REAL-TIME SMALL-SIZE SPACE DEBRIS DETECTION WITH EISCAT RADAR FACILITIES

J Markkanen and M Postila

Final Report
of
ESOC Contract No. 16646/02/D/HK(CS)
with
EISCAT Scientific Association

ESA/ESOC Technical Management
M Landgraf

February 2005

Generated from LaTeX source, August 15, 2012.

Following a feasibility study in 2000–2001 about using EISCAT radars to detect centimetre-sized space debris in the frame of an ESA contract, the present study was aimed at doing the debris detection and parameter estimation in real-time. A requirement in our work is to "piggy-back" space debris measurements on top of EISCAT's normal ionospheric work, without interfering with those measurements, and to be able to handle on the order of 500 hours of measurements per year. We use a special digital receiver in parallel to EISCAT's standard receiver. We sample fast enough to correctly band-pass sample the EISCAT analog frequency band. To increase detection sensitivity, we use amplitude domain integration—coherent pulse-to-pulse integration—of the samples. The coherent integration is built into our method of target parameter estimation, which we call the MF method, for "match function" or "matched filtering". The method is derived from Bayesian statistical inversion, but reduces, with common assumptions about noise and priors, to minimizing the least squares norm $||z(t) - b\,\chi(R, v, a; t)||$, where z is the measured signal and $b\,\chi(t)$ is the model signal. Because the model signal depends linearly on the amplitude b, it is sufficient to maximize the magnitude of the inner product (cross correlation) between z and χ, the amplitude estimate is then determined by direct computation. The magnitude of the inner product, when properly normalized, is the MF. Our original Matlab implementation of the MF method during the precursor study was about four orders of magnitude too slow for real-time applications. In this study, we have gained the required speed factors. A factor of ten comes from using faster computers, another factor of ten comes from coding our key algorithms in plain C instead of Matlab. The largest factor, typically 100–300, comes from using a special, approximative, but in practice quite sufficient, method of finding the MF maximum. Test measurements show that we get real-time speed already when using a single 2 GHz dual-processor G5 Macintosh to do the detection computations. The measurement campaigns also show that the achieved sensitivity at EISCAT UHF radar, with a wavelength of 32 cm, corresponds to detecting a 2.2 cm diameter target at the range of 1000 km. We register typically 15–25 targets per hour in the 500–1500 km altitude band. A final 101 h debris measurement campaign, conducted in parallel with a standard EISCAT ionospheric experiment, confirmed that we can now essentially in real-time handle longish measurements. There remain partially open issues, like the efficiency of the coherent integration on the theory side; the need for more robust and automated execution of the experiments in the software side; and the need of ensuring the availability of eventual replacement of our special hardware. Nevertheless, at the end of 2004, we are technically in the position to start routine space debris measurements with the EISCAT system.

Contents

List of Figures

List of Tables

1. Introduction

1.1. Background and overview

It is estimated that there are approximately 200,000 objects larger than 1 cm currently orbiting the Earth, as an enduring heritage of four decades of space activity. This includes the functioning satellites, but by far most of the objects are what is called space debris (SD), man-made orbital objects which no longer serve any useful purpose. Many of the small-sized (less than 10 cm) particles are due to explosions of spacecraft and rocket upper stages, but there are also exhaust particles from solid rocket motors, leaked cooling agents, and particles put into space intentionally for research purposes. The large (> 10 cm) objects have known orbits and are routinely monitored by the US Space Surveillance Network, but information of the smaller particles is fragmentary and mainly statistical. Especially, in Europe there is no radar that is routinely used for monitoring small-size SD.

In 2000-2001, we, together with our colleagues from Sodankylä Geophysical Observatory, undertook a study for ESA about the feasibility of using the EISCAT ionospheric research radars for space debris measurements [8]. Since the early 1980's, the EISCAT mainland radars—the Tromsø UHF radar operating at 930 MHz and the VHF radar operating at 225 MHz—have been performing ionospheric measurements to the order of 2000 hours per year; and since the late 1990's, after the EISCAT Svalbard radar became operational, EISCAT has been measuring more than 3000 hours annually. The interest is to use a substantial amount of these operating hours for simultaneous space debris measurement in cost-effective way. In the initial study, we showed that it is feasible, and technically straightforward, to perform SD measurements in parallel with normal EISCAT ionospheric measurements, without interfering with those measurements [10].

Our measuring approach is to operate a separate digital receiver back-end, which we call the SD receiver, in parallel with EISCAT standard digital receiver. This allows us to implement our own, amplitude domain data processing, which we call the the match function or matched-filtering (MF) method. The MF method makes use of the long coherence time of a signal reflected from a small target to increase detection sensitivity, via pulse-to-pulse coherent integration. To make the hardware as simple and cheap as possible, the custom-made part of the SD receiver is basically just a fast sampler and digital demodulator; the MF computations are done in fast but still cheap general purpose workstations. The SD receiver samples the EISCAT analog signal, at the second intermediate frequency (10 MHz) level, fast enough to capture the relevant frequency channels into a single digital stream, without doing the customary channel separation. Typically during a measurement, we sample at the rate of about a million complex samples per second continuously, producing more than 10 GBytes of data per hour. Early on, ESA suggested that we should strive to do the data analysis in real-time so that the raw data could be quickly disregarded.

A straightforward implementation of the MF method implies long data vectors, with lengths of hundreds of thousands complex points, to be Fourier-transformed a few thou-

sand times, per every second of raw data; basically, one is computing power spectra for a relatively large number of range gates. At the Space Debris III conference in 2001, we had to concede that with the processing speed that we had achieved at the time, it would take several centuries of CPU time to analyze just one year's quota of EISCAT space debris measurements. However, soon afterwards, M Lehtinen of Sodankylä Geophysical Observatory, who was the project leader of the precursor study, realized that by accepting some loss of detection sensitivity and a small bias in the velocity estimate, it would be possible to speed up MF computation drastically, typically by more than two orders of magnitude. We use the term fast match function algorithm (FMF) for the resulting computation scheme.

The results of the initial study were encouraging. The achieved detection sensitivity was equivalent to being able to observe spherical targets with diameters of about 2 cm from 1000 km range. With the advent of the FMF-algorithm, the processing speed, though still sluggish, was starting to become useful. In 2003, ESA commenced the present study, to bring the analysis of large amounts of EISCAT SD data up to real-time speed [9]. The study has achieved the necessary processing speed. In addition to the factor of 100 delivered by the FMF-algorithm, we now use computers that are about ten times faster than what we had available in 2001. A final required factor of ten to the speed was obtained by coding the MF- and FMF-algorithms in C, instead of using Matlab as was done in the initial study.

The EISCAT system [1, 2, 3] consists of three separate radars: monostatic VHF radar, located near Tromsø, Norway, and operating at 224 MHz; monostatic but two-antenna EISCAT Svalbard Radar in Longyerbyen, Svalbard, operating at 500 MHz; and tristatic EISCAT UHF radar at 930 MHz, with transmitter in Tromsø and receivers in Tromsø and in Kiruna, Sweden, and Sodankylä, Finland. All the transmitters operate in the megawatt peak power range and routinely utilize high (10–20%) duty cycles.

Even though routinely picking-up hard target echoes, standard EISCAT data processing is not optimized for hard targets. The characteristic feature expected from small hard targets is long signal coherence time, several hundred milliseconds. By a signal's (phase-) coherence we mean that the signal phase $\phi_0(t)$ obeys a deterministic functional form for some duration of time, called the coherence time.

EISCAT's normal ionospheric signal has coherence time less than a millisecond in most parts of the ionosphere. This time is much shorter than the interval between transmitted pulses, the interpulse period IPP, which in EISCAT typically is 3–10 ms. Therefore, echoes from individual pulses are uncorrelated, and can only be added up in the power domain. This is done by computing, for each of the received pulses separately, signal autocorrelation functions, or, equivalently, power spectra, for a set of range gates, and then adding these power-domain quantities. This is called non-coherent pulse-to-pulse integration. We emphasize here, and will discuss in more length in section 3.6, that within a *single* transmission-reception (T/R) cycle, computing range-gated power spectra achieves coherent integration of the samples. In fact, for a single uncoded pulse, the MF method, too, in effect just computes range-gated power spectra.

To achieve coherent integration from pulse to pulse, the MF method adds the echoes from different T/R cycles in amplitude domain, taking care that the pulses are added with equal phase. The method, in essence, removes *all* phase variation from the signal before adding the samples. This is achieved by guessing the phase factor $e^{i\phi_0(t)}$ of the signal, and canceling it by multiplying the signal by the complex conjugate of the guess, $e^{-i\phi_0(t)}$. The guesses in our implementation are generated by brute force. We search

through a large set of parametrized model functions, and use the one which achieves best cancellation of the phase, that is, which results in largest integrated amplitude. After the phase variation has successfully been removed, the remaining part of the signal can be safely integrated, both within a single pulse, as well as from pulse to pulse. Incidentally, dividing the radar data into T/R cycles is artificial from the MF method point of view. It is more natural to consider the totality of transmission during an integration period as just a waveform pattern, to be matched against the totality of reception, irrespective of how these two are interleaved. In particular, there is no need for the T/R cycles to be identical, either in terms of length or transmission content.

As long as the signal stays coherent (obeys the assumed model), coherent integration suppresses the non-coherent background noise, so that the effective signal-power to noise-power ratio increases directly proportionally to the number of pulses integrated. This increases detection sensitivity. Non-coherent integration, instead, does not increase signal-to-noise ratio. The drawback in coherent integration, in addition of it being computationally more demanding due to the long data vectors, is that if the signal model is not accurate, the ensuing phase error will quickly eat into the integrated signal amplitude, rendering longer integration useless.[1] In our case, coherent integration beyond about 300 ms does not seem to improve detection sensitivity significantly.

Part of the reason for the unexpectedly short apparent coherence time is that, although we (see section 3.3) will derive a signal model that we believe should be fairly accurate for small structureless targets, for performance reasons we cannot actually use the ideal model. The approximative model that we do use, both in the MF and FMF algorithms, is suitable for narrow-band (single-frequency-channel) transmissions. The different frequency channels in a multi-frequency signal will have slightly different Doppler-shifts because the Doppler-shift depends on transmission frequency. It is impossible to cancel the Doppler phase factors simultaneously using only the single model phase factor which is available in the approximative model. We elaborate on some of these aspects in sections 3.6 and 3.8 of this report, but a detailed study of the coherent integration efficiency will be postponed to later study.

We derive formally (see section 3.1) the MF method via Bayesian statistical inversion. Within the Bayesian approach, the estimates for the basic parameters range, radial velocity, radical acceleration and signal amplitude, or signal total energy, are found as the most probable values, given the measured noisy signal. With our assumptions, this solution is also the one that minimizes the least squares norm between the measured signal and the set of model functions; the solution is also the maximum likelihood solution.

We have, during four measurement campaigns during the two years of this study, collected and analysed about 150 hours of data, all at the EISCAT UHF radar in Tromsø. These data have been taken mostly for method development and verification purposes; assessing any possible physical significance of the about 2500 events that we have got is outside the scope of this work. Nevertheless, we show, with minimal commentary, the bulk analysis results in chapter 6 of this report. There we plot the observed detection rate as function of altitude in six data sets (the peak detection rate is about 2.5 events per hour per 50 km bin near 1000 km altitude); show the effective diameter of the targets as a function of range (we observe events down to effective diameter of about 21–22 mm at 1000 km range); and plot the detection rate as function of target's radar cross section

[1] We admit that there may be a grain of truth in the statement in a recent book which claims that "most radars utilize non-coherent integration", because "maintaining coherency [...] is very costly and challenging to achieve." [7]

(RCS). We must emphasize here, however, that all our results concerning target size are, at best, "lower bounds". We cannot say much about the actual target cross sections, let alone target physical diameters. Perhaps the single major deficiency in EISCAT, in comparison with some other radars used for space debris observations, is that the EISCAT antennas do not have monopulse feed. At the moment at least, there is no way available that would allow pinpointing the actual direction of a target within the radar beam, and so the target's radar cross section cannot be deduced from the measured signal strength. We hope that in the future we can partly alleviate this problem by collecting fairly large amounts of data—perhaps about 500 hours per year—so that the antenna beam pattern can be taken into account statistically, and meaningful comparisons to space debris models made. Finally, we also plot target velocity and acceleration as function of altitude.

1.2. Study objectives

According to our contract agreement with ESA/ESOC [9], the objective of this study was
"To develop methods to perform real-time detection of small-sized debris objects in LEO during routine EISCAT operations. The new methods shall be based on the capabilities (soft- and hardware, as well as data processing procedures) for debris detection that have been developed in the precursor study. The routine real-time detection shall be demonstrated during standard EISCAT experiment campaigns."

1.3. Structure of the study and organization of this report

The basic strategy in the study was to proceed in small, verifiable steps from the "proof-of-concept" phase achieved in the precursor study to the practical execution of SD measurements in real-time. For this purpose, the contract stipulated the following three work packages

1. Updating of the data processing methods and algorithms.
2. Real-time detection and parameter estimation of space debris echoes in raw data acquired during the precursor study.
3. Real-time detection during routine operations of one selected EISCAT radar facility.

In work package 1, we implemented in C language the match function (MF) and the fast match function (FMF) algorithms that we had developed in Matlab during the precursor study. We used recorded raw data from the precursor study to verify that the C implementations gave (essentially) the same parameter values and the same detection sensitivity as the Matlab implementation. Both the MF and FMF algorithms were embedded as subroutines inside a new, C language "main program", the debris scanner DSCAN. The built-in timers of DSCAN were used to evaluate the performance of the newly implemented algorithms, in that time, in a new 1 GHz dual-processor G4 Macintoch (which later became our data collection computer). It was evident after the very first speed measurements that real-time speed was within grips more easily than we had anticipated. For the rest of the study, the emphasis could be shifted from

raw performance improvements to advancing the understanding of the measurements, and building the software and hardware infrastructure necessary in actually carrying out long measurement campaigns.

In work package 2, we did our first test measurement, in October 2003 at the Tromsø UHF radar. At the time, we had only the data collection computer available. We recorded the raw data to disk. We also did a short preliminary test of running the DSCAN in the data collection computer simultaneously with ongoing data collection, and were encouraged by the smooth performance. After the campaign, we used the recorded data, and the raw data of the precursor study, to measure the scanner performance, and to develop data analysis of the detected events in the new environment.

In work package 3, we conducted a 24 hour test measurement campaign in Tromsø, in March 2004. For this campaign, we had our hardware updated to a "baseline final" configuration, where we had, in addition of the data collection computer, a fast new workstation for target detection and parameter estimation, and a fast network connection between the two computers. We demonstrated that it was possible to do all the phases of the data processing at real-time speed, with little manual intervention. But even though we did produce event parameters automatically, in order to demonstrate the speed, it was clear that fully automatic operation was not meaningful, due to the problems of removing bad data before doing parameter estimation. The benefits of some manual data cleaning have come even more evident during a 16 hour beam park 2004 measurement which we performed in September 2004, and a 100 hour measurement that we did in November 2004. Even though neither of these two campaigns does strictly fall inside the contract limits, we did use the opportunity to further develop our data processing and experiment execution method in real measurements; we include the analysis results from those data sets into this report.

In our contract, there was a fourth work package, titled "Investigation of the Possibilities of Orbital Parameter Determination with the EISCAT radars." Our purpose had been to evaluate what use we could make of the tristatic UHF radar system in this context. To that end, during the March 2004 campaign, we recorded raw sample data also in the Kiruna and Sodankylä sites, using the standard EISCAT receivers. However, unanticipated but necessary work to clarify several weak points in our measurement theory, which had become irritant during summer 2004 when we did reassess the theory for a COSPAR 35 talk, caused us to start falling seriously behind the contract schedule in the autumn 2004. In the late autumn, it was agreed to exclude the work package four from the present work.[2]

Even though the incremental work package structure, the packages 1–3, turned out to be both convenient and useful in the course of the work, in this report we are not going to trace the gradual development. Instead, we structure the report to describe the current state of our measuring theory; our hardware; and our software. We start, after this introductory chapter, with the hardware, by describing the space debris receiver and its host radar environment, the EISCAT Tromsø UHF radar in chapter 2.

In chapter 3 and appendix A we give a fairly detailed presentation of our measurement and data processing theory, the match function method. In addition of deriving the ideal, exact method in sections 3.1 and 3.3, we derive in section 3.4 the approximative algorithms which we actually use in the numerical work. We discuss the connection between the common range-gated power-spectrum method of detection and our MF method in

[2] See section 4 of the minutes of progress meeting 5, held in Kiruna, October 28th, 2004.

section 3.6. We deal in this report with several matters that have been omitted in our earlier reports. These include the effect of digital filtering to the signal energy estimate (section 3.2 and appendix A), and the question of background subtraction in section 3.5. We describe, for the first time, the crucial FMF algorithm, and discuss some of its properties in section 3.7. We consider the consequences for coherent integration of dual-frequency transmission in section 3.8.

In chapter 4, and two appendixes, we describe the software. We first give an overview in section 4.1, including an overview of the real-time performance. Then we describe the time-critical program module, the scanner DSCAN, as well as its control environment, in section 4.2 and appendix B. In section 4.3 we describe the event archiver program DARC, and discuss the necessity of interactively cleaning the hitlist files produced by the scanner. In section 4.4 we provide a top-level flowchart of the parameter estimation Matlab script DANALYSIS. Section 4.5 mentions our present ideas and initial implementation for an overall debris measurement real-time control software DROS, using a modified version of the standard EISCAT real-time control software, the so called EROS system. Section 4.6 shows results of DSCAN performance measurements, including internal timing of the GMF and FASTGFM subroutines that implement the MF and FMF algorithms.

In chapter 5 we describe our space debris measurements. We describe in section 5.1 the EISCAT experiments that we have been using in the campaigns, list our data sets in section 5.2, and then show some fifteen pages of summary plots of the analysed data in sections 5.3 through 5.5.

2. Hardware

2.1. EISCAT UHF radar

So far in our SD measurements we have used mainly the EISCAT UHF radar. The 32 m UHF antenna has a fully steerable parabolic dish, has Cassegrain optics, and has rotation rate of about 80°/min both in azimuth and elevation. The antenna pointing direction is calibrated using celestial radio sources, and is believed to be accurate better than 0.1° in most directions.

A block diagram of the UHF radar's Tromsø site is shown in Fig. 2.1. The EISCAT receivers at all three UHF sites, Tromsø, Kiruna, and Sodankylä, are almost identical, the main difference being that at the receiving-only sites Kiruna and Sodankylä there is no need for the duplexer and the receiver protector. Also the polarizer arrangements are somewhat different. The Tromsø UHF receiver has a cooled preamplifier, giving a system temperature $T_{\mathrm{sys}} \approx 110$ K. Kiruna and Sodankylä have recently moved over to uncooled, HEMT-based (high electron mobility transistor) preamplifiers, with system temperatures around 50 K. The radar's radio-frequency (RF) band is centered at 928 MHz, and there are 14 transmission frequencies available, 300 kHz apart. In the most common EISCAT experiment modes, two frequency channels are used. Recently those have been centered at 929.9 MHz (EISCAT frequency F13) and 930.2 MHz (F14). The RF signal is mixed in two stages to the second intermediate frequency (IF2) band, using local oscillators at 812.0 MHz and 128 MHz, so that F13 maps to 10.1 MHz and F14 to 9.8 MHz. The band is formed by the radar's antialiasing filter, which is 6.8 MHz wide and centered at 11.25 MHz (see Fig. 3.2 on p. 27).

In the standard EISCAT data processing, the second IF is digitized by a 14-bit analog-to-digital converter (A/D), which produces a continuous sample stream at the rate of 15 Msamples/s. The stream of IF2 samples is distributed to multi-channel, VME-based, EISCAT digital receiver, each channel occupying one slot in a VME crate. Custom hardware in each channel performs quadrature detection, followed by sampling rate reduction appropriate to the typical 10–50 kHz final channel bandwidth. The baseband sample stream is buffered, and further processing to averaged sample correlation products is done on UNIX-based computers (EISCAT uses computers based on SPARC-processors, and the Solaris flavour of UNIX).

The EISCAT UHF transmitter consists of a programmable radar controller that generates the pulse patterns at DC level, either uncoded on/off pulses or various classes of binary phase codes; an exciter system that converts the radar controller output to RF around 928 MHz; and a klystron power amplifier that consists of two klystron tubes, in principle able to deliver combined peak power of about 2.5 MW. The power during all our space debris measurements has been considerably lower, at 1–1.5 MW. The maximum transmitter duty cycle is 12.5%, and duty cycles near this value are also used in most experiments in practice. The time and frequency base at all EISCATsites is from the GPS system.

2.2. The space debris receiver

To be able to use our own data processing, optimized for hard targets, we use a special digital receiver back-end, the space debris receiver. Signal to the space debris receiver is branched off from the EISCAT analog signal path at the second IF (IF2) level. Figure 2.1 shows the main blocks of the SD receiver, connected to the EISCAT UHF system at the Tromsø site.

EISCAT standard data processing handles a multi-frequency transmission in the traditional way, by feeding the IF2 data to multiple hardware channels, each tuned to a particular center frequency. The end result is several sample streams, one for each channel. Our approach in the SD receiver is different. We sample fast enough to capture the relevant part of the analog IF2 band into a single digital stream. We call this type of data multichannel complex data [5]. According to the (bandpass-) sampling theorem, if the spread of frequencies is B MHz, we need to take B million complex samples per second, in the minimum. For our most often used measuring mode, where there are two frequency channels 300 kHz apart, we have normally used 500 kHz sampling rate. But we have also verified that the SD receiver can handle sampling speeds up to 2.5 Msamples per second.

In addition to the standard reception, our data processing requires that the transmission waveform is measured. As indicated in Fig. 2.1, EISCAT provides the transmission sample signal (TS) time-multiplexed into the same data path as the reception. The multiplexer switch is controlled by the receiver protector bit ("TX bit"), generated by the EISCAT radar controller microprocessor. We routinely record the receiver protector bit into our data stream to mark out the transmission blocks. The bit is stored into the least significant bit of the imaginary part of the $16 + 16$-bit complex integer data words. With this arrangement, the transmission sample signal gets automatically sampled with the same sampling rate as the actual reception (though we would actually like to sample it with a higher rate).

The core of the data acquisition system is a custom PCI-board which performs signal sampling, quadrature detection and sampling rate reduction. The board was developed originally for ionospheric tomography by the now defunct Finnish company Invers Ltd.

The A/D converter on the PCI board samples at 40 MHz. The resulting real-valued sample stream is processed by programmable logic chip, from the Xilinks SpartanXL family, to perform quadrature detection, essentially by doing Hilbert transform. The result of the transform is a complex-valued 10 MHz sample stream, which represent the negative frequency part of the spectral contents of the analog input. The chip then decimates the 10 MHz stream to the final sampling rate. Typical decimation factor M is 20, which yields 500 kHz final sampling rate. The decimation is done by adding samples in blocks of M; this ensures proper filtering. This processing chain is analysed in more detail in section 3.2 and in appendix A, where we especially pay attention to the effects of the decimation filter.

It may be noted that there is no separate multiplication to baseband in this scheme. Instead, the customary frequency component at baseband is created by the undersampling. With the 40 MHz primary sampling rate, the arrangement requires that the band-limited analog input is centered at 10 MHz. Although it is possible to run the A/D converter on the board at other sampling rates, the 40 MHz is a most convenient choice. That the two frequencies EISCAT nowadays most often use in the standard measurements, are 10.1 MHz and 9.8 MHz, is a happy coincidence. The next version of

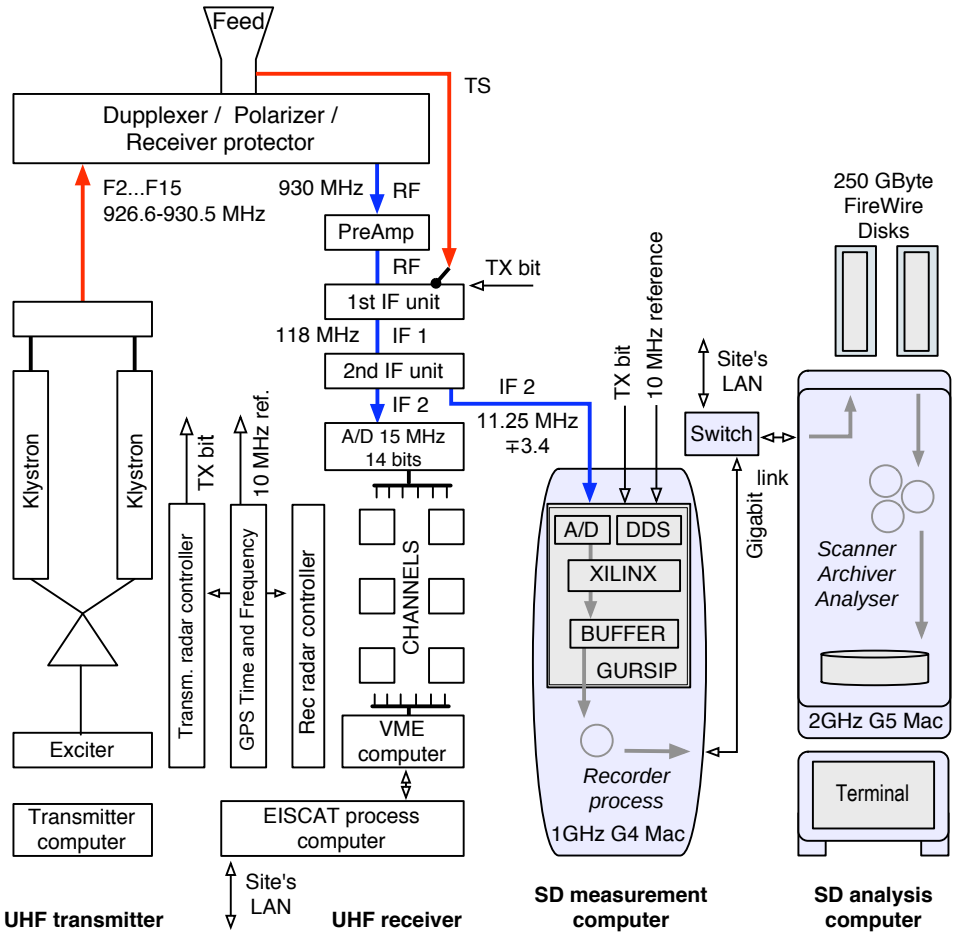

Figure 2.1.: The space debris receiver connected to the EISCAT UHF radar. The SD receiver consists of a measurement computer and an analysis computer. The measurement computer hosts a custom signal processing board (GURSIP). The primary analog input to the SD receiver is the EISCAT second intermediate frequency band. The input contains, time-multiplexed, both the standard received signal and the transmission sample signal (TS). On the processing board, there is an analog-to-digital converter (A/D) taking 40 megasamples per second; a direct-digital-synthesizer chip (DDS) which provides clock signals on the board, phase-locked to the host radar's 10 MHz frequency reference signal; two Xilinks signal processing chips (XILINX) to perform signal demodulation and sampling rate reduction; and a memory buffer for temporary storage of the complex samples. The recorder program running on the measurement computer moves the samples over the gigibit network link to an external FireWire disk, mounted on the analysis computer. Target detection is done by the scanner program running on the analysis computer, using the FMF-algorithm. After detection, two other software modules, the archiver and the analyser, store away the event's raw data and estimate the target parameters.

the SD receiver should have a complex mixer built-in.

The PCI board is mounted on a Macintosh G4 workstation, running under the Mac OS X version of UNIX. The Mac G4 we call the measurement computer. In addition, there is a dual-CPU Macintosh G5 computer for data analysis. The Mac workstations are connected to each other via a gigabit Ethernet link, and are also connected to the site LAN. The measurement computer runs software from Invers Ltd to read the sample data from an onboard buffer and write them to hard disk, either local disk, or, normally, a disk mounted over the gigabit link from the analysis computer. The data accumulation rate to the disk is between 7 and 30 GBytes per hour (2–8 MBytes s^{-1}), depending on the sampling rate. The maximum sustainable transfer rate over the data link in this configuration is more than 20 MBytes s^{-1}, so even the 8 MBytes s^{-1} rate is only a minor load, and does not affect significantly the computing performance of either of the participating workstations. The LAN connection is used to access the EISCAT process computer, to update the time base in the G4 and G5 once every 5 minutes, using the ntp protocol. This ensures that the time base in the Macs stays within 20 ms of the time kept in the EISCAT system. This is more than adequate for time-stamping space debris events.

3. Theory

3.1. The match function method

We want to estimate the parameters of a hard target echo $s(t)$ in the presence of white gaussian noise $\gamma(t)$, in an optimal way. We denote by $z(t)$ the received signal,

$$z(t) = s(t) + \gamma(t)\,. \tag{3.1}$$

We denote by $x(t)$ the transmission sample signal (the signal TS in Fig. 2.1). We ignore here the frequency translations done in the actual receiver, treating z, s, x and γ as complex-valued (detected) signals. The frequency translations affect both the echo signal and the transmission sample signal by a common factor of the form $\exp(i\omega_{\mathrm{LO}}t)$, where ω_{LO} is some local oscillator frequency, and hence cancel out of the correlation products like $s(t)\overline{x}(t)$.

To find an optimal estimate (or at least a well-defined estimate), we will use the approach of Bayesian statistical inversion. The basic idea is to use a parametrized model for the signal $s(t)$ and find the most probable signal among the set of model signals, given the measured signal z.

We specify the model signals explicitly in section 3.3. Here we will make use only of the property that the model depends linearly on one parameter, the complex amplitude b, and in addition depends on some other parameters (range R, radial velocity v and radial acceleration a in our case), which we collectively denote by θ, so that

$$s(t) = b \cdot \chi(\theta; t). \tag{3.2}$$

We sample the signal $z(t)$ using sampling interval τ_{s}, and get N samples z_n during a time interval T_c, which we call the integration time or the length of the coherent integration.

It makes sense that after a specific measurement result (-vector) z, some parameter values (b, θ) are to be considered more likely than others, in a way that depends on z. That is, the probability of the various imaginable values should be describable by some conditional probability density, with z in the condition. In the Bayesian world view, that density is termed the posteriori density, and is denoted here by $D_{\mathrm{p}}(b, \theta|z)$. The inversion problem is to utilize the measurement to find the posteriori density. The posteriori density is the most complete inference that can be made about the parameter values, based on the measurement. Normally, one wants to condense the inference to single numbers, the parameter estimates, together with some simple measures of error like some confidence intervals. There is no unique way to select "best" estimates, but the standard Bayesian criterion is to use the most probable values:

$$(\widehat{b}, \widehat{\theta}) = \arg\max_{b,\theta} D_{\mathrm{p}}(b, \theta|z). \tag{3.3}$$

We now derive the posteriori density. We denote by $D_1(z_n|s_n)$ the conditional probability density of z_n, given s_n. This is just the probability distribution of the value of the

nth noise sample $\gamma_n = z_n - s_n$,

$$D_1(z_n|s_n) = \frac{1}{\pi\sigma^2} \mathrm{e}^{-\frac{1}{\sigma^2}|z_n - s_n|^2}, \tag{3.4}$$

where σ^2 is the variance of the complex gaussian noise. We assume that the noise is white so that the noise samples are uncorrelated. Then the conditional joint probability density to produce a particular measured vector z if the actual signal vector is s, is

$$D(z|s) = \prod_{n=0}^{N-1} D_1(z_n|s_n) = \frac{1}{(\pi\sigma^2)^N} \cdot \mathrm{e}^{-\frac{1}{\sigma^2}\|z-s\|^2}. \tag{3.5}$$

The density $D(z|s)$ is called the direct theory. Given the direct theory, the Baysian solution to the inversion problem is

$$D_\mathrm{p}(b, \theta|z) = C'(z) \cdot D_\mathrm{pr}(b, \theta) \cdot D(z|s). \tag{3.6}$$

Here $C'(z)$ is normalization factor. The new factor, $D_\mathrm{pr}(b, \theta)$, is called the prior density. The prior density is a weight that can be used if it is known a priori—before making the measurement—that some particular signals $s(b, \theta)$ tend to occur more frequently than some others.[1] Using D_pr might actually make sense when measuring space debris, to throw out detections with highly unlikely parameters. But so far we have used constant priors. For constant prior, it follows from Eq. (3.6) and Eq. (3.5) that the sought-for posteriori density is

$$D_\mathrm{p}(b, \theta|z) = C(z) \cdot \mathrm{e}^{-\frac{1}{\sigma^2}\|z - b\cdot\chi(\theta)\|^2}. \tag{3.7}$$

It follows that finding the most probable parameters amounts to minimizing the least-squares norm,

$$(\widehat{b}, \widehat{\theta}) = \arg\min_{b,\theta} \|z - b \cdot \chi(\theta)\|. \tag{3.8}$$

That we should arrive at this most basic technique of parameter estimation, least-squares fitting, is perhaps not surprising. But what we have gained by walking though the Bayesian route, is that not only do we have a method for acquiring the parameter estimates, but we have an explicit expression, Eq. (3.7), for the posteriori density. In the future, we intend to make use of the posteriori density in error analysis. Due to highly non-linear dependence of the model functions χ on the parameters-to-be-fitted, θ, error estimation is not trivial.

A straightforward approach to the minimization problem expressed in Eq. (3.8) is to discretize the parameter space and perform an exhaustive search. We now show that the search space dimension can be reduced by one by making use of the property that the amplitude b enters the problem linearly. Our result can be confirmed analytically, but will be here reasoned from basic vector geometry. Referring to Fig. 3.1, the set $\mathcal{M}$ of model vectors $\{b\,\chi(\theta)\}$ consists of 1-dimensional rays $\mathbb{C}_\chi$ through the origin of N-dimensional complex vector space $\mathbb{C}^N$. The rays are generated by a set of basic vectors $\chi(\theta)$. According to Eq. (3.8), we need to find the shortest distance between the measured point z and $\mathcal{M}$. The figure suggests that we need first to find the ray $\mathbb{C}_{\widehat{\chi}}$ that is as parallel as possible with the vector z. Then the point in $\mathcal{M}$ that is nearest to z

[1] $D_\mathrm{pr}(b, \theta)$ is, by definition, equal to the marginal density $\int D_\mathrm{tot}(b, \theta, z)\, dz$ of the probability distribution $D_\mathrm{tot}(b, \theta, z)$, the probability distribution of the whole system, which contains on equal footing both the measurement results z and the signals $s(b, \theta)$.

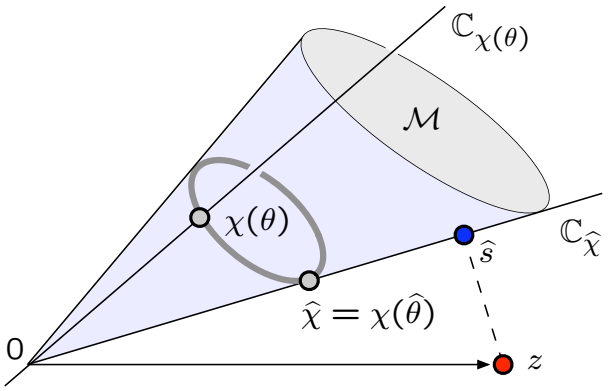

Figure 3.1.: Geometric interpretation of the MF method. The sought-for best estimate of the signal is the point $\widehat{s}$ in the set $\mathcal{M}$ of model functions that is nearest to the measured signal z. The "conically-shaped" set $\mathcal{M}$ consists of rays $\mathbb{C}_\chi = \{a\chi \colon a \in \mathbb{C}\}$, generated by a set of basic model signals $\chi(\theta)$. $\mathrm{MF}(\theta)$ is defined as the length of the orthogonal projection of z onto the ray $\mathbb{C}_{\chi(\theta)}$. Maximizing $\mathrm{MF}(\theta)$ gives the ray $\mathbb{C}_{\widehat{\chi}}$ that gets as near to the point z as is possible in $\mathcal{M}$. The estimate $\widehat{s}$ is the orthogonal projection of z onto $\mathbb{C}_{\widehat{\chi}}$.

is the orthogonal projection $\widehat{s}$ of z onto $\mathbb{C}_{\widehat{\chi}}$ (a theorem in linear algebra says that the orthogonal projection gives the shortest distance of a point from a linear subspace; and the rays are linear subspaces), and is computed in the standard way as

$$\widehat{s} = \frac{\langle z, \widehat{\chi}\rangle}{\|\widehat{\chi}\|^2}\widehat{\chi}\,. \tag{3.9}$$

The real problem is to find $\widehat{\chi}$. With z fixed, a sufficient measure of parallelism of a ray $\mathbb{C}_{\chi(\theta)}$ and the vector z is the length of orthogonal projection of z onto χ; the ray is the more parallel or antiparallel, the longer the projection. We call this measure of parallelism the *match function*[2], MF. We have

$$\mathrm{MF}(\theta) = \frac{|\langle z, \chi(\theta)\rangle|}{\|\chi(\theta)\|}\,. \tag{3.10}$$

The notation indicates that the MF is a function of the parameter θ that determines the vector χ. Note that the MF does not depend on the scale of χ (χ and $a\,\chi$ give the same value of the MF). We need to maximize the MF to get the maximally parallel model vector $\widehat{\chi} = \chi(\widehat{\theta})$:

$$\widehat{\theta} = \arg\max_\theta \mathrm{MF}(\theta)\,. \tag{3.11}$$

How the maximum is computed in practice is discussed in section 3.4; basically, we perform an exhaustive search over a grid of values of θ.

