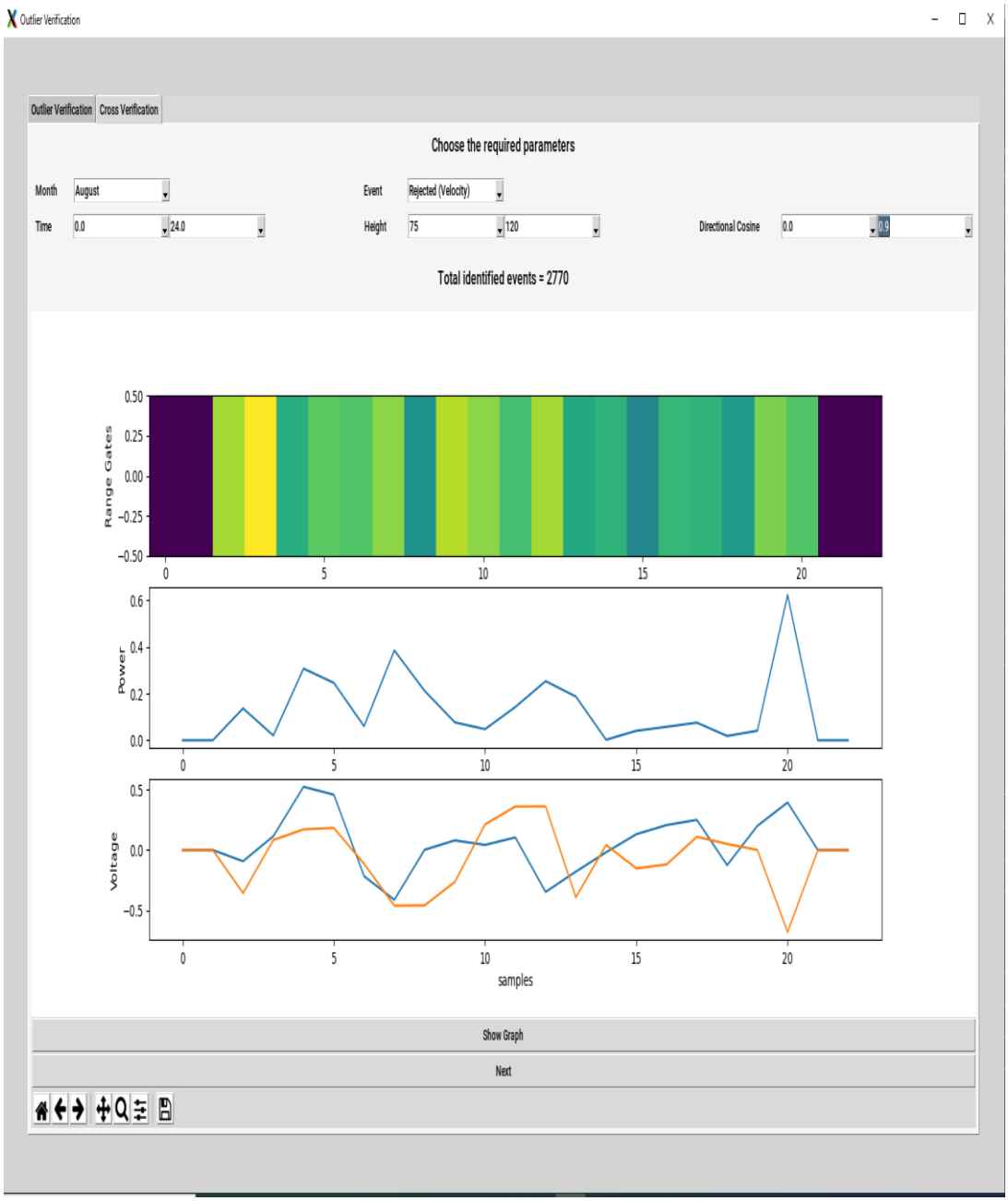

Figure 17: Developed GUI for the Cross Verification of Classified Events

The functions of the widgets are explained below:

1. **Month**: A combobox allows the user to choose the month, in which the events has to be checked. The options for months are pre-defined in the program and similar to the previous case, once a month is chosen; the program tries to access the detection and parameter files of the chosen month from the IAP server.

