SCANNING WING-BEAT-MODULATION LIDAR FOR INSECT STUDIES

by

Martin Jan Tauc

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Electrical Engineering

MONTANA STATE UNIVERSITY
Bozeman, Montana

April, 2017

TABLE OF CONTENTS

TABLE OF CONTENTS — CONTINUED

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES — CONTINUED

LIST OF FIGURES — CONTINUED

LIST OF FIGURES — CONTINUED

LIST OF FIGURES — CONTINUED

LIST OF FIGURES — CONTINUED

LIST OF FIGURES — CONTINUED

LIST OF FIGURES — CONTINUED

ABSTRACT

The spatial distributions of flying insects are not well understood since most sampling methods - Malaise traps, sticky traps, vacuum traps, light traps - are not suited to documenting movements or changing distributions of various insects on short time scales. These methods also capture and kill the insects. To noninvasively monitor the spatial distributions of flying insects, we developed and implemented a scanning lidar system that measured wing-beat-modulation. Transmitting and receiving optics were mounted to a telescope that was attached to a scanning mount. As it scanned, the lidar collected and analyzed the light scattered from insect wings of various species. Mount position and pulse time-of-flight determined spatial location and spectral analysis of the backscattered light provided clues to insect identity. During one day of a four day field campaign at Grand Teton National Park in June of 2016, 76 "very likely" insects and 662 "somewhat likely" insects were detected, with a maximum range to the insect of 87.6 m for "very likely" insects.

INTRODUCTION

Insects make up more than half of all terrestrial species on Earth, and, with the exception of plants, they dominate every level of the food chain [2]. At least 1 million species of insects exist, and some estimates suggest up to 5 million species [3, 4]. By actual population, insects have huge numbers. Ant populations, alone, have been estimated up to 10^{19}. Birds, bats, and amphibians rely on insects for sustenance, and some insects even prey upon other insects. Some pest insects are disease agents, and thus a concern for public health, as well as potential parasites for other animals [5, 6]. Furthermore, agriculture yields can be significantly reduced by the presence of insect pests (loss of $4-5 billion in the US from the diamondback moth, alone) [7–9]. On the other hand, pollinating insects are critical to a variety of plants, including those grown for agriculture purposes, like tomatoes and strawberries [10]. In fact, pollinators have been credited with creating about 10% of the world's food supply (or €153 billion) [4, 11]. Finally, insects can also be trained (or conditioned) and used as tools, such as land mine detectors [12–15]. The incredible abundance and importance of insects makes their decline a huge concern.

Over the last 500 years, animal species have been declining at an alarming rate (about 30% of vertebrate species are declining, and the same is estimated for invertebrates) [4, 16]. Insects are no exception, and the impacts of their decline have become apparent. Although a lot of attention has been focused on colony collapse disorder, other insects have also been affected. Not only are insects declining, but some birds that feed on insects have seen significant population reductions when compared to birds that feed on seeds [17]. The quick and substantial loss of species has led some researchers to designate it as part of the world's sixth mass extinction [4]. The main reason species are declining is humans. The use of pesticides, growth of urban areas, loss of habitats, reduction in crop diversity, increased pollution, introduction or migration of invasive species, human introduced pathogens, and climate change are some reasons for insect decline [4, 16, 18–22]. Although the vertebrate species extinctions have been well documented, some studies have shown that the estimates of invertebrate extinctions could be off by a factor of a thousand [16]. Clearly, insects require more studies.

Insects are critical elements of the ecosystem, but certain aspects of their behavior are not well understood [23]. Namely, local distributions of flying insects (most insect species have wings [2]) in their natural environment have not been adequately studied, in part due to a lack of tools. Proper, non-lethal, and non-invasive methods for insect detection should be developed because understanding insect movement is central to understanding insects [24]. Many variables could potentially influence how insects spatially distribute themselves. Further, different species, sexes, or ages could respond differently to these variables. Natural variations, such as the presence of trees, shrubs, or rivers, or the effects from sunlight, wind speed, or humidity could also influence insect behavior. Additionally, unintentional consequences from human factors that affect variables such as noise and light pollution from cars, machinery, or industrial sites could have substantial impacts on insect movement. The results of designing, developing, and testing an instrument that can non-lethally monitor how such variables affect local insect spatial distributions would be an important tool for the field of entomology.

As once-rural areas become more developed, flying insects within those ecosystems can be significantly affected. The number of visitors at Yellowstone National Park has been increasing for decades. Figure 1.1 shows the number of visitors since

1904. In 2015, more than 4 million people visited the park and thus noise along roadsides and light from campgrounds and villages was likely at an all-time-high. Insects within the park may be significantly impacted by these forms of pollution. Another example is the industrialization of the Bakken oil field near Theodore Roosevelt National Park. The park is home to many different animals including a large variety of birds, and with the introduction of high light and noise pollution right next to the park, this could inadvertently damage nearby ecosystems and should therefore be carefully studied.

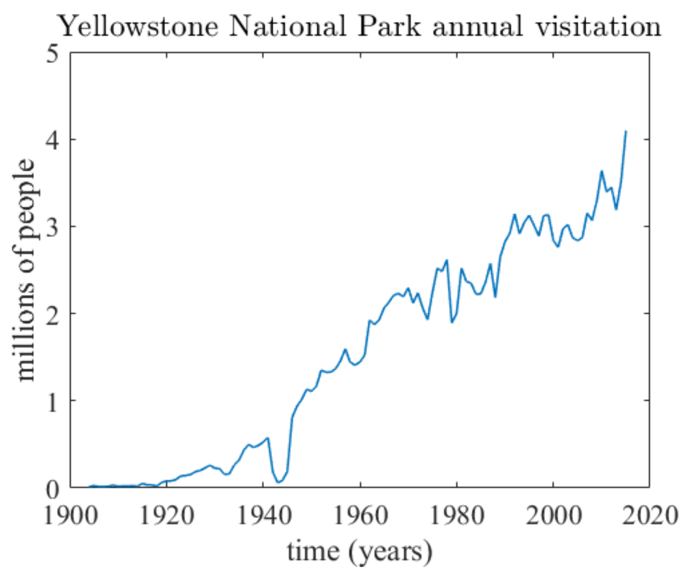


Figure 1.1: The number of annual visitors at Yellowstone National Park has been increasing (data courtesy of [1]).

Ecological Impacts

Flying insects are abundant and important factors in many ecosystems. Researchers are interested in the local spatial distributions of these insects and how these change in response to certain variables. Furthermore, responses to these variables could be different based on the species, sex, and age of the insect. In order to monitor the effects of natural or human-caused factors on flying insects, detections on the order of tens of meters need to be made. To our knowledge, no feasible tool has been able to perform these types of measurements. Many lab and field experiments have been done to learn more about insect behavior and their responses to certain variables. Although this research has improved the fields of entomology, and, as a result, the fields of ornithology and chiropterology, the available tools have some shortcomings for the problem at hand.

Lab and Field Experiments

Performing experiments on insect behavior in a controlled lab environment gives the researcher the ability to reduce the number of variables. Such an environment can have many advantages. For example, the insects are often confined to smaller spaces and monitoring their movement is simpler. The insect species, sex, and age are typically known and thus they can be separated and studied accordingly. Most importantly, perhaps, the researcher has the ability to turn on and off a certain variable and closely monitor how an insect reacts. It is much simpler to monitor properties like wing beat frequency when that insect can only fly within the field of view of a camera. Additionally, if researchers were interested in the predator-prey interactions of certain insect species, they can confine them to a region and watch how they react. The abundance of laboratory measurements has advanced the understanding of insects considerably. A particularly important example is the documentation of insect wing-beat frequencies. The laboratory experiments done to catalog a variety of species and the frequency at which their wings flap [25, 26], along with other, more recent experiments [10, 27–31], has made possible a variety of *in situ* experiments [13, 15, 32, 33], including the work done in this paper.

Laboratory experiments are the proper method to determine some aspects of insect behavior, but other insect characteristics can only be monitored in the field. For example, the monitoring of insect migrations can only be done by tracking their macro-scale movements. More subtly, insect behavior in a controlled environment may not match that of their natural environment. Perhaps mating, food foraging, and other important insect behaviors are not the same within the lab simply from the inability to account for more variables. It is likely that natural variables, such as sunlight, humidity, water sources, and other factors, influence an insect in ways that cannot be exactly mapped to laboratory environments. For example, if a researcher was interested in monitoring how insects behave when exposed to noise pollution, field experiments may show how insects take advantage of the large and open space, or perhaps they take shelter in the trees or in the grass. These questions can only be answered with field experiments. However, due to the number of variables within a field experiment, careful consideration of all factors must be accounted for when interpreting results. Also, many repetitions should be done to corroborate apparent trends.

Past Work and Gaps

Although a large amount of research has been conducted on insects and their behavior, the field has some substantial gaps. Primarily, the field lacks *in situ* measurements of localized insect distributions and behaviors. Since entomological research covers many sub-genres of work, various techniques are employed for various studies, depending on the particular need. Methods range from remote sensing techniques to collecting insect samples from physical traps. None of the past techniques have demonstrated the ability to detect local insect distributions in their natural environment without disruption.

Non-Optical Methods for Insect Detection

A variety of non-optical insect detection methods exist and include remote systems as well as local detection devices. Each system has its merits and was designed to do a particular task. Unfortunately, many of these systems were designed for a need that does not match our own. Because agriculture is a huge market, a lot of research is being done to mitigate the number of insect pests that reduce agriculture yields. Other systems are designed in order to determine the type of insect present in a region rather than what kind of behavior that insect has, and thus capturing the specimen is a viable option. Although these methods are not ideal for our intended research, they have aspects that can be useful.

Perhaps the most obvious detection method is physical trapping. Entomologists and other researchers use a number of different traps for a variety of applications. For the purposes for studying insect behavior *in situ*, traps are not ideal. One type of physical trap is the suction trap, which is used for aerial insect density [34]. Although it is used by some researchers, it has a record of preferentially capturing different insects based on sex, species, or size [35, 36]. A variety of other traps include yellow sticky traps [30, 37, 38], water traps [36, 39], and tow-net traps [35]. All of these methods have their specific and unique problems when it comes to detecting insect distributions *in situ*, but they all share one common concern: by trapping insects, the researcher is directly affecting their natural behavior. Insects are attracted to a yellow sticky trap and so they fly toward it [36], or they are caught in a tow-net trap and the use of these traps removes or disrupts the insect from its natural course.

The convenience of physical trapping gives an entomologist the ability to identify the insects' sex, age, size, and other attributes very easily and accurately. Thus, the researcher can draw some conclusions regarding what kind of insects are in a certain region. However, these traps often attract some insects while not affecting others [38], and do not provide much insight into fine-scale spatial distributions and behaviors. The researcher has to place traps in regions that can be hard to reach, and, occasionally, the location simply does not enable the presence of a trap. Although traps can be very useful for certain types of research, they are not ideal for monitoring insect distributions and behaviors in the field.

Telemetry is a remote detection method, but it requires that the insects are equipped with some hardware. Either a passive or battery-powered tag is attached to an insect and when a radio signal interacts with the tag, a return signal is produced. Such a device gives precise information about where the insect is, and, if battery powered, the tag would also have a specific identifier so the researcher could know exactly which insect was detected. However, these devices are frequently quite large and heavy when compared to insects. Especially for very small insects like mosquitoes, such a tag could not be secured to the insect without drastically impairing its natural ability. Even larger insects that could carry a tag would have to be caught and equipped with it, which could be a substantial process and could also alter the normal behavior of the insect [40]. Since our research focused on a variety of insects (not just large ones), the option of tagging them was simply impossible. Additionally, if other non-tagged insects would interact with the tagged ones, the equipment would be selectively missing some insect behavior.

Acoustic methods for flying insect detection exist as active or passive systems. Active systems emit a sound wave at a given frequency into the field and monitor the backscatter. These systems can resolve the backscatter from large individual flying insects as well as swarms of small flying insects. If the insects were close enough to the system, smaller insects could be distinguished, along with their movements. These systems have not been well developed and their use in the field is very limited [40]. Passive acoustic systems, on the other hand, simply use microphones to detect the presence of flying insects. Since the level of sound produced from insect wing flaps is typically very low compared with the background, such a system can only detect swarms of insects at a distance [40, 41]. Acoustic systems have not been developed enough to suggest that they would be able to locate single insects *in situ*.

Considering the amount of research that went into radar development over many decades, the technology is currently available and thus an attractive option for insect detection. However, due to the large wavelength of radio waves, they can only resolve, and thus detect, large insects or large groups of small insects. The technology has many uses for other insect studies, such as monitoring locust swarms [40]. By shortening the wavelength and using electromagnetic waves in the optical regime, one can utilize all of the positive aspects of radar with the ability to resolve smaller objects.

Optical Methods for Insect Detection

For the purposes of detecting insects and monitoring their behavior on a local scale, the above methods are not appropriate. Typically, such methods either affect the sample physically (with a trap or radio beacon) or they detect on much bigger scales (larger insects, or swarms of insects). Specifically, for the work of identifying insect distributions over short periods of time (hours or days) and on the order of tens of meters, a different detection method is necessary. Naturally, remote systems are required so that the researcher does not influence the natural behavior of the insect. Additionally, optical wavelengths are necessary to resolve the small insects of interests and thus such systems are considered here.

Perhaps the most basic method of detecting insects is with the naked eye and then noting what was observed. A natural extension of this is the use of video to record scenes and detect insects in post-processing, either manually or with algorithms. A video does allow a large scene to be sampled and insect identification can be done by simply watching the footage, or by using algorithms to interpret the scene. Further, insect movement and different insect interactions can be recorded and referenced. The drawback, however, is that a video system has limits in range and resolution, and only allows for a two-dimensional region to be sampled without any direct information about range. Some systems have utilized shadows to deduce insect height or have employed multiple camera systems to extract the third dimension [40]. Processing such data can become fairly cumbersome since the video and locations must be synchronized - not a trivial process. To effectively locate insects in three dimensions and distinguish them from obstructions, video recording does not readily offer the proper abilities.

Lidar has been shown to be the most promising technology for this type of entomological research. Not only does lidar provide a variety of optical wavelengths to use for detection, but off-the-shelf products can be used to construct a feasible lidar system. In some ways, lidar encompasses the usefulness of both radar and active acoustic detection without as many hindrances. Since insects will stay near vegetation during foraging [24], it is important that a system is able to monitor such activity. The photonic nature of lidar allows it to penetrate the atmosphere easily and transmit light through sparse vegetation. In fact, lidar has been used in the past to map out the ground in areas with a dense canopy [42]. This type of system would need to distinguish vegetation from insects by analyzing time-series data. Research has shown that bee wing-beats can be distinguished from other objects, like grass or other vegetation, making lidar a very viable option [12, 13, 43] for general insecct detection. Unlike radar, though, the wavelength of light is orders of magnitude smaller than the intended target of insect wings. The laser light can have enough power to detect insects while remaining safe for human eyes. For these reasons, lidar has become a popular tool for insect detection in some studies, particularly in local behavioral ones.

Perhaps the most work on insect detection with lidar has been done by Mikkel Brydegaard and his colleagues. Over the past several years, they have conducted a variety of experiments which aimed to detect and classify insects based on a few different factors, including wing-beat frequency. Their earlier work coated insects with a fluorescent dye and then released the insects into the field. A lidar pointed toward a black termination cavity (or equivalent) and any of the dyed insects that flew through the beam were detected by the lidar. They found that they were able to detect insects out to over 100 m [44]. The inclusion of the black termination cavity created a high signal-to-background ratio since the system only registered signals from scatterers in the field of view of the telescope. Much like telemetric detection methods, the fluorescent dyes were applied to some insects before being released into the field for detection. The major setback of such a system was that the researchers had to interact with the insects somehow before detecting them in the field, and thus potentially interfered with their natural behavior.

Similar studies were done without the use of fluorescent dyes. These studies relied on the wing-beat frequencies of the insects to reflect light back toward the detector [12, 13]. The use of wing-beats allowed the user to distinguish the insects from other objects instead of using dyes. Some of these systems also used a black terminating cavity to increase the dynamic range [27, 32, 45]. The use of terminating cavities required that the lidar system only be pointed in one direction, and that someone must go out into the field and place the termination cavity. Not all areas allow for the placement of a terminating cavity; for example, a researcher may want to detect insects along the edge of a cliff.

Recent studies have been performed without the use of a terminating cavity. One study used a cliff wall over 11 km away [46], and another used a tree top nearly 2 km away [47]. The cliff wall was likely similar to a termination cavity in that little sunlight was reflected back toward the detector and thus the majority of backscattered signal was from objects that intercepted the beam between the lidar system and the cliff wall. The use of a treetop was probably not as consistent as a termination cavity and thus the researchers took a background measurement every 10 minutes to allow for a background subtraction in post processing. Both experiments showed promising results for insect detections but only operated as stationary (non-scanning) systems.

The work explained above was mainly done to advance the field of entomology. The results hoped to identify insects based on the optical cross sections, their wing beat frequencies, and other parameters so that entomologists could use the tools and data for their studies. Since agriculture yields can suffer substantially from insect pests, specific research has been done to help reduce the number of insects in crops. Mullen et al. designed a lidar system that used video to track insects and lidar to obtain their wing-beat frequencies [33]. This system was designed to determine what insect was present in the camera's field of view and then use a lethal pulse of laser light to exterminate an insect if it was a pest. Such a system only worked out to 8 m and so the viability of larger (tens of meters) in-field experiments was limited. Nonetheless, the system intended to classify insects based on various parameters determined from a camera and lidar combination.

Brydegaard's group also worked to create a database of characteristics which would allow computer algorithms to determine useful information about the insect from remote detections. With the use of cameras and lidar systems, they were able to relate lidar signals with insect movement. Insect orientation and direction of flight could be determined by the camera and then compared with signals found on the quadrant detector of the lidar system. Further, wing-beat frequencies and their harmonics were detected by the lidar and were critical to determining the insects orientation. Also, spectral information about the insect due to molecules like melanin, could help provide information about the sex, age, or species of insect [27].

The fist use of lidar to detect wing-beat frequencies in the field was done by Repasky et al. in order to identify the location of trained honeybees in a land mine field [13, 14, 43]. In an effort to reduce the number of land mines across the world, RAND's Science and Technology Policy Institute for the Office of Science and Technology Policy gathered innovative methods for land mind removal. One method involved training honeybees to detect land mines in a field and then locate them with a lidar system [12, 13, 48]. The lidar system would scan a land-mine field and if a bee was in the field of view, it would register as an event. During the post processing, the researchers could identify high densities of events (bees) and conclude that land mines were present. During the scans, however, obstructions (typically grass) would trigger an event that returned the same signal as a bee, thus creating false positives. During laboratory measurements of bee-wing and bee-body reflections and depolarization ratios, Shaw et al. [12] suggested that wings may provide enough of a reflection to be detected, thus leading to the wing-beat experiments later done by Repasky et al. [13].

A system that temporally resolved wing-beat frequencies in addition to signals from hard targets allowed the user to distinguish bees from obstructions such as grass. Hard targets (DC signals) were easily distinguished from modulated returns by analyzing the results from a Fourier transform on the original time-domain data [13]. Further, by implementing a pulsed laser into the lidar system, range could be resolved. Honeybees could be distinguished by their wing-beat frequencies, the range was determined by the pulse time of flight, and the horizontal and vertical location were determined by the direction the lidar was pointing [14, 15]. The use of a mirror with scanning capabilities allowed a researcher to effectively scan a field. Insect locations could be determined with good accuracy and without the need to interfere with the insects. No termination cavity was used since the lidar system was scanning over a large region and no fluorescent dyes were needed since distinguishing the bees was done by the wing-beat frequency. The ability to remotely detect bees with a scanning lidar system without the need to physically interfere with the specimens made this method particularly attractive.

Proposed Needs and Solutions

The entomological research done over the past decades has paved a path for localized insect detection. Research in insect migrations as a result of factors like climate change, or advances to mitigate agricultural pests have used methods ranging from radar to acoustic detection. Although these methods are appropriate to these fields, they are incapable of solving the problem at hand. The need for localized insect detectors which can track insect responses to variables like noise and light is reaching a critical level. The progress in the lidar community makes this type of system a particularly appropriate option for addressing this need.

Monitoring insect behaviors on a large scale (migrations) or ultra-local scale (laboratory measurements) have been well developed, but there are a lack of tools to observe insects in the natural environment without interfering with their natural behavior. A scanning lidar system that differentiates between insects and obstructions based on wing-beat frequencies was a viable solution. Not only could such a system be constructed with off-the-shelf products, it had the potential to provide more information than any other type of system in this field of study. Observing local changes required that the system be able to detect insects out to 100 m with full range of motion in the horizontal plane (360°) and vertical plane (180°). Furthermore, the system was to be deployed in the field to observe the natural behaviors of insects. Much like the other methods discussed above, the use of a scanning lidar had some drawbacks. For one, the system took time to scan a region, often on the order of minutes, depending on the size of the scan region. In this time, changes in weather or other factors could impact the current behavior of insects. Also, the lidar system stepped through angle bins as it scanned and thus only collected data from a fraction of the entire scan region. These limitations could have been problematic for certain measurements, but for measuring relative insect numbers, it would likely have only a small effect since the randomness was equal for all scans. That is, the bias of the system was equal for insects whereas other systems preferentially detect different species, sexes, or ages. Although laboratory experiments have advanced the field of entomology, without *in situ* measurements, truly understanding insect behavior is impossible.

When Carlsten et al. [15] were detecting bees for land mine detection, they developed an algorithm to automatically locate the bees in a huge amount of data. An automated processing algorithm was necessary because most data represented unwanted signals like background and hard targets. In order to pick out rare events in large datasets, they developed a clever processing routine, which reduced the amount of time spent running computationally intensive Fourier transform algorithms [15]. Bees have a well-defined wing-beat frequency, but our instrument was designed to identify various insects and thus the post processing routines were substantially more difficult. Moreover, a set of thresholds within the data processing routine set by the user should be defined by a statistical analysis for robustness. The thresholds define when an event is classified as an insect and so the preciseness of these values would impact the final results of an experiment considerably.

Outline

From here, the paper is separated into five different chapters and four appendices. **Chapter 2: Instrument Description** explains the lidar system in detail. First, the hardware is discussed in terms of the transmitting and receiving optics. From there, the development of the software is described in three parts. The development of algorithms, which control the scanning mount, the scripts used to set parameters and run the analog to digital converter, and finally the post-processing algorithms are discussed.

Chapter 3: Wing-Beat Modulation Lidar Theory provides relevant theoretical concepts about Fourier transforms and a detailed radiometric analysis. The types of signals produced by insect wings were simulated with various sources of noise in order to approach a realistic signal. Then, Fourier transforms were performed and the frequency domain was analyzed in order to show the affects of imperfections in the time domain on the power spectrum. The radiometric analysis helped determine the maximum range bin based on a few factors, such as wing size and telescope effective aperture.

Chapter 4: Experiments and Results discusses the three main experiments along with the results of each. The experiments and results are included in one section because each experiment had a very specific and unique goal. The first set of experiments intended to show that the instrument functioned as expected and to help improve processing algorithms. These experiments were done at a beehive and detected bees for useful comparisons with previous studies on bee detection. The next experiments were performed to show that the system could graduate from bee detection to general insect detection. The insect signatures from this experiment proved to be unique when compared to bee signatures, suggesting that the system functioned properly. Finally, the main experiment scanned a region of interest and detected various insects as a function of three-dimensional space. The ecological implications of the final experiment are discussed in the Discussion chapter.

Chapter 5: Discussion explains the usefulness of the instrument and interprets the results from the main experiment for ecological impacts. The experiments showed that the instrument had the capabilities to detect insects to a certain level of accuracy. The spatial, temporal, and spectral data provided allowed us to consider how the data changed with different variables, like the time of day. With an abundance of detected insects, certain trends in insect behavior could be interpreted.

Chapter 6: Conclusion explores the modifications and improvements that can be made to the system in both software and hardware and considerations for future experiments. The software algorithms approached an autonomous process. With dedicated time and effort, these algorithms could be optimized. Also, some bugs were discovered in the code, which made data processing slightly more difficult; clearing out these bugs would be a necessity for future experiments. The hardware of the system was quite heavy and thus wiggled at each step in the scanning process producing slow oscillations in the data. Also, a larger beam would allow for the detection of slower moving insects. Future experiments could involve controlled independent variables for testing, such as noise or light pollution sources as well as ones where natural variables are more closely monitored.

Appendix A: Hardware control scripts provides the raw code used to control the scanning mount and analog-to-digital converter.

Appendix B: Post-processing algorithms provides the raw code used to process the collected data.

Appendix C: Simulated Fourier transforms is a collection of plots where the time-domain signal is varied and the corresponding frequency domain is analyzed.

Appendix D: Radiometric analysis code and plots provides the raw code used to simulate the radiometric analysis and has additional plots to aid the reader in understanding how certain variables affected signal-to-noise ratios.

INSTRUMENT DESCRIPTION

Flying insects are very small when compared with other animals, but they are also incredibly abundant, making them particularly important to ecological studies [2]. Their behavior affects other life on this planet, but that behavior is not well understood [3, 4]. Although research in entomology has led to critical advances (e.g., the discovery of insects as plant pollinators [49], advances in biomimetics [50], research in genetics [51], etc...), there is still a gap when it comes to understanding insect behavior in their natural environment. Specifically, there is a lack of tools for detecting insect behavior in the field. The first step in understanding insect behavior is developing a system which can simply detect the position of insects in their natural environment; this chapter describes the design and construction of such a system.

The design of our scanning lidar system was inspired by the work reported by Carlsten et al. [15], who created a scanning lidar for honeybee detection. The fundamentals of their hardware design consisted of a laserhead, expanding and collimating optics, a mirror mounted on goniometer and rotational stages to scan the beam, and a telescope and Photomultiplier tube (PMT) to collect and detect the light. The original laser and PMTs were integrated into the system I built, and my main improvements to the hardware included a larger telescope and placing the entire system on a scanning mount instead of using a scanning mirror. Additionally, new electronics were implemented to control the PMTs and I wrote new software to control the lidar system and to analyze the data. These improvements allowed the lidar system to scan autonomously and detect the three-dimensional position of insects out to nearly one hundred meters.

Hardware Design

The lidar system was composed of two main subsystems: the parts that controlled the outgoing beam (transmitter); and the pieces that controlled the incoming light (receiver). The main component of the receiver was a 304.8-mm-diameter Schmidt-Cassegrain telescope. Atop this telescope sat an optical breadboard with the transmitter optics. Together, the system was about 38.5 kg in mass and had a 0.5 m^2 footprint. It was attached to a pan-and-tilt mount that was mounted to a 1 m^2 optical breadboard that acted as the base of the entire scanning system (Figure 2.1). The hardware for the system changed somewhat over time, but the basic concepts remained the same. In this process, there were two iterations: the first utilized the telescope for both the transmitted and received beams by using a pair of axicon lenses and an annular mirror. The second iteration, described below, was chosen because it had fewer complications. Eventually, these complications were addressed and resolved, but the second iteration proved functional and appropriate for the experiments.

Transmitter

The transmitter essentially sat on top of the telescope. The first component in the transmitter was the laser head (JDS Uniphase NG-10320-100) with a center wavelength of 532 nm, pulse width of 0.6 ns, repetition rate of about 6 kHz, and laser pulse energy of 3 µJ. The laser head was housed in a light-tight box along with a PMT (Hamamatsu H6780-20). This PMT was used to trigger the analog-to-digital converter (ADC) and will be discussed in the Software Design section under the Analog to Digital Converter (ADC) subsection. Inside the box, two folding mirrors (ThorLabs BB1-E02) directed the beam out of the box and into a lens tube assembly.

Figure 2.1: The lidar system setup shows the transmitter and receiver sitting atop a scanning mount. *Photo courtesy of Kurt Fristrup*

As the beam expanded within the tube assembly due to its inherent divergence of 2 mrad, three elliptical mirrors (ThorLabs BBE1-E02) directed the beam toward the aperture of the telescope. The third mirror was used to fold the beam perpendicular to its current plane such that the beam was directed toward the obstruction on the telescope's Schmidt plate; Figure 2.2 (a) shows the top view of the setup up to the point where the beam folded into a new plane, and (b) shows the side view, detailing what happened afterward. After being folded, the beam traveled through two lenses, one with −25-mm focal length (ThorLabs LD2297-A) to expand the beam, and the other with a 250-mm focal length (ThorLabs LA1301-A) to collimate it. A final elliptical mirror (ThorLabs BBE2-E02), located at the telescope's obstruction, pointed the beam away from the telescope so that it traveled toward the desired location.

The laser was used in previous studies [14,15] and thus merged into our hardware redesign. The mirrors and lenses were chosen such that they would work efficiently with the laser. A few other other components, such as the light-tight box, were adapted from the previous system. This box had a threaded ⌀25.4 mm hole that coupled to ThorLabs lens tubes and thus these were used for the majority of the setup. The decision to direct the beam toward the telescope obstruction was made in order to improve the overlap between the transmitter and receiver and to make the alignment process easier. As a result, a portion of the telescope's clear aperture was blocked, but this amount was relatively small - an 5% decrease - and the benefits outweighed the costs. The beam exited the lidar system from the point of the telescope obstruction, which was ⌀101.6 mm. The size of the obstruction helped us choose a final beam diameter. We decided to expand our beam to ⌀50.8 mm because the largest dimension of the housing for the final elliptical mirror was 108 mm and thus fit within the area of the obstruction almost perfectly. Although the large beam diameter (50.8 mm) decreased the beam irradiance, there was an increase to the chances of scattering off an insect. Furthermore, the Radiometric Analysis section suggested that our signal-to-noise ratio at 100 m for a typical case was 59 and thus, the beam irradiance value was likely high enough.

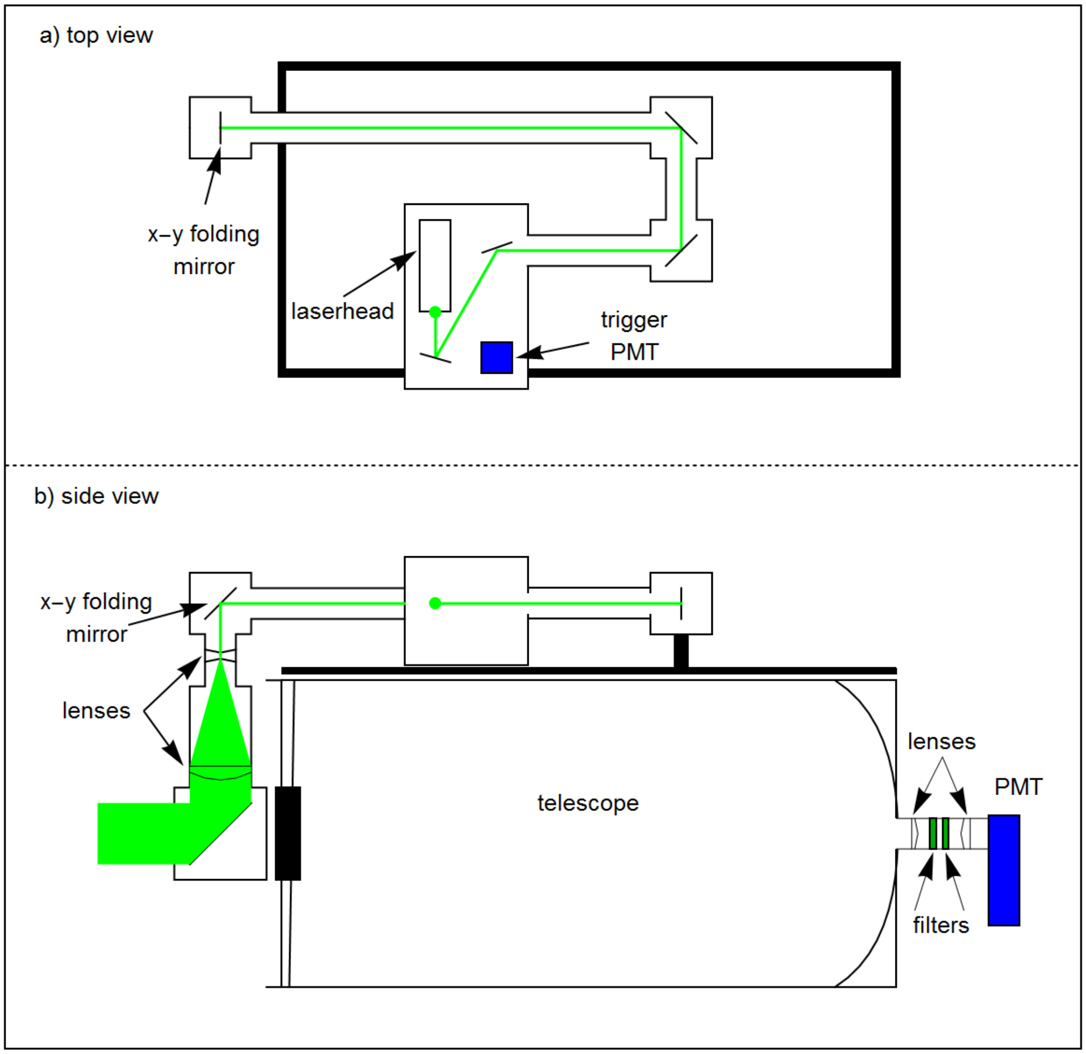


Figure 2.2: Full optical setup split between the topview (a) and side view (b) shows the beam (shown in green) as it travels through the transmitting optics.

With the transmitting system mounted atop the telescope and the telescope on a pan-and-tilt mount, the lidar had the ability to scan. Since the telescope was large and the transmitting optics were fairly small and few in number, the transmitting optics were able to fit into a single, complete system. The laserhead was mounted to an aluminum heat sink that also doubled as the stand. This was mounted inside the light-tight box along with two mirrors and the triggering PMT. A PMT was not necessary for the purpose of triggering, but it was part of previous systems and thus available. One mirror was oriented slightly off axis to reflect 99.6% of the laser light and the other was near 45° and reflected 99.4%. The triggering PMT was not pointed in any particular direction because enough scattered light within the box reached the cathode. A threaded hole on the box coupled to a lens tube assembly with the three 45° mirror mounts (all with reflectivity 99.4%). Once the beam (and lens tube assembly) was folded toward the telescope obstruction, a lens with transmission of 99.5% expanded the beam. The beam expanded to a 50.8 mm diameter and the lens tubes increased in size as well. Then, the beam was collimated with the 50.8 mm lens with 99.5% transmission. Finally, the elliptical mirror (reflectivity 99.4%) was housed in a 45° mount at the obstruction. By keeping all optics within a closed lens tube assembly, dust and debris from field experiments did not contaminate the optics and interfere with the beam. Although the tube assembly had an opening where the beam exited, this was only open to the elements during experiments. During transportation, setup, and when the system was stationary, as seen in Figure 2.1, all the optics were sealed off and and thus protected.

Receiver

The Schmidt-Cassegrain telescope (12" Meade LX200 Classic) had a full aperture of ⌀304.8 mm and a secondary obstruction diameter of 101.6 mm, which gave it an effective area of 64860 mm^2. The length of the telescope was nearly 1 m and had mounting brackets on the top and bottom, which allowed us to mount the transmitting optics on top of the telescope and then mount the telescope onto the pan-and-tilt mount. The back end of the telescope, where the eyepiece would normally go, was fitted with a custom adapter that allowed us to couple lens tubes to the telescope. Within this short lens tube assembly was a collimating lens (ThorLabs LA1951-A), two laser line filters (ThorLabs FL532-3) with a center wavelength at 532 nm and a full-width at half-maximum of 3 nm. The lens prior to the filters was necessary to collimate the beam in order to minimize angle tuning as light passed through the filters. Following these filters was a focusing lens (ThorLabs LA1131-A), with a 50 mm focal length that focused the beam onto a PMT (Hamamatsu H9305-04). The current from the PMT was converted to a digital number on the 14-bit ADC (Gage CS14200).

Much like the laser dictated the optics for the transmission components, the receiving optics worked around the PMT. Perhaps the greatest improvement from the Carlsten et al. [15] design was the implementation of a larger telescope. The previous system used a ⌀102 mm to detect wing reflections from honeybees, which have wing areas near 60 mm^2 [26]. This new lidar instrument was designed to locate various insects, some of which could have much smaller wings. For example, yellow fever mosquitoes typically have wing areas of 5 mm^2 [26]. Detecting areas that were an order of magnitude smaller than what the previous system detected required a larger telescope, thus the ⌀304.8 mm telescope was chosen.

In the previous studies, two laser-line filters were used, and thus the same approach was adapted here. This provided adequate filtering of the background signal without extinguishing too much of the desired return. The photosensitive region on the PMT was small enough that a focusing lens was required to direct the filtered light onto the cathode. Photons that terminated at the PMT marked the end of the optical portion of the design; the following sections describe the electronics and software.

Electronics

Although the hardware mostly consisted of optical components, a few electronic components were necessary for a functioning system. The most important electronic piece was the computer. The other components controlled the PMTs and various cables were used to transfer data. One of the largest and most time consuming challenges involved the transfer of electrical current from the PMT to the ADC card.

The computer was needed to house the ADC card, which was about 330 mm long. A large, rack-mount style computer was needed to house the card and secure it into a case. The computer had serial ports that it used to communicate with the pan-and-tilt mount as well as a sufficiently large hard-disk drive to retain the large quantity of data that were collected. A modified printed circuit board (PCB) was used to control the voltages on both PMTs. The PCB was adapted from a separate project, but used for its BNC cable connections and ease of power supply connection. The PMTs required a specific input voltage that was easy to supply from a wall-plug transformer. Once powered, the PMT supplied 1.2 V that could be adjusted and then sent into the gain lead. The adjustment was made using a simple voltage divider circuit with a potentiometer. The potentiometer was adjusted until the desired voltage was outputted.

The gain on the PMT was an important parameter because the PMT output current had to be large enough to drive the ADC but not so great that it would damage it. In order to increase the output current, a variety of different operation amplifiers were installed over the course of the design. These devices were used because they could increase the current but not exceed a voltage value that would damage the ADC. Although many operational amplifiers were tried, none were fast enough to maintain the original pulse-shape. The solution to increase the current was to carefully increase the gain on the PMT to a hard-to-find "sweet spot" of high current without the possibility of damaging the ADC. This was basically done by pointing the lidar at a hard target and monitoring the real-time backscatter on an oscilloscope (terminated over $50\,\Omega$), then increasing the PMT gain voltage until the backscatter produced the maximum acceptable signal of 1.5 V. This way, the PMT gain was set to the highest level so that objects with small optical cross sections would produce a readable signal, but hard targets would not exceed the ADC threshold.

Software Design

The development of software was required to control the hardware (see Appendix A) and to process and analyze the collected data (see Appendix B). Matlab was used to control the pan-and-tilt mount, the ADC card, and to perform all of the post-processing. I wrote new algorithms for each aspect of the process with the exception of controlling the ADC card, which was done with a software development kit (SDK) provided by the manufacturer. I edited portions of the SDK in order to optimize the process for particular tasks.

Pan-and-Tilt Mount

The MoogS3 Quickset QPT-130IC pan-and-tilt mount came with software that allowed the user to control it with an interface that relied on point and click operations and lacked the ability to program and integrate with other systems. Since the ADC card was controlled by Matlab through the SDK, I wrote software such that the mount would also be controlled by Matlab. With such an integrated system, the lidar system would collect data by executing a single script. Furthermore, the mount could change position while Matlab seamlessly stored data.

The mount was powered by a 24 V DC power supply and controlled by an RS-232 connection with a desktop computer running Matlab. First, the connection between the mount and computer was established and then the mount received absolute coordinates in the horizontal (pan) and vertical (tilt) directions and oriented itself accordingly. The commands were sent as 16-bit signed two's compliment little endian and split between two bytes. An escape character was implemented to avoid having a control character match a coordinate value. Finally, a longitudinal redundancy check was used as a method to make sure the bits were sent across the serial connection properly. Each of these steps was either coded in Matlab or used one of Matlab's built-in functions. Once a command was passed from the computer to the mount, the mount would begin to execute the command and, in the mean time, other non-mount related commands were executed, such as storing data. Since speed was an important factor for the lidar system, minimizing the physical movement of the mount from one coordinate to another and the time the ADC took to complete all of its tasks was an important goal.

Analog to Digital Converter (ADC)

The Gage SDK controlled the ADC card and was structured at four different levels. The lowest of these levels communicated with Windows applications using a Dynamically Linked Library (DLL), which were a set of C subroutines. The next level up was an intermediate DLL that enabled Matlab to communicate with the hardware. The second level from the top contained Matlab functions that the highest level would call. The SDK expected that the user would edit and call the highest level functions for nearly all applications. Indeed, almost all of the functions used to run the lidar system were at the top level, with the exception of a second-tier script that was modified to save the collected data as a digital number rather than a voltage in order to save time.

Much like the mount, the ADC would first initiate contact with the computer and then the computer would send a command to the card instructing it to collect data. Prior to the data collection command, the user was able to define certain parameters. Table 2.1 shows which parameters were available for modification and what they were set to during the main field campaign. Once set, each parameter was called in subsequent functions; thus, the “setup.m” script was to be the only script that required user input, and was changed depending on the user’s needs. Naturally, some other, more permanent, modifications had to be made in other functions (and are described in the next paragraphs), but the “setup.m” function controlled the data collection variables.

The SDK contained a function labeled “GageAcquire.m” and would record a single set of data. If the user were to run this function, 200 samples would have been saved in a vector. Such an experiment would not be very useful for our purposes since the only information provided would be returned signal strength as a function of range, as shown in Figure 2.3. By performing multiple “GageAcquire.m” functions, one would be able to then have both range and time stored in an array, see Figure 2.9. The “GageMultipleRecord.m” function did exactly this: typically recorded 1024 data sets, each with 200 samples. Figure 2.5 illustrates what a typical run (collection of data sets) would look like. That is, range was stored in the vertical component and time was stored in the horizontal one. Range was calculated by a pulse’s time-of-flight, and so we were converting from time to distance. Each pulse was spaced by time and so we converted from pulse number to time and this can be seen in Figure 2.4.

Table 2.1: ADC data collection parameters were easily modified in the "setup.m" script from the SDK.

structure.field	definition	value
acqInfo.SampleRate	sampling rate	200 MS/s
acqInfo.ExtClock	activate external clock	0 (no)
acqInfo.Mode	card mode (number of inputs)	single
acqInfo.SegmentCount	number of times to repeat collection (runs)	1024
acqInfo.Depth	number of data points to be collect to collect per run	200
acqInfo.SegmentSize	size of memory allocated	200
acqInfo.TriggerTimeout	trigger timeout value	0.5 s
acqInfo.TriggerHoldoff	number of samples to wait before looking for next trigger	25 KS
acqInfo.TriggerDelay	number of samples to delay	0
acqInfo.TimeStampConfig	non-zero value starts the time-stamp at start of acquisition	0
chan.Channel	chanel number	1
chan.Coupling	input coupling	DC
chan.DiffInput	non-zero sets to differential coupling	0
chan.InputRange	full-scale input range	2 V
chan.Impedance	terminating impedance	$50\,\Omega$
chan.DcOffset	input DC offset	-500 mV
chan.Filter	non-zero sets to Direct-to-ADC input coupling	0
trig.Trigger	trigger engine number	1
trig.Slop	trigger slope orientation	negative
trig.Level	level of trigger input range	-30%
trig.Source	source/port	external
trig.ExtCoupling	trigger coupling	DC
trig.ExtRange	trigger full-scale input range	2 V

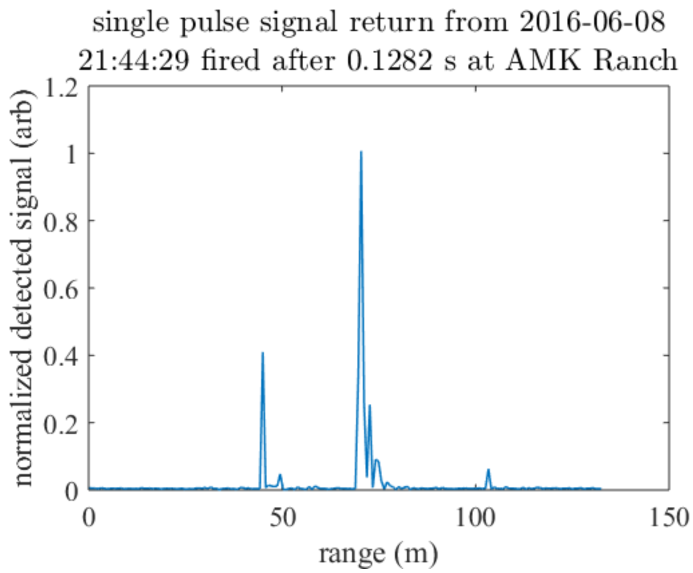


Figure 2.3: The return signal for a single laser pulse. The time of flight was converted to range on the x-axis. The pulse represented here is the 544^{th} pulse in the run. The run is shown in Figure 2.9 and the 45.01 m range bin of the run is shown in Figure 2.4.

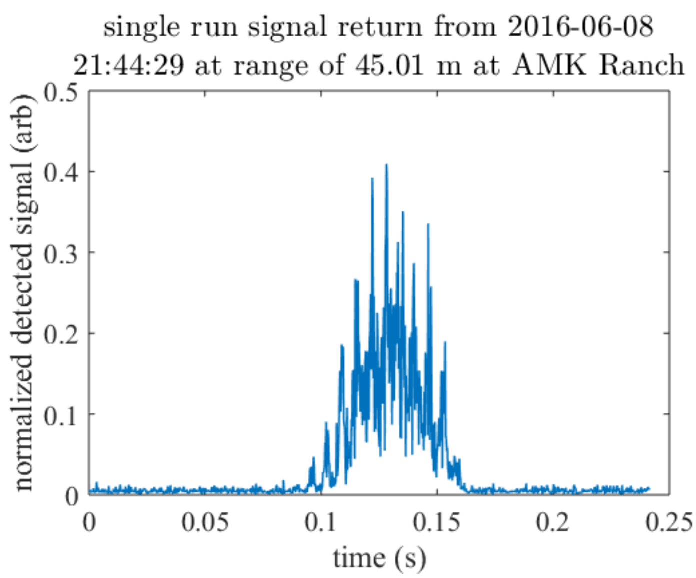


Figure 2.4: The return signal for a single run at 45.01 m. Pulse number was converted to time on the x-axis. The run is shown in Figure 2.9 and the single pulse at 0.1282 s is plotted in Figure 2.3.

During an experiment, multiple runs were performed at each coordinate. The runs could be classified in two different ways: either by chronological order where each run was pointed in a different direction, which we called a scan; or by grouping all runs that pointed in a single direction, which we called a vector. A run was collected by firing multiple laser pulses (typically 1024 pulses) in a single direction. The triggering PMT would tell the ADC to begin collecting 200 data points. The repetition rate of the laser was about 6 kHz (or pulse separation time of about 170 µs). This value did vary somewhat from run to run and Figure 2.6 shows how the repetition rate may have changed during a single scan. Most of the runs were constructed from 1024 laser pulses for a total duration of about 170 ms. Within a single run, in fact, the pulse repetition rate varied slightly; Figure 2.7 shows an example the pulse repetition instability during one run. This small (a few hundred nanoseconds) inconsistency did not have a significant impact on the experiments. The change in total duration of each run slightly impacted the Nyquist frequency, but this was already much higher than most insect wing-beats. The non-uniform pulse separation could cause artifacts in a Fourier Transform, but this was circumvented by interpolating the data over a consistent change in time. This is discussed in more detail in the next section under Post-Processing.

The repetition rate of the laser determined the upper limit to what frequency could be detected and the duration of a single run determined the minimum frequency that could be detected. For a duration of about 170 ms, a the minimum detectable frequency was about 5.7 Hz. Either the speed of data acquisition, 200 MS/s, (where MS is mega sample) or the pulse duration, 0.6 ns, determined the shortest range bin (more detail in Range Calibration subsection of the next chapter). The ADC acquisition speed limited the range bins to about 0.75 m. The ADC did not always start data acquisition with the first sample, but logged the start address instead. The

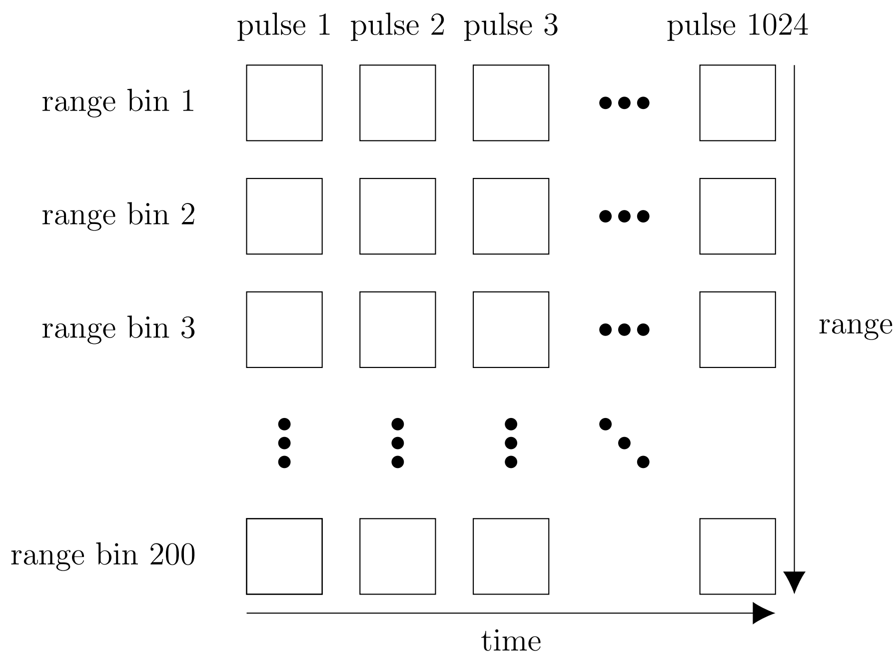


Figure 2.5: Data were collected in an array such that pulses (time) were stored in one axis and range was stored in the other.

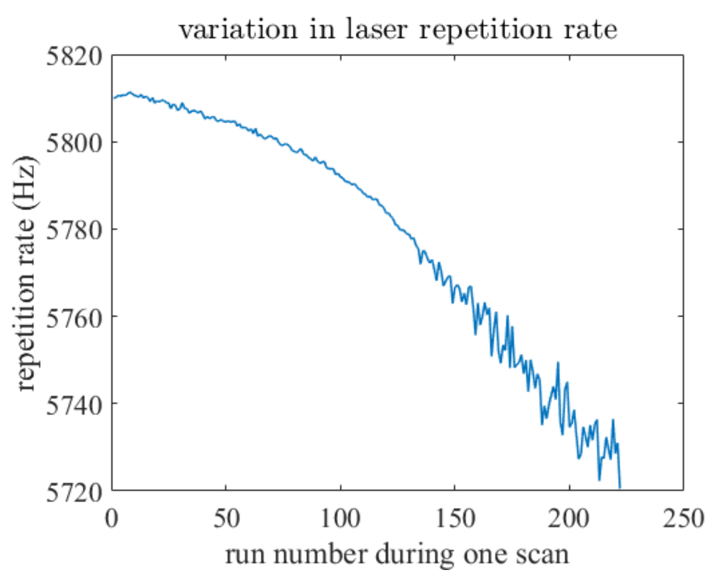


Figure 2.6: The laser repetition rate changed slightly within one scan. This was likely due to external temperature variations.

start address varied from 6 to 13, so the final twelve range bins needed to be removed, plus an additional two were deleted in case there was some variability. Also, the first eight range bins were deleted due to the amount of time it took the laser pulse to travel from the laser head to the output of the system. Thus, the usable depth, or sample size, was typically 178. Each sample represented 0.75 m and thus the furthest range bin was at about 133.5 m.

It should be noted that no signal-strength calibration was performed on the instrument. That is, the signal collected was only interpreted as signal strength and not as an optical cross section. Although other researchers have calibrated similar instruments using small polystyrene spheres with a known reflectance [52], such a calibration was not in the scope of these experiments.

Post-Processing

The ultimate goal of the project was to detect insects and the post-processing routines were designed to find unique insect signals among background, noise, and hard-target returns. The data were collected during field experiments and then processed at a later date.

Once the data were collected, they were stored as integers on the hard disk. The data were separated into days, then into scans, and finally into runs. For example, 5 June 2016 contained multiple scans, one of which was labeled *AMK_Ranch-211450*, which indicated that the scan was performed at the AMK Ranch and started at 9:14:50 pm (local time). Within this folder, the specific runs resided. These were labeled with the vector number and the pan-and-tilt location, such as *00001_P3900T0149*. This meant that the orientation of the lidar system was at a pan of 39.00° and a tilt of 1.49°. The time of each run within a scan was determined by taking the time difference between adjacent scans and diving by the number of runs. The approximate time of each run was then assigned accordingly. Each of these files contained the raw data (an array of 14−bit unsigned integers), the start address of each pulse, and the time stamp corresponding to each pulse for the entire run. Due to an error in the code, the values did not range from 1 to 2^{14}, but instead included some slight offset. The arrays were usually 200 by 1024. All these raw data were saved onto the hard drive and eventually backed up to the server "Box." In order to attempt different processing routines, the raw data were never altered, just called into other functions.

Due to the imperfect storage methods of the ADC, a simple but time-consuming routine corrected for the most basic offsets. The first of these was correcting the range bin value. Since the ADC didn't begin collecting data on the first sample, the array of data had to be adjusted so that each pulse was shifted such that all the pulses had the same start address. Since the ADC SDK stored the start address of each pulse, it was not difficult to simply remove all data prior to this start address for each pulse; however, this slightly reduced the maximum range. The other major correction was that of inconsistent pulse separation time. Since these differences were only a few hundred nanoseconds apart, an interpolation would not have a significant affect on the data. First, the average time difference between pulses was used to construct a new time vector with equal spacing. This was then used to sinc interpolate (using a script from a Mathworks file exchange [53]) between the inconsistent points.

After this process, each run underwent a normalization routine. Conditions, such as cloud cover and temperature, varied constantly during scans and it was reflected in the data. The overlap function and the r^2 correction changed from scan to scan as well. This was likely due to thermal effects or slight shifts in the optics as the system changed positions. To correct for this, an overlap and r^2 correction was determined for each scan in the same way. First, multiple runs of background-only signal were isolated and averaged. This average was then used as the overlap and r^2 correction to

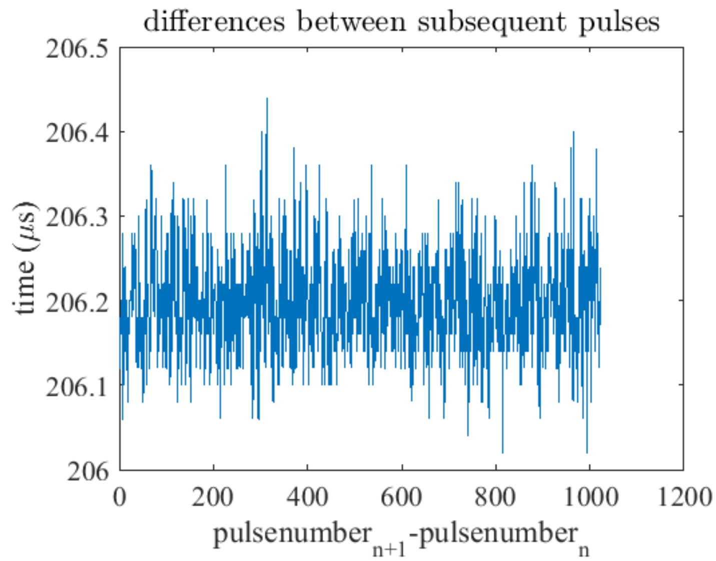


Figure 2.7: The time difference between pulses. The values differ by a few hundred nanoseconds.

each run in that scan. Considering the large variability between signals, processing normalized data allowed the same threshold values to be used on subsequent routines.

At this point, data were organized into vectors, that is, clustered by run location rather than time. For example, every time the lidar system was pointed at a tilt of 1.50° and a pan of 40.00° those data were stored in a Matlab structure. The precision of the pan-and-tilt mount was less than half of a degree, but varied nonetheless. Runs were grouped together if the pan-and-tilt angles were within ±0.245° of the intended orientation. This had little effect on the processing routines because nearly any artifacts that would come from this misalignment would be addressed in later steps of the routine.