[2] Intuitively, the more parallel two signal vectors (functions) are, the more they look alike, which is one reason for our nomenclature. A more serious reason is that "MF" also stands for matched filter. With velocity and acceleration fixed, so that MF is function of the range variable only, $R \to \langle z, \chi(R)\rangle$ amounts to ordinary filtering of z by the filter $h(t) = \chi(0)(t)$ which is matched to the transmitted signal. The MF is a generalization of this concept to more general kind of pattern matching.

The energy[3] W_y of any correctly sampled complex-valued signal $y(t)$ is

$$W_y = \int |y(t)|^2 dt = \tau_s \sum |y_n|^2 = \tau_s \|y\|^2 . \tag{3.12}$$

From Eq. (3.9)–(3.11), the energy $W_{\widehat{s}}$ of the signal estimate $\widehat{s}$ is

$$\frac{W_{\widehat{s}}}{\tau_s} = \|\widehat{s}\|^2 = \frac{|\langle z, \widehat{\chi}\rangle|^2}{\|\widehat{\chi}\|^2} = [\mathrm{MF}(\widehat{\theta})]^2 = \max \mathrm{MF}^2. \tag{3.13}$$

In all our data analysis we have used $W_{\widehat{s}}$ as the estimator of the signal energy W_s. However, as will be discuss in section 3.5, the estimator contains traces of the noise background, and is a biased estimator.

We summarize the match function method of parameter estimation

- Get the parameters θ by locating the position of MF maximum, Eq. (3.11).
- Get the signal energy as the square of the value of the MF maximum, Eq. (3.13).

According to Eq. (3.9), the complex amplitude b is be estimated as

$$\widehat{b} = \frac{\langle z, \widehat{\chi}\rangle}{\|\widehat{\chi}\|^2}, \tag{3.14}$$

but because the scale of the $\widehat{\chi}$ is arbitrary, so is the scale of $\widehat{b}$. Only the product $\widehat{s} = \widehat{b}\widehat{\chi}$ is well-defined.

A noise-free MF is useful for theoretical considerations. Without noise, both factors in the inner product in Eq. (3.10) are model functions. We will reserve a separate notation, AF, and use the standard name, ambiguity function [6], for the noise-free match function,

$$\mathrm{AF}(\theta_0; \theta) = \frac{|\langle \chi(\theta_0), \chi(\theta)\rangle|}{\|\chi(\theta)\|} . \tag{3.15}$$

In the MF method, target detection is based on the estimated signal energy $W_{\widehat{s}}$ exceeding a threshold. We have so far set the threshold, by visual inspection of the data, to be so high that there are only very few false alarms. In fact, we need to use a range-dependent threshold, such as shown in Fig. 4.4 on p. 64, because the lower altitudes, typically up to about 500 km, are often affected by strong clutter from the ionosphere, and need a higher threshold.

We set the detection threshold in terms of a dimensionless quantity, the ratio of signal energy to the noise power spectral density (PSD) G_γ. We call this ratio the energy-to-noise ratio, and denote it by $\mathrm{SNR_N}$,

$$\mathrm{SNR_N} = \frac{W_s}{G_\gamma} . \tag{3.16}$$

We assume that the system noise temperature T_{sys} is defined in such a way that the noise PSD density of complex-valued wide-band noise can be written as

$$G_\gamma = kT_{\mathrm{sys}} , \tag{3.17}$$

[3] We consider the signals $y(t)$ to have the dimension of voltage, and assume unit impedance, so that $|y|^2$ is the signal power. The impedance does not matter, for only signal ratios, like $|s|^2/|\gamma|^2$, are used when comparing to the physical world. We could also do without explicitly tracking the sampling interval, except that we do not want to change the dimension of energy in the *middle* of a chain of equations.

where k is the Boltzmann constant. The power of such a noise after being filtered with a boxcar-shaped low-pass filter that extends from frequency $-B/2$ to $B/2$ is

$$P_\gamma = kT_{\text{sys}}B. \tag{3.18}$$

We call B (rather than $B/2$, as is often done in the literature) the filter bandwidth. The digitally implemented filter in the SD receiver (see appendix A) is boxcar-shaped in time, not in frequency. It's impulse response is by design precisely matched to the sampling frequency. For such a filter, it can be shown that the noise-equivalent bandwidth B' is equal to the sampling frequency, so that, on the one hand, we have

$$B' = \frac{1}{\tau_{\text{s}}}\,, \tag{3.19}$$

and on the other hand, B' also satisfies Eq. (3.18), by definition of the noise equivalence. From Eq. (3.16)–(3.19) and (3.13) we get

$$\widehat{\text{SNR}_{\text{N}}} = \frac{W_{\widehat{s}}}{kT_{\text{sys}}} = \frac{W_{\widehat{s}}\cdot B'}{kT_{\text{sys}}\cdot B'} = \frac{(\max \text{MF}^2\cdot\tau_{\text{s}})\cdot(1/\tau_{\text{s}})}{P_\gamma} = \frac{\max \text{MF}^2}{P_\gamma}. \tag{3.20}$$

We treat the system temperature as a known, fixed radar parameter, and use Eq. (3.20) to find the signal energy in physical units. We use that estimate to find a lower limit, RCS_{min}, for the target's radar cross section (RCS). From the standard radar equation it follows

$$\text{RCS}(\phi) = \frac{(4\pi)^3\, kT_{\text{sys}}\cdot R^4\cdot \text{W}_{\text{s}}}{G(\phi)^2\cdot\lambda^2\cdot P_x\cdot\mathcal{D}T_c}\,. \tag{3.21}$$

Here R is target range, λ is radar wavelength, P_x transmission power, $\mathcal{D}$ transmission duty cycle so that $\mathcal{D}T_c$ is the actual length of transmission during the integration T_c. The factor $G(\phi)$ is the antenna power gain in the direction of the target within the radar beam, an angle ϕ offset from the known direction of the antenna optical axes. In the EISCAT system, it is normally not possible to find the offset angle. As a way of cataloguing the observed signal strength, we therefore normally quote RCS_{min}, which we get from Eq. (3.21) by setting $\phi = 0$,

$$\text{RCS}_{\text{min}} = \text{RCS}(0)\,. \tag{3.22}$$

We will argue in section 3.5 that the energy estimate $W_{\widehat{s}}$, defined in Eq. (3.13), is a biased estimate. In a typical situations $\text{E}W_{\widehat{s}}$ is larger than W_s by as much as about ten times the mean background noise power σ^2. The positive bias in the energy estimate means that we tend to overestimate the radar cross section when we use $W_{\widehat{s}}$ in Eq. (3.21) without correction. In the data analysis reported in chapter 5, we, nevertheless, have not applied any such correction.

3.2. Receiver effects

The MF method provides an estimate of the energy-to-noise ratio SNR_{N} for the signal and noise which have been processes through the receiver. But to estimate the target cross section, we need an estimate of the signal energy *before* the signal has been "corrupted" by the receiver. The noise that enters into the SNR_{N} estimate has come

through the same receiver path as the signal. Nevertheless, there is in general no guarantee that the relatively wide-band noise is attenuated by the same amount in the receiver as the signal, which typically consists of a few relatively narrowband frequency channels, transmitted cyclically.

In the standard EISCAT data analysis, the receiver effects are taken into account by incorporating the receiver impulse response[4] into the estimation theory, in the model functions to be fitted. We might ultimately need to do so for the space debris data analysis also. But in this paper, we try to keep the theory simpler by ignoring the receiver response in the model functions. The price is that we need to be prepared to correct the estimated energy. We handle the space debris receiver signal processing in more detail in appendix A, see also Fig. 3.2. It turns out that for our most common measuring configuration, the estimated energy of the processed signal (relative to noise power) is about 75% of the value in front of the receiver.

For small spherical targets the radar cross section, and therefore the received power, varies proportionally to the sixth power of the target diameter. Thus, a 30% underestimate of received power results only in 5% underestimate of the effective diameter. Considering other serious problems that we have in determining the targets' radar cross section, we have so far simply ignored the receive effect on energy estimate. The other problems include us not knowing the efficiency of the coherent integration (that is, how coherent the "coherent integration" actually is); the problem of not knowing the position of the target within the radar beam; and the problem of (not) knowing accurately the radar's transmitted power.

Ignoring the receiver impulse response is not expected to effect velocity estimate. As of the range estimate, the filter causes a "filter delay" of the order of the sampling interval (0.5–2 μs in our case), and this would in many systems shift the estimated range. But because our model functions are constructed from the actually measured transmission sample signal, and that signal travels though the same receiver path as the target echo, the filter delay should not be an issue.

3.3. The signal model

We will model the phase of the received signal $s(t)$ by assuming that the phase behaves as if the signal would reflect from a mirror that moves with constant radial acceleration a_0. We will assume that during an integration time T_c, which typically is about 300 ms, the amplitude b of the signal stays constant. Denoting by $x(t)$ the transmitted signal, and by t' the delayed time, with reference to Fig. 3.3 we take

$$s(t) = bx(t')\,. \tag{3.23}$$

For any given target radial motion $r(t)$, the delayed time for reflection from a point target is determined by

$$t - t' = \frac{2r(\frac{t'+t}{2})}{c}\,. \tag{3.24}$$

With constant radial acceleration, the target range is

$$r = r(R_0, v_0, a_0; t) = R_0 + v_0 t + \frac{1}{2}a_0 t^2\,. \tag{3.25}$$

[4] Normally, only the impulse response of the post-detection filters is incorporated. But in the SD receiver, the receiver bandwidth can be up to 25% of the width of the IF2 band, and then the IF2 band shape cannot be automatically ignored.

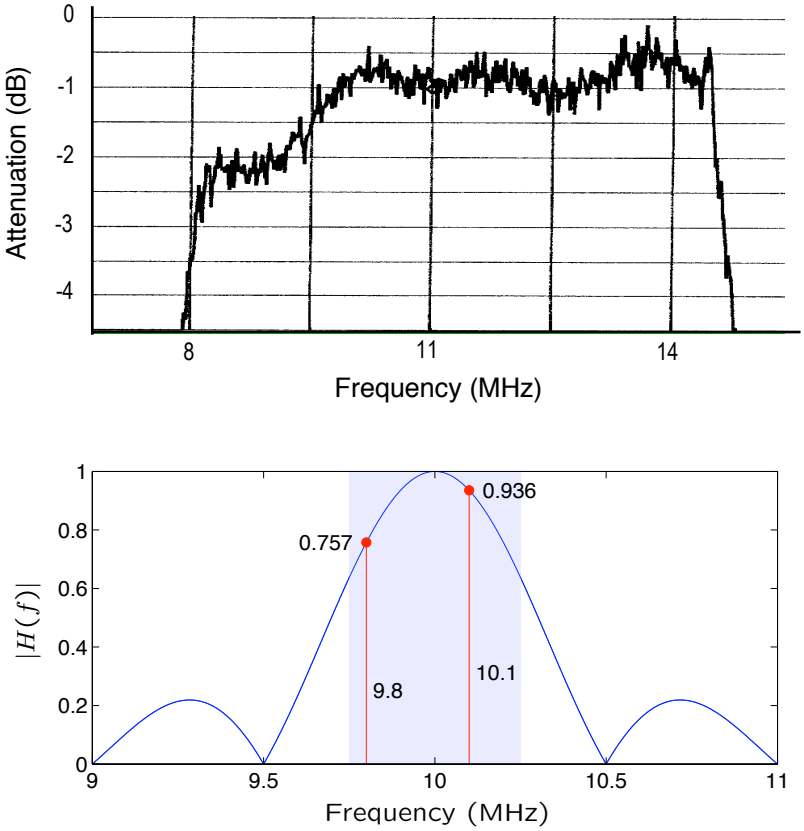

Figure 3.2.: Filtering in the SD receiver. The top panel shows the positive frequency component of the spectrum of EISCAT second IF analog band, essentially, the system's antialiazing filter shape (zero dB corresponds to signal level -50 dBm). This is also the power spectrum of analog system noise at this point. The IF2 band is sampled at 40 MHz as the first step in the SD receiver, and then "quadrature detected" and decimated to a 10 MHz complex signal, which has only a single spectral component. We ignore the band shaping in the Hilbert-transform/decimation step, and assume that the top panel also approximates the shape of the digital band, and digitized noise, at the 10 MHz sampling rate level. For the standard tau1 and tau2 experiments, the signal frequency channels are 300 kHz apart and map to 9.8 MHz and 10.1 MHz in IF2. Thus, 500 kHz sampling rate is more than enough to capture the information content of both of these channels simultaneously. This allows undersampling the 10 MHz data stream to a $\pm$250 kHz baseband, by decimating the stream by a factor of 20. The decimation is done by averaging over 20 samples, then taking next the 20 samples, and so on. The averaging amounts to filtering by a digital filter that has the transfer function shown in the bottom panel. The digital filter attenuates the two frequencies (unevenly), so that the energy of the measured signal is only about $(0.757^2 + 0.936^2)/(1^2 + 1^2) = 75\%$ of the energy at filter input. The noise-equivalent bandwidth of the filter is equal to 500 kHz, the final sampling rate.

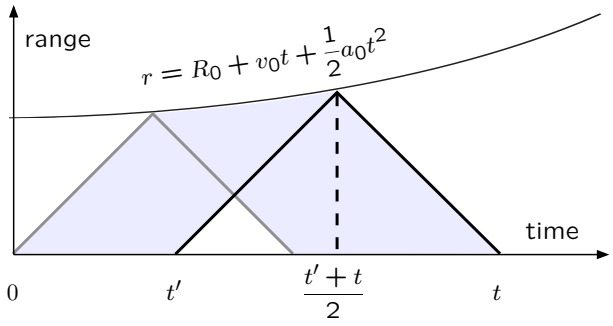

Figure 3.3.: Transmitted wave reflected from a point-like target which is moving with constant radial acceleration a_0. The parabola shows the radial component of the target's position vector in the coordinate frame of a stationary radar antenna, during the few hundred milliseconds of a coherent integration. The full three-dimensional velocity vector typically is very nearly constant in that frame during the integration time. The integration starts at time 0 with the transmission of the first pulse belonging to the integration. At the start of the integration, target range is R_0 and radial velocity is v_0. Note that the diagram is not drawn to scale.

For the motion (3.25), Eq. (3.24) is quadratic in t'. The solution of the equation for the pulse propagation time $t - t'$, with an appropriate choice of the sign of the square root, is

$$t - t' = \frac{2c}{a_0} \left\{ 1 + \frac{v_0}{c} + \frac{a_0}{c} t - \left[1 + \frac{2v_0}{c} + \left(\frac{v_0}{c} \right)^2 + \frac{2a_0}{c} (t - \frac{R_0}{c}) \right]^{\frac{1}{2}} \right\} . \tag{3.26}$$

Equation (3.26) can be simplified by expanding the square root to a power series. Care must be exercised regarding to which terms can be dropped from the expansion. With parameter values that are typical at EISCAT UHF when antenna is pointed nearly vertically,

$$\begin{aligned} R_0 &\approx 10^6 \text{ m}, \\ v_0 &\approx 10^3 \text{ m}s^{-1}, \\ a_0 &\approx 10^2 \text{ ms}^{-2}, \\ \omega_1 &\approx 6 \cdot 10^9 \text{ Hz}, \end{aligned}$$

all terms following the "1" inside the square brackets in Eq. (3.26) are quite small compared to unity. But what actually determines which terms X can be dropped is the requirement that the corresponding phase factor $\omega_1 \frac{2c}{a_0} X$, where ω_1 is the radar transmission frequency, stays very small during the integration time. Using the first three terms of the power series expansion of $[1 + (\ldots)]^{\frac{1}{2}}$, and then dropping all the individual terms for which the corresponding phase factor is less than 0.1 rad when integration time is less than a second, we are left with

$$t - t' \approx \frac{2}{c} \left[R_0 + v_0 t + \frac{1}{2} a_0 t^2 - (v_0 + a_0 t) \frac{R_0}{c} \right] \tag{3.27}$$

$$= \frac{2}{c}\left[R_0 + v_0(t - \frac{R_0}{c}) + \frac{1}{2}a_0(t - \frac{R_0}{c})^2\right] \tag{3.28}$$

$$= \frac{2}{c}r(t - \frac{R_0}{c})\,. \tag{3.29}$$

The term $\frac{-R_0}{c}$ is a natural first order correction to the time instant of pulse reflection; the only non-trivial aspect is that this correction already is sufficient (for our typical measuring configurations). Thus, the model functions $\chi(R, v, a; t)$ to be used in the MF computation, Eq. (3.10), are of the form

$$\chi(R, v, a; t) = x(t - \frac{2}{c}\, r(R, v, a;\, t - \frac{R}{c}))\,. \tag{3.30}$$

Note that *nothing* has been assumed about the transmission $x(t)$ in this derivation so far. In principle, as long as the transmission can be accurately measured via the transmission sample signal, we do not even need (ever) to know what transmission has been used; the MF machinery incorporates the transmission transparently. This is a good thing for automated piggy-back measurements, where we do not have any control on the transmission EISCAT might be using at any given time.

The reality, of course, is rather different. A basic problem is that the radar's noise-environment is often poorly approximated by our assumption that it consists only of stationary gaussian noise. Distortions happen in practice, one of them being that the ionosphere often becomes visible in our data (Fig. 4.4). More or less ad hoc, manual, experiment-specific solutions are used to counter these problems. Also, we cannot at the moment at all handle the case that the antenna pointing may change during a measurement; but many EISCAT measurements use cyclical antenna pointing schemes. In practice, both now and into the foreseeable future, we need to know a priori, and even select, the EISCAT measurements that we are making use of in the SD work.

3.4. Computational aspects

Here we derive the approximation for the signal model that we have been using in our work so far. Assume that transmission can be described as

$$x(t) = \epsilon(t)\mathrm{e}^{i\omega_1 t}\,, \tag{3.31}$$

where ω_1 is the carrier frequency, and the transmission envelope $\epsilon(t)$ is a slowly changing function, describing, say, a binary phase modulation, as is often the case in EISCAT. This description is good for a single-frequency-channel transmission. We drop the correction $-R/c$ to the pulse reflection time in Eq. (3.30), and use the special form Eq. (3.31) of transmission to write the model function as

$$\chi(t) = \epsilon(t - \frac{2}{c}r(t))\, e^{i\omega_1[t - \frac{2}{c}r(t)]}\,. \tag{3.32}$$

Inside the slowly varying transmission envelope, we can assume $r(t)$ stays constant, $r(t) = R$, during the integration time. Then, from Eq. (3.32) and Eq. (3.31),

$$\begin{aligned}\chi(t) &= \epsilon(t - \frac{2R}{c})\,\mathrm{e}^{i\omega_1(t - \frac{2}{c}R)}\,\mathrm{e}^{i[(-\omega_1\frac{2}{c}v)t + (-\omega_1\frac{a}{c})t^2]} \\ &= x(t - \frac{2R}{c})\,\mathrm{e}^{i(\omega_{\mathrm{D}}t + \alpha_{\mathrm{D}}t^2)}\,,\end{aligned} \tag{3.33}$$

where $\omega_{\mathrm{D}} = -\omega_1 \frac{2v}{c}$ and $\alpha_{\mathrm{D}} = -\omega_1 \frac{a}{c}$ are the Doppler-frequency and the rate of change of the Doppler-frequency, which we call the Doppler-drift. The approximation (3.33) is often used in the literature (usually without the drift term), and is described by saying that the received signal is a delayed-in-time, Doppler-shifted replica of the transmission. With the model (3.33), the match function definition in Eq. (3.10) can be expanded for continuous-time signals as

$$\mathrm{MF}(R,v,a) = \frac{|\int_0^{T_c} z(t)\overline{x}(t-\frac{2R}{c})\mathrm{e}^{-i(\omega_{\mathrm{D}}t+\alpha_{\mathrm{D}}t^2)}\,dt|}{\sqrt{W_{\mathrm{x}}}}\,, \tag{3.34}$$

where $W_{\mathrm{x}} = \int |x(t)|^2 dt$ is the energy of the transmission sample signal.

For signal vectors, we need to take into account that the transmission samples x_n are only available at times $n\tau_{\mathrm{s}}$. This already forces us to discretize the range variable. With

$$R_j = j \cdot \frac{c\tau_{\mathrm{s}}}{2} \tag{3.35}$$

the match function becomes

$$\mathrm{MF}(R_j,v,a) = \frac{|\sum_{n=0}^{N-1} z_n \overline{x}_{n-j}\mathrm{e}^{-i(\omega_{\mathrm{d}}n+\alpha_{\mathrm{d}}n^2)}|}{\|x\|}\,, \tag{3.36}$$

where the normalized Doppler-shift and Doppler-drift are

$$\omega_{\mathrm{d}} = -\omega_1\tau_{\mathrm{s}}\frac{2v}{c}\,, \tag{3.37}$$

$$\alpha_{\mathrm{d}} = -\omega_1\tau_{\mathrm{s}}\frac{a\tau_{\mathrm{s}}}{c}\,. \tag{3.38}$$

At the points

$$v_k = k\frac{2\pi c}{\omega_1 T_{\mathrm{c}}} \tag{3.39}$$

Eq. (3.36) can be written as

$$\mathrm{MF}(R_j,v_k,a) = \frac{|\sum_{n=0}^{N-1}(z_n\overline{x}_{n-j}\mathrm{e}^{-i\alpha_{\mathrm{d}}n^2})\mathrm{e}^{-i\frac{2\pi kn}{N}}|}{\|x\|}\,, \tag{3.40}$$

which shows that at these points the MF can be evaluated using FFT. The denominator $\|x\|$ is the square root of transmission sample energy, and is (of course) independent of R, v and a. The MF computation via Eq. (3.40) is summarized in Fig. C.1 on p. 116.

In most of our data analysis, we have taken the radial acceleration to be a deterministic function of range, $a = a(R)$. We have used the acceleration that corresponds to target being on circular orbit and the antenna being pointed vertical,

$$a = g_0 \cdot \frac{R_{\mathrm{E}}}{h} \cdot (\frac{R_{\mathrm{E}}}{R_{\mathrm{E}}+h})^2, \tag{3.41}$$

where R_{E} is the Earth radius 6360 km, g_0 is the acceleration of gravity at zero altitude, 9.8 m s^{-2}, and h is the target altitude. Experimentation with data has shown that not much sensitivity is lost in practice even if the acceleration is not varied.[5]

[5] Which is perhaps a bad thing, because a priori, we would expect the MF to be rather sensitive to the acceleration being correct. For instance, we show in Fig. 3.5 on p. 38 that with 0.3 s integration time, an 10 m s^{-2} error in a should cause the coherently integrated amplitude to drop by about 50% from the ideal case. That this appears not to happen when we vary the acceleration in the analysis of real data, may indicate that there are other factors that are causing the signal model to be seriously non-ideal to begin with.

In our routine analysis therefore, we search the MF maximum only over the (R_j, v_k)-grid. Even then, the detection computation, using full resolution and without any further approximation, becomes too large. Assume we want to cover 1000 km in range and use 0.3 s coherent integration. Assume that the sampling interval is 0.5 μs. Then the input data vector is 600,000 points long, and the FFT requires about 60 million floating-point operations. The $1000/0.075 \approx 13,000$ range gates require about 800×10^9 floating-point operations. On a dual-processor 2 GHz G5 Mac, we get about 1 GFlops combined speed in FFT of this length, so will need about 800 s to handle the 0.3 s of data. Normal EISCAT UHF phase-coded transmission uses baud length of about 20 μs or more. For these modulations, we can safely relax the range gate separation in the detection phase somewhat, say by a factor of 10 (this also ensures sufficient Doppler-frequency coverage). But this still leaves us more than two orders of magnitude short of real-time speed.

3.5. Bias in the energy estimate

In signal energy estimation, there is an additional worry besides the receiver response. By design, the MF method gives the most probable value $\widehat{s}$ of the signal vector, given the measured vector z. The $\widehat{s}$ is an unbiased estimator of the signal, that is, if we could repeat the measurement so that the actual signal vector always is s_0, due to noise we would get different observed signals z, but the mean of the estimates computed from those z via the MF method would be s_0. But the situation with the estimate of the signal energy is not as good. We are using as the energy estimate the energy $W_{\widehat{s}} = \|\widehat{s}\|^2$ of $\widehat{s}$. This choice seems natural enough, and the estimate is readily available (once $\widehat{s}$ is found). However, in general, the estimate does not provide the most probable value for signal energy among those signals that are compatible with the measured vector z. Neither is it an unbiased estimate, in repeated trials with the same signal s_0, the mean of the estimated energy would not be W_{s_0}. The mean of the estimated energy would be W_{s_0} plus some "background" value.

Unfortunately, we have been unable to decide what if anything we should use as the "background" here. Normally, when radar people compute signal power, or power spectrum, the practice is to subtract from the power (-spectrum) the corresponding mean background power (-spectrum), computed from a segment of data where it is known a priori that no signal is present. But it is not apparent to us that such an approach is relevant here. The coherent integration does not quite correspond to the typical repeated-trial radar measurement where one averages over several measurements of identically repeated radar pulses. And even if we could imagine that our MF scheme were some kind of repeated trial, the MF method involves maximization over noise-contaminated quantities, and it is not clear what we should use as the "mean background" then. If, in the computation of the MF maximum, we would know range and velocity a priori, and so would *not* need to maximize MF over those variables, the bias could be removed simply by subtracting unity from our standard $\mathrm{SNR_N}$ estimate. This is seen as follows.

Take, for definiteness, the signal model functions $\chi(R, \omega; t)$ to be of the often-used form of delayed-in-time, Doppler-shifted-in-frequency replica of the transmission $x(t)$,

$$\chi(t) = x(t - \frac{2R}{c})\mathrm{e}^{i\omega t}\,, \tag{3.42}$$

and let the noise by $\gamma(t)$. Assume that the target Doppler-shift is known to be ω_0 and

range to be R_0, so that the echo signal is

$$s_0(t) = b_0\,\chi(R_0, \omega_0; t) \equiv \chi_0(t)\,. \tag{3.43}$$

In the MF method we now need only to determine the signal amplitude b_0. The best model function needs not to be searched for over R and ω, but can immediately be nailed down as $\widehat{\chi} = \chi_0$. The standard MF method signal estimate then becomes

$$\widehat{s} = \frac{\langle b_0\chi_0 + \gamma, \chi_0\rangle}{\|\chi_0\|^2}\chi_0. \tag{3.44}$$

Taking norm squared and then expectation value of Eq. (3.44), and assuming that noise over repeated trials has zero mean, $\mathrm{E}\gamma = 0$, gives

$$\mathrm{E}(\|\widehat{s}\|^2) = \|b_0\chi_0\|^2 + \frac{\mathrm{E}|\langle\gamma, \chi_0\rangle|^2}{\|\chi_0\|^2}\,. \tag{3.45}$$

By expanding the inner product in the denominator of the second term in terms of the noise samples γ_n and model samples $x_{0,n}$ and using the premise that the noise samples are uncorrelated and have variance σ^2, so that

$$\mathrm{E}\,(\gamma_n\overline{\gamma_{n'}}) = \delta_{n,n'}\cdot\sigma^2\,, \tag{3.46}$$

we get from Eq. (3.45)

$$\mathrm{E}(\|\widehat{s}\|^2) = \|s_0\|^2 + \sigma^2\,. \tag{3.47}$$

This shows that there is a bias equal to the mean noise power in the signal energy estimate in this case, and thus the energy-to-noise ratio $\mathrm{SNR_N}$, estimated by $\|\widehat{s}\|^2/\sigma^2$, has the bias of unity.

However, normally we do not know the correct model signal a priori, but base the energy estimate to

$$\|\widehat{s}\|^2 = \max_{\omega,R}\frac{|\langle b_0\chi_0 + \gamma, \chi(R,\omega)\rangle|^2}{\|\chi_0\|^2} \tag{3.48}$$

(we used χ_0 in the nominator instead of $\chi(R,\omega)$, for the norm does not depend on R and ω for the model functions (3.42)). It is not clear what the bias is in this case; there seems to be no immediate way to evaluate $\mathrm{E}\|\widehat{s}\|^2$. However, for the case of no signal, we can argue that the bias must be (considerably) larger than the σ^2 of Eq. (3.47). With $\chi_0 = 0$, we get from Eq. (3.48)

$$\mathrm{E}(\|\widehat{s}\|^2) = \mathrm{E}\max_{\omega,R}\frac{|\langle\gamma, \chi(R,\omega)\rangle|^2}{\|\chi_0\|^2}\,. \tag{3.49}$$

That is, we first, for a given trial noise γ, pick the maximum value of the noise-only, two-parameter MF, and then we form the expectation by averaging over those values. We saw in Eq. (3.47) that at any fixed point (ω, R), MF^2 has expectation σ^2. That implies that from trial to trial, MF^2 must get values both smaller and larger than σ^2. In each trial, $\mathrm{MF}(R,\omega)$ is estimated on several tens of thousands of points, and will *somewhere* have values that are considerably larger than σ^2.

Figure 3.4 shows $\mathrm{MF}(R,\omega)$, for a fixed R, as function of ω for typical data. On this single slice already, the highest noise peak has value of about 3. (There is actual signal visible in the data in Fig. 3.4, but its effects are local and do not affect the noise

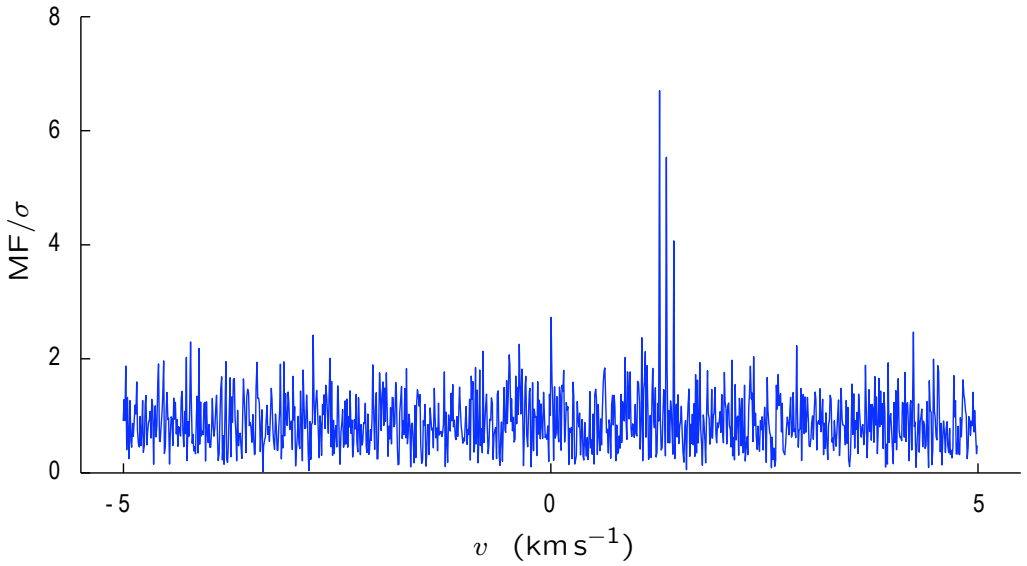

Figure 3.4.: The problem of background subtraction. The figure shows the velocity slice $v \mapsto \mathrm{MF}(R, v)$ through the range $R = \widehat{R}$ that maximizes the MF. The slice is essentially the noisy power spectral density of the tau2 transmission envelope over about 30 IPPs. The noise background has spectral density of about σ^2 (the mean of MF/σ is 0.89), while the maximum value of the background, away from the immediate neighborhood of the three signal peaks, is about 3, and is approached in several positions. The spectral peak value, $\mathrm{MF}/\sigma \approx 7$, equals to $\sqrt{W_{\widehat{s}}}$, the square root of the energy of the Bayesian signal estimate $\widehat{s}$. We use $W_{\widehat{s}}$ unmodified as the signal energy estimate $\widehat{W_s}$. The question is, how much does the noise affect the peak value, and should we make some kind of background subtraction in order to get an unbiased energy estimate?

background further away.) Note that the mean noise level is about σ^2 as expected. Therefore, typically $\max \mathrm{MF}^2$ will be about $9\sigma^2$ or even slightly larger. So we seem to have $\mathrm{E}\|\widehat{s}\|^2 \approx 10\sigma^2$ when there, actually, is no signal.

Should we start subtracting $10\sigma^2$ from $\max \mathrm{MF}^2$ in our energy estimates? We don't know. We would also like to point out that when doing incoherent pulse-to-pulse integration (see next section), we do not seem to have this kind of trouble. We have then a clear-cut repetitive-trial experiment arrangement, and it seems to make better sense to form *first* the expectation value of the MF^2 at every point as in Eq. (3.48), and *then* pick the maximum from the averaged MF. It seems reasonable to make a background subtraction by σ^2 (= the background spectral density) in that case. On the other hand, as will also be argued in the next section, when there is only a single transmission pulse to be integrated, the coherent and non-coherent integration are identical operations. What is the correct "background subtraction" then?

3.6. Coherent integration

We have been saying that the MF method of detection and parameter estimation, the recipe

$$\widehat{\theta} = \arg\max_{\theta} \mathrm{MF}(\theta)\,, \tag{3.50}$$

accomplishes "coherent integration" of the signal (θ refers to the parameters of the signal model, $\theta = (R, v, a)$). Considering that our MF, and the entity that people are calling the radar ambiguity function, are pretty much the same thing, it is odd that the books we have consulted don't mention coherent integration in connection of the ambiguity function. Typically, we integrate the complex-valued signal $z(t)$ over a few tens of consecutive transmission-reception (T/R) cycles. In an EISCAT measurement, a single T/R cycle has length $T_{\mathrm{rep}} \approx 6$–$10$ ms. Typically we use integration time $T_c \approx 300$ ms.

Before elaborating on coherent integration, we explain what we mean by signal phase coherence (another concept often carelessly defined). In the simplest form of the concept, a complex signal

$$s(t) = a(t)\mathrm{e}^{i\phi(t)}, \text{ where } a(t) \geq 0, \tag{3.51}$$

is said to be phase coherent if it consists of segments of a single sinusoid $\mathrm{e}^{i\omega t}$. The essential aspect here for our purposes is not that the phase can be accurately described by the particularly simple function $\phi(t) = \omega t + \phi_0$, but rather, that the phase can be described accurately and deterministically by *some* parametrized function, during some time interval, called the signal coherence time. For times substantially longer than the coherence time, the actual phase deviates from the model phase by larger, and typically ultimately random, amounts. Viewed this way, the coherence of a signal is always relative to some reference—either a computed model, or an actual signal like an oscillator signal for example—so that a signal may be non-coherent with respect to some model, but can still be coherent with a more accurate model.

In its simplest form, coherent integration of pulsed radar signal has meant just adding up the signals z_m,

$$z_m(t) = z(t - mT_{\mathrm{rep}}), \quad t \in [0, T_{\mathrm{rep}}], \tag{3.52}$$

$$= s_m(t) + \gamma_m(t), \tag{3.53}$$

from M successive T/R cycles, to form the integrated signal

$$w_{\mathrm{coh}}(t) = \sum_{m=1}^{M} z_m(t). \tag{3.54}$$

To make sense, coherent integration in the basic form (3.54) requires that the signals $s_m(t)$ in Eq. (3.53) are identical. In practice this means that the transmission must be identical from pulse to pulse. It also means that the target must be stationary so that there is no Doppler-shift, because a non-zero Doppler-shift ω_D would introduce unequal phase factors $\mathrm{e}^{i\omega_D m T_{\mathrm{rep}}}$ to the z_m's. In addition, all the oscillators used in the radar must be well-behaved so that the reception itself does not introduce extra phase shifts. Neither of the first two conditions is satisfied in EISCAT measurements; a more sophisticated approach is required to implement coherent integration from pulse to pulse in real life. Such an approach is provided by the MF method, as we will show shortly. But first we will discuss the general benefit of coherent integration. We will also recall the standard method of using range-gated power spectra for detection, so that we can compare it to the MF method.

When the coherent integration (3.54) can be used, the benefit is that, for white noise γ, the signal-to-noise ratio in $w_{\mathrm{coh}}(t)$ is M-times larger than the signal-to-noise ratio in an individual $z_m(t)$. The alternative to coherent integration is non-coherent integration, achieved by computing

$$w_{\mathrm{noncoh}}(t) = \sum^{M} |z_m(t)|^2. \tag{3.55}$$

Non-coherent integration does not improve signal-to-noise ratio, but it still increases threshold detection sensitivity, by reducing the variance in power ($\sigma^2[w_{\mathrm{noncoh}}] < \sigma^2[|z_m|^2]$), so that a given threshold will result in fewer false alarms.