2. **Event**: This another combobox widget that allows the user to choose the events that are to be displayed. It could be the accepted outliers or the rejected ones.

3. **Time**: This a combination of two comboboxes. The input to these widgets are used identify the time range within which the events are to be displayed.

4. **Height**: This is similar to time. It also allows choosing the events within a particular height range.

5. **Directional Cosine**: These widgets has the functionality similar to that of time and height.

6. **Show Graph**: This tkinter button allows the user to display the graph of events within the given constraints. When the button is pressed, the attached function accesses the data and plots the graph on to the pre-programmed canvas.

7. **Next**: The next button allows the user to navigate through all the identified events.

8. **Toolbar**: This is a matplotlib toolbar, which provides the user with options like zoom and save.

The functioning of this GUI is better explained in the flowchart below:

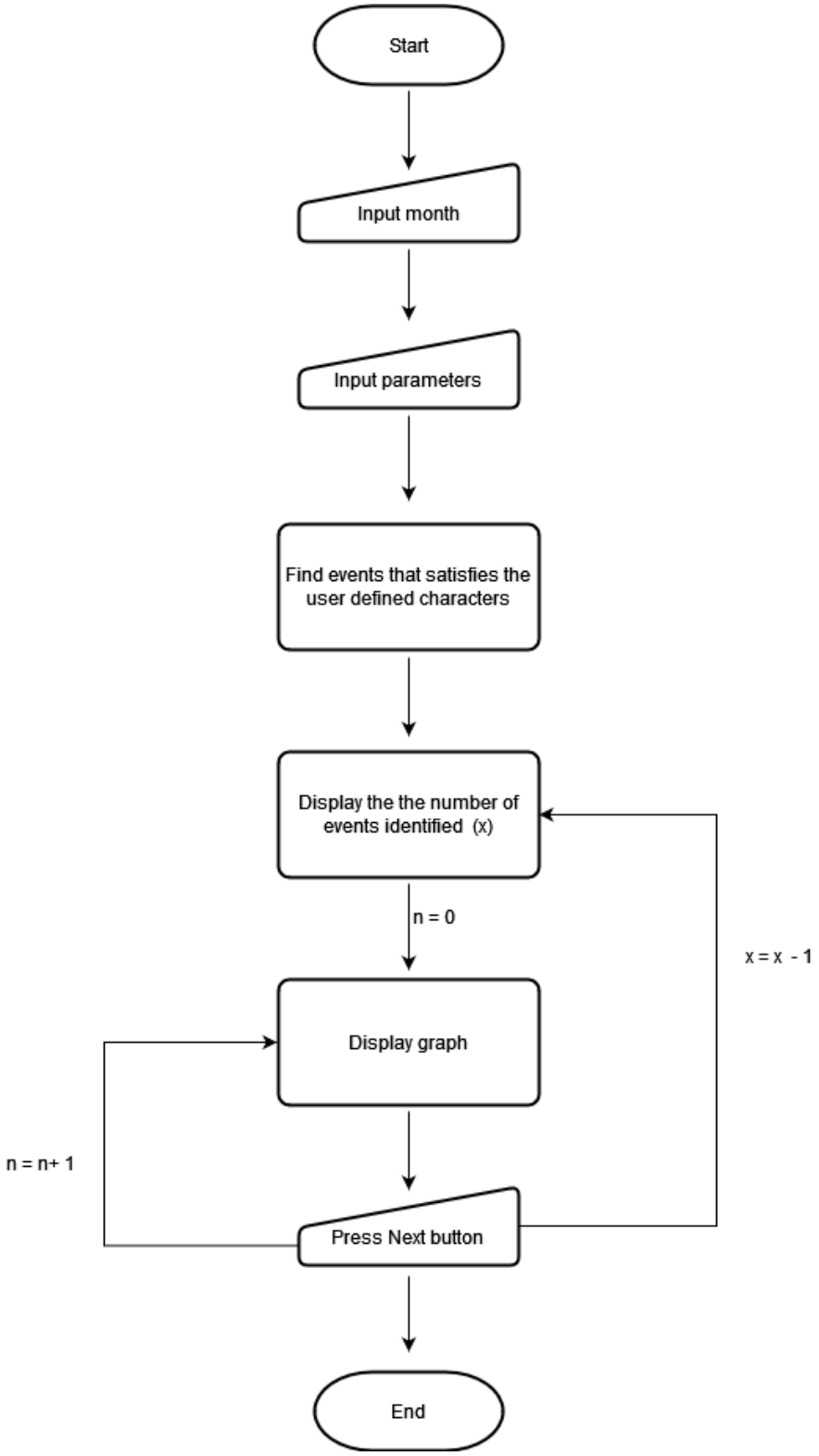

Figure 18: Flowchart of Cross Verification using the GUI

When the user inputs the month to checked, the bound program attached to the