These steps arranged the data into a usable format without really processing any information. The post processing challenge for these data was the high variation in signals. Time signals of noise or hard targets occasionally looked like the signals from insects and therefore a Fourier transform had to be applied in order to extract the frequency information. However, a (Fast Fourier Transform) FFT algorithm was time consuming and thus we only wanted to run it on a few select signals. Therefore the data were processed in a manner similar to the process outlined by Carlsten et al [15]. The algorithm described in this process was intended for honeybee detection and not general insect detection and so the algorithm thresholds could not be optimized. Instead, a manual and slightly modified approach to the process was applied.

The first step of the manual detection process was conducted in order to locate irregularities in a run. At this point, data were organized by vectors and so each run within the vector was looking in the same direction. This meant that the data from a run within that vector should look fairly similar to the other runs, thus making it possible to pick out potential anomalies. For example, each run within a vector had a hard target at the same range bin and so this signal return was high. If an insect were to enter the beam, then a strong return signal should only be in one run and the adjacent runs should have low signal at that range bin. This idea was used to locate anomalies (or "events") for further analysis. A typical run was displayed as a false-color image in Matlab as shown in Figure 2.8. In the figure, two independent hard targets were present at 69.68 m and 103.3 m. The strong return signal from the closer target implied that more light was reflected off of it than the further one. Nonetheless, each signal was consistent in time, which suggested a stationary target and not an insect. The next run within this vector is displayed in Figure 2.9. In this example, the hard target at 69.68 m is present, but the further target did not return enough signal; perhaps because the 0.01° difference in pan and tilt caused the beam variation to be more significant at the 103.3 m range bin. More importantly, however, this plot has a temporary signal at a distance of 45.01 m between 0.108 and 0.155 s. This event was flagged for further processing, where it was determined to be an insect (this is described in the next steps). Although this event was an insect, other, similar looking events did not turn out to be insects. For example, Figure 2.10, has an event between 0.028 and 0.034 s at a distance of 66.69 m. Similarly, this event was flagged for further processing and in those steps it was classified as a non-insect.

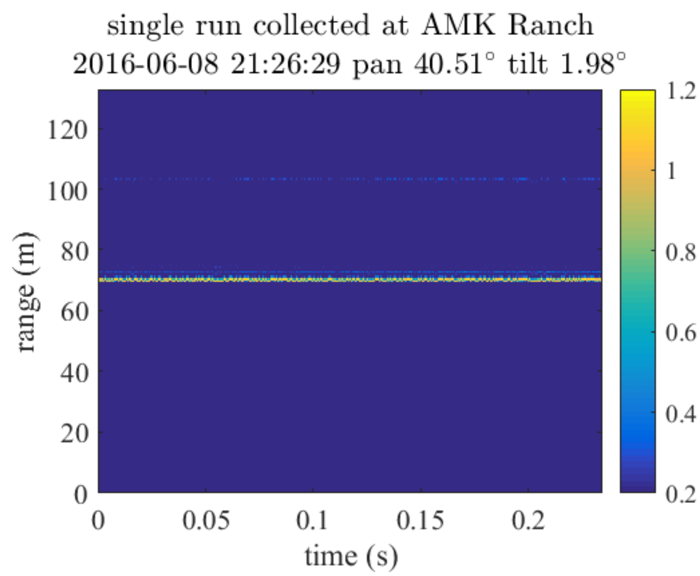


Figure 2.8: False-color image of a typical run with no event. The low signal (dark blue) suggested that no object was present while a high signal (yellow) suggested an object. At 69.68 m and 103.3 m a constant return signal suggested a hard target like a tree, blade of grass, or other obstruction.

The second step in the manual detection process involved looking at each flagged image and determining the range bin in which the event happened. Occasionally, a single image would have multiple events, and all of these were flagged separately. In Figure 2.11, two events were present in two distinct range bins. The closer event was at 26.33 m between 0.141 and 0.164 s, and the further event at 48.00 m between 0.162 and 0.192 s. This step could have been intertwined with the previous step, but due to the huge volume of runs that needed to be analyzed, performing a two-step process was quicker, and the inclusion of the second step gave the user the ability to double check instances from the first step and potentially not include them for further processing.

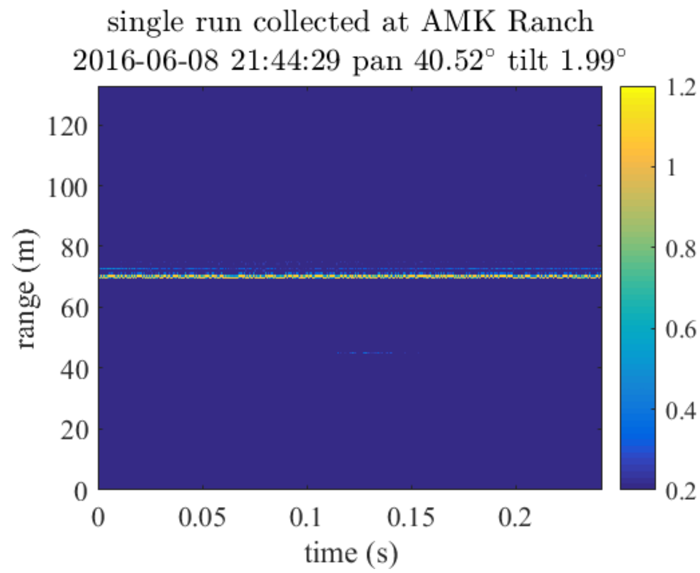


Figure 2.9: False-color image of a run with an insect event. The low signal (dark blue) suggested that no object was present while a high signal (yellow) suggested an object. A temporary signal is present between 0.108 and 0.155 s at 45.01 m.

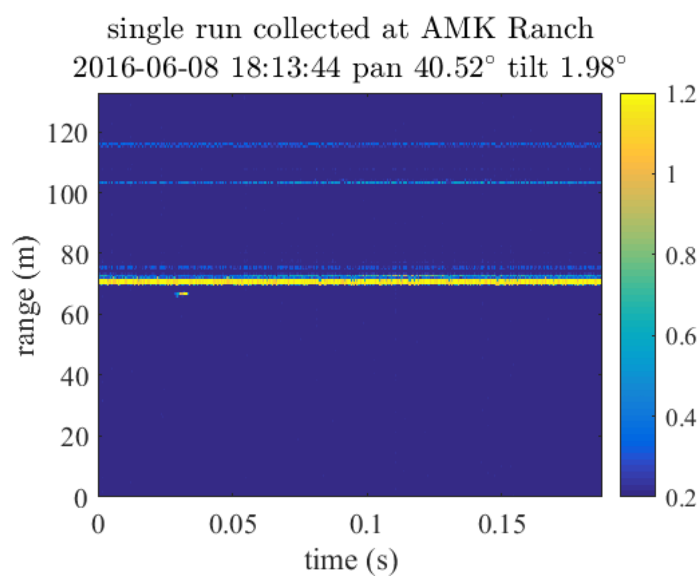


Figure 2.10: False-color image of a run with a non-insect event. The low signal (dark blue) suggested that no object was present while a high signal (yellow) suggested an object. A temporary signal is present between 0.028 and 0.034 s at 66.69 m.

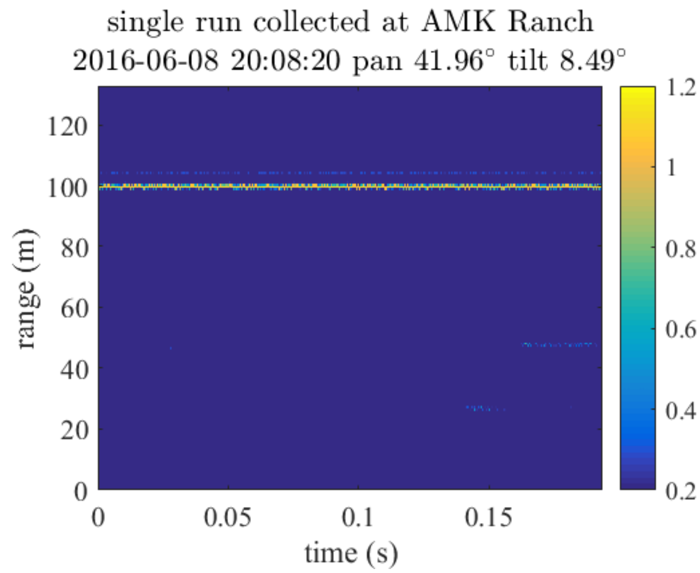


Figure 2.11: False-color image of a run with multiple events. The low signal (dark blue) suggested that no object was present while a high signal (yellow) suggested an object. A temporary signal is present in two range bins: at 26.33 m between 0.141 and 0.164 s; and at 48.00 m between 0.162 and 0.192 s.

The third step in the manual detection process plotted every flagged event in the time and frequency domains. In this step, the full range bin in which the event took place was plotted against pulse number in order to determine which part of the data to window. Figure 2.12 illustrates the windowing process. The user picked the starting and stopping location of the Gaussian window so that it included only the event. This could occasionally be tricky if the event was a low signal buried in noise, but the operator would ultimately decide what appeared to be the event. After selecting the Gaussian window, the algorithm would plot the time series along with the frequency spectrum and the Gaussian windowed time series and the corresponding frequency spectrum. This is shown in Figure 2.13. The plot has the time series along the left side and frequency spectra along the right. The top two plots correspond to the original data and the bottom two represent the windowed region. In the bottom left plot, the black circles represent the peaks that the algorithm used to determine mean peak spacing. The slider underneath the plot was used to set a threshold (represented by the black line on the top left plot) so that only the proper peaks were isolated. The magenta triangles in the bottom right plot also identified the peaks for mean peak spacing (in order to determine the fundamental frequency). The slider underneath that plot adjusted a variable such that different peaks were isolated. The red circles in the top right plot identified all peaks in the original frequency spectrum; these could not be altered and were simply there as an aid to the user.

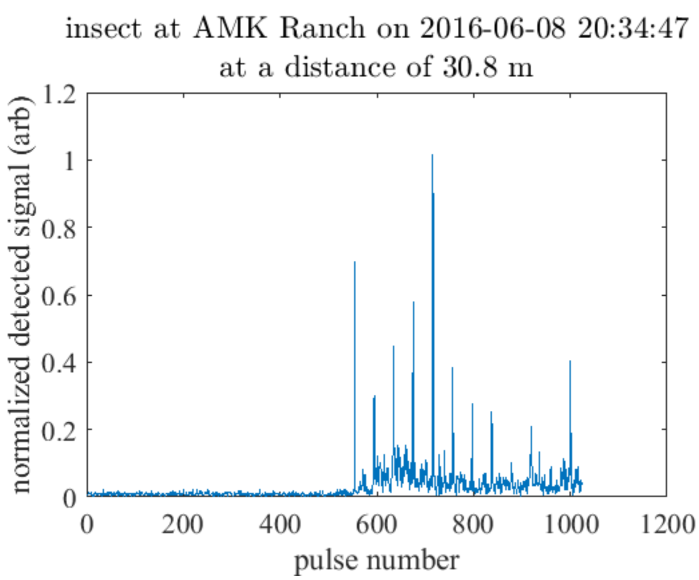


Figure 2.12: The first plot displayed in step 3 of the manual detection process. The algorithm required the user to input the bounds for the Gaussian window.

The user typically had to adjust the slider in order to pick out the proper peaks so that an accurate decision could be made. Depending on the mean peak spacing in the time and frequency domain in addition to the relative magnitude of the largest peak in frequency, a final classification was made. Figure 2.14 represents the same plots as Figure 2.13 but the sliders have been moved and the thresholds thus adjusted. In this particular example the return signal from the event was so large, adjusting the time domain threshold did not change which peaks had been

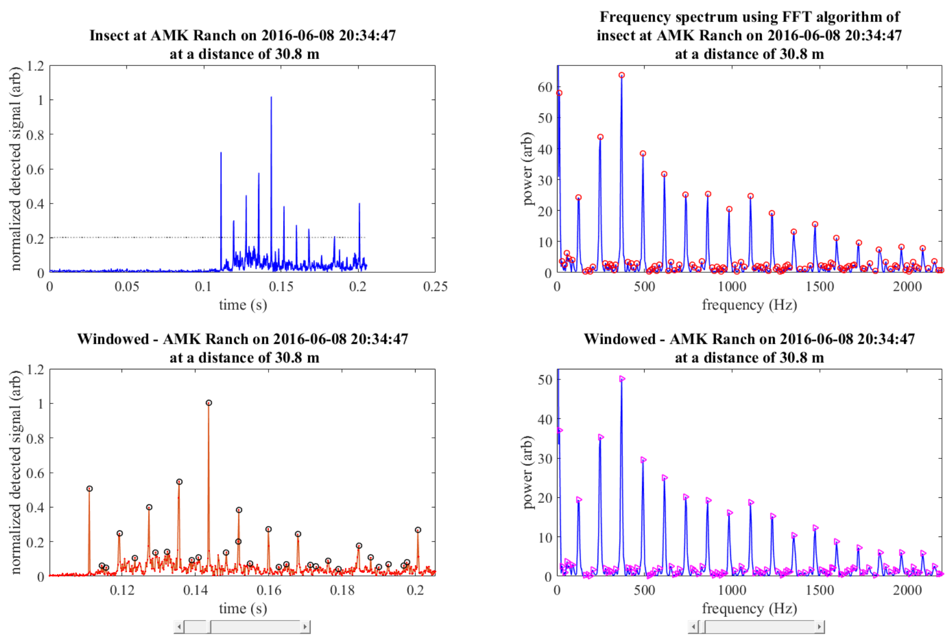


Figure 2.13: All plots displayed in step 3 of the manual detection process helped the user decide if the event was an insect or not. Top left: time series of data; top right: frequency spectrum of data; bottom left: time series of windowed data; bottom right: frequency spectrum of windowed data. The black circles in the bottom left plot show which peaks were being used to calculate the average peak spacing. The slider underneath that plot gave the user the ability to select a threshold (which was represented by the black dotted line in the top left plot). Similarly, the magenta triangles in the bottom right plot indicated the peaks that the algorithm used to calculate the mean peak spacing (in order to determine the fundamental frequency).

isolated. Additionally, the non-zero peak in the frequency domain was larger than the fundamental frequency, and so the time domain picked out incorrect peaks. In the frequency domain, however, only the large peaks had been isolated. After this step, the algorithm would display information like that presented in Table 2.2 to the user followed by the prompt "rate from not likely to likely" where the user would decide whether the event was "very unlikely," "somewhat unlikely" (although these eventually ended up as a single "unlikely" category), "somewhat likely," and "very likely" to be an insect. The strong harmonics and the equal spacing between frequency peaks implied that this was "very likely" to be an insect.

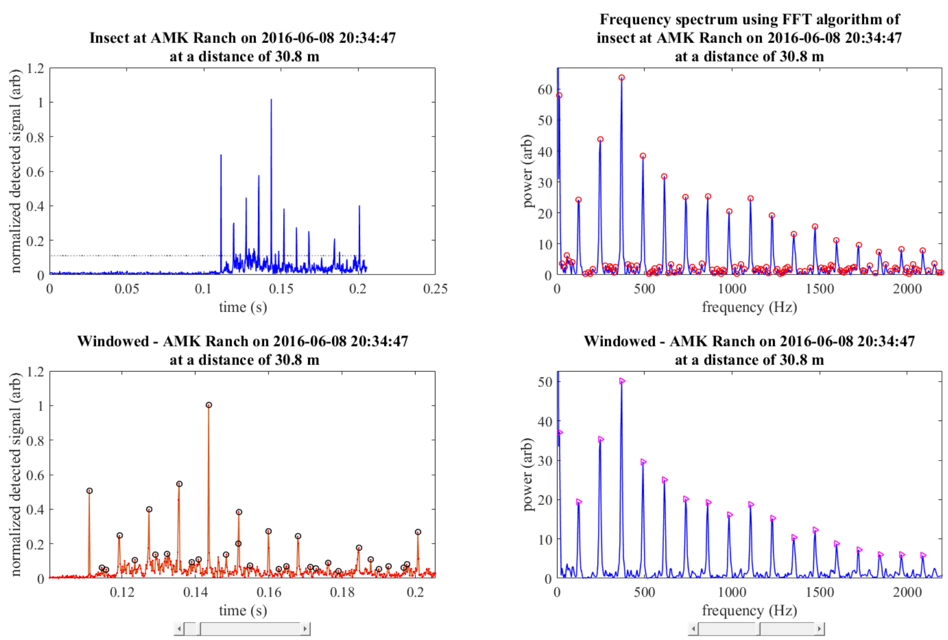


Figure 2.14: An adjustment to the slider affected the number of peaks used in step 3 of the manual detection process that helped the user decide if the event was an insect or not. top left: time series of data; top right: frequency spectrum of data; bottom left: time series of windowed data; bottom right: frequency spectrum of windowed data. The black circles in the bottom left plot show which peaks were being used to calculate the average peak spacing. In the frequency domain, different peaks were isolated for the mean peak spacing calculation, but the black circles were the same because the threshold change was not significant enough.

Table 2.2: In step 3 of the manual detection process, this was displayed to the user. Interpreting this information along with the plots (Figure 2.14) allowed the user to make a final decision on whether the event was an insect or not.

mean difference between peaks (no zero)	121.10 Hz
std of difference between peaks (no zero)	5.68 Hz
mean difference between peaks	110.45 Hz
std of difference between peaks	34.10 Hz
maximum frequency peak	369.80 Hz
mean frequency between isolated time peaks (black circles)	334.82 Hz
mean frequency between all time peaks	329.114 Hz

The “very likely” insect cases were inspected one final time to make sure no accidental classifications were made and no duplicates were classified. Occasionally, a large signal triggered on subsequent range bins and thus appeared as though an insect was present in both. By comparing subsequent range bins, duplicates were easily removed from the database. The “somewhat likely” insect cases were also analyzed, although not as throughly due to the large quantity of events. These cases were not inspected a final time, but duplicates were removed and fundamental frequencies were estimated (and should therefore be analyzed cautiously). The final“very likely” and “somewhat likely” events are presented in the Discussion chapter and criterion for insect classifications are discussed.

Chapter Summary

The instrument was described in two parts discussing the hardware and electronics, and software. The hardware was a complete redesign of previous systems, although it did take inspiration and some components from those previous works. The new design had the ability to scan a large region while controlled from a single computer. The post-processing routines were performed manually, but built a training set so that a future algorithm could be written.

The physical lidar system was optimized for outdoor data collection. The transmitting optics were all sealed off except at the point where the beam exited toward the field, and this was only open during experiments. Similarly, the receiving optics were also sealed except for the telescope's front aperture, which, again, was only open during experiments. The robust pan-and-tilt mount allowed the lidar to be pointed in any direction without modifying the beam shape or overlap region. Although the system was somewhat heavy, and thus a little slow, it used eye-safe laser light and was modular so that it could be taken apart and transported.

The software that controlled the mount and data acquisition systems was critical to the control of the lidar. The integration of mount control and data acquisition allowed us to determine the spatial and temporal location of insect events. The post-processing algorithms required that the user go through a thorough analysis of all events and classify them. This task was time-consuming, but it provided data for analysis and constructed a training set so that an autonomous algorithm could be written to do the classifications independently.

WING-BEAT MODULATION LIDAR THEORY

Insect studies are important to understanding small and large ecosystems. Insects are a critical link in the food chain, they are a necessity for pollination of crops, and they transmit diseases and reduce agriculture yields. Researchers have been studying insects for decades, and in that time, they have developed novel and useful tools for various studies. Detecting insects in their natural environment without physically disrupting them has been a challenge for the entomological community. In order to spatially locate insects on the order of tens-of-meters, a scanning-lidar system has been implemented.

The scanning-lidar system detected the scattered light from oscillating insect wings. In order to better guide the hardware and post-processing algorithms, some simulations were performed. First, the Fourier transform is defined and a few important examples are explored. Since the expected time-domain signals are sharp spikes, we have modeled insect-wing signals as Dirac delta functions. To make these signals more realistic, different types of noise were added and then the entire signal was Fourier transformed and the power spectrum was analyzed. At the end of this chapter is a detailed radiometric analysis that helped guide the design of the system.

Fourier Transform

In terms of signal processing, The Discrete Fourier transform (DFT) was the most important tool for this work. The Fourier transform extracts frequency information from the time-domain so that a spectrum analysis can be performed. This reveals which frequencies and harmonics are dominant and thus sheds light on the original signal. The DFT is critical for insect detection because the insects have wing-beat frequencies that are much higher than most similar-sized objects in nature (e.g., leaves or grass moving in the wind), and these high frequencies are easily determined after performing a DFT. Since the DFT requires a significant amount of computing power, the brute-force process is slow, so instead, a Fast Fourier Transform (FFT) algorithm is commonly used in digital signal processing applications.

The Continuous Fourier transform is defined by Equation 3.1 and is useful for examining a few specific Fourier transforms that are important to insect wing-beat signals (the definition of the DFT is presented in Equation 3.8). The mechanics of flapping wings produce sharp peaks in the time domain rather than sinusoidal shapes. The ideal approximation of time-domain wing-beat signals is a series of Dirac delta functions, called a Dirac comb function and defined by Equation 3.2. The Fourier Transform of a Dirac comb is simply another Dirac comb (with different period and amplitude), as shown in Equation 3.3. For example, if a Dirac comb had a period of T, then the transformed Dirac comb would have an amplitude scaling factor of T and period of $\frac{1}{T}$, so if the spacing of the comb in the time domain widened, its spectrum would become more narrow in the frequency domain. The ideal time-domain signal would be a Dirac comb function only if the signal was infinite in time, but since our time window was only about a quarter of a second and the event usually only took a portion of that time, we assume that our time signal was windowed by a rectangular function, see Equation 3.4. The Fourier transform of a rectangular function is a *sinc* function (defined by Equation 3.6) and is shown in Equation 3.5. Much like in the case of the Dirac comb, as the rectangular function widened, the *sinc* function narrowed and scaled in amplitude.

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-2\pi j f t} dt \tag{3.1}$$

$$comb\left(\frac{t}{T}\right) = T\sum_{k=-\infty}^{\infty}\delta(\frac{t}{T}-k) \tag{3.2}$$

$$\mathscr{F}\left\{comb\left(\frac{t}{T}\right)\right\} = T\left(\frac{1}{T}\sum_{k=-\infty}^{\infty}\delta(Tf-k)\right) = Tcomb(Tf) \tag{3.3}$$

$$rect\left(\frac{t}{U}\right) = \begin{cases} 0 & \text{if } |\frac{t}{U}| > \frac{1}{2} \\ 1 & \text{if } |\frac{t}{U}| \leq \frac{1}{2} \end{cases} \tag{3.4}$$

$$\mathscr{F}\left\{rect\left(\frac{t}{U}\right)\right\} = Usinc(Uf) \tag{3.5}$$

$$sinc(Uf) = \begin{cases} \frac{sin(\pi Uf)}{\pi Uf} & \text{if } Uf \neq 0 \\ 1 & \text{if } Uf = 0 \end{cases} \tag{3.6}$$

All figures in this section are presented in pairs. The first figure of the pair is always the time-domain signal and the figure which immediately follows it is the corresponding frequency-domain information. Additionally, the time and frequency values of the simulations did not intend to match the collected data. Typically, the time-scale of collected data was a few milliseconds and the frequency-scale was hundreds of Hertz. For convenience, however, the simulated data the time scale was seconds, and the frequency-scale was less than one Hertz.

Considering that most of our insect signals in the time domain approximate a multiplication of a Dirac comb function and a rectangle function, we simulated ideal time-domain signals, as shown in Figure 3.1. The multiplication of two functions in the time domain represented a convolution in the frequency domain and this is shown in Equation 3.7. The Fourier transform of this signal is shown in Figure 3.2. The peaks in the frequency domain were wider than the Dirac delta functions within the time series *comb*, and that is because they were being convolved with a *sinc* function, and thus widened. These two plots begin to show us what types of signals we may expect to see in the time and frequency domain, if our insect was ideal. Further investigation shows that with a spacing of $T = 20$ s and time window of $U = 200$ s in the time-domain, the frequency-domain had spacing of $\frac{1}{T} = 0.05$ Hz and the width of the Dirac delta functions increased by $\frac{1}{U} = 0.005$ Hz. Furthermore, the power (i.e., signal^2) was $(U \times T \times \frac{1}{T})^2 = 200^2 = 4 \times 10^4$.

$$\mathscr{F}\left\{comb\left(\frac{t}{T}\right) \times rect\left(\frac{t}{U}\right)\right\} = Tcomb(Tf) * Usinc(Uf) \tag{3.7}$$

$$X_{f+1} = \sum_{t=0}^{N-1} x_{t+1} e^{\frac{-2\pi jft}{N}} \tag{3.8}$$

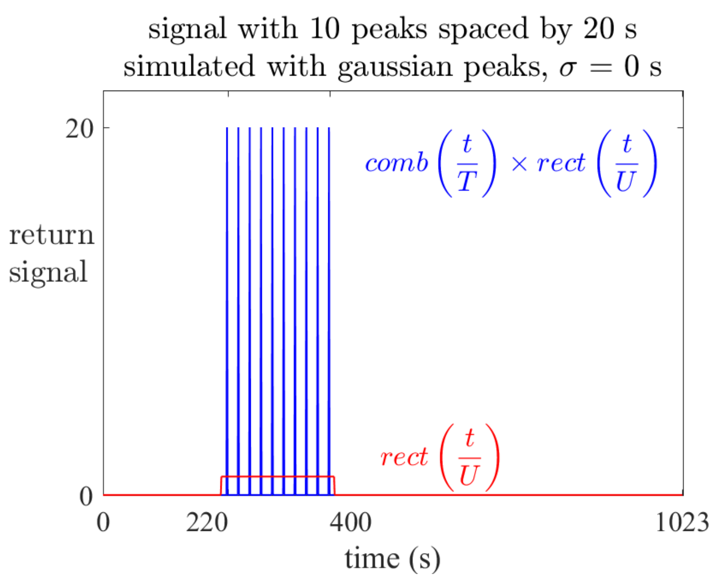


Figure 3.1: The time domain for an ideal insect was a perfect *comb* windowed by a *rect*.

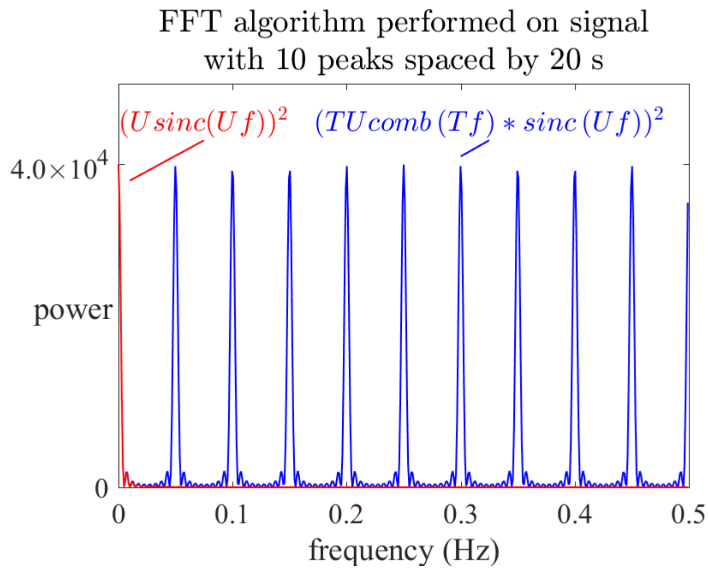


Figure 3.2: The power spectrum for an ideal insect was a perfect *comb* convolved with a *sinc* function. The *comb* was spaced by $\frac{1}{20\ s} = 0.05$ Hz and each peak was widened by the *sinc* function (red).

In Figure 3.1, there are ten peaks, but the lidar would often collect fewer. In Figures 3.3 to 3.10 we present time-domain simulation of backscattered light from insect wings with increasing peak number and the corresponding DFT. The spacing between time-domain peaks remained the same, the width and power of the frequency-domain peaks changed because the rectangular function got larger as more peaks were detected and so the frequency-domain peaks got narrower.

From these simulations, it was easy to determine the fundamental frequency and the harmonics of the modeled insect. When only two peaks were present, the magnitude of the spectrum was small and the width of each peak was very wide. However, for four or more peaks, the frequency spectrum became clear. So far, the peaks were presented as Dirac delta functions, but it was more likely that these time-domain signals would have some width and thus should be approximated as narrow Gaussian signals. A Gaussian function is represented in Equation 3.9 and the

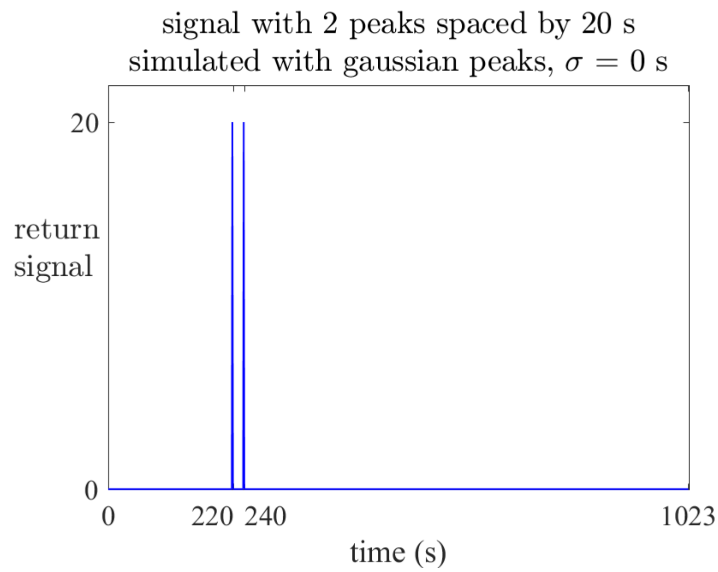


Figure 3.3: The time-domain for an ideal insect with two peaks equated to a *comb* multiplied by a *rect* function with $U = 20$ s.

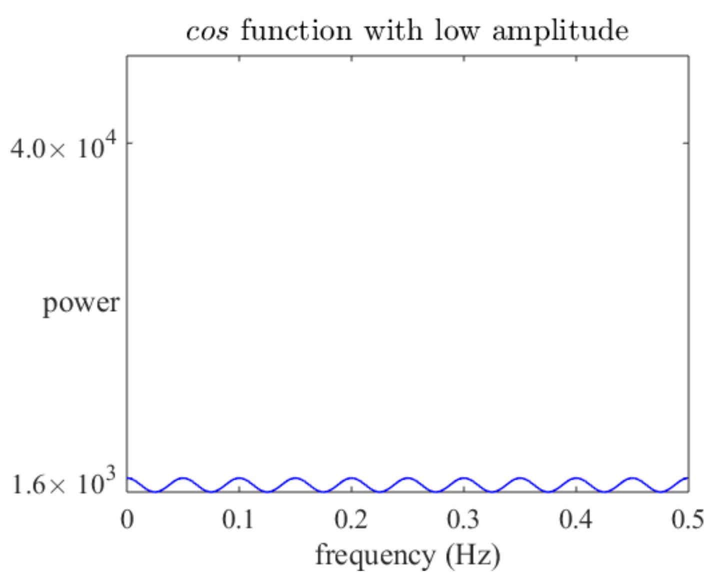


Figure 3.4: The power spectrum (of Figure 3.3) was a *cos* function with period of $\frac{1}{20\,\mathrm{s}} = 0.05$ Hz.

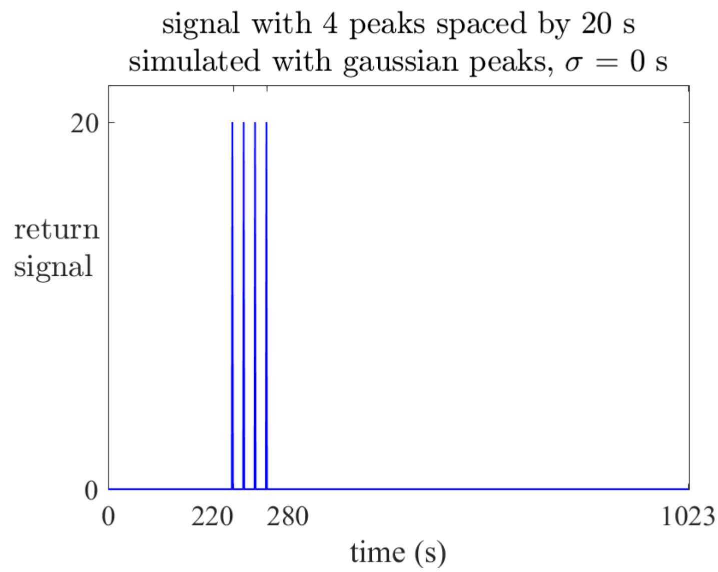


Figure 3.5: A simulated signal with 4 peaks equated to a *comb* multiplied by a *rect* function that had been widened to $U = 60$ s.

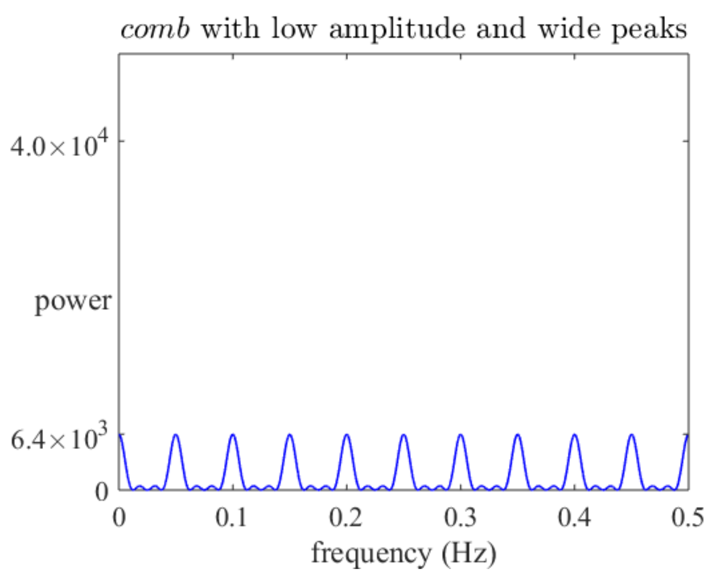


Figure 3.6: The power spectrum (of Figure 3.5) had wide and low-amplitude peaks. The peaks were spaced by $\frac{1}{20\,\mathrm{s}} = 0.05$ Hz and widened by $\frac{1}{60\,\mathrm{s}} = 0.0167$ Hz.

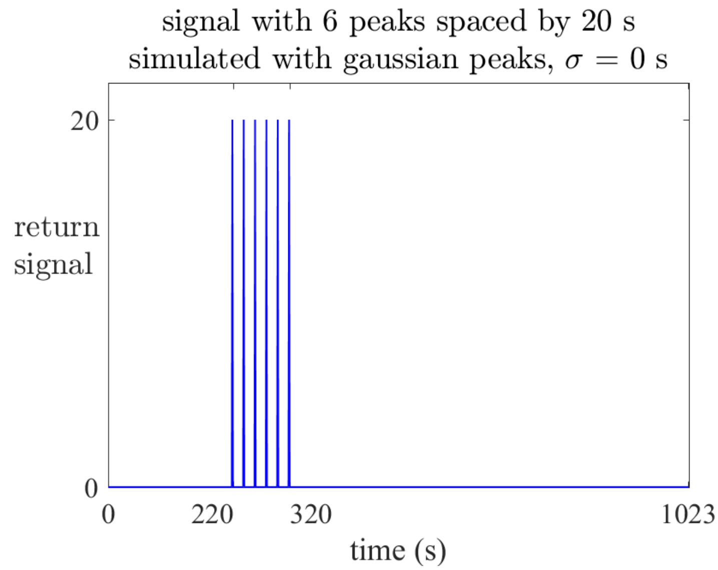


Figure 3.7: A simulated signal with 6 peaks equated to a *comb* multiplied by a wider *rect* function with $U = 100$ s.

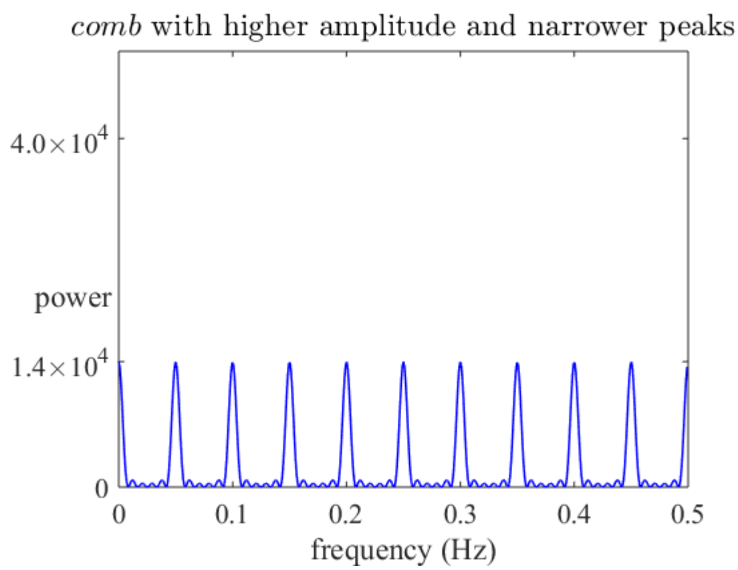


Figure 3.8: The power spectrum (of Figure 3.7) had narrower peaks with higher amplitudes. The peaks were spaced by $\frac{1}{20\,\mathrm{s}} = 0.05$ Hz and widened by $\frac{1}{100\,\mathrm{s}} = 0.010$ Hz, and were thus narrower than Figure 3.6.

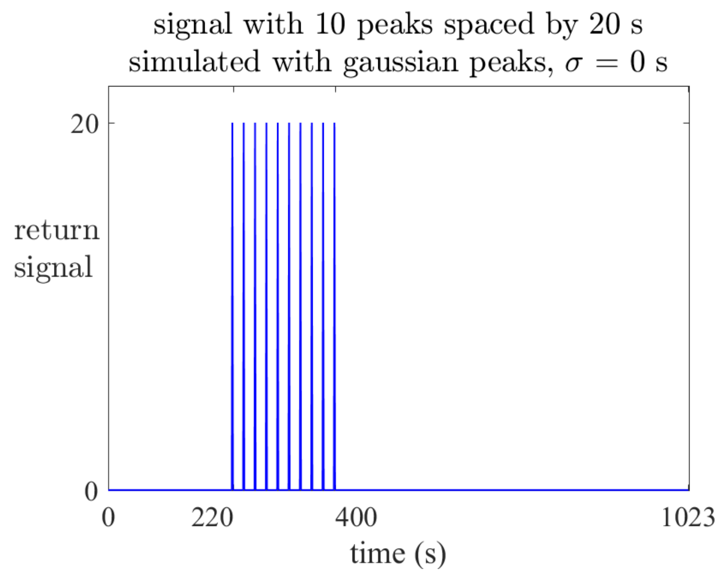


Figure 3.9: A simulated signal with 10 peaks equated to a *comb* multiplied by a wide *rect* function with $U = 180$ s.

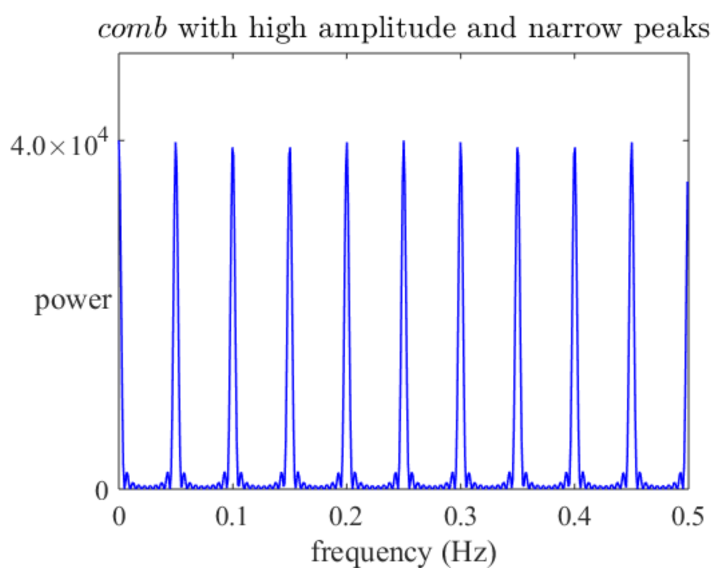


Figure 3.10: The power spectrum (of Figure 3.9) had very tall and narrow peaks. The large width of the *rect* function in the time domain equated to a narrow *sinc* function in the frequency domain.

corresponding Fourier transform in Equation 3.10. To widen each Dirac delta function in the time domain, we convolved the ideal signal with a Gaussian function of some width. The convolution in the time-domain corresponded to a multiplication in the frequency-domain and is represented mathematically in Equation 3.11 and graphically in Figures 3.11 and 3.12. It is important to note that Figure 3.11 has a Gaussian Function centered at a non-zero position. This was done so that the Fourier Transform would be executed across the entire function; if it were centered at zero, only half of the function would be present. A time-shift in the time-domain corresponds to a phase-shift in the frequency domain, and does not affect the magnitude.

$$gauss\left(\frac{t}{\sigma}\right) = e^{-\frac{1}{2}\left(\frac{t}{\sigma}\right)^2} \tag{3.9}$$

$$\mathscr{F}\left\{gauss\left(\frac{t}{\sigma}\right)\right\} = \sqrt{2\pi\sigma^2}e^{-\frac{1}{2}(2\pi\sigma f)^2} = \sqrt{2\pi\sigma^2}gauss(2\pi\sigma f) \tag{3.10}$$

$$\begin{aligned}\mathscr{F}\left\{gauss\left(\frac{t}{\sigma}\right) * comb\left(\frac{t}{T}\right) \times rect\left(\frac{t}{U}\right)\right\} = \\ \sqrt{2\pi\sigma^2}gauss(2\pi\sigma f) \times Tcomb(Tf) * Usinc(Uf)\end{aligned} \tag{3.11}$$

A narrow Gaussian function in the time domain corresponded to a wide Gaussian function in the frequency domain, and as the time domain function got wider, the frequency domain got narrower. Figures 3.13 and 3.14 are nearly identical to Figures 3.11 and 3.12, respectively, except the time domain σ increased form 2 to 3 s. The frequency domain, was multiplied by a narrower function, and so the low frequency peaks appeared higher in magnitude while the higher frequency peaks appeared lower

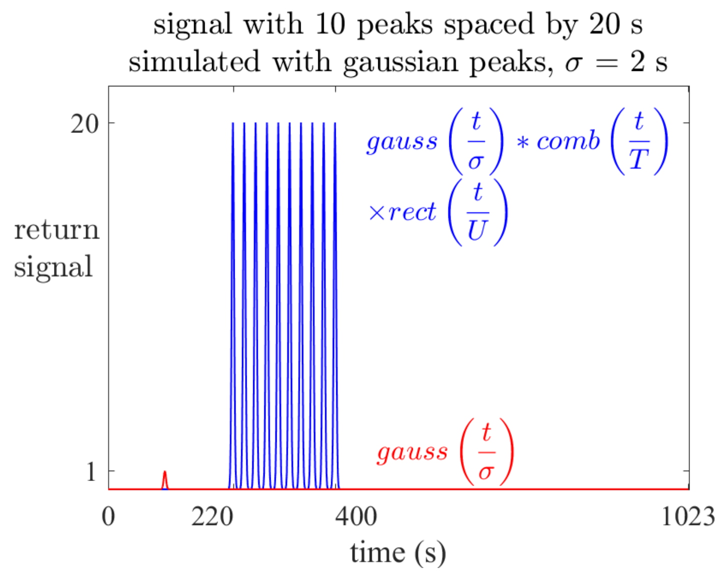


Figure 3.11: A Gaussian function (with $\sigma = 2$ s) convolved with a perfect Dirac comb and windowed by a rectangular function widened the peaks in the time domain. Note that the time shift of 100 s of the Gaussian function (in red) does not affect the magnitude of the function power spectrum after a DFT (it only affects the phase).

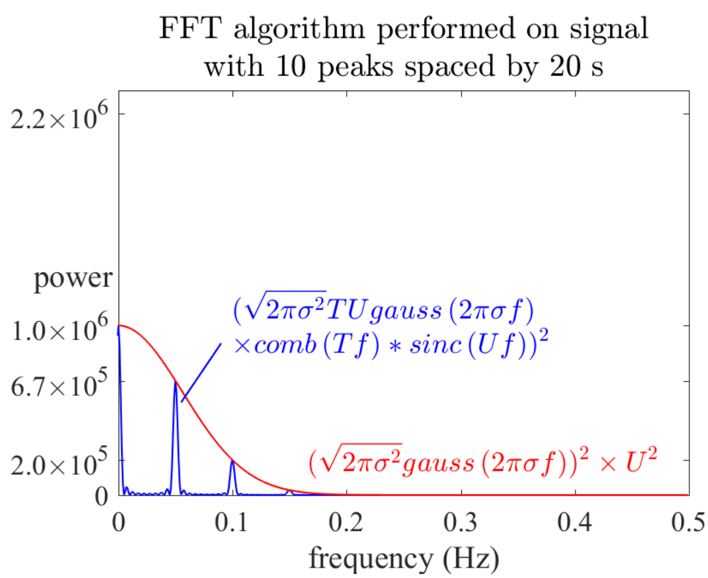


Figure 3.12: The addition of a wide Gaussian function (with $\sigma = \frac{1}{(2\pi)\times 2\,\mathrm{s}}$) reduced the higher order terms and increased the magnitude of the lower frequencies. Note that the Gaussian function (in red) was multiplied by $U^2 = 4 \times 10^4$ in order to compare with the other function.

in magnitude. This showed us that insects with wing-beat patterns that produce wide Gaussian functions would suppress the higher oder harmonics, but increase the magnitude of the fundamental and lower order harmonics.

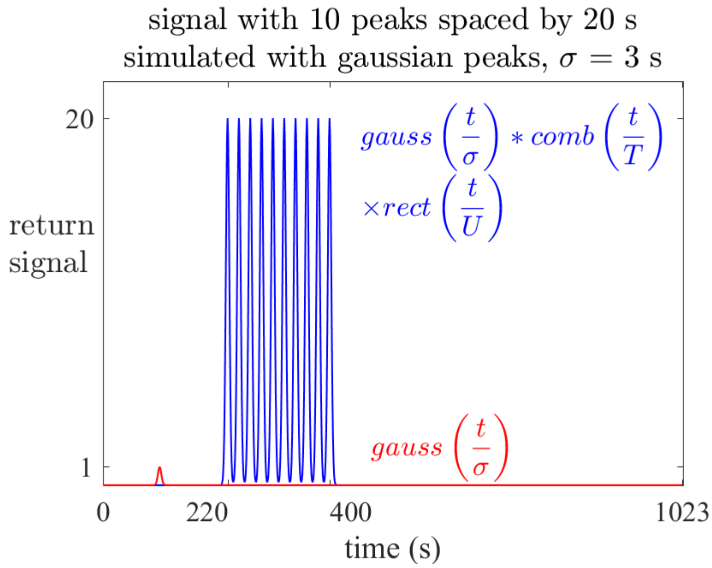


Figure 3.13: A wider Gaussian function (with $\sigma = 3$ s) convolved with a *comb* and windowed by a *rect* widened the peaks more than in Figure 3.11. Note that the time shift of 100 s of the Gaussian function (in red) does not affect the magnitude of the Fourier Transformed function (it only affects the phase).

The above functions were helpful in informing us how different wing-beat patterns shaped the frequency space. In real-world measurements, however, noise was a certainty. Insect wing-beat frequencies would not be perfectly consistent and they would produce a variety of inconsistent amplitudes. The following series of figures represents what a realistic insect wing-beat signal would likely look like in the time and frequency domains. The first set of plots included a noise floor, that is, instead of a signal of zero around the sharp peaks, random noise was present. Then noise was added to the entire signal, including amplitude variations in the sharp peaks. Further, variations between peaks in time was presented, and finally, the width of the time-domain peaks was altered.

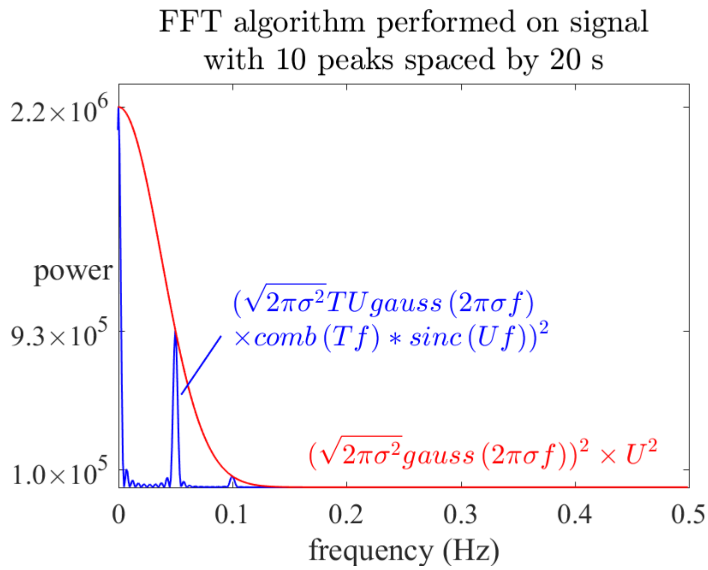


Figure 3.14: The narrower Gaussian function (with $\sigma = \frac{1}{(2\pi)\times 3\,\mathrm{s}}$) reduced the higher order terms more than Figure 3.12 and also increased the magnitude of the low frequencies. Note that the Gaussian function (in red) was multiplied by a $U^2 = 4\times10^4$ in order to make the comparison easier.

Noise Floor

The inclusion of a small noise floor had almost no affect on the Fourier Transform. There was a slight reduction in the magnitude of peaks in the frequency domain, but nothing too significant. If the noise increased in amplitude by an amount in the range of 0 to 10% to a range of 0 to 80%, there was a significant increase in noise within the frequency domain. Then, if the number of peaks reduced from 10 to 4, we saw a dramatic drop in the fundamental and higher order harmonic amplitudes, but a decrease in noise.

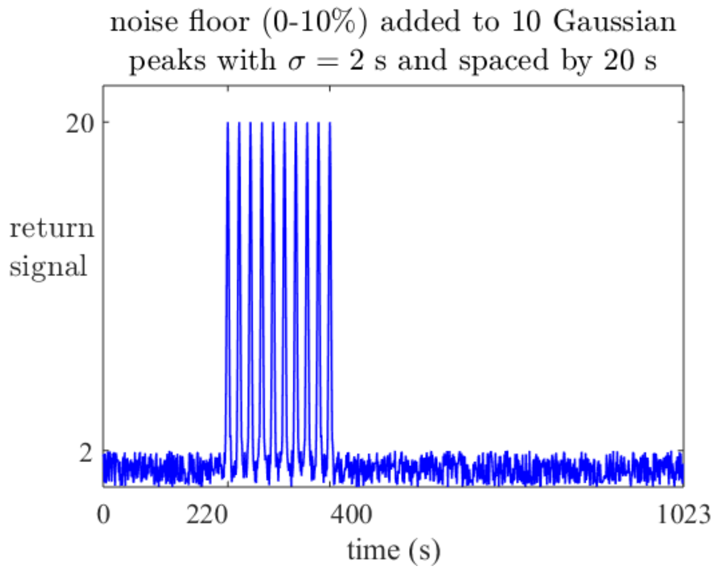


Figure 3.15: The addition of a noise floor ranging from 0 to 10% simulated a more realistic time-domain signal for an insect with a Gaussian shaped wing-beat function (with $\sigma = 2$ s).

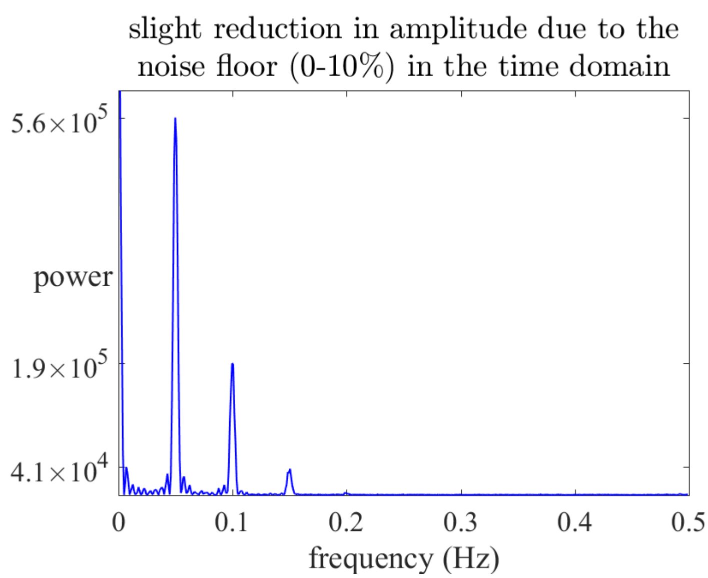


Figure 3.16: A slight reduction in amplitude was caused by the addition of a noise floor.

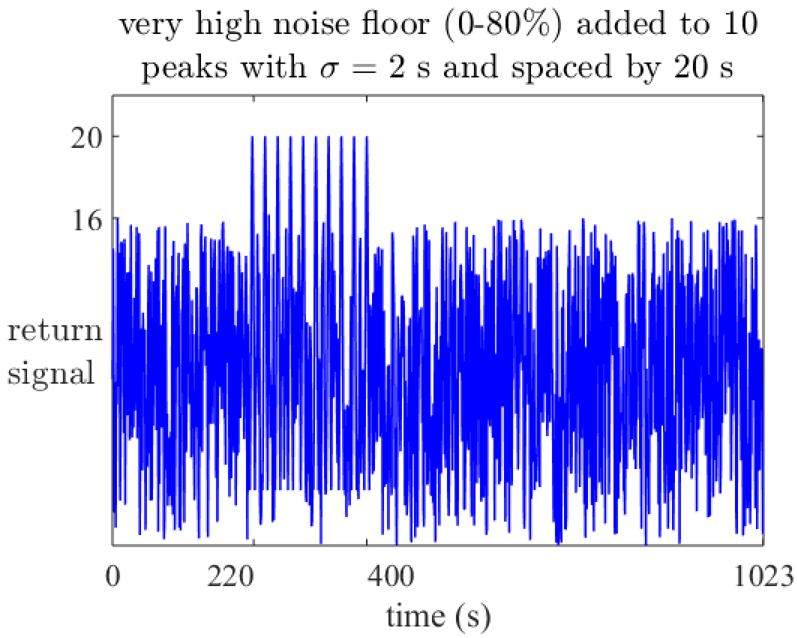


Figure 3.17: A much greater noise floor ranging from 0 to 80% simulated a very weak insect signal.

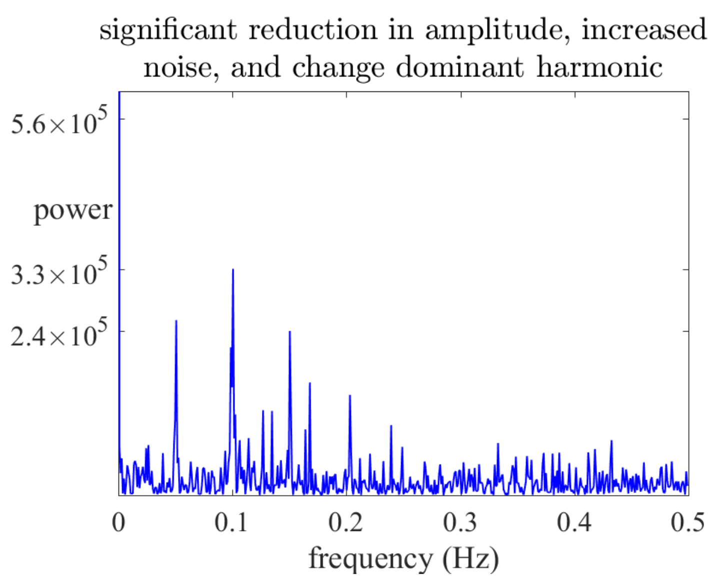


Figure 3.18: A large reduction in amplitudes, a reordering of the dominant harmonic, and an increase in noise were caused from the large noise floor in the time-domain.