Whether one uses coherent or non-coherent integration, or no integration at all, a simple detection criterion is to use envelope detection. In envelope detection, one searches for the power-domain envelope $|x(t-2R/c)|^2$ of the transmitted waveform $x(t)$ in the measured power-domain signal such as $|z_n(t)|^2$, $|w_{\mathrm{coh}}(t)|^2$ or $w_{\mathrm{noncoh}}(t)$. The pattern matching may be done by cross-correlation, so that the detection criterion could be

$$\max_R \langle\, |w_{\mathrm{coh}}(t)|^2, |x(t-2R/c)|^2 \,\rangle > \text{Threshold}\,. \tag{3.56}$$

The advantage in working in the power domain is that the signal needs not to be phase coherent in any way, not even within a single pulse, because *any* phase factor $\mathrm{e}^{i\phi(t)}$ in the signal will just yield unity. However, for phase coherent signals, like the Doppler-shifted replica of transmission, which has phase factor $\mathrm{e}^{i\omega_D t}$, the standard way to increase detection sensitivity (and to find the value of the Doppler-shift), is to compute the range-gated power spectrum $G(R,\omega)$. For a single (non-coded) pulse z_m of length L the power spectrum is

$$G(R,\omega) = \frac{1}{L}\Big| \int_{2R/c}^{2R/c+L} z_m(t)\mathrm{e}^{-i\omega t}\, dt\Big|^2\,. \tag{3.57}$$

The spectrum may in practice also be computed via the signal autocorrelation function, as is normally done in EISCAT. The detection criterion is

$$\max_{R,\omega} G(R,\omega) > \text{Threshold}. \tag{3.58}$$

For a signal that has small bandwidth $B_\mathrm{s} \approx 1/L$ compared to the noise bandwidth B_γ, performing the detection via power spectral density instead via total power can increase the signal-to-noise ratio substantially. The increase is roughly by the factor B_γ/B_s relative to $\mathrm{SNR} = P_\mathrm{s}/P_\gamma$, where P_s and P_γ are the total signal and noise power, respectively. The increase comes about because the total signal power W_s/L, where W_s is energy of the pulse of length L, is condensed into a spectral band that is narrower than the noise band by this factor. Assuming the noise power spectral density G_γ to be roughly constant with value kT_sys, and denoting by G_s the power spectral density of the pulse, we can write the above estimate of the peak signal-to-noise ratio as

$$\frac{\max G_\mathrm{s}}{G_\gamma} \approx \frac{B_\gamma}{B_\mathrm{s}} \cdot \frac{P_\gamma}{P_\mathrm{s}} \approx \frac{B_\gamma}{1/L} \cdot \frac{E_\mathrm{s}/L}{kT_\mathrm{sys}B_\gamma} = \frac{E_\mathrm{s}}{kT_\mathrm{sys}}\,. \tag{3.59}$$

For signal correctly sampled with sampling interval τ_s, we can write Eq. (3.59) also in the form

$$\frac{\max G_\mathrm{s}}{G_\gamma} \approx \frac{B_\gamma}{B_\mathrm{s}} \cdot \mathrm{SNR} \approx \frac{(1/\tau_\mathrm{s})}{(1/L)} \cdot \mathrm{SNR} = N_L \cdot \mathrm{SNR}, \tag{3.60}$$

where N_L is the number of samples taken from the pulse L. The N_L-fold increase of the signal-to-noise ratio, over the signal-to-noise ratio of any typical single sample, signifies coherent integration of the individual samples, and is consistent with our underlying assumption of signal phase coherence.

We now verify that, for a single non-coded pulse, the square of the match function $\mathrm{MF}(R,\omega)$ reduces to the range-gated power spectrum as defined in Eq. (3.57). For such a pulse, the model function (ignoring the acceleration term) is

$$\chi(R,\omega;t) = x(t - 2R/c)\mathrm{e}^{i\omega t}, \tag{3.61}$$

where $x(t)$ represents the transmitted pulse,

$$x(t) = \begin{cases} 1 & \text{when } t \in [0,L] \\ 0 & \text{otherwise}\,. \end{cases} \tag{3.62}$$

Thus

$$\begin{aligned} \mathrm{MF}(R,\omega)^2 &= \frac{|\langle z, \chi(R,\omega)\rangle|^2}{\|\chi(R,\omega)\|^2} && (3.63)\\ &= \frac{\int z(t)\overline{x(t-2R/c)\mathrm{e}^{i\omega t}}\,dt}{\int |x(t-2R/c)|^2\,dt} && (3.64)\\ &= \frac{|\int_{2R/c}^{2R/c+L} z(t)\mathrm{e}^{-i\omega t}\,dt|^2}{L} && (3.65)\\ &= G(R,\omega)\,. && (3.66) \end{aligned}$$

For a single non-coded pulse, the MF method of detection and parameter estimation is identical to the standard method of maximization of the range-gated power spectrum; especially, both methods achieve coherent integration within the single pulse. But the MF method is more general as a tool for implementing coherent integration, for a large variety of radar experiments.

Both methods achieve correct coherent integration of Doppler-shifted signals within a single T/R period. But we cannot use the range-gated power spectrum method unmodified to integrate coherently over multiple T/R periods, if the signal has non-zero Doppler-shift. There is no degree-of-freedom to take into account the unequal phase factors

$e^{i\omega_D m T_{\rm rep}}$ that would appear in the Fourier-transforms. Those time-independent phase factors are not a problem for non-coherent integration, though, and the range-gated power-spectra computed from individual pulses can be integrated. Pulse-to-pulse coherent integration is built-in into the MF method. Essentially, the troublesome Doppler phase factors are removed by a the correctly matching model function—the one that maximizes the MF—so that only signal with zero Doppler-shift remains to be integrated.

Coherent pulse-to-pulse integration of M pulses increases the effective signal-to-noise ratio by the factor M over the single pulse case (whether that case is handled by the MF method or the range-gated power spectrum method), while the non-coherent pulse-to-pulse integration by the power-spectrum method does not increase the signal-to-noise ratio, but improves the detection by reducing the variance in the integrated spectrum $G(R, \omega)$. On the other hand, the MF method requires that the signal model stays accurate over most of the integration time. It is obviously easier to keep a signal model accurate for a shorter time than for a longer, so a combination of pulse-to-pulse coherent integration and non-coherent integration seems a good way to increase detection sensitivity.

The effect of not having a correct signal model in coherent integration can be studied quantitatively using the ambiguity function. Figure 3.5 shows how much the peak value of the AF is reduced from the ideal, fully coherently integrated value if the acceleration used in the model signal is not correct. For instance, the top panel shows the situation when the integration time is 300 ms. In the panel, we plot three velocity ambiguity functions, shown by the grey, red, and blue curves. By velocity ambiguity function we refer to the velocity slice $v \mapsto \mathrm{AF}(R_0, v_0, a_0; R, v, a)$ of the three-parameter AF. The constants R_0, v_0 and a_0 are the actual target parameters, the variables R, v and a are the parameters of the model function. In the velocity slice, we take $R = R_0$ and take a to be a constant that differs by the indicated amount from a_0.

To compute the AF, we use the standard approximation for the model functions,

$$\chi(R, v, a; t) = x(t - 2R/c)e^{i\omega t}e^{i\alpha t^2} \tag{3.67}$$

where $\omega = -2\pi\frac{v}{\lambda/2}$ and $\alpha = -\pi\frac{a}{\lambda/2}$. We assume single-frequency EISCAT tau2 transmission with unit magnitude, $|x(t)| = 1$, and wavelength λ. With Eq. (3.67), the AF in continuous-time version is then

$$\begin{aligned} \mathrm{AF}(R_0, v_0, a_0; R_0, v, a) &= \frac{|\langle \chi_0, \chi \rangle|}{\|\chi\|} \\ &= \frac{1}{\sqrt{T_c}} \int_0^{T_c} dt\, |x(t - \frac{2R_0}{c})|^2 e^{i(\omega_0 - \omega)t} e^{i(\alpha_0 - \alpha)t^2}. \end{aligned} \tag{3.68}$$

As is also indicated in the Fig. 3.5, the AF depends only on the differences $a - a_0$ and $v - v_0$ in this case.

In the single-frequency tau2 transmission, the pulse length is 576 μs and the interpulse period P is 5580 μs. The distance Δv between the main peaks of the ambiguity function corresponds to the pulse repetition frequency, $\Delta v = (\lambda/2)(1/P) = 28.9\ \mathrm{m\,s^{-1}}$ in this case. The ambiguity function has its absolute maximum at $v - v_0 = 0$, $a - a_0 = 0$. The figure shows also two of the side maxima, of value 0.98 and 0.93 times the absolute maximum. In the presence of noise, there is the possibility that the MF gets distorted (as compared to the AF) so that the maximum shifts to the wrong position, by an integer multiple of Δv. This is the Doppler part of the range-Doppler ambiguity.

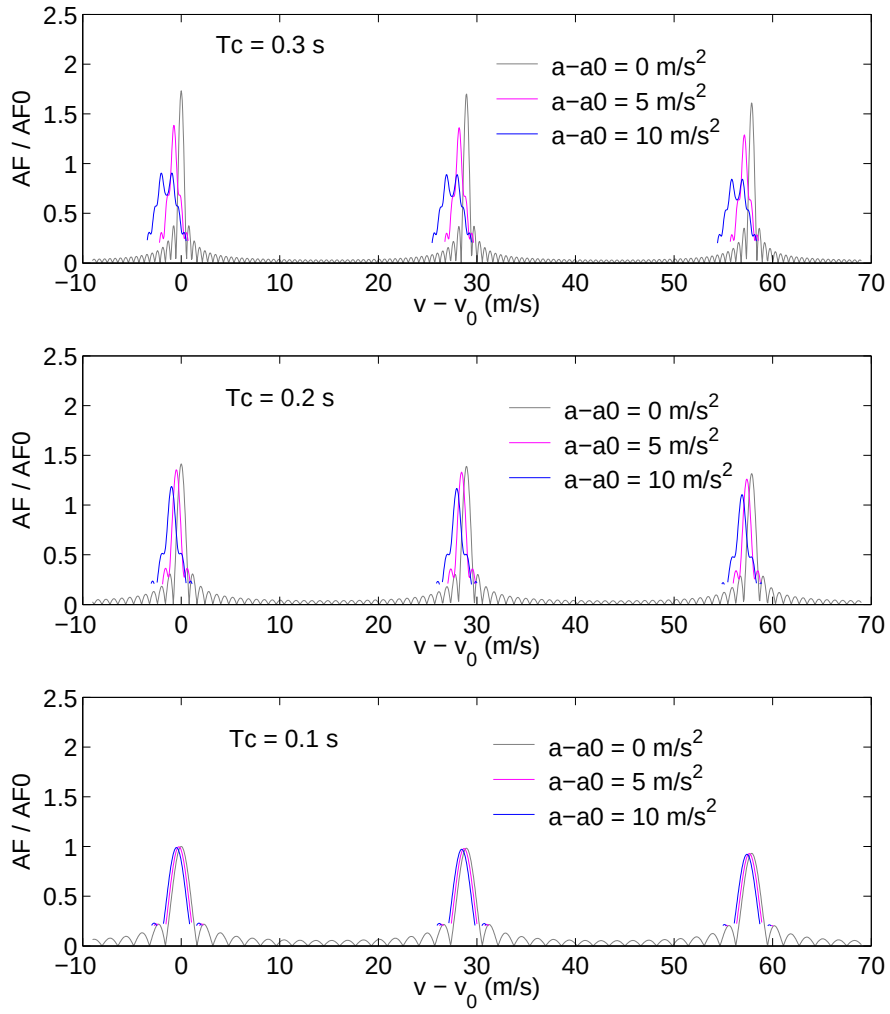

Figure 3.5.: Effect of an acceleration error to the signal amplitude estimate. The plotted quantity is the ambiguity function $v \mapsto \mathrm{AF}(a_0, v_0; a, v; T_\mathrm{c})$, normalized by $\mathrm{AF}_0 = \mathrm{AF}(a_0, v_0; a_0, v_0; 0.1\mathrm{s})$. We assume here tau2 transmission patterns, but have set the two tau2 frequencies to the same value. The top panel shows that with 0.3 s integration T_c, if the radial acceleration a used in the integration differs by 10 $\mathrm{m\,s^{-2}}$ from the actual acceleration a_0, almost 50% of the signal amplitude is lost; in fact, the signal amplitude is less than what it would be with only 0.1 s integration. Note that in the top and middle panels the best matching velocity, $\widehat{v} = \max_v \mathrm{AF}$, has shifted slightly towards negative values from the correct value $\widehat{v} = v_0$, to partially compensate for the wrong value of the acceleration. This shows the non-surprising fact that in the MF method, acceleration and velocity cannot be determined independently of each other.

It is not important for our final results, but it is still interesting to note how this distortion actually shows up. The initial intuition when seeing a noise-free plot like Fig. 3.5, which consists of almost equal-sized peaks, might be that when noise enters the picture, it would directly and randomly add to the spikes, and so a wrong maximum could be selected when the MF maximum is sought for. In fact, this is *not* what happens (until the noise comes quite large). One must remember that the figure shows the signal spectrum, not the time-domain signal; then it is not a priori so clear how the noise should affect the curves. What happens is more as if the whole spectrum, while keeping its original smooth shape, would shift by an integer number of Δv. When there is noise, those MF which are in the "correct place", do not look any smoother or "AF-like" than those MF that are shifted by $n \times \Delta v$. Of course, this only underlines the fact that the Doppler-ambiguity is a real ambiguity that reflects real loss of information, caused by pulsing the radar transmission in a (too) regular way.

What is more important for sensitivity is how much a wrongly assigned acceleration parameter affects the maximum amplitude. (Remember that we do not perform a search over acceleration values, but use a deterministic, range-dependent guess, appropriate for circular orbits and vertical antenna pointing.) All curves in Fig. 3.5 are normalized by a the velocity slice that corresponds to 0.1 s integration time and no acceleration error. The maximum value equals to the square root of the total signal energy, and (hence) increases as $\sqrt{T_\mathrm{c}}$. For example, the main maximum of the $T_\mathrm{c} = 0.3\,\mathrm{s}$-curve is 1.7 times larger than the the main maximum of the $T_\mathrm{c} = 0.1\,\mathrm{s}$ curve. The red and blue curves show how much sensitivity is lost if the acceleration is not correct. With 0.1 s integration, a small error such as $10\ \mathrm{m\,s^{-2}}$ does not matter. But with 0.3 s integration, almost 50% of the peak amplitude is lost if acceleration is wrong by $10\ \mathrm{m\,s^{-2}}$.

The effect of the acceleration error to the peak amplitude is illustrated in even more detail in Fig. 3.6. There we plot the acceleration ambiguity function, that is, the slice

$$a \mapsto \mathrm{AF}(R_0, v_0, a_0; R_0, v_0, a) \tag{3.69}$$

for four different lengths of the coherent integration, ranging from 0.1 s to 0.5 s. The slices are computed from Eq. (3.68) by setting $R_0 = 0$ and $\omega = \omega_0$. According to Fig. 3.6, a $10\ \mathrm{m\,s^{-2}}$ error would cause the amplitude to drop down to a mere 20% of the fully coherent value, while according to Fig. 3.5, the drop is only to about 50% of the ideal case. Figure 3.5 is the more relevant one, for there we allow both the model velocity and the acceleration to vary when searching for the AF maximum. As is seen in Fig. 3.5, if there is an error in the acceleration parameter, say $a > a_0$, then the AF maximum does not occur at the actual target velocity v_0, but at a somewhat smaller value, $v - v_0 < 0$. This is to be expected, for then the average velocity of the model signal will be more correct during the integration, and therefore a better overall match results. Figure 3.6 can be used also for analysis design, for it shows that if one wants to determine acceleration with an accuracy of about $1\ \mathrm{m\,s^{-2}}$, one must use integration time of the order of 0.5 s at least.

The ambiguity functions shown in Fig. 3.5 and 3.6 are helpful in gaining insight into the properties of the MF method, but one should not make too strong conclusions based on them alone. One must remember that for real data there are other factors beside the acceleration error that will affect the coherent integration, for examples, the presence of multiple frequencies in the transmission (this case is handled in chapter 3.8). A conclusion we can safely draw from figures such as Fig. 3.5 and Fig. 3.6 is that the length of coherent integration in the analysis phase should be kept quite short, perhaps

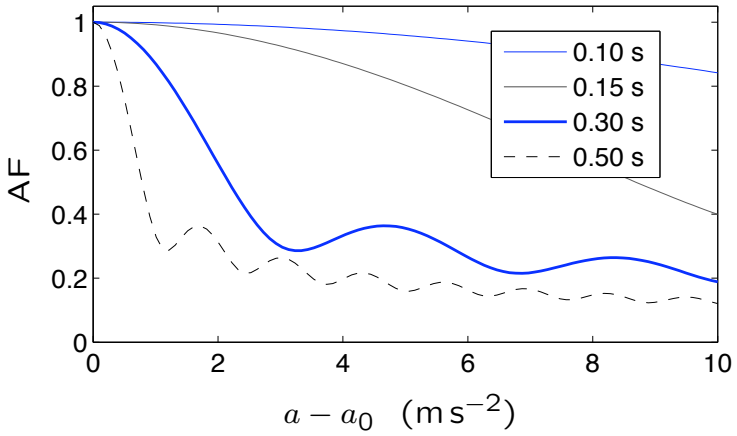

Figure 3.6.: Loss of integrated amplitude due to acceleration error. The figure shows the acceleration ambiguity function $a \mapsto \mathrm{AF}(a_0; a)$ for four integration times. All the AF are normalized by their maximum value. We assume that the range and the velocity are fixed to the target values. The figure suggests that, for example, with 0.3 s integration a 2 m s^{-2} error in acceleration would cost us almost 50% of the amplitude. In reality the situation is not quite that bad, but nevertheless, a short integration time like 0.1 s might be preferable, at least in the parameter estimation phase.

as short as 0.1 s, in order not to distort the amplitude estimates too much. (For detection phase we seem to gain sensitivity from longer integration, up to about 0.3 s integration time.) A 0.1 s integration time would still mean integration over 10–20 pulses in typical experiments, and ought to be a definite improvement over non-coherent integration.

3.7. The fast match function algorithm

Since spring 2001 we have used the fast match function algorithm, FMF, for all our practical target detection computations. We showed in the final report of our previous study that in practice we do not lose much accuracy even if we use the FMF also for parameter estimation, at least when the estimates can be done by fitting to multiple points of time series data (things might be different if we only had a single point available—as for the weakest signal we might have.) By using the FMF, which is about 300 times faster than the MF in a typical situation, we can easily achieve real-time speed for the overall data processing. We now have enough computing power, albeit just barely, and only when using less-than-maximal sampling rates, and by using two dedicated G5's in parallel, to do parameter estimation on the detected events in real time also using the standard MF. We have analysed, off-line, all our data from the present study both with the MF and FMF, and a comparison of the results confirms the conclusion from the previous study.

In this chapter, we will document some of the emerging understanding of the properties of the FMF and the FAF. From the look of the data processed with the FMF and MF algorithm, it is evident that there are conspicuous and pertinent differences between the two functions. (Perhaps one should think the FMF as an transformation of its own right, rather than just an "approximation" of the MF). Typical differences are shown,

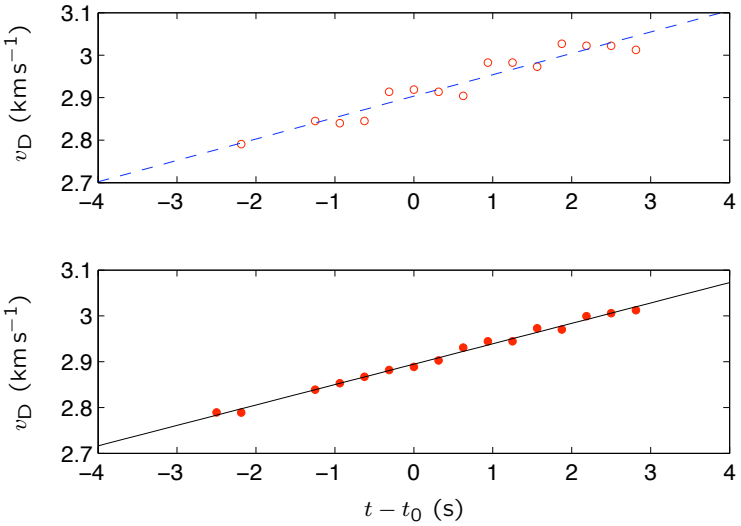

Figure 3.7.: Typical Doppler-velocity behaviour during a beam passage under very high SNR conditions. The experiment is tau1. For the top panel, analysis was done using FMF, for the bottom panel, using MF. The fitted lines are $v_D = 2.90\,\mathrm{km\,s^{-1}} + 50\,\mathrm{m\,s^{-2}} \times (\mathrm{t} - \mathrm{t_0})$ and $v_D = 2.89\,\mathrm{km\,s^{-1}} + 45\,\mathrm{m\,s^{-2}} \times (\mathrm{t} - \mathrm{t_0})$ for the top and bottom panel, respectively. The conspicuous feature is the non-physical, staircase-like change in the FMF data. For instance, the jump between the 4th and 5th point in the top panel is 74 $\mathrm{m\,s^{-1}}$. The integration time is 0.3 s for each point. The time instant t_0 is 7-Sep-2004, 21:12:47.4 UT, the EISCAT Tromsø UHF antenna was pointed to azimuth 133.3° and elevation 61.6°.

for high-SNR tau1-data, in Fig. 3.7 and in Figures 3.8 and 3.9.

The prominent difference between MF and FMF is in the spectral shape, that is, in the shape of the velocity slices. In parameter estimation, that difference shows up in large unphysical jumps in the $v_\mathrm{D}(t)$ FMF data. In fact, as shown in the top panel of Fig. 3.8, the apparent velocity can even decrease for a while although the physical radial acceleration is an all cases positive, and fairly large (at least for circular orbits). The decrease is not due to noise, but is a property of the FMF. How such systematic discrete jumps can arise in our presumably smooth data is illustrated in Fig. 3.8. The essential feature is that when the target velocity changes, the FMF does *not* merely shift by that amount, but also its detailed shape changes. The situation is analogous to the difference between group and phase velocity in wave propagation (but now in velocity space), so that even though in the mean the overall pattern, and thus any particular peak, shifts with the correct target acceleration, there is an embedded "modulation" that moves along the overall pattern by changing the relative heights of the local maxima. This causes the observed maximum to move with an acceleration that is different from the target acceleration (and which can even be negative), and also every now and then causes the absolute maximum to jump from a peak to the next. The typical size of the jump is roughly the distance between the main peaks of the FMF. It can be shown, by an argument similar to that leading to Eq. (3.82), but applied to the two-frequency case instead of a single-frequency case, that this distance in the tau1 and tau2 EISCAT

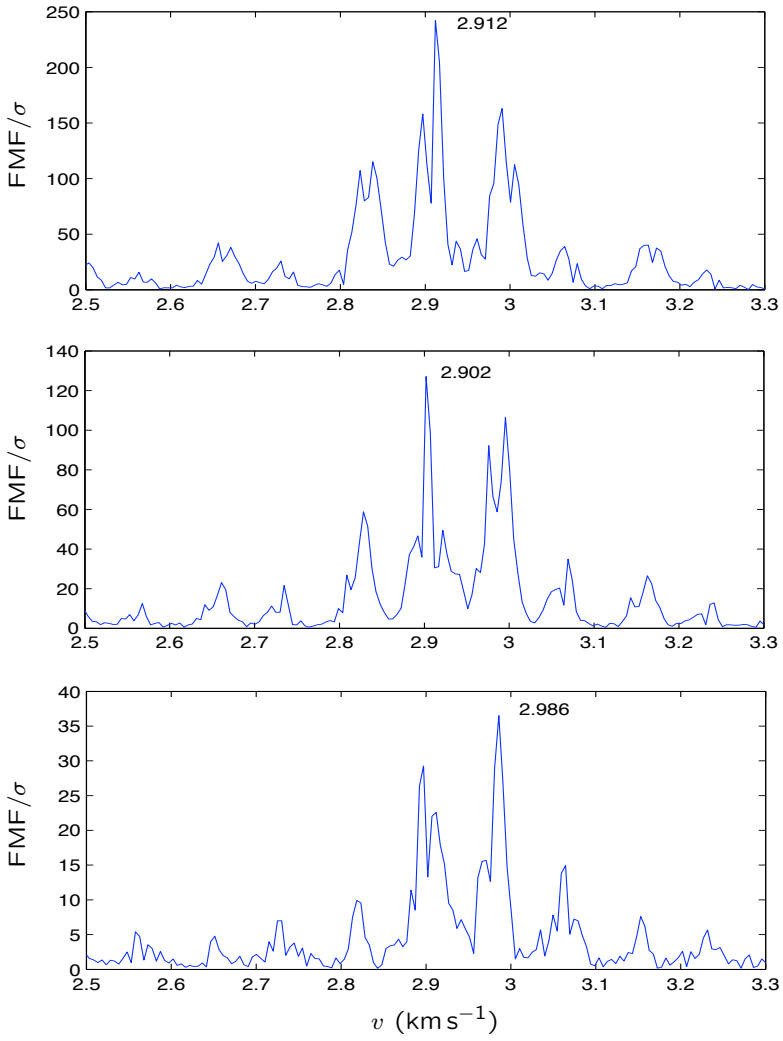

Figure 3.8.: Anatomy of velocity jumps when using the FMF algorithm. The figure shows FMF velocity slices $v \mapsto \mathrm{FMF}(\widehat{R}, v)$ from three consecutive 0.31 s integration. The integrations correspond to the 7th, 8th, and 9th point in the top panel of Fig. 3.7. The data are from the outgoing edge of a target's passage through the beam main lobe, so the overall signal amplitude is rapidly decreasing. The target's radial acceleration is 45 $\mathrm{m\,s^{-2}}$, which would give velocity step of 15 $\mathrm{m\,s^{-1}}$ from panel to panel. Instead, what is observed is a slight decrease from the top panel to the middle panel, and then a jump of 84 $\mathrm{m\,s^{-1}}$ from the middle panel to the bottom panel, when the FMF maximum jumps from the main peak at 2.9 $\mathrm{km\,s^{-1}}$ to the next main peak at 3 $\mathrm{km\,s^{-1}}$. The distance between the FMF main peaks is $(\lambda/2)/(2L) = 84\,\mathrm{m\,s^{-1}}$ in this two-frequency tau1 data where the pulse length L is 960 μs. Although there is a lot of structure in the curves, almost all of it is a genuine property of the FAF. The effect of noise is insignificant in these data, for even in the bottom panel the $\mathrm{SNR_N}$ is still about $37^2 = 1400$.

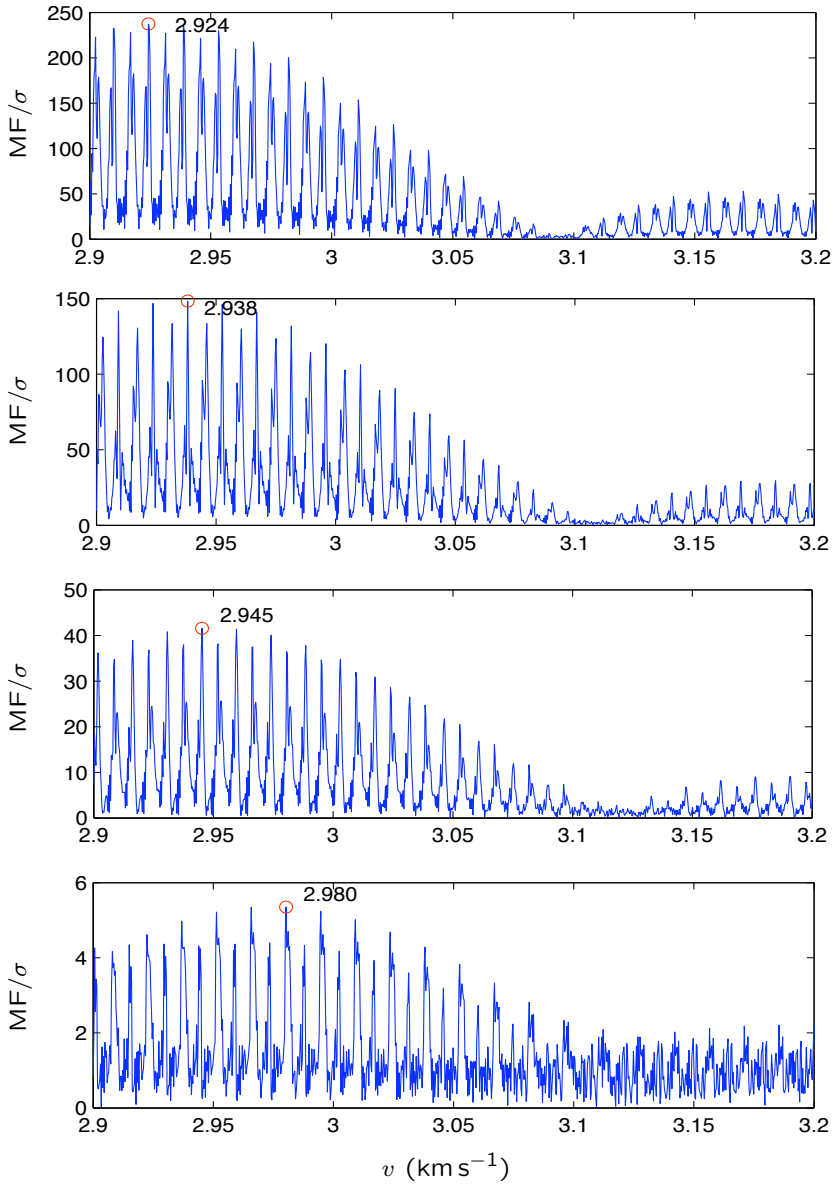

Figure 3.9.: Anatomy of velocity jumps when using the MF algorithm. The panels show the MF velocity slice $v \mapsto \mathrm{MF}(\widehat{R}, v)$ in four consecutive 0.31 s integration. The target's radial acceleration is 45 $\mathrm{m\,s^{-2}}$, so we would expect the MF maximum position to shift by about 15 $\mathrm{m\,s^{-1}}$ from panel to panel. But due to the inherent velocity ambiguity of $n \times 7.2\,\mathrm{m\,s^{-1}}$, the maximum may be found displaced from the expected position by that amount. For example, the "correct" maximum position in the third panel would be not $2.945\,\mathrm{km\,s^{-1}}$, but rather the position of the next peak, $2.952\ \mathrm{km\,s^{-1}}$.

experiments is

$$\Delta v_{\mathrm{D}} = \frac{\lambda}{2}\frac{1}{2L}\,, \tag{3.70}$$

where λ is the radar wave length (0.32 m) and L is the length of a single transmitted pulse, so that Δv_{D} is 84 $\mathrm{m\,s^{-1}}$ in tau1 and 140 $\mathrm{m\,s^{-1}}$ in tau2.

There are jumps also in MF velocity data, even with very high SNR, as shown in the bottom panel of Fig. 3.7. However, these are smaller than the main jumps in the FMF data, and have different origin. The jumps are due to the inherent velocity ambiguity (the one that gives rise to the name ambiguity function) of pulsed Doppler radar data, and have typical magnitude equal to a small integer times the separation of the main peaks of the MF. For tau1 and tau2 the peak separation is $\delta v = n \times (\lambda/2) \times (1/P_{ch})$, where P_{ch} is the pulse repetition period on a given frequency channel, 11160 μs for tau2 and 22320 μs for tau1. This gives $\delta v = n \times 7.2$ $\mathrm{m\,s^{-1}}$ in tau1 and $\delta v = n \times 14.5$ $\mathrm{m\,s^{-1}}$ in tau2. These small jumps should occur at random times, and with positive and negative n, so they should not bias the fitted velocity and acceleration in fits like the one shown in Fig. 3.7.

Comparison of the FMF in Fig. 3.8 and the MF in Fig. 3.9, both computed from the same raw data, shows that the main peaks in the FMF are much more sparsely placed than those of the MF. The ratio of the peak separation is $\delta v/\Delta v = L/P$, that is, equals the duty cycle of the experiment (0.086 in tau1 and 0.103 in tau2).

Observing and comparing the non-random jumps in $v_{\mathrm{D}}(t)$ in MF and FMF analysis is not the best way to find out an experiment's duty cycle, but there is another aspect here that deserves to be inspected. In the context of the classical range-Doppler dilemma, it makes sense that with a given pulse length, the velocity is the less ambiguous the further the MF peaks are from each other. In terms of the MF, this comes about because the envelope curve of the MF peaks is $\mathrm{diric}(\omega - \omega_0, L)$, where the Dirichlet kernel[6] is

$$\mathrm{diric}(x, M) = \left| \frac{\sin x\frac{M}{2}}{\sin \frac{x}{2}} \right|\,, \tag{3.71}$$

so that the side peaks will be the smaller compared to the main peak the larger the peak separation is. Does the dependency of the size of the velocity ambiguity on the peak separation now imply that the FMF method, with its large separation of the main spectral peaks, somehow manages to reduce the ambiguity compared to the MF. Not so; the velocity ambiguity with the FMF is the same as with the MF, $n \times 1/P$, determined by the pulse repetition rate $1/P$. Figure 3.10 shows how this comes about. Consider the top and bottom panels of the right-hand-side column in Fig. 3.10. The two panels show a FAF frequency slice, and look pretty much the same. Especially, the main peak is in the same position in the two cases, so the FMF algorithm will give the same Doppler-frequency. Nevertheless, the actual target Doppler-frequency differs by $1/P$ in the two panels. When noise is added into the picture, there is a real danger that the two situations will be confused. Unless we know a priori on what particular interval of length $1/P$ the velocity is, in the presence of noise we cannot invert the observed peak location to find the actual target velocity. (If there is no noise, we can always invert the FAF to find its parameters; this, of course, applies to the MF also.) So, we have to live with the usual Doppler ambiguity here also. The center panel shows an intermediate

[6] Note that we include taking the absolute value into the definition of the diric function. This is not normally done.

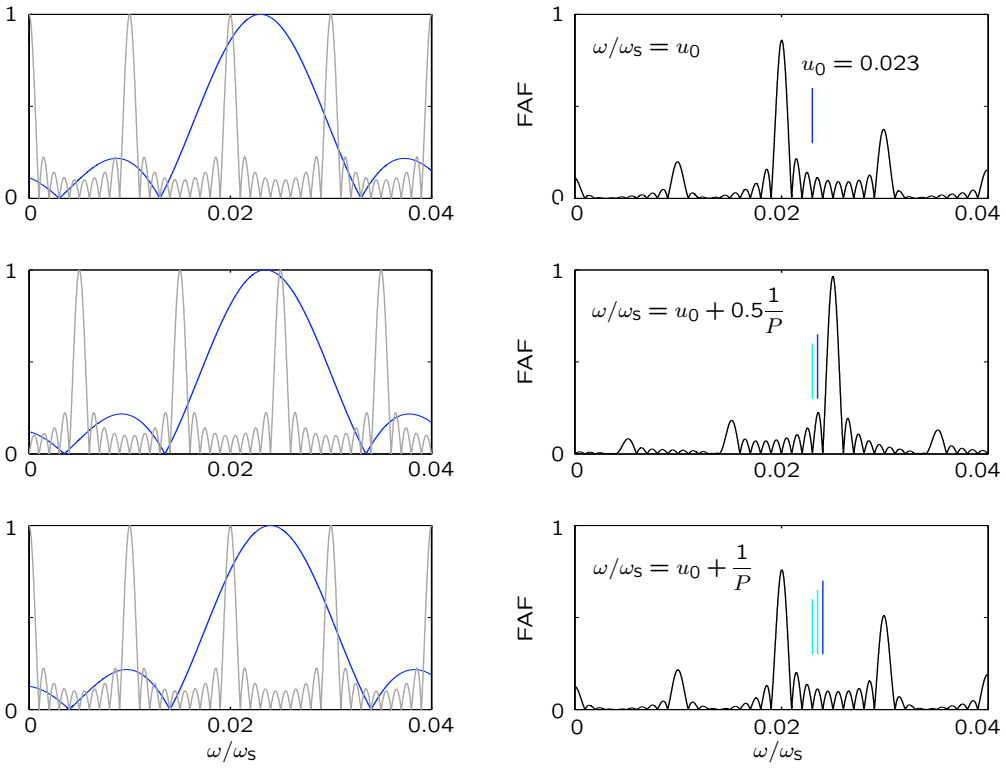

Figure 3.10.: Mechanism of the $n \times \frac{1}{P}$ Doppler-frequency ambiguity in the FMF algorithm. We assume pulse length $L = 100$ samples, interpulse period $P = 1000$ samples, coherent integration over $M = 10$ pulses. The target's Doppler-frequency ω_0 in the top panel is $\omega_0 = u_0\,\omega_s$, where $u_0 = 0.023$ and $\omega_s = \frac{2\pi}{\tau_s}$ is the normalized sampling frequency. In the middle panel $\omega_0 = u_0\,\omega_s + 0.5\frac{\omega_s}{P}$, and in the bottom panel $\omega_0 = u_0\,\omega_s + \frac{\omega_s}{P}$. The left-hand-side columns show the composition of the function $\omega \mapsto \mathrm{FAF}(\omega 0; \omega)$ to a product of two factors. The blue curve is $\mathrm{diric}((\omega_0 - \omega)\tau_s, L)$, the gray curve is $\mathrm{diric}((P\omega_0 - L\omega)\tau_s, M)$. The maximum of the blue curve is at ω_0; its shift to the right is traced by the short vertical lines in the right-hand-side panels. When ω_0 increases, the grey curve shifts P/L times faster to the right than the blue curve. When ω_0 has increased by $\frac{2\pi}{P\tau_s}$, the grey curve has shifted by precisely $\frac{2\pi}{L}$, the interval between its main peaks. Meanwhile, the blue curve has shifted only very little. Therefore, the two values of ω_0, differing by $\frac{2\pi}{P\tau_s}$, give almost identical overall shape of the FMF, and produce essentially identical position for the FMF maximum. This is the Doppler ambiguity.

situation, where the target's actual Doppler-shift is halfway between the top and the bottom cases, while the left-hand-side panels show how the FAF is constructed as a product of two factors in accordance of Eq. (3.80). For the specific parameter values used in this example, refer to the figure's caption.

We now describe the FMF algorithm, originally introduced my M Lehtinen in an internal unpublished note, April 2001, but never earlier described in the project documentation.