combobox fetches all detection and parameter files from the IAP server, so that it can be read one by one later. Then the GUI takes user inputs on time, height and the dcos values. If the user does not choose a value, the values remain default. The user must always specify whether user wants to see confirmed outliers or rejected outliers in the *Event* combobox.

In order to display the graph, the Show Graph button needs to be pressed. Once this button is pressed, the bound program reads all the previous inputs, and tries to find a list of events that's satisfies all the user inputs. This is essentially done with the help of their unique event ids. The program looks for the intersection of the event ids in different arrays and a list of feasible event id are generated, the graph of the first event will be displayed, and along with a label, that shows the total number of identified events that satisfies all the user-defined parameters.

The *Next* button allows the user to navigate to the other events, by fetching the other event ids in the list. The navigation toolbar provides the user all the functionalities of the matplotlib toolbar.

5 Results

After completing the analysis, the following plots were generated to further understand the outcome of the analysis.

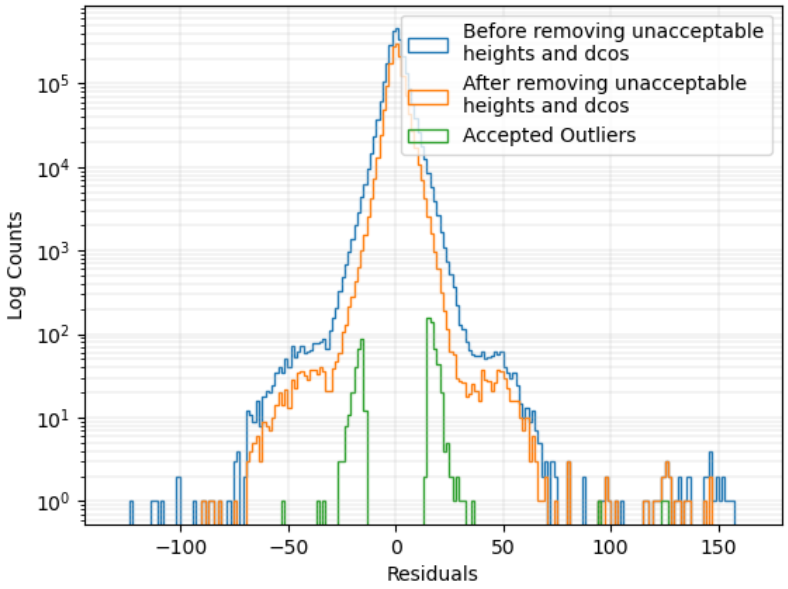

Figure 19: Overall Residual Distribution

Figure 19, illustrates the overall distribution of outliers for a period of 3 months. The outlier values that could be real are also plotted in the figure.