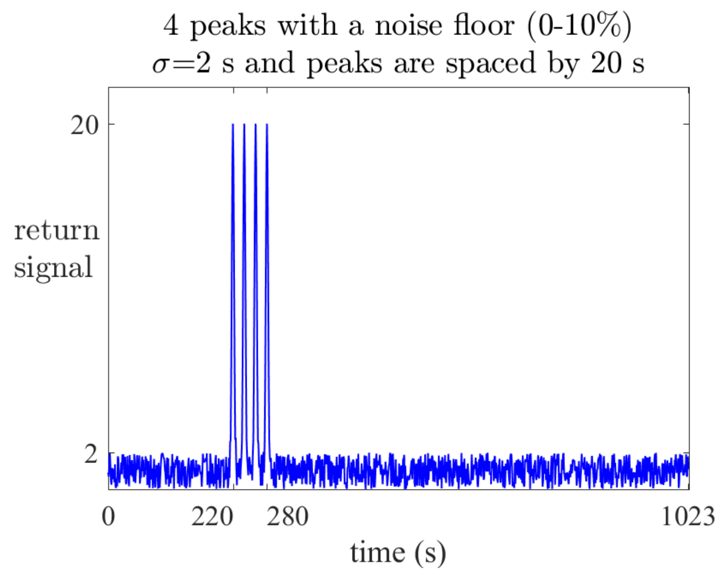


Figure 3.19: Four peaks with a noise floor ranging from 0 to 10% simulated an insect with a brief dwell time within the beam.

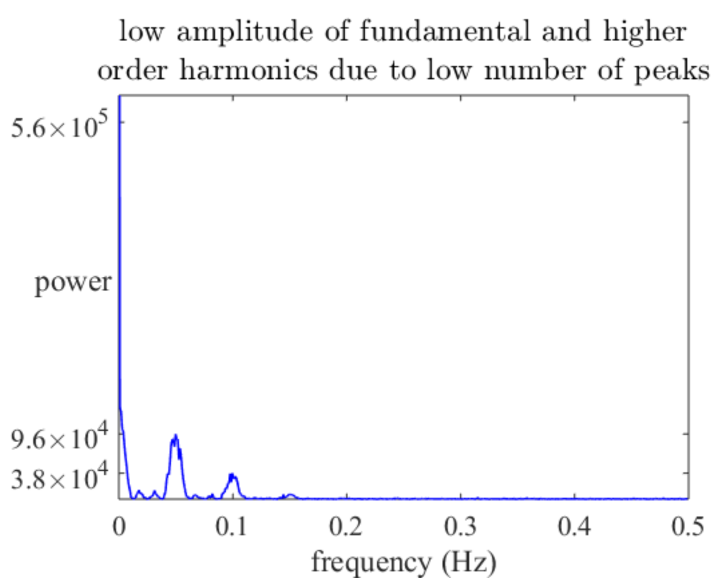


Figure 3.20: A reduction in amplitude in the fundamental and harmonics in the power spectrum was caused by the low number peaks and noise floor in the time-domain.

Peak Noise and Noise Floor

The addition of noise in the amplitude of the peaks (ranging from -50 to 0%) did not alter the power spectrum much. The amplitudes of the fundamental and higher order harmonics reduced slightly, but the amount of noise in the power spectrum was small.

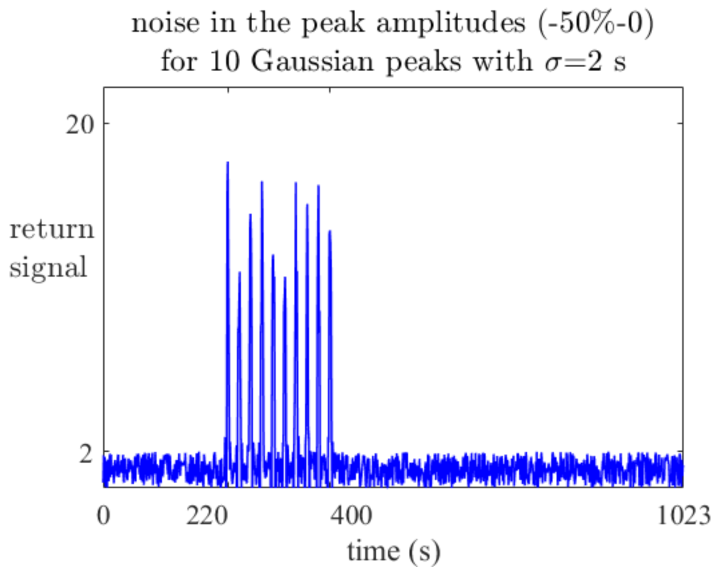


Figure 3.21: Noise added into the amplitude of each peak (ranging from -50 to 0%) represented possible variations in insect orientation. Ten Gaussian peaks (with $\sigma = 2\,\text{s}$) spaced by 20 s were simulated with a noise floor ranging from 0 to 10%.

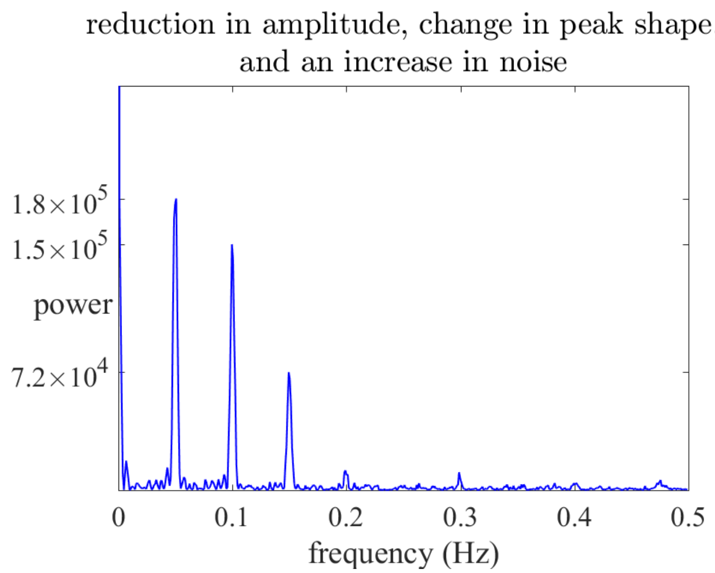


Figure 3.22: A decrease in the amplitude of the fundamental frequency, a minimal change in peak shape, and a slight increase in noise in the power spectrum were caused by noise in peak amplitude in the time domain.

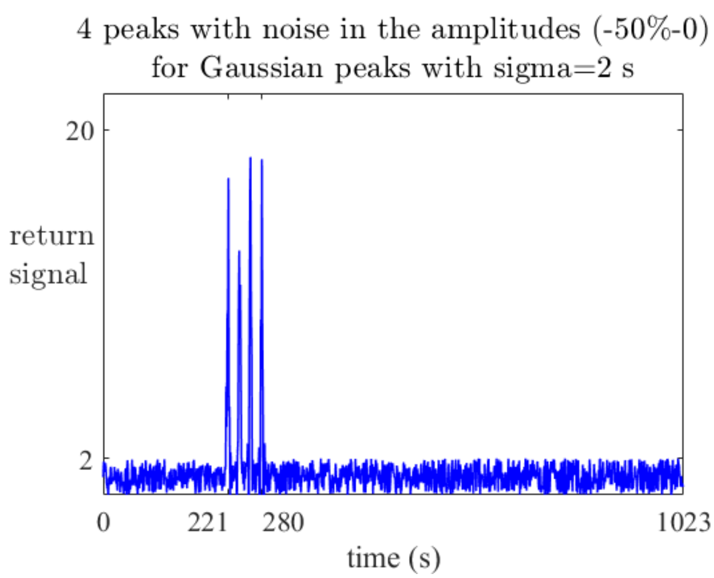


Figure 3.23: Four peaks with noise in the peak amplitudes (ranging from −50 to 0%) and a noise floor ranging from 0 to 10% were simulated.

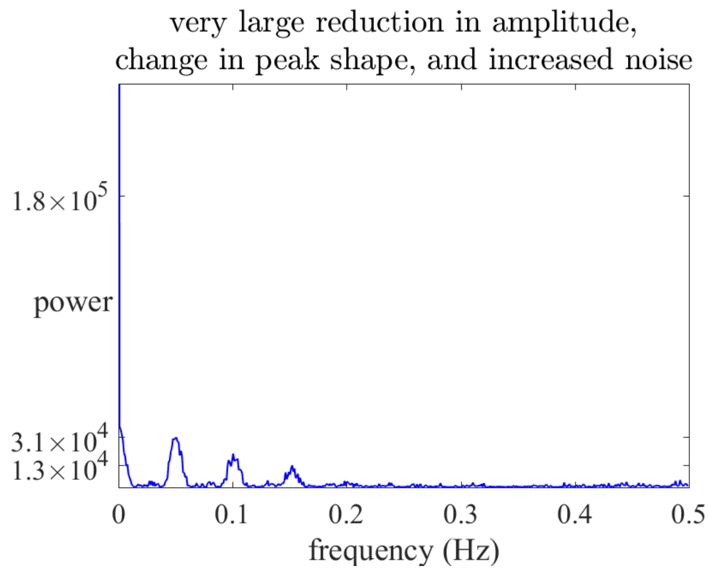


Figure 3.24: The reduction in amplitude size, the change in peak width and shape in the power spectrum were caused by the low number of peaks and noise in the time domain.

Time Jitter, Peak Noise, and Noise Floor

The addition of noise into the locations of peaks in time was done randomly between the interval of -2 and 2 s. This reduced the power of the fundamental and harmonics in the power spectrum and added in some noise.

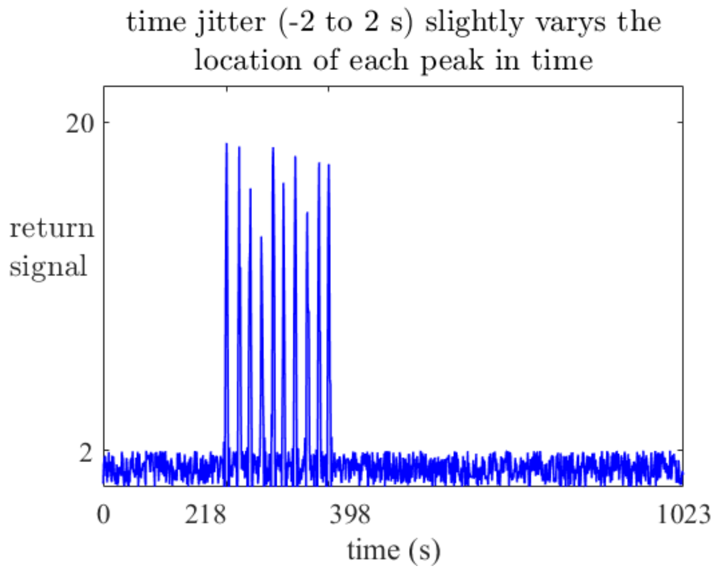


Figure 3.25: Time jitter ranging from -2 to 2 s moved the location of each peak in time. This simulation used Gaussian peaks (with $\sigma = 2$ s), a noise floor ranging from 0 to 10% and noise in the peaks ranging from -50 to 0%.

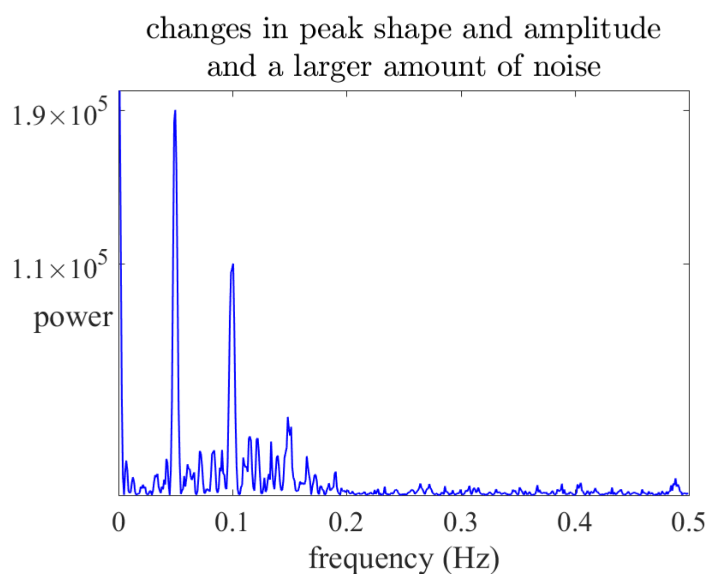


Figure 3.26: The reduction of harmonics as well as the large amount of noise in the power spectrum were caused by the time jitter in the time domain.

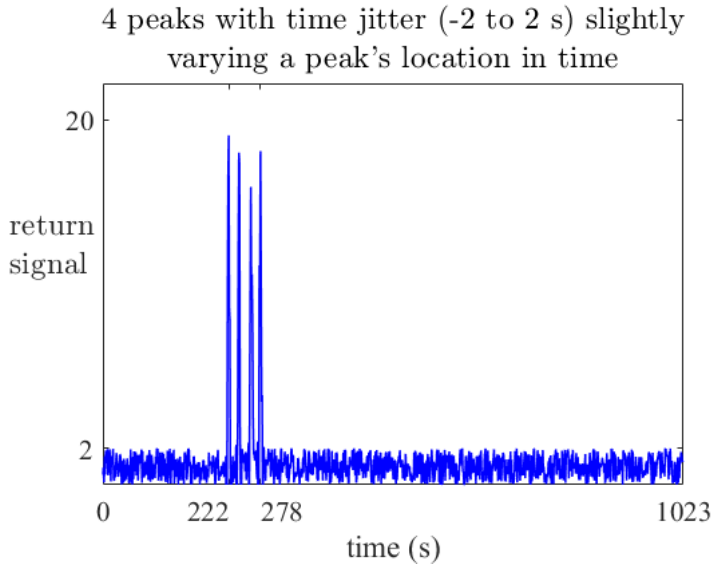


Figure 3.27: Four peaks were simulated with time jitter ranging from -2 to 2 s, Gaussian profiles (with $\sigma = 2$ s), a noise floor ranging from 0 to 10%, and noise in the peaks ranging from -50 to 0%.

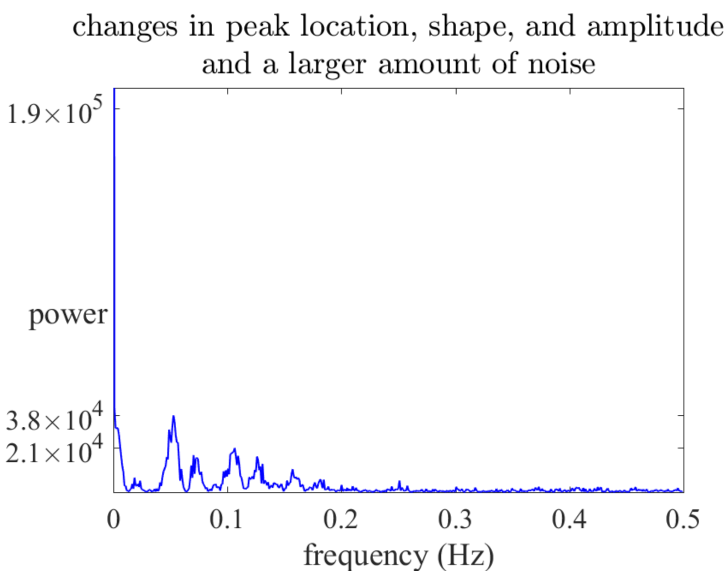


Figure 3.28: Peak locations were time-shifted, peak shapes and amplitudes were modified, and a large amount of noise in the power spectrum were caused by the low number of peaks and time jitter in the time domain.

<u>σ Jitter, Time Jitter, Peak Noise, and Noise Floor</u>

Finally, by keeping the above parameters the same and adding noise (in the range of -1–2 s) to the width (σ) of the Gaussian function for each peak, we observed a further reduction in fundamental and harmonic amplitude in the power spectrum.

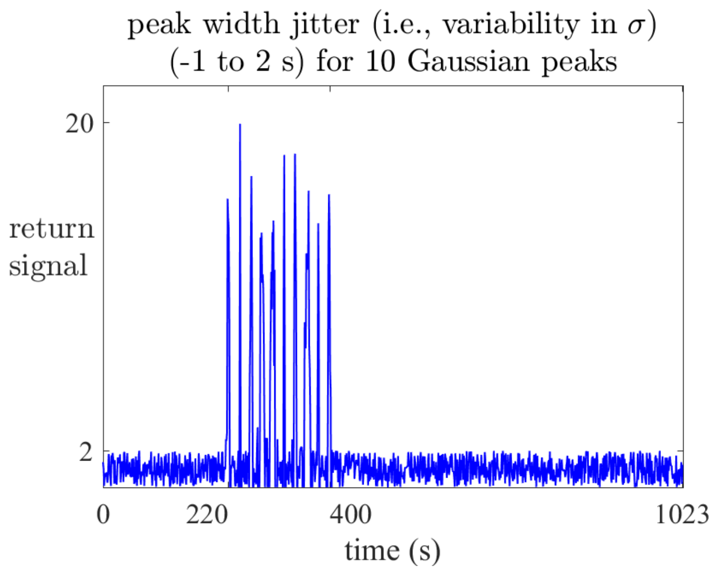


Figure 3.29: Variability in peak width, that is, noise in σ ranging from -1 to 2 s in the time-domain signal for ten Gaussian shaped peaks (with starting $\sigma = 2$ s), a noise floor ranging from 0 to 10%, noise in the peaks ranging from -50 to 0%, and time uncertainty ranging from -2 to 2 s were simulated.

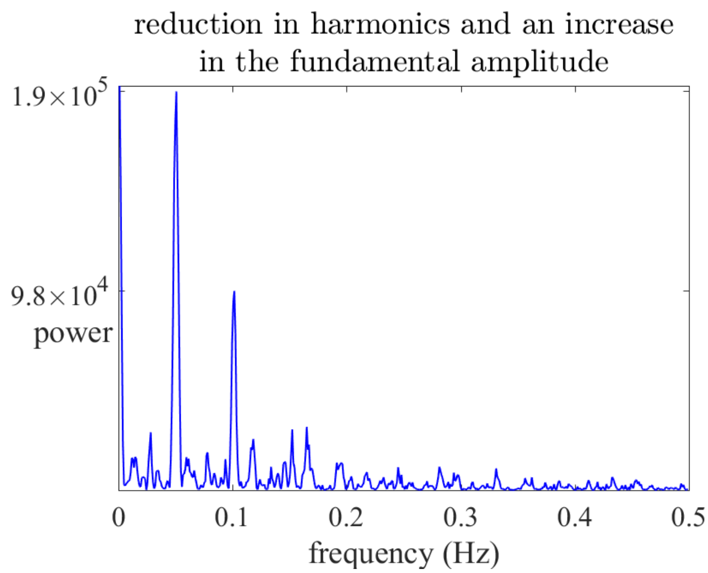


Figure 3.30: An slight increase in the second harmonic amplitude in the power spectrum was caused by the jitter in σ in the time domain. This plot is very similar to Figure 3.26 where the only difference is the inclusion of jitter in σ.

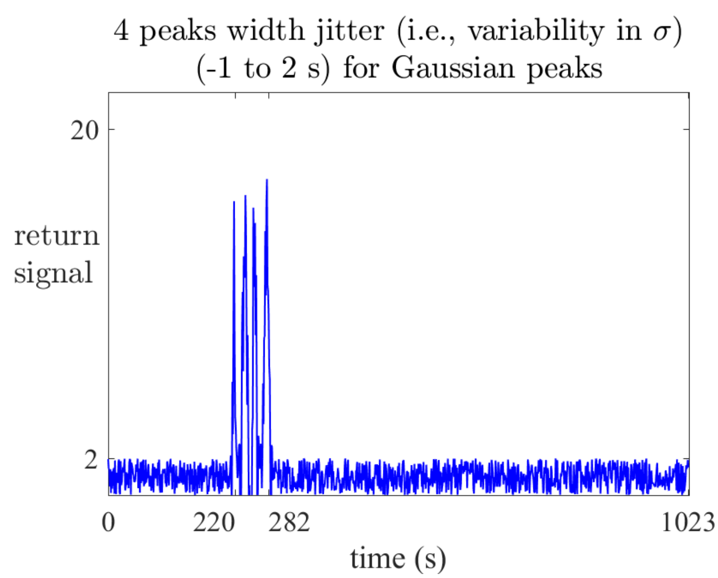


Figure 3.31: Four peaks with variability in peak width, that is, noise in σ ranging from -1 to 2 s in the time-domain signal for Gaussian shaped peaks (with starting $\sigma = 2$ s), a noise floor ranging from 0 to 10%, noise in the peaks ranging from -50 to 0%, and time uncertainty ranging from -2 to 2 s were simulated.

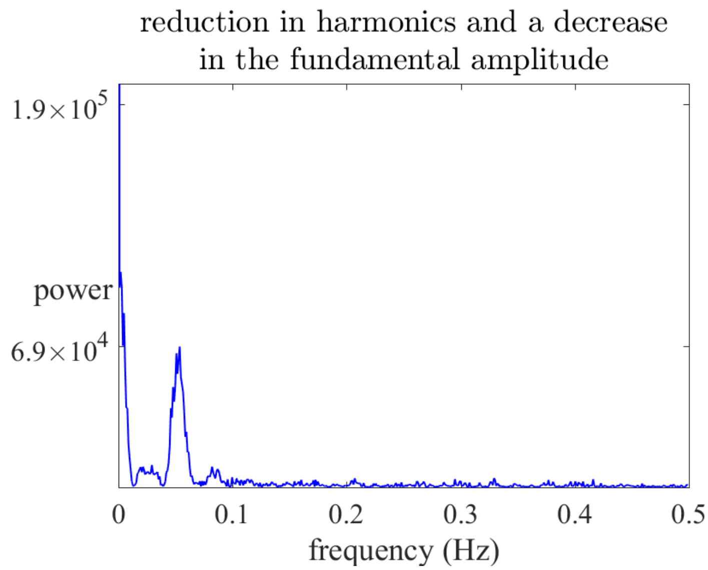


Figure 3.32: A reduction in the amplitude of the harmonics in the power spectrum was caused by a low number of peaks the jitter in σ in the time domain.

σ Jitter

In order to observe the affects from jitter in σ, exclusively, that parameter was isolated and varied from -1 to 2 s. This caused artifacts to appear and reduced the amplitude of higher order terms.

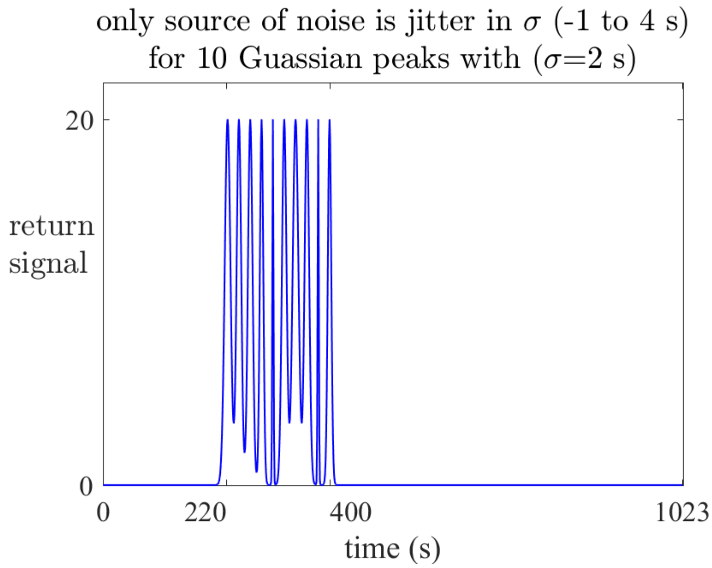


Figure 3.33: By removing all sources of amplitude noise and only simulating jitter in σ, we could see how noise in peak shape affected the frequency domain. This simulated time-domain signal had a Gaussian peak (with $\sigma = 2$ s) and σ noise ranging from -1 to 4 s.

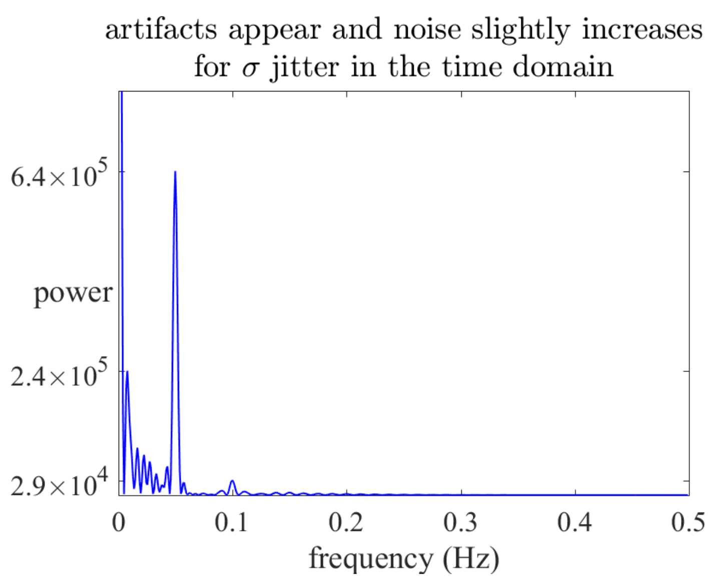


Figure 3.34: Some artifacts appear in the lower frequencies, a decrease in harmonics, and a slight increase in noise were noticeable in the power spectrum when jitter in σ was simulated with no other noise in the time domain.

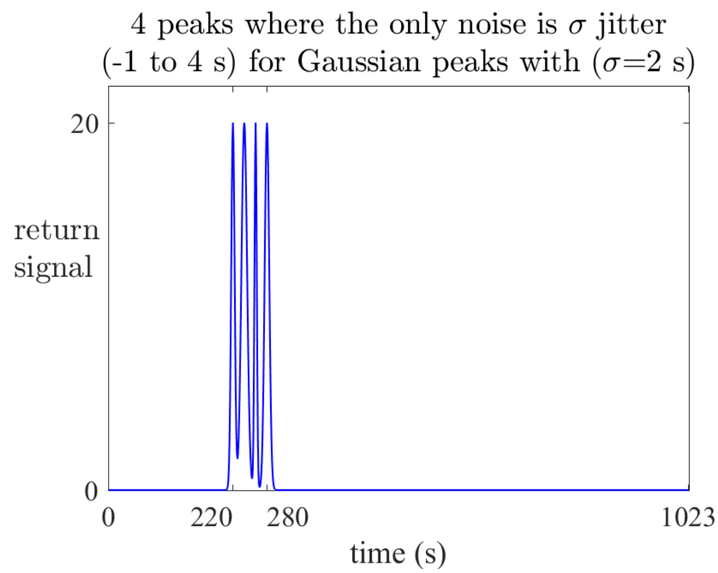


Figure 3.35: Four peaks with only jitter in σ simulated with Gaussian peaks (with $\sigma = 2$ s) and σ noise ranging from -1 to 4 s.

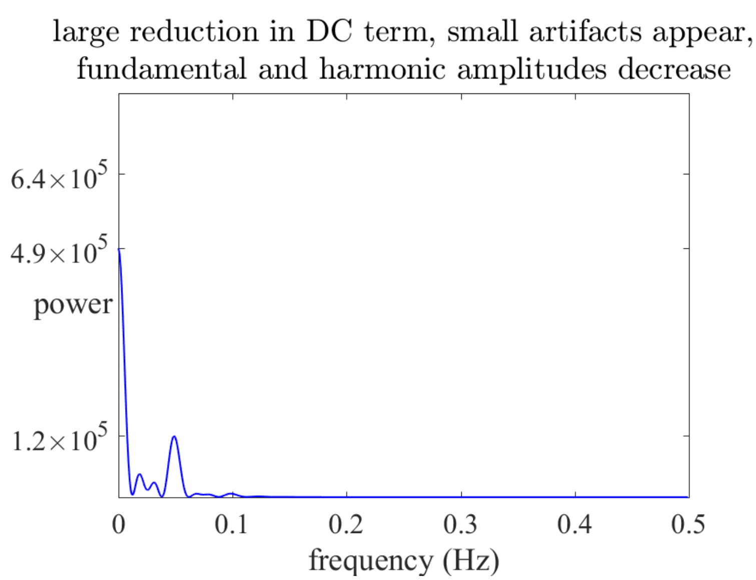


Figure 3.36: Higher order harmonics were significantly reduced and some artifacts appeared in the low frequencies in the power spectrum as a result a few peaks and jitter in σ in the time domain.

<u>Time Jitter</u>

Variations in time jitter (from -4 to 4 s) also suppressed higher order harmonics and introduced some noise into the low frequencies.

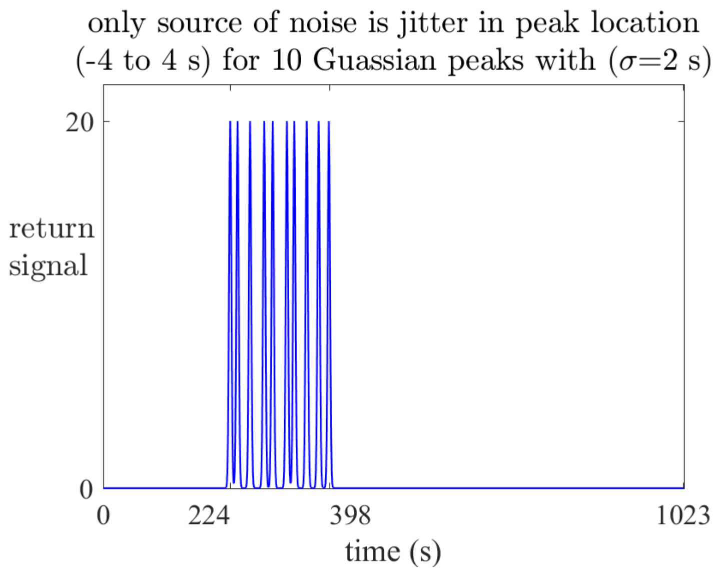


Figure 3.37: By removing all sources of amplitude noise and only simulating jitter in peak location, we could see how noise in peak location affected the frequency domain. This simulated time-domain signal had a Gaussian peak (with $\sigma = 2$ s) and time noise ranging from -4 to 4 s.

The figures presented in this section gave us an idea of what realistic signals may look like. Even for cases with a fair amount of noise, useful frequency information could be extracted. For a more detailed look at how each type of noise affects the frequency domain, Appendix C provides more plots.

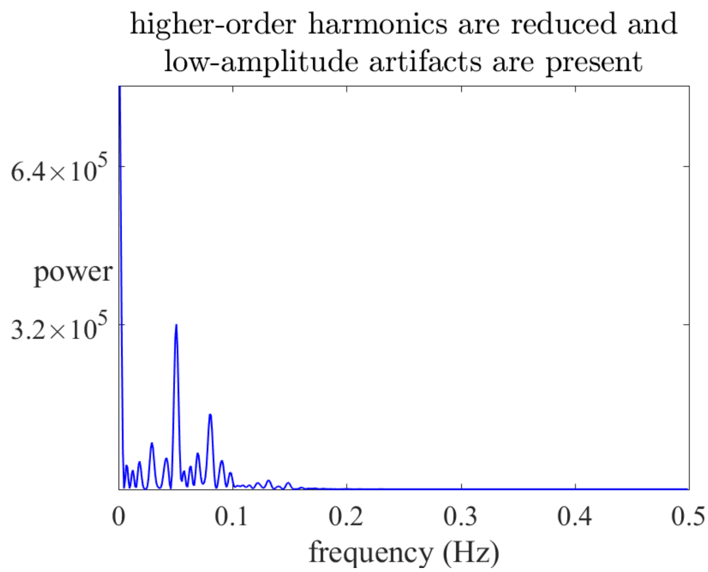


Figure 3.38: Higher order harmonics were significantly reduced and some artifacts were high in the power spectrum as a result of jitter in time in the time domain.

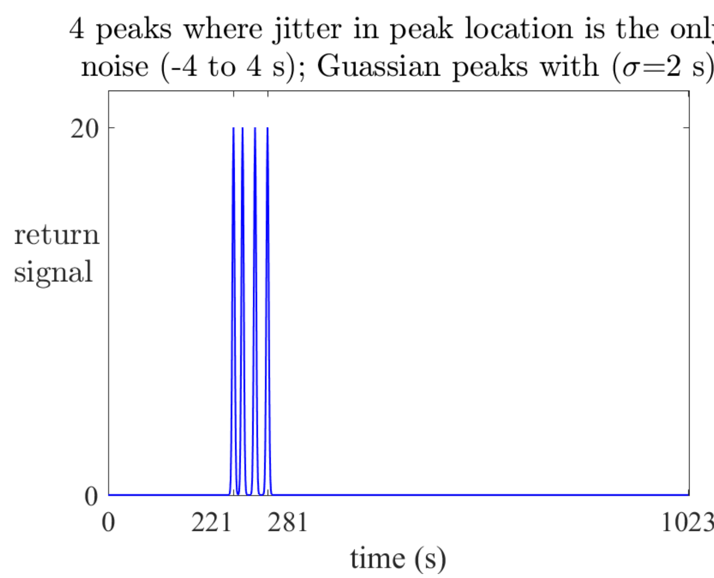


Figure 3.39: Four peaks with only jitter in peak location were simulated with Gaussian peaks (with $\sigma = 2$ s) and time noise ranging from -4 to 4 s.

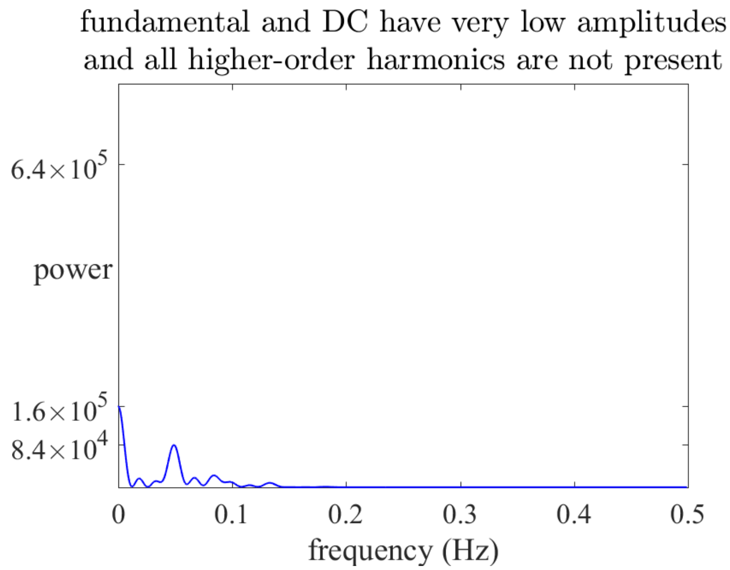


Figure 3.40: Higher order harmonics were significantly reduced along with the fundamental amplitude in the power spectrum as a result of time jitter in the time domain.

Radiometric Analysis

The radiometric analysis helped determine the maximum range bin in which the lidar system could detect an insect, that is, have a large signal-to-noise ratio (SNR). This value was critical in determining how large of a region we were able to scan and dictated certain parameters of our setup, such as how large of a beam diameter we could use. Additionally, our daytime signal-to-background ratio was calculated and dictated whether or not we could run experiments during the day as well as during the night.

The radiometric analysis was performed in two parts. First, we made assumptions about certain values (shown in Table 3.2), such as the reflectivity of an insect's wings, and calculated the detected flux, the signal-to-background ratio (SBR), and the SNR. Then we varied these factors for a set SNR value. Table 3.1

defines each fixed parameter in the following calculations, and the variables (along with their assumed values for the first set of calculations) are defined by Table 3.2.

SNR Calculation

The system used a pulsed laser with typical pulse duration (Δt_{laser}) of 0.6 ns and pulse energy of 3 µJ (Q_{laser}). So the pulse flux (Φ_{laser}) was determined by the following equation

$$\Phi_{laser} = \frac{Q_{laser}}{\Delta t_{laser}} = 5\,\mathrm{kW} \tag{3.12}$$

The amount of flux was reduced as the beam exited the system due to the imperfect mirror reflections and lens transmissions. The reflectivity of one mirror oriented at about 8° (ρ_{mir8}) was 99.6% and the reflectivity of the other mirrors oriented at 45° (ρ_{mir45}) was 99.4%. Finally, the transmission of the two lenses (τ_{lens}) in the transmitter optics had transmission values of 99.5%. Considering that one mirror was at about 8°, five mirrors were at 45°, and two lenses were in the path of the beam, the flux reduced to

$$\Phi_{out} = \Phi_{laser} \times (\rho_{mir8} \times \rho_{mir45}^5 \times \tau_{lens}^2) = 4.79\,\mathrm{kW}. \tag{3.13}$$

For 100 m or so, the atmospheric transmission (τ_{atm}) was no more than 98.3% which meant the amount of flux transmitted was

$$\Phi_{tm} = \Phi_{laser} \times \tau_{atm} = 4.71\,\mathrm{kW}. \tag{3.14}$$

The laser had an inherent divergence angle (α_{laser}) of 0.002 rad and the beam expanded by a factor of 10 due to the expanding lens with a focal length of −25 mm and the collimating lens with focal length of 250 mm placed 225 mm from the

Table 3.1: Symbols, descriptions, and set or calculated values for the radiometric analysis with assumptions defined by Table 3.2.

symbol	description	fixed or calculated value
Δt_{laser}	typical pulse duration	0.6 ns
Q_{laser}	pulse energy	3 µJ
λ	laser wavelength	532 nm
Φ_{laser}	pulse flux	5 kW
ρ_{mir8}	mirror reflectivity at 8°	99.6%
ρ_{mir45}	mirror reflectivity at 45°	99.4%
τ_{lens}	transmission of lens	99.5%
Φ_{out}	flux at lidar output	4.79 kW
τ_{atm}	atmospheric transmission	98.3%
Φ_{tm}	transmitted flux	4.71 kW
α_{laser}	laser divergence	0.002 rad
D_{beam}	beam initial diameter	50.8 mm
E_{beam}	beam initial irradiance	739.7 kW/m^2
L_{wing}	scattered wing radiance	35.3 kW/(m^2 sr)
$r_{obstruction}$	radius of telescope obstruction	50.8 mm
$A_{lenstube}$	lens tube obstruction	7099.3 mm^2
A_{Teff}	effective telescope aperture	577.6 cm^2
$\Omega_{s,ep}$	solid angle	2.5×10^{-9} sr
TP	throughput	1.45×1.44^{-10} m^2 sr
ρ_{tel}	telescope mirror reflectivity	94%
τ_{tel}	Schmidt plate transmission	98%
τ_{filter}	laser line filter transmission	61.2%
$\Phi_{detected}$	detected power	1.6 µW
A_{det}	detector area	48.1 mm^2
f_T	telescope's focal length	of 3048 mm
E_{sun}	solar irradiance at the ground	10.83 W/m^2
ρ_{bg}	background (vegetation) reflectivity	15%
Φ_{bg}	background flux	0.0492 µW
η_{pmt}	PMT quantum efficiency	0.158
q	charge of an electron	1.602×10^{-19} C
h	Planck's constant	6.626×10^{-34} m^2 kg/s
c	speed of light	2.998×10^{8} m/s
$\bar{i}$	RMS PMT current	0.111 µA
i_d	PMT dark current	1 nA
i_c	cathode current	0.112 µA
Δf	ADC electrical bandwidth	100 MHz
k	Boltzmann constant	1.38×10^{-23} J/K
T	temperature	300 K
R	resistance	50 Ω

Table 3.2: Variable symbols, thier descriptions, and their assumed values for the radiometric analysis.

symbol	description	assumed value
α_{beam}	beam divergence angle	0.0002 rad
ρ_{wing}	insect wing reflectivity	15%
A_{wing}	area of wing that reflected laser light	25 mm^2
$r_{aperture}$	full aperture radius	152.4 mm
d_{ins}	distance to insect	100 m

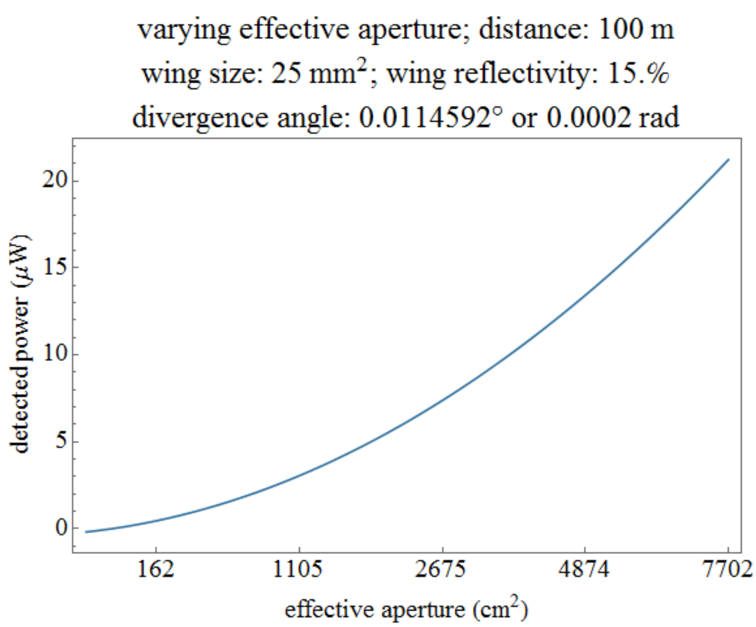


Figure 3.41: The detected power increased significantly as the telescope effective aperture area increased. The simulation was processed assuming that the distance to the insect (d_{ins}) was 100 m, the insect wing size was 25 mm^2, the wing reflectivity was 15%, and the divergence angle was 0.0002 rad.

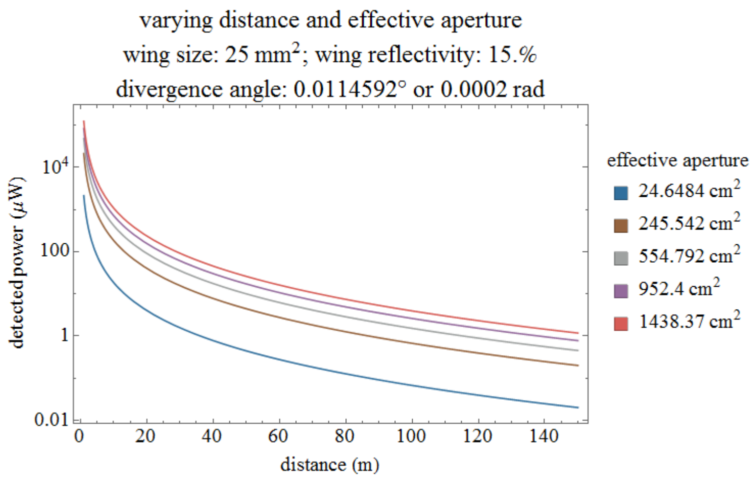


Figure 3.42: The detected flux as a function of distance from lidar fell off quickly for various effective apertures.

first lens. Thus, the divergence angle reduced by a factor of 10. The expanded and collimated beam had a diameter (D_{beam}) of 50.8 mm and divergence angle (α_{beam}) of 0.0002 rad. This divergence angle was an estimate based on calculations, but it was likely that turbulence in the atmosphere made this value larger. For the following calculations, we assumed that $\alpha_{beam} = 0.0002$ rad (this varies in the next section). The irradiance of the beam (E_{beam}) was

$$E_{beam} = \frac{\Phi_{tm}}{\pi \times (d_{ins} \times tan(\alpha_{beam}) + \frac{D_{beam}}{2})^2} = 739.7 \, \frac{\mathrm{kW}}{\mathrm{m}^2}. \tag{3.15}$$

Another assumption we made was the reflectivity of an insect wing (ρ_{wing}) and the area from which the laser light reflected (A_{wing}). It should be noted that A_{wing} did not necessarily represent the entire wing of an insect. The shape of the wing could have been curved and only a portion of it would reflect light. Therefore, it was more likely that this term represented a smaller region from which a glint reflected. We estimated that the reflectivity was 15% and the glint area was 25 mm^2 (these vary in the next section). The scattered radiance off of the wing (L_{wing}) was therefore

$$L_{wing} = \frac{E_{beam} \times \rho_{wing}}{\pi \ sr} = 35.3 \, \frac{\mathrm{kW}}{\mathrm{m}^2 \, \mathrm{sr}}. \tag{3.16}$$

The effective clear aperture of the telescope was one of the most important factors in the design. The relationship between effective clear aperture area and detected power is plotted in Figure 3.41. Naturally, a large clear aperture was necessary for detecting insects at distant range bins. The largest available telescope we had was a 304.8 mm Schmidt-Cassegrain. Since we were using a Schmidt-Cassegrain telescope, we had to account for the obstruction from the secondary mirror at the front aperture. The radius of the full aperture ($r_{aperture}$) was 152.4 mm (this varies in the next section), the radius of the obstruction ($r_{obstruction}$) was 50.8 mm, and the projected area from the lens tube assembly ($A_{lenstube}$) was estimated to be 55.9 mm × 127 mm. The effective aperture (A_{Teff}) was therefore

$$A_{Teff} = \pi \times (r^2_{aperture} - r^2_{obstruction}) - A_{lenstube} = 577.6 \, \mathrm{cm}^2. \tag{3.17}$$

Figure 3.42 plots how the detected flux falls off in range for various telescope aperture areas. The importance of the telescope's effective aperture can be seen in the drastic differences in detected power.

To determine the throughput (TP) for the under-filled field of view scenario we had to consider the solid angle of the source (an insect wing) as seen from the sensor ($\Omega_{s,ep}$), which was

$$\Omega_{s,ep} = \frac{A_{wing}}{d_{ins}} = 2.5 \times 10^{-9} \, \mathrm{sr} \tag{3.18}$$

and the throughput was therefore

$$TP = A_{Teff} \times \Omega_{s,ep} = 1.44 \times 10^{-10} \, \mathrm{m}^2 \, \mathrm{sr}. \tag{3.19}$$

Much like the transmitter optics, the receiver optics also had imperfect transmissions and reflections. The mirrors on the telescope (ρ_{tel}) had reflectivity of 94% and the Schmidt plate (τ_{tel}) had transmission of 98%. Finally, the laser line filters (τ_{filter}) had peak transmission of 61.2%. Considering these optics and the throughput of the under-filled system, our detected power ($\Phi_{detected}$) was calculated to be

$$\Phi_{detected} = L_{wing} \times (\tau_{atm} \times \tau_T \times \tau_{lens} \times \tau_{filter}^2 \times \rho_T^2) \times \Omega_{s,ep} = 1.6\,\mu\text{W}. \tag{3.20}$$

The two main ratios we were interested in were signal-to-background ratio (SBR) and signal-to-noise ratio (SNR). Each of these provided insight into our ability to register a signal and thus identify the presence of an insect. The throughput for an over-filled field of view was determined by the telescope's effective aperture (A_{Teff}), detector area (A_{det}) of 48.1 mm^2, and the telescope's focal length (f_T) of 3048 mm. The background radiance was achieved by modeling solar irradiance at the ground (E_{sun}) for the band dictated by the laser line filters (530.5 to 533.5 nm), and found to be $10.83\,\text{W/m}^2$. Assuming the reflection coefficient for vegetation or other objects in the background (ρ_{bg}) was 15% and Lambertion, and finally, again, considering all the reflection and transmission coefficients of the receiving optics, we were able to determine the background flux (Φ_{bg}) to be

$$\Phi_{bg} = \frac{E_{sun} \times \rho_{bg}}{\pi \ sr} \times A_{Teff} \times \frac{A_{det}}{f_T^2} \times (\tau_{atm} \times \tau_T \times \tau_{lens} \times \tau_{filter}^2 \times \rho_T^2) = 0.049\,\mu\text{W} \tag{3.21}$$

which gives a

$$SBR = \frac{\Phi_{detected}}{\Phi_{bg}} = 32. \tag{3.22}$$

Thus, when we made the assumptions listed in Table 3.2, our SBR was 32 during the day.

To determine the SNR, we needed to take into account various characteristics in the system. First of all, the quantum efficiency of the PMT (η_{pmt}) was 0.158, and was determined from the provided data sheet (Figure 3.43) by interpolating between the obvious points. With this value and the charge of an electron (q), Planck's constant (h), and the speed of light (c), the root mean squared (RMS) current ($\bar{\imath}$) generated by our signal on the PMT was

$$\bar{\imath} = \frac{\eta_{pmt} \times q \times (\Phi_{detected} + \Phi_{bg}) \times \lambda}{h \times c} = 0.111\,\mu\text{A} \tag{3.23}$$

and then adding in the dark current (i_d) of 1 nA, the cathode current (i_c) was

$$\imath_c + i_d = 0.112\,\mu\text{A}. \tag{3.24}$$

For PMTs, the dominant noise source is usually shot noise. Johnson noise can occasionally be significant, but in this case it was not. Equations 3.25 and 3.26 show the equations for shot noise and Johnson noise, respectively. For shot noise, we assumed that our PMT gain (G) value was 10^5. Typically, in order to calculate the SNR, one must take the gain of the detector into account, however for the case of shot noise limited PMTs, this term falls out of the equation in the final calculations and thus is irrelevant. With the addition of the electrical bandwidth (Δf) of the ADC, which was 100 MHz, the SNR was calculated to be 59.

$$\sigma_{shot} = \sqrt{2 \times q \times G^2 \times (\bar{i} + i_d) \times \Delta f} = 1.9 \times 10^{-4}\,\mathrm{A} \tag{3.25}$$

$$\sigma_{johnson} = \sqrt{\frac{4kT\Delta f}{R}} = 1.3 \times 10^{-9}\,\mathrm{A} \tag{3.26}$$

$$SNR = \frac{G \times i_c}{\sigma_{shot}} = \frac{G \times i_c}{\sqrt{2 \times q \times G^2 \times (\bar{i} + i_d) \times \Delta f}} = 59 \tag{3.27}$$

The SNR of a system where the lens tube does not block a portion of the clear aperture would have been 62. Thus, the inclusion of the lens tube obstruction only reduced the SNR by 5%. Considering that the SNR was still much greater than one, and the relative decrease was small, the benefits of designing a system with the beam exiting at the telescope obstruction outweighed the slight decrease in SNR.

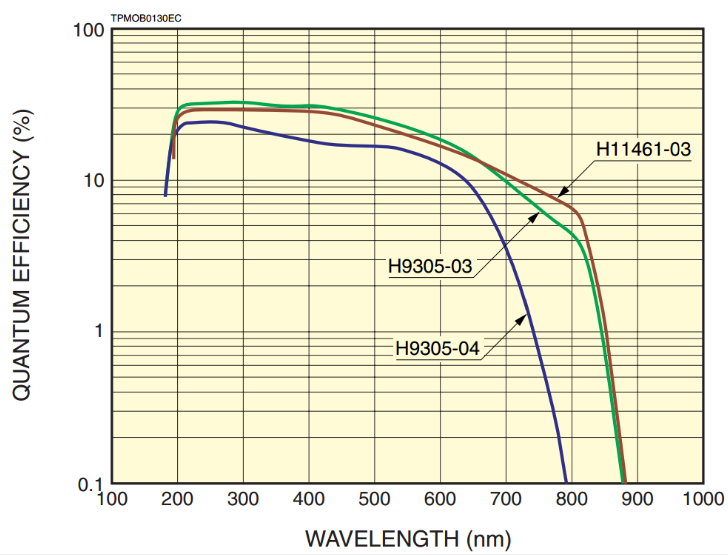


Figure 3.43: The quantum efficiency for the PMT used as the detector was necessary for our SNR calculations (H9305-04 shown in blue). *This figure is from the PMT data sheet provided by Hamamatsu*

Experimental Parameters for a Given SNR

In Table 3.2, we made some assumptions about our system, namely, the glint area of an insects wing, the reflectivity of that wing, and the divergence angle of the outgoing beam. We also fixed the telescope's effective aperture; even though it was fixed in this system, it could be changed in later iterations and thus it was treated as a variable. In the following plots, we fixed the SNR to the lowest acceptable value, 10, and monitored how different parameters affected the maximum range bin location. A more detailed set of plots along with the code developed for this radiometric analysis are also available in Appendix D.

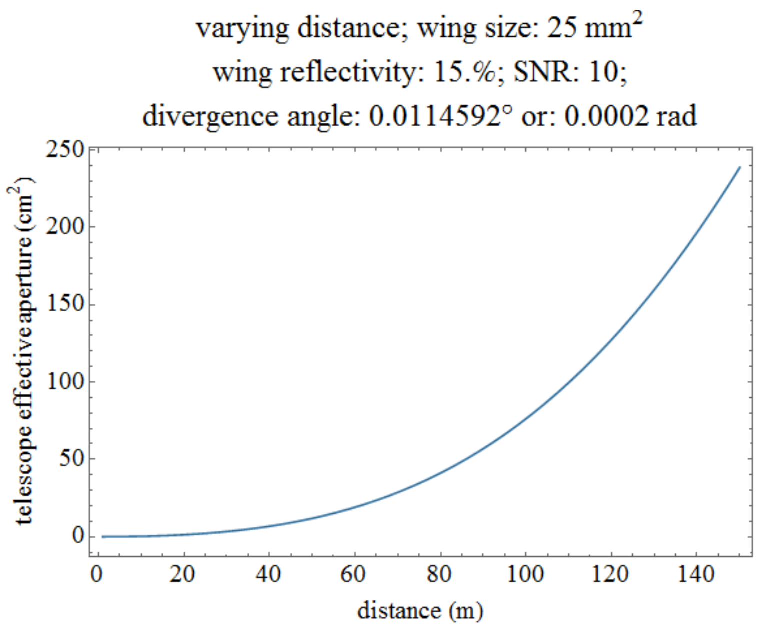


Figure 3.44: Telescope effective aperture as a function of distance. (Or: "what effective aperture is necessary to detect an insect out to a desired distance?")

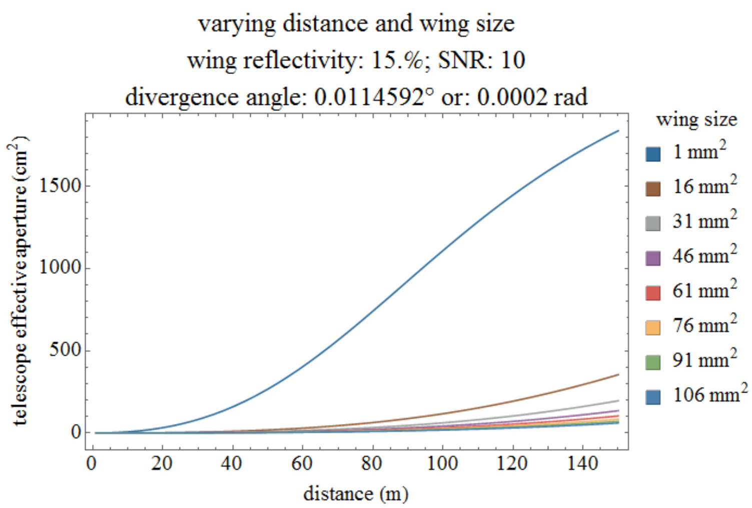


Figure 3.45: Telescope effective aperture as a function of distance for various wing sizes.

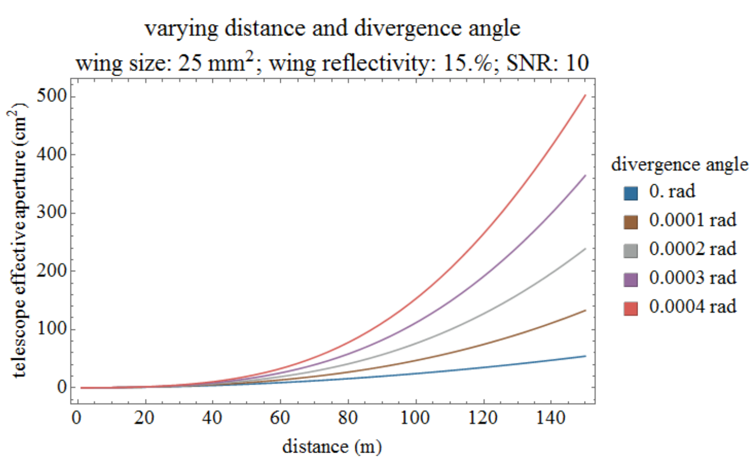


Figure 3.46: Telescope effective aperture as a function of distance for various divergence angles.