The FMF algorithm was designed originally as an approximation to the well-known—though we did not know at the time that is was so well-known—radar ambiguity function, generalized to our case by including target acceleration. It was the explicit purpose to use the FMF for target detection purposes only, so it was designed to give a good approximation of the ambiguity function near its absolute maximum position. Especially, it was important that the FMF would achieve coherent integration to a good approximation.

We make use of two special properties of our measuring situation in EISCAT: the property that we are using a pulsed radar; and the property that the space debris receiver "oversamples" the debris signal heavily. The quotes are needed, for the high sampling rate is necessary to correctly sample the multi-frequency transmission. But we are oversampling with respect to the Doppler-shifted, inherently narrow-band signal on any given frequency channel. The time consuming phase in the MF evaluation is to compute the velocity slices, that is, the power spectra for a set of range gates. We make use of the two properties to drastically reduce the length of the FFT input vectors.

First, we note that the Doppler-velocity interval that we need to monitor is much narrower than the interval that is actually available with the high sampling rates f_s that we use in the SD receiver. For the 930 MHz radar frequency (0.32 m wavelength), the benchmark 2 MHz sampling gives unambiguous velocities in the interval $\pm(f_s/2)\cdot(\lambda/2) = \pm 160\ \mathrm{km\,s^{-1}}$. Typically, for near-vertical pointing, it suffices to monitor velocity interval $\pm 5\ \mathrm{km\,s^{-1}}$. Therefore, for each range gate j, we can downsample (decimate) the to-be-Fourier-transformed vector w, which according to Eq. (3.40) is formed from transmission samples x_n, signal samples z_n, and an acceleration correction term $\exp(-i\alpha_j n^2)$,

$$w_n = z_n \overline{x}_{n-j} \mathrm{e}^{-i\alpha_j n^2}\,, \tag{3.72}$$

by a factor M_{dec}, which is $160/5 = 32$ for 2 MHz sampling, and 8 for the more typical 0.5 MHz sampling. This we do by forming a new sequence w'_n by adding w_n's in blocks of M_{dec}. At the same time, we make use of the fact that within such a block, the acceleration factor is almost constant. For each block we use the average phase ϕ' within the block and take $\exp(-i\phi')$ out of the decimation sum. This reduces both the number of multiplications and the number of complex exponentials that need to be evaluated.

Second, we make use of the fact that most of the elements of w are zeros, in known locations (formally, the terms x_n are zeros in most places). When we form w' from w, we only compute and decimate explicitly the products (3.72) at the points where we know there can be non-zero data. The transmission duty cycle in EISCAT experiments is about 10% in the UHF and about 20% at ESR. Therefore, about 80–90% of the elements of w are zeros, in regularly placed blocks. We now simply concatenate the non-zero blocks of w'. We get a vector w'', which typically is two orders of magnitude shorter than the original FFT input vector w. For example, in the benchmark case with 600,000 point raw data input vector, using decimation factor 15 as we normally do, w'' has the length $N'' = (1/15)\cdot 0.1 \cdot 600,000 = 4000$. Finally, we Fourier-transform and

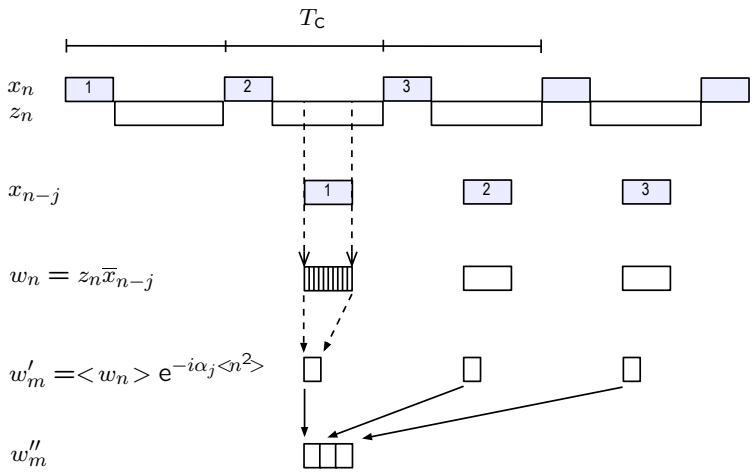

Figure 3.11.: Forming the Fourier-transform input vector for a given range gate in the FMF algorithm. For each of M IPP's (M=3 in the figure), one first forms the point-wise product w of the reception z and the shifted, complex-conjugated transmission $\overline{x}$. Next, for each IPP, w is decimated, and the decimated vector multiplied by an appropriate mean acceleration correction factor to get the vector w'. Finally, the w'-vectors from the IPPs are concatenated to form the input vector w''.

normalize w'' to get the FMF at range gate R_j and Doppler-frequency ω,

$$\mathrm{FMF}(R_j, \omega) = \frac{|\sum_{n=0}^{N''-1} w''_n e^{-i\omega\tau_s n}|}{\|x\|} . \tag{3.73}$$

By restricting ω to the points $\omega_k = 2\pi k/(N''\tau_s)$, the FMF can be evaluated using FFT. This is the fast match function algorithm. Due to the much shorter FFT input vector, even allowing for the extra computations needed in decimation, the FMF is 100–300 times faster than the MF in typical cases. The computation of the FFT input vector w'' in Eq. (3.73) is summarized in Fig. 3.11.

Gaining speed by the FMF algorithm is not in doubt. But what is the price? Decimation, the first step in the algorithm, does not lose us much information. Basically, we are just backtracking from the initial oversampling. We can backtrack at this stage, because the signal w has already been demodulated. Reception is multiplied by the complex conjugate of the transmission, so the carriers cancel out. Very near to the MF maximum also a possible transmission phase modulation is cancelled out. So the sampling requirement of w is determined by the size of the maximum anticipated Doppler-shift only.

We now inspect the second step. Remembering our worry in section 3.6 concerning the difficult-to-control phase shifts in pulse-to-pulse coherent integration, we might expect trouble here where we have concatenated data blocks by dropping the zero-segments between the pulses. One might expect that the phase jumps caused by deleting segments of data before taking the FFT would mean that the FFT phase factor, which should cancel the Doppler-phase factor from the signal, cannot continue to do so from pulse to pulse. So it perhaps comes as a surprise that the damage keeps limited, and the

sensitivity stays above a lower limit that does not depend on the experiment. Perhaps there is some simple way to see how and why the lower bound really comes about; but here we will proceed with a direct evaluation of Eq. (3.73) in a relevant special case.

What effect does disregarding the zero-blocks have to the Fourier transform? We ignore the decimation step, taking $M_{\text{dec}} = 1$. We are interested in the behaviour of the MF near its maximum. Thus, we will assume that the correct range and acceleration have already been found, and only a Doppler-term $e^{i\omega_0 n}$ in w still needs to be cancelled (matched). We ignore noise, so we really are computing the FAF here.

We assume that the transmission consists of M pulses of length L samples each, sent P samples apart. Then, near the maximum, w consists of M pulses of, say, unit amplitude and L samples each, Doppler-shifted by the normalized frequency ω_0, with $P - L$ zeros between each pair of pulses. The non-zero part of w consists of M blocks, and in the m'th block, w has elements

$$w_n^{(m)} = e^{i\omega_0(n+mP)},\ n = 0, \ldots, L-1. \tag{3.74}$$

For computing FAF(ω) the blocks $\{w_n^{(m)}\}$ are first concatenated and then the result is multiplied by $\mathrm{e}^{-i\omega n}$. The m'th block gets multiplied by

$$u_n^{(m)} = \mathrm{e}^{-i\omega(n+mL)},\ n = 0, \ldots, L-1\,. \tag{3.75}$$

The contribution I_m of the mth block to the inner product in the denominator of Eq. (3.73) is

$$I_m = \sum_{n=0}^{L-1} w_n^{(m)} \cdot u_n^{(m)}\,. \tag{3.76}$$

The norm $\|x\|$ in the nominator of Eq. (3.73) is the sum of ML terms, all equal to unity, so we get from Eq. (3.73) and (3.76)–(3.74)

$$\begin{aligned}
\mathrm{FAF}(\omega) &= \frac{1}{ML}\left|\sum_{m=0}^{M-1} I_m\right| && (3.77)\\
&= \frac{1}{ML}\left|\sum_m \sum_n \mathrm{e}^{i\omega_0(n+mP)}\mathrm{e}^{-i\omega(n+mL)}\right| && (3.78)\\
&= \frac{1}{L}\left|\sum_{n=0}^{L-1} \mathrm{e}^{i(\omega_0-\omega)n}\right| \cdot \frac{1}{M}\left|\sum_{m=0}^{M-1} \mathrm{e}^{i(\omega_0 P-\omega L)m}\right| && (3.79)\\
&= \mathrm{diric}(\omega_0-\omega, L)\cdot \mathrm{diric}(\omega_0 P-\omega L, M)\,, && (3.80)
\end{aligned}$$

The Dirichlet kernel diric was defined in Eq. (3.71).

For comparison, we mention without derivation that the standard single-channel AF has an expression where the last "L" in Eq. (3.80) is replaced by a "P",

$$\mathrm{AF}(\omega_0;\omega) = \mathrm{diric}(\omega_0-\omega, L)\cdot \mathrm{diric}\left((\omega_0-\omega)P, M\right)\,, \tag{3.81}$$

See Fig. 3.12 and Fig. 3.10 for illustrations of Eq. (3.80). The first factor in Eq. (3.80) encodes the Doppler-information available from a single pulse. The factor has its absolute maximum at the target Doppler-frequency ω_0, and first zeros at $\omega_\pm = \omega_0 \pm 2\pi/L$. The second factor in Eq. (3.80) results from pulse repetition. It has maxima, equal to unity, at the points

$$\omega_n = \frac{P}{L}\omega_0 + n\frac{2\pi}{L}\,. \tag{3.82}$$

In general, none of the maxima ω_n coincides with ω_0. Therefore, the maximum of FAF(ω) is not situated in the correct place ω_0, even when there is no noise. This means that the estimated velocity will be biased. This is not serious. The bias is rather small, less than 0.2 kms^{-1} in our typical measurement modes. For some values of the target velocity, the bias vanishes, as shown in Fig. 3.12.

What is more worrisome for target detection, is the loss of maximum integrated amplitude. The maximum value of the FAF occurs (very near) that ω_n which is nearest to ω_0. Such an ω_n, according to Eq. (3.82), is never further away from ω_0 than half the spacing $2\pi/L$ between the ω_n's. Therefore, the FAF maximum value, in the worst case, is roughly equal to diric$(\pi/L, L) \approx 2/\pi$, or about 64% of the ideal value.

We must also remember that when we use FFT, and can only evaluate FAF (and MF) at the discrete set of points ω_k, the worst-case maximum value can get worse by an additional factor of 0.6 due to the "picket-fence" effect. On the other hand, we normally can observe a target for several seconds, and during that time, its velocity, hence ω_0, typically changes so much that during some integration, the maximum gets nearer to the ideal value.

3.8. Ambiguity functions in dual-frequency experiments

Our discussion up to now has been mostly concerned with single-frequency-channel experiments. However, the standard EISCAT experiments tau1 and tau2 use two frequency channels. The particular frequencies used are subject to change without warning by EISCAT, but so far in all our SD experiments the frequencies have been the frequencies F13 and F14, 929.9 MHz and 930.2 MHz. In the derivation of the MF method, we mentioned that the method in principle applies unmodified to *all* transmissions. But we also said that for numerical efficiency, we had to apply approximations to the signal model. These approximations led to the standard signal model functions χ of Eq. (3.33). Both the AF and the FAF have so far been expressed using these approximative model functions. The approximation is good for single-channel data. But for our actual two-channel data the approximation breaks down, the worse the larger the target velocity is, and the longer the integration time we use. The break-down of the signal model causes the estimated signal amplitude to fall below the ideal, "fully coherently integrated", value. This causes some loss of detection sensitivity. The bad news is that already with our standard 300 ms integration time, for targets with Doppler-velocities larger than about 2 km s^{-1}, we are only just marginally more sensitive than if we were making use only of a single channel's data. The good news is that this is about as bad as things will get; the sensitivity will not fall below the single-channel sensitivity.

With N_{ch} narrowband channels in the transmission, but with only a single channel being transmitted or received at any given instant of time, the proper signal model would be a sum of the single-channel models of Eq. (3.33),

$$\chi(R, v, a; t) = \sum_{l=1}^{N_{\text{ch}}} x_l(t - \frac{2R}{c})\, \mathrm{e}^{i(\omega_l^{\text{D}} t + \alpha_l^{\text{D}} t^2)}\,, \tag{3.83}$$

where $x_l(t)$ is the transmission sample signal on channel l, and the channel's Doppler-shift and Doppler-drift are related to the target's radial velocity v, radial acceleration a,

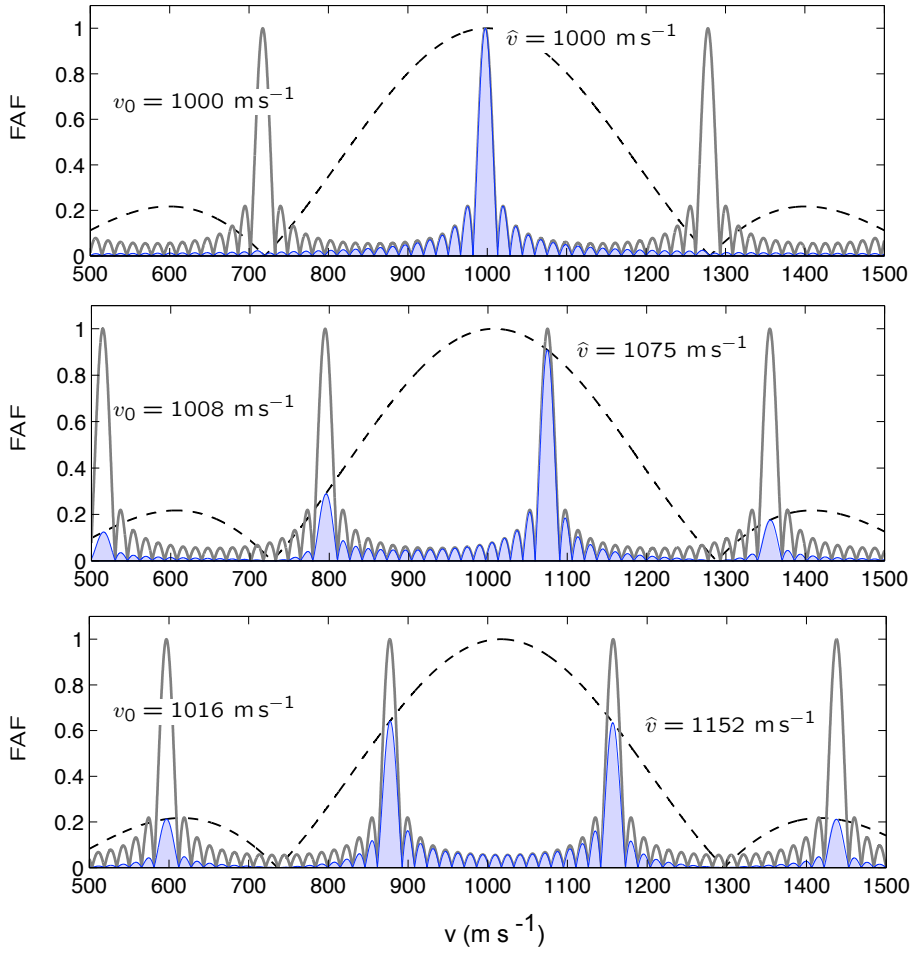

Figure 3.12.: Structure of the FAF in single-frequency case. The panels show the velocity slice $v \mapsto \mathrm{FAF}(v_0, v)$ for a single-frequency tau2 transmission scheme, with different values of the target velocity v_0. The length of integration is 0.1 s. According to Eq. (3.80), the FAF (solid color) is the product of two factors, shown here by the slowly varying dashed curve and the faster varying solid-line curve. The dashed curve is $\mathrm{diric}(\frac{4\pi(v-v_0)\tau_\mathrm{s}}{\lambda}, \frac{L}{\tau_\mathrm{s}})$ where L is the pulse length, τ_s is the sampling interval, and λ is the radar wavelength. The curve is has absolute maximum at v_0. The panels illustrate what happens when v_0 changes from 1000 m s^{-1} to 1016 m s^{-1}. The solid-line curve shifts to the right much faster than the dashed curve when v_0 is increased, and therefore also the maximum of FAF shifts to right faster. The top panel shows that with a suitable v_0, in the single-frequency case it is possible to get an ideal match with the FAF, with no sensitivity loss and with a correct velocity estimate. The bottom panel shows the worst-case situation. The velocity estimate is wrong by 0.136 km s^{-1} and the amplitude estimate is wrong by 35%.

and channel's carrier frequency ω_l, by

$$\omega_l^{\mathrm{D}} = -\omega_l \frac{2v}{c} \tag{3.84}$$

and

$$\alpha_l^{\mathrm{D}} = -\omega_l \frac{a}{c}\,. \tag{3.85}$$

The model (3.83) is reasonably accurate when the integration time is not too long, and should be used in the MF algorithm for multichannel data. Then, for instance, the multichannel velocity ambiguity function would be

$$\begin{aligned} \mathrm{AF}(v_0; v) &= \frac{|\langle \chi(R_0, v_0), \chi(R_0, v)\rangle|}{\sqrt{W_x}} \\ &= \frac{1}{\sqrt{W_x}} \sum_l \int_0^{T_c} \left| x_l(t - \frac{2R_0}{c}) \right|^2 \mathrm{e}^{i\frac{2\omega_l}{c}(v_0 - v)t}\, dt \end{aligned} \tag{3.86}$$

where W_{x} is the transmission energy as usual. Note that there are no crossed terms containing $x_l(t)\overline{x}_{l'}(t)$ with $l' \neq l$, because only a single transmission is received at a time. As it should, the ambiguity function in Eq. (3.86) is at maximum at $v = v_0$, equal to $\sqrt{W_x}$, the square root of the energy of $\chi(R_0, v_0)$. And if the model functions (3.83) are used in the MF method to cross-correlate against a noisy measurement $z(t)$, we can expect the procedure to yield the correct Bayesian estimates, just as it does in the single-frequency case.

However, for computing efficiency, we need to use FFT in the evaluation of the velocity slices, and then we cannot use the correct model functions Eq. (3.83). Instead, we ignore the channel structure in the data, take the vector z as a whole, and essentially compute power spectra from the entire vector for the relevant range gates, using FFT. In effect, instead of Eq. (3.83), we use the model functions

$$\xi(R, v, a; t) = \left(\sum_{l=1}^{N_{\mathrm{ch}}} x_l(t - \frac{2R}{c})\, \mathrm{e}^{i\,\alpha_l^{\mathrm{D}}\, t^2} \right) \mathrm{e}^{i\,\omega\, t}\,, \tag{3.87}$$

where some typical radar frequency like the mean $\omega_{12} = 0.5(\omega_1 + \omega_2)$ is used to convert between ω and the model's velocity parameter,

$$\omega = -\omega_{12}\, \frac{2v}{c}\,. \tag{3.88}$$

All processing of real multichannel data has been done using MF and FMF based on the models ξ; for example, we have computed the MF from

$$\mathrm{MF} = \frac{|\langle z, \xi\rangle|}{\|x\|}\,. \tag{3.89}$$

It obviously is not possible to match in Eq. (3.89) both Doppler terms

$$e^{i\,\omega_1^{\mathrm{D}}\, t} \text{ and } e^{i\,\omega_2^{\mathrm{D}}\, t}\,,$$

which are present in the actual two-channel signal z, by using only the single Fourier factor available in ξ,

$$\overline{\xi} \propto e^{-i\omega t}\,,$$

so some of the integrated amplitude is necessarily lost. How big the loss is can be studied quantitatively via the ambiguity functions, where the term in the inner product representing the target is the correct model signal χ, and the other term is the simplified model function ξ. For example, the velocity ambiguity function corresponding to Eq. (3.89) is

$$v \mapsto \mathrm{AF}(v_0; v) = \frac{|\langle \chi(v_0), \xi(v) \rangle|}{\|x\|} \,. \tag{3.90}$$

An expression for the ambiguity functions in the actual two-frequency tau1 and tau2 experiments can be derived by adopting from the steps leading to the single-frequency expressions Eq. (3.80) and Eq. (3.81). Not surprisingly, the ambiguity functions are now obtained as the magnitude of the sum of two complex-valued terms, both of the general type of diric() × diric(), but with some extra phase factors appearing. For example, the fast velocity ambiguity function $\mathrm{FAF}(v_0; v)$ in the two-frequency case is obtained by computing through the following definitions ($l = 1, 2$):

$$\begin{aligned}
\phi_l &= \omega_l^{\mathrm{D}} \cdot 2P - \omega \cdot 2L \,, \\
\Omega_l &= \omega_l^{\mathrm{D}} - \omega \,, \\
A_1 &= \mathrm{e}^{i(\frac{M}{2}-1)\cdot\phi_1} \cdot \mathrm{diric}(\phi_1, \frac{M}{2}) \,, \\
A_2 &= \mathrm{e}^{i\frac{M}{2}\cdot\phi_2} \cdot \mathrm{diric}(\phi_2, \frac{M}{2}) \,, \\
B_l &= \mathrm{e}^{i(L-\tau_s)\,\Omega_l} \cdot \mathrm{diric}(\Omega_l \tau_\mathrm{s}, \frac{L}{\tau_\mathrm{s}}) \,, \\
C_l &= A_l \cdot B_l \,, \\
\mathrm{FAF} &= \frac{|C_1 + C_2|}{2} \,.
\end{aligned} \tag{3.91}$$

The channels' Doppler-frequencies ω_l^{D} corresponding to target velocity v_0 are obtained from Eq. (3.85), the ω in Ω_l corresponding to the model's v is obtained from Eq. (3.88). P is the interpulse period (in time units), L is the pulse length, M is the (even) number of interpulse periods in the integration, and τ_s in the expression for B_l is the sampling interval. The ambiguity function AF is obtained similarly to Eq. (3.91), with the single difference that the "L" in the expression of ϕ_l is replaced by "P". When $\omega_1^{\mathrm{D}} = \omega_2^{\mathrm{D}}$, Eq. (3.91) reduces to the single-frequency case, Eq. (3.80).

Figure 3.13 illustrates two features of the FAF (3.91) that are due to the two-frequency transmission. The first feature is specific to the FAF, the second applies also to AF. The first feature is that the interval between the FMF main maxima is only half as much as in the single frequency case. The frequency difference between the maxima is $\Delta f = 1/(2L)$ instead of $1/L$. In tau1, $1/(2L)$ corresponds to velocity difference 84 $\mathrm{m\,s^{-1}}$; in tau2, to 140 $\mathrm{m\,s^{-1}}$. Figure 3.13 shows how a new peak grows halfway between the single-frequency peaks when the frequency difference between the transmission frequencies is increased from zero to 300 kHz. The fact that the local FMF maxima are separated by $1/(2L)$ means that there is always one such peaks within $1/(4L)$, and two such peeks within $1/(2L)$, from the correct target velocity v_0, located at the maximum of the main lobe of width $1/2L$ of the term $\mathrm{diric}(\omega_l^D - \omega, L)$. A third peak is also near the maximum, but the other peaks are more strongly suppressed. In a noisy environment this results in the characteristic 2- or 3-peak structure near the maximum location of the velocity slices seen in Fig. 3.4. The "extra" peak, compared to the situations that both frequencies

would be the same, implies that the maximum velocity bias in the two-frequency FMF algorithm is only 0.04 $\mathrm{km\,s^{-1}}$ in tau1 and 0.07 $\mathrm{km\,s^{-1}}$ in tau2. Arguably, the bias is not the crucial factor here, for it is more the velocity ambiguity that determines the random velocity measurement errors. With the ambiguous peaks more near to each other now, the error distributes onto a more fain-grained grid, but its mean magnitude does not necessarily change much. (Here is a place for a simulation study.)

Second, Fig. 3.13 illustrates the main problem caused by the dual-frequency transmission being matched with the oversimplified model functions ξ. The velocity ambiguity function is now essentially the spectrum of the combined, repeated transmission. The two Doppler-frequencies will go to two different locations, separated by $\delta f = |f_1^{\mathrm{D}} - f_2^{\mathrm{D}}|$ (and then the pair is repeated with interval Δf). Due to finite integration time, the frequencies show up as "spectral lines", of width $2/T_{\mathrm{c}}$ (counted from zero-to-zero). When δf becomes greater than about a quarter of the line width, the two peaks start become clearly separated. When the separation increases, no matter what the phase-difference between the complex-valued sub-spectra, represented by the terms C_1 and C_2 in Eq. (3.91), will be, the maximum attainable integrated amplitude will tend towards the single-frequency level. For instance, with $T_{\mathrm{c}} = 300\,\mathrm{ms}$ and with 300 kHz difference between transmission frequencies, the condition $\delta f = (1/4)(2/T_{\mathrm{c}})$ gives target velocity $v_0 = 830\,\mathrm{m\,s^{-1}}$. And soon after δf exceeds $1/T_{\mathrm{c}}$, we expect the ambiguity function, or at least its maximum value, to behave is if there would be two incoherent signals. The total signal energy does not vanish anywhere, but it is then represented by two peaks, each of height $\sqrt{W_{\mathrm{tot}}}/2$. The problem for our threshold-based detection is evident. Moreover, after detection, we obviously would need to modify our standard energy estimate to account for this "loss of coherence". Such a correction we have not done in any of our actual data analysis yet.

In Fig. 3.14 we show the loss of the integrated amplitude as a function of target velocity, for two integration times in the tau2 experiment. In the figure, we plot the maximum value of FAF and AF as a function of target radial velocity. Normalization is such that value of unity represents fully coherent integration. The maximum value for each target velocity v_0 is found by computing the velocity slice $\mathrm{FAF}(v_0; v)$ and $\mathrm{AF}(v_0; v)$ from Eq. (3.91), and searching its maximum value. The top panel corresponds to 100 ms integration time, the bottom panel to 300 ms integration. It is seen that when the target velocity increases, both the AF maximum and the FAF maximum approach the level one half. But that level can be obtained even when one of the two frequency channels is completely killed-off.

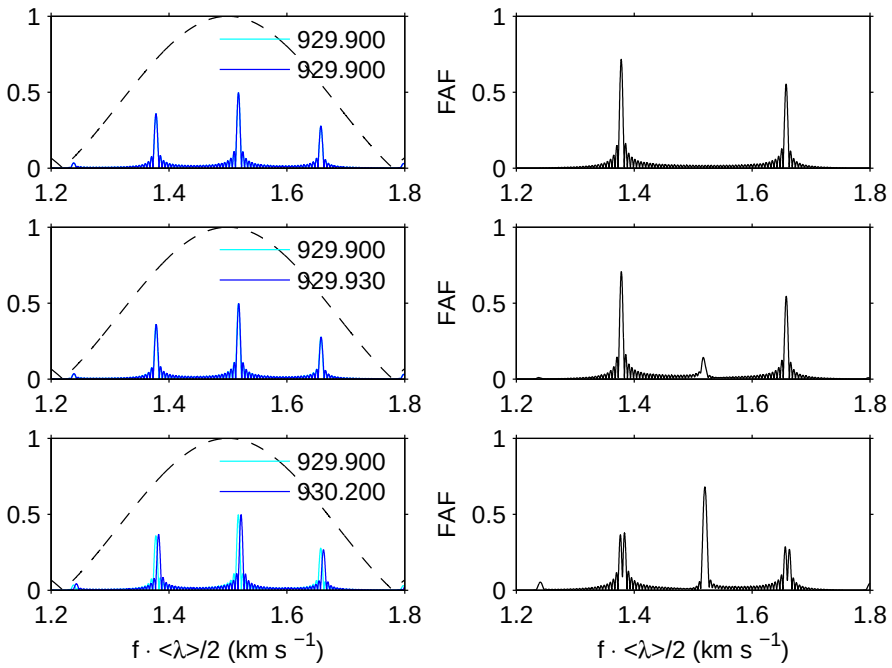

Figure 3.13.: If a single-frequency experiment is changed to two-frequency experiment by alternating the frequency from pulse-to-pulse, the distance of the maxima in the FAF is halved. This series of plots of $f \mapsto \mathrm{FAF}(v_0; f)$ shows how this change comes about when the frequency difference between the channels is increased, starting from zero. The frequency axes has been converted to velocity units by multiplying the frequency by half of the average radar wavelength. The left-hand-side panels show the magnitude of the two complex-valued terms, C_1 and C_2 in Eq. (3.91), shown in blue and cyan curves, which are summed to form the FAF, shown in the right-hand-side panels. The dashed curve is $\mathrm{diric}(\omega_0 - \omega, L)$. The top panels show the single-frequency case, the bottom panels the actual case in the tau2 experiment. Integration time is 300 ms, target velocity is $1.500\,\mathrm{km\,s^{-1}}$. With these parameters, the two Doppler-frequencies in the bottom panel are separated by 3.0 Hz, while the FMF peak's width, counted from zero to zero, is $2/(300\,\mathrm{ms}) = 6.7\,\mathrm{Hz}$. Therefore, the maxima of C_1 and C_2 are becoming well-separated. This reduces the maximum possible value of the FMF from 1 to about 0.7.

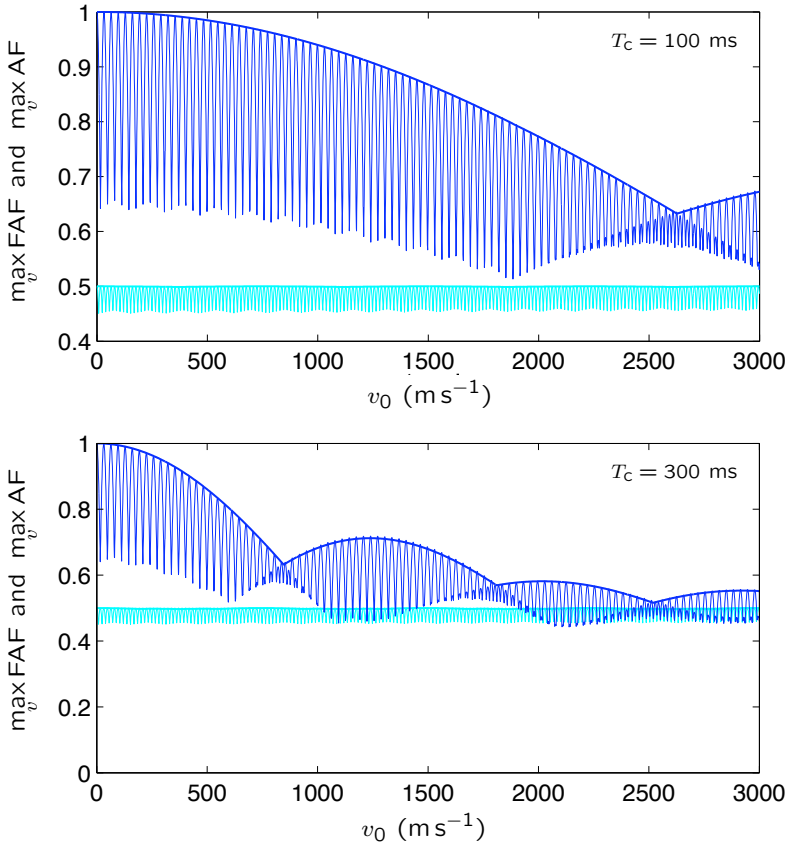

Figure 3.14.: Failure of coherent integration in standard (dual-frequency) tau2 measurements. The curves show the maximum value of the FAF (fast oscillating dark coloured curve) and the AF (slowly oscillating envelope curve, with larger line width) as a function of the target's radial velocity v_0. Normalization is such that unity represents fully coherent integration. The top panel corresponds to 100 ms integration time, the bottom panel to 300 ms. With increasing target velocity, both max AF and max FAF approach the single-channel level. The single-channel case is shown with the lightly coloured curves at the near-constant value of about one-half; for that data, the value 0.5 represents fully coherent integration. The single-channel curves were computed by setting one of the two frequency channels in the tau2 transmission pattern equal to zero.

4. Software

4.1. Overview

An overview of the real-time SD data processing software is shown in Fig. 4.1; recall also Fig. 2.1 on p. 19. The system consists of four main processing units and an overall control system.

The SD receiver board's firmware delivers baseband complex samples to an onboard FIFO, which is also visible in the memory space of the measurement computer.

The recorder program GUMP, provided by the hardware vendor, reads the data from the FIFO and dumps them to disk files. The disk, typically an external 250 GByte FireWire disk, is network-mounted from the analysis computer to the measurement computer. The data are organized into directories which we call the stream directories, or just the streams. Typically, one stream contains 60 minutes of uninterrupted sample data, in time-stamp labeled files, each containing one million complex points.

The streams are processed, one stream at a time, by the scanner program DSCAN. Two DSCAN processes can be running simultaneously in the two-processor analysis computer, nearly doubling the processing speed. The scanner reads a segment of raw data from a stream and searches through the segment for hard targets using the FMF algorithm. When a pre-determined detection threshold is exceeded, we say we have a hit. The scanner saves the hit's description to a file and proceeds to the next data segment. Scanning is by far the most time-consuming step in our standard data analysis. Therefore, DSCAN is implemented as a C program that makes use of the AltiVec vector processor onboard the G5 measurement computer, by calling routines in Apple's DSP library (vdsp). The scanner performance depends strongly on the length of the input data vector. For our most common measurement configurations, we get a little over 2 GFlops per processor.

The next program in the processing chain, the event archiver DARC, inspects the stream's list of hits, trying to combine to an event the hits that correspond to a single target passing through the radar beam. Having determined the time boundaries of the event, the archiver copies the raw data belonging to the event to a separate event directory, and goes looking for more events. The event archiver is also a C program, but it is not performance critical. Most of its time goes to data copying, so its speed is mainly limited by disk speed. We have saved all raw data from most of our test measurement campaigns so far—somewhat less than a terabyte—but in routine measurements, at most the raw data of events will be saved. With the event rates observed in the test measurements, saving all events from all the $\sim$400 hours of space debris measurements that we anticipate to be able to do annually, would require (only) about a terabyte of storage per year.

As the last processing step, the analyser program DANALYSER picks a stream's events from the event directories and deduces and saves the event parameters. The way to compute the final target parameters is still under development. What the analyser now does is basically to call DSCAN to re-scan the data using FMF or MF, but with a maximum time and range resolution over a narrow range interval, and then make linear

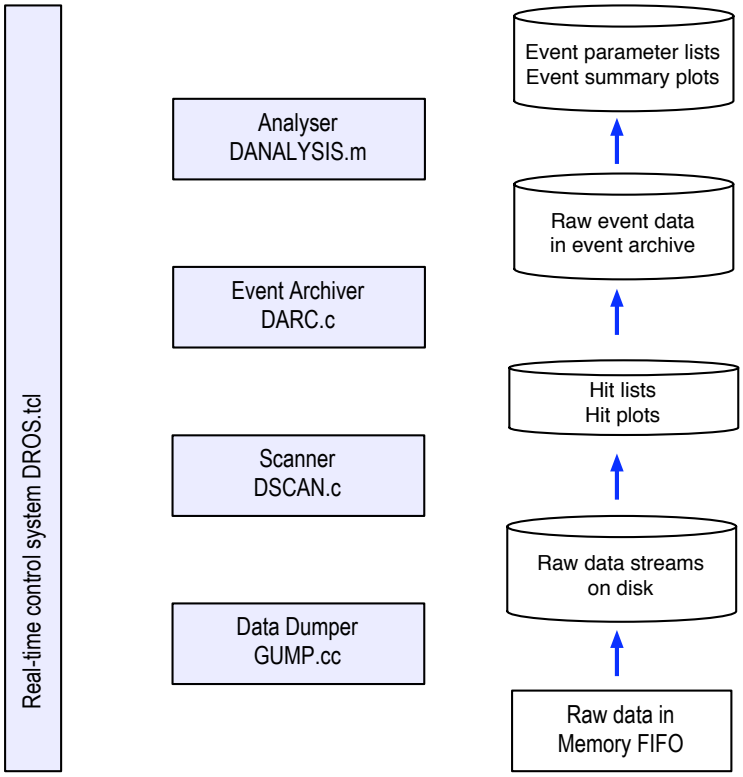

Figure 4.1.: Main modules of our real-time SD data processing software.

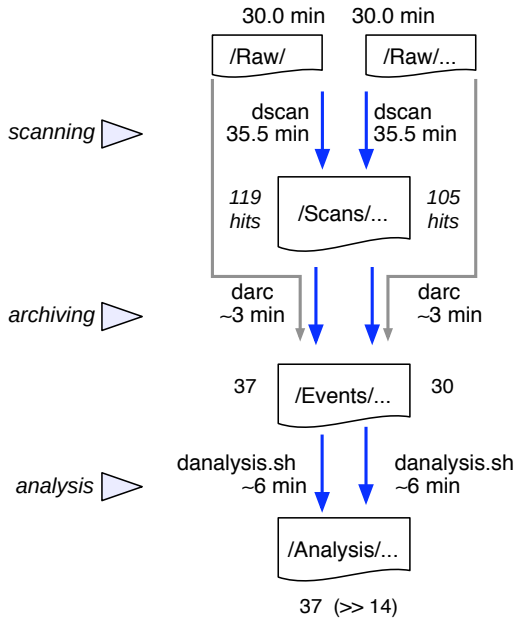

Figure 4.2.: Data processing software's performance under benchmark load. The figure shows the time required by the scanner, archiver, and analyzer, running on the analysis computer, to handle 60 minutes of data, composed of two 30 minutes streams on a disk. During the test, new data was being transferred with the benchmark rate of 26.8 GBytes h^{-1} from the measurement computer to the analysis computer. The two streams were processes in parallel by the two processors of the G5 workstation, and both required about $35.5 + 3 + 6 = 45$ minutes to complete.

or quadratic fits to the range and Doppler-velocity time series. The range and velocity parameters that we normally quote are taken from these fits, for the time instant of maximum signal strength. The analyser is a Matlab program.