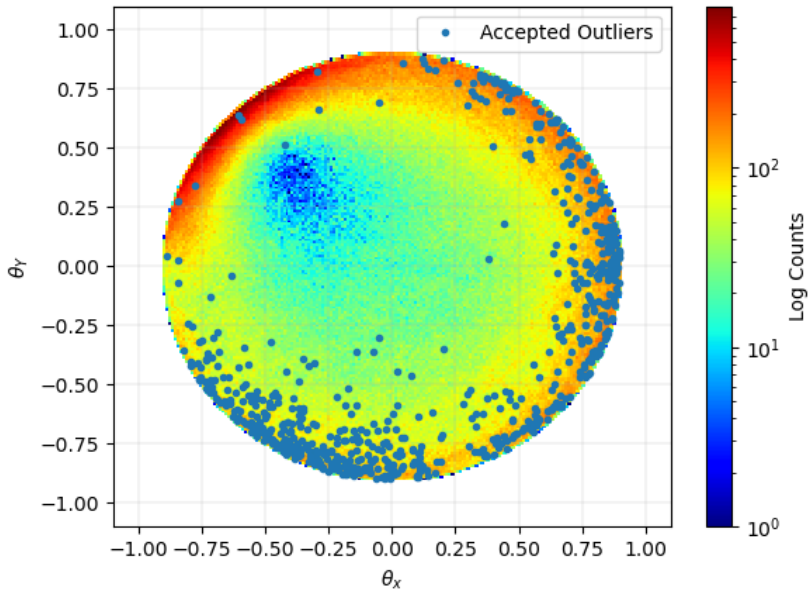

Figure 20: Overall Dcos Distribution

Figure 20 portrays the overall angle of arrival distribution with respect to the receiver after the analysis. The blue dots represents the accepted outliers that are real.

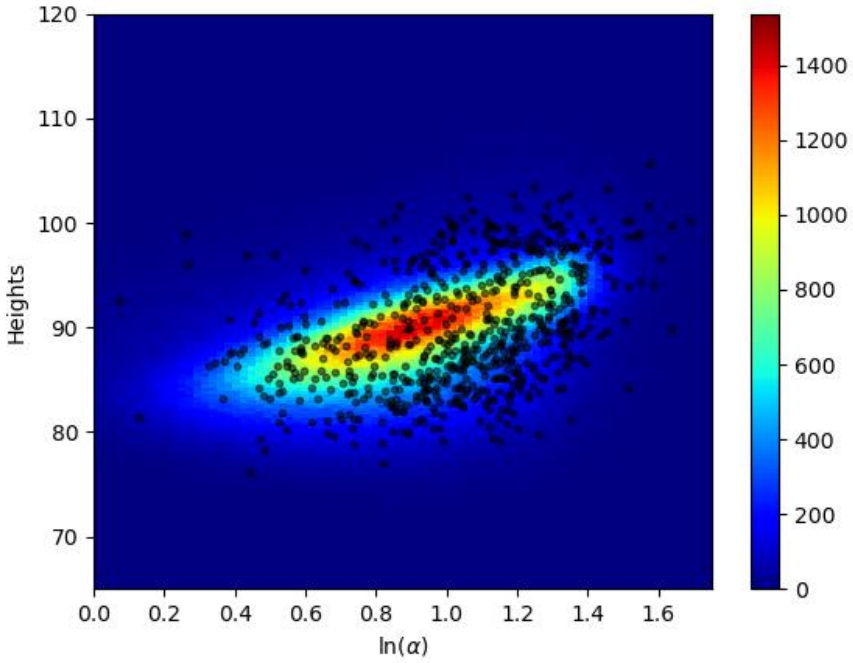

Figure 21: 2D Histogram for Decay Rate vs Heights

Figure 21 depicts a 2D histogram of height vs the decay rate. The black dots represents the good geophysical outliers.

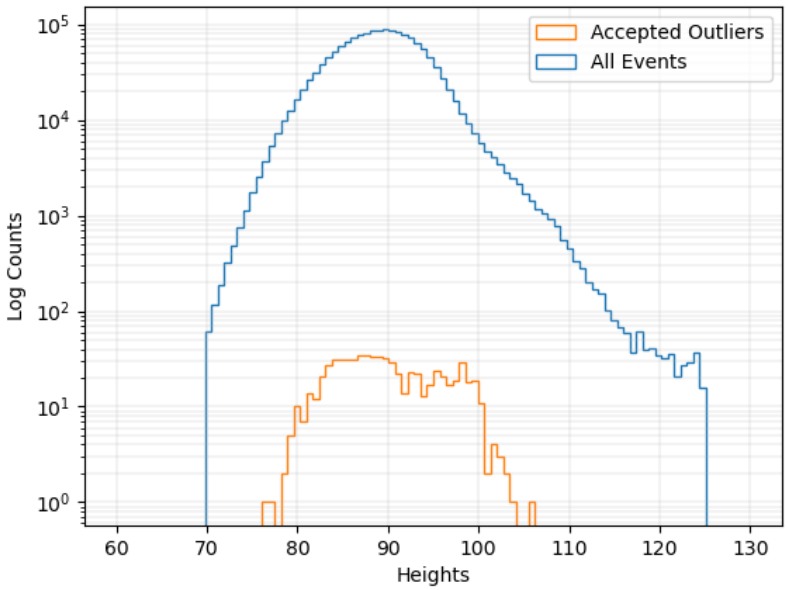

Figure 22: Overall Height Distribution

In Figure 22, the height distribution of all accepted events is depicted. The heights are represented in km. It can be confirmed that the accepted outliers are observed at an altitude of approx. 77 – 105 km.