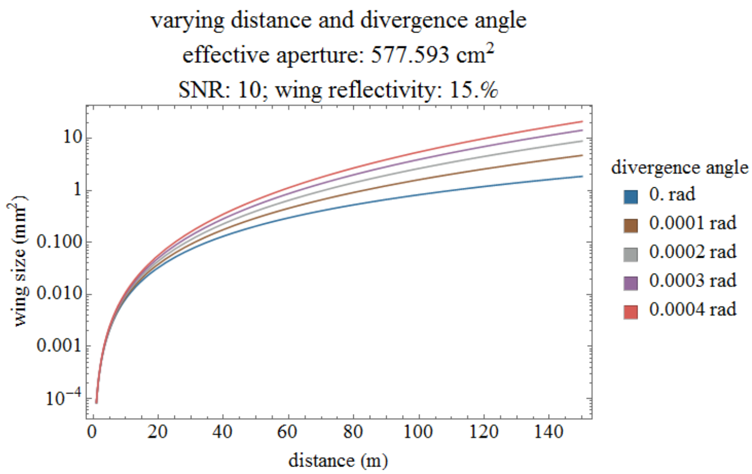


Figure 3.47: Wing size as a function of distance for various divergence angles.

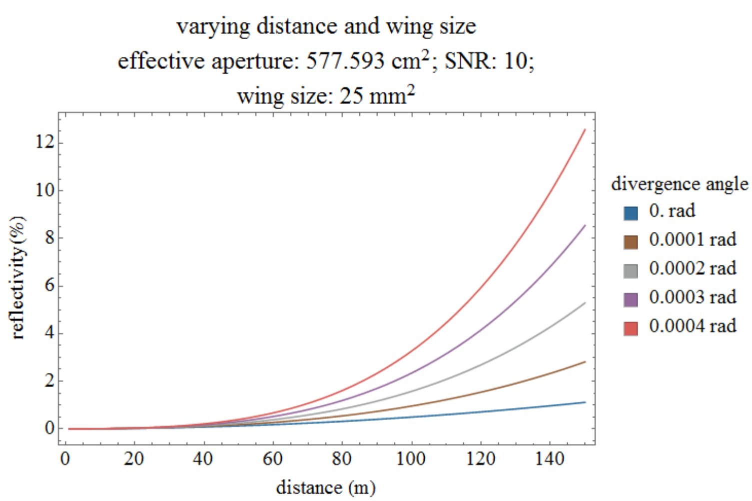


Figure 3.48: Reflectivity as a function of distance for various divergence angles.

Range Calibration

An important aspect of the lidar system was its ability to range resolve insect events. The pulsed nature of the laser allowed the lidar system to measure time of flight for each pulse and determine the range bin from which a pulse of light scattered off of an object. Either the length of pulse or speed of the ADC limited the range resolution of the system. Both values were calculated to determine the limiting factor. The pulse duration of the laser was around 0.6 ns and so the resolvable range bin was deduced by the following equation

$$\Delta R_{pw} = \frac{\Delta t_{laser} \times c}{2} = 9\,\mathrm{cm}. \tag{3.28}$$

The ADC sample rate was 200 MS/s which meant that each sample duration (Δt_{adc}) was 5 ns, and so the resolvable range bin was

$$\Delta R_{adc} = \frac{\Delta t_{adc} \times c}{2} = 74.95\,\mathrm{cm}. \tag{3.29}$$

Since $\Delta R_{adc} > \Delta R_{pw}$, the minimum resolvable range was dictated by the ADC to be $\Delta R_{adc} = 0.7495$ m. This also forced the relationship between sample number (S) from the ADC and range in meters. Thus, every sample number represented a different range bin in increments of 0.7495 m. However, the first sample number did not equate to 0 m in range due to the time a pulse spent within the transmitting optics and the length of data-transferring cables.

Although determining range was a simple calculation, the complexities in the transmitter and receiver optics did not allow for a straight forward calculation of absolute range. Measuring the distance between the laser and the output of the system as well as the path from the front of the telescope to the PMT was not a trivial task. Furthermore, the time delays from electronics and cables would also be difficult and imprecise to calculate. Instead of reaching a range correction mathematically, the correction was done experimentally.

The range bins determined by the ADC had to be calibrated to represent actual range. In order to accomplish this, we placed the lidar so that it was facing a hard target only a few centimeters away (essentially a range of 0 m). The lidar then fired a laser pulse at the hard target and monitored the reflection. The pulse was registered by the PMT and the sample number where the pulse was at a maximum, indicating the location of the hard target, was noted. In increments of about 1 m the lidar moved further from the target and performed the same procedure of firing a pulse and matching the peak with a sample number. After 36 data points, we determined the linear relationship between between the sample number in the ADC output and the distance from the lidar to be

$$range = 0.7474\,\frac{\mathrm{m}}{\mathrm{S}} \times \mathrm{S} - 5.8095\,\mathrm{m}.$$

The slope of 0.7474 m/S was very close to our calculated value of 0.7495 m/S, and although the offset of 5.8 m seemed slightly high, the value was still very reasonable considering the length of the lens tube assembly and potential electronic delays. Furthermore the small difference between slopes could only produce an error of about 20 cm at range of 100 m, which was not a large error considering the purpose of this device.

Field of View (FOV)

The first range bin of any use was the one where the laser overlapped with the field of view (FOV) of the telescope. The receiving optics were modeled with a collimated beam entering the telescope. This was done to show the chosen spacing of the optics and determine the effective focal length (EFL), and entrance pupil location ($EP_{location}$). Figure 3.49 shows the locations of the two lenses and the detector with respect to the telescope. The collimated beam entered the telescope and focused behind the telescope and began to expand, then the first lens collimated the beam, and the final lens focused it down onto the PMT. The EP was located 2030.58 mm from the focus, and the size of the detector was 3.7 mm $\times$ 13.0 mm ($x \times y$), and so the FOV was 0.911 mrad, in the horizontal direction, 3.20 mrad in the vertical, and 3.33 mrad in the diagonal, as defined by Equation 3.30 (where h was half of x, y, or $\sqrt{x^2 + y^2}$).

$$FOV = arctan(\frac{h}{EP_{location}}) \tag{3.30}$$

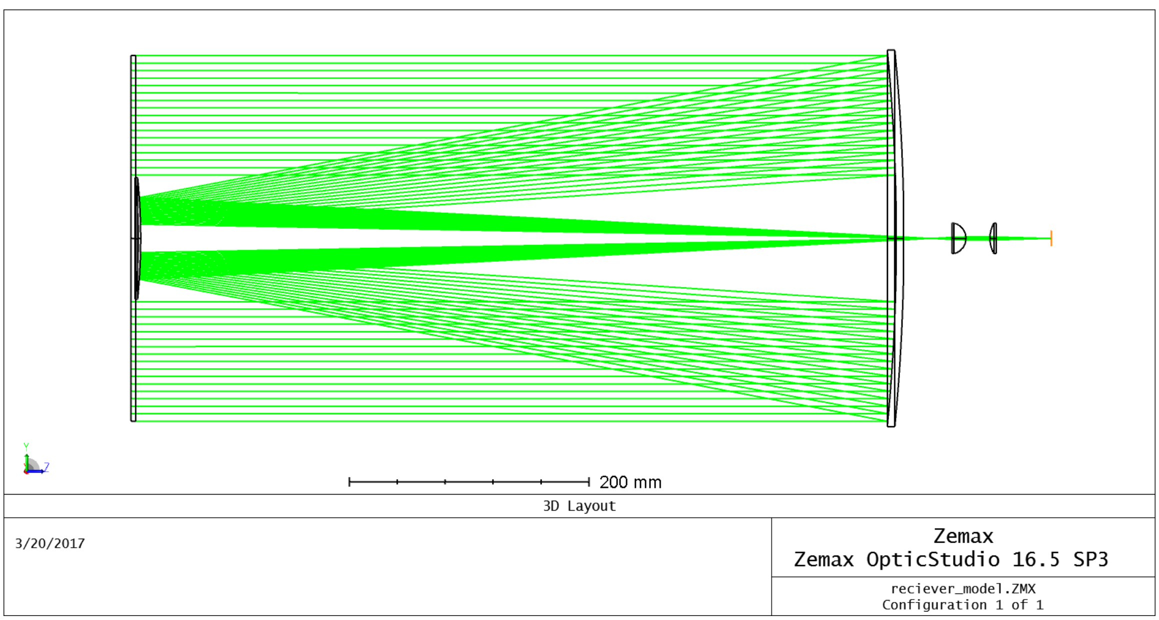


Figure 3.49: Zemax model of receiving optics for a collimated input beam. The distance from the EP to the focus was 2030.58 mm and the size of the detector was 3.7 mm $\times$ 13.0 mm ($x \times y$).

The overlap was calculated by modeling the laser divergence (of 0.2 mrad) and the telescopes *FOV* (using the largest value at the diagonal of 3.33 mrad) as two intersecting lines on a plot (see Figure 3.50). The two began to overlap at 7.2 m, and fully overlapped at 15.3 m, which was about the distance to the closest insect that was actually detected at 15.1 m. If the divergence of the laser was larger due to turbulence in the atmosphere, the overlap distance may have been even shorter. A model of the lidar system is presented in Figure 3.51, where a single ray from the expanded beam scattered from an insect wing 100 m away and was then collected by the optics and detected on the PMT.

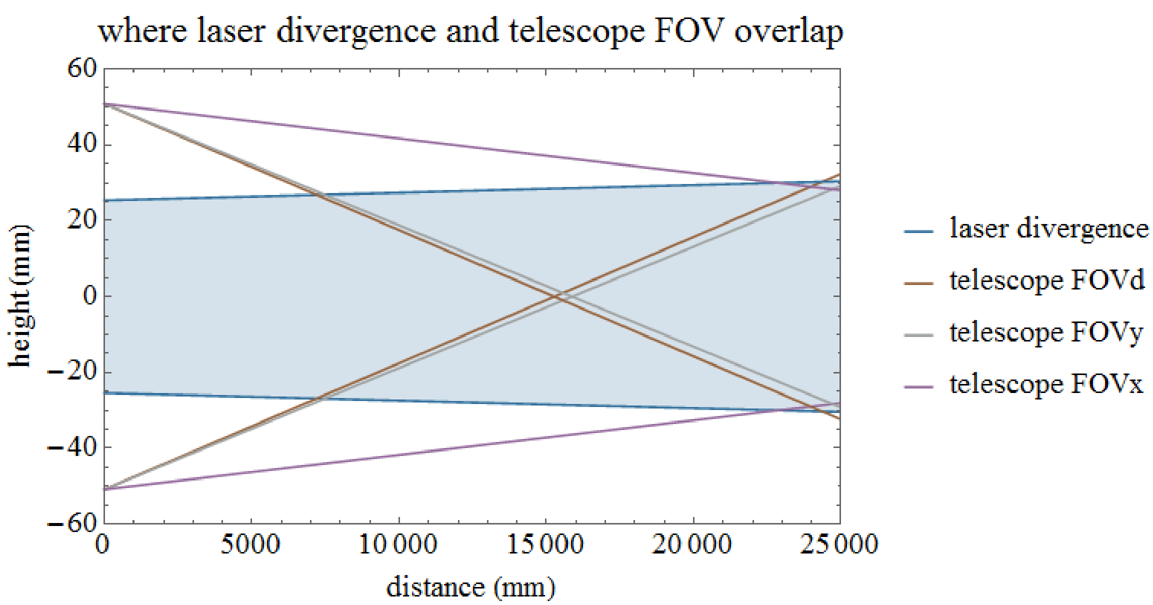


Figure 3.50: A simulation of the laser divergence and telescope *FOV* for the three cases of a rectangular detector. For the largest *FOV*, *FOVd*, the overlap began to occur at 7.2 m.

Chapter Summary

The simulated insect signals along with the radiometric analysis provided us with useful information to help us anticipate *in situ* data. The number of peaks

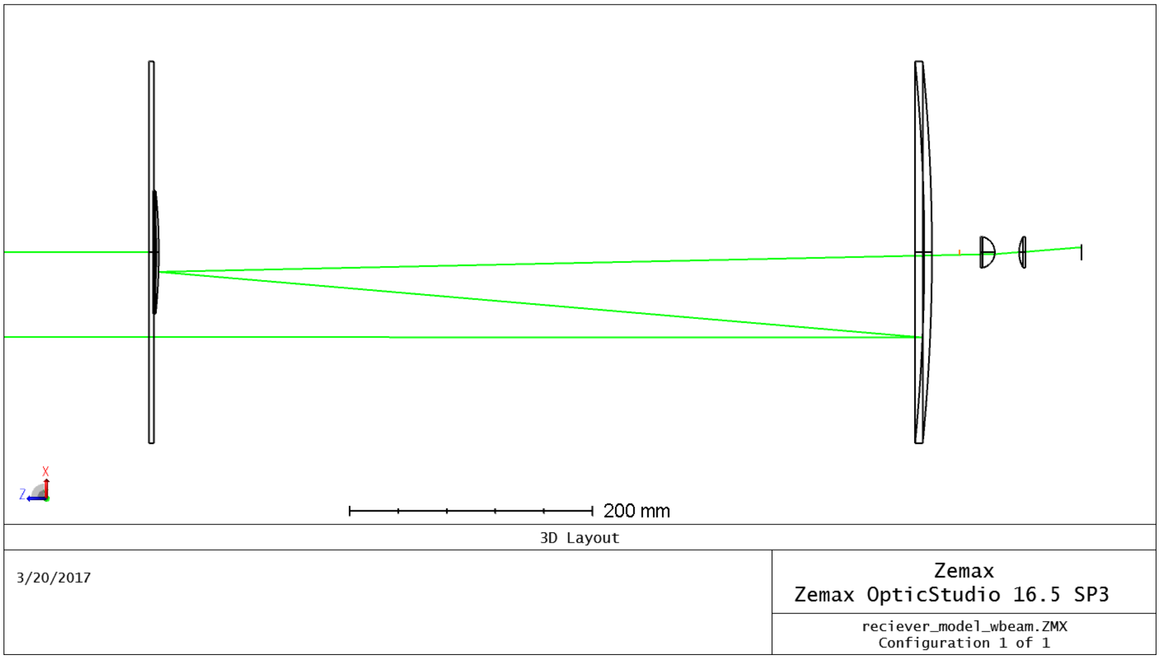


Figure 3.51: Zemax model of the lidar starting from the point where the beam leaves the system for a scenario where an insect was at a distance of 100 m.

in a time-domain signal became a very important factor as the amount of noise increased. If we were to detect an event with less than five peaks, and a low SNR, the frequency domain may look noisy and not necessarily be of much help. However, the fact that an event was detected at all, and that the time-domain signal may have some characteristics of a wing-beat signal, such an event may thus be classified as a "somewhat likely" insect.

The radiometric analysis helped us determine our maximum range bin, but it also provided us with a very useful collection of plots that we could use to make better assumptions about the scenario. For example, for a minimum acceptable SNR of 10, in order to see an insect with a wing glint area of 25 mm^2, divergence angle 0.2 mrad, and an effective telescope clear aperture of 577.6 cm^2, to see an insect out to 120 m, the wing reflectivity needed to only be about 3% (see Figure 3.48). Thus we could roughly determine what size and type of insect the system could resolve (based on wing size and reflectivity for a set SNR) and determine the specifications (divergence angle, effective aperture, etc...) of the system so that it could detect various insects. Furthermore, these plots could be used to analyze each individual detected insect. Considering the SNR and distance of a particular event, wing size and reflectivity could be estimated.

This collection of simulated wing-beat signals and detected backscatter strength not only helped guide the design of the system, but it could also be a valuable tool in learning about each individual insect event during the analysis phase. Insect wing size and reflectivity could be inferred, and insect classification could be improved, especially for very noisy signals.

EXPERIMENTS

The abundance and diversity of the insect class makes it incredibly important to biological studies, especially in the field of ecology. Many lab experiments have been conducted which have led to numerous discoveries. Similarly, new discoveries have been made thanks to field experiments, but certain technological limits have prevented entomologists from making some *in situ* measurements. Perhaps the most fundamental question to researchers is simply: where are insects in a given region and how do their positions change in response to certain variables? To be able to answer this question, a tool that can locate and count the insects was required. This chapter describes the field experiments that were conducted with the lidar system described in the Instrument Design chapter.

Three experiments are described below. Each subsequent experiment introduced more robustness to the system and also collected a larger variety of data. The experiments are presented in chronological order, and they also move toward answering the question of how insects are distributed. The first round of experiments was conducted near a beehive and was a critical step in moving forward with the instrument development. The next experiment collected non-bee insect data at a river and suggested that the instrument would be successful in the field. Finally, the instrument collected multiple days worth of insect data during the primary field experiment in Grand Teton National Park.

Beehive Measurements

The first experiments were conducted at a beehive near the campus of Montana State University. Previous studies [12–15] showed that bee detection was possible using a similar lidar and a wing-beat detection approach. In order to verify that our system was functional, we conducted numerous tests at this beehive. The entrance of the hive had a lot of activity due to the frequent entering and exiting of bees. Considering this, we pointed the laser beam near the entrance so that many bees would fly through the beam and be detected by the lidar system. We were able to detect bees on multiple occasions, during varying conditions, and at increasing distances. Over the course of a few hours, the lidar would run almost continuously while clouds would cast shadows over the beehive and on nearby fields, affecting the background signal as a function of time. Additionally, the location of the lidar was moved further away from the hive out to over 80 m and detected a wing-beat-modulated signal.

Equipment and Setup

The first iteration of the working lidar system had all the optics described in the Instrument Design chapter; unlike that system, however, it lacked the pan-and-tilt mount. Without the mount to secure the system, it had to be attached to an optical breadboard for stability. This breadboard and the attached optics were placed in the back of a car. The supporting equipment included a computer, power supply, and oscilloscope and was also placed in the trunk (Figure 4.1).

The Lidar system was essentially stationary in the back of the car. To point it at the hive, the car had to be driven so that the orientation of the lidar system was roughly pointing in the correct direction. Then, any minor adjustments were made by pushing and pulling on the optical breadboard within the trunk. Although this method lacked elegance and it was often difficult to maneuver the bulky instrument, it was sufficient for pointing the beam toward the desired direction. Once properly situated, we were able to run the system and collect data. Typically, the system would run for a few hours and collect various data over that time. As discussed in the

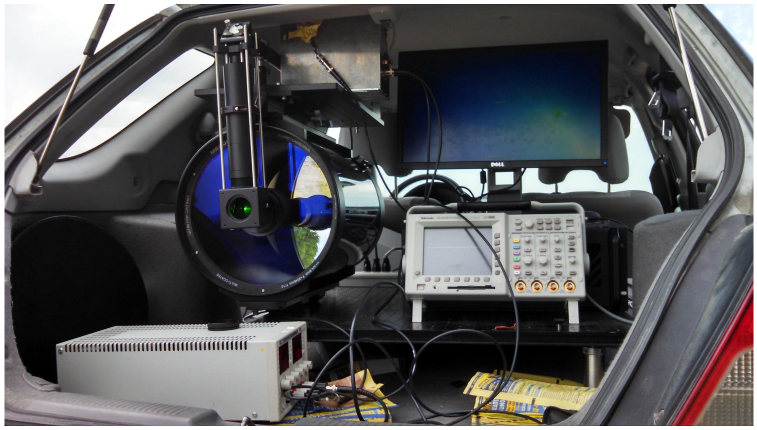


Figure 4.1: Photograph of instrument inside of a car. The first iteration of a working system did not include the scanning mount.

Instrument Design chapter, raw data had to be slightly processed before they could be analyzed, Figure 4.2 shows what some of these minimally processed data looked like.

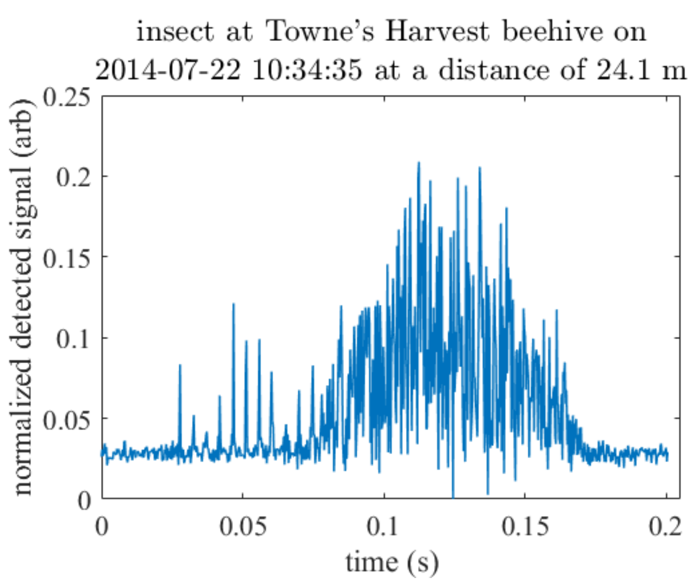


Figure 4.2: The scattered signal off a bee was collected by the lidar system at a fixed range and showed the modulated signal from the bee wings (0.025 to 0.0754 s) as well as the hard target return from the body (0.075 to 0.175 s). The distinction is critical because only a modulated signal implied that a bee was present.

Results

The data shown in Figure 4.2 were particularly informative because they showed the three different types of return signals: noise, modulated, and constant (or DC). From 0 s to about 0.025 s, the detected signal was just noise, then the modulated signal occurred from about 0.025 until roughly 0.075 s, and finally, the DC signal took place between 0.075 and 0.175 s. The series of signals could be interpreted in the following way: during the first time segment, nothing was present in that location bin; in the middle segment, a bee-wing was likely present; and finally, the system detected a hard target. A likely scenario for the presented data could be described as a bee having flown up through the beam; first the bee-wings entered the beam and that produced a modulated signal, then the body entered the beam and so the signal changed from a modulated response to a DC response.

In order to confirm that the modulated signal was due to the presence of a bee, we had to determine the frequency at which the peaks were occurring. Honeybees have wing-beat frequencies typically between 170 and 270 Hz [13] and certain obstructions, such as grass or branches blowing in the wind, would have very low oscillation frequencies in comparison. By isolating the modulated signal with a Gaussian window (see Figure 4.3) and then performing a Fast Fourier Transform (FFT) algorithm on those data, a frequency spectrum could be analyzed. In Figure 4.4, the spectrum showed some low frequencies, which were typically clumped into the DC signal and thus ignored, but also a strong peak near 200 Hz. Additionally, the harmonics were at intervals of about 200 Hz, strongly suggesting that the culprit which caused this modulated signal was, indeed, a bee.

The bee event described with Figures 4.2-4.4 was at a range of 24.1 m. The signal in the time domain had a good signal-to-noise ratio (SNR) and the frequency spectrum had distinct peaks that allowed us to distinguish the event as a bee

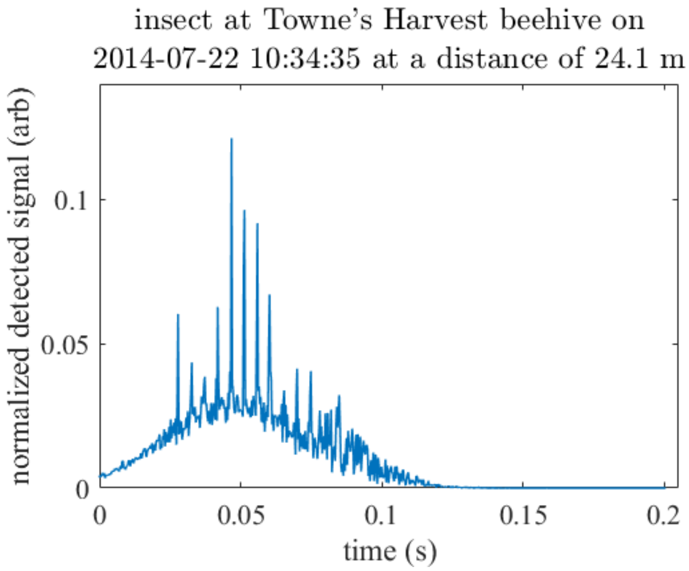


Figure 4.3: The plot shows the same data from Figure 4.2 with a Gaussian window over the modulated signal. The suppression of the non-modulated signals allowed the Fourier transform to produce a cleaner frequency spectrum.

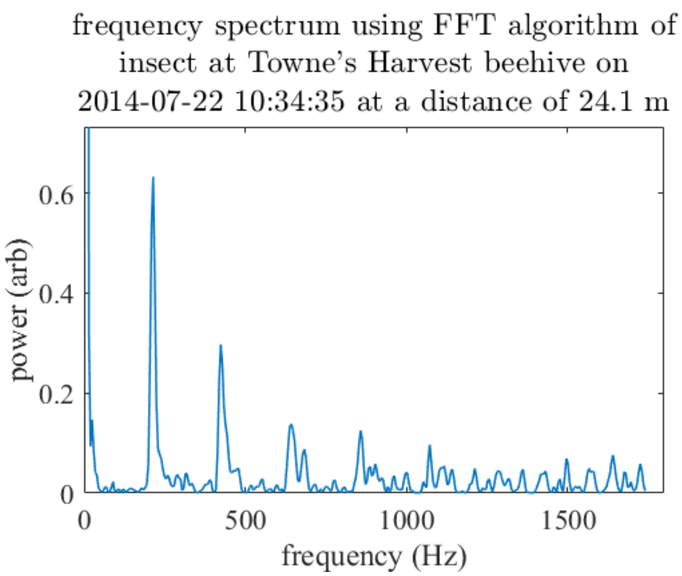


Figure 4.4: Frequency spectrum of the data plotted in Figure 4.3. The clear peak at 200 Hz defined the fundamental frequency and the harmonics are in multiples of this frequency.

detection. Naturally, the orientation of the bee with respect to to the lidar would influence the modulation depth and harmonics in the frequency domain. Even if an insect were to intercept the beam and scatter light back to the detector, it may not be distinguished as insect due to its orientation. Its wings may not have been in the path of the beam or the return from the wings may have been too small. Furthermore, even if the insect was oriented such that a strong signal was scattered back toward the detector, the insect may have been too far from the lidar system to produce a large enough SNR.

Experimentally determining the maximum range at which an insect could be detect was difficult and inconclusive. Considering that the beehive was located at the edge of a farm, there were only a few areas where a car could be parked with the lidar pointing in the proper direction. The furthest distance which we detected did not necessarily imply that that was the furthest resolvable range of the system. By parking the car about 80 m from the hive along the edge of the farm, we were able to detect and distinguish a bee. Figure 4.5 shows the signal for an insect at 85.4 m. By inspection, the SNR was much lower than that of the bee from Figure 4.2, but it was still significant.

By applying a Gaussian window (see Figure 4.6) and performing the FFT, we produced a frequency spectrum of the event. Figure 4.7 displays a fundamental peak at about 170 Hz and the intervals between the peaks were spaced by this amount as well. Although this wing-beat frequency was on the low end of the typical honeybee, it was still within the acceptable range.

During these experiments, photomultiplier tube (PMT) ringing was discovered to be a problem. The ADC terminated the signal across a $50\,\Omega$ resistor and determined the digital number based on the voltage. The gain on the PMT had to be set high enough such that the output current could convert to a readable signal. However, if

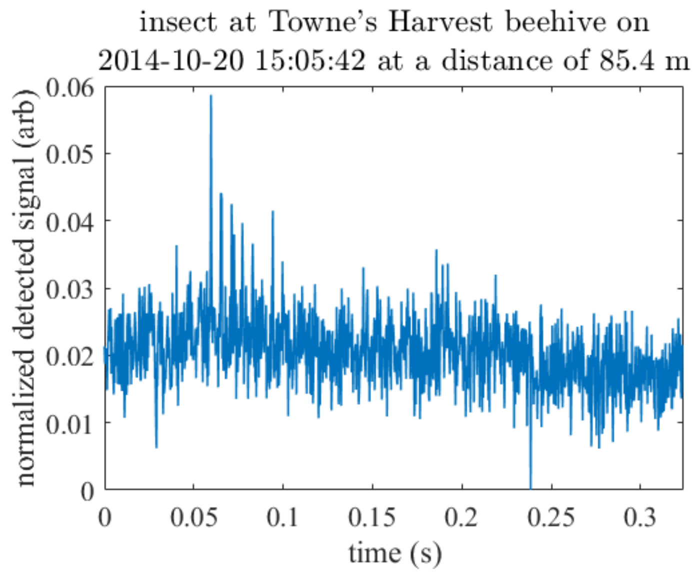


Figure 4.5: For an insect at 85.4 m, a modulated signal produced a significant SNR

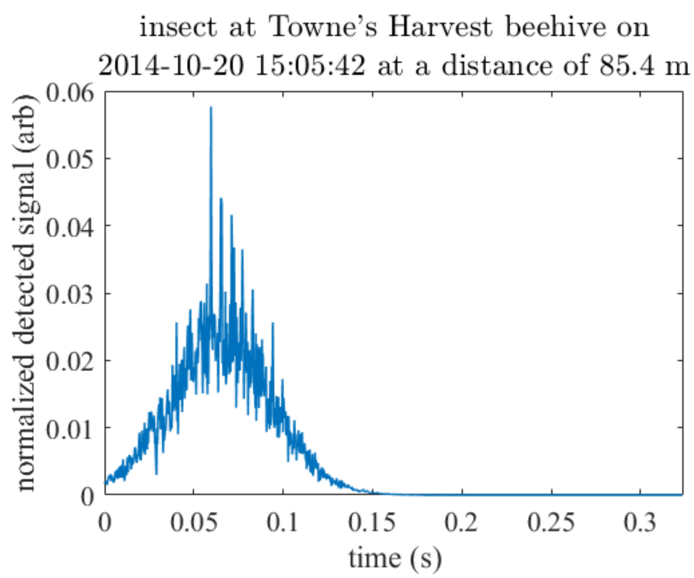


Figure 4.6: Time-series data after a Gaussian window was applied to the modulated signal from Figure 4.5.

the PMT gain was set too high, a strong reflection from a hard target could damage the ADC. On the other hand, if the PMT gain was set too low, a fair amount of ringing drastically reduced the range resolution because a phantom signal was detected in

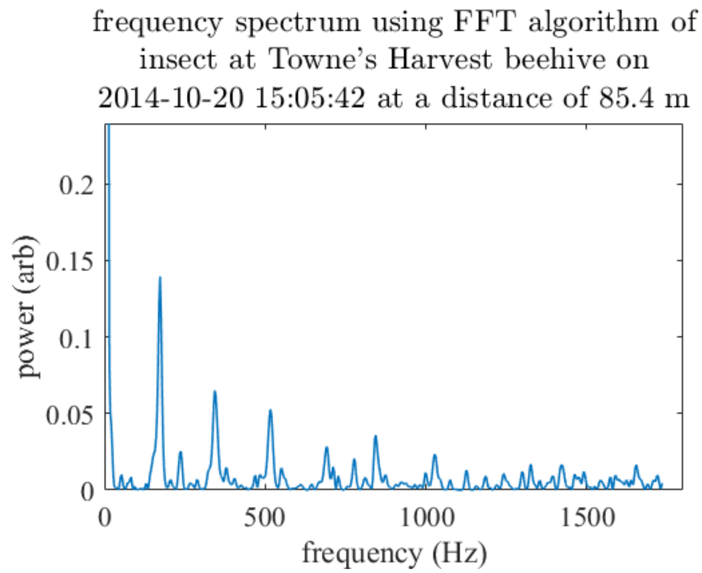


Figure 4.7: The power spectrum of the signal from Figure 4.6 showed a peak at about 170 Hz strongly suggesting the presence of a bee.

further range bins. Since this system was designed to look for very small signals but had to be able to survive comparatively strong returns, dialing in this value was important.

The results from these experiments guided the improvements made on the hardware and software, including an effort to set the proper gain on the PMT, suggesting a maximum range bin for insect detection, and building a library of events that could help improve processing algorithms. Although some results guided advances, certain improvements were not foreseen. For example, the speed at which data were saved was not a concern, and so the algorithms used for saving and storing data were not optimized. These data were saved in volts rather than digital numbers, which was slow, but this was remedied in later iterations.

Gallatin River Measurements

The results from the previous section drove modifications to the lidar system such that it could be implemented in the field. The experiment done at the Gallatin River was the intermediate step before the primary field experiments. This was the first experiment done with this lidar system that detected non-bee insects. Prior to conducting the primary experiment at Grand Teton National Park, we wanted to confirm that the instrument could indeed detect non-bee insects and that we could develop processing routines to help find the insect events in large data files.

Equipment and Setup

This experiment was performed at the Axtell Bridge fishing access site on the Gallatin River. The beam was directed across the water and terminated against the shore on the opposite side. As shown in Figure 4.8, the equipment was housed within the trunk of the car and all components were powered by an external generator. The full experiment took place on 6 October 2015 during dusk. After a certain point, the lack of sunlight made the beam very visible and I could see insects and other objects flying or falling through the beam.

Results

An example of an insect detection at 25.6 m is shown in Figure 4.9. This detection showed isolated peaks simultaneously with a DC signal. It was likely that the insect wings and body were being detected by the lidar system at the same time.

After windowing the data with a Gaussian profile (Figure 4.10) and performing an FFT, the results suggested that an insect with a 30 Hz wing-beat frequency was detected. The frequency domain (Figure 4.11) indicated that there were two different but distinct peaks. The first peak appeared to be part of the DC signal, but after

Figure 4.8: Photograph of instrument inside of car with power coming from an external generator. *Photo courtesy of Joseph Shaw*

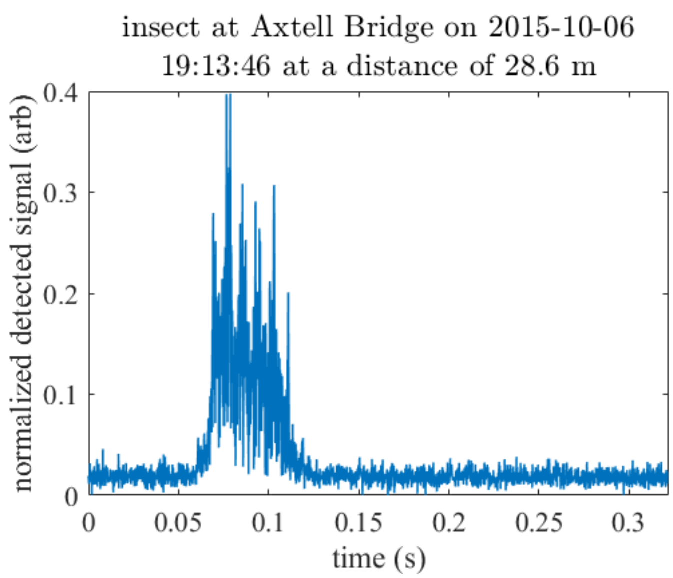


Figure 4.9: For an insect at 28.6 m, a modulated signal produced a significant SNR.

a closer look, it became clear that there was a unique peak at 30 Hz. In Figure 3.40, jitter in σ caused a low frequency artifact that appeared similar to this case. However, in the simulated figure, the artifacct was located at 0.006836 Hz and the main peak was at 0.0498 Hz, which was therefore not a harmonic ($\frac{0.0498\,\mathrm{Hz}}{0.006\,836\,\mathrm{Hz}} = 7.29$). In Figure 4.11, the second peak was at 120 Hz and slightly lower in magnitude than the first peak. The fact that 120 Hz is the fourth harmonic for a fundamental 30 Hz signal strongly implied that the signal was due to an insect with a 30 Hz wing-beat frequency, rather than an artifact from σ jitter.

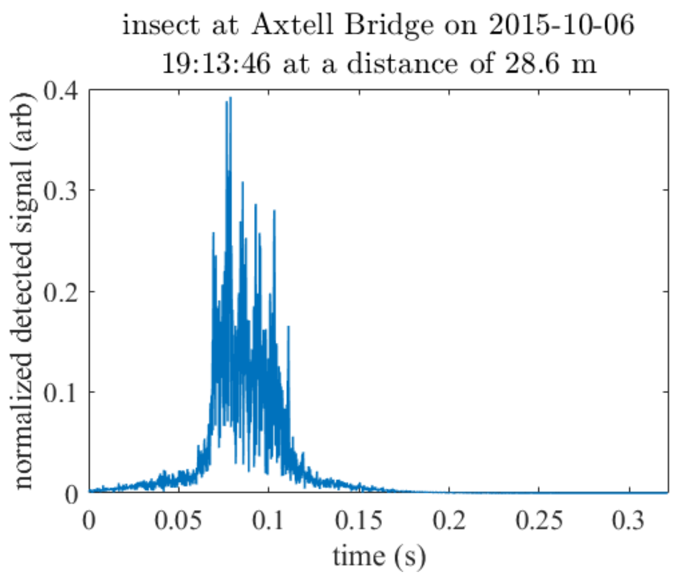


Figure 4.10: Time-series data after a Gaussian window was applied to the modulated signal from Figure 4.9.

These events validated that non-bee insects could indeed by detected by this lidar system. Additionally, it hinted at what range of wing-beat frequencies could be temporally resolved by the instrument. This in itself was a major addition to previous studies. Without the use of flourescense or black terminating cavities, the device was able to detect insect wing modulations *in situ* with good SNR.

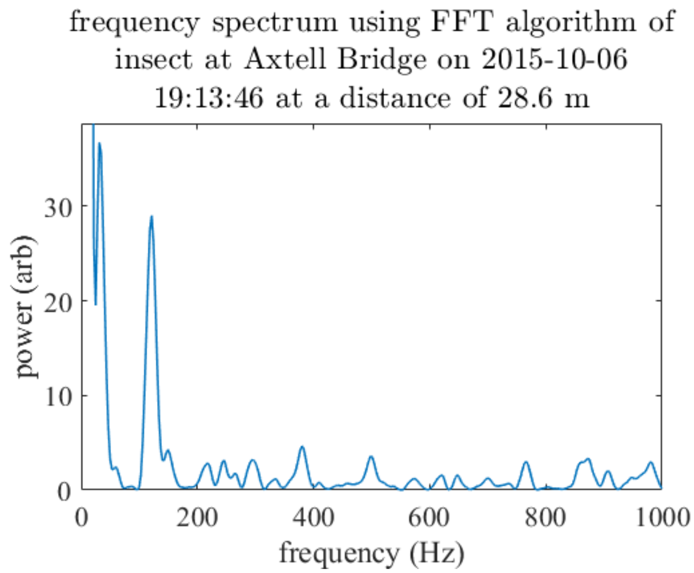


Figure 4.11: The FFT algorithm applied to Figure 4.10 shows a peak at about 30 Hz and a harmonic at 120 Hz strongly suggesting the presence of a non-bee insect.

AMK Ranch at Grand Teton National Park

Both the experiment at the Towne's Harvest Beehive and the Gallatin River were steps leading up to the field experiments at the AMK Ranch in Grand Teton National Park. The experiments conducted there were intended to collect data that could be used to infer information about insect behavior. The location provided a variety of insects and vegetation that could influence insect behavior. Additionally, the equipment was setup near an electrical outlet and so the system could run continuously without a generator.

Equipment and Setup

Over the course of four days (6-9 June 2016), the lidar scanned different regions at the AMK Ranch research station in Grand Teton National Park. On the first day, the lidar scanned closely over vegetation and terminated on a building about 100 m away. The scan lasted a few hours during the evening. On the second day, the lidar scanned a more open area and performed its operation all day with the exception of about two hours during a rain storm. The third day repeated the previous day's task with only a 9 minute break prior to the final run. Finally, the instrument collected data for a few hours in the morning on the last day. With the exception of the first day, the region scanned is pictured in Figure 4.12. The lidar system was shaded by the tarp and it scanned a volume just to the right of the cabin.

Figure 4.12: The site during the AMK Ranch experiments. The lidar was under the tarp to shade it from the sun and scanned the region to the right of the cabin.

The two middle days provided the most data due to the data collection duration. The scanning region contained shrubs and tall trees as well as bumps or inconsistencies in the terrain. Some trees were as tall as 30 m and the lidar would scan near the tops of these. During the daytime, the temperatures varied and the amount of sunlight waxed and waned. All of these factors could have influenced the behavior of flying insects in this region. By scanning the lidar through the volume, a profile of insect locations was later determined.

The largest difference in the equipment between the AMK Ranch experiments and the previous ones was the addition of the scanning mount. As pictured in Figure

4.13, the optics were all attached to the mount and it was fastened to a large optical breadboard for stability. The addition of the mount made it possible to point the lidar in any desired location (with physical limits in the tilt). This ability allowed volumes to be scanned over time and thus compare insect responses to a variety of variables in the scan region.

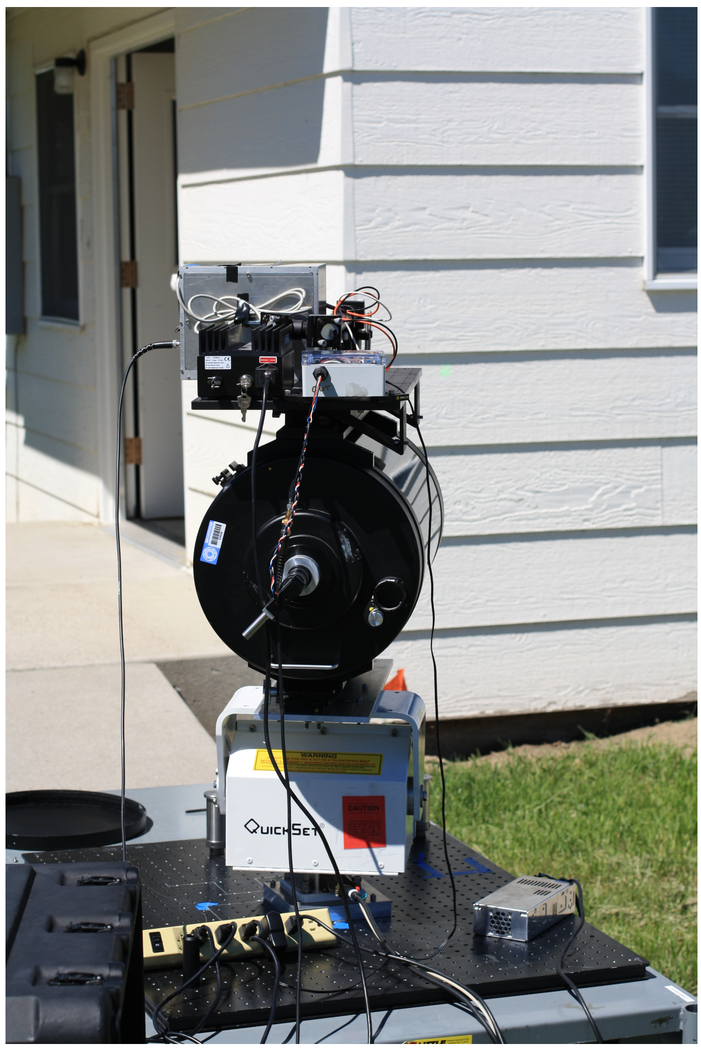


Figure 4.13: Photograph of instrument prior to departure to AMK Ranch. The transmitting optics sat atop the receiving telescope and all of that was attached to a scanning mount.

The other major improvement to this system was the optimization of data collection and processing algorithms. The improved software collected more data quickly and stored it in manner that was easier to process.

Results

Much like the results form the beehive and the Gallatin river experiments, insects were detected at the AMK Ranch. Due to the lengthy data collection periods, numerous interesting signals were detected. Some insects had very high wing-beat frequencies, and others were very low. Some frequency spectra provided many strong harmonics and others simply had a single distinguishable peak.

On 8 June 2016, one of many insects was detected and the time domain for this event is shown in Figure 4.14. Four peaks with a substantial SNR could be seen between 0.15 and 0.2 s. A Gaussian window (Figure 4.15) and then a FFT algorithm were applied to these data and the results are plotted in Figure 4.16. Unlike other signals we had seen before, the harmonics present for this particular insect were very well defined.

An important feature about the frequency information of this insect was the fact that the fundamental frequency was not the largest peak. The first significant peak occurred at 100 Hz; however, the largest peak was at 193 Hz. To determine the fundamental frequency when multiple harmonics were present, we used the spacing between harmonics. In this instance, the mean spacing between peaks was 96.8 Hz and the standard deviation of the spacing was 4 Hz. Thus we concluded that this insect had a wing-beat modulation of 96.8 Hz.

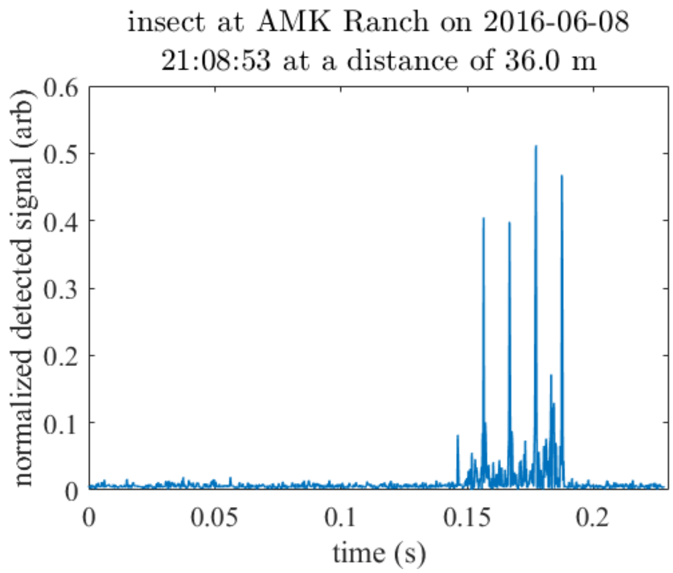


Figure 4.14: For an insect at 36 m, the modulated signal produced a significant SNR.

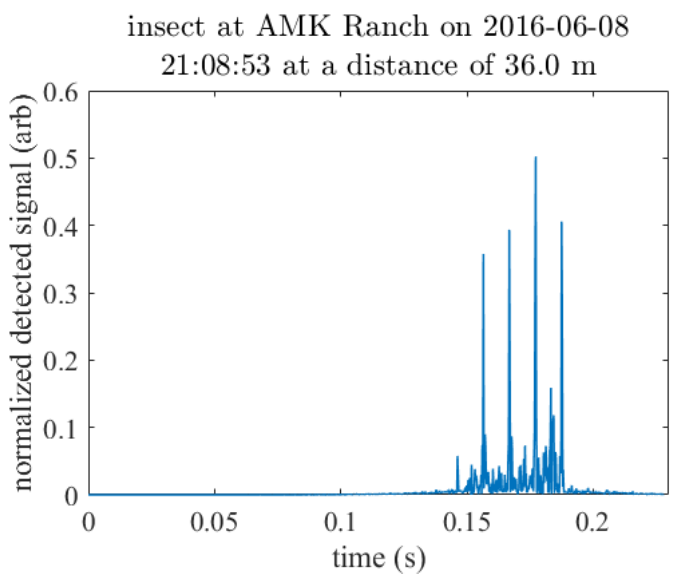


Figure 4.15: Time-series data after a Gaussian window was applied to the modulated signal from Figure 4.14.

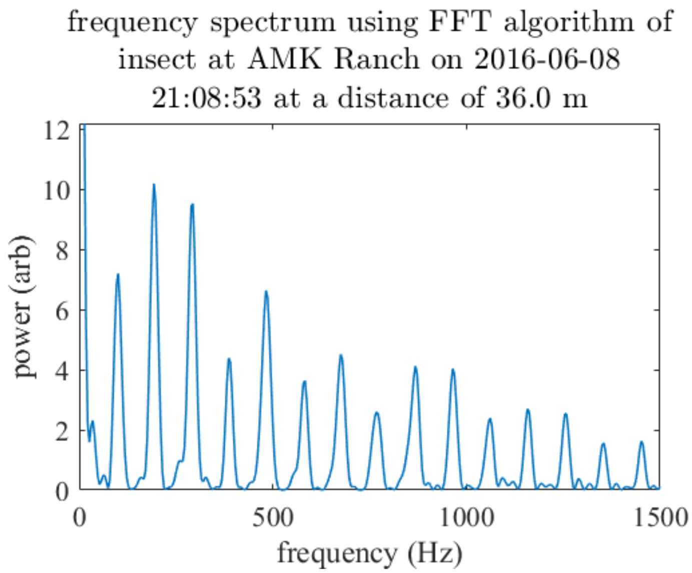


Figure 4.16: The FFT algorithm applied to Figure 4.15. Since a harmonic had a larger peak than the fundamental, the fundamental frequency was determined by finding the difference between each harmonic, which was 96.8 Hz for this insect.

Chapter Summary

Over the course of this project, three main experiments were conducted. First, beehive measurements were taken because of the close proximity of the research site and the fact that previous studies successfully conducted such experiments. Many tests were done at this site as the instrument improved. Eventually, the lidar system was tested at a fishing access site on the Gallatin River. The results showed that non-bee insects could indeed be detected and distinguished from obstructions. Finally, the main field experiment was conducted at the AMK Ranch. The insect lidar system scanned a variety of regions over the course of many days. The variety of insect locations detected in those experiments allowed us to build a three-dimensional spatial plot.

The ability to locate various insects in three dimensions was the ultimate goal of this instrument. Such a device had not been implemented before for general insect studies and the promising results from the experiments suggested that such a device could be a very useful tool for the entomological community. The next chapter analyzes the results to further instill the capabilities of the instrument.

DISCUSSION

The health of any ecosystem depends on the state of the living organisms within it. Biologists of various sub-disciples monitor life within different environments. Within the field of entomology, researchers are interested in the behaviors of insects and how they interact with their natural environment and with external variables. The complexity of environments, the size of insects, and the abundance of obstructions and inconsistencies in the natural world make it difficult to find insects. In order to detect insects' spatial locations without disrupting them, a special tool is required. We have developed a scanning lidar system for the purpose of detecting insects within their natural habitat. Rural or undeveloped regions around the world are becoming populated and the presence of people creates variables such as noise and light pollution in the surrounding areas, potentially affecting insects and thus other life as well. With an instrument capable of measuring insect spatial distributions, we can determine how insects respond to certain variables such as noise and light sources.

In this chapter, we describe the usefulness of the lidar system and its ability to detect insects. Additionally, we describe the various ecological impacts that could be measured with the scanning lidar system. The data presented in this section are from 8 June 2016 at the AMK Ranch research station within Grand Teton National Park. Although more data were collected during these primary field experiments at the AMK Ranch, the time-consuming process of manually classifying insects restricted us to only analyzing a single day. This indicates the potential utility of automated algorithms for detecting and classifying insects from wing-beat modulation lidar systems, which has not been done in this work.

Instrument Performance

To draw conclusions about insect behavior, the lidar system needed to detect backscattered light from insect wings, differentiate insects from obstructions, and determine insect spatial and temporal location based on the mount orientation. The combination of hardware and software were able to accomplish each of these goals, with the exception of an automated algorithm for insect classification.

Ability to Detect Insect Wings

The most basic function of the insect lidar instrument involved detecting light scattered from an insect wing. This concept had been used in many previous studies [12, 13, 32], but the ability to detect insects of various species *in situ* with no support device - such as a black terminating cavity or fluorescence - and with the ability to scan a region instead of a single point not only made the system novel, but would also provide researchers with the ability to answer questions about insects' spatial distributions.

Perhaps the most challenging specification of the instrument was the ability to detect insects out to 100 m. The furthest detected insect was at 87.6 m. Figure 5.1 shows the time series of this particular insect detection. The mean signal-to-noise ratio ($mSNR$) of the the three peaks, defined in Equation 5.1 where $\bar{A}_{peaks}$ is the mean of the peak values, and σ_{noise} is the standard deviation of the noise floor, was 23.8. The peak SNR ($pSNR$) of the largest peak only (i.e., not the average of all the peaks), defined by Equation 5.2, was 37.4. These SNR values were fairly high considering that an insect at 41.3 m produced a smaller $mSNR$ of 14.5 and $pSNR$ of 16.9. This insect is represented in Figure 5.2. The $pSNR$ of an insect's wing-beat frequency at a range 87.6 m suggested that an insect could be detected at even further range bins.

$$mSNR = \frac{\bar{A}_{peaks}}{\sigma_{noise}} \tag{5.1}$$

$$pSNR = \frac{A_{peak}}{\sigma_{noise}} \tag{5.2}$$

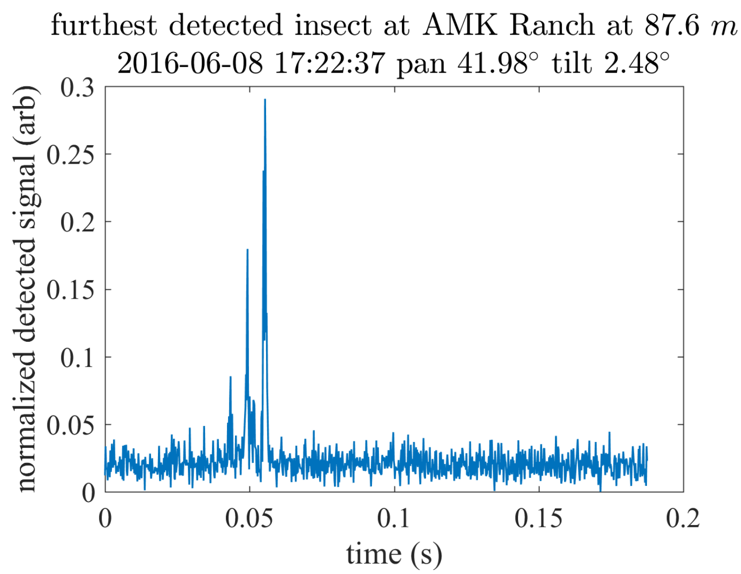


Figure 5.1: The furthest insect detected at the AMK Ranch on 2016-06-08 still had a significant $pSNR$.

Inspecting the time series of events, alone, could not provide enough information to determine if the event was an insect or not, nor could it determine the fundamental frequency and higher order harmonics. By inspecting the frequency spectrum, we could analyze the wing-beat frequencies and make conclusions about the insect based on the fundamental and harmonics. Other research suggested that knowing the fundamental and harmonics can imply insect orientation [27]. Furthermore, it could help lead toward classifying insect species, age, and sex. The frequency spectra

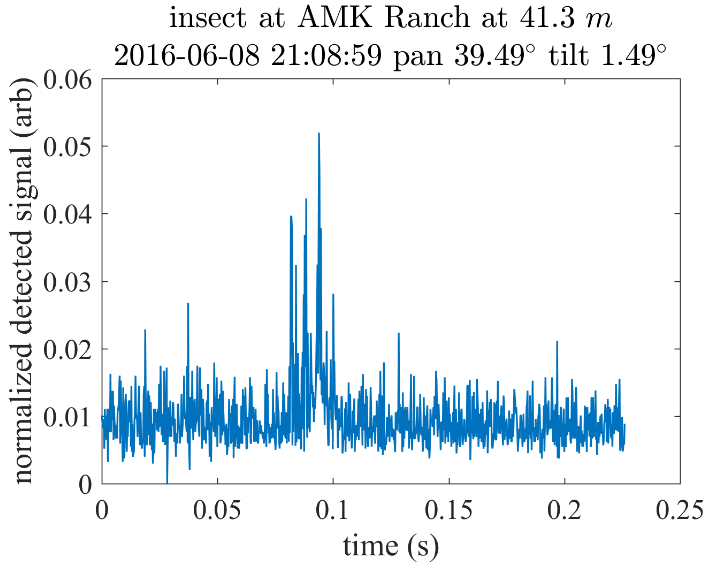


Figure 5.2: An insect detected at the AMK Ranch on 2016-06-08 that was much closer than the furthest detected insect and had a smaller $pSNR$.

corresponding to Figures 5.1 and 5.2 (after a Gaussian window was applied) are presented in Figures 5.3 and 5.4, respectively. Indeed, the frequency signals are clear enough such that we were able to decide what the fundamental frequencies were.