The combined processing speed is such that for data taken with the 2 MHz benchmark sampling rate, it takes 40–45 minutes to scan, archive and analyse one hour of raw data, while keeping the raw data access running at the same time, see Fig. 4.2. Then we need to make use of both CPU's in the analysis computer. For our more typical 500 kHz sampling rate, we need about 20 minutes to handle one hour, and then we use only a single DSCAN for the scanning.

The four processing blocks are independent programs that are run as independent, stand-alone UNIX timesharing processes, which do their specific job once and then die. The processes themselves do not know anything about each other. The processing chain is created and organized by software we call DROS. The name is a twisted form of "EROS", and is meant to indicate that the DROS system is a slightly tailored copy of the standard EISCAT real-time radar operating system of that name. Based on an experiment-specific configuration file and a given start time, the DROS system generates the required input files and command line parameters for the processing modules, starts and restarts the processes in the two computers as required, and maintains and logs state information. The DROS system can query the running EROS at the host radar to find antenna pointing

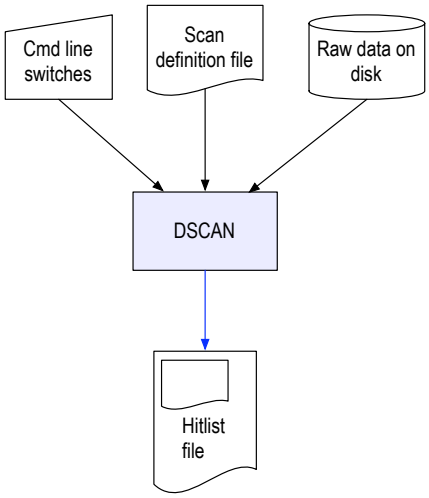

Figure 4.3.: The SD scanner program DSCAN.

direction and transmission power information directly from the official source. We now describe the processing blocks in more detail.

4.2. Scanner

The task of the space debris scanner program DSCAN (Fig. 4.3) is to read raw data, the complex samples, from disk, to inspect them for targets using the MF or FMF algorithms, and to save to file information about the location of the target-containing data, together with initial estimates of the target parameters. We call the process of looking the raw data for targets scanning.

4.2.1. Scandef file and hitlist file

We introduce some further terminology. The MF algorithm is applied to a segment of raw data at a time. The time length of the segment is the integration time, T_c. Processing one such segment is a scan. Typically, we use $T_c \approx 0.3$ s, and then the input data vector consists of several hundred thousand complex samples. The operation of the DSCAN program is configured in a scan definition file, for which we use the file name extension .sdef.

When the threshold detector of DSCAN triggers, we say we have a hit. The collection of hits caused by a single target going through the radar beam is an event. When a hit occurs, DSCAN both outputs to screen and records to a file an initial estimate of the range and velocity of the target, as well as the value of the MF maximum. In addition, DSCAN also records the stream position of the data segment that contains the hit. The file, into which the hits are recorded, is the hitlist file. It has the extension .hlist. The hitlist file also inherits a complete copy of the currently active scan definition file.

4.2.2. G-streams

The raw data, produced by the SD data acquisition system, are in binary files as 16+16 bit complex integers, 1,000,000 points per file. The files are named like xxx_00000, xxx_00001, and so on, by the GUMP recorder program. We call this set of equally sized files of uninterrupted sample data a stream. It is required that no samples are missing, and that there are no extra samples. Especially, if for any reason the data access is stopped and then restarted, always a new stream is created, with different xxx, and again starting from file number 00000.

We have written a set of interfacing C routines to mask out the file boundaries in a stream, so that the calling C program like DSCAN sees only a single, long data vector, and can conveniently access any part of it. The interface routines have names reminiscent of the streams routines in the standard C library, but with the letter 'g' prefixed to the name. We have GOPEN, GSEEK, GREAD, GCLOSE, and some others.

An addition to the g-streams compared to the standard C language streams is that a g-stream has, conceptually, a time stamp associated with every sample, so that the calling program can get the time instant of any piece of data, with a nominal microsecond accuracy. The time information is inserted into the stream as a required parameter when the stream is created, in the GOPEN call. DSCAN gets the time information either directly from the scandef file, or, normally, the scandef file provides a pointer to the time-stamp file that GUMP creates when it starts a new stream. A stream is the basic unit in our data processing. A single call of DSCAN processes one stream.

Another addition of convenience is that the GREAD routine performs transparent conversion from the raw 16+16 bit complex integer format, where every other 16-bit quantity is the real part and every other the imaginary part, to floating-point split-complex format, where the real part and the imaginary parts go to two separate vectors as 32-bit floats. This format is required in order to be able to use vectorized library routines in the Apple's signal processing library.

4.2.3. The DSCAN program

We now describe in detail the space debris scanner main program. The numbers in the left margin in this section refer to the DSCAN main program listing in appendix B on p. 111.

1 DSCAN is a normal UNIX command-line program. Like many UNIX programs, it requires command line parameters and supports some options. The present version supports the options shown in the DSCAN online-help message in Table 4.1, printed out by the command 'dscan -H'.

2 DSCAN operation is controlled by a scandef file, which is a required parameter when DSCAN is invoked. An example of a scandef file is shown in Table 4.2. The example is the template file, /deb/kst/debris/sdef/tau2_2000.sdef, which the DROS system uses to generate scandef files for tau2 experiment, by modifying the '$name', 'file1' and 'time1' entries to point to actual places in the filesystem.

The scandef file consists of lines having a keyword and a value, and any number of empty and comment (%) lines. The value can also be a Matlab-format vector, but one should not attempt to insert anything fancy there. Some explanations about the various entries follows.

Table 4.1.: DSCAN usage and options.

```
Usage: dscan_mat [-options] ScandefFile
         -H : Print this help message
         -R : Save a cycle of Raw data for hit scans
         -RR: Save a cycle of Raw data for all scans
         -S : Save Ratio(r) for hit scans
         -SS: Save Ratio(r) for all scans
         -T : Test mode (no hitlist written)
         -V : Save peak V-slice for hit scans
         -VV: Save peak V-slice for all scans
         -a : Syncronize to iper boundary
         -c : Check sync in every scan
         -i : Request confirmation before starting scanning
         -t : Detailed performance timing
 -N nscans  : Set maximum number of scans
 -o dirname : Output directory for hitlist and .mat files
 -n name    : Set and/or override $name in sdef file
```

The scandef file contains some information about the EISCAT experiment, like the locations of transmission samples (entries IPPlen, TXon, TXlen), and the length of EISCAT radar cycling period and the size of a possible idle time at the end of the period (gapsize, iperlen). These numbers are in terms of samples, not, say, microseconds. The scandef file contains pointer to the begin of the stream to be scanned (file1). Further, there is the time instant as a six-component Matlab vector (yyyy, ... ,sec), or, as here, name of the file where the time instant can be read (time1).

There is information to actually control the scanner. The entry "ncycles" determines the length of coherent integration, the entry "nskipcycles" determines how much data to skip over after an integration before starting next integration. A *cycle* is a subunit of the full EISCAT radar period. It consists of an integer number of transmission-reception periods or IPPs. The length of the IPPs can vary in a radar period, but a cycle has a constant length, and is thus better suitable for address counting. For example, some old versions of the tau2 experiment, which we had to work with, used two different lengths of IPPs, 6516 μs and 6504 μs. In principle the cycle structure can be deciphered from the transmission ON/OFF bit included in the stream data, but in practice we have extracted IPPlen and TXlen by studying EISCAT experiment definition files. The entry "shift" defines the range gates to be scanned. The set of shifts need not be uniformly spaced. The entry "noiseshift" defines which part of the received data should be used to perform background noise estimate. We want to ensure that even if there is a target in the data, the target does not contaminate the noise estimate. Our strategy is to define several segments, compute the noise estimate (of length TXlen) from each of those, and use the minimum as the actual estimate.

The last-but-one item in the example scandef file is the detection threshold definition. The simplest possibility would be to give a single number here, to define a constant, range-independent, threshold. However, during much of the time, it is impractical to use a constant threshold. The typical range-dependent threshold that we have been using for the tau1 scans of October 2003 data is shown as the red line in Fig. 4.4. The ratio-profile (top panel of Fig. 4.4) is pretty much constant

Table 4.2.: Scandef file for tau2, using 2 μs sampling in the SD receiver.

```
% A scandef template file for space debris scanner
% For experiment tau2_pl 1.00, with 2 us sampling
%
%   IPP = 5580 us, cycle = 11160 us, loop = 357120 us.
%   5 sec integration: 14 loops = 4999680 us + gap 320 us
%
%   F13 929.9  IF2 = 10.1 MHz
%   F14 930.2  IF2 =  9.8 MHz
%   576 16x32us code
%
$name           RADYYYYMMDDhhmmss
file1           /XXX/$name/$name_00001
time1           /XXX/$name/timestamp.data
offs            -2
fnum1           0

gapsize         160
iperlen         2499840
iperoffs        0

ccflag          1

fradar          930.00
expid           tau2_2000
tau             2.0

IPPlen          [ 2790 2790 ]
TXon            [ 46 46 ]
TXlen           [ 288 288 ]

ncycles         28
nskipcycles     16

usefastgmf      1
blocksize       10
decim           4
accflag         1

shift           [ 1150:5:2450 3200:5:5250 ]
noiseshift      [ 1500 2000 3250 4000 ]

threshold       topside ( 1150, 1670, 7, 5, 5.0 )
maxvel          5000
```

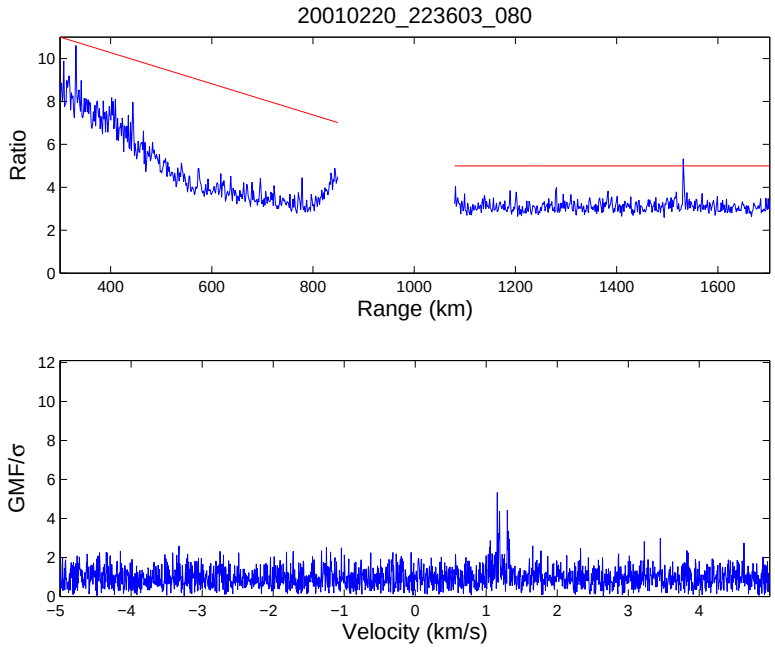

Figure 4.4.: Range-dependent detection threshold. Ionospheric clutter is quite pronounced below about 500 km altitude. Using a constant threshold would be problematic in conditions like this.

from about 500 km range upwards, but has a bump in the beginning. This bump is ionospheric "clutter" . The size, shape and location of the bump depends on the state of the ionosphere and can change significantly in the course of a few minutes. We would sacrifice too much sensitivity if we were to insist on having a spatially constant threshold. Using a constant threshold, perhaps about 12 to 13 in Fig. 4.4, would of course mean that the hit shown in the figure would be missed.

It is not clear how we should best handle detection sensitivity variation caused by the changing (both in range and time) clutter level. In most of our work we have used the "topside" range-dependent threshold as in Fig. 4.4, but not always with the same parameters. For statistical studies, it seems necessary to maintain a record of what actually has been the threshold at any given time and range. So far, we have kept the threshold-profile constant during a measuring campaign. As the scan definition is saved to the event directory, and we save the event directories, the threshold is in principle recoverable afterwards. Perhaps we should include the threshold specification in the analysis results also. Strong clutter obviously will have some effect also to the signal energy estimate, but that we have not attempted to take into account. There are not many events in the altitude region below about 500 km affected by clutter, anyway.

3 Information in the scandef file is moved to a data structure of type SCANDEF, defined in the header file scannerlib.h. This structure is used to build the actual scan control information for DSCAN runtime use. This entails computing the sizes and creating the work areas for intermediate products, computing the range dependent acceleration values, computing TXon times for each IPP, computing the

FFT fiddle factors, and so on. These data are kept in a single structure of type GMFSETUP, defined in the header file gmflib.h. DSCAN prints out a summary of the parameters in the GMFSETUP structure before starting the actual scanning. For example, when the DSCAN is called manually in interactive mode (option -i) using the UNIX command line command

```
%> dscan_mat -iac -RVS -n UHF20041112040001 -o testscan tau2_2000
    .sdef
```

with a scandef based to the template of Table 4.2, the program prints to the screen the messages shown in Table 4.3, then waits for permission to proceed.

Table 4.3.: DSCAN startup messages.

```
sampling     = 2000 ns
n to read    = 158988
nipps        = 56 (28 cycl)
integr       = 156240 (312.480 ms)
skip         = 89280 (178.560 ms)
n shifts     = 672
shift step   = 5 (1.500 km)
shift 0      = 1150 (345.0 km)
shift end    = 5250 (1575.0 km)
blocksize    = 10
use_fastgmf  = 1
n fftin      = 4032
fftlen       = 4096 (2^12)
gmflen       = 2033
decim        = 4
velo max     = 5001 m/s
velostep     = 4.92 m/s
0-velo       = -0.000 m/s
cc tx        = 1
accflag      = 1
acc 0        = 163 m/s^2 (-1.3e-08)

  Iper boundary at: 2004-11-12  04:00:05.005983  (file 1 offset
     335020)

Gapseek took 142 ms (9.0 MB/s)
Start position
  /Volumes/FWB/0411/Deb0/tau2_2000/UHF20041112040001/
     UHF20041112040001_00001:5640 (0 2170620)
  2004-11-12  04:00:04.347223
  TX at offset 0

Output directory = testscan
Start scanning (Y/N):  n
```

4 The MF method requires that the location of the transmission samples in the sample stream is known precisely, so that the transmitted code can be extracted correctly for the MF model functions. An EISCAT transmission has a strictly repetitive structure, with period called the experiment integration period, typically 5,000,000 μs (not to be confused with the coherent integration time T_c in the MF method). If one once locates the start of an integration period in a sample stream, and knows the location of transmission samples within it, it is possible to calculate a priori the positions of transmission samples in the stream for all in-

stants of time. The required information is available from the EISCAT experiment definition files. The only real problem is how to find the position of the start of an integration period in the stream. The subroutine SYNCHRONIZE achieves this, based on the lengths of the IPPs in the EISCAT integration period. Normally, there are only one or two different IPP lengths that are cycled during the integration period, followed by a single final IPP that is significantly longer than the others. The expected lengths of all the IPPs can be found from the EISCAT experiment definition files. SYNCHRONIZE finds the particular IPP that marks the end of an EISCAT integration period by determining the locations and lengths of the IPPs in the data by inspecting the transmission ON bit of the sample stream. Note that we neither require nor use the actual transmitted codes from the EISCAT files, only the locations of the transmission segments.

It would be possible to manage entirely without the initial synchronization to the EISCAT integration period. We could always use the transmission bit to find the location of the transmission samples at the time when new data is read in for coherent integration, for all IPPs. This was actually the original idea, and although the search takes some time, it might be worthwhile to implement the synchronization that way. Then no knowledge of the EISCAT transmission cycling would be needed a priori. This option remains to be studied.

5 Once SYNCHRONIZE has determined the locations of the transmission samples, the stream can be opened for actual scanning and positioned at the precise start of an IPP. Subsequent positioning can then be based on sample counting. When the stream is opened, it is also required to specify the time instant of the first sample at the beginning of the first file of the stream, Also, the sampling interval must be specified.

6 The purpose of scanning is finding hits in the data, so that the useful data can be stored for later, detailed analysis, and the other data thrown away. DSCAN records the hit locations to the hitlist file. The hitlist file name is formed automatically, based on the name of the scandef file. The "detailed analysis" basically means to scan with better resolution. DSCAN saves the information required for the detailed scanning directly to the hitlist file, so subsequent processing can use the hitlist file as control input.

7 At this point, the scanner is initialized, so actual search for the hits can start. The scanner operates in a loop, reading data, searching, and recording hits, until it is either interrupted by the operator, or the end of the g-stream is found.

8 Even though it should be possible to base data block addressing within a stream entirely to sample counting, once the stream has been initially synchronized, we do not completely rely on this. One can trust that a computer can do its integer arithmetic correctly even for large integers. But the data access system may be causing problems, and there might be missing samples, or extra samples may be generated. This initially happened in the SD receiver. So we check that the transmission edges are precisely—within a sample or two—in the places we expect them to be. Then we can also trust that the target ranges can be derived accurately.

9 The first step in a scan is to read in the raw data required by the MF method. We read first an integer number of cycles and then a sufficient number of "extra"

samples so that when we shift the transmission blocks to the right near the end of the data segment, there are received samples there, too.

10 The targets normally are visible in the beam for a few seconds, so it is possible to speed up the scanning by skipping over some raw data altogether. The amount of skipping is controlled by the "nskipcyles" in the scan definition file, and is here visible as the parameter "Nskip".

11 The quantity that we use for the threshold detection is the maximum value of the quantity we call the "Ratio". It is the $\mathrm{MF}(R, v)$, maximized with respect to the velocity variable, and normalized by r.m.s noise,

$$\mathrm{Ratio}(R) = \max_{\omega} \frac{|\langle z, \chi(R, \omega)\rangle|}{\|x\|\,\sigma}\,. \tag{4.1}$$

The nominator is evaluated once per scan. This strategy presumes that the noise is independent of time within a reception period, which we know is not a very good approximation.

12 In a scan, DCAN computes $\mathrm{Ratio}(R_j)$ for the required set of range gates j. The computation requires repeated evaluation of the match function. Our match function evaluation routines, GMF and FASTGMF, are written to handle several range gates (a *block*) in a single call. To compute the Ratio, we therefore have a loop, the BLOCK LOOP, that is executed Nshifts/Blocksize times, where Nshifts is the total number of ranges required, the length of the vector "shift", defined in the scandef file, and Blocksize is the number of gates in a block (that number also is specified in the scandef file).

13 Most of its time DSCAN spends in the match function evaluation routine, either GMF or FASTGMF, depending which one is specified in the scandef file. A single call to either routine returns, in the matrix gmf2, BlockSize rows, say the rows $j_1 \ldots j_2$ of the two-dimensional un-normalized MF matrix

$$\mathrm{MF}(R_j, \omega_k) = |\langle z, \chi(R_j, \omega_k, \alpha(R_j))\rangle|.$$

Each row is of length GmfLen, and represents a velocity slice of the MF at a fixed range. GmfLen is determined at DSCAN initialization time to cover the required set of target radial velocities, as specified by the "maxvel" parameter in the scandef file.

14 For the block $[j_1 \ldots j_2]$ of ranges, DSCAN next computes the Ratio.

15 After the block loops have been done, the Ratio is ready and can be compared to the range-dependent threshold, defined in the scan definition file.

17 It is possible to save the profile $R \mapsto \mathrm{Ratio}(R)$, the velocity slice $v \mapsto \mathrm{MF}(R_0, v)$, and a cycle's worth of plain raw data into Matlab .mat files for visual inspection in Matlab. The files are written to the directory specified by the -o option of DSCAN. For example, the option -SS causes the Ratio for every scan to be saved, and the option -V causes the velocity slices to be saved, but only from those scans that actually contain a hit.

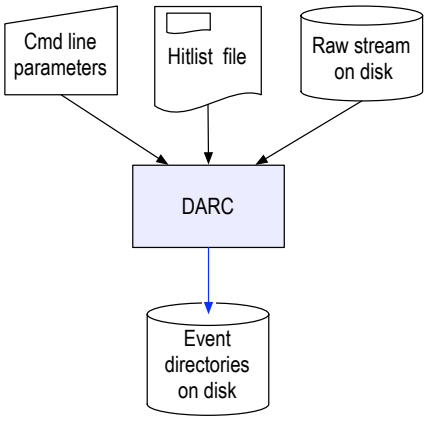

Figure 4.5.: The event composer/archiver program DARC.

18 As the last step in a scan, DSCAN prints some statistics to the screen, and if a hit occurred, saves the location of the hit and the preliminary basic target parameters, to the hitlist file. The hitlist file name is out.hlist, and the file is placed to the DSCAN output directory.

19 After printing out the scan statistics, DSCAN loops back for the next scan. The program normally exits (via the exit point in line 62 of the DSCAN listing) when end-of-stream is met. The program can also exit when a predefined number of scans has been performed or via an error. From many errors DSCAN exits gracefully, but there is also a rare problem that causes a bus error when DSCAN reads in event data during analysis high resolution re-scans.

4.3. Archieving–from hits to events

The scanner records the hits it finds into a hitlist file. The hitlist file is an input to the data archiver program DARC (Fig. 4.5). The primary purpose of the archiver is to save to permanent storage only those segments of the raw data that contain targets. However, it is not a good idea to, say, simply save those files that the scanner finds containing a hit. Normally, also the files adjacent to the hit will contain useful signal. Therefore, DARC tries to save enough files surrounding the hit to collect into a single place all the raw data files that contain, or may plausibly contain, the signal due to the target when it traverses the radar beam. There is no unique way to do that (even the concept of a radar beam is ill-defined), and a heuristic algorithm is used.

Our implementation of DARC will evolve, but at the moment the program groups hits to events based on their separation in time (for instance, two consecutive hits separated by more than 15 s are unconditionally taken to belong to different events), separation in range (two consecutive hits separated by more than 50 km of range are unconditionally placed to different events), and, optionally, separation in velocity. It is also possible, by a command line switch, to put DARC into a tracking mode, where it tries to check whether the change in range from hit to hit corresponds to the estimated velocity and the known time interval.

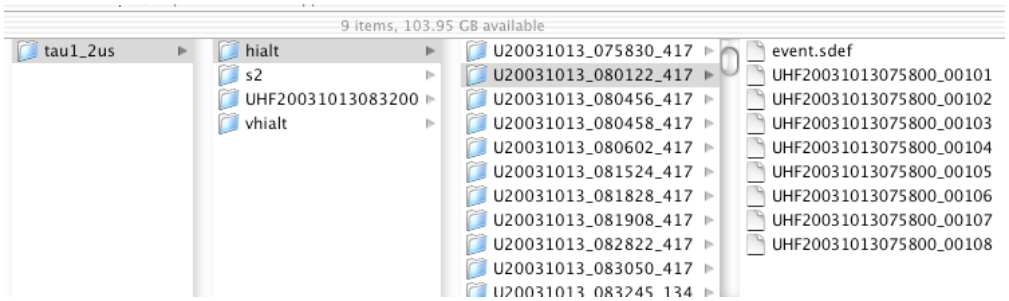

Figure 4.6.: Event directory hierarchy. Based on a hitlist, the event archiver DARC determines the raw data files belonging to an event and copies the files to directories, one event per directory. In addition to the raw data files, DARC copies also the relevant part of the hitlist file to the event directory. Only those hits belonging to the event are copied, in addition to the scan definition part. By convention, this hitlist file is named as "event.sdef".

Thus, in the first phase DARC creates (and optionally prints out for interactive inspection) a list of event start and end hits. Then a fixed amount of time is added to both ends of the implied segment of the raw data stream. In the last phase, DARC uses the expanded list to copy files from the raw data directories to the event directories, such as those shown in Fig. 4.6. The original raw data can then be deleted. Parameter estimation uses the raw data from the event directories.

An example DARC run is shown in Fig. 4.7. The program is fast. Almost all of the time goes to copying the files, so the speed is mainly limited by hard disk performance. With our typical event rates, storing permanently only the events and not all of the raw data saves considerable amount of disk space. In the example shown in Fig. 4.7, the 28 saved events consume 378 MB, while the original 3600 s of raw data required 6870 MB.

Manual cleaning of hitlists

A problematic point in the data processing is the interface between detection and analysis, the step that involves grouping the detector hits into events. This step is handled by the event-archiver program DARC. If DARC manages to produce a "good" event, the analysis programs can in most cases make reasonable sense of it automatically. But we have found that we can vastly improve the event selection by looking the data—the ratio profiles and DSCAN output hitlist files—interactively. If we do not perform this interactive step, the analysis results contain an unacceptably high proportion of dubious events. Even if we could remove the bad events in, or after, the analysis phase, we would be wasting resources by archiving and processing the invalid events.

We seem to have arrived to one of the situations where a computer program is badly handicapped compared to the trained human eye. The most convenient phase for the human intervention in the present case is when we still have all the raw data available. This is unfortunate, for then the trained human is required to intervene fast, before disk space runs out.

There seems to be no easy way out. For example, just trading out some sensitivity by raising the detection threshold does not solve the problem. We probably need to teach the computer some of the heuristics the trained human would be using to select the good events, and see what comes out of it.

```
/scans/oct03/hialt/UHF20031013083200.out> darc -d /events/timingtest_events *.hlist

Event   1/28  U20031013_083245_134       22-26     10.0 s     15.3 MB    1.0 s
Event   2/28  U20031013_083833_134      196-199     8.0 s     11.4 MB    0.7 s
Event   3/28  U20031013_084229_134      314-317     8.0 s     11.4 MB    0.8 s
Event   4/28  U20031013_084819_134      489-493    10.0 s     15.3 MB    0.9 s
Event   5/28  U20031013_085029_134      554-558    10.0 s     15.3 MB    1.0 s
Event   6/28  U20031013_085339_134      649-652     8.0 s     11.4 MB    0.8 s
Event   7/28  U20031013_085645_134      742-744     6.0 s      7.6 MB    0.6 s
Event   8/28  U20031013_085719_134      759-763    10.0 s     15.3 MB    0.9 s
Event   9/28  U20031013_085725_134      762-767    12.0 s     19.1 MB    0.9 s
Event  10/28  U20031013_085743_134      771-774     8.0 s     11.4 MB    0.7 s
Event  11/28  U20031013_085835_134      797-799     6.0 s      7.6 MB    0.6 s
Event  12/28  U20031013_090101_134      870-874    10.0 s     15.3 MB    0.9 s
Event  13/28  U20031013_090201_134      900-903     8.0 s     11.4 MB    0.8 s
Event  14/28  U20031013_090649_134     1044-1047    8.0 s     11.4 MB    0.8 s
Event  15/28  U20031013_090651_134     1045-1048    8.0 s     11.4 MB    0.4 s
Event  16/28  U20031013_090651_134     1045-1048    8.0 s     11.4 MB    0.2 s
Event  17/28  U20031013_090653_134     1046-1050   10.0 s     15.3 MB    0.5 s
Event  18/28  U20031013_090859_134     1109-1112    8.0 s     11.4 MB    0.8 s
Event  19/28  U20031013_090931_134     1125-1128    8.0 s     11.4 MB    0.7 s
Event  20/28  U20031013_091041_134     1160-1164   10.0 s     15.3 MB    0.9 s
Event  21/28  U20031013_091239_134     1219-1222    8.0 s     11.4 MB    0.7 s
Event  22/28  U20031013_092043_134     1461-1465   10.0 s     15.3 MB    0.9 s
Event  23/28  U20031013_092131_134     1485-1489   10.0 s     15.3 MB    0.9 s
Event  24/28  U20031013_092151_134     1495-1503   18.0 s     30.5 MB    1.6 s
Event  25/28  U20031013_092401_134     1560-1564   10.0 s     15.3 MB    1.0 s
Event  26/28  U20031013_092503_134     1591-1594    8.0 s     11.4 MB    0.7 s
Event  27/28  U20031013_092611_134     1625-1628    8.0 s     11.4 MB    0.7 s
Event  28/28  U20031013_092755_134     1677-1680    8.0 s     11.4 MB    0.7 s

Saved 378 Mbytes (198.00 s) data in 22.1 s
```

Figure 4.7.: A DARC run on 3600 s of tau1 2 μs data.

Even a cursory look at the analysis summary plots reveals many cases that have only a single point in the time series, just above the detection threshold. Many of the suspicious events are also near the edges of the search range (there are four edges in tau2 search range, because the used range 195–1575 km contains a blind zone between about 735 and 960 km).

If one then looks at the Ratio curves that the scanner uses to perform the threshold detection, one finds that, indeed, many of the near-edge events probably are associated with targets that actually are outside the search range. These events must be disregarded. Also, looking at the Ratio curves and the hitlists produced by the scanner, one sometimes finds that in the middle of a clear, ongoing event, due either to the target's internal scintillation or to the target moving to a low-gain part of the antenna beam pattern, the target vanishes for a while. During that moment of a weak echo, some spurious spike elsewhere in phase space can grab the attention of the event archiver. The result is that a set of hits, which really represents only a single event, gets split into two or perhaps even three separate events. These are treated by separate events in the analysis, which will distort the event count statistics.

For the time being, we have resorted to going graphically through the Ratio and velocity slice files produced by the DSCAN. The Matlab program that we use for this purpose is also connected to the stream's hitlist file, so that we can comfortably (by a mouse click) remove the bad-looking hits from the hitlist. It normally takes less than an hour to clean 24 hours of raw data. We remove the obviously non-belonging, spurious-looking spikes; and the spikes near the search range edges; and the hits that most probably are caused by range-aliased targets (these have a rather conspicuous signature also). Then we run the archiver and analyser using the manually cleaned hitlists as DARC input. The cleaning helps the archiver to make correct grouping decisions, and no other manual intervention is required to produce sensible-looking results. For example, instead of 67 events found when using the original, un-cleaned hitlists in the performance test run shown in Fig. 4.2, the archiver after manual hitlist cleaning picked only 16. From those 16, the analyser disregarded two as too weak, recording only 14 true events to the result files. These numbers are representative.

4.4. Analysis—from event data to event parameters

The basic idea in the event parameter estimation is to perform a high resolution re-scan of the event's raw data, in order to produce time series of range, velocity, and MF maximum value, and then make linear or quadratic fits to the time series points to get $R(t)$ and $v_D(t)$. The re-scanning is done with DSCAN, while the fitting and the associated graphics is done with Matlab routine DAN.M, from our previous study. These tasks are done under Matlab control, by the function SD_ANALYSE_EVENT, as charted in Fig. 4.8. Matlab can invoke external UNIX commands like DSCAN in a standard way, so SD_ANALYSE_EVENT prepares a suitable scan definition file for DSCAN—based on the event's hitlist file event.sdef—launches the scanner, and reads the scan results from a hitlist file. Then it calls the fitting and plotting m-file DAN.m, to find and plot the event parameters. The plots we call event summary plots, and the file were the event parameters are written has the standard name eventlist.txt. The summary plots are accumulated to a single multipage postscript file, events.ps. The eventlist.txt and the events.ps file are meant for quick-look purposes. The actual results are saved into an event-specific result directory. The current version of the analysis software saves the

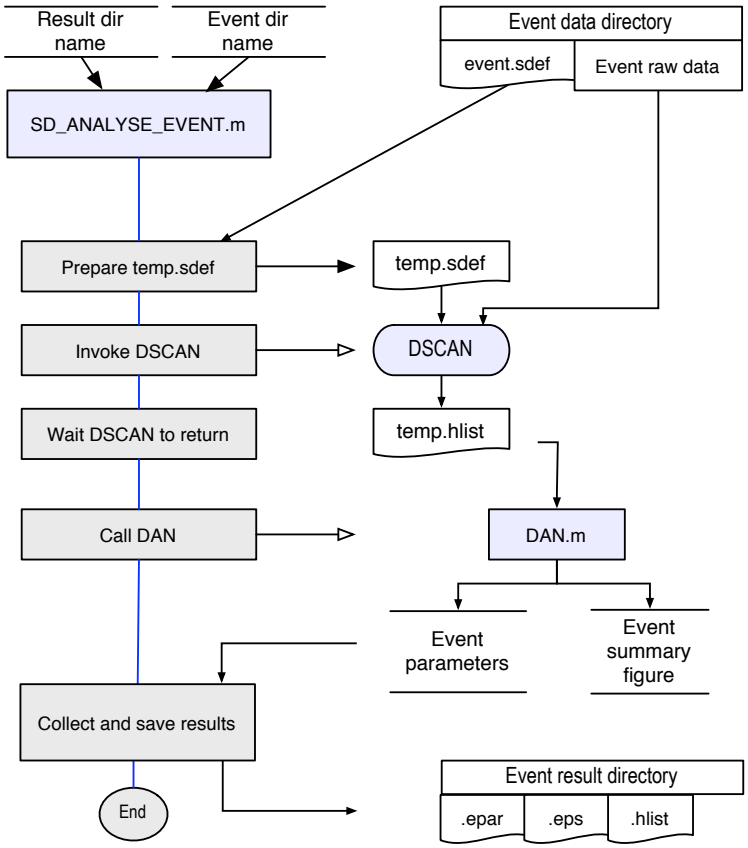

Figure 4.8.: Flowchart of the event analyser m-file.

event summary plot as an encapsulated postscript file, and saves the plotted time-series data also in numerical form, simply by copying the DSCAN output file temp.hlist to the event's result directory. The event parameters are saved to a separate event parameter file, for which we use the extension '.epar'. An example of such an event parameter file is shown in Table 4.4. The file has a simple format, so that it is readily machine-readable; but it is also human-readable; and it should be easy to add more parameters into it, like some error bounds, as they become available.

Table 4.4.: Event parameter file U20040907_165038_519.epar.

```
%       NM  Event name UT.                                                  1
%       XI  Experiment ID string.                                          2
%       TM  UT of max Ratio.                                               3
%       ST  System temperature K.                                          4
%       AG  Antenna gain dB.                                               5
%       WL  Radar wavelength m.                                            6
%       PW  Transmission power MW.                                         7
%       AZ  Azimuth degr, N=0, E = 90.                                     8
%       EL  Elevation degr.                                                9
%       RT  Max Ratio ( = estimate of sqrt(SNR_N)).                        10
%       RG  Range km.                                                      11
%       RR  Range rate (km/s).                                             12
%       VD  Doppler velocity (km/s), positive away from radar.             13
%       AD  Acceleration from VD, m/s^2.                                   14
%       DI  Effective diameter cm.  Estimated from ST,PW,RT,RN,AG,WL.      15
%       CS  Lower bound of radar cross section, cm^2. Estimated as DI.     16
%       TS  (Transmission sample power)/(Noise power)                      17
%       EN  Event number.                                                  18
%                                                                          19
%       NaN = Bad.                                                         20
%                                                                          21
%       25-Nov-2004 18:37:43                                               22
%                                                                          23
NM = U20040907_165038_519                                                  24
XI = tau1_2000                                                             25
TM = 2004  9  7  16 50 43.848                                              26
ST = 100                                                                   27
AG = 48.1                                                                  28
WL = 0.323                                                                 29
PW = 1.20                                                                  30
AZ = 133.3                                                                 31
EL = 61.6                                                                  32
RT = 311.8                                                                 33
RN = 523.841                                                               34
RR = -3.261                                                                35
VD = -3.181                                                                36
AD = 84.73                                                                 37
DI = 5.31                                                                  38
CS = 14.3286                                                               39
TS = 18.73                                                                 40
EN = 22                                                                    41
```

4.5. Real-time control of SD measurements

Normal ionospheric measurements at the EISCAT radar sites are performed by running an experiment-specific script under the radar's real-time control system EROS. Similarly, a space debris measurement is performed in practice by running a specific "experiment

debris.elan

```
MAIN { } {
 # 1. Define campaign name.
 # 2. Define log file used by recorder.
 # 3. Define scanner output root directory.
 # 4. Define event archive root directory.
 # 5. Define analysis result root directory.
 # 6. Wait for recorder/scanner commands.
}

BLOCK Recorder {exp sampling side length} {
  # Start raw data recording loop
}

BLOCK Scanner { args } {
 # Process options
 while {1} {
    # 1. Pop topmost stream from FIFO
    # 2. Find a free SCANNER.sh and
    #    launch it on the stream
    # 3. Sleep for a while
 }
}
```

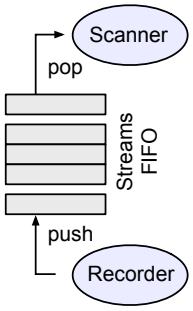

Figure 4.9.: SD measurement as a DROS experiment. The experiment script debris.elan is reminiscent of all other EISCAT experiment scripts. The script first defines the various places in the filesystem used in the data recording and analysis, and then goes to an idle loop, where it waits for an operator command to start/stop the Recorder and the Scanner subroutines, defined in the script. The operation of the two subroutines is organized around a FIFO data structure, which consists of g-streams of raw data. The recorder inserts new 1-hour steams into the FIFO, and the Scanner removes them as fast as it can. The Scanner subroutine calls the master UNIX shell script SCANNER.SH, which handles the stream by calling the DSCAN, DARC and ANALYSER processing blocks.

script", debris.elan, under an adapted "EROS", the DROS system. An outline of the script that we have been using in our recent campaigns, debris.elan, is shown in Fig. 4.9.