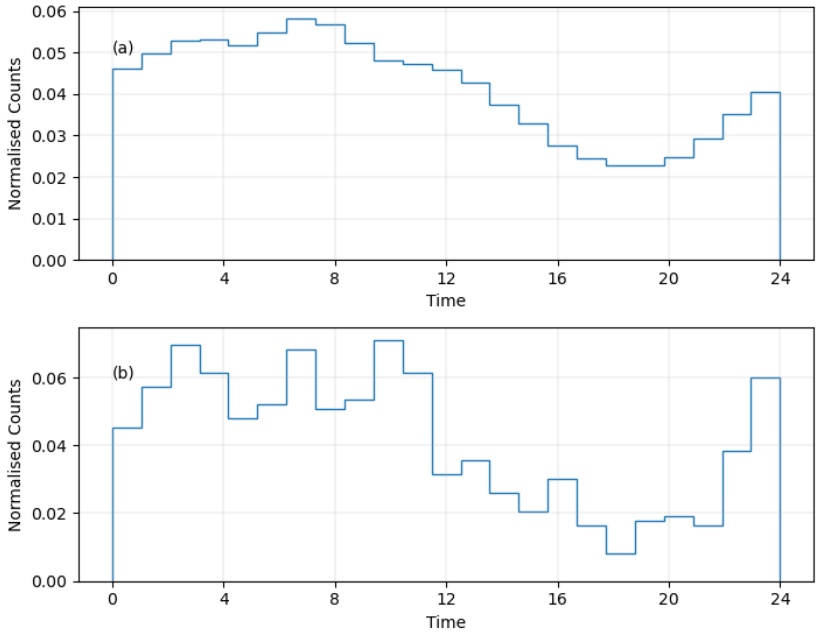

Figure 23: Overall Time Distribution

Figure 23 depicts the time distribution of the events and time is expressed in

hours. The individual plots are normalized so that the integral density remains one. The first subplot (a) corresponds to the distribution of all events and (b) corresponds to the accepted outliers. It can be inferred from the plot that there is no particular time of the day in which these events with higher residual value occurs.

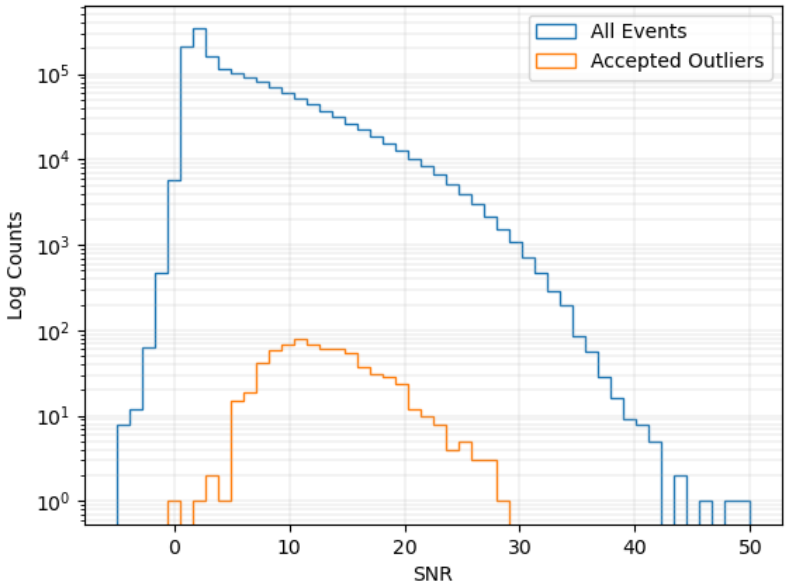

Figure 24: SNR Distribution

Figure 24 illustrates the SNR distribution of the overall events. The SNR of the accepted outliers range from approx. 0 to 29dB. A kurtosis analysis was done to analyze the dependency of the Gaussian characteristics on the SNR value. The result is illustrated in the below plot.