The inspections of time- and frequency-domain plots ultimately led to the final decisions of which events were insects. These decisions were not always trivial because the plots were often ambiguous, making this process one of the largest challenges. The lack of a "truth" dataset, that is, one where we knew whether the event was an insect or not based on a different detection method, forced us to create a set of criteria that we used to classify the insects. The first clue to an insect event was in the time series. If some event had obvious peaks and troughs, it was almost certainly an insect. After performing FFTs on such events, if a large peak was present in the frequency domain, it would support our initial assumption. Occasionally, the frequency domain would have multiple well-defined peaks that were evenly spaced. These events were the simplest to classify as insects because the consistent spacing

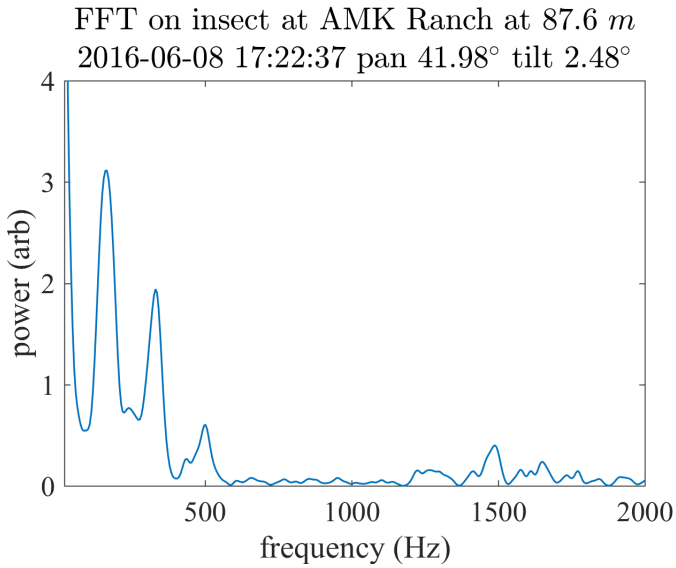


Figure 5.3: The FFT that corresponds to the time series data from Figure 5.1. The two largest peaks (not including the DC term) are at 160 Hz and 336.1 Hz, which strongly suggested an insect event.

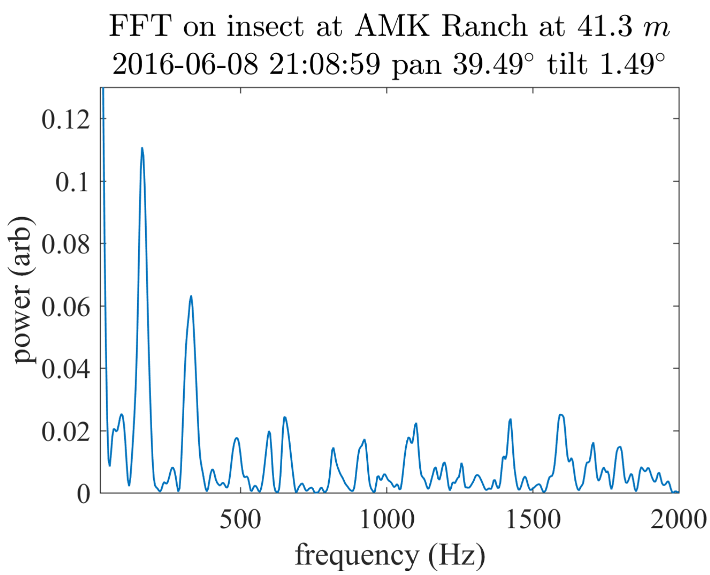


Figure 5.4: The FFT that corresponds to the time series data from Figure 5.2. The two largest peaks (not including the DC term) are at 163.7 Hz and 331.8 Hz, and not as well defined as those in Figure 5.3.

Table 5.1: Number of events for each classification group. The "unlikely" insect events were approximated because repeat detections were not deleted.

classification group	total number of events
very likely	76
somewhat likely	662
unlikely	≈ 2000

of harmonics was an obvious sign. On the other hand, the most challenging cases occurred when a time series appeared to show a modulated signal but the peaks in the frequency domain were not that well defined. This led to a classification system where events were grouped into three categories, "very likely," "somewhat likely," and "unlikely." "Very likely" insects were events which had large or multiple peaks in the frequency domain and clear, evenly spaced peaks in the time domain. They also underwent an extra processing step to determine the fundamental frequency. The "somewhat likely" insects had ambiguous time- and frequency-domain signals, but still appeared to be insects (more detail below). Furthermore, these did not go through the extra processing step and so the determined fundamental frequency could have been incorrect. Nonetheless, both "very likely" and "somewhat likely" cases were analyzed since the total number of "very likely" insects was low whereas the "somewhat likely" number of cases was an order of magnitude greater (Table 5.1).

The difficulty in classifying "very likely" cases from "somewhat likely" (or possible insect) events was due to the imperfections in time- and frequency-domain signals. Figure 5.5 shows the time series of a possible insect event and the corresponding frequency spectrum is plotted in Figure 5.6. Although a peak frequency was at 75 Hz, the other frequencies were not very suppressed. Perhaps the peak frequency truly represented the fundamental wing-beat frequency of the event, but the presence of the other well-defined, smaller peaks did not make this an obvious classification. Another example, presented in Figures 5.7 and 5.8, showed a very noisy signal with a peak frequency out to nearly 3 kHz which was almost certainly an artifact because the literature suggests that fundamental wing-beat frequencies have not been detected over about 1 kHz (to our knowledge). For these reasons, these types of events were classified as "somewhat likely" to be insects and were analyzed separately from the "very likely" cases.

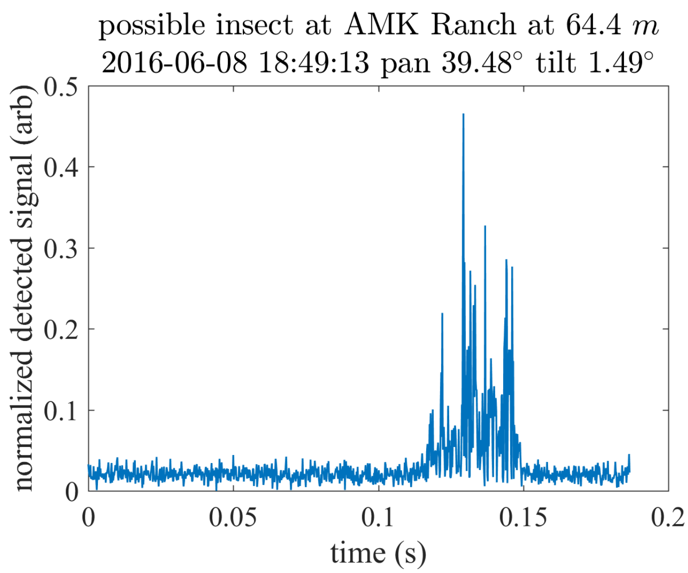


Figure 5.5:]

Time series of an event that was possibly an insect.

Small details between "very likely" and "somewhat likely" insect events made the classification process lengthy and difficult. Although initial attempts were made to develop an algorithm to sort each event accordingly, we did not yet have enough information to construct a functioning script. A classification algorithm would have to make a decision based on certain parameters with set thresholds. The decision of which parameters to use in order to make the ultimate classification was nearly impossible because we did not know which parameters would allow for separability. For example, classifying an insect was not as simple as choosing the largest frequency

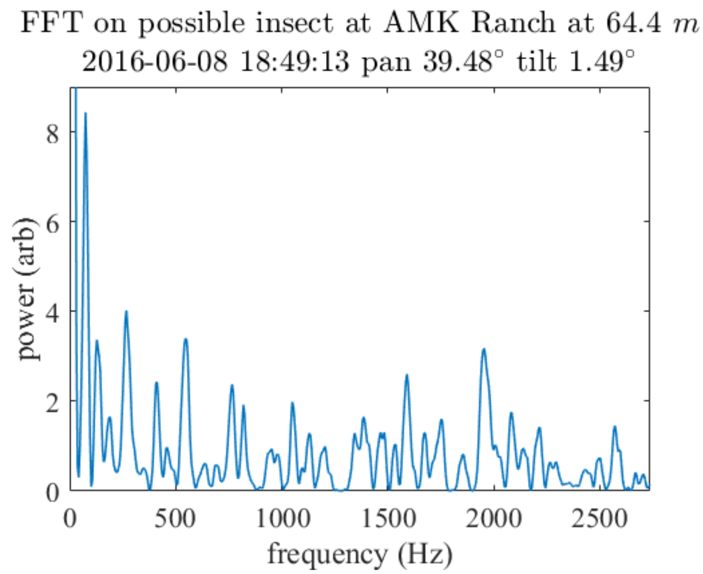


Figure 5.6:]

Frequency spectrum of the signature of a possible insect event.

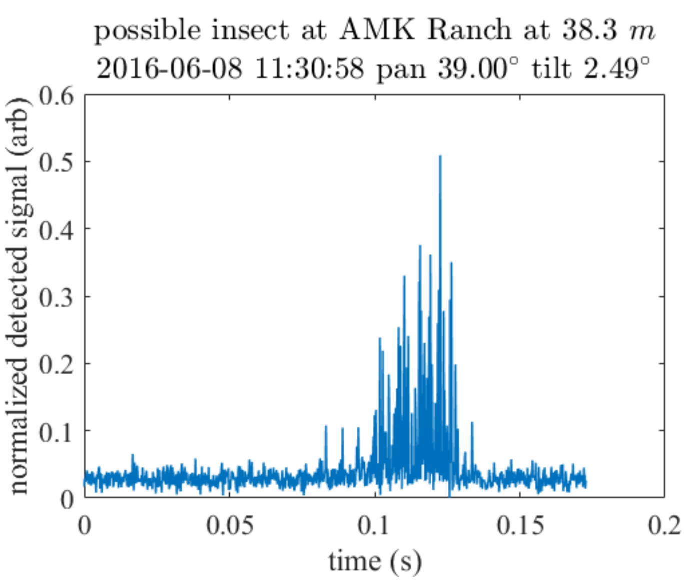


Figure 5.7: Time series of an event that was possibly an insect.

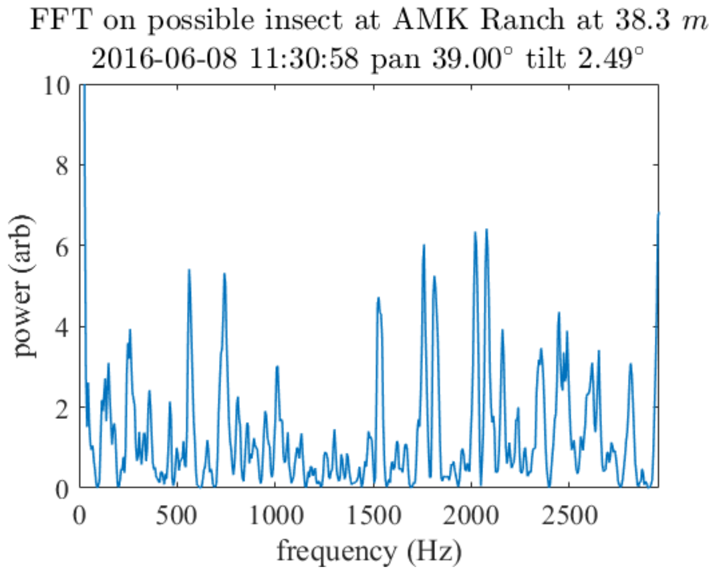


Figure 5.8: Frequency spectrum of the signature of a possible insect event.

peak because often it was not the fundamental, and classifying based on relative peak size led to misclassifications. The algorithm had to be written with multiple different parameters and we had not determined the proper ones to make the best decisions. For these reasons, we manually classified insects so that we would have a data set to analyze, and so we could use those cases to build a training set for a future algorithm. With a known set of insect events and their corresponding time and frequency information, we could determine the best parameters to use and construct an efficient algorithm.

The scanning mount allowed us to pinpoint insect locations. An insect's spatial position was determined by the range (r), pan (p), and tilt (t) values. In order to plot an insect on an $x-y-z$ Cartesian plot, the values had to be converted. The z (or vertical) component was determined using simple trigonometry ($z = r\ sin(t)$). The x and y components depended on the range, tilt and pan angles. In this case, the projection of range (er) onto the $x-y$ plane was first calculated ($er = r\ cos(t)$). The projected range was then used to determine the final components ($y = er\ sin(p)$

and $x = er\ cos(p)$). The first step of this process required the range of the insect, and since the range resolution of the lidar system was 0.75 m, our spatial accuracy in x,y, and z was also reduced. Considering the uncertainty in range, pan, and tilt angles, the maximum error was calculated as a volume element of a sphere. The maximum error occurred when the tilt was at 1.5°. Equation 5.3 (note that $\phi = 0$ at the positive vertical axis) shows the calculation to determine the volume of error. These errors essentially expand a small insect into a volume just larger than 75^3 cm^3, which is considered large in the lidar community, but small in the entomological one.

$$\begin{aligned} \int_{\phi_1}^{\phi_2}\int_{\theta_1}^{\theta_2}\int_{r_1}^{r_2} sin(\phi)\rho^2\ d\rho d\theta d\phi &= \\ \int_{39.755^\circ}^{40.245^\circ}\int_{88.255^\circ}^{88.745^\circ}\int_{100\ m}^{100.75\ m} sin(\phi)\rho^2\ d\rho d\theta d\phi &= 0.55\ m^3 \\ &= 550,000\ cm^3 \end{aligned} \tag{5.3}$$

For entomological studies, the insect lidar system had to be precise and accurate such that insect locations, time domain signals, and the frequency information could produce interpretable results. Some small errors existed in spatial resolution and some ambiguity was present in the insect classification process; to reduce these errors a faster ADC with higher bit-depth and a faster laser repetition rate could be implemented. Since the experiments scanned a region of several hundred cubic meters, the spacial accuracy was fine enough. Furthermore, since only the most obvious insect events were analyzed for the "very likely" cases (that is, ones with well defined fundamental frequencies or evenly spaced harmonics), the frequencies were very likely to be accurate; however for the "somewhat likely" cases, these were not analyzed and should thus be treated with skepticism.

Entomological Impacts

Monitoring how insects respond to natural and human-caused variables with an effective tool can improve studies within the entomological community. The lidar system could help researchers determine what variables affect their spatial distribution, abundance in the atmosphere, and other factors. Additionally, different sexes, ages, and species likely respond differently to each variable and with the knowledge of an insect's fundamental frequency and higher-order harmonics some of those qualities could potentially be inferred in future iterations of the lidar system. Below are some examples of how the data could be interpreted for entomological studies.

Time Distributions

Although the main advantage of this insect lidar over other systems was its ability to determine insects' spatial distributions (discussed below), other information could also be extracted. For example, every insect detection was time-stamped. In Figure 5.9 the number of insects was plotted against time in addition to the temperature, relative humidity, and wind speed (from a weather station a few miles away) during the day of 8 June 2016. Assuming insect counting was a Poisson process (since the system was counting discrete random events), the standard deviation for each bin was determined by $\sqrt{N}$, where N was the total number of events in that bin. In this case, there was a correlation between the number of insects during the day and the temperature and humidity. During the hot parts of the day, the insects were not present in the atmosphere, but during the cooler (and not yet cold) times, they were higher in number. It has been shown that some insects tend to be most active during dawn and dusk, and that wind and humidity can greatly affect their behavior [54].

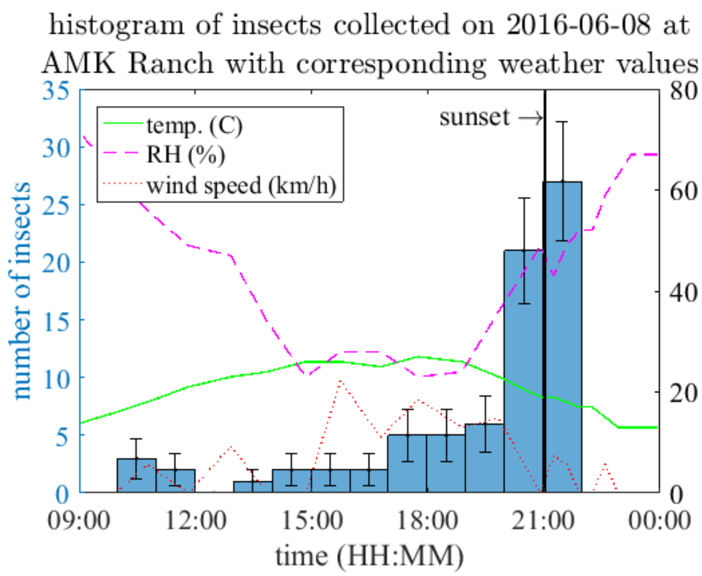


Figure 5.9: Histogram of insect detections in time correlated with temperature, relative humidity, and wind speed of that day. The left axis represents the number of insects and the right axis corresponds to temperature, humidity, and wind speed. Data collection began at 9:09 and ended at 21:53.

In Figure 5.10, the possible insect events were plotted in the same fashion as Figure 5.9 in order to help determine the validity of the “somewhat likely” events. The fact that the two plots showed the same trend led to the conclusion that the “somewhat likely” events are mostly insects.

Spatial Distributions

Determining the spatial distribution of flying insects was the primary goal of the insect lidar system. Figure 5.11 illustrates the the locations of (“very likely”) flying insects (blue dots) within the scanning region (green volume). Figure 5.12 shows the locations of possible insects (“somewhat likely”). The scanned region was a fairly open space with trees in the distance and few trees and bushes scattered throughout.

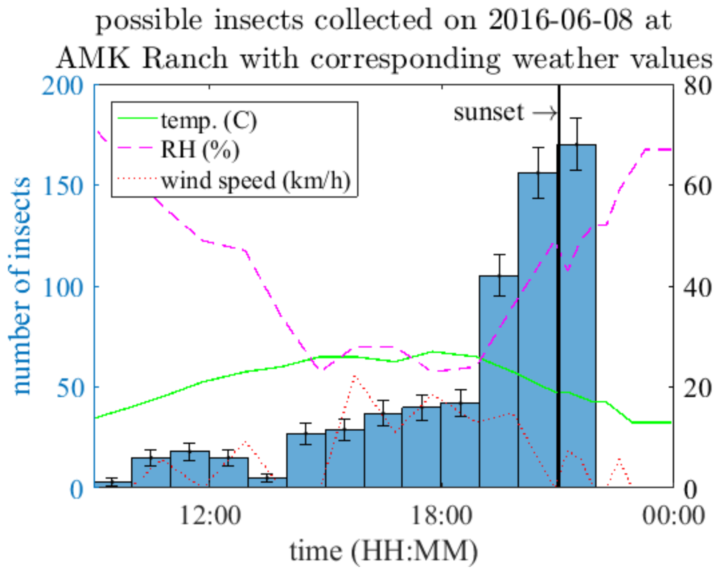


Figure 5.10: Histogram of possible insects. Temperature, relative humidity, and wind speed of that day were plotted for comparison. The left axis represents the number of insects and the right axis corresponds to temperature, humidity, and wind speed. Data collection began at 9:09 and ended at 21:53.

An inspection of the Figure 5.11 showed that many insects were detected near the bottom of the scan region, suggesting that insects tend to stay close to vegetation. Insects do stay near plants when foraging [24] and so seeing so many insects so low made sense, and a histogram of "very likely" insects in each tilt angle (Figure 5.13) corroborated this theory. However, the possible-insect events in Figure 5.12 did not appear to indicate that insects preferred to fly at low altitudes and the histogram of possible insects as a function of tilt angle (Figure 5.14) did not indicate a similar pattern as with the "very likely" insect case. In fact, it appeared to suggest that insects may prefer higher altitudes rather than low ones. We concluded that the "very likely" data set may have too few events to draw significant conclusions, and the range of errors was relatively high.

Similar spatial distribution plots could shed light on different insect patterns. Figures 5.15 and 5.16 show a spatial distribution with a distinction in wing-beat

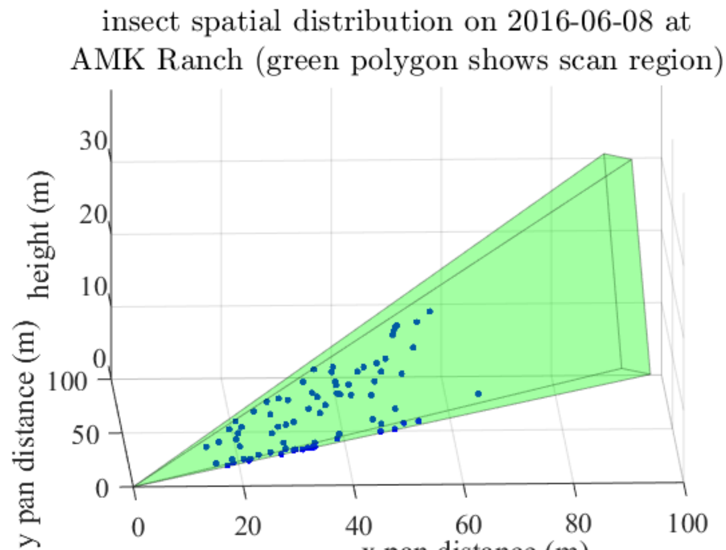


Figure 5.11: Full spatial distribution of flying insects. Blue dots represent the insects and the green volume represents the scanning region.

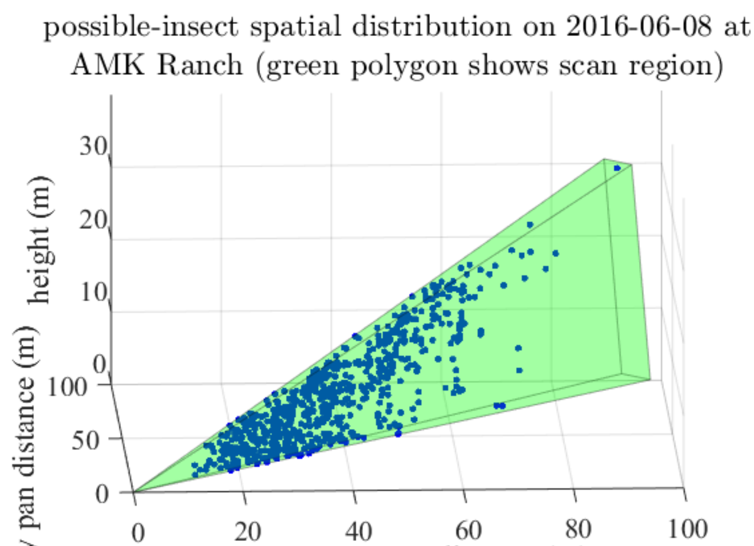


Figure 5.12: Possible-insect full spatial distribution. Blue dots represent the insects and the green volume represents the scanning region.

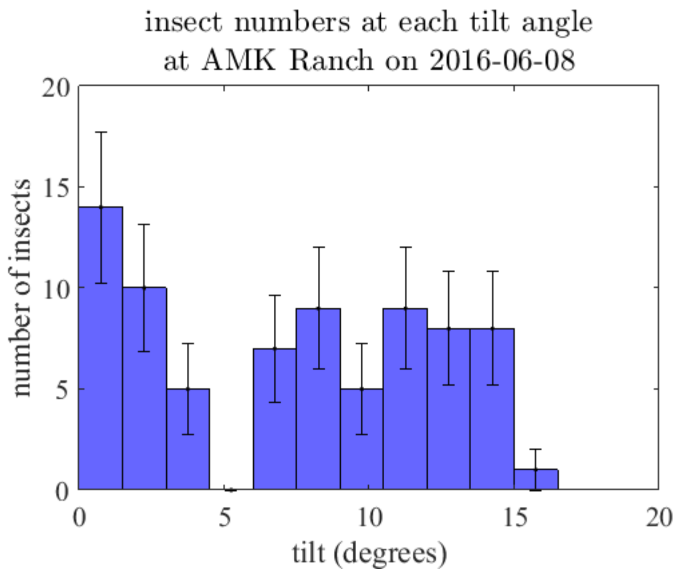


Figure 5.13: Histogram of insects as a function of tilt angle. The lowest tilt bin counted the most insects, which suggested that insects may prefer to stay close to vegetation.

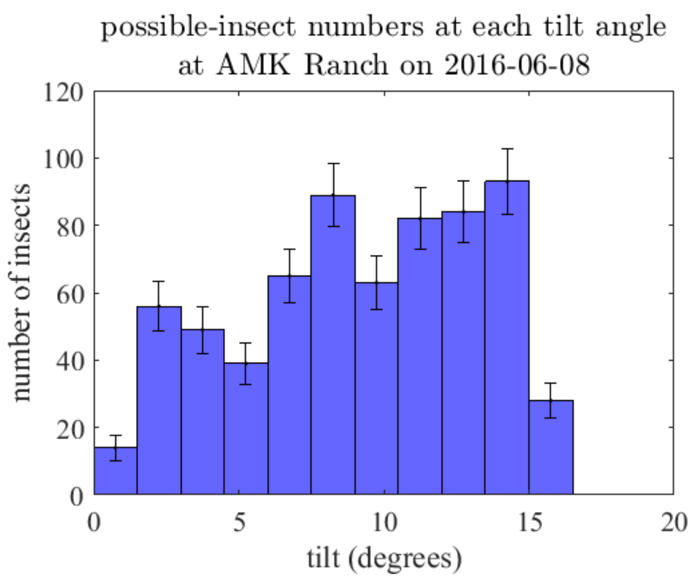


Figure 5.14: Histogram of possible insects as a function of tilt angle. The lowest tilt bin did not count the most insects, in fact it counted the least.

frequencies, for "very likely" and "somewhat likely" insects, respectively. The red diamonds represent insects with wing-beat frequencies over 300 Hz and the blue dots represent insects with wing-beat frequencies under 300 Hz. Similarly, Figures 5.17 and 5.18 show a spatial distribution for "very likely" and "somewhat likely" events, respectively, where red diamonds represent insects detected prior to 19:00 Mountain Daylight Time and where blue dots represent insects which were detected after that. These types of plots provide the information which the entomological community could use to advance their studies.

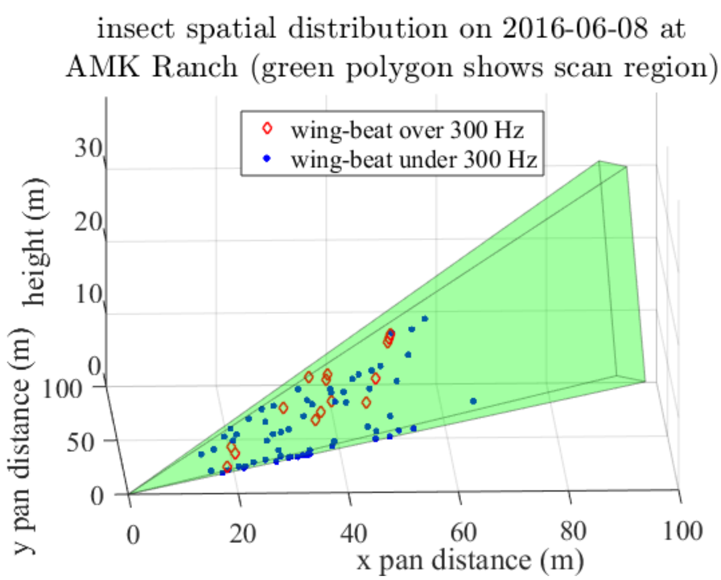


Figure 5.15: Spatial distribution of insects with low and high frequencies marked differently.

A careful inspection of these figures revealed that there may have been some correlation between time of day and tilt angle. In Figure 5.17, the insects detected before 19:00 tended to stay lower in the atmosphere. Figures 5.21 and 5.22 were histograms for "very likely" and "somewhat likely" events, respectively, where time of detection was separated. In Figure 5.21, the insects before 19:00 were mostly below a tilt of 3°. In Figure 5.22 insects after 19:00 were more abundant in higher

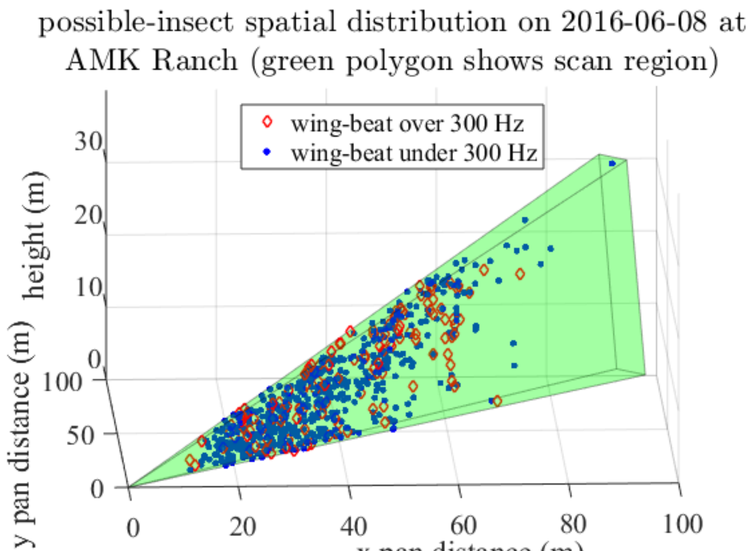


Figure 5.16: Possible-insect spatial distribution with low and high frequencies marked differently. Note that some frequencies for “somewhat likely” insects may be incorrect due to the lack of a processing step.

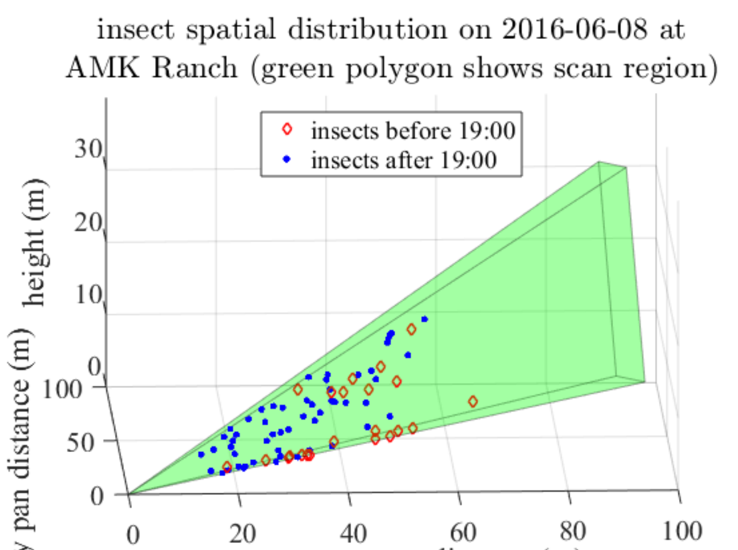


Figure 5.17: Spatial distribution of insects with early and late events marked differently.

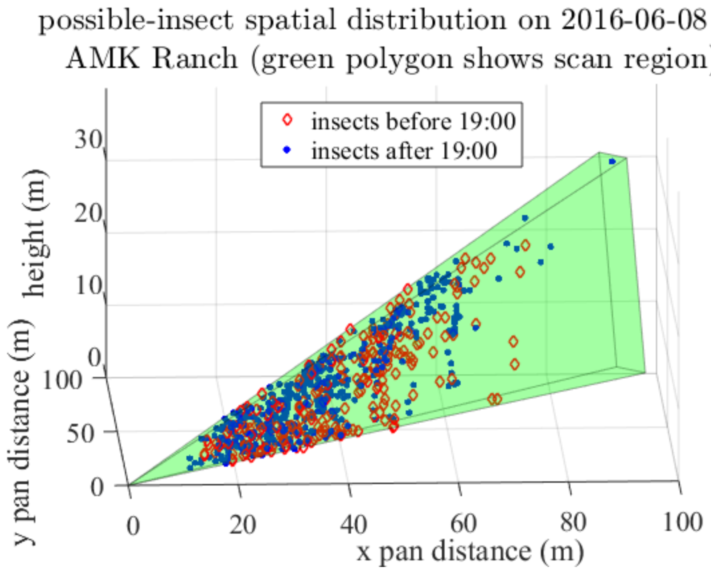


Figure 5.18: Possible-insect spatial distribution of insects with early and late events marked differently.

tilt angles. These two figures together appeared to show that insects fly at higher altitudes later in the day. Figures 5.15 and 5.16 showed that most insects with wing-beat frequencies over 300 Hz were rare in the low tilt angles. The corresponding histograms in Figures 5.19 and 5.20 also showed that insects with high wing-beat frequencies preferred higher elevations. However, the fact that the frequency values for the "somewhat likely" cases were not properly processed, the results were (and should be) treated with some skepticism.

Signal Analysis

In addition to spatial and temporal distributions, the time and frequency signals could provide useful information as well. The modulation depth or fundamental and harmonic frequencies could help indicate which type of insect was detected. Figure 5.23 shows a spectrogram of ("very likely") insect wing-beat frequencies with the fundamental frequency circled in red for each insect. The insects were ordered by

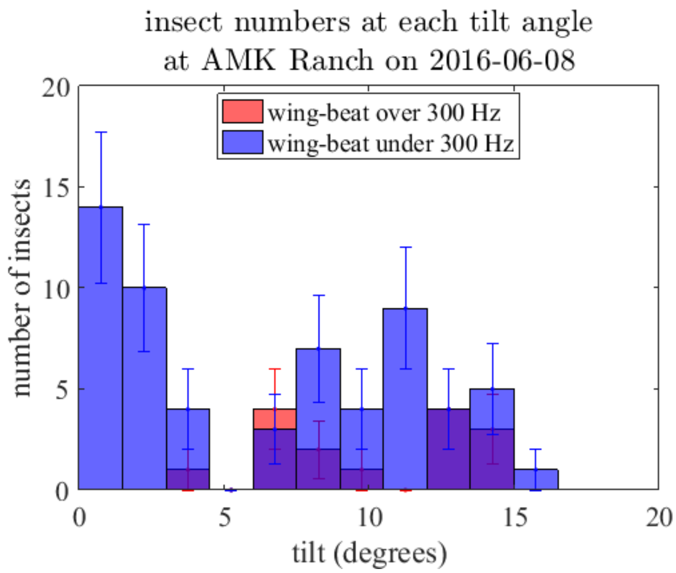


Figure 5.19: Histogram of insects as a function of tilt angle where insects were separated by wing-beat frequency. Insects with wing-beat frequencies above 300 Hz were not detected below a tilt angle of 3°.

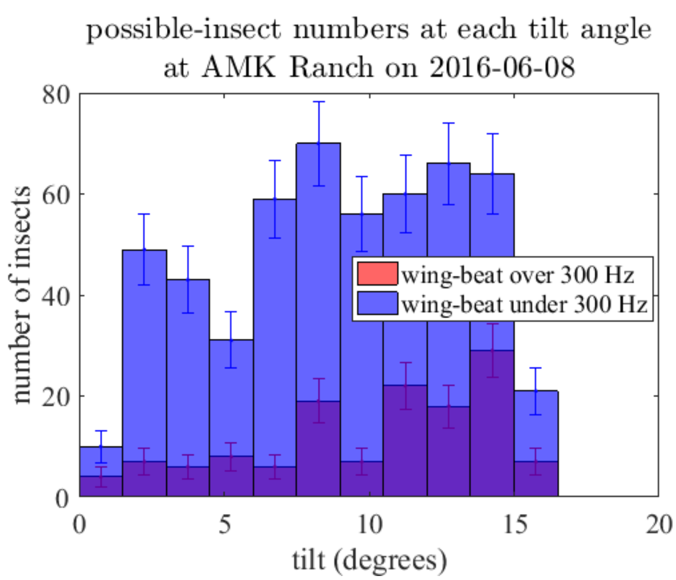


Figure 5.20: Histogram of possible insects as a function of tilt angle where insects were separated by wing-beat frequency. Note that some frequencies for “somewhat likely” insects may be incorrect due to the lack of a processing step.

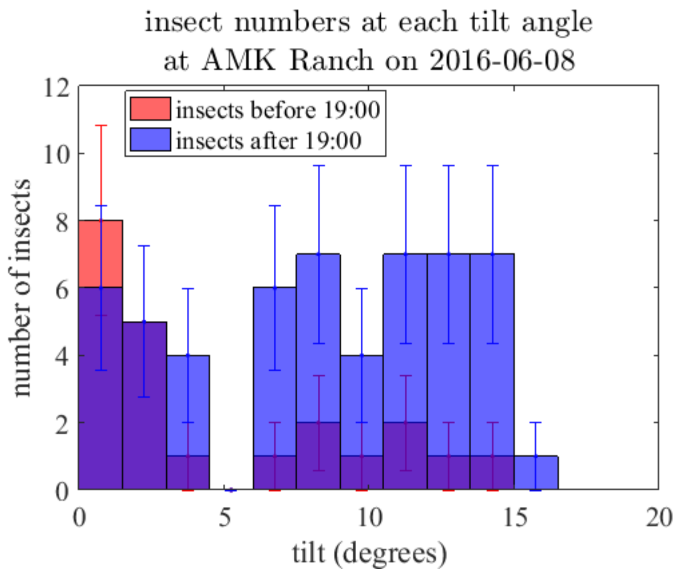


Figure 5.21: Histogram of insects as a function of tilt angle where insects were separated by time of detection. Insects detected before 19:00 were mostly seen in lower tilt bins whereas insects detect later had a light preference to higher altitudes.

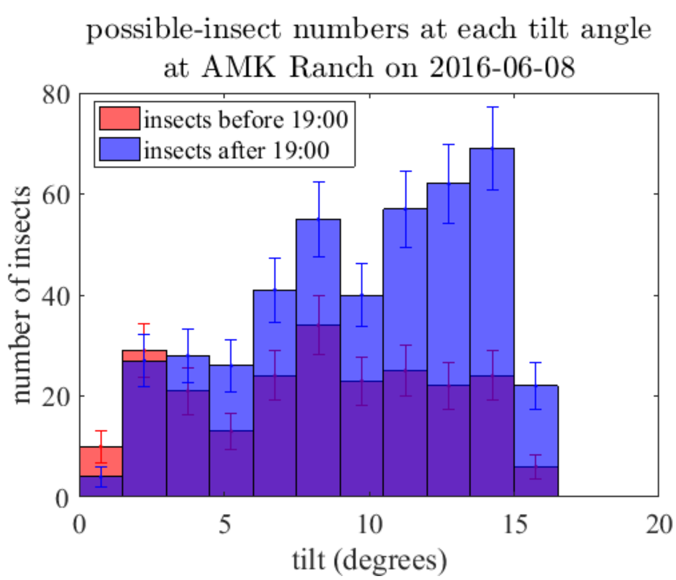


Figure 5.22: Histogram of possible insects as a function of tilt angle where insects were separated by time of detection. Insects detected after 19:00 were more abundant in the higher altitudes, which corroborates the trend for the “very likely” cases.

their fundamental frequency, whereas Figure 5.25 plotted insects in chronological order. From Figure 5.23, it is clear that most insects had a wing-beat frequency below 300 Hz. Figure 5.26 is a histogram of insect wing-beat frequencies from that day. Indeed, most frequencies occurred at the lower end of the spectrum with only a handful of faster frequencies. The "somewhat likely" insects, displayed in Figures 5.24 and 5.27, showed the same type of trend: most insects had wing-beat frequencies below 300 Hz.

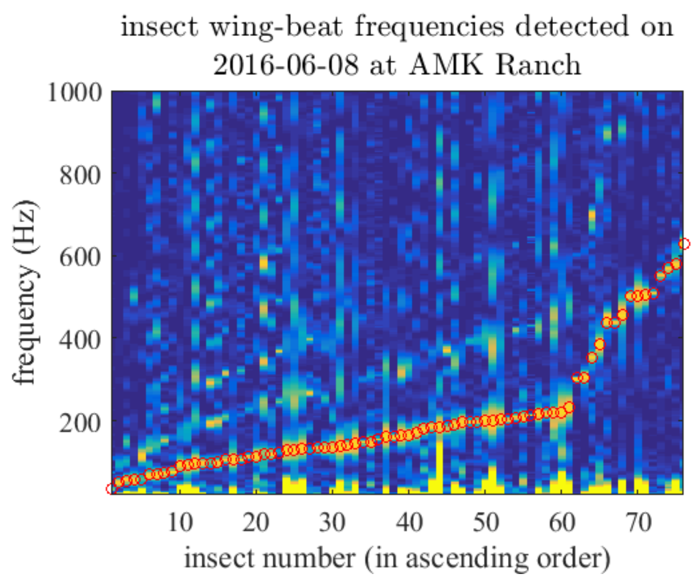


Figure 5.23: Spectrogram of insect wing-beat frequencies with the red circle representing the fundamental frequency. The insects are ordered by their fundamental wing-beat frequency. Most insects have wing-beat frequencies below 300 Hz but a few have frequencies between 300 and 700 Hz.

Similar plots and interpretations about insect characteristics could be made with plots of modulation depth, optical cross section (with a calibrated system), or duration of flight within the beam. These were just some examples of the potentially many ways that etymologically useful information could be extracted from the scanning insect lidar system.

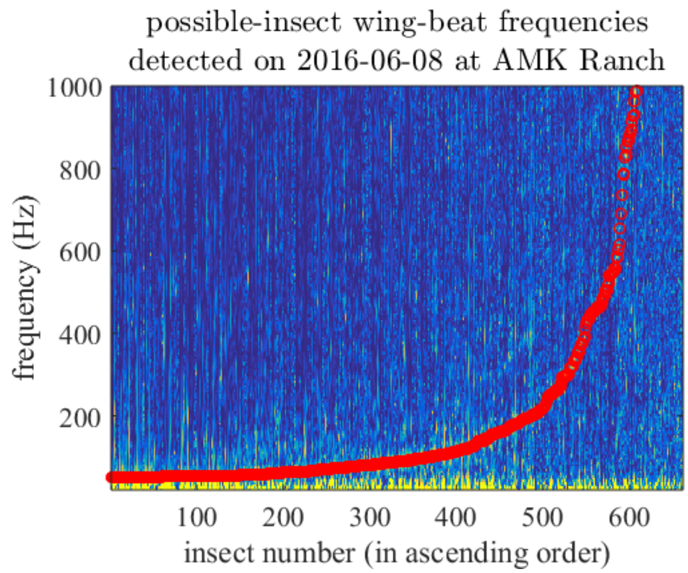


Figure 5.24: Spectrogram of possible insects with wing-beat frequencies with the red circle representing the fundamental frequency (note that some "somewhat likely" insect may have incorrect values due to the lack of a processing step). The insects are in order of fundamental frequency. Much like case of "very likely" insects, most have wing-beat frequencies below 300 Hz.

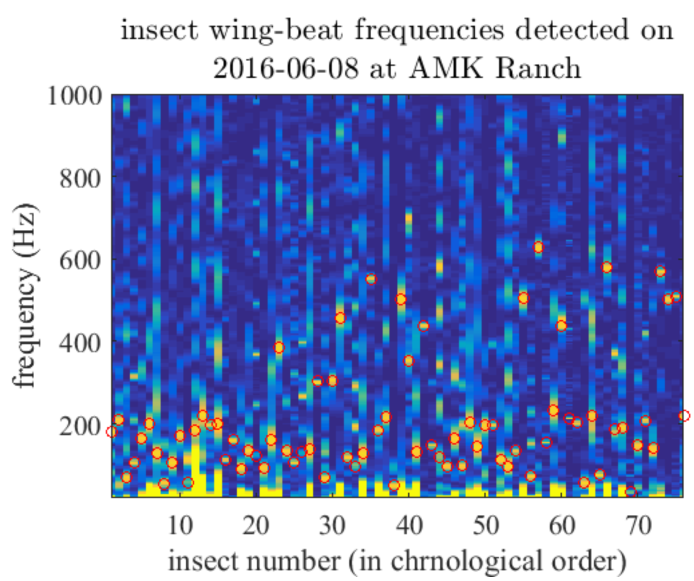


Figure 5.25: Spectrogram of insect wing-beat frequencies with the red circle representing the fundamental frequency. The insects are in chronological order.

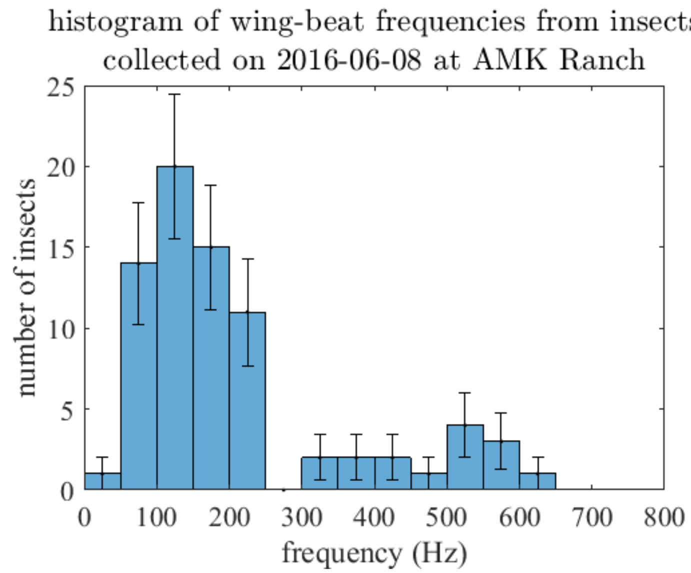


Figure 5.26: Histogram of wing-beat frequencies shows that most insects had wing-beat frequencies under 300 Hz.

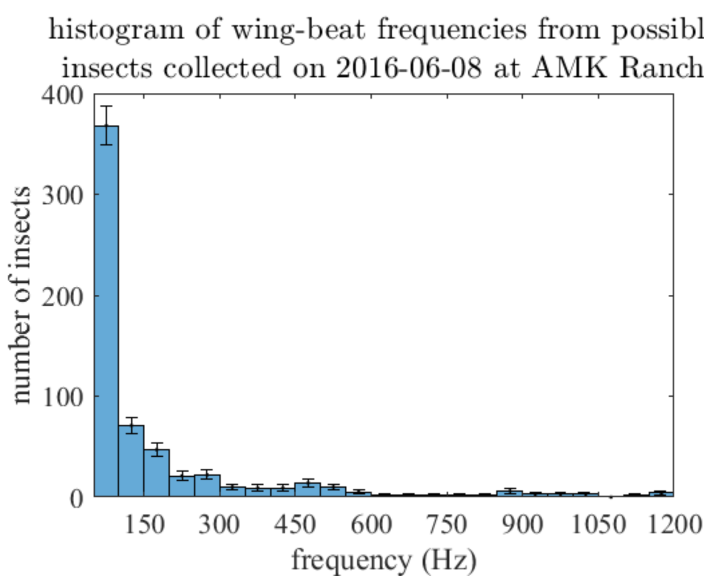


Figure 5.27: Possible-insect histogram of wing-beat frequencies corroborates the "very likely" case.

Chapter Summary

The insect lidar system was designed for the specific task of locating insects in a three-dimensional space in order to monitor their spatial distributions and how these change in response to variables. The system collected time, pan, tilt, and range information for every insect event. The uncertainty in range, pan, and tilt values increased the effective detection location from a point to a volume smaller than a cubic meter. The classification of insects into two sub-groups ("very likely" and "somewhat likely") was done manually because most signals were not clean enough to be considered "very likely" insects. However, the fact that many trends in the "very likely" cases matched those from the "somewhat likely" ones, most of the possible insects were actual insects.

The spatial and temporal distributions of both "very likely" and "somewhat likely" insects provided some entomological conclusions. First, the increase in insect numbers during the dusk part of the day was very apparent and previous studies note the importance of humidity and other factors on insect abundance [54]. Variations in altitude also suggested a dependence on time. During the main part of the day, most insects were low and close to the vegetation, and entered higher altitudes more frequently during dusk. Although the wing-beat frequency information for possible insects was only preliminary, it mostly matched that of the "very likely" insects and most of the wing-beat oscillations were slow.

The results presented in the plots and the discussion surrounding them show the usefulness of the instrument and just some of the ways the data can be interpreted. Further analysis by entomologists could shed light on a variety of trends that are useful for studying insects.

CONCLUSION

Insect behaviors in their natural environments are not well understood and entomologists require novel tools in order to research them. We developed a scanning lidar system that can detect insects of various wing-beat frequencies in their natural environment and map out their locations in space and time. Additionally, future iterations could, potentially, use the frequency and temporal information to infer insect species, age, and sex.

I designed, constructed, tested, and deployed the instrument in field experiments. The design was inspired by previous work and available components. It was built in order to show the feasibility of such a system in the field. Although the hardware and software require improvements before a user-friendly instrument can be presented to the entomological community, the insect lidar was able to accomplish the task of detecting flying insects in their natural environment and can be deployed as is. The following sections describe my contributions, the improvements, and future work toward an entomological lidar system.

My Contributions

My work on the project ranged from designing the system to implementing it in the field experiments. Over the course of the project, new ideas and discoveries led us to approach the project differently. I have outlined the chronological order of the project below.

I designed and built the first iteration which used the telescope as the transmitter and receiver along with axicon lenses and an annular mirror. I worked to align and optimize the system such that half the clear aperture was used for transmitting the beam. In this design, the transmitting beam passed through the hole in the annular mirror and then into the telescope where it expanded. The received beam used the outer half of the telescope aperture and thus reflected off of the annular mirror and into the photomultiplier tube (PMT), which registered this signal and output a current to the analog to digital converter (ADC). As the transmitting light passed thorough the hole in the annular mirror, some photons scattered into the PMT and a fair amount of ringing eliminated the first range bins. I worked to eliminate the ringing by implementing operational amplifiers, but I was not able to reduce it enough to regain the first range bins, and so I redesigned the system.

In the second (and current) iteration, I designed and constructed a system where the transmitted beam was controlled by an independent set of optical components. Once built, I used and modified the software development kit (SDK) provided by the ADC manufacturer to collect data. I wrote algorithms to process and analyze the data. The first experiments were conducted within the lab by pointing the lidar at a mechanical chopper. The modulated signal would produce well defined peaks in the time-series data and thus was a useful tool. These experiments helped tweak the hardware and software to a point where it was ready to send into the field. I conducted the next experiments at the beehive. Again, I used these to make modifications to the system. Throughout this process, I was also attempting to reduce the PMT ringing with different operational amplifiers.

I built custom adapters for the telescope eye-piece and mount. The back end of the telescope did not couple into our lens-tubes and so I designed and manufactured a custom adapter that coupled the telescope to the lens-tube assembly. Additionally, the scanning mount required custom adapters so that it could be mounted to an optical breadboard and so that the telescope could be mounted to it. I also designed and manufactured these parts.

The scanning mount connected to the computer via an RS-232 serial port. I wrote software to control the mount and integrated it into the software which controlled the data collection on the ADC. By executing a single script (see Appendix A), the lidar could run and collect data autonomously. Modifications could easily be made to change step size or scan region.

Once the hardware was functional and the algorithms were operational, I took the system to the AMK Ranch research station at Grand Teton National Park. Over the course of four days, I ran the lidar system and collected gigabytes of data. I then tried to write an algorithm to automatically classify insect events but was only partially successful. The intricacies of various insect signals made absolute classification very difficult. In an attempt to built a training set, I manually looked through a single day's worth of data in a time-consuming, but thorough, three-step process. Although I did not have time to use the training set to write an effective automatic classification algorithm, I did collect 76 "very likely" insect events and 662 "somewhat likely" insect events that I then used to analyze insect behavior.

I analyzed the "very likely" and "somewhat likely" insect events against multiple variables. I viewed how the number of detected insects varied with time, how insect spatial distributions were different during different times of day or for different fundamental wing-beat frequencies. I also inspected the fundamental frequencies and how these varied for a single day. This analysis showed me that this instrument can be quite powerful in monitoring insect spatial distributions and how they change in response to variables.

Instrument Modifications and Improvements

The scanning lidar was designed to detect small, flying insects many meters away. In order to meet this goal, it required a large telescope and other corresponding equipment. Furthermore, it was designed to scan under varying conditions that produced signals that were difficult to classify as insects. After deployment in the field and the processing of multiple gigabytes of raw data, certain hardware and software improvements are clear.

Hardware

The instrument was based around a ⌀304.8 mm telescope in order to collect a lot more eye-safe laser-light scattered from small insect wings. The accompanying optics were attached to the telescope and together the system had a mass of 38.5 kg. This large weight sat on top of a scanning mount which rotated in steps of about 0.5°. As it stepped from one angle to the next, the instrument wiggled at each stop. As the system was wiggling, it was also collecting data. Within some time-series data, a low-frequency oscillation can be seen in what would normally look like a hard target. Although such an issue did not cause us to misclassify an insect, it potentially made it difficult to write an algorithm to automate the process. Also, the high inertia of the instrument slowed the rotations down. With a lighter system, quicker movement from bin to bin would allow for more data collection and the lack of wiggling artifacts in the data would allow for a better classification algorithm.

In addition to changes in receiving optics, transmitting optics could also use improvements. The use of a low-power laser restricted the maximum transmitted beam size. By moving to a more powerful laser, the beam size could be increased while maintaining a maximum range bin of about 100 m and eye-safety. Furthermore, the axicon system (used in the first iteration) could be revamped such that the outgoing and incoming beam share the telescope's aperture. On the other hand, moving to an even higher-powered laser would allow for a smaller telescope and thus a smaller mass of the instrument. Either way, changes to the fundamental hardware could reduce mass and increase beam diameter while maintaining the maximum range bin.

Analog-to-digital conversion speed dictated the size of the range bins and the bit depth determined the dynamic range. Newer digitizer cards have faster acquisition speeds that also store and save the data in more efficient ways. Fast cards with higher bit depths also exist and can potentially increase the number of insect detection events. With proper funding and clever designs, one could construct a lidar system that can not only detect more insects quicker, but can be easily transported and deployed.

Software

The software that ran the experiments was able to control multiple components and store data at the same time. There exist two serious problems that need to be addressed. Within the analog-to-digital converter's software development kit, a line of code can be edited to either save data as voltages or 14-bit integers. The code was modified to save as integers, but somewhere an error caused the values to drift above 2^{14}. Fixing this problem would allow the lidar system to be calibrated and it would allow a more robust processing algorithm to be designed. Also, the scanning speed of the system could likely be increased by minimizing or rewriting slow processes. Speeding up the system would provide a finer look at temporal information.

A more pressing issue is the development of an automated classification algorithm. The data set collected and manually processed from 8 June 2016 can provide a very valuable training set in order to learn which parameters distinguish insects from obstructions. With careful consideration of parameters and a statistically verified threshold for each one, a robust and significant algorithm can be produced. Due to the time-consuming nature of manually classifying one day's worth of data, there was not enough time to write a classification algorithm; however, future researchers could develop one with relative ease.

Future Experiments

Either with a redesigned system or the current iteration, many useful experiments can be conducted next. Many man-made variables might influence the way insects distribute themselves in three dimensions. By scanning a region while the variable is absent and then again when it is present, one can observe the effects. A likely scenario may involve a light source placed in a field and turned on for a number of scans and then off for another set of scans. The researchers can answer questions like: do insects move toward or away from the light? are insects with different fundamental wing-beat frequencies attracted or repelled differently? how far do insects flee from the source? or how close do they get if attracted? Similar questions can be asked of a noise source instead of a light source. Different wavelengths of light and sound can be experimented with. Insect spatial distribution plots are simple to make and easy to interpret and entomologists could deduce a lot of information.

The addition and subtraction of various contrived variables are the easiest cases to interpret due to their relative values. Since the insect lidar cannot count absolute insect numbers, a relative measurement provides the most information. The turning on and off of a device would allow a researcher to observe the effect. Some experiments would not necessarily require that a variable is activated or deactivated. Much like the number of insects detected in time (Figure 5.10), we see that insects tend to be more present during dusk than other hours.

Whether or not this tool is improved in further iterations, in its current state it is a functioning tool that can potentially answer many questions in the entomological community. The system has proven its ability to spatially locate insects, identify their existence in time, and determine their fundamental wing-beat frequency. Using these attributes, researchers can perform countless experiments and begin to answer important questions about natural insect behavior and how humans are affecting it.

REFERENCES CITED

[1] National Park Service - U.S. Department of the Interior. Yellowstone National Park Annual Visitation.

[2] George. McGavin. *Essential entomology : an order-by-order introduction*. Oxford University Press, 2001.

[3] Arthur D. Chapman. Numbers of Living Species in Australia and the World. *Australian Biodiversity Information Services2*, 2, 2009.