The the whole processing chain—scanning, event archiving and parameter estimation—is controlled via scripts, launched by DROS. The master do-it-all UNIX shell script is DSCANNER.sh. The script first invokes the DSCAN scanner on a stream and waits DSCAN to get the scanning done. Then the script invokes the event-archiver DARC to group the hits into events, and copy the raw data associated with the hits to event-specific directories. When this is done, the script invokes the Matlab script SD_ANALYSE_EVENT to perform parameter estimation. The estimation results are written to an eventlist file, and a summary plot is prepared also. When all this is done, DSCANNER.SH returns, and can be called again, to process another stream. Two copies of the script and its child processes can be running simultaneously.

4.6. Processing speed

The most time-consuming part of the data processing is the scanning done by DSCAN. An example of DSCAN performance during real operation is shown in Table 4.5. The table shows the standard performance info reported by each DSAN after a stream has been processed. The measurement was done during the campaign in March 2004. The EISCAT experiment was tau1 and the SD receiver was sampling with 2000 ns sampling interval. The scanners had no problem in keeping up with real time. Each scanner processes 0.3 seconds of input data and then skips 0.2 seconds. According to the table, it took less than 200 ms to scan this 1.0 second of input data, while new data was being collected at the same time.

Table 4.5.: DSCAN speed when scanning tau1 2 μs data with FMF. Both the recorder GUMP and two DSCAN scanners were active simultaneously. The recorder writes sample data at the rate of $2 \cdot 10^6$ Bytes/s over the network to the same FireWire disk on the analysis computer from which the two DSCAN scanners read it. The processing speed is about 4000 MFlops.

```
scanner 1
  7318 scans -- 1400.0 sec -- 98 hits
  Time per scan : 191 ms
  Read per scan : 41.6 ms 0.60 MBytes 14.3 MBytes/s
  Time per gate : 0.260 ms
  FMF per gate  : 0.199 ms 0.398 MOp 2000 MFlops
scanner 2
  7318 scans -- 1405.2 sec -- 135 hits
  Time per scan : 192 ms
  Read per scan : 41.5 ms 0.60 MBytes 14.3 MBytes/s
  Time per gate : 0.261 ms
  FMF per gate  : 0.200 ms 0.398 MOp 1993 MFlops
```

Table 4.6.: DSCAN speed when scanning tau2 0.5 μs data with FMF. Benchmark conditions (0.3 s integration), with raw data recording running onto the same disk in the G5 analysis computer from which the scanners read their data. This test was done in Sodankylä using 30 minutes of tau2 500 ns data for each scanner, originally recorded 9:30–10:30 UT, March 10, 2004. The processing speed is about 2600 MFlops.

```
scanner 1
  3654 scans -- 2111.4 sec -- 119 hits
  Time per scan : 578 ms
  Read per scan : 161.1 ms 2.38 MBytes 14.8 MBytes/s
  Time per gate : 0.860 ms
  FMF per gate  : 0.611 ms  0.811 MOp 1327 MFlops
scanner 2
  3654 scans -- 2114.4 sec -- 105 hits
  Time per scan : 579 ms
  Read per scan : 161.2 ms 2.38 MBytes 14.8 MBytes/s
  Time per gate : 0.861 ms
  FMF per gate  : 0.613 ms  0.811 MOp 1324 MFlops
```

We have not recorded timer information during the test campaigns when using faster sampling rates. Instead, we have timed the whole DROS processing chain under bench-

mark condition in Sodankylä. The GUMP recorder just recorded noise, but the data read-in by the DSCANNER.SH was real data from the campaign. The performance measurement therefore is representative of real operations in all essential respects. When the input data was one hour of tau2 500 ns data, originally recorded between 9:30 and 10:30 UT, March 10, we got the following performance. The DSCAN scanners' statistics is shown in Table 4.6. According to the table, both scanners required about 35.5 minutes to handle their 30 minute set of raw data. The FMF algorithm proceeds at about 1.3 GFlops on both processors. After the scanning, we thus had 24.5 minutes left from the hour for other tasks. The DARC archiver grouped the 119 and 105 hits produced by the DSCANs to 37 and 30 events, and used about three minutes to copy the event raw data from the FireWire disk to the event archive on the internal hard disk of the analysis computer. Finally, the analysis, running on two DSCANNER.SH processes, took about six minutes to complete. The analysis produced 37 "final" events, out of its $67 = 37 + 30$ input events. Thus, the whole processing of the 60 minutes of raw data took about 45 minutes from beginning to the end.

No manual intervention was used in this speed measurement. As was therefore to be feared, in this test both the archiver and the analyser performed a lot of futile work, which could have been avoided if DARC would have been more up to its task. We point out in the next chapter that there appears to have been only 14 real events in the data. True archiving and analysing processing time requirement for this data set is thus more likely $(14/67) * (3 + 6) \approx 2$ minutes instead of 9 minutes.

Speed of the match function and fast match function algorithm as implemented by the GMF and FASTGMF routines, called by DSCAN, is shown in Tables 4.7 and 4.8. The timing measurements were done using the 1 GHz G4 Mac that we normally use as the measurement computer. The 2 GHz G5 Mac where the DSCAN is run in normal configuration is two times faster. With our standard 300 ms integration time, almost all of the processing time in the standard MF goes to the long (2^{18} to 2^{20} points, depending on the used sampling rate) FFT. We call the FFT_ZIP routine in Apples vdsp library, which uses the AltiVec vector processor onboard the G4 and G5 systems. For the cache-limited problem, we get (only) about 400 MFlops in the FFT on a 1 GHz machine. But for data vectors that fit into the cache, the AltiVec-boosted FFT is fast; for the four kiloword FFT that we normally use in FMF, we get about 5000 MFlops in the 1 GHz machine. Our timing might not be quite reliable here, though, due to the very short time spend in a single fft call. We use the Apple DSP library also for complex multiplication, but there the speed is somewhat of a disappointment (plain C code which does not touch the AltiVec was only about twice as slow).

4.7. Verification of the C implementation of the scanner

In the precursor study we had implemented the MF and the FMF algorithms with Matlab. Here we show that the new C language implementation reproduces the earlier results. We will also show that our new, more automated processing does not lose us much sensitivity. We re-processed with the new software the two data-sets collected during the previous study, and compared with the old analysis. The data sets are both from February 2001, one which we label feb01-cp1-600 and the other that we label feb01-tau2-500, the notation indicating the EISCAT experiment name, and the SD receiver sampling interval in nanoseconds.

Table 4.7.: Detailed performance when DSCAN is scanning tau2 0.5 μs data using the MF, with 312 ms integration. The test was done on a 1 GHz G4 Mac. The speed is determined by the speed, 410 MFlops, of the 2^{20} point FFT. Time per range gate is 310 ms, with mean computation speed of 390 MFlops.

```
Scan parameters
  n to read     635804
  n ipps        48 (24 cycles)
  n per tx      1152
  integration   624960 (312.480 ms)
  skip          312480 (156.240 ms)
  n shifts      692
  shift step    20 (1.500 km)
  shift 0       6000 (450.0 km)
  shift end     22700 (1702.5 km)
MF parameters
  fftlen        1048576 (2^20)
  mflen         32477
  decim         -
  velo max      5000 m/s
  velostep      0.31 m/s
Scanner timing
  Read/scan     99.8 ms 2.38 MBytes 23.9 MBytes/s
  Time/gate     310.4 ms
  MF/gate       300.5 ms 112.0 MOp 373 MFlops
MF internal timing per range gate
  Operation             N         Op      us    MFlops
  ------------------------------------------------------
  x(t)*y(t)       48*1152     332000    3270       102
  X(t)*e^(iat^2)  48*1152     555000   14900        37
  FFT                2^20  111149300  271000       410
  Total                    112040000  289000       387
```

Table 4.8.: Performance of DSCAN when scanning tau2 2.0 μs data using FMF, with 312 ms integration. Th test was done on a 1 GHz G4 Mac. Time per range gate is 0.47 ms, with mean computation speed 1000 MFlops.

```
Scan parameters
  sampling    2000 ns
  n to read   158988
  n ipps      56 (28 cycl)
  n per tx    288
  integration 156240 (312.480 ms)
  skip        89280 (178.560 ms)
  n shifts    686
  shift step  5 (1.500 km)
  shift 0     1150 (345.0 km)
  shift end   5250 (1575.0 km)
FMF parameters
  n fftin     4032
  fft length  4096 (2^12)
  fmf length  2033
  decim       4
  velo max    5001 m/s
  velostep    4.92 m/s
Scanner timing
  Time/scan  323 ms
  Read/scan  35.3 ms 0.60 MBytes 16.9 MBytes/s
  Time/gate  0.470 ms
  FMF/gate   0.403 ms 0.424 MOp 1053 MFlops
FMF internal timing per range gate
  Operation              N      Op      us    MFlops
  ------------------------------------------------
  x(t)*y(t)          56*288   96768     176       550
  Sum xy_n           56*288   32256     127       250
  X(t)*e^(iat_0^2)    56*72   24416(a)   74       330
  FFT                  4096  270486      48      5600
  Total                      423926     425(b)   1000
  --
  a) 56*4 + 4032*6
  b) Measured FMF/gate was 488 us.
```

Verification of the similarity of the results

Figure 4.10 shows the scanning result for the benchmark event of the previous study (Fig. 4.12 on p. 74 of [10]), when the FMF algorithm is used. Taking into account that there are some smallish differences between the two implementation, the agreement of all basic quantities (Ratio, range and velocity) probably is as good as can be hoped for. The implementation differences include the following.

- In the C implementation, the MF normalization by r.m.s noise to form the Ratio is done on scan-per-scan basis. In the Matlab implementation, the noise background may well get contaminated by a target, but we try to compensate this by (the cumbersome means of) estimating the contamination from a large set of recorded scans. Moreover, in the Matlab implementation, an average Ratio(r) is formed and subtracted from every individual Ratio, to reduce systematic "distortions" in the Ratio profile. This is not done in the C implementation.

- Acceleration correction in the FMF in the C implementation is done using only a single phase value per IPP. That is, for each IPP, we approximate the correction vector
$$\mathrm{e}^{i\cdot\alpha\cdot n^2}, \quad \text{for } n = n_1 \ldots n_2,$$
by a constant vector
$$\mathrm{e}^{i\cdot\alpha\cdot<n>^2}(1,\ldots,1),$$
where $<n>$ is the mean of n_1 and n_2. This reduces the number of complex exponentials that we need to evaluate by a factor of several tens at least, while still being sufficiently accurate for integrations that are some sizable fraction of a second.

- The internal double precision (64-bit reals) Matlab numerical accuracy is somewhat better than the single precision (32-bit reals) accuracy that the C implementation uses. Conceivably, this could become visible in the long FFTs.

- The C implementation performs FFT using vectorized code from Apple's DSP library. The library routines accept only input lengths in powers of 2, so we zero-pad our input vectors when necessary. In the Matlab implementation, we use whatever lengths the input vectors naturally are. This may slightly affect velocity resolution, and therefore also the Ratio.

We never had the patience to compute the event of Fig. 4.10 with the MF in the Matlab implementation—it would have taken about 24 hours. With the C-based MF, this became possible. In Fig. 4.11 we show the benchmark event computed both with the new MF and the new FMF, as invoked in DSCAN. Clearly, the two C algorithms give mutually consistent results, the MF being slightly more accurate as expected. Thus the C version of the MF probably is alright, too.

Verification of the detection sensitivity

Before considering the scan results, we need to point out that we cannot really expect precisely the same sensitivity here as we got earlier. In the earlier study, we went to great lengths to painstakingly scan and re-scan the data small segments at a time, to always

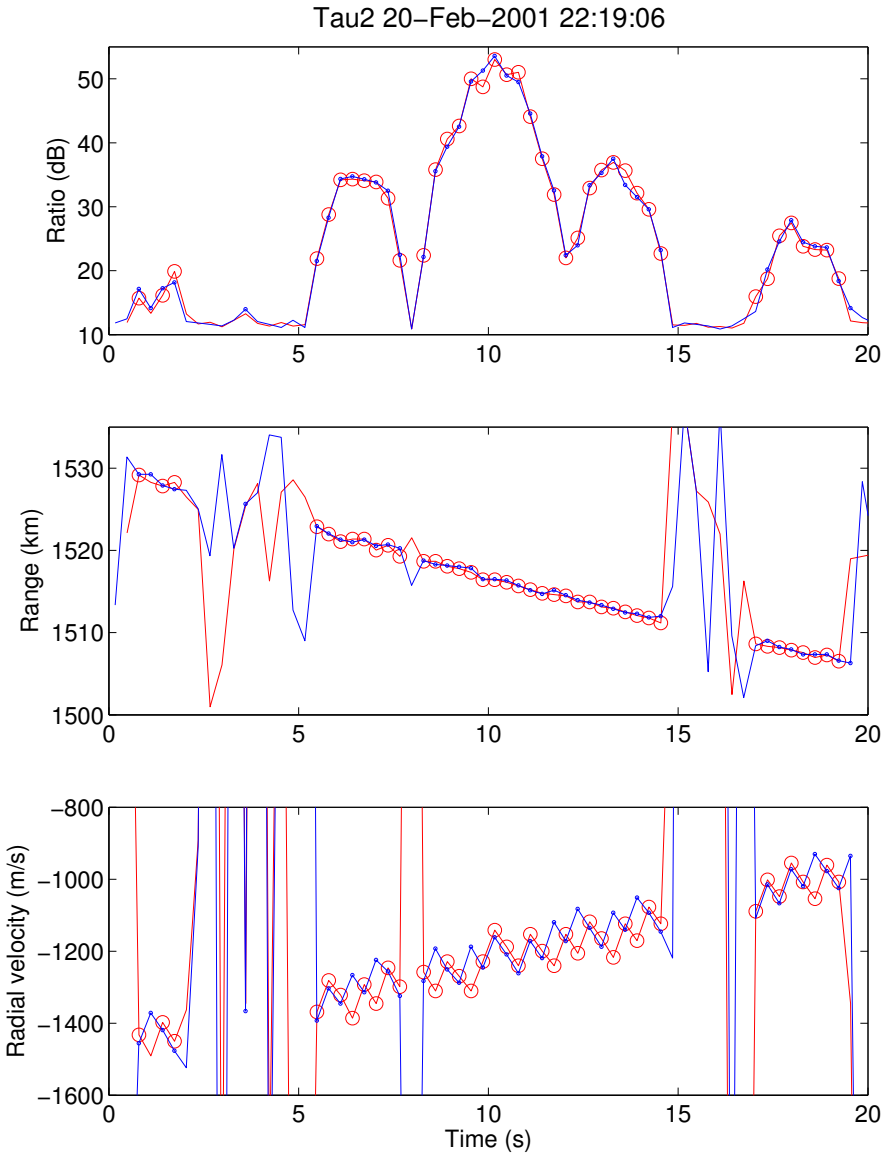

Figure 4.10.: Comparison of the C and Matlab implementations of the FMF algorithm. Red line with big circles is Matlab data, blue line with small circles were computed with the C implementation.

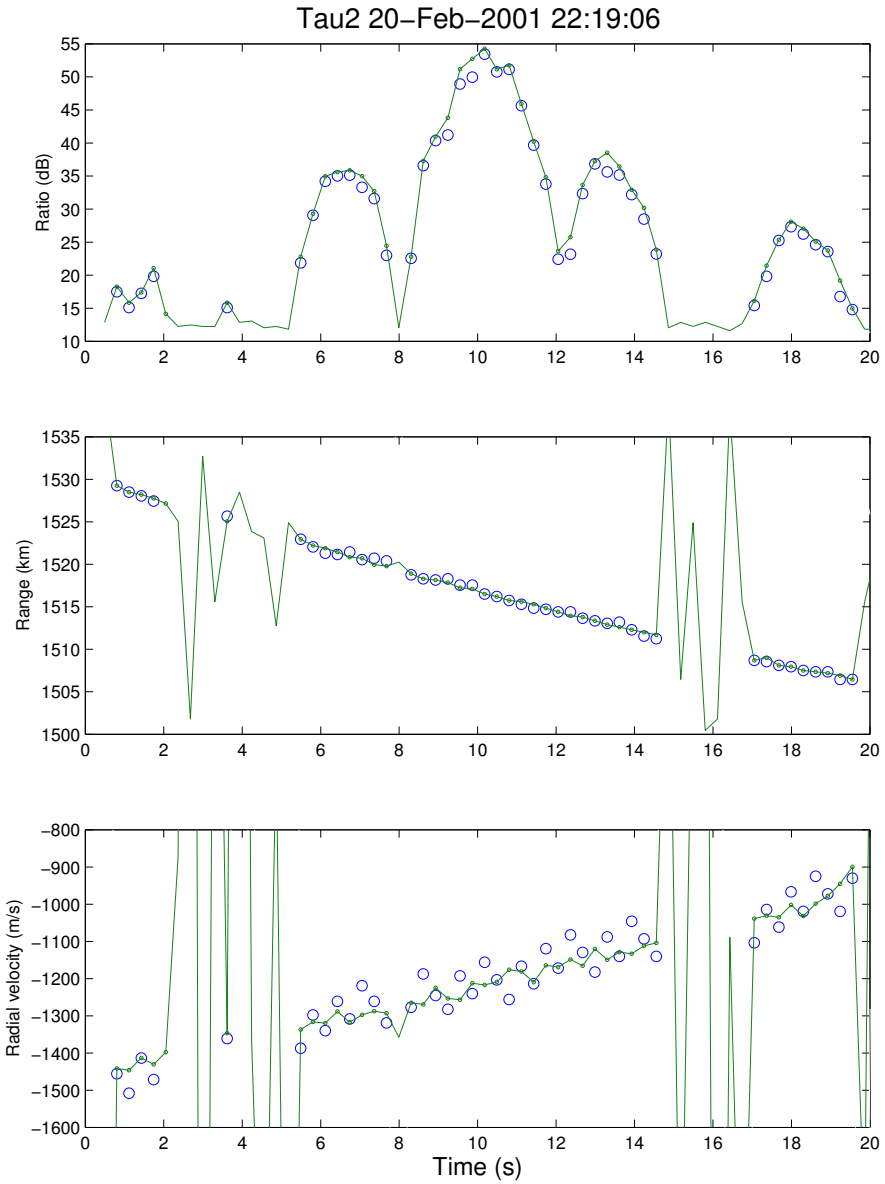

Figure 4.11.: Comparison of the C implementation of the FMF and the MF algorithms. Large blue circles are DSCAN output from FMF, the green line and the small circles are from the MF.

achieve maximum sensitivity. Basically, we were setting a very low detection threshold, hand-picked for the particular range interval, and then disregarding the numerous false alarms by manual inspection. Some loss of sensitivity was therefore to be expected.

We have only one stream of cp1 data available. Processing the 10255 seconds of data took only 5515 seconds on the G5, even when we were using only a single DSCAN process for the scanning. From the tau2 experiment of February 2001 we have now studied three data sets, of sizes 1242 s, 3704 s, and 1269 s, of which only the 3704 s set had been processed earlier.

We were able ultimately to detect all the 45 cp1 events that we had found earlier, and which are collected in Table C.1, on p. 105 of [10]. However, due to the manual cleaning of the hitlists, in one of our several analysis re-runs of the feb01-cp1 data sets, one of the old, week events was overlooked. On the other hand, in addition to the hits belonging to 43 old events, the re-analysis found 23 events that we have not reported earlier, see Fig. 4.12. Many of them were found now simply because of the larger range coverage, made feasible by the increased processing speed, but some of them should have been visible in the old scans also.

Of the 11 tau2 events found in the old scans in the 3704 s data set, listed in Table C.1, p. 105 in [10], our first re-analysis run found only 8. One event was just a trifle too small to exceed the rather conservative range-dependent threshold used in the re-analysis, but was clearly visible in the Ratio plot. The two other missing events (events 4 and 9 in the Table C.1), were simply too weak to show up now. We do not believe that this is is due to any real differences between the Matlab and C implementation. Rather, this is an example of the case where we must trade some detection sensitivity for the possibility to process data more automatically. However, because our contract agreement said that we should reproduce *all* the 45+11 events of the previous study, we of course did dig out the remaining two weak tau2 events also. With prior knowledge of their range this was possible, by scanning over a much limited range interval.

The occurrence of some new events and the missing of some old events underlines a problem in manual data processing: it is difficult to achieve the same level of repeatability as with a fully automated processing.

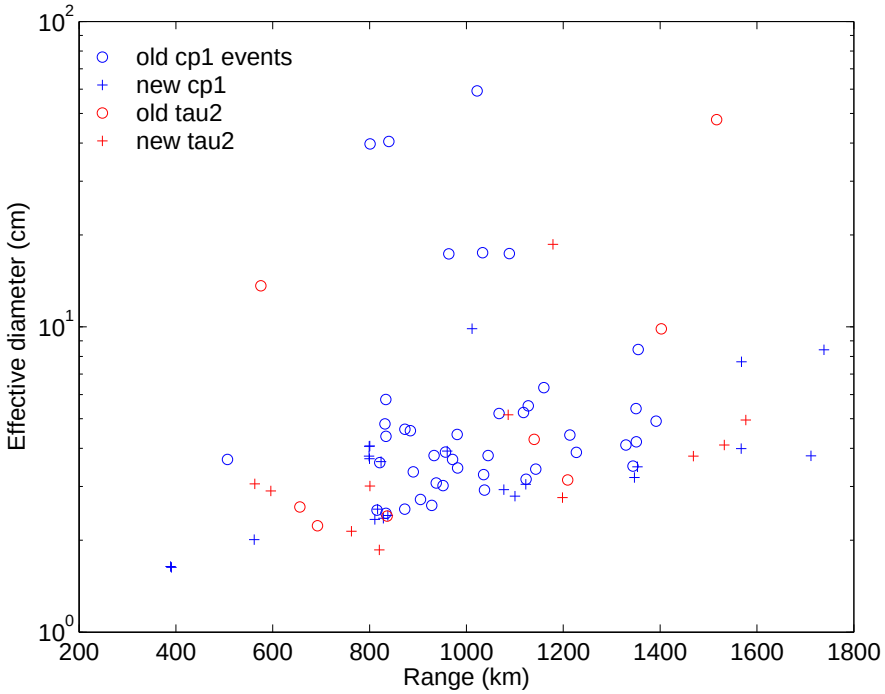

Figure 4.12.: Effective diameter versus range in re-analysed February 2001 data. The cp1 data was scanned using alternating codes only. From the 45 cp1 events catalogued in [10], the run shown here found 43. In addition, the run found 23 previously neglected events. The scanned ranges for cp1 were 300–630 km, 808–1403 km, and 1550–1810 km. In tau2, the plot shows 8 old events out of 11, and in addition 11 new events; ranges were 300-850 km and 1080–1700 km.

5. Measurements

In this chapter we show examples of the analysed results of the EISCAT measurement campaigns performed for this study. To incorporate late updates of the analysis procedure, we have for this report re-analysed all the data sets, both using the FMF and the MF algorithm. We have not re-scanned the raw data (even where we still have it available), though, but our starting point has been the set of events found in the original DSCAN runs done at the campaign times. One consequence is that the range coverage of the data sets vary, for we realised only late that it is useful to monitor higher-than LEO regions even when the primary interest is in LEO. In the future, we intent to monitor ranges up to and including the third reception window, but now we have done so only for the beam-park experiment data, such as shown in the top panel of Fig. 5.5 on p. 92. Even where we have data from higher ranges, in this chapter we mostly show the data only from the LEO region, up to an altitude of 2000 km.

In our measurement campaigns, all conducted at the EISCAT UHF radar in Tromsø, two EISCAT experiments were used, the so called tau1 and tau2 experiments. These experiments are representative. In fact, the two transmission patterns, when combined with user-defined ways of pointing the UHF antenna, have been pretty much the only experiment modes used at the UHF system in the last two or three years. The situation is not expected to change anytime soon, either. We recall the tau1 and tau2 transmission schemes in section 5.1. In section 5.2 we list our data sets: five sets collected during the two contract-stipulated test campaigns and the 2004 beam park campaign; and in addition, a sixth data set that we happened to be in the position to collect, outside our original schedule, very late in the project. Detailed analysis of the last mentioned, 100 hour data set is still pending, and will be relegated to future work. In sections 5.3–5.5 we plot the basic parameters of 2653 events in various ways. The plots are only for illustration of what kind of data we have; it is outside the scope of this work to make inferences about the properties of the target population.

5.1. The EISCAT transmissions tau1 and tau2

The tau1 transmission has, as EISCAT UHF ionospheric experiments go, an exceptionally long range, giving uninterrupted range coverage from about 200 km to about 1500 km. The transmission consists of alternating codes[1], each of duration $16 \times 60\,\mu$s. Two frequency channels are used interleaved. The interpulse period is $11160\,\mu$s. The duty cycle is 8.8%. Timing diagram of the transmission is in Fig. 5.1. The experiment's present transmission frequencies, EISCAT frequencies F13 and F14, are only 300 kHz apart, therefore, it is sufficient to band-pass sample the signal using 2000 ns sampling interval in the SD receiver.

The tau2 transmission is at the moment by far the most popular in EISCAT UHF experiments. The transmission cycle consists of 32 alternating codes per frequency

[1] An *alternating code* is a special binary phase code, developed for incoherent scatter applications by M Lehtinen and others. There are 32 different codes in the complete code set.

channel, each code of duration $16 \times 36\,\mu$s. Two frequency channels are used interleaved. The interpulse period is $5580\,\mu$s. The experiment has somewhat higher duty cycle, 10.3%, than tau1, and is therefore inherently slightly more sensitive. However, tau2 typically gives considerably lower total detection rate than tau1. As the timing diagram in bottom panel of Fig. 5.1 shows, that is because tau2 has a wide "blind zone" which includes the altitude region of maximum debris density around 850 km, when one is using a near-vertical antenna pointing.

5.2. The data sets

During this study, we performed two test measurement campaigns as stipulated in our work contract, the first in October 2003, the second in March 2004. In addition, we took part in the beam park 2004 multi-radar SD measurement campaign, by a run in September 2004. Finally, we measured four days during a standard EISCAT CP1 experiment in November 2004. The data sets are the following.

1. 10.6 hours of tau1 data, recorded in three intervals in October 13, 14 and 16, 2003, using 2000 ns sampling interval in the SD receiver. We refer to this data set of 232 events by *oct03-tau1-2000*. In the analysis, we assume transmission power 1.2 MW and system temperature 100 K for these data. The mean event rate is 21.9 events/hour.

2. 13.9 hours of tau1 data, recorded between 11:30 UT, March 4 and 01:00 UT, March 5, 2004, using 2000 ns sampling interval. We refer to this data set of 286 events by *mar04-tau1-2000*. The mean event rate is 20.6 events per hour. During the March campaign, both UHF transmitter klystrons were in use for the first time in the course of our SD work, and the transmission power was fairly stable, nominally at 1.5 MW $\pm$ 0.2MW during almost all of the recording time. The value 1.5 MW is used for all March 2004 data. This value might be an overestimate though, by as much as about 30%, for the value is based on the klystron high voltage readout and a torturous formula, calibrated for the previous single-klystron configuration. We use system temperature 100 K for all the March 2004 data.

3. 10.9 hours of tau2 data, recorded 01:00–11:00 UT, March 5, and 14:50-16:00 UT, March 10, 2004, using 2000 ns sampling interval. We refer to this data set of 146 events by *mar04-tau2-2000*. The mean event rate is 15.0 events/hour.

4. 5.4 hours of tau2 data, recorded 08:30–14:00, March 10, 2004, using 500 ns sampling interval. We refer to this data set of 102 events by *mar04-tau2-500*. The mean event rate is 18.8 events/hour.

5. 16.6 hours of tau1 data, recorded from 15:49 UT, September 7, to 08:29 UT, September 8, 2004, using 2000 ns sampling interval. We refer to this data set of 368 events by *sep04-tau1-2000*. The mean event rate is 22.1 events/hour. This data set is part of an international, multi-radar SD measurement campaign, so called beam park campaign. We use T_{sys} 100 K and transmission power 1.2 MW for this data set.

6. 100 hours of tau2 data, recorded from 09:02 UT, November 09, to 14:00 UT, November 13, 2004 using 2000 ns sampling interval, during a standard EISCAT

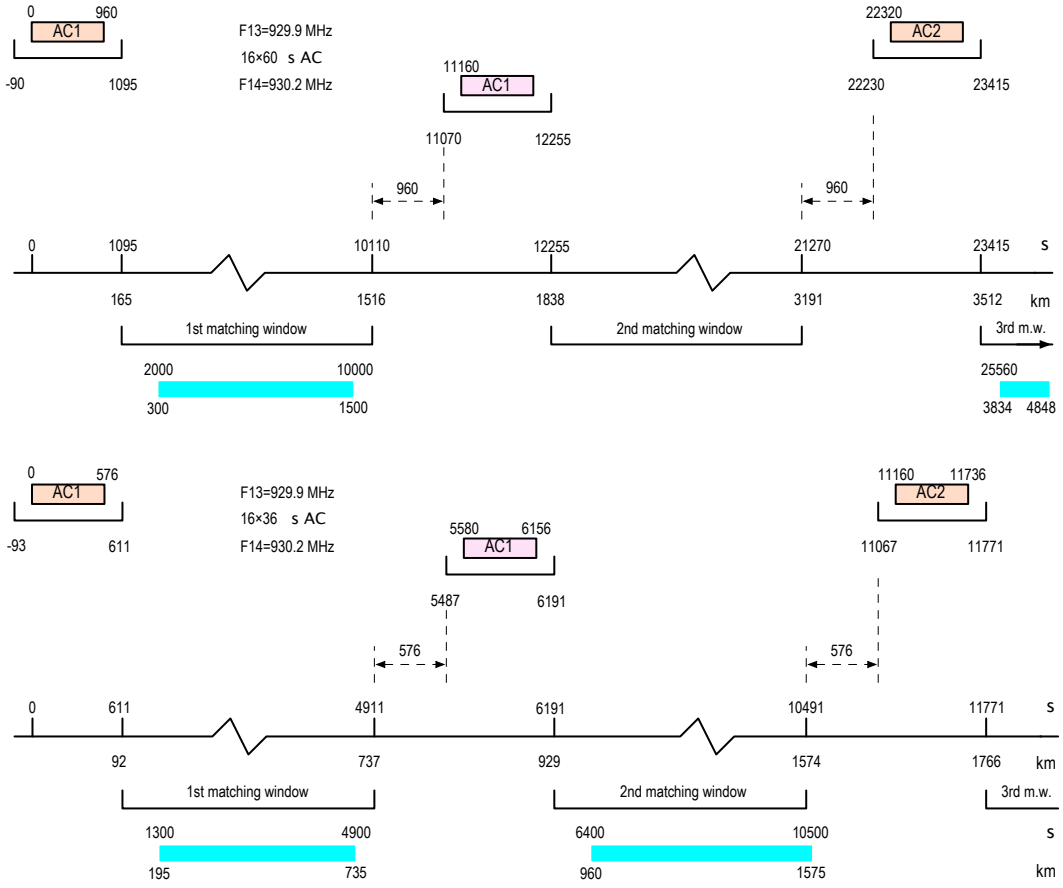

Figure 5.1.: Timing diagrams of the tau1 (top panel) and tau2 (bottom panel) EISCAT transmissions. Both transmissions use two frequencies and phase-code modulation. A complete transmission cycle consists of 32 different codes AC1 . . . AC32 on each frequency channel, so that the first truly ambiguous range occurs only after 64 interpulse periods (53,500 km and 107,000 km for tau2 and tau1, respectively). The diagrams indicate the time of transmissions, the interval of receiver protection around the transmissions, and the time/range windows inside which it makes sense to look for hard target echoes with the MF method. Note that the usable window is cut by a code length before the start of next receiver protection window, for we are not, in the present implementation of DSCAN, prepared to handle the case that the tail part of target echo would fall into the protection window. The blue stripes show the range intervals that DSCAN actually used in some of the data analysis described in this report; the stripes take into account the skipping done to avoid regions of strong ionospheric clutter.

ionospheric experiment. We refer to this data set of about 1500 events by *nov04-tau2-2000*. We use T_{sys} 120 K, transmission power 1.2 MW. According to EISCAT's (quite new, and not fully tested calorimetric) transmission power monitoring, the power varied between 0.9 MW and 1.4 MW during the run, but we have not attempted to account for the variation in the analysis. The anomalously high system temperature was caused, as it later turned out, by a loose connection in the second, uncooled, stage of the two-stage GaAsFet preamplifier unit. This is our first SD campaign where we have not saved the original raw data. Not having the raw data is lamentable, because we would now like to extend the range coverage of target search.

For the September 2004 data, the Tromsø UHF antenna (geographical latitude 69.6°N and longitude 19.2°E) was pointed to azimuth 133.3°, elevation 61.6°, for all other data sets, the antenna was pointed to the magnetic field aligned direction, azimuth 184.0°, elevation 77.1°.

5.3. Detection rate versus altitude

Figures 5.2 and 5.3 show target detection rate as a function of altitude for tau1 and tau2 data sets, in 50 km bins. The figures also indicate by cross-hatching the altitude regions where targets were not looked for by DSCAN. Bins adjacent to those regions partially overlap with them, and have an artificially low detection rates. In addition, there may be "selection effects" near the edges of the forbidden zones, due to the manual screening of detection hits. In the tau1 data, the well-known generic features of LEO SD data are nevertheless reproduced, with the peak around 900 km altitude and a maximum developing, but not fully visible, towards 1500 km altitude. The tau2 data (middle and bottom panel in Fig. 5.3) has maximum at 1000 km, but as is seen, is blind to precisely those bins, centered at 800 km, 850 km and 900 km, where the tau1 data have maximum. Nevertheless, the peak value is about 2.5 events/hour/50 km bin in both cases. This is not inconsistent with the expected greater sensitivity of the tau2 experiment. On the other hand, in the beam park tau1 data (bottom panel of Fig. 5.2), where we used somewhat lower elevation than in the other data sets, the maximum detection rate is at 1000 km altitude.

5.4. Effective diameter and RCS

Figures 5.5–5.10 illustrate the observed signal strength, in terms of the effective diameter and the lower bound $\mathrm{RCS}_{\mathrm{min}}$ of the radar cross section. The $\mathrm{RCS}_{\mathrm{min}}$ is computed from Eq. (3.22), using the measured $\mathrm{SNR}_{\mathrm{N}}$, the measured range, and known radar parameters. The effective diameter d_{eff}, corresponding to a given $\mathrm{RCS}_{\mathrm{min}}$, is the diameter of a conducting sphere that would yield the same cross section, but with the simplification that we ignore the resonant region in the standard cross section formula. We get the simplification by extending the optical region formula $\mathrm{RCS} = \pi d^2/4$ towards smaller sizes until we can start using the Rayleigh formula

$$\mathrm{RCS} = \frac{\pi d^2}{4} \cdot 9 \left(\frac{\pi d}{\lambda} \right)^4 \tag{5.1}$$

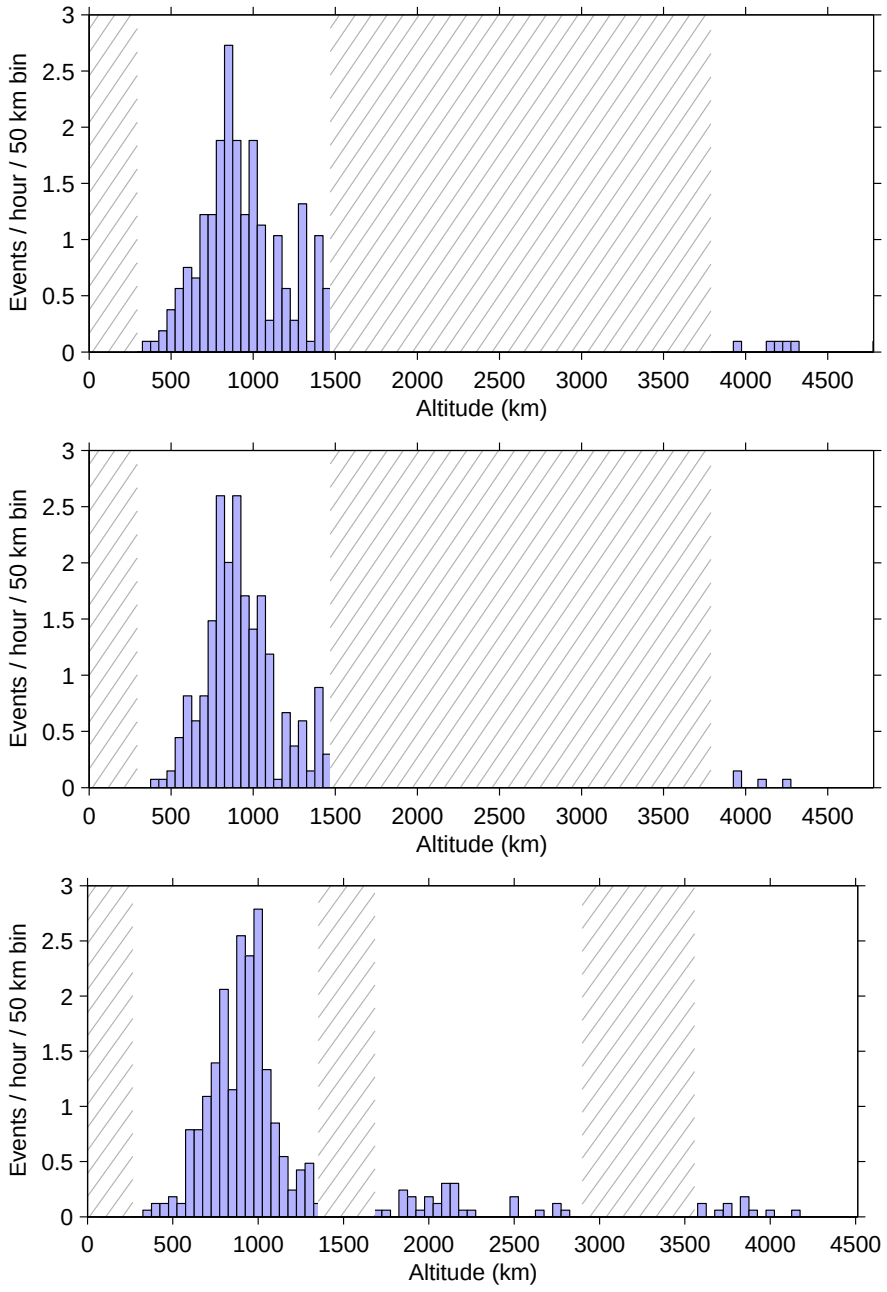

Figure 5.2.: Detection rate vs. altitude in tau1 experiments. The data sets from top to bottom are *oct03-tau1-2000*, *mar04-tau1-2000* and *sep04-tau1-2000*.