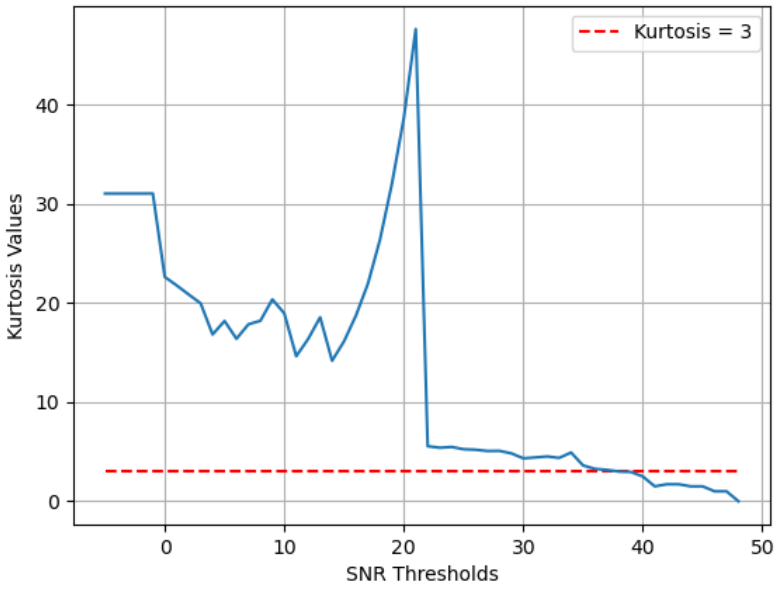

Figure 25: Kurtosis Values vs SNR Thresholds

Figure 25, shows the dependency of the distribution on the SNR value. The red line indicates a kurtosis value of 3, which corresponds to a Gaussian distribution. The distribution becomes Gaussian at a very high SNR threshold value of approx. 35. However, from Figure 24, it can be inferred that majority of the detections, including the three sigma outliers will be removed at this SNR threshold value.

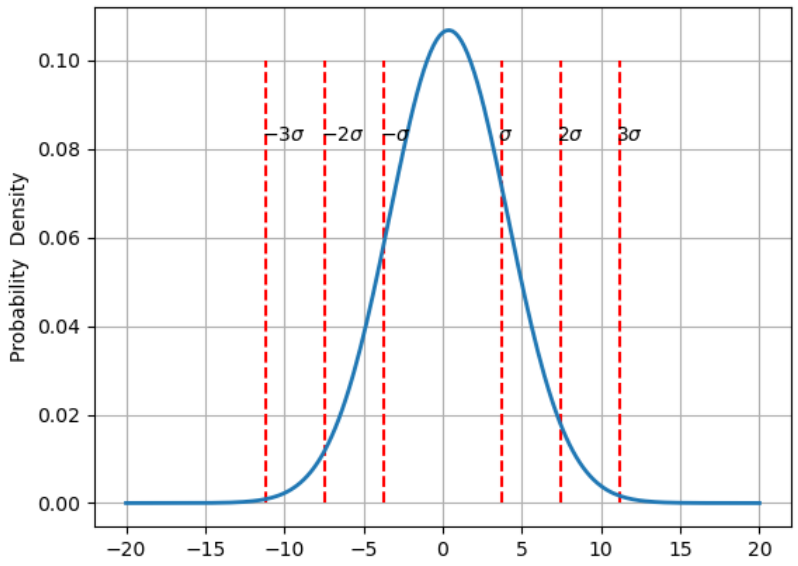

Figure 26: Probability Distribution after the Analysis

Figure 26 represents probability distribution of after analysis. The kurtosis

analysis of the distribution gave a value of 6.90 (when no SNR threshold was applied). This suggests the non – Gaussian nature of the distribution.