[4] Rodolfo Dirzo, Hillary S Young, Mauro Galetti, Gerardo Ceballos, Nick J B Isaac, and Ben Collen. Defaunation in the Anthropocene.

[5] Birgit Fessl and Sabine Tebbich. Philornis downsi- a recently discovered parasite on the Galápagos archipelago - a threat for Darwin's finches? *Ibis*, 144(3):445–451, jun 2002.

[6] Jennifer A H Koop, Peter S. Kim, Sarah A. Knutie, Fred Adler, and Dale H. Clayton. An introduced parasitic fly may lead to local extinction of Darwin's finch populations. *Journal of Applied Ecology*, 53(2):511–518, 2016.

[7] R. Aljaryian and L. Kumar. Changing global risk of invading greenbug Schizaphis graminum under climate change. *Crop Protection*, 88, 2016.

[8] R. J C Cannon. The implications of predicted climate change for insect pests in the UK, with emphasis on non-indigenous species. *Global Change Biology*, 4(7):785–796, 1998.

[9] Myron P Zalucki, Asad Shabbir, Rehan Silva, David Adamson, Liu Shu-Sheng, and Michael J Furlong. Estimating the Economic Cost of One of the World's Major Insect Pests, Plutella xylostella (Lepidoptera: Plutellidae): Just How Long is a Piece of String? *Source: Journal of Economic Entomology*, 105(4):1115–1129.

[10] J. van Roy, J. De Baerdemaeker, W. Saeys, and B. De Ketelaere. Optical identification of bumblebee species: Effect of morphology on wingbeat frequency. *Computers and Electronics in Agriculture*, 109:94–100, 2014.

[11] Simon G. Potts, Jacobus C. Biesmeijer, Claire Kremen, Peter Neumann, Oliver Schweiger, and William E. Kunin. Global pollinator declines: Trends, impacts and drivers. *Trends in Ecology and Evolution*, 25(6):345–353, 2010.

[12] Joseph Shaw, Nathan Seldomridge, Dustin Dunkle, Paul Nugent, Lee Spangler, Jerry Bromenshenk, Colin Henderson, James Churnside, and James Wilson. Polarization lidar measurements of honey bees in flight for locating land mines. *Optics express*, 13(15):5853–5863, 2005.

[13] Kevin S Repasky, Joseph a Shaw, Ryan Scheppele, Chrostopher Melton, John L. Carsten, and Lee H. Spangler. Optical detection of honeybees by use of wing-beat modulation of scattered laser light for location explosives and land mines. *Applied optics*, 45(8):1839–1843, 2006.

[14] David S Hoffman, Amin R Nehrir, Kevin S Repasky, Joseph a Shaw, and John L Carlsten. Range-resolved optical detection of honeybees by use of wing-beat modulation of scattered light for locating land mines. *Applied optics*, 46(15):3007–3012, 2007.

[15] Erik S Carlsten, Geoffrey R Wicks, Kevin S Repasky, John L Carlsten, Jerry J Bromenshenk, and Colin B Henderson. Field demonstration of a scanning lidar and detection algorithm for spatially mapping honeybees for biological detection of land mines. *Applied optics*, 50:2112–2123, 2011.

[16] Ben Collen, Monika Böhm, Rachael Kemp, and Jonathan E M Baillie. Spineless: status and trends of the world's invertebrates. *Zoological Society of London*, 2012.

[17] Silke Nebel, Alex Mills, Jon D Mccracken, and Philip D Taylor. Declines of aerial insectivores in North America follow a geographic gradient. *Avian Conservation & Ecology*, 5(2):1, 2010.

[18] J.A. Thomas. Butterf ly communities under threat: Butterfly populations are declining worldwide as a result of habitat loss and degradation. *Science*, 353(6296), 2016.

[19] J.C. Habel, A. Segerer, W. Ulrich, O. Torchyk, W.W. Weisser, and T. Schmitt. Butterfly community shifts over two centuries. *Conservation Biology*, 30(4), 2016.

[20] C Parmesan, N Ryrholm, C Stefanescu, J K Hill, C D Thomas, H Descimon, B Huntley, L Kaila, J Kullberg, T Tammaru, W J Tennent, J a Thomas, and M Warren. Poleward shifts in geographical ranges of butterfly species associated with regional warming. *Nature*, 399(June):579–583, 1999.

[21] Camille Parmesan. Ecological and Evolutionary Responses to Recent Climate Change. *Annual Review of Ecology, Evolution, and Systematics*, 37(1):637–669, dec 2006.

[22] R.J. Warren, L.D. Chick, B. DeMarco, A. McMillan, V. De Stefano, R. Gibson, and P. Pinzone. Climate-driven range shift prompts species replacement. *Insectes Sociaux*, 63(4), 2016.

[23] R. Chris Shortall, Alison Moore, Emma Smith, J. Mike Hall, P. Ian Woiwod, and Richard Harrington. Long-term changes in the abundance of flying insects. *Insect Conservation and Diversity*, 2(4):251–260, 2009.

[24] Jason W Chapman, V Alistair Drake, and Don R Reynolds. Recent Insights from Radar Studies of Insect Flight. *Annual Review of Entomology*, 56(1):337–356, 2011.

[25] S. C. Reed, C. M. Williams, and L. E. Chadwick. Frequency of Wing-Beat as a Character for Sepearating Species Races and Geographic Varieties of Drosophila. *Genetics*, 27(May):349–361, 1942.

[26] Crawford H. Greenewalt. Dimensional relationships for flying animals. *Smithsonian Miscellaneous Collections*, 144(2):1–45, 1962.

[27] Mikkel Brydegaard. Towards Quantitative Optical Cross Sections in Entomological Laser Radar - Potential of Temporal and Spherical Parameterizations for Identifying Atmospheric Fauna. *PloS one*, 10(8):e0135231, 2015.

[28] SA Combes, DE Rundle, and JM Iwasaki. Linking biomechanics and ecology through predatorprey interactions: flight performance of dragonflies and their prey. *Journal of Experimental*, 2012.

[29] Ge Batista and Ej Keogh. SIGKDD demo: sensors and software to allow computational entomology, an emerging application of data mining. *... Discovery and Data Mining*, pages 761–764, 2011.

[30] Ilyas Potamitis. Classifying insects on the fly. *Ecological Informatics*, 21:40–49, 2014.

[31] a. Moore and R. H. Miller. Automated Identification of Optically Sensed Aphid (Homoptera: Aphidae) Wingbeat Waveforms. *Annals of the Entomological Society of America*, 95(1):1–8, 2002.

[32] Mikkel Brydegaard, Alem Gebru, and Sune Svanberg. Super Resolution Laser Radar with Blinking Atmospheric Particles—-Application to Interacting Flying Insects. *Progress In Electromagnetics Research*, 147(147):141–151, 2014.

[33] Emma R. Mullen, Phillip Rutschman, Nathan Pegram, Joseph M. Patt, John J. Adamczyk, and Eric Johanson. Laser system for identification, tracking, and control of flying insects. *Optics Express*, 24(11):11828, 2016.

[34] David Allison and K S Pike. An inexpensive suction trap and its use in an aphid monitoring network. *Journal of Agricultural Entomology*, 5(2):103–107, 1988.

[35] M. Leskinen, I. Markkula, J. Koistinen, P. Pylkkö, S. Ooperi, P. Siljamo, H. Ojanen, S. Raiskio, and K. Tiilikkala. Pest insect immigration warning by an atmospheric dispersion model, weather radars and traps. *Journal of Applied Entomology*, 135(1-2):55–67, 2011.

[36] E A Laubertie, S D Wratten, and J R Sedcole. The role of odour and visual cues in the pan-trap catching of hoverflies (Diptera : Syrphidae). 148:173–178, 2006.

[37] D. R. Gillespie and D. Quiring. Yellow Sticky Traps for Detecting and Monitoring Greenhouse Whitefly (Homoptera: Aleyrodidae) Adults on Greenhouse Tomato Crops. *Journal of Economic Entomology*, 80(3):675–679, 1987.

[38] Ralph E Webb, Floyd F Smith, Henry Affeldtt, Richard W Thimijan˜ ’, Richard F Dudleyt, and Helen F Webb˜. Trapping greenhouse whitefly with coloured surfaces: variables affecting efficacy. *CROP PROTECTION*, 4(3):38–393, 1985.

[39] Fernando E Vega and Pedro Barbosa. An Adjustable Water-Pan Trap for Simultaneous Sampling of Insects at Different Heights Author (s): Fernando E . Vega , Pedro Barbosa and Alejandro Pérez Panduro Source : The Florida Entomologist , Vol . 73 , No . 4 (Dec ., 1990), pp . 656-660 Publishe. *The Florida Entomologist*, 73(4):656–660, 1990.

[40] D. R. Reynolds and J. R. Riley. Remote-sensing, telemetric and computer-based technologies for investigating insect movement: A survey of existing and potential techniques. *Computers and Electronics in Agriculture*, 35(2-3):271–307, 2002.

[41] R W Mankin. Acoustical detection of Aedes taeniorhynchus swarms and emergence exoduses in remote salt marshes. *Journal of the American Mosquito Control Association*, 10(Nayar 1985):302–308, 1994.

[42] Stephen E Reutebuch, Robert J McGaughey, Hans-Erik Andersen, and Ward E Carson. Accuracy of a high-resolution lidar terrain model under a conifer forest canopy. *Canadian journal of Remote Sensing*, 29(5):527–535, 2003.

[43] Joseph A. Shaw, Kevin S. Repasky, John L Carlsten, Lee H. Spangler, and David S Hoffman. Optical detection of oscillating targets using modulation of scattered laser light, 2009.

[44] Zuguang Guan, Mikkel Brydegaard, Patrik Lundin, Maren Wellenreuther, Anna Runemark, Erik I. Svensson, and Sune Svanberg. Insect monitoring with fluorescence lidar techniques: field experiments. *Applied Optics*, 49(27):5133, sep 2010.

[45] Carsten Kirkeby, Maren Wellenreuther, and Mikkel Brydegaard. Observations of movement dynamics of flying insects using high resolution lidar. *Scientific Reports*, 6:29083, jul 2016.

[46] Mikkel Brydegaard, Aboma Merdasa, Alem Gebru, Hiran Jayaweera, and Sune Svanberg. Realistic Instrumentation Platform for Active and Passive Optical Remote Sensing. *Applied Spectroscopy*, 70(2):372–385, 2016.

[47] Elin Malmqvist, Samuel Jansson, Sandra Torok, and Mikkel Brydegaard. Effective parameterization of laser radar observations of atmospheric fauna. *IEEE Journal on Selected Topics in Quantum Electronics*, 22(3), 2016.

[48] Jerry J. Bromenshenk, Colin B. Henderson, Robert A. Seccomb, Steven D. Rice, Robert T. Etter, Susan F.A. Bender, Philip j. Rodacy, Joseph A. Shaw, Nathan L. Seldomridge, Lee H.James Spangler, and James J. Wilson. Can Honey Bees Assist in Area Reduction and Landmine Detection? *The Journal of Mine Action*, 7 (3)(3), 2003.

[49] Arthur dobbs (1750) and the discovery of the pollination of flowers by insects. *Bulletin of the Torrey Botanical Club*, 76(3):217–219, 1949.

[50] Xinyan Deng, L. Schenato, Wei Chung Wu, and S. S. Sastry. Flapping flight for biomimetic robotic insects: part i-system modeling. *IEEE Transactions on Robotics*, 22(4):776–788, Aug 2006.

[51] Barbara H. Jennings. Drosophila a versatile model in biology & medicine. *Materials Today*, 14(5):190–195, may 2011.

[52] Anna Runemark, Maren Wellenreuther, Hiran H E Jayaweera, Sune Svanberg, and Mikkel Brydegaard. Rare events in remote dark-field spectroscopy: An ecological case study of insects. *IEEE Journal on Selected Topics in Quantum Electronics*, 18(5):1573–1582, 2012.

[53] Michael Minardi. Copyright (c) 2009, michael j. minardi phd. all rights reserved. https://www.mathworks.com/matlabcentral/fileexchange/13-interpsinc-m?s_tid=srchtitle. 2009.

[54] Cedric Gillott. *Entomology*. Plenum Press, New York, 1980.

APPENDICES

APPENDIX A

HARDWARE CONTROL SCRIPTS

The following pages contain the code I wrote to run the hardware. The highest level script is first, and the called scripts appear in no particular order.

Insect_Lidar_Run_Experiment | Martin Tauc | 2016_02_18

Table of Contents

user must define:

tilt (in degrees)

```
starting_tilt=1.50;
final_tilt=15.0;
d_tilt=0.5;
% pan (in degrees)
starting_pan=39.00;
final_pan=42.50;
d_pan=0.5;
% define string for saving files (ex., 'AMK_Ranch-')
location='AMK_Ranch-';
```

This function runs the entire lidar experiment. Once the Setup.m file has been modified along with the paramaters above to what the user needs, and all equipment is hooked up,this script will run autonomously.

```
clear
% connect to the pan-and-tilt mount
Insect_Lidar_QPT130_connect;
% connect to the ADC card (using SDK)
Insect_Lidar_GAGE_CS14200_connect;
% define a pause time (mount requires 0.1 s of pause)
pause_time=0.12;
% setup wait bar
wb=waitbar(0,'please wait...');


% define the date directory
date_dir=['C:\Martin_Tauc\Research\Insect Lidar\Field Tests\Stored
 Data\',datestr(now,'yyyy-mm-dd')];
% define the time directory
time_dir=[date_dir,'\',location,datestr(now,'HHMMSS')];

% create a directory if it does not yet exist
if exist(date_dir,'dir') ~= 7
    mkdir(date_dir);
end
```

```
% create the time directory to hold all the run files
mkdir(time_dir);
% preallocate raw data array and time stamp vector
full_data=zeros(200,1024);
tsdata=zeros(1,1024);
```

begin scan

define variable for waitbar

```
tl=starting_tilt*100; %100*degrees
% define temproary variable and tilt start
tld=tl;
% define pan start
start_pan=starting_pan*100; % 100*degrees
% define pan end
end_pan=final_pan*100;   % 100*degrees
% create pan vector
pan_s_e=[start_pan end_pan];
% define change in pan
delta_pan=d_pan*100; % 100*degrees
% define change in tilt
delta_tilt=d_tilt*100; % 100*degrees
% set tilt end
tilt_final=final_tilt*100; % 100*degrees
% define number of runs for waitbar
nor=((-start_pan+end_pan)./delta_pan)*((-tld+tilt_final+1)./
delta_tilt);
% set counter variables
q=0;
n=0;
nn=1;
% run while tilt location is less than the final tilt
while tld<tilt_final
```

move to absolute coordinate

increase n counter

```
    n=n+1;
    % begin timer
    tic
    % set counter variable
    oc=0;
    % set check status to trigger first loop iteration
    check=false;
    % set to trigger first loop eteration
    out(1)=uint8(15);
    % check for the redudancy check and that the the transmission is
    % acknowledged (indicated by a '06')
    while check~=true || out(1)~=uint8(06)
        % send the absolute coordinates ('33') to the mount in two
        % bytes and do a lrc
```

```
        fwrite(QPT130,char(Insect_Lidar_LRC_out([uint8(hex2dec('33')),Insect_Lidar_QPT_in
                % pause is required by the mount
                pause(pause_time)
                % if the command was recieved, the mount sends back the
                % original command for check
                if QPT130.BytesAvailable > 0
                    % confirm proper bytes were sent
                    out=fread(QPT130,QPT130.BytesAvailable);
                    % determine if lrc matched
                    check=Insect_Lidar_LRC_in(out');
                    % increase count
                    oc=oc+1;
                    % store output at cell
                    raw_output{n,oc}=out';
                    % if lrc was true
                    if check==true
                        % save output as cell
                        output{n,oc}=Insect_Lidar_Remove_ESC_Char(out');
                        pause(pause_time)
                    end
                end
            end
            % set to trigger
            check=false;
            out(1)=uint8(15);
            % send "jog" command
            while check~=true || out(1)~=uint8(06)
                fwrite(QPT130,jog);
                pause(pause_time)
                if QPT130.BytesAvailable > 0
                    out=fread(QPT130,QPT130.BytesAvailable);
                    check=Insect_Lidar_LRC_in(out');
                    oc=oc+1;
                    raw_output{n,oc}=out';
                    if check == true
                        output{n,oc}=Insect_Lidar_Remove_ESC_Char(out');
                        pause(pause_time)
                    end
                end
            end
            % set to trigger
            check=false;
            out(1)=uint8(15);
            k=1;
            % wait for mount to reach destination
            while isequal(output{n,k},output{n,k+1})==0 || check~=true ||
        out(1)~=uint8(06)
                fwrite(QPT130,jog);
                pause(pause_time)
                if QPT130.BytesAvailable > 0
                    out=fread(QPT130,QPT130.BytesAvailable);
                    check=Insect_Lidar_LRC_in(out');
                    raw_output{n,k+2}=out';
```

```
            if check==true
                k=k+1;
                output{n,k+1}=Insect_Lidar_Remove_ESC_Char(out');
                pause(pause_time)
            end
        end
    end
    % increase count
    oc=k+1;
    % store current location
    cur_loc=output{n,oc};
    % store current location as hex files
    [pld,pli]=Insect_Lidar_QPT_hex2int(cur_loc(3),cur_loc(4));
    [tld,tli]=Insect_Lidar_QPT_hex2int(cur_loc(5),cur_loc(6));
    panloc=pld/100;
    tiltloc=tld/100;
    % display current location of mount in degrees
    disp(['Pan location is ', sprintf('%4.2f',panloc), ' degrees'])
    disp(['Tilt location is ', sprintf('%4.2f',tiltloc),' degrees'])
    fdm=0;
    q=q+1;
    % record data using the modified SDK
    Insect_Lidar_GageContinuousRecord;
    % save timer
    tt=toc;
    % used to move pan right or left depending on orientation to save
    % time
    nn=nn+(-1)^(n+1);
```

Move QPT through entire sequence moving x degrees at a time

```
    while (-1)^(n+1)*pld<(-1)^(n+1)*pan_s_e(nn)
        % set counter
        fdm=fdm+1;
        % start timer
        tic
        % set to trigger
        check=false;
        out(1)=uint8(15);
        while check~=true || out(1)~=uint8(06)
            % send coordinate to mount and perform check

 fwrite(QPT130,char(Insect_Lidar_LRC_out([uint8(hex2dec('34')),Insect_Lidar_QPT_in
+1)*delta_pan),Insect_Lidar_QPT_int2hex(0000)])));
            pause(pause_time)
            if QPT130.BytesAvailable > 0
                out=fread(QPT130,QPT130.BytesAvailable);
                check=Insect_Lidar_LRC_in(out');
                raw_output{n,oc+1}=out';
                if check == true
                    output{n,oc+1}=Insect_Lidar_Remove_ESC_Char(out');
```

```
                    oc=oc+1;
                    pause(pause_time)
                end
            end
        end
        % save the raw data, start address, pulse time stamp, and pan
        % and tilt locations while mount moves to new location
        save(fullfile(time_dir,
[sprintf('%05.0f',q),'_','P',sprintf('%04.0f',pli),'T',sprintf('%04.0f',tli)]),'fu

        % set to trigger
        check=false;
        out(1)=uint8(15);
        m=oc-1;
        % send "jog" command and wait for mount to get to proper
        % location
        while isequal(output{n,m},output{n,m+1})==0 || check~=true ||
 out(1)~=uint8(06)
            fwrite(QPT130,jog);
            pause(pause_time)
            if QPT130.BytesAvailable > 0
                out=fread(QPT130,QPT130.BytesAvailable);
                check=Insect_Lidar_LRC_in(out');
                raw_output{n,m+2}=out';
                if check==true
                    m=m+1;
                    output{n,m+1}=Insect_Lidar_Remove_ESC_Char(out');
                    pause(pause_time)
                end
            end
        end
        % increase counter
        oc=m+1;
        clear cur_loc panloc tiltloc
        % store current location
        cur_loc=output{n,oc};
        % convert location into integers
        [pld,pli]=Insect_Lidar_QPT_hex2int(cur_loc(3),cur_loc(4));
        [tld,tli]=Insect_Lidar_QPT_hex2int(cur_loc(5),cur_loc(6));
        panloc=pld/100;
        tiltloc=tld/100;
        % display pan and tilt locations
        disp(['Pan location is ', sprintf('%4.2f',panloc), '
 degrees'])
        disp(['Tilt location is ', sprintf('%4.2f',tiltloc),'
 degrees'])
        % increase counter
        q=q+1;
        % collect data
        Insect_Lidar_GageContinuousRecord;
        % save timer
        tt=toc;
        % display timer
        disp(tt);
```

```
        % update waitbar
        waitbar(q/nor,wb,['estimated time remaining is
 ',sprintf('%4.1f',tt*(nor-q)/60),' minutes'])

    end
    % increase tilt angle by delta tilt
    tl=tl+delta_tilt;

end
% save final run
save(fullfile(time_dir,
[sprintf('%05.0f',q),'_','P',sprintf('%04.0f',pli),'T',sprintf('%04.0f',tli)]),'fu
% clear and release the mount and ADC card and waitbar
ret = CsMl_FreeSystem(handle);
fclose(QPT130);
delete(QPT130);
delete(wb);
clear
```

Published with MATLAB® R2016b

QPT130_connect | Martin Tauc | 2016_01_28

This script is used to establish communication with the MoogS3 QPT-130 pan/tilt mount.

```
clear
% define the device through the serial port
QPT130=serial('com3');
% open communication and check some properties
fopen(QPT130);
get(QPT130,
{'BaudRate','DataBits','Parity','StopBits','Terminator','Status'});

% import starting protocol and convert from hex to dec
[Tx_dec,Tx_str]=Insect_Lidar_add_H;

% send starting protocol to mount
for n=1:size(Tx_str,2)
    fwrite(QPT130,char(Tx_dec{n}));
    pause(0.1);
end
% set "jog" command
jog=char(Tx_dec{1});
% send "jog" command to mount
fwrite(QPT130,jog);
% pause is required by mount
pause(0.25);
% the mount has recieved and proccessed the command
if QPT130.BytesAvailable > 0
    out=fread(QPT130,QPT130.BytesAvailable);
end
```

Published with MATLAB® R2016b

Insect_Lidar_GAGE_CS14200_connect | Martin Tauc | 2017-01-30

connect with the ADC card the following commands are borrowed from the SDK

```
systems = CsMl_Initialize;
CsMl_ErrorHandler(systems);
[ret, handle] = CsMl_GetSystem;
CsMl_ErrorHandler(ret);
[ret, sysinfo] = CsMl_GetSystemInfo(handle);
s = sprintf('-----Board name: %s\n', sysinfo.BoardName);
disp(s);
Setup(handle);
CsMl_ResetTimeStamp(handle);
ret = CsMl_Commit(handle);
CsMl_ErrorHandler(ret, 1, handle);
```

Published with MATLAB® R2016b

Table of Contents

Insect_Lidar_import_starting_protocol | Martin Tauc | 2017-01-30

this script imports a .csv file that contains the initial mount setup commands

```
function [RX, TX]=Insect_Lidar_import_starting_protocol
```

Import data from text file.

Script for importing data from the following text file:

```
C:\Users\user\Documents\Research\Insect Lidar\Field
Tests\Scripts\QPT130 Mount Control\starting_protocol.csv
```

To extend the code to different selected data or a different text file, generate a function instead of a script.

```
% Auto-generated by MATLAB on 2016/01/28 13:40:51
```

Initialize variables.

```
filename = 'C:\Users\user\Documents\Research\Insect Lidar\Field Tests
\Scripts\QPT130 Mount Control\starting_protocol00.csv';
delimiter = ',';
startRow = 2;
```

Format string for each line of text:

```
column1: text (%s)
```

```
% column2: text (%s)
% For more information, see the TEXTSCAN documentation.
formatSpec = '%s%s%[^\n\r]';
```

Open the text file.

```
fileID = fopen(filename,'r');
```

Read columns of data according to format string.

This call is based on the structure of the file used to generate this code. If an error occurs for a different file, try regenerating the code from the Import Tool.

```
dataArray = textscan(fileID, formatSpec, 'Delimiter',
 delimiter, 'HeaderLines' ,startRow-1, 'ReturnOnError', false);
```

Close the text file.

```
fclose(fileID);
```

Post processing for unimportable data.

No unimportable data rules were applied during the import, so no post processing code is included. To generate code which works for unimportable data, select unimportable cells in a file and regenerate the script.

Allocate imported array to column variable names

```
TX = dataArray{:, 1};
RX = dataArray{:, 2};
```

Clear temporary variables

```
clearvars filename delimiter startRow formatSpec fileID dataArray ans;
```

Published with MATLAB® R2016b

Insect_Lidar_add_H | Martin Tauc | 2017-01-30

this script converts the hexadecimal string into a decimal string

```
function [Tx_dec, Tx_str]=Insect_Lidar_add_H

% shake hands with mount. The sequence was borrowed from the
 accompanying
% software
[Rx,Tx]=Insect_Lidar_import_starting_protocol;
```

convert from a hex string to decimal string

```
for n=1:size(Tx,1)
    in_hex=Tx{n};
    % split string into row vector
    sp=strsplit(in_hex);
    % convert from hex to dec
    in_dec=hex2dec(sp);
    % convert dec to cell and row to column vector
    Tx_dec{n}=in_dec';
    % clear
    clear in_hex
    clear sp
    clear in_dec
end

for n=1:size(Tx_dec,2)
    Tx_str{n}=sprintf('%1.0f ',Tx_dec{n});
end

end
```

Published with MATLAB® R2016b

Remove_ESC_Char Martin Tauc | 2016-03-07

This function removes esc characters so bytes coming from the pan tilt mount are readable

```
function code_out=Insect_Lidar_Remove_ESC_Char(code)
% define esc character and locate it
control_char=uint8(27);
index=find(code==control_char);
% set counter
n=1;
while n<=length(index)
    % write into cell
    index_new{n}=index(n);
    % subtract 128 from value after esc character
    code(index(n)+1)=code(index(n)+1)-128;
    % if not the last value
    if n~=length(index)
        % if the next index value is also an esc, skip it
        if index(n+1)==index(n)+1
            % skip it
            n=n+2;
        else
            % otherwise don't
            n=n+1;
        end
    else
        % if this is the last value, move to end
        n=n+1;
    end
end

if exist('index_new') == 1
    % convert from cell to integer
    index_n=cell2mat(index_new);
    % remove the esc characters
    for n=length(index_n):-1:1
        code(index_n(n))=[];
    end
end
code_out=code;
```

Published with MATLAB® R2016b

Insect_Lidar_QPT_int2hex_Martin Tauc | 2016-03-04

QPT_integer2hex The QPT 130 mount requires that integer values get passed as two's compliment little endian. This program takes an integer between -32768 and 32767 and converts it into a 2 byte integer value

```
function unsigned8=Insect_Lidar_QPT_int2hex(int_val)

% set value to unsigned 8 integer
unsigned8=typecast(int16(int_val),'uint8');
% turn it into a hex value seperated by commas
hex_val=[num2str(unsigned8(1)),',',num2str(unsigned8(2)),','];
```

Published with MATLAB® R2016b

LRC_out | Martin Tauc | 2016_02_18

This program adds the Logarithmic Redundancy Check (LRC) and the start command (02) and the end command (03) user must imput everything inbetween in hex form.

```
function [output]=Insect_Lidar_LRC_out(input)
% input a vector of numbers in hex
[x,y]=size(input);
% set the first value for the lrc
lrc=input(1);
% exclusive or with the other values for the lrc
for m=2:y
    lrc=bitxor(lrc,input(m));
end
% add lrc into original vector
code=[input,lrc];
% define the control characters including the escape character
control_char=uint8([02, 03, 06, 15, 27]);
% set counter
k=0;
% check for control characters and add in an escape character if found
for n=1:length(control_char)
    % check if any control characters exist in the vector
    k=k+any(code==control_char(n));
end
% if at least one control character was found, add in escape character
if k~=0
    code_N=Insect_Lidar_Insert_ESC_Char(code);
else
    code_N=code;
end

output=[uint8(02), code_N, uint8(03)];
```

Published with MATLAB® R2016b

Insect_Lidar_LRC_in | Martin Tauc | 2016_03_07

This program looks at the incoming bytes from the mount and checks the Logarithmic Redundancy Check (LRC).

```
function [check]=Insect_Lidar_LRC_in(in_from_QPT)
% remove esc characters
input=Insect_Lidar_Remove_ESC_Char(in_from_QPT);
% identify the final character
lrc_final=uint8(input(end-1));
% identify middle commands
com_int=uint8(input(2:end-2));
% determine number of commands
[x,y]=size(com_int);
% set first lrc value
lrc=com_int(1);
% perform exclusive or on each byte
for m=2:y
    lrc=bitxor(lrc,com_int(m));
end
% output whether the lrc matched or not
if lrc_final==lrc
    check=true;
else
    check=false;
end
```

Published with MATLAB® R2016b

Insect_Lidar_GageMultipleRecord | from GAGE SDK (modified by Martin Tauc) | 2017-01-30

This sample program demonstrates how to do a multiple record capture. The system's acquistion, channel and trigger parameters are set. The data is captured and retrieved and the data for each channel and multiple segment is saved to a separate file.

```
% clearvars -except panloc tiltloc QPT130 jog output tl full_data;
clearvars -
except panloc tiltloc QPT130 jog output tl systems ret handle...

  raw_output sysinfo s acqInfo full_data fdy fdm n acqInfo.SegmentCount...
    start_pan end_pan delta_pan delta_tilt pause_time oc tilt_final;

% date stamp current date and time
datestamp=datestr(now,30);
% run SDK scripts
CsMl_ResetTimeStamp(handle);
[ret, acqInfo] = CsMl_QueryAcquisition(handle);
ret = CsMl_Capture(handle);
CsMl_ErrorHandler(ret, 1, handle);
status = CsMl_QueryStatus(handle);
while status ~= 0
    status = CsMl_QueryStatus(handle);
end
% get information from Setup.m
transfer.Channel = 1;
transfer.Mode = CsMl_Translate('TimeStamp', 'TxMode');
transfer.Length = acqInfo.SegmentCount;
transfer.Segment = 1;
[ret, tsdata, tickfr] = CsMl_Transfer(handle, transfer);
transfer.Mode = CsMl_Translate('Default', 'TxMode');
transfer.Start = -acqInfo.TriggerHoldoff;
transfer.Length = acqInfo.SegmentSize;

% Regardless  of the Acquisition mode, numbers are assigned to
 channels in a
% CompuScope system as if they all are in use.
% For example an 8 channel system channels are numbered 1, 2, 3, 4, ..
 8.
% All modes make use of channel 1. The rest of the channels indices
 are evenly
% spaced throughout the CompuScope system. To calculate the index
 increment,
% user must determine the number of channels on one CompuScope board
 and then
% divide this number by the number of channels currently in use on one
 board.
% The latter number is lower 12 bits of acquisition mode.

MaskedMode = bitand(acqInfo.Mode, 15);
```

```
ChannelsPerBoard = sysinfo.ChannelCount / sysinfo.BoardCount;
ChannelSkip = ChannelsPerBoard / MaskedMode;

% Format a string with the number of segments and channels so all
 filenames
% have the same number of characters.
format_string = sprintf('%d', acqInfo.SegmentCount);
MaxSegmentNumber = length(format_string);
format_string = sprintf('%d', sysinfo.ChannelCount);
MaxChannelNumber = length(format_string);
format_string = sprintf([datestamp, '%%s_CH%%0%dd-%%0%dd.mat'],
 MaxChannelNumber, MaxSegmentNumber);

for channel = 1:ChannelSkip:sysinfo.ChannelCount
    transfer.Channel = channel;
    for i = 1:acqInfo.SegmentCount
        transfer.Segment = i;
        [ret, data, actual] = CsMl_Transfer_Tauc(handle, transfer,1);
        CsMl_ErrorHandler(ret, 1, handle);

        % Note: to optimize the transfer loop, everything from
        % this point on in the loop could be moved out and done
        % after all the channels are transferred.

        % adjust the size so only the actual length of data is saved
 to the
        % file
        length = size(data, 2);
        if length > actual.ActualLength
            data(actual.ActualLength:end) = [];
            length = size(data, 2);
        end;
        % Get channel info for file header
        [ret, chanInfo] = CsMl_QueryChannel(handle, channel);
        CsMl_ErrorHandler(ret, 1, handle);
        % Get information for ASCII file header
        info.Start = actual.ActualStart;
        full_info(1)=info.Start;
        info.Length = actual.ActualLength;
        full_info(2)=info.Length;
        info.SampleSize = acqInfo.SampleSize;
        full_info(3)=info.SampleSize;
        info.SampleRes = acqInfo.SampleResolution;
        full_info(4)=info.SampleRes;
        info.SampleOffset = acqInfo.SampleOffset;
        full_info(5)=info.SampleOffset;
        info.InputRange = chanInfo.InputRange;
        full_info(6)=info.InputRange;
        info.DcOffset = chanInfo.DcOffset;
        full_info(7)=info.DcOffset;
        info.SegmentCount = acqInfo.SegmentCount;
        full_info(8)=info.SegmentCount;
        info.SegmentNumber = i;
        full_info(9)=info.SegmentNumber;
```

```
        info.TimeStamp = tsdata(i);
        full_info(10)=info.TimeStamp;
        % added pan and tilt location from mount
        info.Pan=panloc;
        full_info(11)=info.Pan;
        info.Tilt=tiltloc;
        full_info(12)=info.Tilt;
        % date and time
        year=str2num(datestamp(1:4));
        month=str2num(datestamp(5:6));
        day=str2num(datestamp(7:8));
        hour=str2num(datestamp(10:11));
        minute=str2num(datestamp(12:13));
        second=str2num(datestamp(14:15));
        full_info(13)=year;
        full_info(14)=month;
        full_info(15)=day;
        full_info(16)=hour;
        full_info(17)=minute;
        full_info(18)=second;
        % save raw data as cell
        full_data{((2*n)-1),i+(fdm*acqInfo.SegmentCount)}=data';
        full_data{(2*n),i+(fdm*acqInfo.SegmentCount)}=full_info';

    end
end
```

Published with MATLAB® R2016b

Insect_Lidar_Insert_ESC_Char Martin Tauc | 2016-03-07

This function enters esc characters so that a byte doesn't place a command character in the wrong place.

```
function esc_corrected_data=Insect_Lidar_Insert_ESC_Char(code)
% set control characters
control_char=uint8([02, 03, 06, 15, 27]);

% save the index where a control character was found
for h=1:length(control_char)
    if isempty(find(code==control_char(h)))~=1
        index{h}=find(code==control_char(h));
    end
end
% put the index vector in order
ind_t=sort(cell2mat(index));
% set counters
k=2;
ind(1)=0;
% create vector of all indexes starting with 0
ind(k:length(ind_t)+(k-1))=ind_t;
% if the final value is a control character, set l=1,else l=0;
if ind_t(end)~=length(code)
    ind(length(ind_t)+k)=length(code);
    l=0;
else
    l=1;
end
% insert the escape character
for h = 1:length(ind)-1
    % if there are no more control characters, print the rest of the
    % vector, otherwise add in the esc character
    if h==length(ind)-1 && l==0
        cn{h}=[code(ind(h)+1:ind(h+1)-1) code(ind(h+1))];
    else
        % esc character is 27 and it is followed by the value + 128
        cn{h}=[code(ind(h)+1:ind(h+1)-1) uint8(hex2dec('1B'))
 code(ind(h+1))+128];
    end
end
% convert from cell to integers
esc_corrected_data=cell2mat(cn);
```

Published with MATLAB® R2016b

APPENDIX B

POST-PROCESSING ALGORITHMS

The following pages contain the algorithms I wrote to process the data. The first two scripts were executed in order (Insect_Lidar_fix_cell_struct was a subscript). The Following three algorithms were used to manually classify insects.

Table of Contents

Insect_Lidar_Adjusted_data | Martin Tauc | 2017-02-01

```
function []=Insect_Lidar_Adjust_Data(date)
```

Folders in a date folder

define directory based on input date

```
stored_data='C:\Martin_Tauc\Research\Stored_data\';
date_dir=[stored_data, date];
% scan folders: create vector of all folders with desired title [NOTE
 THAT TITLE IS
% DEFINED IN 'Insect_Lidar_Run_Experiment.m']
runs=dir([date_dir,'\AMK_Ranch*']);

rn_vec=1:size(runs,1);
```

Files in a run folder

```
for rn = rn_vec
    clear vecs adjusted_data
    %scan folder: create vector of all files within folder
    vecs=dir([date_dir,'\',runs(rn).name]);
    vecs=vecs(~ismember({vecs.name},
{'.','..','processed_data.mat','adjusted_data.mat'}));
    % preallocate adjusted_data structure
    adjusted_data(size(vecs,1))=struct('tilt',[],'pan',[],'data',
[],'time',[],'range',[],'filename',[]);
    % parallel for loop
    parfor vn = 1:size(vecs,1)
        % load each run within scan folder
        rd=load([date_dir,'\',runs(rn).name,'\',vecs(vn).name]);
        % line up data, time, and range

 [adjusteddata,tcdata,range]=Insect_Lidar_fix_cell_struct(rd.full_data,rd.start_ad

        % store important variables in structure
        adjusted_data(vn).tilt=rd.tiltloc;
        adjusted_data(vn).pan=rd.panloc;
        adjusted_data(vn).data=adjusteddata;
        adjusted_data(vn).time=tcdata;
```

```
        adjusted_data(vn).range=range;
        adjusted_data(vn).filename=[runs(rn).name,'\',vecs(vn).name];
    end
    save(fullfile(date_dir,runs(rn).name,
['adjusted_data','.mat']),'adjusted_data','-v7.3');
end

end
```

Published with MATLAB® R2016b

Insect_Lidar_Combine_Vectors | Martin Tauc | 2017-02-01

```
function [vector_data]=Insect_Lidar_Combine_Vectors(date)
```

```
% base directory
stored_data='C:\Martin_Tauc\Research\Insect Lidar\Field Tests\Stored
 Data\';
```

ATTENTION ATTENTION ATTENTION: ATTENTION: tilt_vec and pan_vec may need to be adjusted based on settings set in 'Insect_Lidar_Run_Experiment.m' create appropriate vectors for tilt and pan locations

```
tilt_vec=[];for n=1.5:.5:15;tilt_vec=[tilt_vec repmat(n,[1,8])];end
pan_vec=[];vv=39:.5:42.5;for n=1:size(tilt_vec,2)/8/2;
 pan_vec=[pan_vec vv flip(vv)];end
```

date directory

```
date_dir=[stored_data, date];
% scans within date directory
runs=dir([date_dir,'\AMK_Ranch*']);
rn_vec=1:size(runs,1);
% clump data based on pan and tilt location
for pn=1:size(tilt_vec,2)
    clear vector_data nf
    int=1;
    % pan and tilt values were stored in filenames, extract here
    nf = ['T',sprintf('%3.2f',...
        tilt_vec(pn)),...
        'P',sprintf('%3.2f',...
        pan_vec(pn))];
    % filenames used _ instead of .
    nf(nf=='.')='_';
    % save it in cell
    name_file{pn}=nf;
    % loop through each run to find match with tilt_vec and pan_vec
    for rn = rn_vec
        clear adjusted_data pm
        % load the adjusted data
        load(fullfile(date_dir,runs(rn).name,'adjusted_data.mat'))
        % find matches within 0.245 degress
        pm=find(abs([adjusted_data.tilt]-tilt_vec(pn))<0.245 &
 abs([adjusted_data.pan]-pan_vec(pn))<0.245);
        if exist('pm','var')
            % save the vector as the pan and tilt location
            for spm=1:size(pm,2)
                vector_data.(name_file{pn})
(int)=adjusted_data(pm(spm));
                int=int+1;
            end
        end
```

```
        fprintf('%03.0f - %03.0f\n',pn,rn)
    end
    if exist('vector_data','var')
        % save in 'processed_data' folder
        save(fullfile(date_dir,'processed_data',sprintf('%03.0f-
%s.mat',pn,name_file{pn})),'-struct','vector_data');
    end
end
```

Published with MATLAB® R2016b

Insect_Lidar_fix_cell_struct | Martin Tauc | 2017-02-01

```
function
 [adjusted_data,tcdata_es,range]=Insect_Lidar_fix_cell_struct(full_data,start_addr
% old data is false unless the data is from before 2015 (typically)
if nargin<4
    old_data=false;
end

% start full_data at proper start_address
for n = 1:size(full_data,2)
    adjusted_data(1:length(full_data(:,n))+1+start_address(n),n) = ...
        full_data(-start_address(n):end,n);
end
% remove the final 10 range bins (start address vary between 3 and 13)
adjusted_data=adjusted_data(1:end-10,:);
% number of points
nop=size(adjusted_data,2);
% convert from microseconds to seconds
tcdata=(tsdata-tsdata(1))./1e6;
% create a uniform time vector
dt=diff(tcdata);
tcdata_es=mean(dt)*(0:nop-1);
% sinc interpolate for a uniform time spacing
for n = 1:size(adjusted_data,1)
    adjusted_data(n,:) =
 interpsinc(tcdata,adjusted_data(n,:),mean(dt),1);
end
% remove the first 8 points (they correspond to negative range values)
adjusted_data=adjusted_data(8:end,:);
range=rangefind(8:size(adjusted_data,1)+7);
```

Published with MATLAB® R2016b

Insect_Lidar_manual_insect_finder_step1 | Martin Tauc | 2017-02-01

```
% fols is all vectors
fols=dir(fullfile(pwd));
fols=fols(~ismember({fols.name},{'.','..','events'}));
% if the manual process was already started, this will load the saved
 file
% and start counting at the right values
if exist('C:\Martin_Tauc\Research\Insect Lidar\Field Tests\Stored Data
\2016-06-08\processed_data\events\manual.mat','file')~=0
    load('C:\Martin_Tauc\Research\Insect Lidar\Field Tests\Stored Data
\2016-06-08\processed_data\events\manual.mat');
    int=size(manual.insects,2);
    mnt=size(manual.maybe_insect,2);
    nnt=size(manual.noninsect,2);
else
    int=0;
    mnt=0;
    nnt=0;
end


% loop through each run within each vector and classify the if insect
 or
% maybe or not
for x=1:size(fols,1)
    clear data fn
    % load vector
    data=load(fullfile(pwd,fols(x).name));
    % determine filenames
    fn=fieldnames(data);
    for y=1:size(data.(fn{1}),2)
        % plot each run
        imagesc(data.(fn{1})(y).normalized_data)
        % title for reference
        title(sprintf('file %s with x=%i and y=
%i',fols(x).name,x,y),'interpreter','none');
        % user decides if there is an insecct present, maybe an insect
        % present, or not an insect present
        yn=input('insect (e) or not (w) or maybe (q) ? ','s');
        % depending on classification, the vector name and run name
 are
        % stored along with the corresponding filename from the scan
 folder
        if strcmp(yn,'e')
            int=int+1;
            manual.insects(int).name=fn;
            manual.insects(int).x=x;
            manual.insects(int).y=y;
        elseif strcmp(yn,'w')
```

```
            nnt=nnt+1;
            manual.noninsect(nnt).name=fn;
            manual.noninsect(nnt).x=x;
            manual.noninsect(nnt).y=y;
        else
            mnt=mnt+1;
            manual.maybe_insect(mnt).name=fn;
            manual.maybe_insect(mnt).x=x;
            manual.maybe_insect(mnt).y=y;

        end
    end
    save(fullfile(pwd,'events','manual.mat'),'manual','-v7.3')
end
```

Published with MATLAB® R2016b

Insect_Lidar_manual_insect_finder_step2 | Martin Tauc | 2017-02-01

```
% fols is all vectors
fols=dir(fullfile(pwd));
fols=fols(~ismember({fols.name},{'.','..','events'}));
% if the manual process was already started, this will load the saved
 file
% and start counting at the right values
load('C:\Martin_Tauc\Research\Insect Lidar\Field Tests\Stored Data
\2016-06-08\processed_data\events\manual.mat');

if exist('C:\Martin_Tauc\Research\Insect Lidar\Field Tests\Stored Data
\2016-06-08\processed_data\events\manual_insects.mat','file')~=0
    load('C:\Martin_Tauc\Research\Insect Lidar\Field Tests\Stored Data
\2016-06-08\processed_data\events\manual_insects.mat');
    int=size(manual_insects,2);

else
    int=0;
end
% first choose insects, then maybe_insect
for n=1:size(manual.insects,2)
    % are there insects in multiple range bins?
    more='y';
    while strcmp(more,'y')==1
        clear data fn
        % load in the run
        data=load(fullfile(pwd,fols(manual.insects(n).x).name));
        fn=fieldnames(data);
        figure(1)
        % display the run
        imagesc(data.(fn{1})(manual.insects(n).y).normalized_data)
        title(sprintf('file %s with x=%i and y=%i and n=
%i',fols(manual.insects(n).x).name,manual.insects(n).x,manual.insects(n).y,n),'int
        % input the range number where the insect is
        loc2=input('range number ');
        figure(2)
        % plot that range bin in plot and not image
        plot(data.(fn{1})
(manual.insects(n).y).normalized_data(loc2,:))
        title(sprintf('file %s with x=%i and y=%i and z=%i and n=
%i',fols(manual.insects(n).x).name,manual.insects(n).x,manual.insects(n).y,loc2,n)
        % determine liklyhood of insect
        isinsect=input('is this an insect (y/n/m)? ','s');
        % store insect or non insect based on input
        if strcmp(isinsect,'y') || strcmp(isinsect,'m')
            int=int+1;
            manual_insects(int).filename=fn;
            manual_insects(int).x=manual.insects(n).x;
            manual_insects(int).y=manual.insects(n).y;
```

```
            manual_insects(int).z=loc2;
            manual_insects(int).yesORmaybe=isinsect;
        end
        % is there another insect in a different range bin?
        more=input('are there other insects (y/n)? ','s');

    end

end
save(fullfile(pwd,'events','manual_insects.mat'),'manual_insects','-
v7.3')
```

Published with MATLAB® R2016b

Table of Contents

Insect_Lidar_manual_insect_finder_step3 | Martin Tauc | 2017-02-02

```
function []=Insect_lidar_manual_insect_finder_step3()
set_plot_defaults
start_file='C:\Martin_Tauc\Research\Insect Lidar\Field Tests\';
date='2016-06-08';
filename=fullfile(start_file,'Stored Data',date,'processed_data');
load(fullfile(filename,'events','manual_insects.mat'))
fols=dir(filename);
fols=fols(~ismember({fols.name},{'.','..','events'}));
try
    load('C:\Martin_Tauc\Research\Insect Lidar\Field Tests\Stored Data
\2016-06-08\processed_data\events\manual_final_insect.mat')
catch
    disp('no mfi yet');
end
try
    inz=size(manual_final_insect.very_unlikely,2);
catch
    inz=0;
end
try
    inx=size(manual_final_insect.somewhat_unlikely,2);
catch
    inx=0;
end
try
    inc=size(manual_final_insect.somewhat_likely,2);
catch
    inc=0;
end
try
    inv=size(manual_final_insect.very_likely,2);
    catch53
    inv=0;
end
```

```
try
    inb=size(manual_final_insect.multiple_insects,2);
catch
    inb=0;
end
global CB
global tcb

n_start=input('value of n to start? ');
for n=n_start:size(manual_insects,2)

    figone=figure(1);
    datacursormode on
```

load data and calculate positive data

```
    data=load(fullfile(filename,fols(manual_insects(n).x).name));
    fn=fieldnames(data);
    positive_data=abs(data.(fn{:})
(manual_insects(n).y).normalized_data(manual_insects(n).z,:)...
        -max(data.(fn{:})
(manual_insects(n).y).normalized_data(manual_insects(n).z,:)));
```

find exact time of insect incident

```
    dateinfo_dir=[start_file,'\Stored Data\',date];
    dateinfo=dir(fullfile(dateinfo_dir,'AMK*'));
    dateinfo=dateinfo([dateinfo.isdir]);
    for
 k=1:size(dateinfo,1);run_time(k,:)=datenum(dateinfo(k).name(11:end),'HHMMSS');end
    mode_run_time=mode(diff(run_time));

    exact_dnum=datenum([date,'-',num2str(str2num(data.(fn{:})
(manual_insects(n).y).filename(11:16))+...
        str2num(datestr(mode_run_time*(str2num(data.(fn{:})
(manual_insects(n).y).filename(18:22))/230),'HHMMSS')))],...
        'yyyy-mm-dd-HHMMSS');

    exact_time=datestr(exact_dnum,'yyyy-mm-dd HH:MM:SS');
```

plot

```
    subplot(2,2,1)
    plot(positive_data)
    title(sprintf('Insect at AMK Ranch on %s\n at a distance of %2.1f
 m',...
        exact_time, data.(fn{:})
(manual_insects(n).y).range(manual_insects(n).z)))
    xlabel('pulse number')
    ylabel('normalized detected signal (arb)')
    set(figone,'outerposition',[1043 515 1696 1026])
```

```
    %     multiple_insects=input('are there multiple insects (yes (c)/
no (v))? ','s');


    lower_bound=input('lower bound of insect event ');
    upper_bound=input('upper bound of insect event ');
    if lower_bound==0
        inb=inb+1;

 manual_final_insect.multiple_insects(inb).x=manual_insects(n).x;

 manual_final_insect.multiple_insects(inb).y=manual_insects(n).y;

 manual_final_insect.multiple_insects(inb).z=manual_insects(n).z;
    else

        subplot(2,2,1)
        tcb.plt=plot(data.(fn{:})
(manual_insects(n).y).time,positive_data,'b',...
            data.(fn{:})(manual_insects(n).y).time,...
            ones(size(data.(fn{:})
(manual_insects(n).y).time)).*max(positive_data),'k:');
        title(sprintf('Insect at AMK Ranch on %s\n at a distance of
 %2.1f m',...
            exact_time, data.(fn{:})
(manual_insects(n).y).range(manual_insects(n).z)))
        xlabel('time (s)')
        ylabel('normalized detected signal (arb)')
```

do guass

```
        wind=31;
        pks=input('number of peaks ');
        g=gaussmf(1:length(positive_data),[(upper_bound-
lower_bound)/2,mean([upper_bound lower_bound])]);
        np=g.*positive_data;
        gtmaxv=movmax(np,wind);
        gtminv=movmin(np,wind);

        sg=unique(sort(gtmaxv));
        fsg=sg(end-pks:end);
        int=0;
        for f=fsg
            int=int+1;
            loc(int)=find(np==f);
        end
        pk_f_freq=abs(1/mean(diff(data.(fn{:})
(manual_insects(n).y).time(loc))));
```

do fft on positive_data

```
        nop=size(positive_data,2);
        delta_t=mean(diff(data.(fn{:})(manual_insects(n).y).time));
```

```
        max_freq=1./(2*delta_t);
        delta_f=1/data.(fn{:})(manual_insects(n).y).time(end);
        fqdata=(-nop/2:nop/2-1).*delta_f;
        freq_data=fftshift(fft(positive_data,nop,2),2);
        [maxv,maxi]=findpeaks(abs(freq_data(513:end)).^2);
        [tminv,mini]=findpeaks(1.01*max(abs(freq_data(513:end)).^2)-
abs(freq_data(513:end)).^2);
        subplot(2,2,2)
        plot(fqdata,
(abs(freq_data)).^2,'b',fqdata(512+maxi),maxv,'ro')
        xlabel('frequency (Hz)')
        ylabel('power (arb)')
        title(sprintf('Frequency spectrum using
 FFT algorithm of\ninsect at AMK Ranch on %s\nat
 a distance of %2.1f m', exact_time,data.(fn{:})
(manual_insects(n).y).range(manual_insects(n).z)))
        xlim([0 2200])
        ylim([0
 max((abs(freq_data(517:end)).^2))+max((abs(freq_data(517:end)).^2))*.05])
```

do fft on gauss_data

```
        sl_coef=.2;
        CB.gfreq_data=fftshift(fft(np,nop,2),2);
        CB.np=np;
        CB.nop=nop;
        CB.fqdata=fqdata;
        [gmaxv,gmaxi]=findpeaks(abs(CB.gfreq_data(513:end)).^2);

 gmovmx=movmax((abs(CB.gfreq_data(513:end))).^2,round(sl_coef*mean(diff(gmaxi(2:en
        [umx,igmxnf,igvx]=intersect(gmaxv,gmovmx);
        [igmx,indx_igmx]=sort(igmxnf,'ascend');
        slmin=.010;
        slmax=20;

        subplot(2,2,4);

        CB.gfax=plot(CB.fqdata,
(abs(CB.gfreq_data)).^2,'b',CB.fqdata(512+gmaxi(igmx)),umx(indx_igmx),'m>'); %,CB.

 uicontrol('Style','slider','min',slmin,'Max',slmax,'SliderStep',
 [.01 .1]./(slmax-slmin),'Value',sl_coef,'Position',[1150 20 200
 20],'Callback',@SliderCB);

        xlabel('frequency (Hz)')
        ylabel('power (arb)')
        title(sprintf('Windowed - AMK Ranch on %s
\nat a distance of %2.1f m', exact_time,data.(fn{:})
(manual_insects(n).y).range(manual_insects(n).z)))
        xlim([0 2200])
        ylim([0
 max((abs(CB.gfreq_data(518:end)).^2))+max((abs(CB.gfreq_data(518:end)).^2))*.05])
        max_peak=CB.fqdata(512+gmaxi(find(gmaxv==max(gmaxv))));
```

peaks

```
        clear mxval mxtime
        wnd_ts_th=.2;
        freq_space=1/max_peak;
        freq_val=find(data.(fn{:})
(manual_insects(n).y).time<=freq_space/2,1,'last');
        tcb.freq_val=freq_val;


 cur_loc=find(positive_data(1:upper_bound)>=wnd_ts_th*max(positive_data),1,'first'

        inu=0;
        set(tcb.plt(2),'ydata',ones(size(data.(fn{:})
(manual_insects(n).y).time)).*max(positive_data).*wnd_ts_th)

        while cur_loc<=upper_bound-freq_val
            inu=inu+1;
            if cur_loc<=freq_val
                mxval(inu)=max(np(cur_loc:cur_loc+freq_val));
            else
                mxval(inu)=max(np(cur_loc-freq_val:cur_loc+freq_val));
            end
            cur_loc=cur_loc+freq_val*2+1;
            mxtime(inu)=data.(fn{:})
(manual_insects(n).y).time(find(np==mxval(inu)));
        end
        tlmin=.01;
        tlmax=1;
        tcb.positive_data=positive_data;
        tcb.upper_bound=upper_bound;
        tcb.lower_bound=lower_bound;
        tcb.np=np;

        tcb.data=data;
        tcb.fn=fn;
        tcb.manual_insects=manual_insects;

        tcb.n=n;
        subplot(2,2,3)
        tcb.tax=plot(data.(fn{:})(manual_insects(n).y).time,np,data.
(fn{:})(manual_insects(n).y).time,np,'b',...
            data.(fn{:})(manual_insects(n).y).time,np,data.(fn{:})
(manual_insects(n).y).time,np,'r.',...
            mxtime,mxval,'ko');
        title(sprintf('Windowed - AMK Ranch on %s\n at a distance of
 %2.1f m',...
            exact_time, data.(fn{:})
(manual_insects(n).y).range(manual_insects(n).z)))
        xlim([data.(fn{:})(manual_insects(n).y).time(lower_bound)...
            data.(fn{:})(manual_insects(n).y).time(upper_bound)])
        xlabel('time (s)')
        ylabel('normalized detected signal (arb)')
```

```
uicontrol('Style','slider','min',tlmin,'Max',tlmax,'SliderStep',
[.001 .01]./(slmax-slmin),'Value',wnd_ts_th,'Position',[400 20 200
20],'Callback',@SliderTCB);
       set(figone,'outerposition',[1043 515 1696 1026])

       peak=input('which peaks should be isolated? ');
       pk_freq=1/mean(diff(tcb.mxtime));

mean_pksnz=mean(diff([CB.fqdata(512+CB.gmaxi(CB.igmx(1:peak)))]));

std_pksnz=std(diff([CB.fqdata(512+CB.gmaxi(CB.igmx(1:peak)))]));
       mean_peaks=mean(diff([0
CB.fqdata(512+CB.gmaxi(CB.igmx(1:peak)))]));
       std_peaks=std(diff([0
CB.fqdata(512+CB.gmaxi(CB.igmx(1:peak)))]));
       sprintf('mean nonz frequency peaks: %4.2f Hz\nstd no ze
frequency peaks: %4.2f Hz\nmean btwn frequency peaks: %4.2f Hz\nstd
betwn frequency peaks: %4.2f Hz\nmaximum frequency peak is: %4.2f
Hz\nmean freq some time peaks: %4.2f Hz\nmean freq from time peaks:
%4.2f Hz\n%s at a distance of %2.1f m',...

mean_pksnz,std_pksnz,mean_peaks,std_peaks,max_peak,pk_freq,pk_f_freq,exact_time,d
(fn{:})(manual_insects(n).y).range(manual_insects(n).z))
```

rateing

```
       rate=input('rate from not likly to likely (z->x->c->v)','s');
       if strcmp(rate,'z')==1
           inz=inz+1;

manual_final_insect.very_unlikely(inz).x=manual_insects(n).x;

manual_final_insect.very_unlikely(inz).y=manual_insects(n).y;

manual_final_insect.very_unlikely(inz).z=manual_insects(n).z;


manual_final_insect.very_unlikely(inz).f_d_mean_pks=mean_peaks;

manual_final_insect.very_unlikely(inz).f_d_std_pks=std_peaks;
           manual_final_insect.very_unlikely(inz).f_max_pk=max_peak;
           manual_final_insect.very_unlikely(inz).t_s_pks=pk_freq;

manual_final_insect.very_unlikely(inz).t_all_pks=pk_f_freq;
       elseif strcmp(rate,'x')==1
           inx=inx+1;

manual_final_insect.somewhat_unlikely(inx).x=manual_insects(n).x;

manual_final_insect.somewhat_unlikely(inx).y=manual_insects(n).y;

manual_final_insect.somewhat_unlikely(inx).z=manual_insects(n).z;
```

```
manual_final_insect.somewhat_unlikely(inx).f_d_mean_pks=mean_peaks;

manual_final_insect.somewhat_unlikely(inx).f_d_std_pks=std_peaks;

manual_final_insect.somewhat_unlikely(inx).f_max_pk=max_peak;

manual_final_insect.somewhat_unlikely(inx).t_s_pks=pk_freq;

manual_final_insect.somewhat_unlikely(inx).t_all_pks=pk_f_freq;
       elseif strcmp(rate,'c')==1
           inc=inc+1;

manual_final_insect.somewhat_likely(inc).x=manual_insects(n).x;

manual_final_insect.somewhat_likely(inc).y=manual_insects(n).y;

manual_final_insect.somewhat_likely(inc).z=manual_insects(n).z;


manual_final_insect.somewhat_likely(inc).f_d_mean_pks=mean_peaks;

manual_final_insect.somewhat_likely(inc).f_d_std_pks=std_peaks;

manual_final_insect.somewhat_likely(inc).f_max_pk=max_peak;
           manual_final_insect.somewhat_likely(inc).t_s_pks=pk_freq;

manual_final_insect.somewhat_likely(inc).t_all_pks=pk_f_freq;
       elseif strcmp(rate,'v')==1
           inv=inv+1;

manual_final_insect.very_likely(inv).x=manual_insects(n).x;

manual_final_insect.very_likely(inv).y=manual_insects(n).y;

manual_final_insect.very_likely(inv).z=manual_insects(n).z;


manual_final_insect.very_likely(inv).f_d_mean_pks=mean_peaks;

manual_final_insect.very_likely(inv).f_d_std_pks=std_peaks;
           manual_final_insect.very_likely(inv).f_max_pk=max_peak;
           manual_final_insect.very_likely(inv).t_s_pks=pk_freq;
           manual_final_insect.very_likely(inv).t_all_pks=pk_f_freq;
       else
           inb=inb+1;

manual_final_insect.multiple_insects(inb).x=manual_insects(n).x;

manual_final_insect.multiple_insects(inb).y=manual_insects(n).y;

manual_final_insect.multiple_insects(inb).z=manual_insects(n).z;
       end
```

```
    end
    close all
    disp(manual_final_insect)

 save(fullfile(filename,'events','manual_final_insect.mat'),'manual_final_insect',
v7.3')
    disp(n)

end
function []=SliderCB(source,event)
global CB
sl_coef=source.Value;
CB.gfreq_data=fftshift(fft(CB.np,CB.nop,2),2);
[gmaxv,gmaxi]=findpeaks(abs(CB.gfreq_data(513:end)).^2);
rndval=round(sl_coef*mean(diff(gmaxi(2:end))));
gmovmx=movmax((abs(CB.gfreq_data(513:end))).^2,rndval);
[umx,igmxnf,igvx]=intersect(gmaxv,gmovmx);
[igmx,indx_igmx]=sort(igmxnf,'ascend');
set(CB.gfax(2),'ydata',umx(indx_igmx))
set(CB.gfax(2),'xdata',CB.fqdata(512+gmaxi(igmx)))
CB.gmaxv=gmaxv;
CB.gmaxi=gmaxi;
CB.igmx=igmx;

function []=SliderTCB(source,event)
global tcb
wnd_ts_th=source.Value;
cur_loc=find(tcb.positive_data(1:tcb.upper_bound)>=wnd_ts_th*max(tcb.positive_data
inu=0;
while cur_loc<=tcb.upper_bound-tcb.freq_val
    if cur_loc<=tcb.freq_val
        inu=inu+1;
        mxval(inu)=max(tcb.np(cur_loc:cur_loc+tcb.freq_val));
    elseif cur_loc==tcb.upper_bound
        inu=inu+1;
        mxva(inu)=max(tcb.np(cur_loc-tcb.freq_val));
    else
        inu=inu+1;
        mxval(inu)=max(tcb.np(cur_loc-tcb.freq_val:cur_loc
+tcb.freq_val));
    end
    cur_loc=cur_loc+tcb.freq_val*2+1;
    if exist('mxval','var')
        mxtime(inu)=tcb.data.(tcb.fn{:})
(tcb.manual_insects(tcb.n).y).time(find(tcb.np==mxval(inu)));
    end

end
try
    set(tcb.tax(5),'ydata',mxval)
    set(tcb.tax(5),'xdata',mxtime)

    set(tcb.plt(2),'ydata',ones(size(tcb.data.(tcb.fn{:})
(tcb.manual_insects(tcb.n).y).time)).*max(tcb.positive_data).*wnd_ts_th)
```

```
    tcb.mxtime=mxtime;
    tcb.mxval=mxval;
    tcb.wnd_ts_th=wnd_ts_th;
catch
    disp('impossible value')
end
```

Published with MATLAB® R2016b

APPENDIX C

SIMULATED FOURIER TRANSFORMS

The following plots are various simulated time-domain signals and their Fourier transforms. The intent is to help the reader understand how differences in the time domain affect the frequency domain. Unlike the plots in Fourier transform section of the Wing-Beat Modulation Lidar Theory chapter, the time-domain plots here have a peak amplitude of 1 instead of 20. This scaling factor has nearly no significance, since it arbitrarily scales the power spectrum.