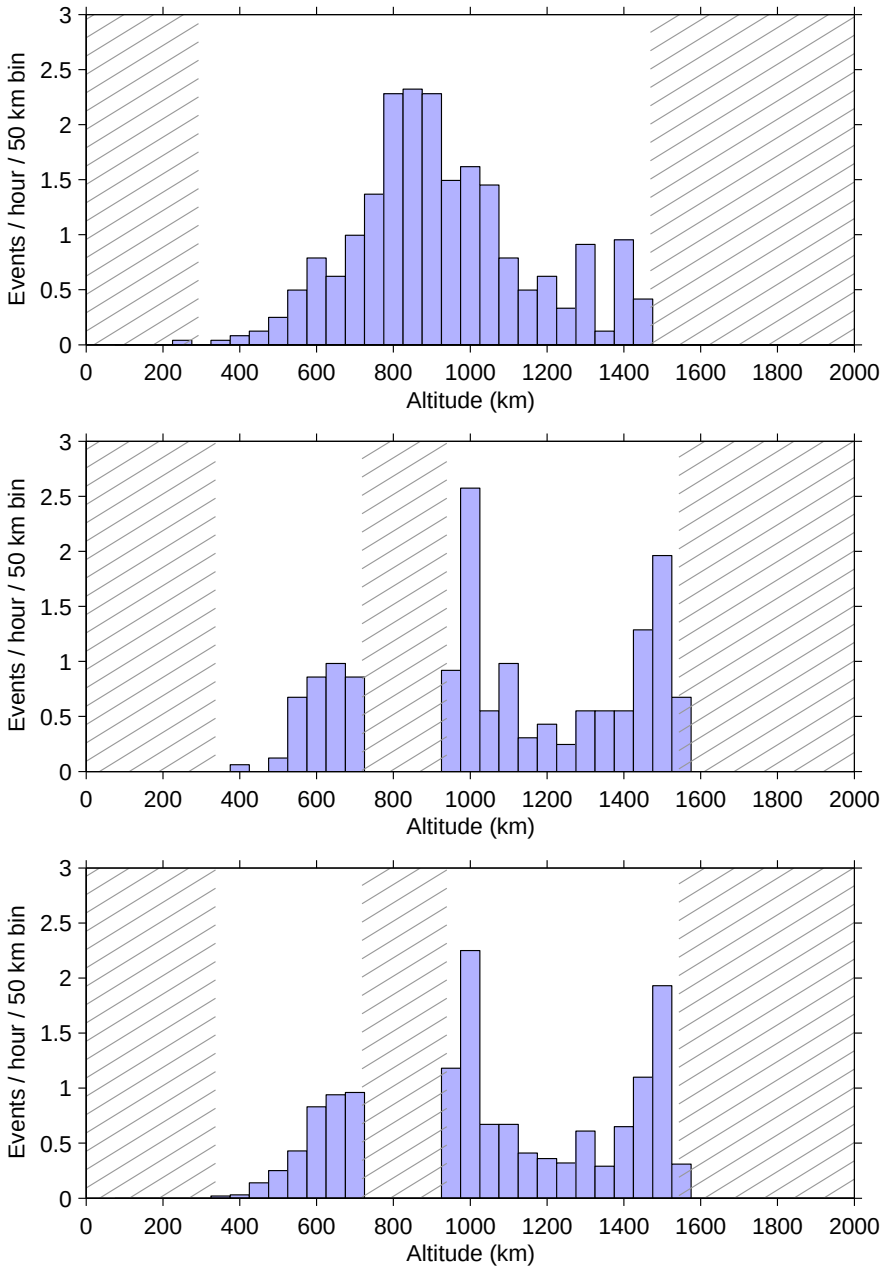

Figure 5.3.: Detection rate as function of altitude for field-aligned pointing. The top panel shows the combined data from sets *oct03-tau1-2000* and *mar04-tau1-2000*, total of 24.1 h and 514 events. The middle panel is combined data from the sets *mar04-tau2-2000* and *mar04-tau2-500*, total of 16.3 h and 247 events. The bottom panel is the set *nov04-tau2-2000*, total of 100 h and 1518 events.

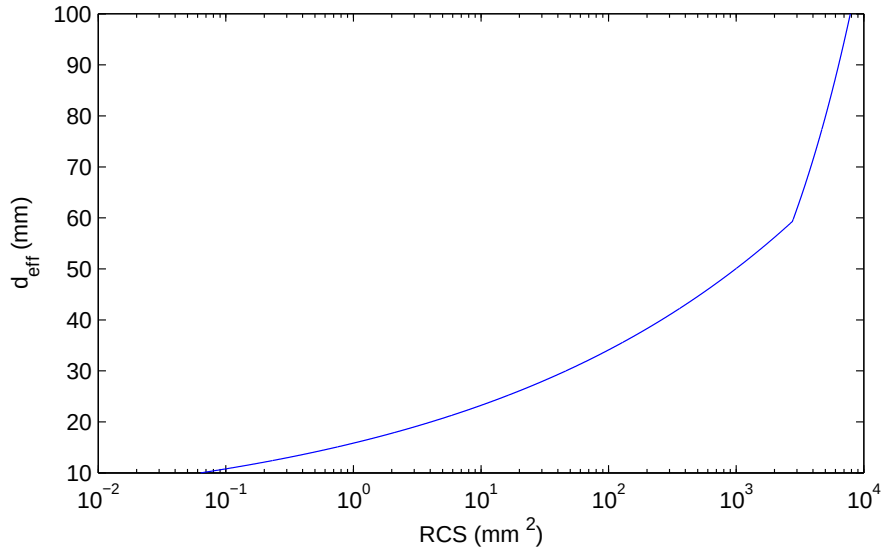

Figure 5.4.: Effective diameter as function of RCS for EISCAT UHF radar.

for the cross section of a small conducting sphere. For EISCAT UHF, the cross-over point where we change the RCS/d_{eff} relation is at $d = 5.9\,\mathrm{cm}$, RCS $= 27\,\mathrm{cm}^2$. The relation between RCS and d_{eff} for the EISCAT UHF is shown in Fig. 5.4.

Figure 5.5 mainly serves to illustrate the detection limit as function of range. The top panel in the figure also shows that the sensitivity is not improved by faster sampling. All panels in Fig. 5.5 show that our detection limit at 1000 km range is about 20 mm in terms of the effective diameter. Figure 5.6 shows the region near 1000 km range in more detail, both for tau1 and tau2 transmission. The tau2 transmission appears to give a little better detection sensitivity, something like 2.15 mm versus 2.20 mm (using the MF estimates), though this miniscule difference can be just the effect of the tau1 data having much fewer data points. What would we expect in this case? The detection criterion at and around 1000 km range was

$$\frac{\max \mathrm{MF}}{\sigma_{\mathrm{noise}}} > 5$$

in all the data sets. At 1000 km range, with the values of T_{sys} and transmission power that we have been using for the shown data sets, this corresponds to effective diameter 20.7 mm (RCS $= 5.2\,\mathrm{mm}^2$) for both tau1 and tau2 data. Normally, we would expect tau2 data to be about a mm more sensitive in terms of the effective diameter at this range, but for the data sets shown in Fig. 5.6, the system temperature in tau2 was 20% higher than in tau1, and this cancels the benefit of the 20% higher duty cycle.

5.5. Velocity and acceleration

Figures 5.11–5.14 show radial velocity in all six data sets, both the directly observed Doppler-velocity v_{D}, and the velocity v_{RR} fitted from the $R = R(t)$ time series generated by the beam passage. The analyser DAN (Fig. 4.8) does not attempt to determine v_{RR} if there are too few good points $R(t_n)$ available. Both which points are considered "good", and the actual required number of them, are tunable; presently, we require at least

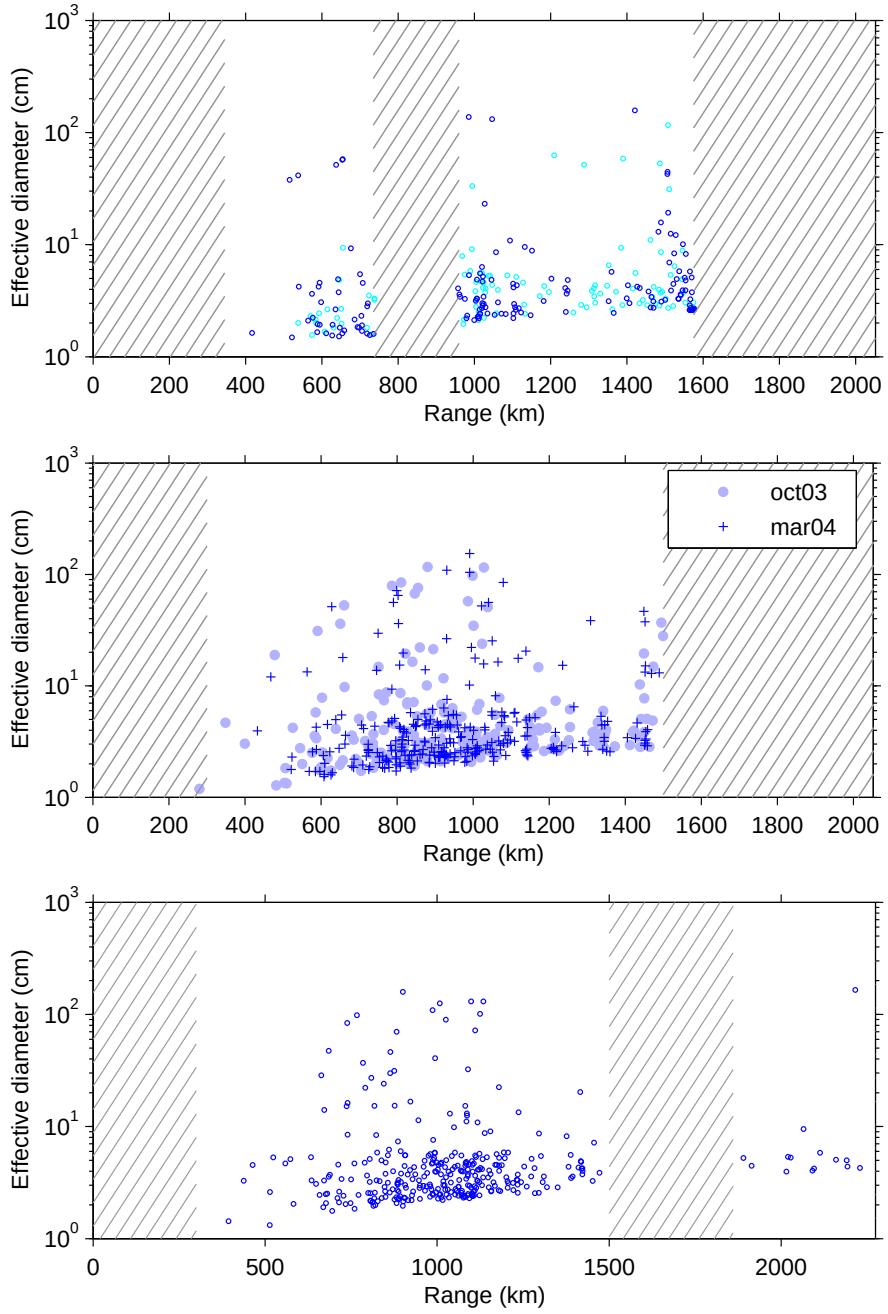

Figure 5.5.: Effective diameter vs. range. The top panel shows *mar04-tau2-500* (cyan) and *mar04-tau2-2000* (blue) data. The middle panel shows *oct03-tau1-2000* and *mar04-tau1-2000* data. The bottom panel shows *sep04-tau1-2000* data. The x-axis upper limit corresponds to altitude 2000 km in all panels.

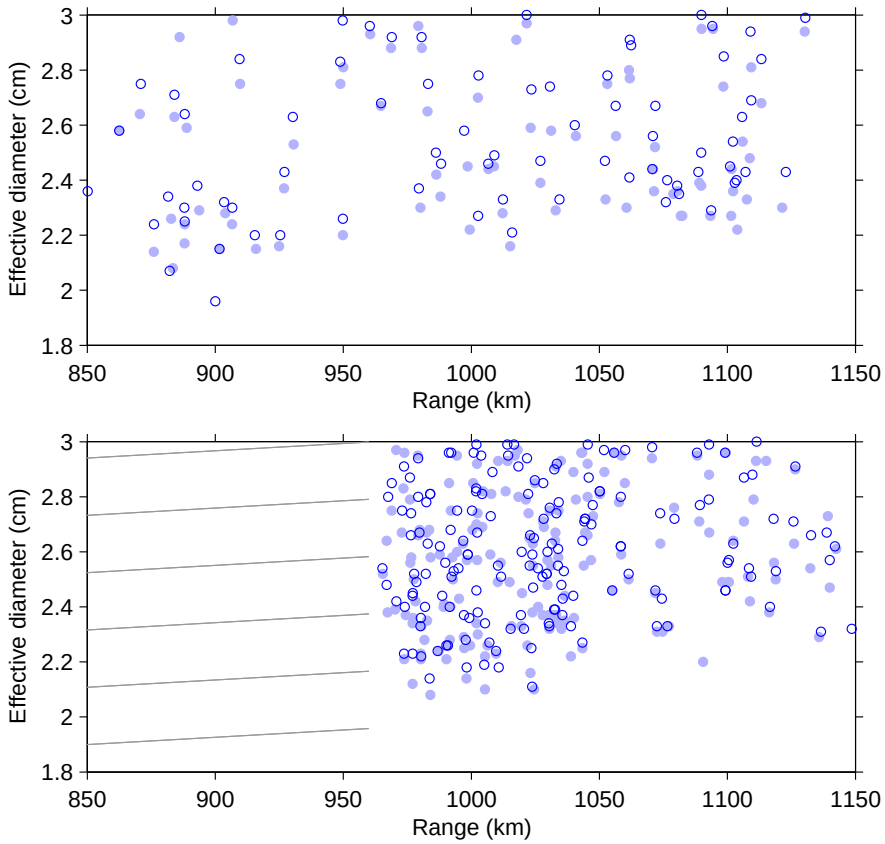

Figure 5.6.: Effective diameter vs. range, MF/FMF comparison. The top panel is part of *sep04-tau1-2000* data, bottom panel is part of *nov04-tau2-2000* data. The closed circles are computed using the FMF algorithm, the open circles are computed using MF.

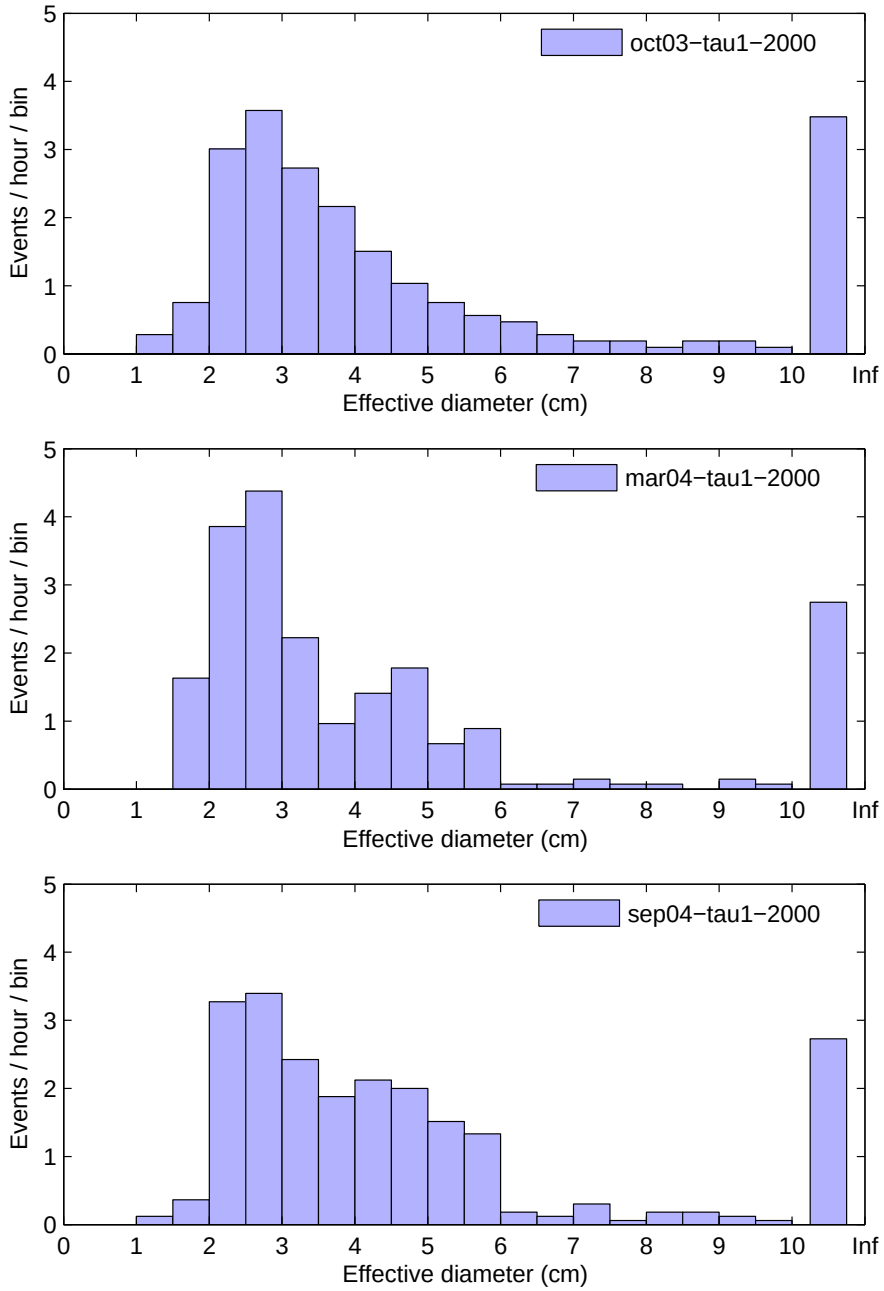

Figure 5.7.: Effective diameter in tau1 experiments.

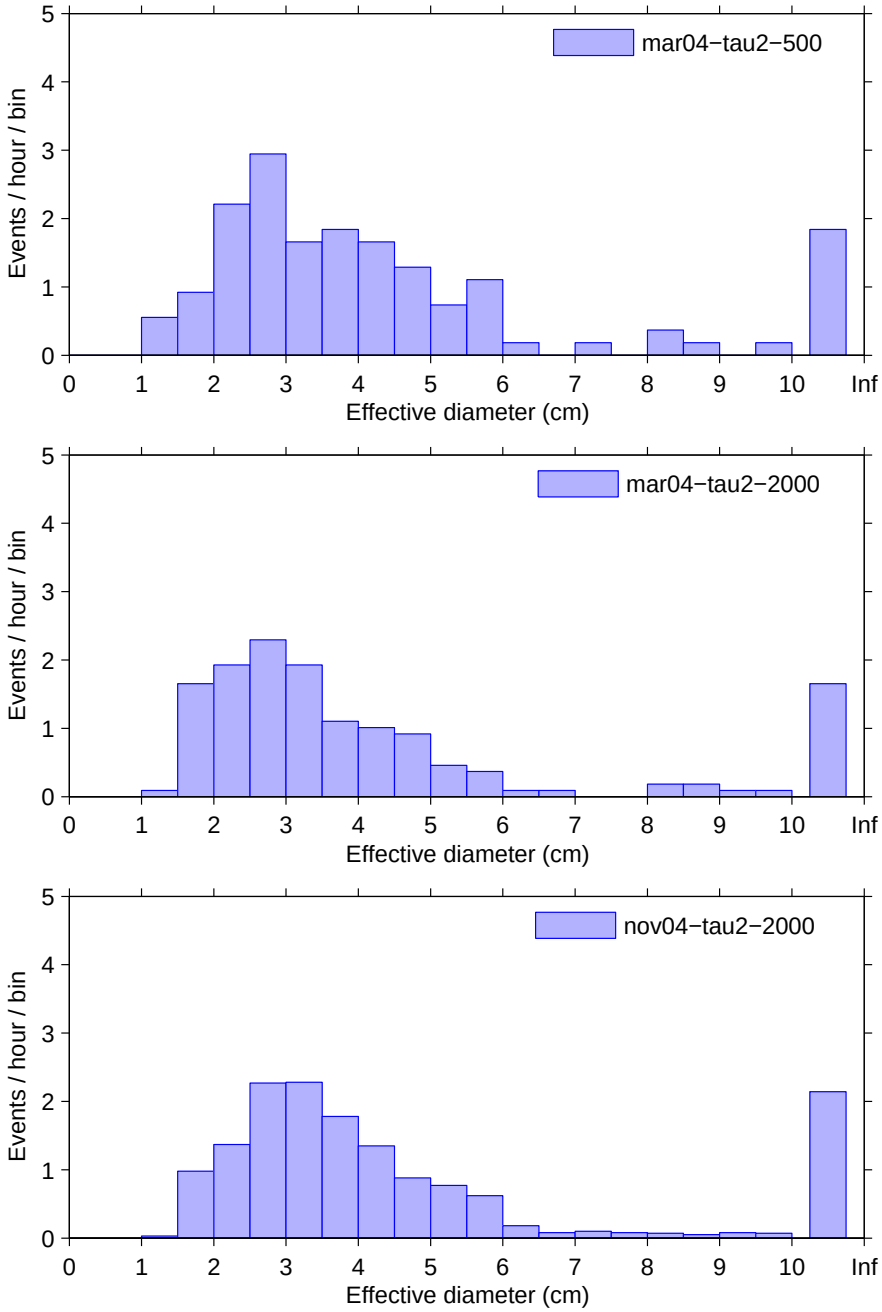

Figure 5.8.: Effective diameter in tau2 experiments.

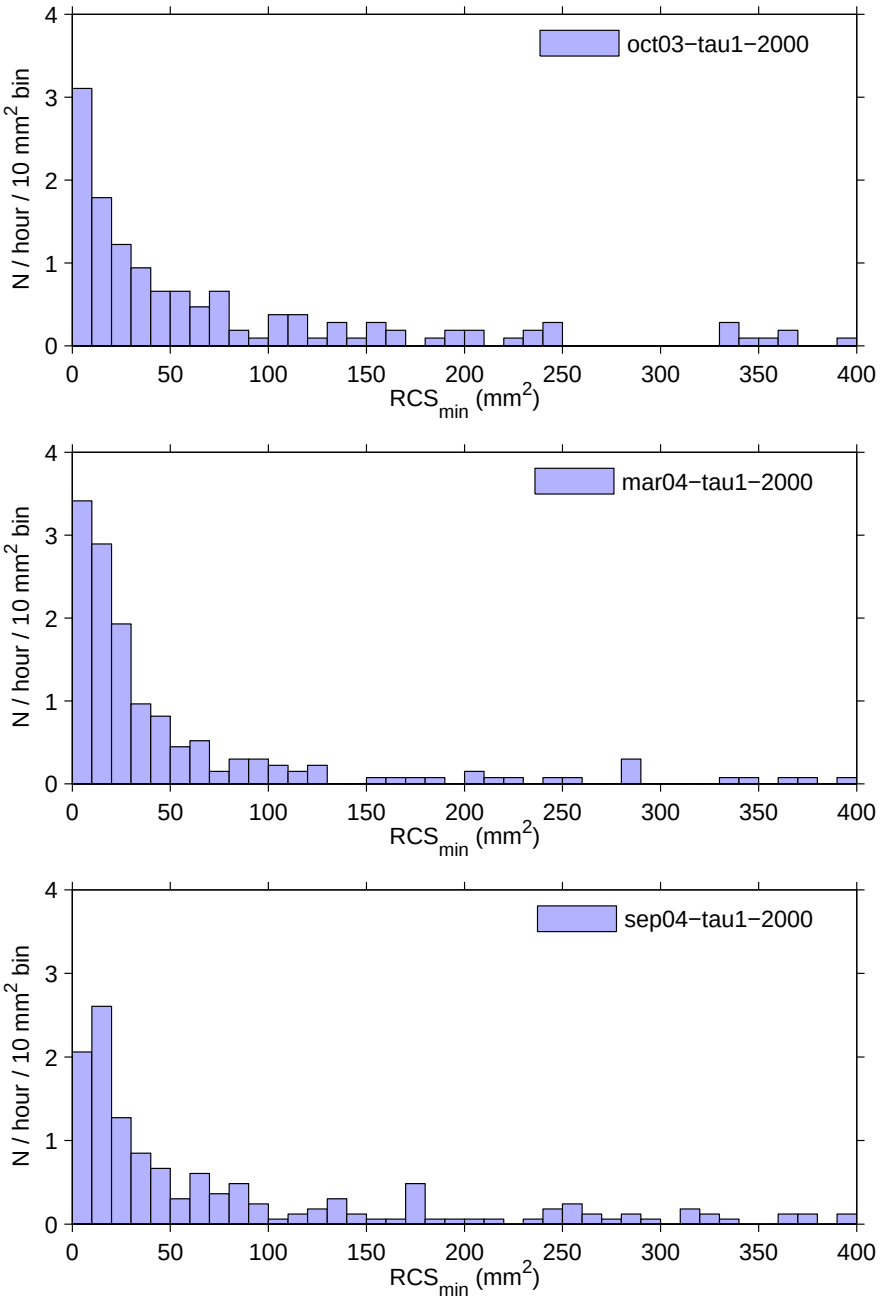

Figure 5.9.: Detection rate vs RCS in tau1 experiments.

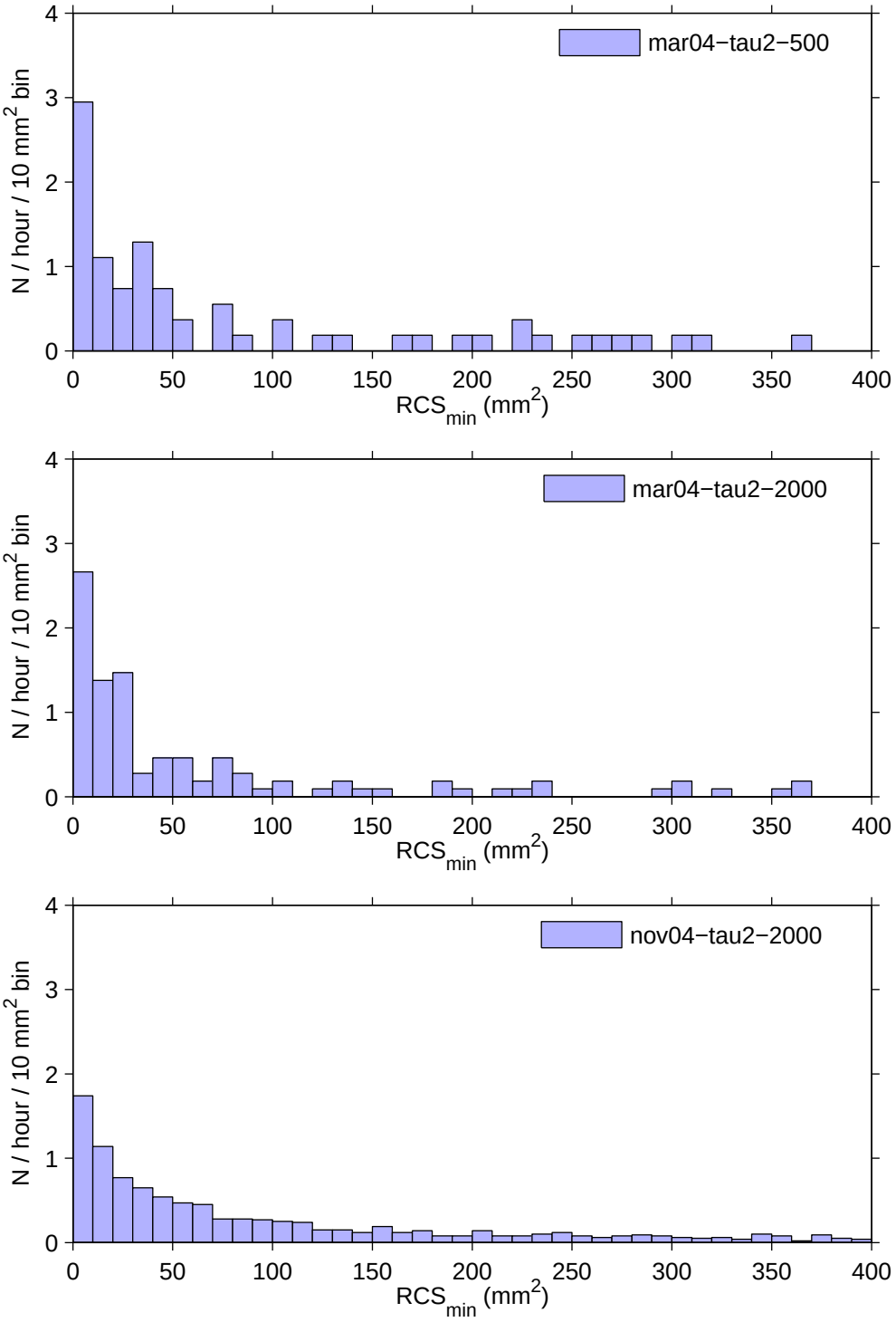

Figure 5.10.: Detection rate vs. RCS in tau2 experiments.

four good points. From the look of the event summary plots, it is obvious that many highly dubious fits are included by this crude selection method. As we have indicated in Figures 5.13 and 5.14, v_{RR} is computed in 85–90% of the events. It is not surprising that the v_{RR} data show more scatter and less structure than the v_{D} data. For instance, the splitting of the "towards" ($v_r < 0$) and "away" data into two closely separated branches, very visible in the v_{D} data in the bottom panel of Fig. 5.12, is barely observable in the corresponding v_{RR} data in the bottom panel of Fig. 5.14. As of the splitting itself, we do not know what feature in the target population it corresponds to. Another feature presumably washed out by measurement errors in v_{RR} data is the concentration of $v_r > 0$ points near the value 3 $\mathrm{km\,s^{-1}}$ around 1000 km altitude in the bottom panel of Fig. 5.11.

Figures 5.15 and 5.16 show radial acceleration, computed by linear fit to the velocity time series $v_{\mathrm{D}}(t)$ when enough data points are available. As we have indicated in the figures, this happened only in about two thirds of the events. In the figures, we have also plotted the acceleration value that was used in the detection scans. The values are represented by the upper edge of the grayed arc, and correspond to circular-orbit and strictly vertical pointing. The lower edge of the arc is computed assuming the antenna is pointed towards south, at the elevation actually used, and the target is moving in circular polar orbit across the beam. In both cases, the rotation of the Earth is ignored, but nevertheless, the shaded region should be representative of the possible values of radial acceleration of targets in circular orbits. As in the case of velocity v_{RR}, much of the scatter of the data points is more likely due to bad fits than the targets actually being in non-circular orbits.

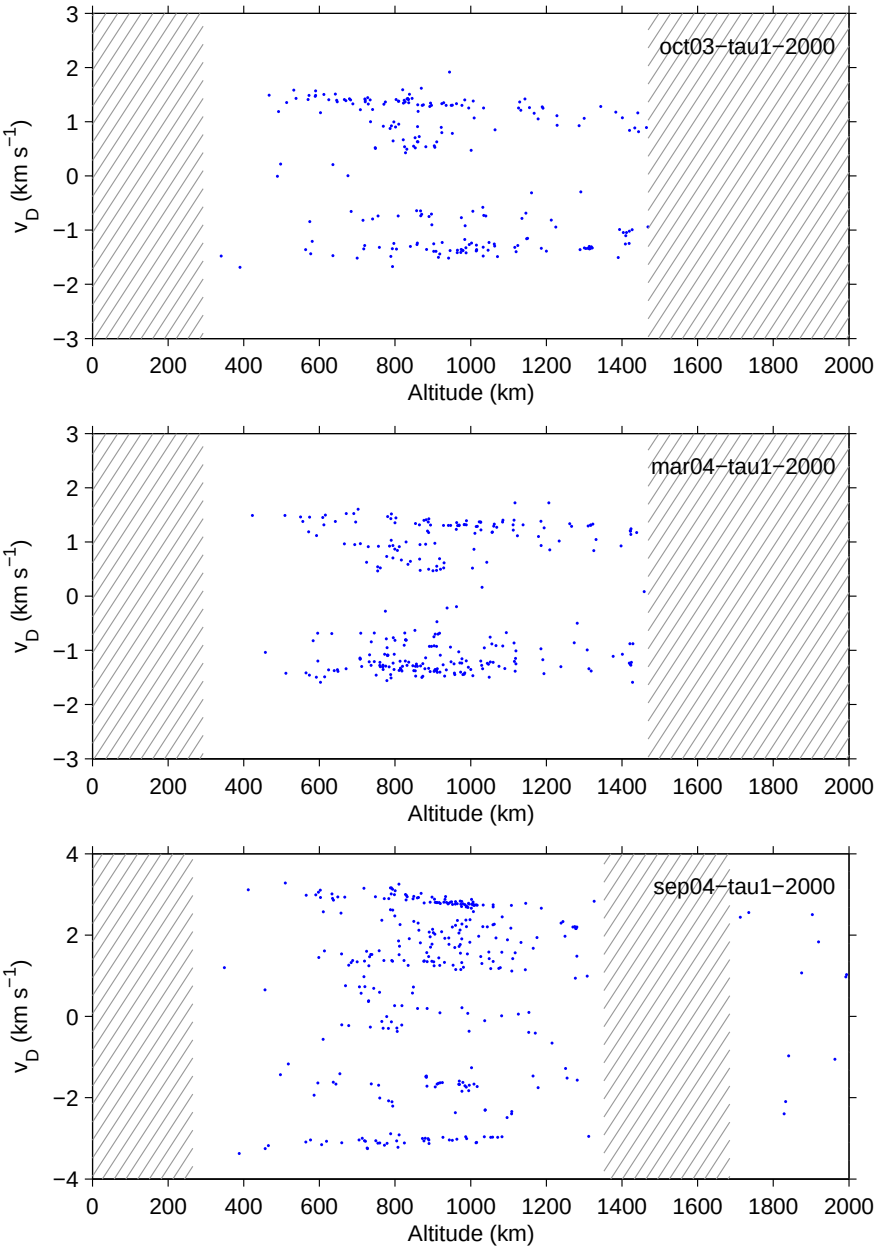

Figure 5.11.: Radial velocity v_{D} in the tau1 data sets.

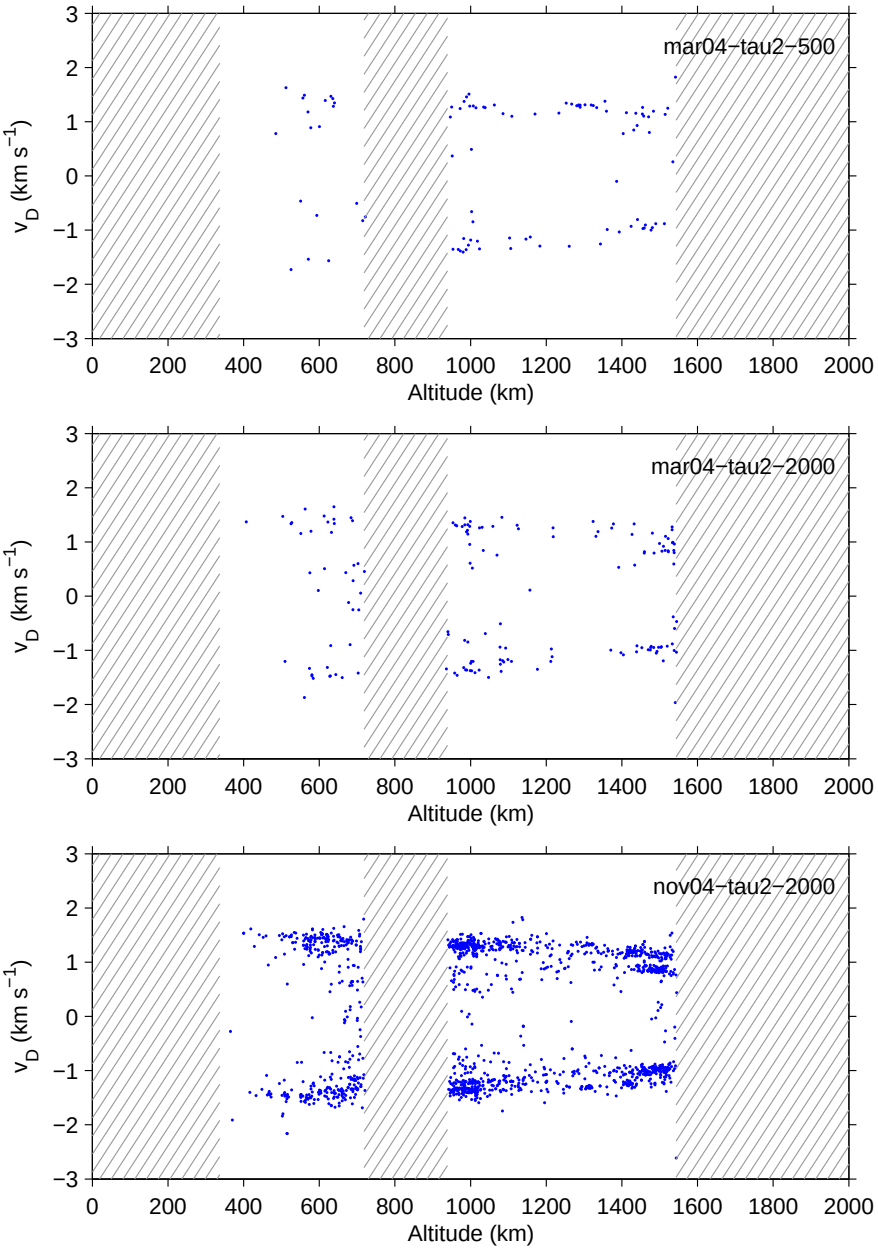

Figure 5.12.: Radial velocity v_{D} in tau2 experiments.