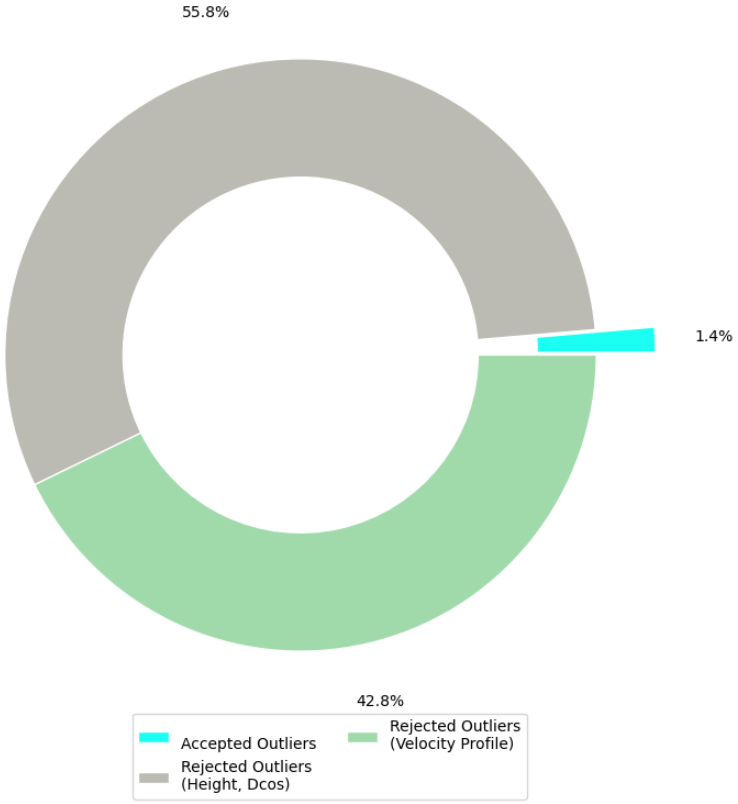

Figure 27: Proportion of the Rejected and Accepted Outliers

The plots shows the probability distribution and the percentage of percentage of the events accepted (rejected). Although, the acceptance percentage of the events, which were previously cleaned (greater than 3σ), is only 1.4%, these events needs to be further analyzed to examine the possible effects of these events in the atmosphere. However, these accepted outliers corresponds to only 0.045% of the overall good meteor detections.

6 Conclusions and Future Work

In this thesis, the outliers, which were cleaned in the routine process, has been analyzed for a duration of 3 months. Different tools based on which the outliers could be classified has been developed. Based on the tools and the voltage profile of the events the outliers have been classified. Additionally, a graphical user interface was developed for the verification of the events. All the analyzed data has also been saved into HDF5 files.

In future, the data saved into the HDF5 file could be used for identifying extreme vertical velocities similar to the one reported at IAP, if any; which is an ongoing research topic at IAP. Since the event id of the events are saved, the id could be used for determining their velocities. Further details, on extreme velocities can be found in ([3]).

Further improvements could also be done to the developed GUI. The GUI does not have an option to choose a different year, as at the time of developing the scope was limited to an analysis of 3 months. Adding an option to choose different years will indeed increase the usability.

In addition, in the Cross Verification tab, there is only possibility to look for events that occurred in a month. It is not possible to identify the day in which the event occurred. Adding an option to view events based on day of the year or a particular date could also be useful. This might be helpful to identify the events due to a meteor shower easily.

References

[1] W.K.HOCKING, B.FULLER, B.VANDEPEER : Real – Time Determination of Meteor – related Parameters Utilizing Modern Digital Technology, Journal of Atmospheric and Solar – Terrestrial Physics 63, 2001, p. 155 – 169.

[2] DAVID A. HOLDSWORTH, MASAKI TSUTSUMI, IAIN M. REID, TAKUJI NAKAMURA, TOSHITAKA TSUDA: Interferometric Meteor Radar Phase Calibration Using Meteor Echoes, Radio Science Vol 39, 2004.

[3] J.L. CHAU, R.MARINO, F. FERRACO, M. URCO, F.J: LÜBKEN, W.K.HOCKING, C.SCHULT, T. RENKWITZ, R. LATTECK : Radar Observation of Extreme Vertical Drafts in the Polar Summer Mesosphere, Geophysical Research Letters, Vol 48, Issue 16, 2021.

[4] J.L. CHAU, J.M. URCO, J. VIERINEN, B.J. HARDING, M. CLAHSEN, N.PFEFFER, K.M. KUYENG, M.A.MILLA, P.J. ERICKSON: Multistatic Specular Meteor Radar Network in Peru: System Description and Initial Results, Advancing Earth and Space Science Research Article, 2020.