Figure C.1: noise floor: [0 0]; peak noise: [0 0]; time jitter: [0 0] s; σ jitter [0 0] s

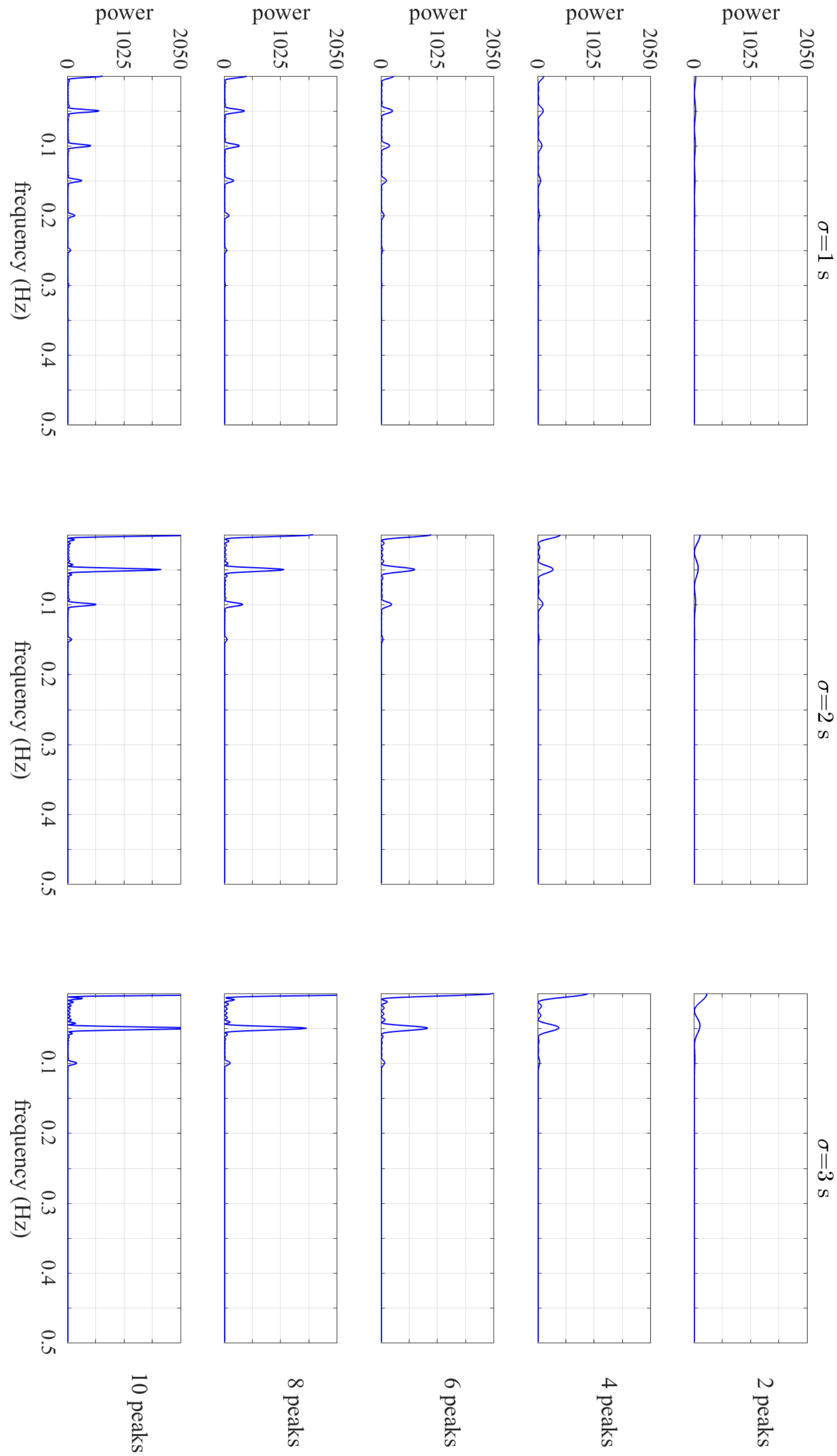


Figure C.2: noise floor: [0 0]; peak noise: [0 0]; time jitter: [0 0] s; σ jitter [0 0] s

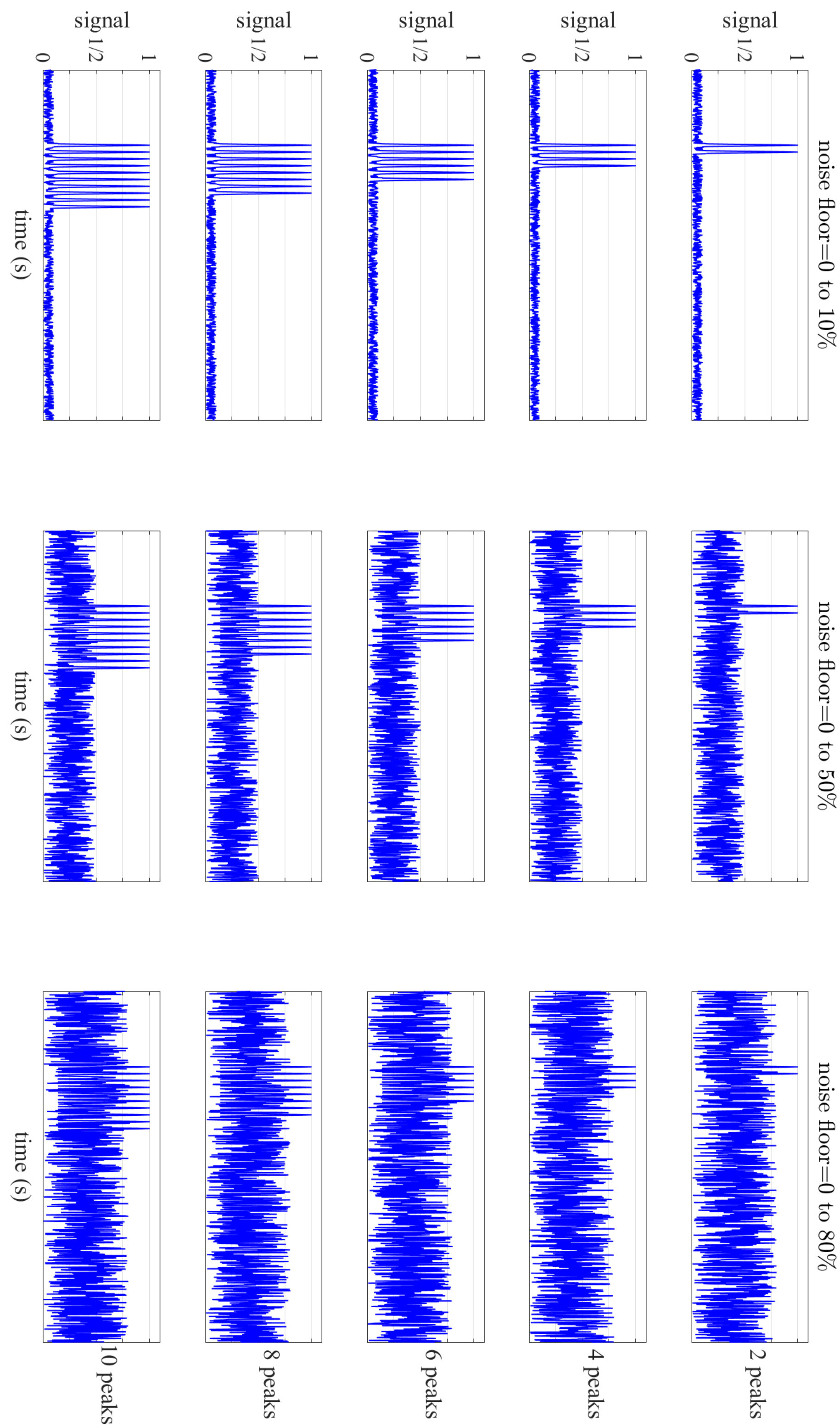


Figure C.3: $\sigma = 2\ s$; peak noise: [0 0]; time jitter: [0 0] s; σ jitter [0 0] s

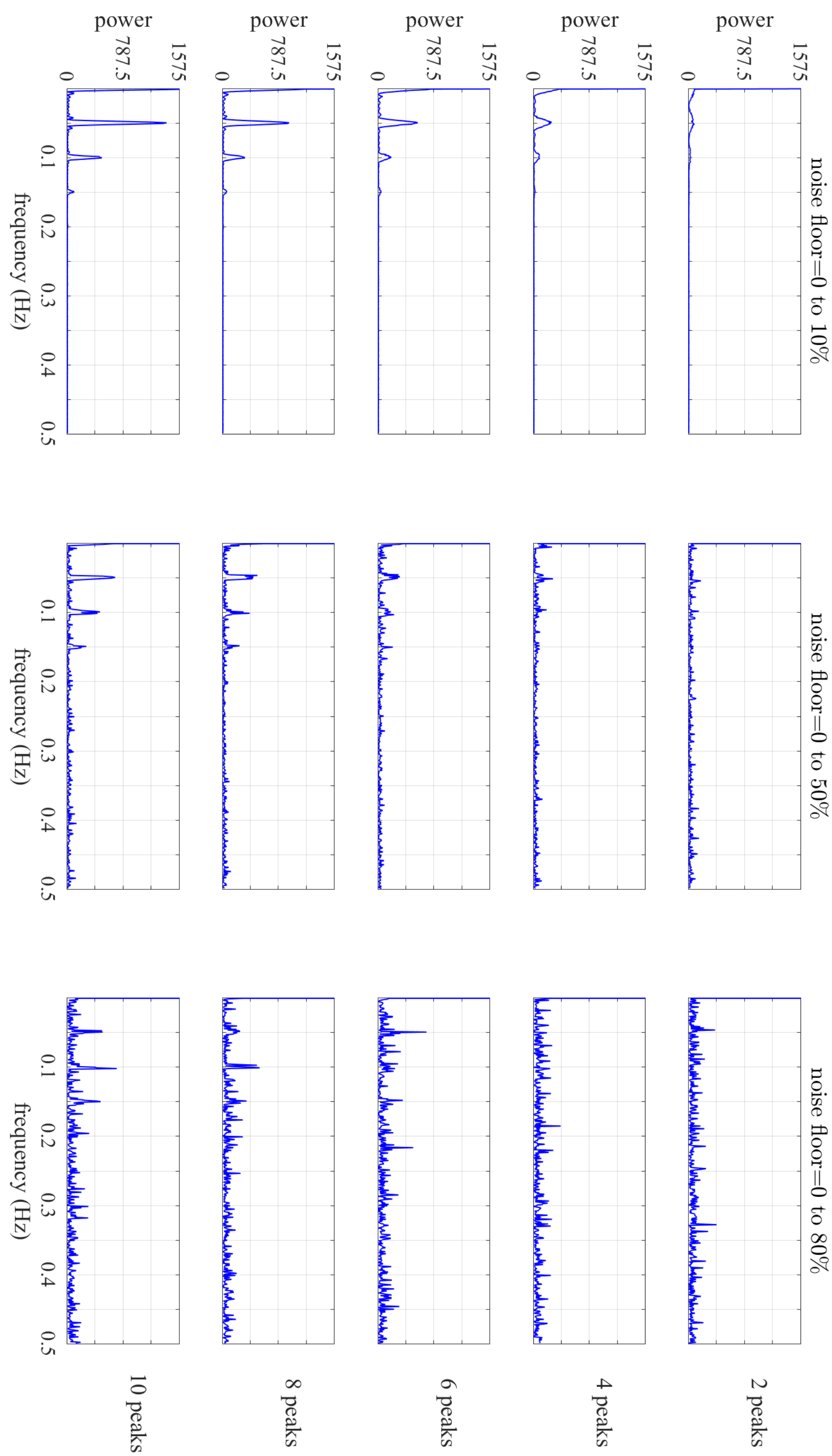


Figure C.4: $\sigma = 2\ s$; peak noise: [0 0]; time jitter: [0 0] s; σ jitter [0 0] s

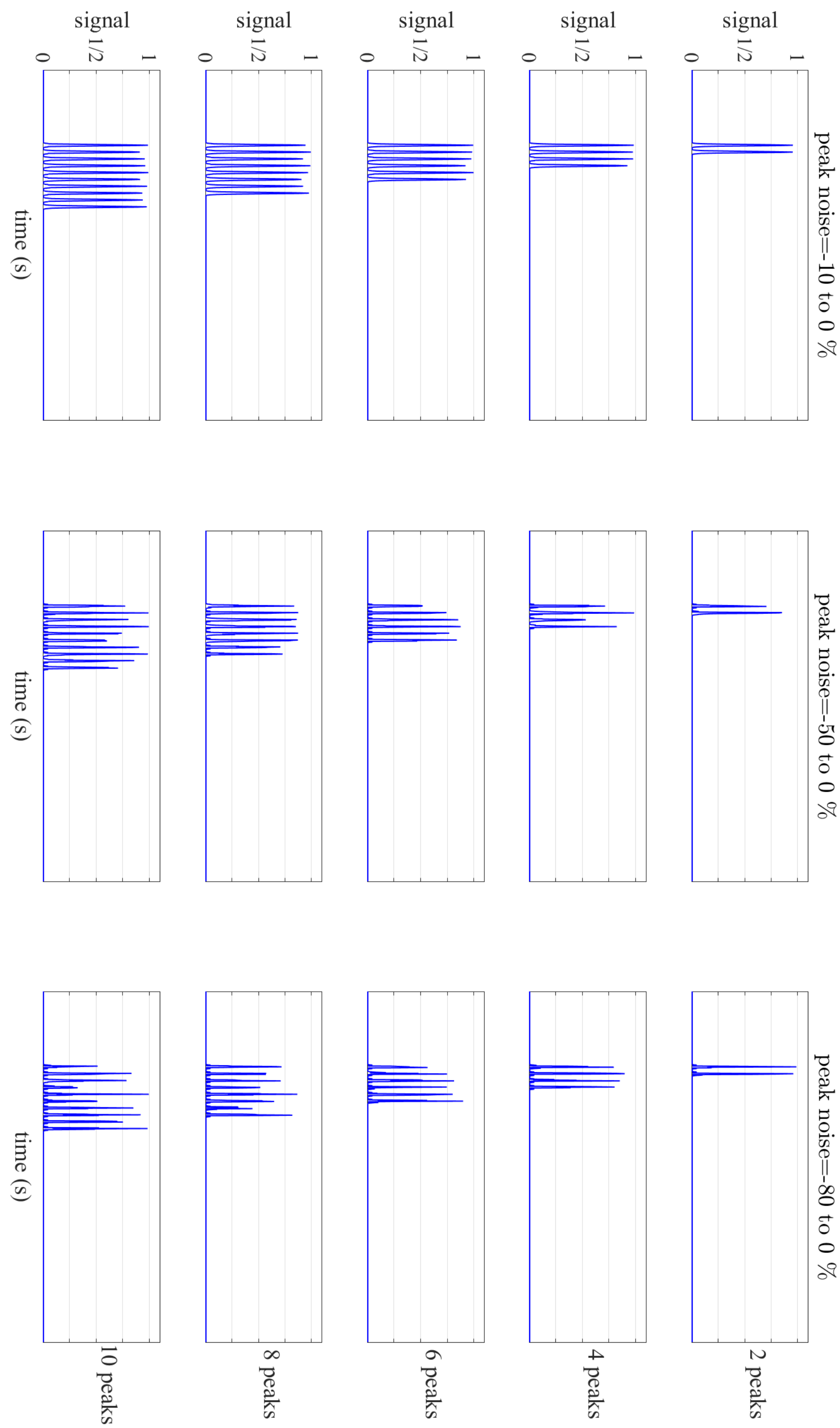


Figure C.5: noise floor [0 0]; $\sigma = 2\ s$; time jitter: [0 0] s; σ jitter [0 0] s

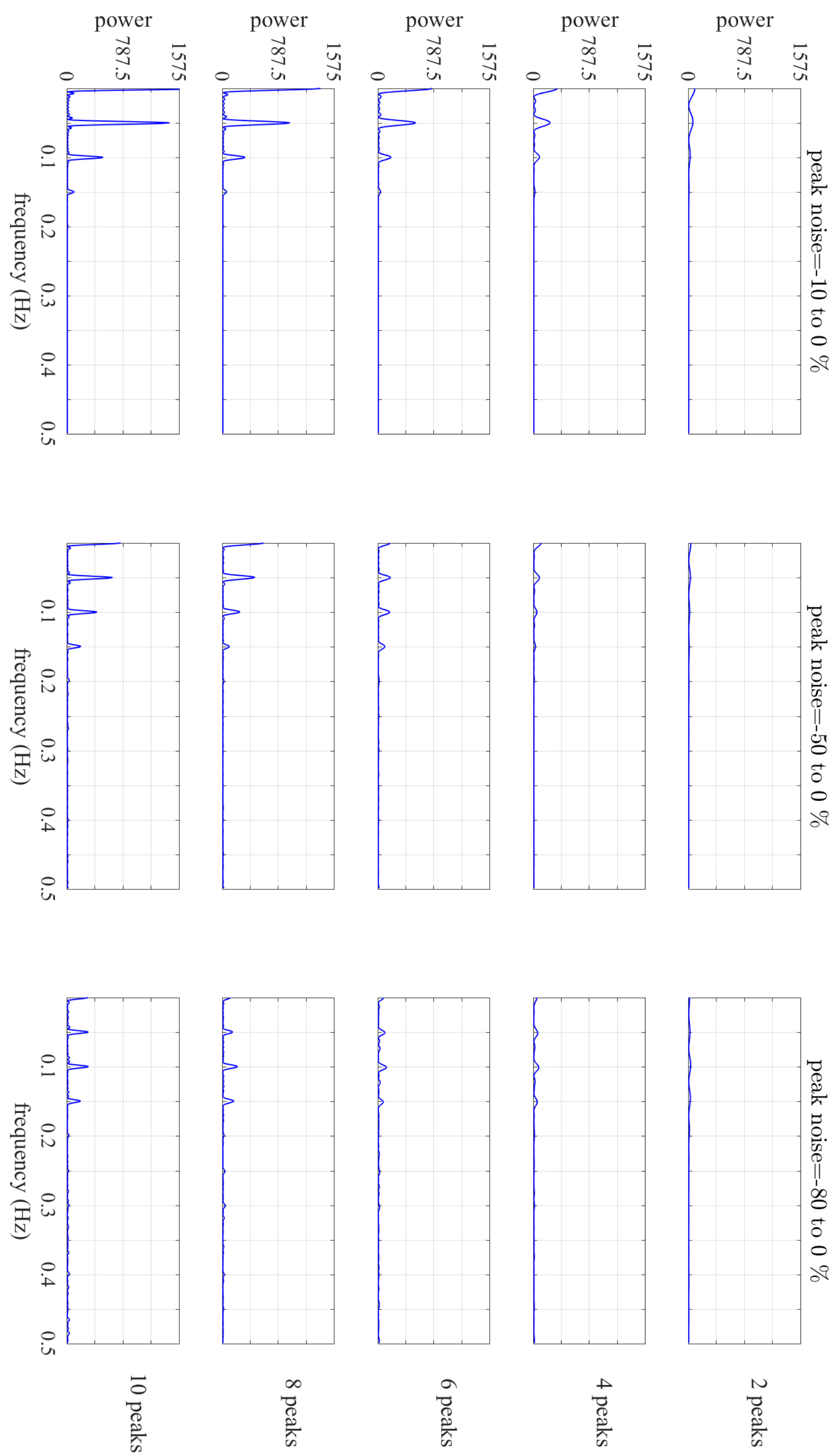


Figure C.6: noise floor: [0 0]; $\sigma = 2$ s; time jitter: [0 0] s; σ jitter [0 0] s

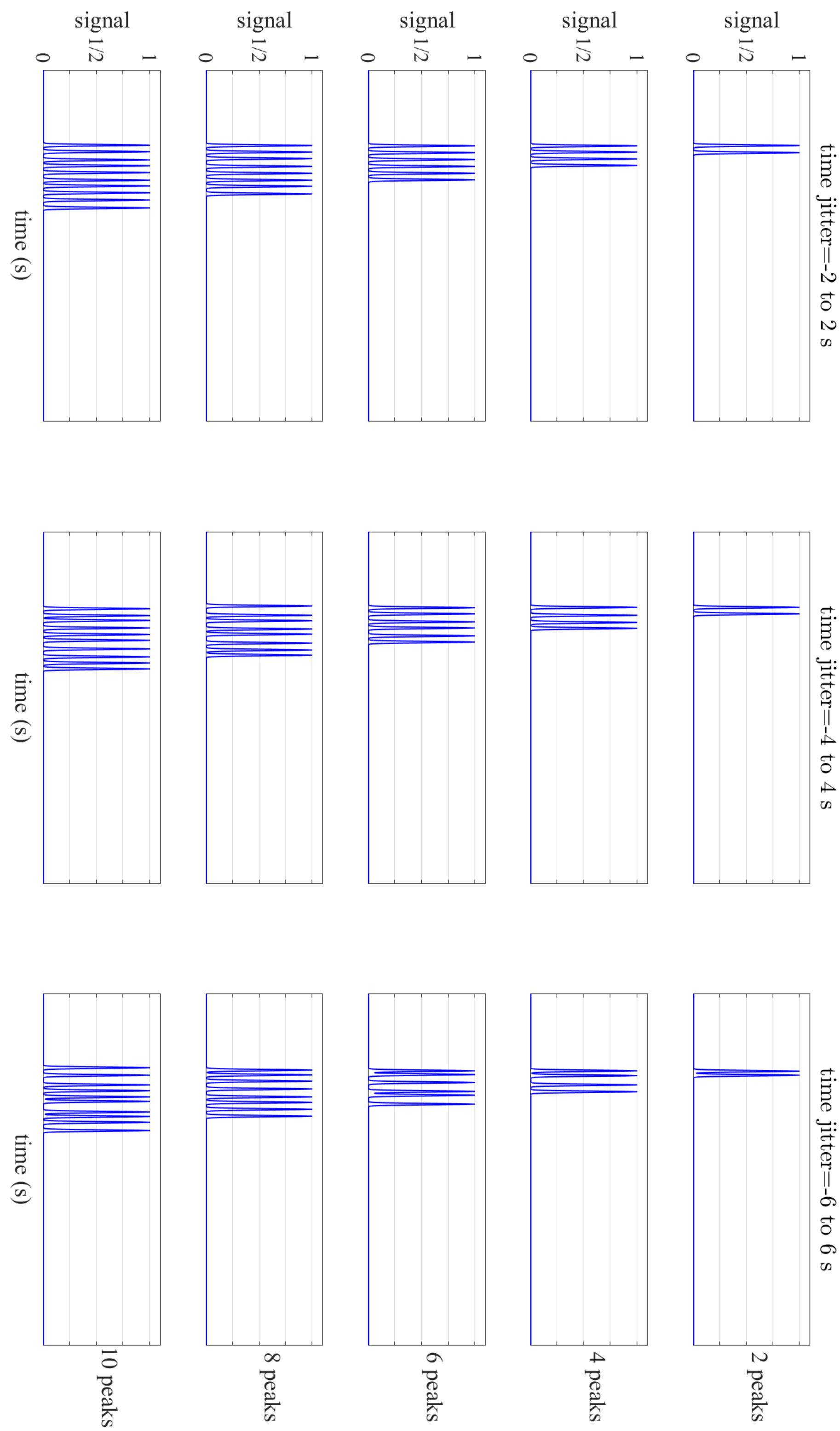


Figure C.7: noise floor [0 0]; peak noise [0 0]; $\sigma = 2\ s$; σ jitter [0 0] s

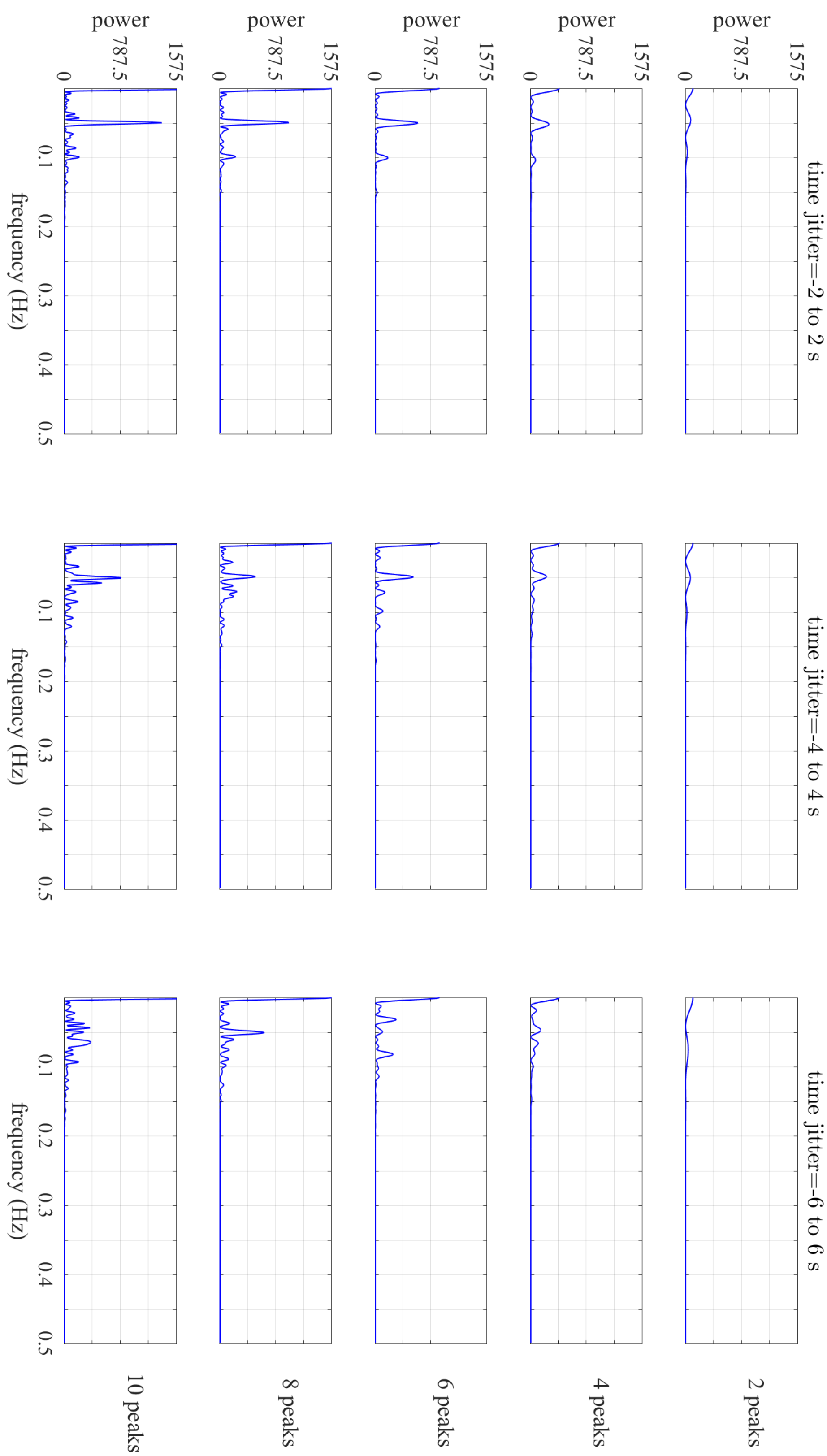


Figure C.8: noise floor: [0 0]; peak noise [0 0]; $\sigma = 2\ s$; σ jitter [0 0] s

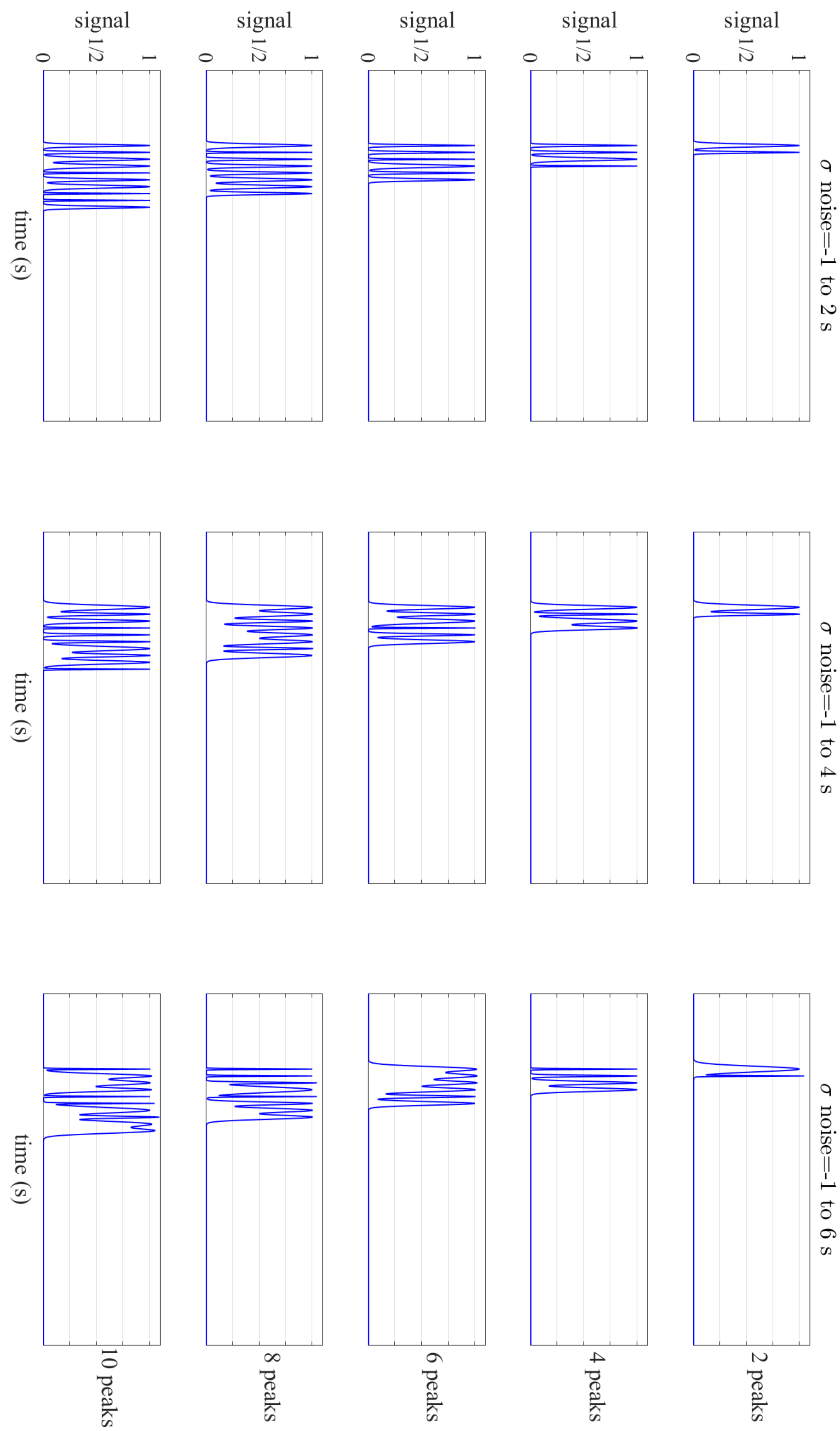


Figure C.9: noise floor [0 0]; peak noise [0 0]; time jitter [0 0] s; $\sigma = 2\ s$

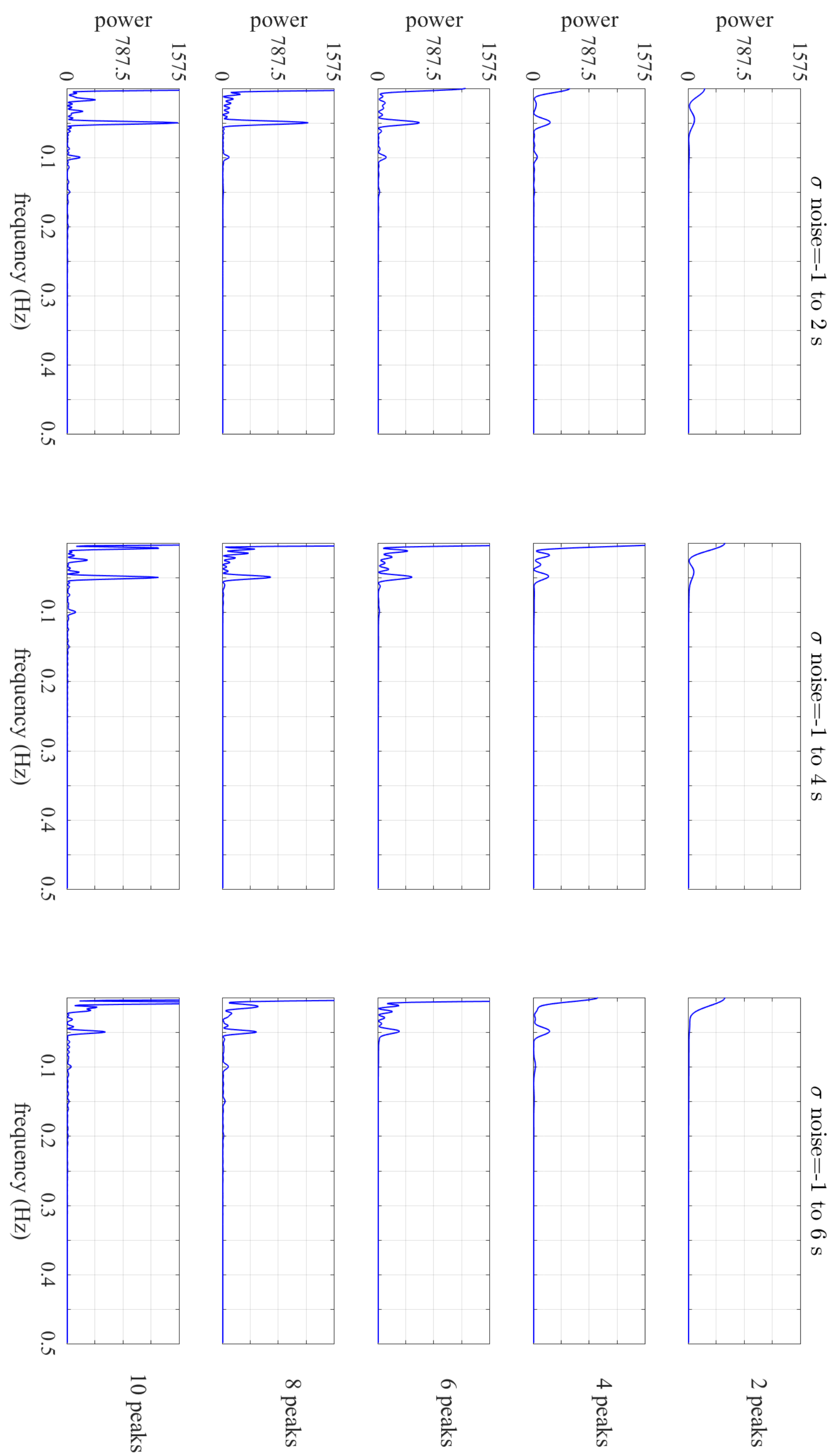


Figure C.10: noise floor: [0 0]; peak noise [0 0]; time jitter [0 0] s; $\sigma = 2\ s$

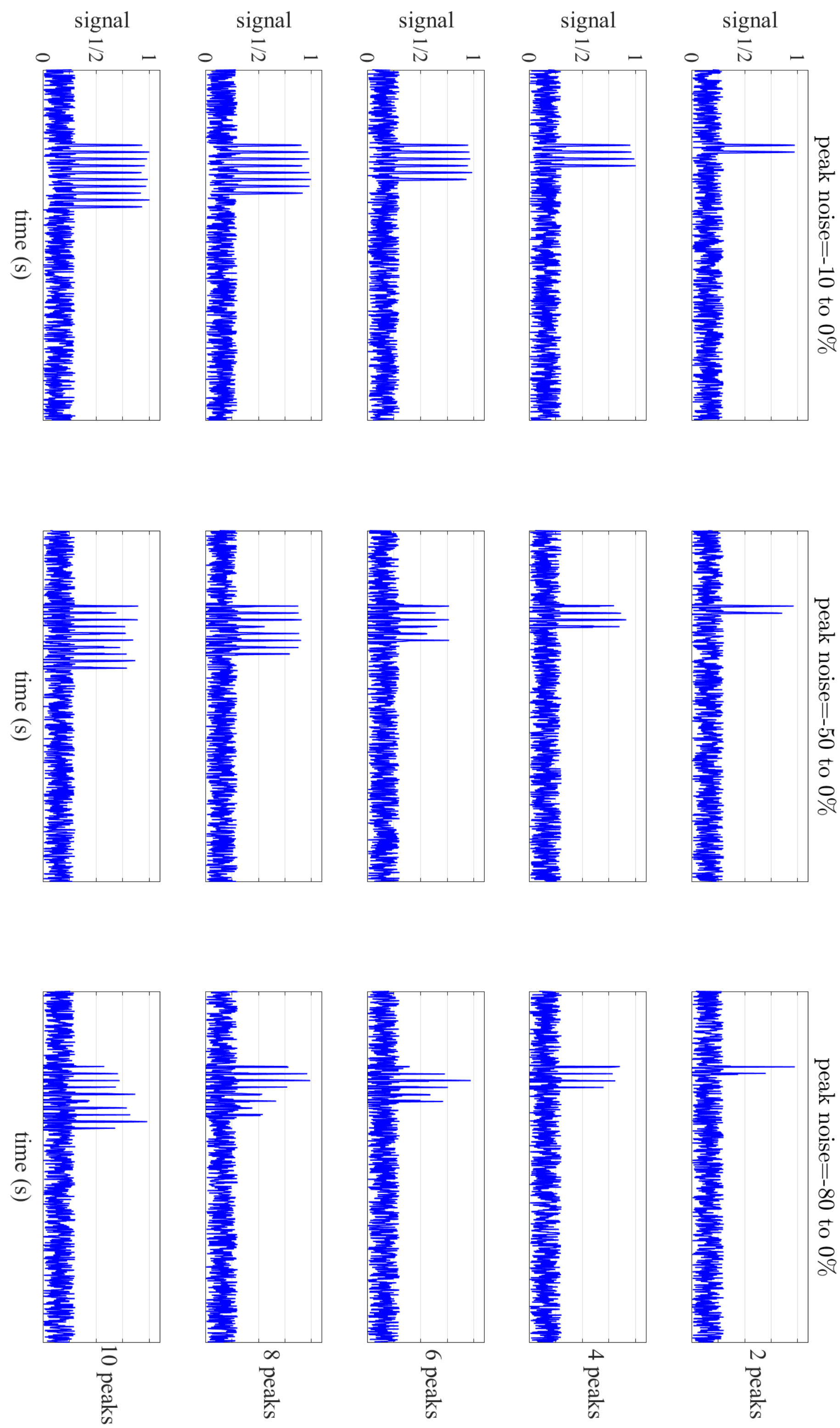


Figure C.11: noise floor [0 30%]; $\sigma = 2$ s; time jitter: [0 0] s; σ jitter [0 0] s

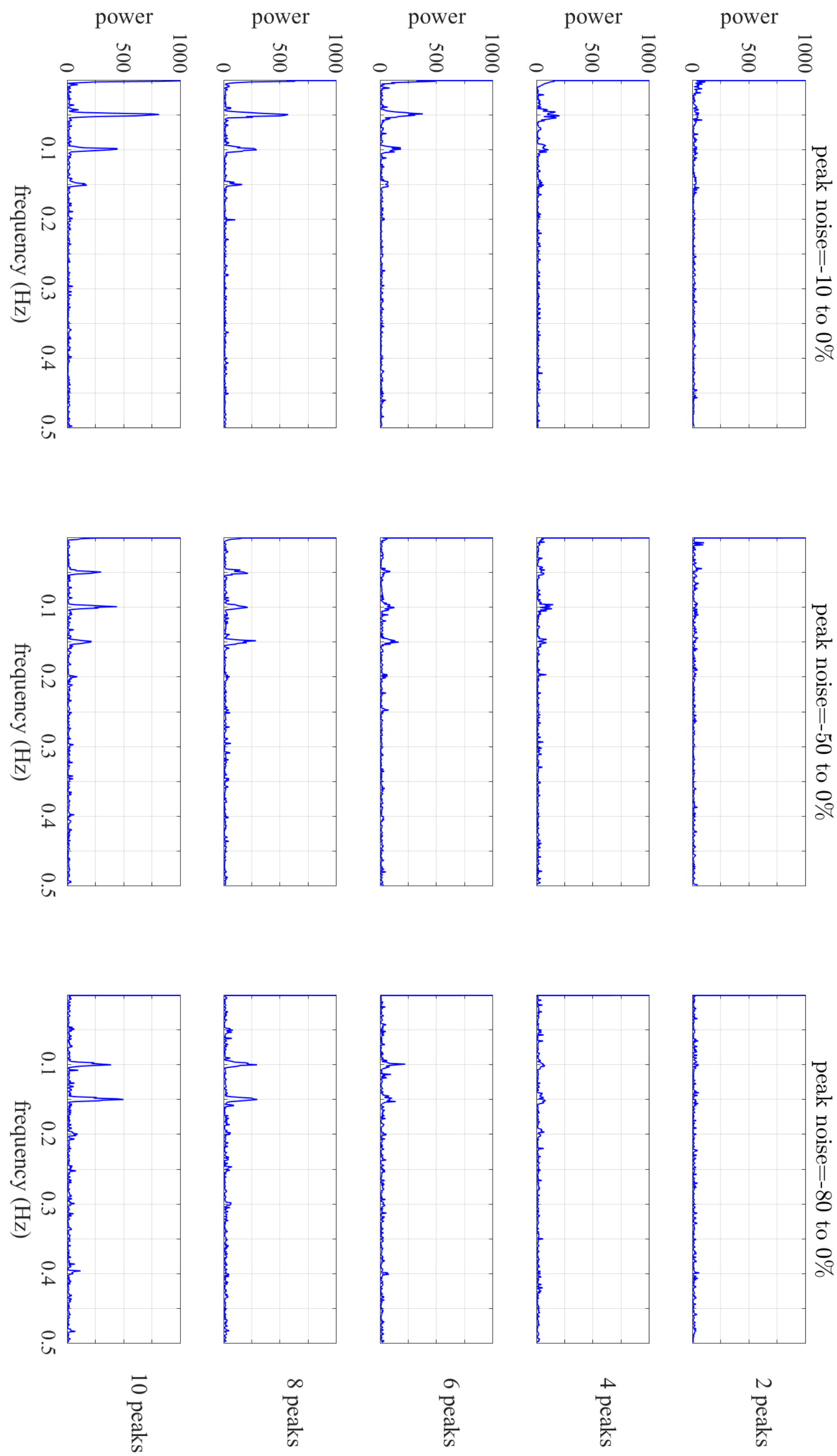


Figure C.12: noise floor [0 30%]; $\sigma = 2$ s; time jitter: [0 0] s; σ jitter [0 0] s

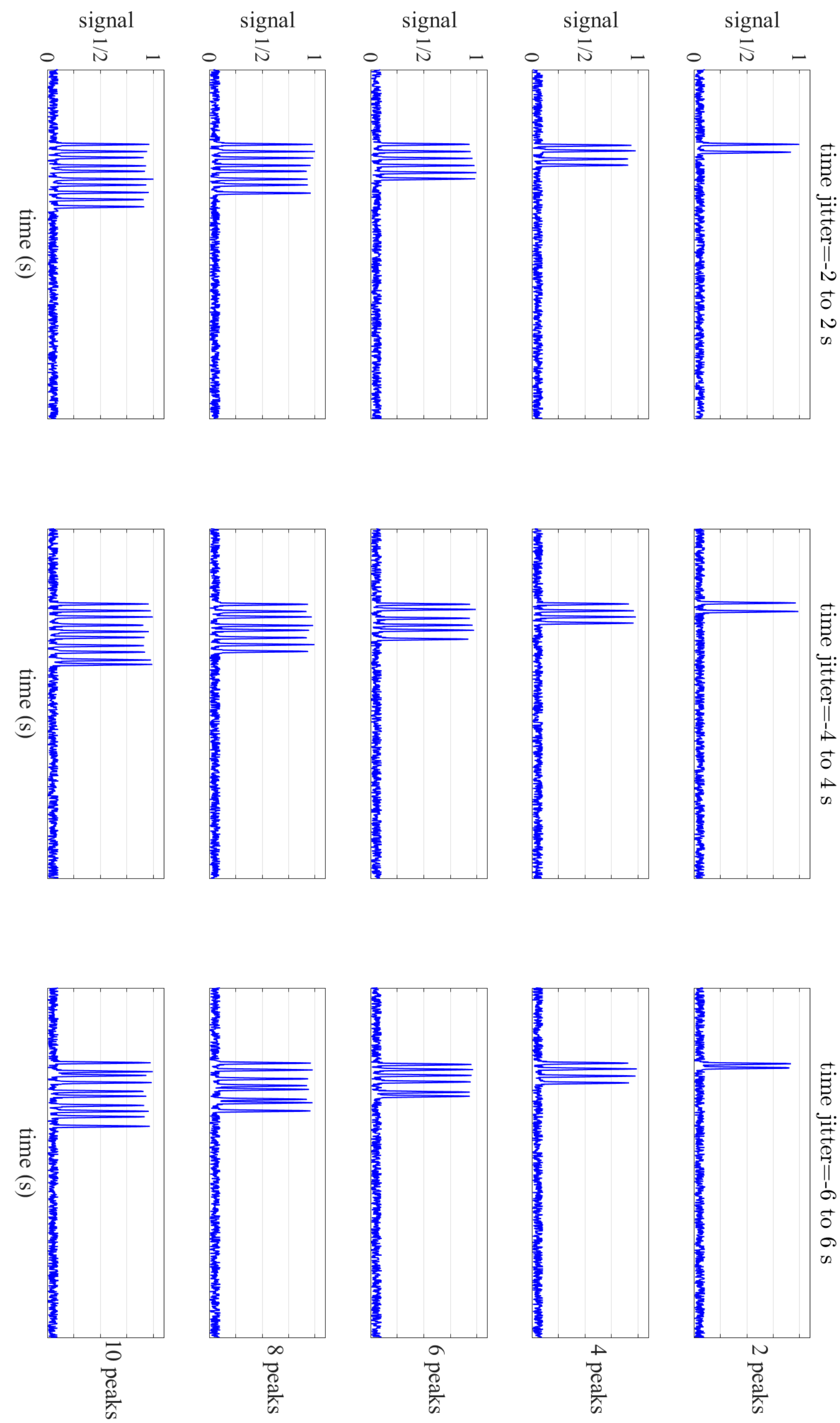


Figure C.13: noise floor [0 10%]; peak noise [-10 0%]; $\sigma = 2\ s$; σ jitter [0 0] s

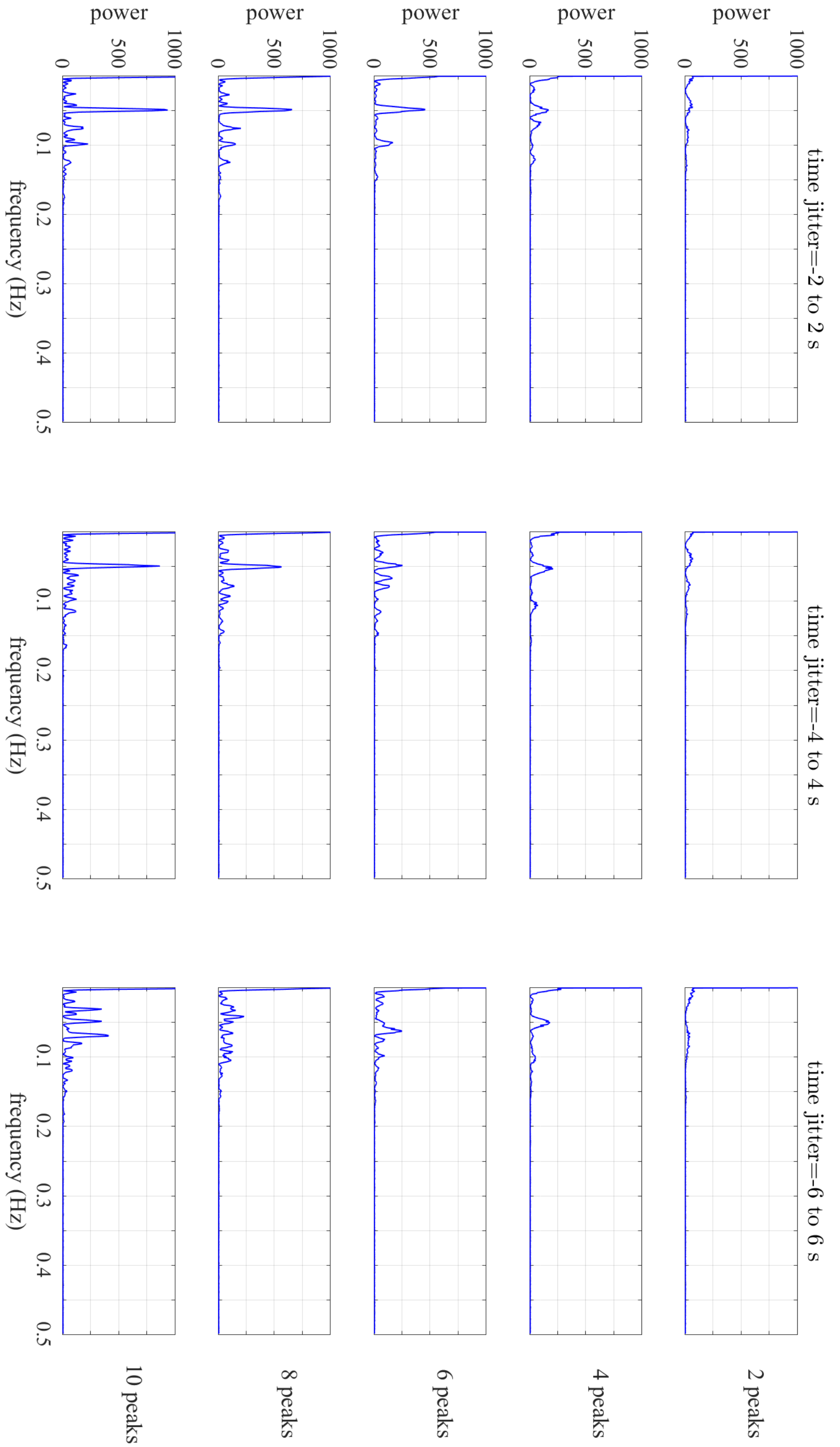


Figure C.14: noise floor [0 10%]; peak noise [-10 0%]; $\sigma = 2\ s$; σ jitter [0 0] s