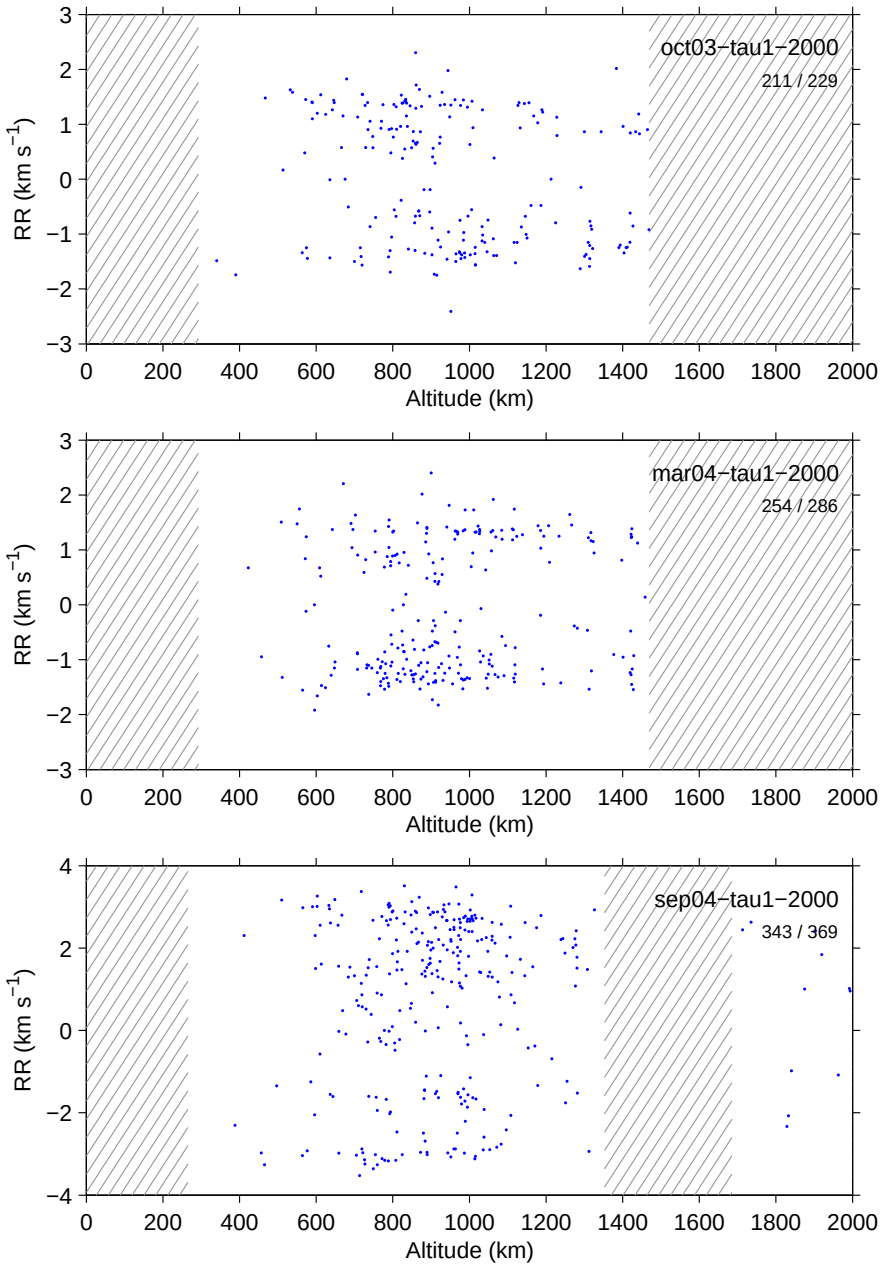

Figure 5.13.: Radial velocity v_{RR} in tau1 experiments.

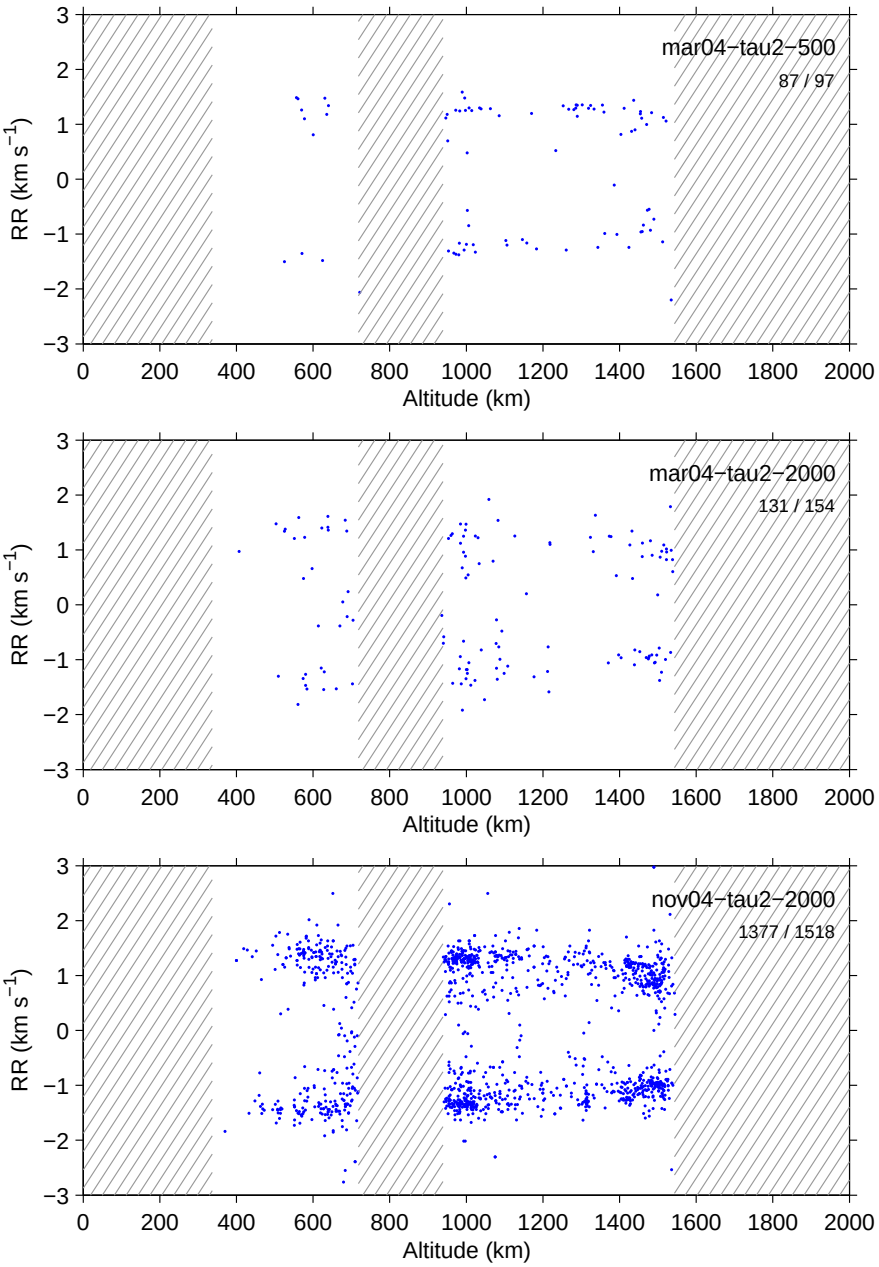

Figure 5.14.: Radial velocity v_{RR} in tau2 experiments.

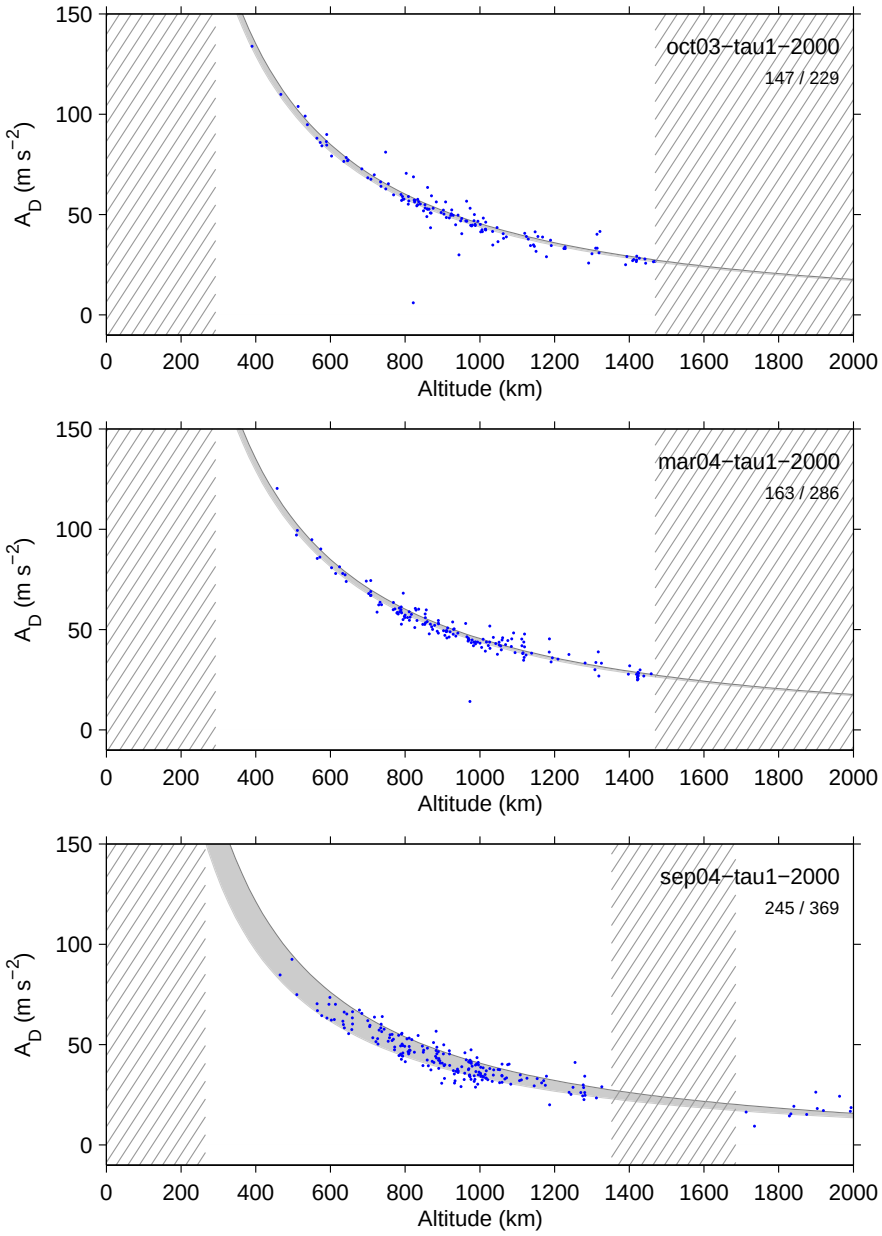

Figure 5.15.: Radial acceleration a_{D} in tau1 experiments.

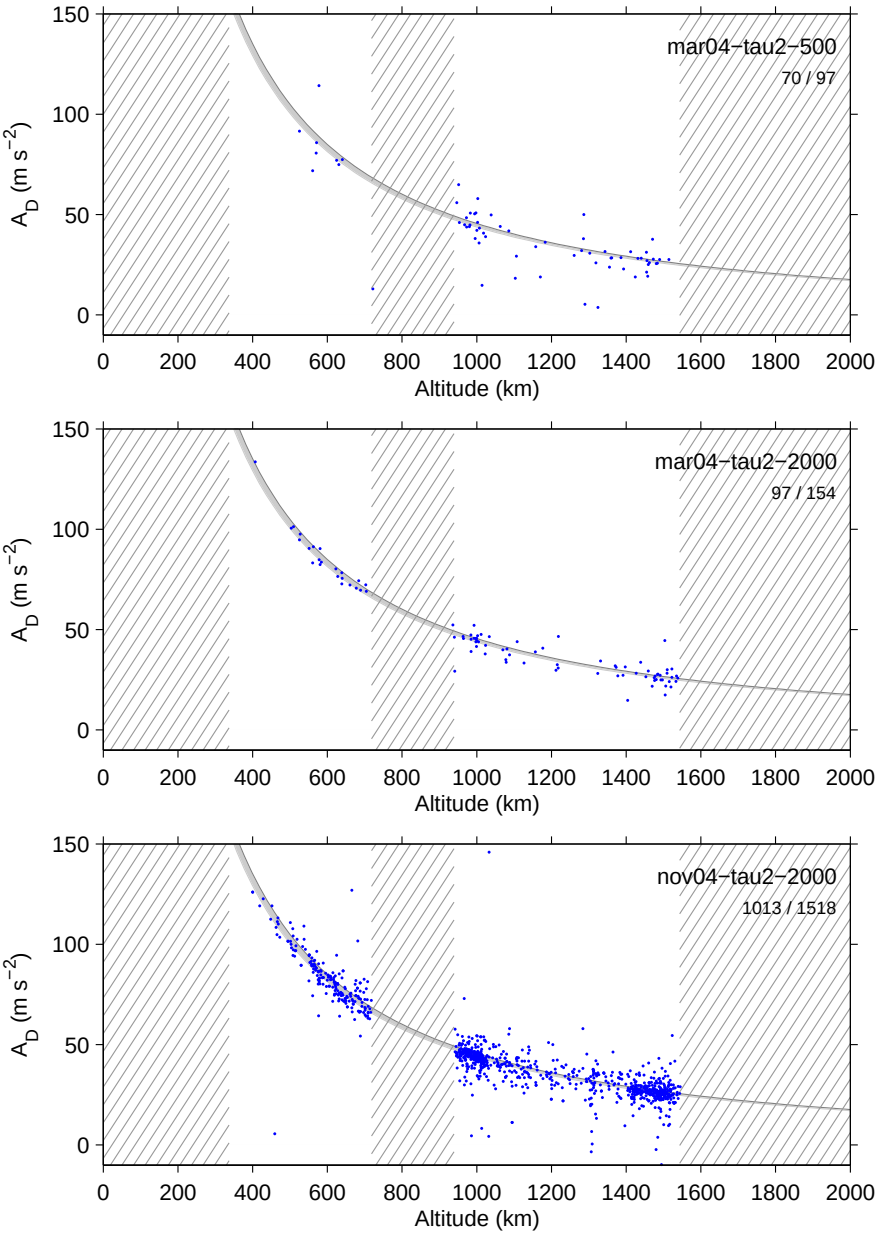

Figure 5.16.: Radial acceleration a_{D} in tau2 experiments.

6. Acknowledgements

In the late Lappish Summer 2004, our colleague and friend Juha Pirttilä died. In 2000, Juha wrote the initial MEX implementation of the MF algorithm, which showed that the present work might become possible. The development of his GUMP program that launches every time we do a measurement with the SD receiver is frozen now; there is no need to modify the program; in the shape he left it, it will be supporting this work a long time into the future. Juha was one of the kindest persons we have ever worked with; his entirely untimely demise is a loss that we are finding hard to cope with.

The EISCAT facility is supported by Finland (SA), France (CNRS), the Federal Republic of Germany (MPG), Japan (NIPR), Norway (NFR), Sweden (NFR), and the United Kingdom (PPARC).

A. Space debris receiver signal processing

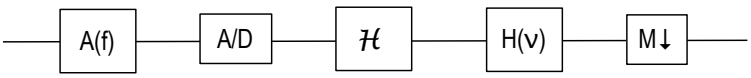

Figure A.1.: Space debris receiver signal processing logical blocks. The EISCAT 7 MHz wide IF2 bandpass filter (A) centered at 11 MHz is followed by an A/D converter sampling at 40 MHz, followed by a "Hilbert transformer" $\mathcal{H}$, followed by a sampling rate reduction system system which consist of a decimation lowpass filter H and a decimator M.

The signal handling blocks of the SD receiver are shown in Fig. A.1. We assume that the only filters we need to take into account are an analog bandpass filter A, the system's antialiazing filter, with frequency response $A(f)$, and a digital lowpass filter H, which is needed in sampling rate reduction (decimation) by a factor of M from rate $f_{10} = 10$ MHz to the final sampling rate $f_s = f_{10}/M$. The digital filter has a frequency response $H(\nu)$ and an impulse response $\{h_i\}$, where ν is the normalized frequency so that the physical frequency is achieved by multiplying ν with the sampling frequency, and the h_i are the coefficients of the decimation filter. In the SD receiver there are M coefficient h_i, all equal to unity (or equal to 1/M if we normalize the impulse response to unit area as is customary).

The frequency response of a digital filter is, by definition,

$$H(\nu) = \sum_n h_n \exp(-i2\pi n\nu), \tag{A.1}$$

so for the SD receiver we have

$$|H(\nu)| = \mathrm{diric}(2\pi\nu, M)\,, \tag{A.2}$$

where diric is the Dirichlet kernel defined in Eq. (3.71).

We assume that in front of the antialiazing filter, the system noise is gaussian, with system temperature T_{sys} and power spectral density[1] $G(f) = kT_{\mathrm{sys}}$, and that the energy of signal $s(t)$ at this point is the sought-for energy W_s. After the 6.8 MHz wide antialiazing filter, centered at 11.25 MHz, the signal is taken to the SD receiver. In the SD receiver, the signal is first sampled at 40 MHz, to produce a real-valued sample stream $\{x_n\}$. Then a Hilbert transform is performed, to get rid of the positive frequency component of the spectrum. This is done by generating complex-valued sample stream $\{y_n\}$ with 10 MHz sampling rate, with real and imaginary part defined as

$$\begin{aligned} \mathrm{Re(y_n)} &= x_{4n} - x_{4n+2} \\ \mathrm{Im}(y_n) &= x_{4n+1} - x_{4n+3}\,. \end{aligned} \tag{A.3}$$

[1] Normally, in the literature the noise power spectral density is written as $G(f) = 2kT_{\mathrm{sys}}$. We *hope*, but are by no means sure, that the EISCAT system temperature is so defined that the factor 2 does not enter.

The subtraction operation in Eq. (A.3) removes a possible DC component from the signal by high-pass filtering. That high-pass filter is so wide that its effect is ignored. After the Hilbert transformer block, the digital signal is periodic by the sampling frequency f_{10}, and consists of replicas of the negative frequency part of the original double-sided spectrum.

The next phase is filtering by the digital lowpass filter H. The baseband component of this filter has first nulls at $\nu = \pm 1/M$ in terms of the normalized frequency, or at $f = \pm f_{10}/M = \pm f_s$ in terms of non-normalized frequency, here f_s is the sampling rate after decimation. Even though for our most common case, decimation by $M = 20$, the antialiazing filter is so wide that its gain can be considered same for all the signal frequency channels, the gain of the decimation filter affects the different channels by different amounts. In our most often used experiments, we use two signal channels, one at $f_{\mathrm{ch1}} = 10.1$ MHz and the other at $f_{\mathrm{ch2}} = 9.8$ MHz. For these, the decimation filter has power gain $|H_{\mathrm{ch1}}|^2 = |H(f_{\mathrm{ch1}}/f_{10})|^2 = [\mathrm{diric}(2\pi \cdot 1.01, 20)]^2 = 0.875$ and $|H_{\mathrm{ch2}}|^2 = 0.574$.

For the apparent energy E'_{ch} of the observed signal s'_{ch} on a frequency channel, we have

$$\tau_{\mathrm{s}} \cdot \|s'_{\mathrm{ch}}\|^2 = E'_{\mathrm{ch}} \approx |A(f_{\mathrm{ch}})|^2 \cdot |H_{\mathrm{ch}}|^2 \cdot E_{\mathrm{ch}}\,, \tag{A.4}$$

where we have made use of the property that the signal spectrum on a given channel is narrow compared to the width of the decimation filter. That is, the observed energy is the original energy E_{ch} scaled by the combined power gain of the filters A and H at the frequency f_{ch}.

In filtering, the noise power spectrum transforms by the square of the filter transfer function. For any digital filter H, the Parseval's theorem is

$$\int_{-0.5}^{0.5} |H(\nu)|^2 d\nu = \sum_i |h_i|^2\,. \tag{A.5}$$

If we also can assume that the antialiazing filter gain is roughly constant over the width of the decimation filter, that is,

$$A(\nu \cdot f_{10} - f_{10}) \approx A(f_{10}) \text{ for } \nu \in [\frac{-1}{M}, \frac{1}{M}]\,,$$

we get for the observed noise power P'_{n} at the output of H, and thus also for the power of the final decimated noise,

$$P'_n \approx kT_{\mathrm{sys}} \cdot |A(f_{10})|^2 \cdot f_{10} \sum_i |h_i|^2\,. \tag{A.6}$$

By adding signal energies from all frequency channels, we get from Eq. (A.4) and (A.6)

$$\frac{\|s'\|^2}{P'_{\mathrm{n}}} \approx \frac{\sum_{\mathrm{ch}} |A(f_{\mathrm{ch}})|^2 |H_{\mathrm{ch}}|^2 E_{\mathrm{ch}}/\tau_{\mathrm{s}}}{kT_{\mathrm{sys}} \cdot |A(f_{10})|^2 \cdot f_{10} \sum_i |h_i|^2}\,. \tag{A.7}$$

For the special boxcar decimating filter used in the SD receiver,

$$\sum_i |h_i|^2 = \frac{1}{M} \tag{A.8}$$

so that

$$\tau_{\mathrm{s}} \cdot f_{10} \sum_i |h_i|^2 = 1. \tag{A.9}$$

We write the received signal energy on a channel in term of total received signal energy as

$$E_{\mathrm{ch}} = \kappa_{\mathrm{ch}} E_s \,. \tag{A.10}$$

Using this, and Eq. (A.8), we can write Eq. (A.7) as

$$\frac{\|s'\|^2}{P'_{\mathrm{n}}} \approx \frac{\sum_{\mathrm{ch}} |A(f_{\mathrm{ch}})|^2 |H_{\mathrm{ch}}|^2 \kappa_{\mathrm{ch}}}{|A(f_{10})|^2} \cdot \frac{E_{\mathrm{s}}}{kT_{\mathrm{sys}}} \,. \tag{A.11}$$

Finally, if we can approximate

$$|A(f_{\mathrm{ch}})| \approx |A(f_{10})| \tag{A.12}$$

for all signal channels, we get rid of the antialiazing filter effect alltogether, and are left with

$$\frac{\|s'\|^2}{P'_{\mathrm{n}}} \approx \left(\sum_{\mathrm{ch}} \kappa_{\mathrm{ch}} |H_{\mathrm{ch}}|^2 \right) \cdot \frac{E_{\mathrm{s}}}{kT_{\mathrm{sys}}} \,. \tag{A.13}$$

For example, if the received power divides equally to both of the two channels ch1 and ch2 in our standard experiments, (as would not appear unreasonable, but we don't really know), the correction term were

$$\sum_{\mathrm{ch}} \kappa_{\mathrm{ch}} |H_{\mathrm{ch}}|^2 \approx (0.875 + 0.574)/2 = 0.72 \,. \tag{A.14}$$


15
8
7
0
HV3B

B. DSCAN program listing

In the following, we give a somewhat stripped-down listing of the actual DSCAN code (file name dscan.c). We have removed all error-handling and cleanup code, all timing calls, all variable declarations and many of the "less essential" call parameters. Otherwise, the code, including the comments, is the actual C language code that we are currently using. Comments about the points marked by "nn]" in the listing are given in section 4.2.3 of the main text, starting on p. 61.

```
1]  // Parse the command line

    getpars ( argc, argv, sdeffile, &Autosync, );

2]  // Read scan definition from scandef file

    sid = read_scandef ( sdeffile,  );

3]  //  Initialize for MF processing

    X = create_gmfsetup ( sid,  );

    if ( Autosync )
    {
4]      // Synchronize scan start position against an EISCAT
        // integration period boundary if there are "gaps",
        // else synchronize only to next cycle boundary.

        synchronize ( sid, );
    }

5]  // Open input gstream for scanning

    tau_ns = (long) (sid->tau) * (double)1000;
    gid = gopen ( sid->file1, "r",
                      sid_time1, sid->offs, , tau_ns, );

    if ( ! Testmode )
    {
6]      // Open a hitlist file, and write current scandef to
        // it's beginning.

        hid = fopen ( hitfile, "w" );
        print_SCANDEF ( hid, sid );
    }

7]  // An infinite scanning loop. Each pass is one scan.
    // The scan goes through all the requested ranges r
    // and produces the profile, Ratio(r), which
    // at end of the loop is inspected by a threshold
    // detector.

    gates_to_do = X->nshifts;

    while ( 1 )      // SCAN LOOP //
    {
```

```
47   8]     // Check that TX rising edge is located as expected
48
49           if ( Synccheck )
50               tx_pos = checksync ( gid, -2, +2, );
51
52   9]      // Get data for next integration. "Rawlen" is the
53           // length of the integration in integer number of
54           // cycles. An addition, we need to read some
55           // "extra" points if we search targets beyond the
56           // first matching window.
57
58           gread ( gid, X->rawlen + X->nextra, raw->realp,
59                        raw->imagp, msg );
60
61           if ( *msg )
62               break;   // Typically exit due to the end
63                        // of input gstream.
64
65  10]      // Skip over "nskip" points, positioning us
66           // ready for next integration.
67
68           gseek ( gid, -(X->nextra) + X->nskip, );
69
70  11]      // Estimate TX energy and mean background noise
71           // power for this scan.
72
73           tx_and_noise ( X, raw, noiseshift, &noisepower,
74                             &txenergy );
75
76           normalizer = 1.0 / ( noisepower * txenergy );
77
78  12]      // In the scan, we compute X->nshifts range gates
79           // r. We compute X->blocksize range gates by a
80           // single call to the gmf() or fastgmf(). The next
81           // loop produces Ratio for the required ranges,
82           // each pass handling one block of ranges.
83
84           while ( gates_to_do > 0 )       // BLOCK LOOP //
85           {
86               if ( gates_to_do >= X->blocksize )
87                   ngates = X->blocksize;
88               else
89                   ngates = gates_to_do;
90
91  13]          // Call the appropriate MF routine to
92               // compute MF velocity slices, each
93               // of length X->gmflen,
94               //      v --> MF(r, v)^2,
95               // for the ngates range gates of the
96               // current block.
97
98               if ( X->usefastgmf )
99                   fastgmf ( X, raw, ngates, gate0, gmf2 );
100              else
101                  gmf ( X, raw, ngates, gate0, gmf2 );
102
103 14]          // Compute Ratio(r_j) =
104              //   max_v { sqrt [ MF^2 / (P_noise * W_tx) ] }
105
106              for ( gate = 0; gate < ngates; gate++ )
107              {
108                  g2max = floatmax ( gmf2[gate], X->gmflen,
109                                                 &k_max );
110                  *rat++ = sqrt ( g2max * normalizer );
111              }
```

```
112
113             gate0 += ngates;
114             gates_to_do -= ngates;
115
116         }// END OF BLOCK LOOP
117
118 15]     // Now we have all the range gates of the scan
119         // ready. Do threshold detection.
120
121         if ( Dedector ( ratio, X, &RatMax,, &gMax ) > 0 )
122         {
123 16]         // For hit scans we compute also a velocity
124             // estimate. To get it, we need to
125             // re-generate the velocity slice that
126             // goes though the hit's range gate.
127
128             if ( X->usefastgmf )
129                 fastgmf ( X, raw, 1, gMax, gmf2 );
130             else
131                 gmf_1 ( X, raw, 1, gMax, gmf2 );
132
133             g2max = floatmax ( gmf2[0], X->gmflen, &kMax );
134
135             Range = gate_to_range ( X, gMax );
136             Velocity = k_to_veloc ( X, kMax );
137
138             hit = 1;
139             hitcount++;
140         }
141         else
142         {
143             Range = gate_to_range ( X, gMax );
144             hit = 0;
145         }
146
147         if ( ! Testmode )
148         {
149 17]         // Optionally, save to matlab .mat files
150             //   - a piece of rawdata,
151             //   - the Ratio(R),
152             //   - the velocity slice of the MF through
153             //     the maximizing R.
154         }
155
156 18]     // Report statistics from the scan,
157         // report and save the hit if any.
158
159         save_hit ( hid, hit, ratMax, Range, Velocity,,,,);
160
161 19] } // END OF SCAN LOOP
```

C. C implementation of the MF algorithm

In this section, we give a simplified listing of the main part of the C code of our implementation of the MF algorithm, the routine GMF.

```
1  // INPUT:
2  //
3  // X:       Pointer to gmfsetup struct, containing work
4  //          arrays and control parameters.
5  // raw:     Pointer to the beginning of raw split-complex
6  //          32+32-bit float data.
7  // ngates: Number of range gates to handle.
8  // gate0:  Index of the first location to be used from
9  //          the X.shifts[] and X.acc[] arrays.
10 //          Ngates locations are used.
11 //
12 // OUTPUT:
13 //
14 // gmf2:   Array of vectors, one vector per range gate.
15 //          Each vector is a velocity
16 //          slice v-->(MF(r, v, acc(r)))^2
17
18 void gmf ( ... )
19 {
20
21      // Initialize pointers to the FFT input vectors
22      // FFTIN(r,:), for the ngates range gates. These record
23      // the current position in each vector.
24
25     for ( gate = 0; gate < ngates; gate++ )
26     {
27         fftinpos[gate].realp = ...
28         fftinpos[gate].imagp = ...
29     }
30
31     // The next loop produces the FFT input vectors
32     // FFTIN(r,:).  We handle data IPP per IPP,
33     // growing the FFTIN(r,:) vectors "in parallel".
34     // Each vector is of the form
35     // FFTIN(r,:) = TX .* RX .* exp ( -i * acc(r) * n^2 ),
36     // where TX is the IPP's transmission, shifted
37     // by shift(r).
38
39     for ( ipp = 0; ipp < nipp; ipp++ )  // IPP LOOP //
40     {
41         // Point to transmission samples of this IPP.
42
43         tx.realp = ...
44         tx.imagp = ...
45
46         for ( gate = 0; gate < ngates; gate++ )// GATE LOOP
47         {
48             // Fill in zeros in FFTIN, in front of
49             // this IPP's TX .* RX
50
51             memset ( ... );
52
53             // Point to the reception samples starting at
```

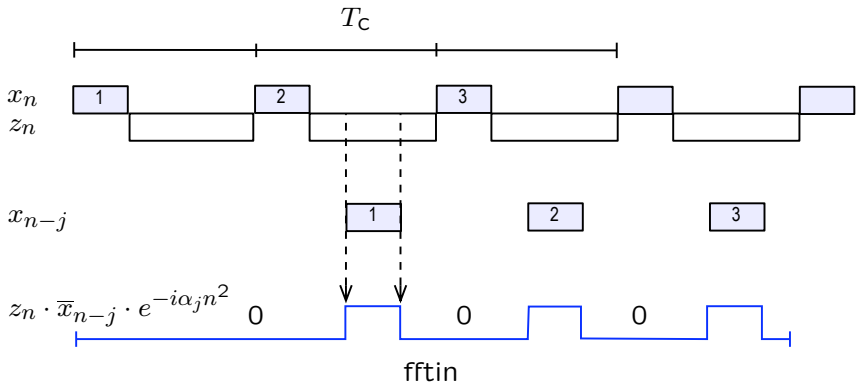

Figure C.1.: Forming the FFT input vector FFTIN in the MF computation routine GMF. The input vector is the point-wise product of the transmission samples x_n, which are shifted by the number of samples j given as a call parameter and complex conjugated, with the reception samples z_n, and with a range-dependent acceleration term. The integration in this example is over three IPPs. Both the transmission samples and the reception samples are actually part of of a single split complex (real and imaginary part stored in separate arrays rather than interleaved) data vector RAW. Note that RAW must contain somewhat more data than just the IPPs that cover the integration time T_c.

```
54              // gate, in this IPP. Note that we actually
55              // can cross the IPP boundary here, and use
56              // data from the second and even third
57              // reception window.
58
59              rx.realp = tx.realp + shift[gate];
60              rx.imagp = tx.imagp + shift[gate];
61
62              // Multiply the reception samples by the
63              // shifted TX. Matlab code:
64              //   FFTIN(gate,pos1:pos2) =
65              //       conj(TX) .* RX
66              // We use a vectorized routine
67              // from Apple's vDSP library.
68
69              zvmul ( &tx,, &rx,, &fftinpos[gate],, tlen,,);
70
71              // Perform acceleration correction, using
72              // pre-computed, range-specific acceleration
73              // value. Eq uivalent Matlab code:
74              //   FFTIN(gate,pos1:pos2) =
75              //     FFTIN(gate,pos1:pos2) .*
76              //          exp(-i*acc(gate)*(n1:n2).^2)
77
78              acc_correct_gmf ( &fftinpos[gate],
79                                tlen, accel[gate],
80                                   txon[ipp] + shift[gate]);
81
82              // If this is the last IPP, zero-padd
83              // FFTIN(gate,:) to length fftlen
84
85              if ( ipp == nipp - 1 )
```

```
86                  memset ( ... )
87
88          }// END OF GATE LOOP
89      }// END OF IPP LOOP
90
91      // We now have the FFTIN(1:ngates,1:nfftin) matrix ready
92
93      for ( gate = 0; gate < ngates; gate++ )
94      {
95          // Perform FFT
96          // Matlab code:
97          //    FFTIN(gate,:) = fft ( FFTIN(gate,:) )
98          // Use optimized routine from Apple's vDSP library.
99
100         fft_zip ( fftsetup, &fftin[gate], stride,
101                     log2n, FFT_FORWARD );
102
103         // Take gmflen points around the zero frequency, and
104         // compute squared magnitude.
105         // Matlab code:
106         //    GMF2(gate,1:gmflen) =
107         //        ( abs(FFTIN(gate,...) ) .^ 2
108
109         fft2gmf ( fftin[gate], fftlen, gmflen, gmf2[gate] );
110     }
111
112     return;
113 }
```

Auxiliary illustrations

Bibliography

[1] M. Baron, The EISCAT facility, *J. atmos. terr. Phys.* **46** (1984) 469.

[2] M. Baron, EISCAT progress 1983–1985, *J. atmos. terr. Phys.* **48** (1986) 767.

[3] G. Wannberg, I. Wolf, L.-G. Vanhainen, K. Koskenniemi, J. Röttger, M. Postila, J. Markkanen, R. Jacobsen, A. Stenberg, R. Larssen, S. Eliassen, S. Heck and A. Huuskonen, The EISCAT Svalbard radar: A case study in modern incoherent scatter radar system design, *Radio Sci.* **32** (1997) 2283.

[4] M. Lehtinen, Statistical theory of incoherent scatter radar measurements, *EISCAT Techn. Note* **86/45** (Eur. Incoherent Scatter Sci. Assoc., Kiruna, Sweden, 1986).

[5] M. Lehtinen, J. Markkanen, A. Väänänen, A. Huuskonen, B. Damtie, T. Nygren and J. Rahkola, A new incoherent scatter technique in the EISCAT Svalbard radar, *Radio Sci.* **37** (2002) 3-1.

[6] M. I. Skolnik, *Introduction to radar systems* (second edition, McGraw-Hill, Singapore, 1981).

[7] B. R. Mahafza, *Radar system analysis and design using MATLAB* (ChapmanHall/CRC, 2000).

[8] ESA Directorate of Technical and Operational Support ESOC Ground Segment Engineering Department Mission Analysis Section, *Study specification, measurements of small-size debris with backscatter of radio waves* (Darmstadt, Germany, 1999).

[9] ESA Directorate of Technical and Operational Support ESOC Ground Segment Engineering Department Mission Analysis Section, *Study specification, real-time space debris detection with EISCAT radar facilities* (Darmstadt, Germany, 2002).

[10] J. Markkanen, M. Lehtinen, A. Huuskonen and A. Väänänen, Measurements of Small-Size Debris with Backscatter of Radio Waves, (Final Report, ESOC Contract No. 13945/99/D/CD, March 2002).