[5] Zdenek Celplecha, Jiri Borovicka, W.Graham Elford, Douglas O. Revelle, Robert L. Hawkes, Vladmir Porubcan, Milos Simek : Meteor Phenomena and Bodies, Space Science Reviews 84, 1998, p. 327 - 471.

[6] G. STORBER, J.L. CHAU: A multistatic and multifrequency novel approach for specular meteor radars to improve wind measurements in the MLT region, Radio Science, Vol. 50, Issue 5, 2015, p. 431 - 442.

[7] J.L. CHAU, M. CLAHSEN: Empirical phase calibration for multi-static specular meteor radars using a beam-forming approach, Radio Science, Vol. 54, 2019, p. 60 - 71.

[8] M. CLAHSEN: Detektion und Identifikation von mit Hilfe des MMARIA Konzepts gewonnenen Specular-Meteor-Echos, Bachelor Thesis, Univeristät Rostock, 2015.

[9] M. CLAHSEN: Error analysis of wind estimates in specular meteor radar systems, Master Thesis, Univeristät Rostock, 2017.

[10] TUTORIALS POINT: Python GUI Programming, https://www.tutorialspoint.com/python/python_gui_programming.htm, Accessed on July 2021.

[11] PYTHON TUTORIALS: Tkinter Notebook, https://www.pythontutorial.net/tkinter/tkinter-notebook/ , Accessed on July 2021.

[12] PYTHON PROGRAMMING: How to Embed a Matplotlib Graph to Your GUI, https://pythonprogramming.net/how-to-embed-matplotlib-graph-tkinter-gui/ , Accessed on July 2021.

[13] REAL PYTHON: Python GUI Programming with Tkinter, https://realpython.com/python-gui-tkinter/ , Accessed on September 2021.

[14] KURTOSIS: Kurtosis, https://en.wikipedia.org/wiki/Kurtosis , Accessed on September 2021.

[15] SCIPY: scipy.stats.kurtosis, https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.kurtosis.html , Accessed on September 2021.

[16] GAUSSIAN DISTRIBUTION: Normal Distribution, https://en.wikipedia.org/wiki/Normal_distribution , Accessed on September 2021.

List of Figures

List of Abbreviations

AFE	Analog Front End
AOA	Angle of Arrival
AOD	Angle of Departure
CW	Continuous Wave
Dcos	Direction Cosines
DRX	Digital Receiver
DTX	Digital Transmitting Unit
GUI	Graphical User Interface
HDF5	Hierarchical Data Format Version 5
HPA	High Power Amplifier
IAP	Institut für Atmosphärenphysik
MIMO	Multiple Input Multiple Output
NaN	Not a Number
MLT	Mesosphere and the Lower Thermosphere
MMARIA	Multistatic Multi-frequency Agile Radar for Investigation of the Atmosphere
OS	Operating Systems
PMSE	Polar Mesospheric Summer Echoes
PRF	Pulse Repetition Frequency
RDA	Radar Data Archive
RDB	Radar Data Buffer
SDR	Software Defined Radio
SIMONe	Spread Spectrum Interferometric Multistate Meteor Radar Observing Network
SMR	Specular Meteor Radar
SNR	Signal to Noise Ratio

Acknowledgements

I would like to express my sincere thanks to Prof. Dr.-Ing. habil Andreas Ahrens for providing me the research opportunity by being my supervisor at the university.

I would like to extend my gratitude to Prof. Dr. Jorge L. Chau and Dr. Toralf Renkwitz for providing me the opportunity to work on this thesis topic at IAP, Kühlungsborn.

My sincere thanks goes to Prof. Dr. Jorge L. Chau and MSc. Matthias Clahsen for their great supervision and constant support throughout this thesis.

Finally, I would like to thank my parents and friends for their perpetual encouragement throughout my studies.

Declaration of Independent Work

I Reshma Mary Roy; hereby declare that I completed this work independently and that I have used no aids other than those referenced.

The part of the work, which include phrases or points taken from other sources, are clearly marked with the origin of information. This also applies to diagrams, sketches, visual representations as well as for sources from the Internet.

I also declare that I have not submitted this work in any other testing procedure as an examination paper, nor will I in the future.

The submitted written version matches the version stored in the data medium.

Wismar, 27.10.2021

Place, Date Signature