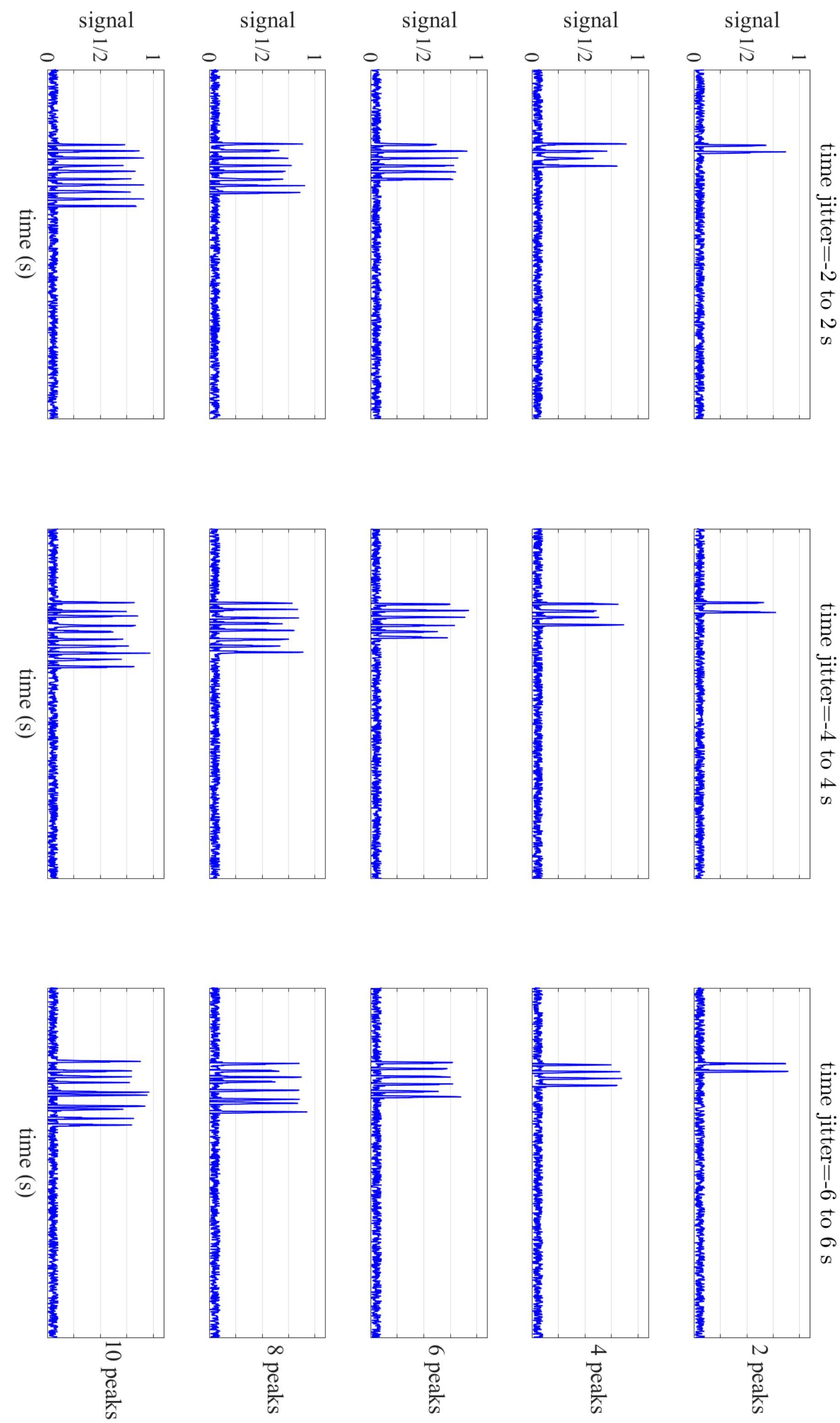


Figure C.15: noise floor [0 10%]; peak noise [-50 0%]; $\sigma = 2$ s; σ jitter [0 0] s

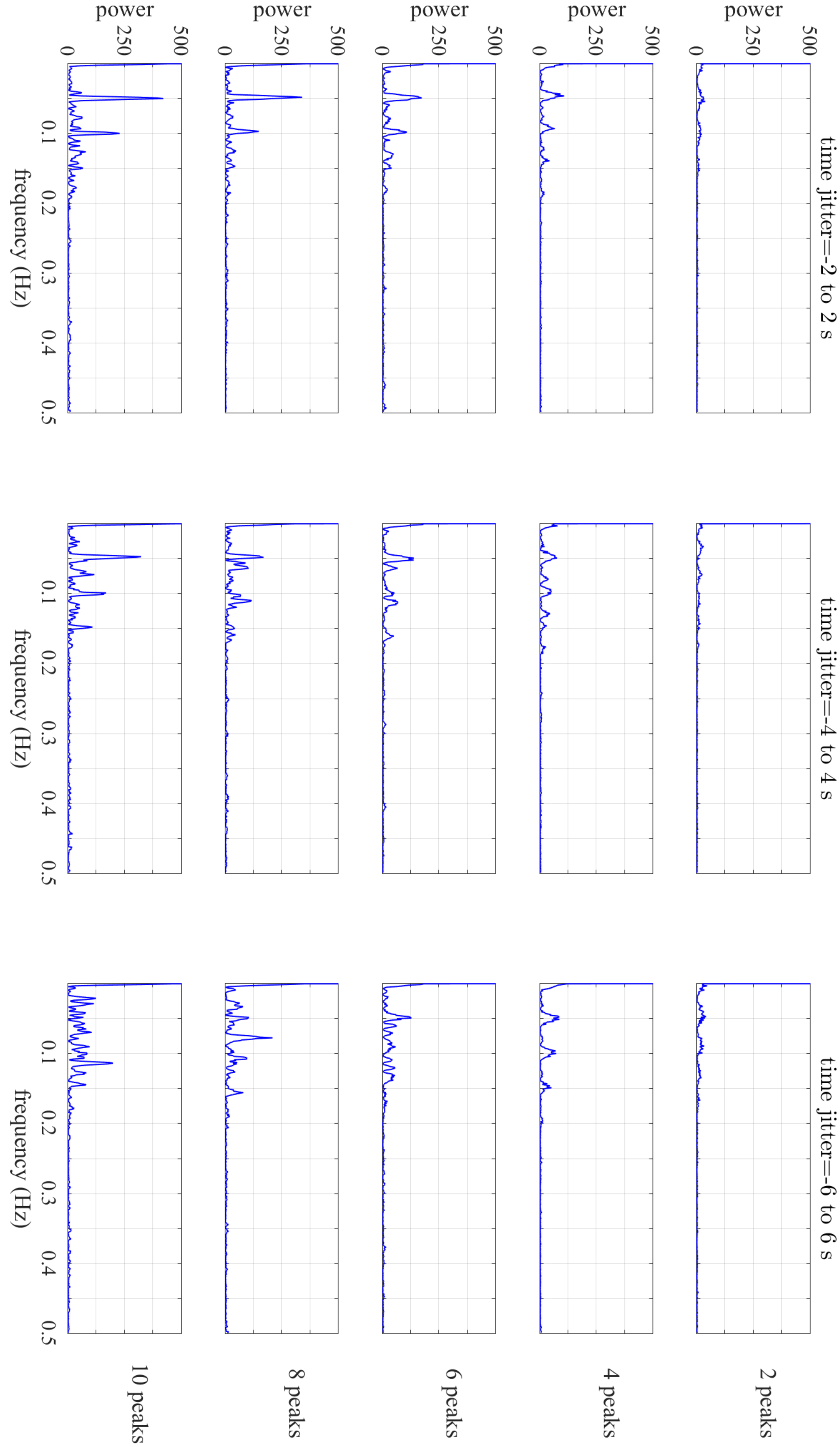


Figure C.16: noise floor [0 10%]; peak noise [-50 0%]; $\sigma = 2\ s$; σ jitter [0 0] s

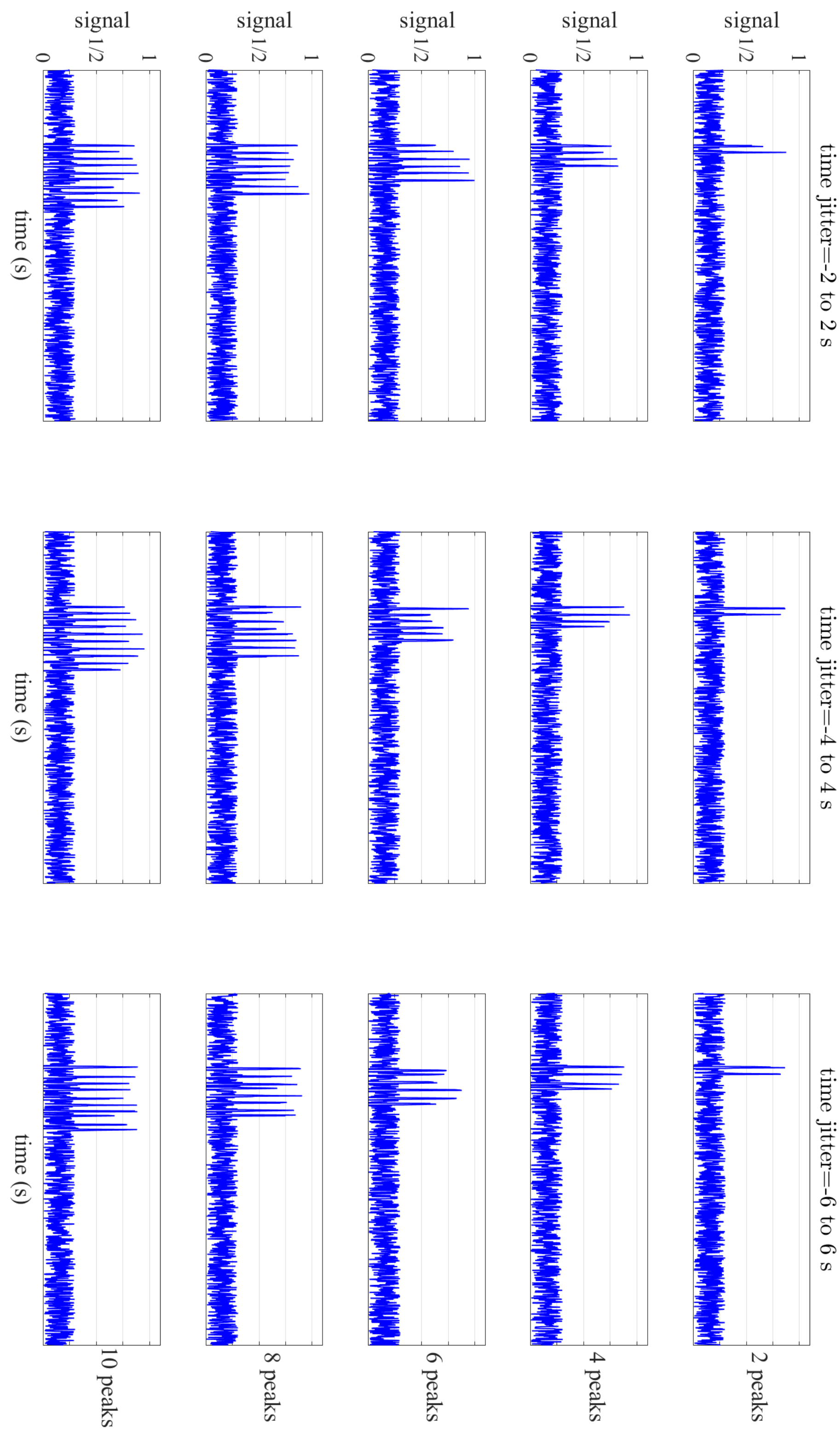


Figure C.17: noise floor [0 30%]; peak noise [-50 0%]; $\sigma = 2$ s; σ jitter [0 0] s

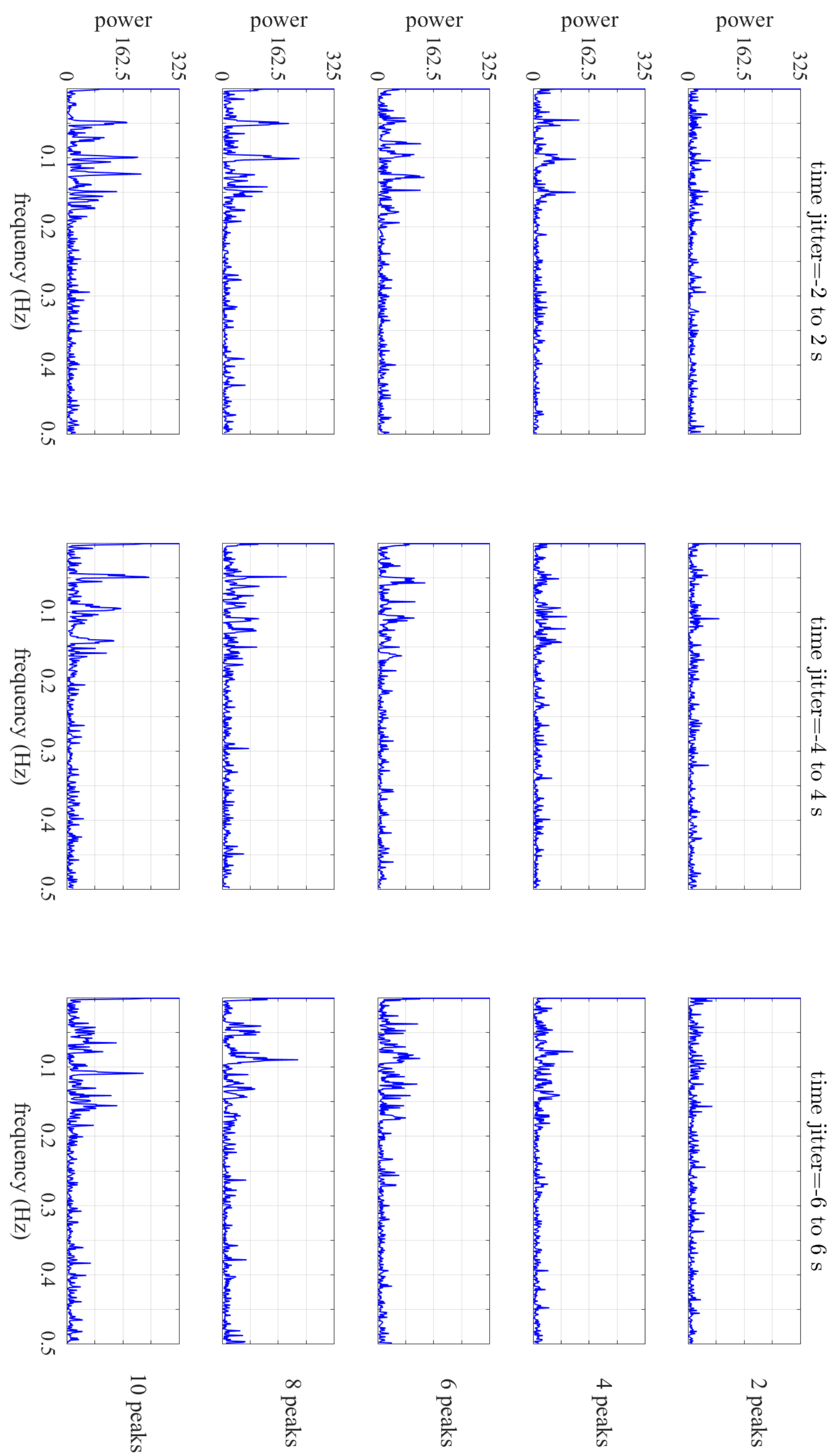


Figure C.18: noise floor [0 30%]; peak noise [-50 0%]; $\sigma = 2\ s$; σ jitter [0 0] s

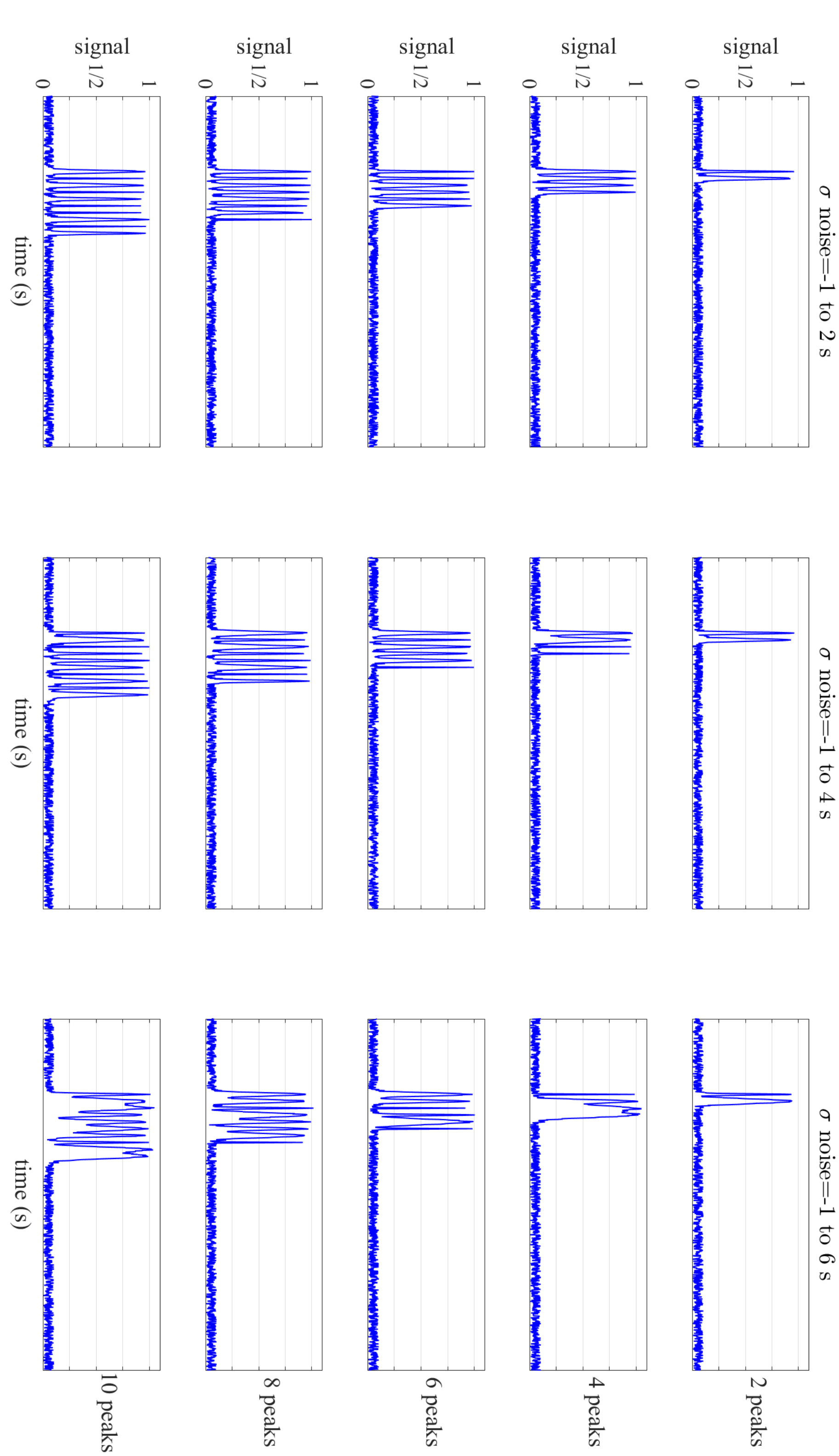


Figure C.19: noise floor [0 10%]; peak noise [-10 0%]; time jitter [0 0] s; $\sigma = 2\ s$

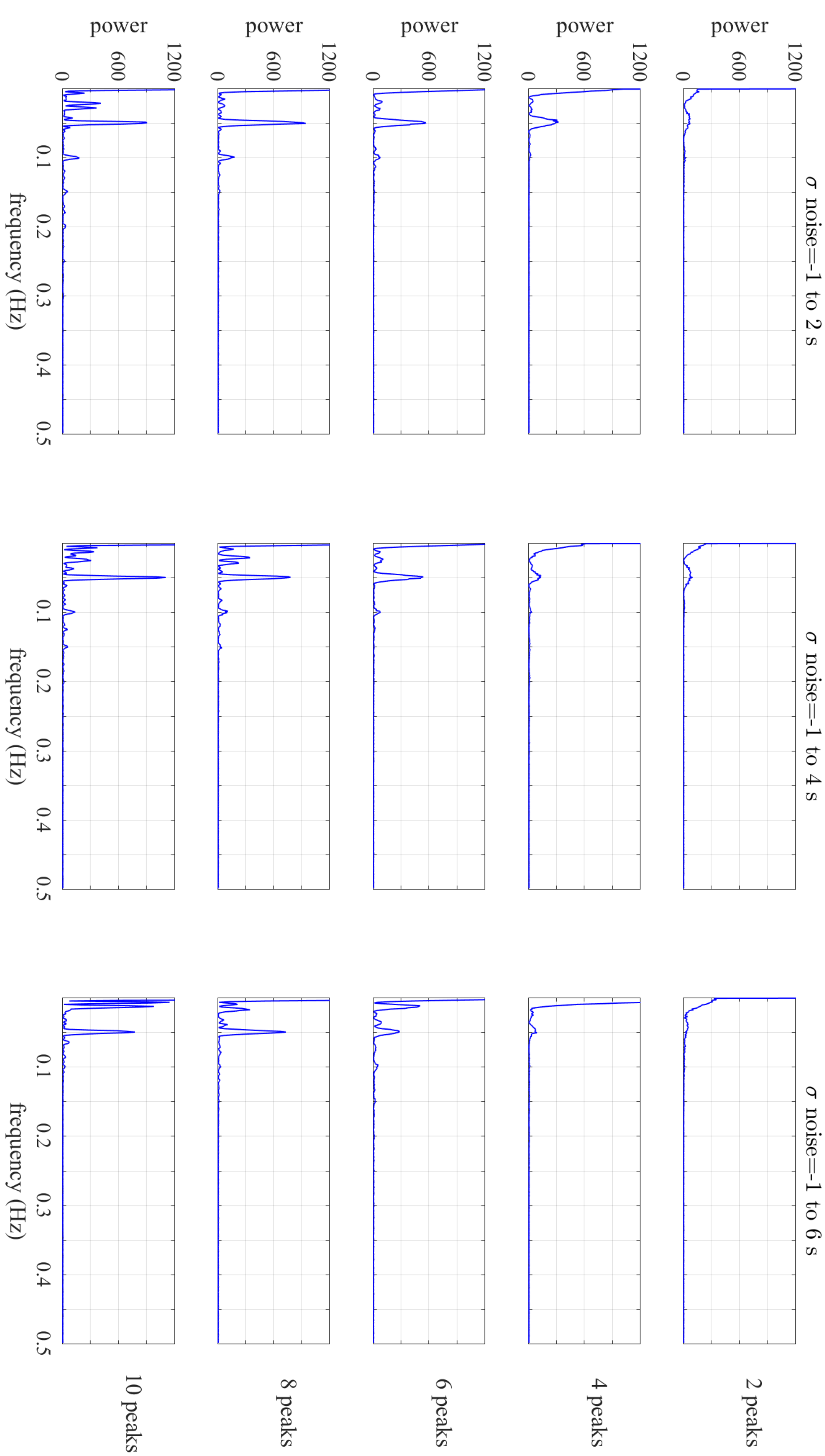


Figure C.20: noise floor [0 10%]; peak noise [-10 0%]; time jitter [0 0] s; $\sigma = 2\ s$

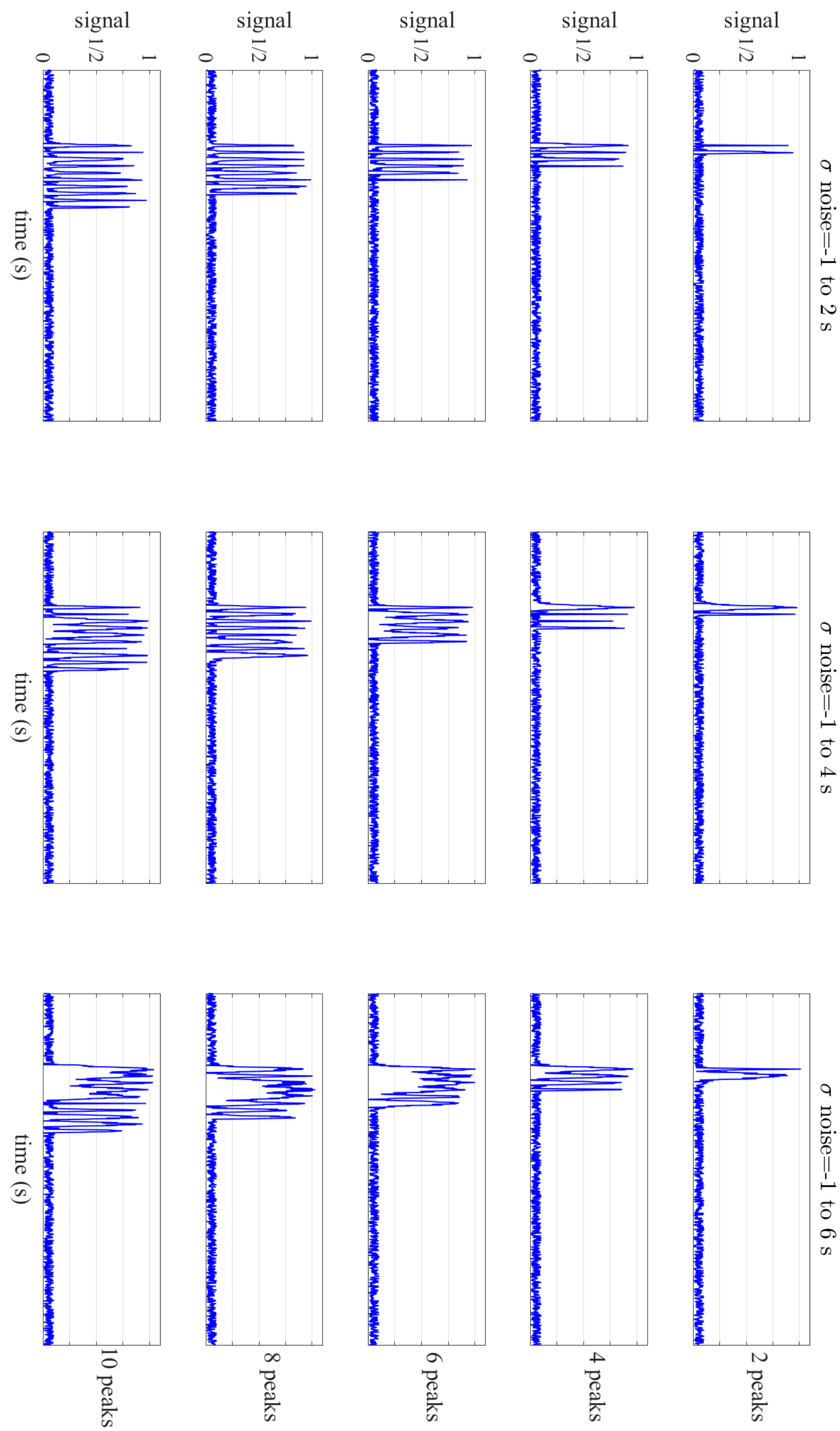


Figure C.21: noise floor [0 10%]; peak noise [-30 0%]; time jitter [0 0] s; $\sigma = 2\ s$

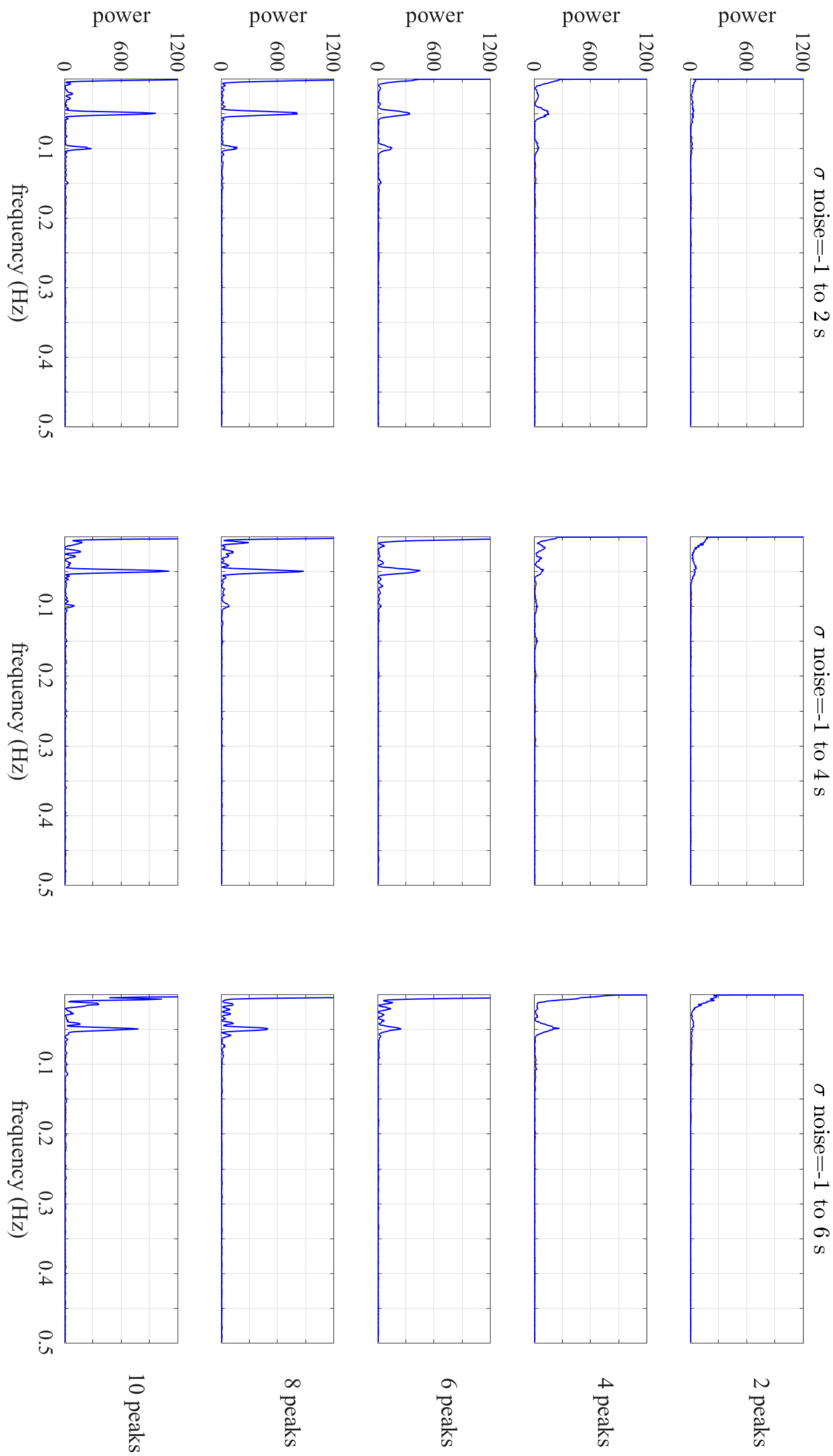


Figure C.22: noise floor [0 10%]; peak noise [-30 0%]; time jitter [0 0] s; $\sigma = 2\ s$

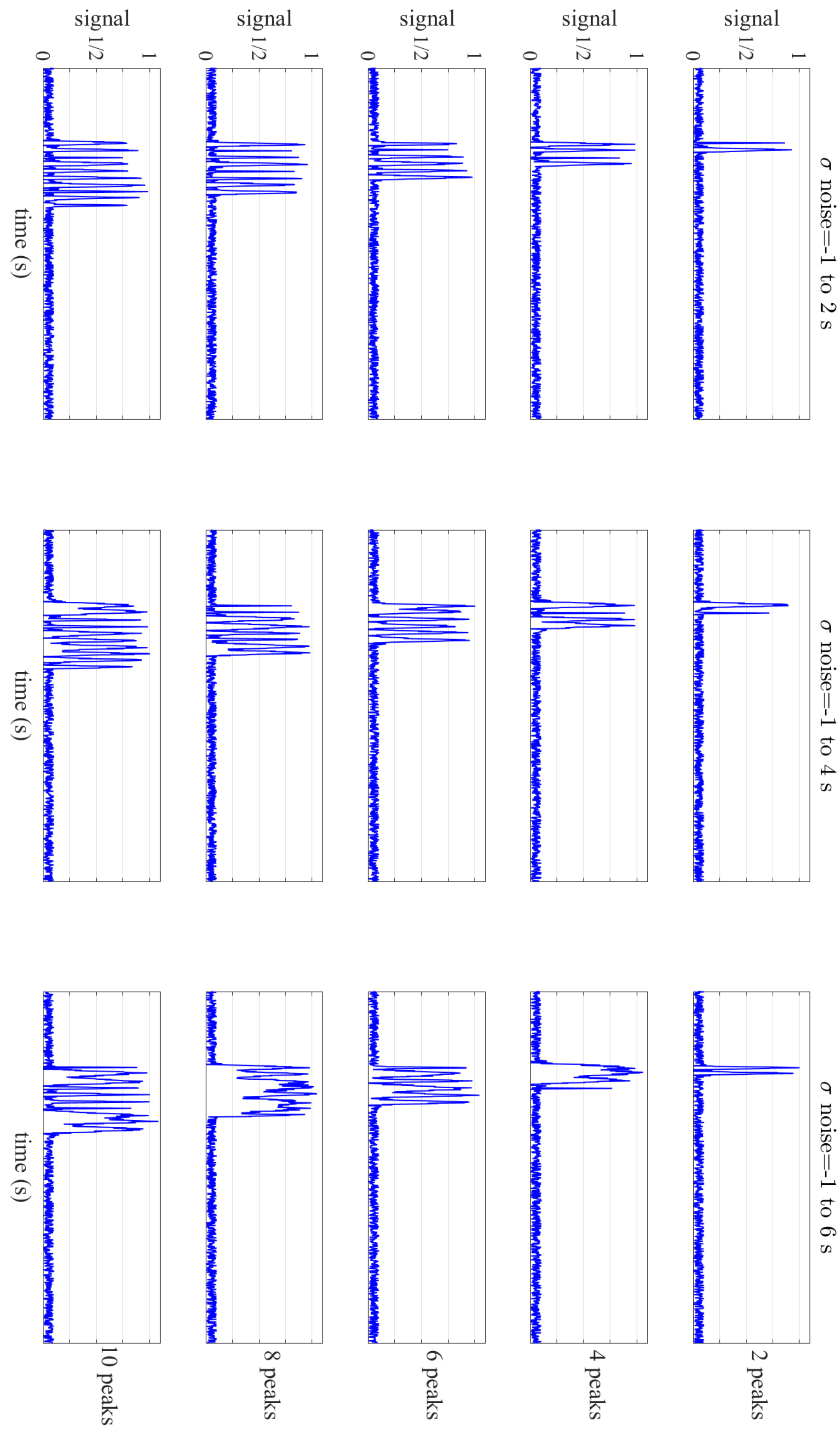


Figure C.23: noise floor [0 10%]; peak noise [-30 0%]; time jitter [-2 2] s; $\sigma = 2\ s$

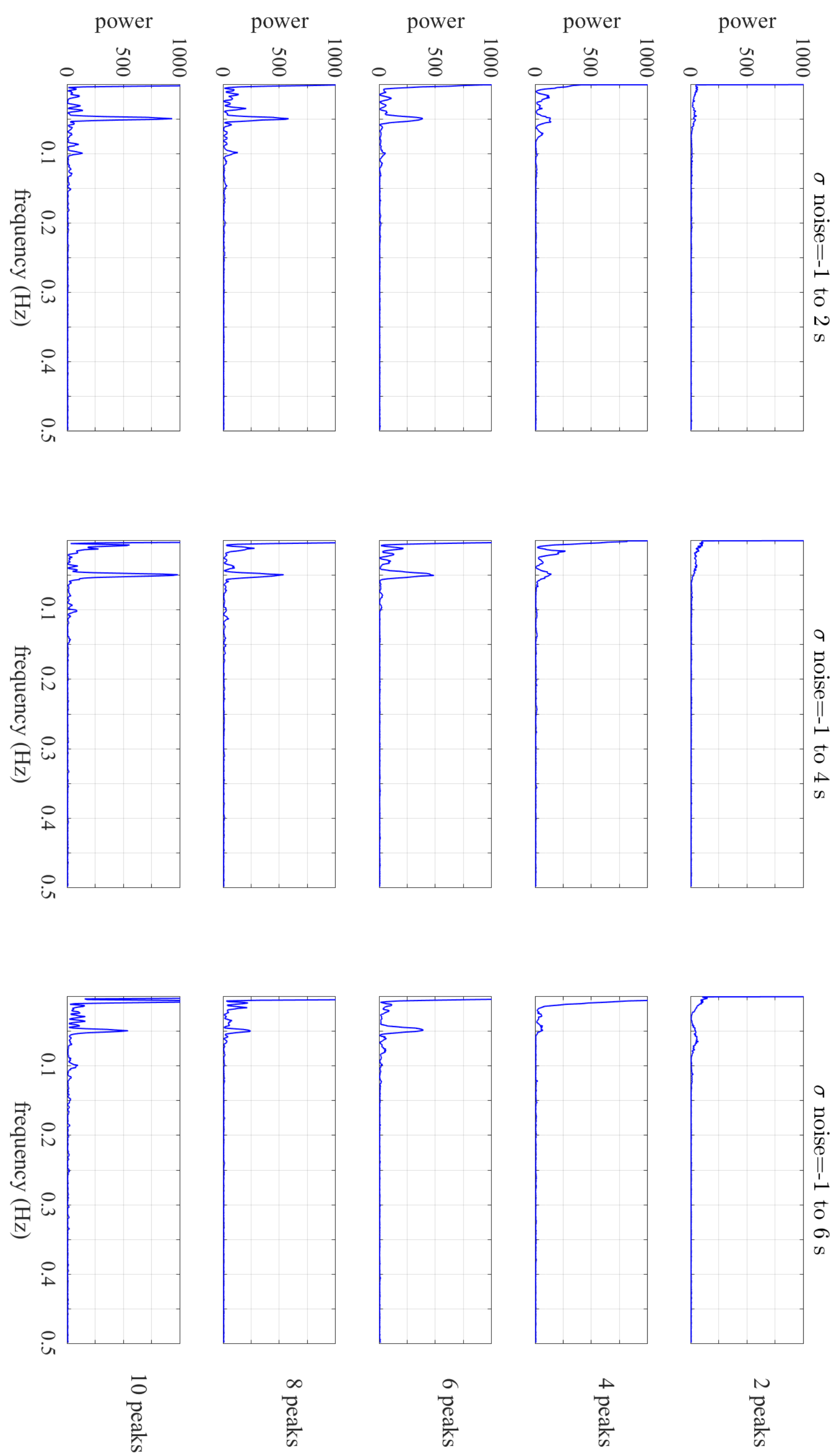


Figure C.24: noise floor [0 10%]; peak noise [-30 0%]; time jitter [-2 2] s; $\sigma = 2\ s$

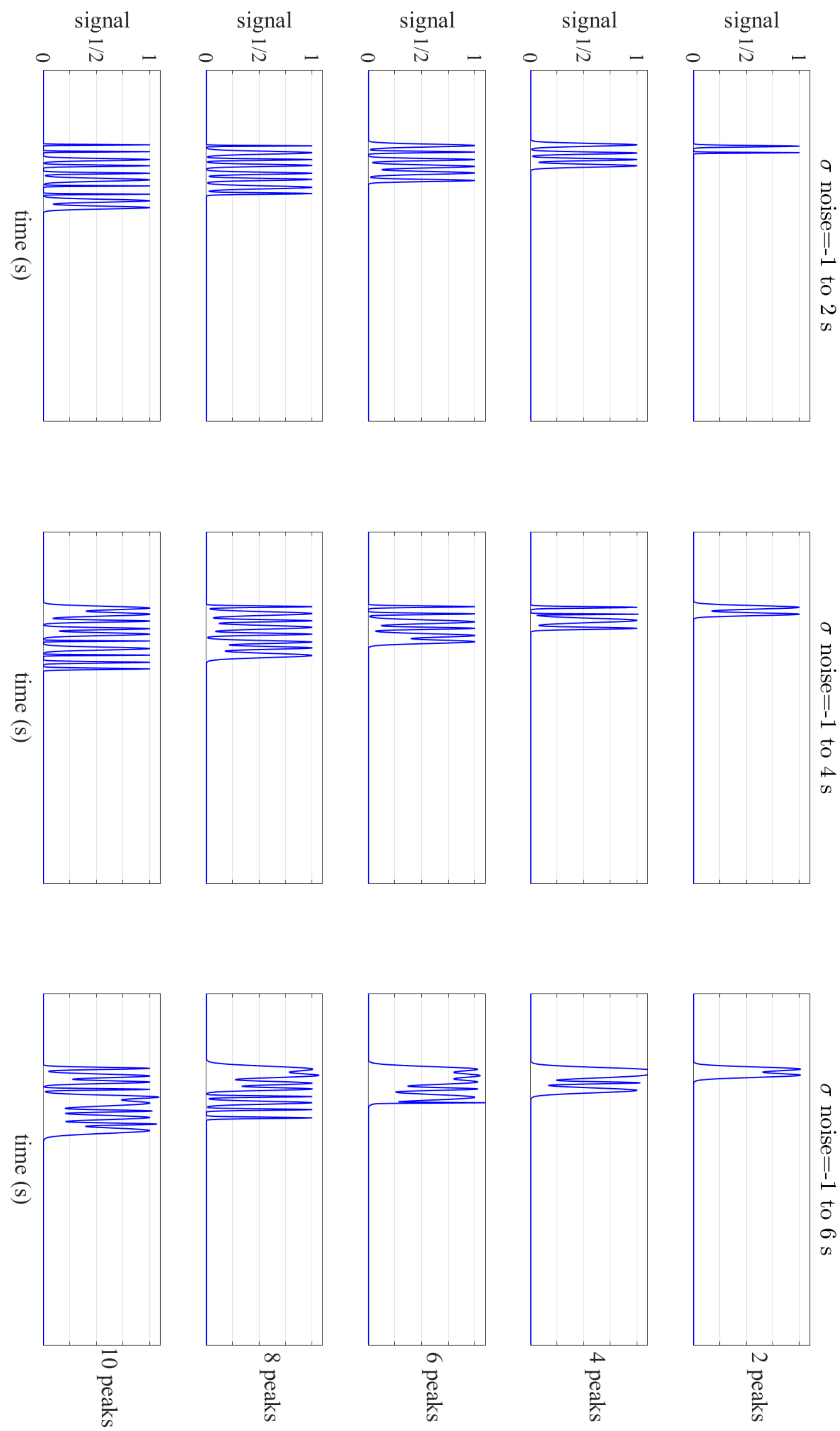


Figure C.25: noise floor [0 0]; peak noise [0 0]; time jitter [-2 2] s; $\sigma = 2\ s$

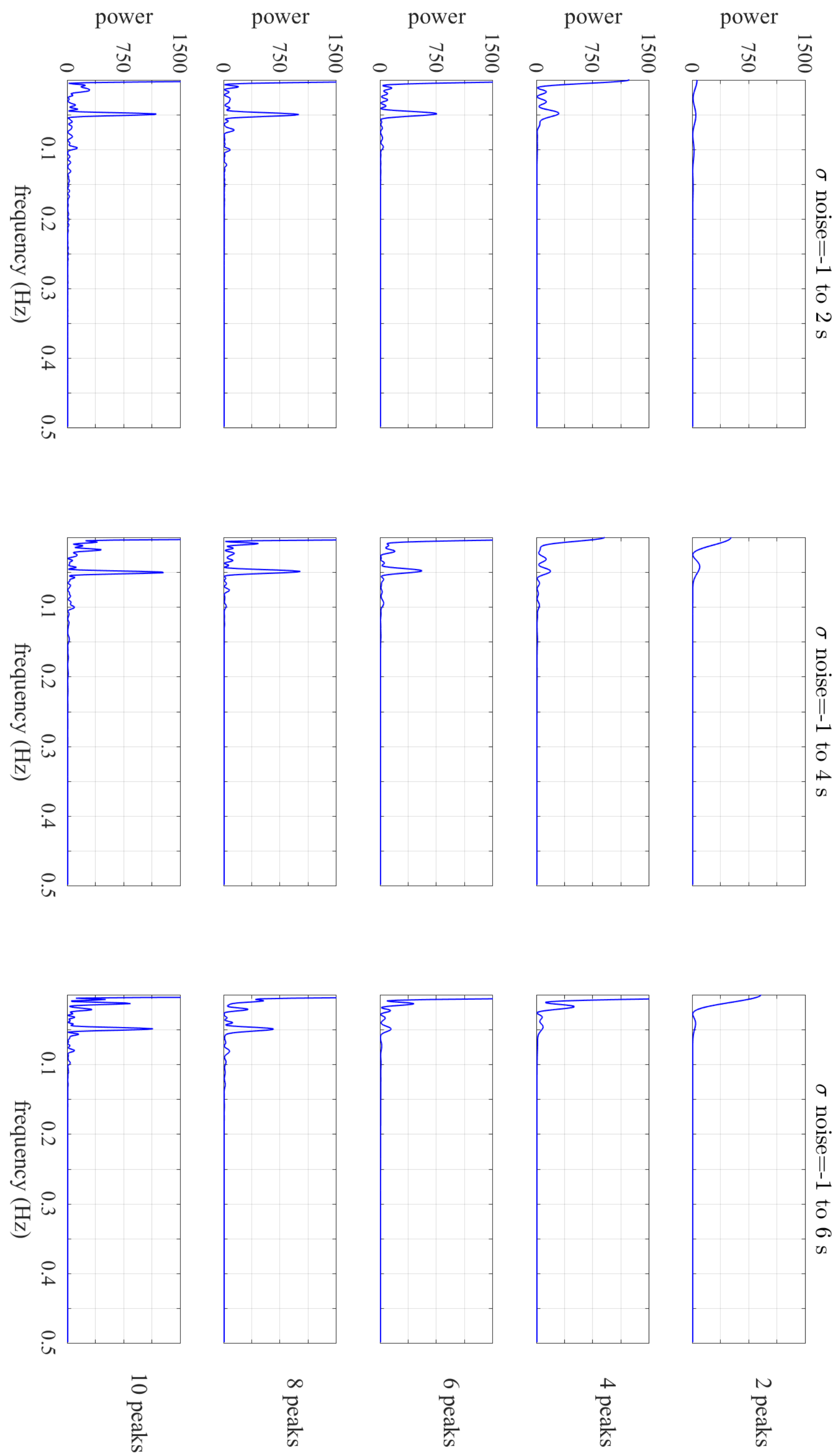


Figure C.26: noise floor [0 0]; peak noise [0 0]; time jitter [-2 2] s; $\sigma = 2\ s$

APPENDIX D

RADIOMETRIC ANLYSIS CODE AND PLOTS

The following pages contain the algorithms I wrote to simulate the radiometric model and plots.

Initialize

In[34]:=
```
ClearAll[LaserEnergy, LaserIrradiance, LaserPulseTime, LaserPower,
  TLE02Reflection, wl8, ref8, wl45, ref45, reflection8, reflection45,
  TLARCoatingA, wlA, refA, transmissionA, NumberOfMirrorsAt8,
  NumberOfMirrorsAt45, NumberOfLenses, LidarOutputPower, BeamDiameter,
  BeamIrradiance, WingSize, Reflectivity, RangeAtInsect, ScatteredRadiance,
  TelescopeOuterRadius, TelescopeObstructionRadius, EffectiveAperture,
  ProjectedSolidAngle, Throughput, ScatteredPower, AtmosphericTransmission,
  TelescopeReflectance, TelescopePlateTransmission, NumberOfBEFilters,
  NumberOfBELenses, NumberOfBEMirrors, NumberOfBEPlates, TLFLtran,
  wlf, traf, transmissionFL, DetectedPower, VariedAperture, x,
  y, TelescopeFocus, EffectiveInterestDiameter, TelescopeRatio,
  ProjectedSolidAngleOverFilled, PMTQE, ElectronCharge, BackgroundPower,
  WaveLength, PlancksConstant, SpeedOfLight, PMTCurrentSignal, PMTDarkCurrent,
  ElectricalBW, LensTubeObstruction, interpref8, interpref45, interprefA,
  DivergenceAngle, LensTubeObstruction, EffectiveApertureDiameter,
  DetectorArea, LambertianVegReflection, IrradianceSun, RadianceBackground,
  SBR, TempOfSun, AreaOfSun, k, DetectorArea, Lsun, Ωse, Esun,
  Lbg, Pbg, dfs, dis, QuantumEfficiencyPMT, QuantumEfficiencyWL,
  interpPMT, DistEarthToSun, PMTGain, LaserFreq, T, R, pulsereprate,
  BeamStartingDiameter, SNRv, tp, c, cv, Time, MPE, n, Cp, MPEwCp, MPEwCW,
  ES, PMTnoise, SNR, rng, tlr, wng, rlf, bmd, da, FOVx, FOVy, FOVd, EPL]
dfs = (*20*) 13;
dis = (*575*)420;
SetOptions[Plot, Axes → False,
  AxesStyle → Black, ImageSize → dis, LabelStyle →
   Directive[{FontFamily -> "Times New Roman", dfs, Thick, Black}],
  PlotStyle → {Thickness[.005], ColorData["DarkRainbow"]}, Frame → True];
SetOptions[SwatchLegend, LabelStyle →
   Directive[{FontFamily -> "Times New Roman", dfs, Thick, Black}],
  LegendMarkerSize → {15, 15}];
SetOptions[LogPlot, Axes → False, AxesStyle → Black,
  ImageSize → dis, LabelStyle →
   Directive[{FontFamily -> "Times New Roman", dfs, Thick, Black}],
  PlotStyle → {Thickness[.004], ColorData["DarkRainbow"]}, Frame → True];
SetOptions[LogLinearPlot, Axes → False, AxesStyle → Black,
  ImageSize → dis, LabelStyle →
   Directive[{FontFamily -> "Times New Roman", dfs, Thick, Black}],
  PlotStyle → {Thickness[.004], ColorData["DarkRainbow"]}, Frame → True];
TwoAxisPlot[{f_, g_}, {x_, x1_, x2_}] :=
 Module[{fgraph, ggraph, frange, grange, fticks, gticks},
```

```
  {fgraph, ggraph} = MapIndexed[Plot[#, {x, x1, x2},
      Axes → True, PlotStyle → ColorData[1][#2[[1]]]] &, {f, g}];
  {frange, grange} = (PlotRange /. AbsoluteOptions[#, PlotRange])[[2]] & /@
    {fgraph, ggraph};
  fticks = N@FindDivisions[frange, 5];
  gticks =
   Quiet@Transpose@{fticks, ToString[NumberForm[#, 2], StandardForm] & /@
       Rescale[fticks, frange, grange]};
  Show[fgraph, ggraph /. Graphics[graph_, s___] :→
     Graphics[GeometricTransformation[graph, RescalingTransform[
        {{0, 1}, grange}, {{0, 1}, frange}]], s], Axes → False, Frame → True,
   FrameStyle → {ColorData[1] /@ {1, 2}, {Automatic, Automatic}},
   FrameTicks → {{fticks, gticks}, {Automatic, Automatic}}]]
fullfilecld =
  "C:\\Users\\user\\OneDrive\\Documents\\Martin\\Research\\Thesis
    Paper\\Drafts\\figures\\rad_figs\\";
fullfileloc = "C:\\Users\\user\\Documents\\Research\\Insect
    Lidar\\Field Tests\\Figures\\Thesis Figures\\rad_figs\\";
```

Define quantities

■ Colors

```
In[44]:= c[01] = RGBColor[18.8 / 100, 43.5 / 100, 61.6 / 100];
c[02] = RGBColor[58.4 / 100, 38.8 / 100, 23.9 / 100];
c[03] = RGBColor[62.4 / 100, 63.5 / 100, 63.1 / 100];
c[04] = RGBColor[58.0 / 100, 41.6 / 100, 61.2 / 100];
c[05] = RGBColor[84.3 / 100, 36.1 / 100, 33.7 / 100];
c[06] = RGBColor[96.1 / 100, 71.8 / 100, 40.8 / 100];
c[07] = RGBColor[50.2 / 100, 67.8 / 100, 43.5 / 100];
c[08] = RGBColor[29.0 / 100, 50.2 / 100, 67.1 / 100];
c[09] = RGBColor[56.1 / 100, 57.3 / 100, 56.9 / 100];
c[10] = RGBColor[51.4 / 100, 34.5 / 100, 54.5 / 100];
c[11] = RGBColor[81.2 / 100, 27.5 / 100, 23.5 / 100];
c[12] = RGBColor[94.9 / 100, 67.8 / 100, 31.0 / 100];
c[13] = RGBColor[42.7 / 100, 62.7 / 100, 36.1 / 100];
```

Laser Quantities

```
In[24]:= LaserEnergy = Quantity[3, "micro joules"];
LaserPulseTime = Quantity[0.6, "nano second"];
LaserPower = UnitSimplify[LaserEnergy / LaserPulseTime];
```

Optics Quantities

```
In[27]:= TLE02Reflection = Import["C:\\Martin_Tauc\\Research\\Radiometric
      Analysis\\E02ReflectionData.xlsx",
    {"Data", 1, All, {3, 4, 5, 6, 7, 8}}];
wl8 = Drop[TLE02Reflection[[All, 1]], 1];
ref8 = Drop[TLE02Reflection[[All, 2]], 1];
wl45 = Drop[Drop[TLE02Reflection[[All, 3]], 1], -27];
ref45 = Drop[Drop[TLE02Reflection[[All, 4]], 1], -27];
interpref8 = Interpolation[Transpose[{wl8, ref8}]];
interpref45 = Interpolation[Transpose[{wl45, ref45}]];
reflection8 = interpref8[532] / 100;
reflection45 = interpref45[532] / 100;
```

```
In[36]:= TLARCoatingA = Import["C:\\Martin_Tauc\\Research\\Radiometric
      Analysis\\A_Broadband_AR-Coating.xlsx", {"Data", 1, All, {3, 4}}];
wlA = Drop[TLARCoatingA[[All, 1]], 2];
refA = Drop[TLARCoatingA[[All, 2]], 2];
interprefA = Interpolation[Transpose[{wlA, refA}]];
transmissionA = (100 - interprefA[532]) / 100;
NumberOfMirrorsAt8 = 1;
NumberOfMirrorsAt45 = 5;
NumberOfLenses = 2;
```

Telescope Quantities

```
In[44]:= TelescopeObstructionRadius = Quantity[0.1016, "m"] / 2;
LensTubeObstruction = Quantity[55.9, "mm"] * Quantity[127, "mm"];
```

Atmospheric Quantities

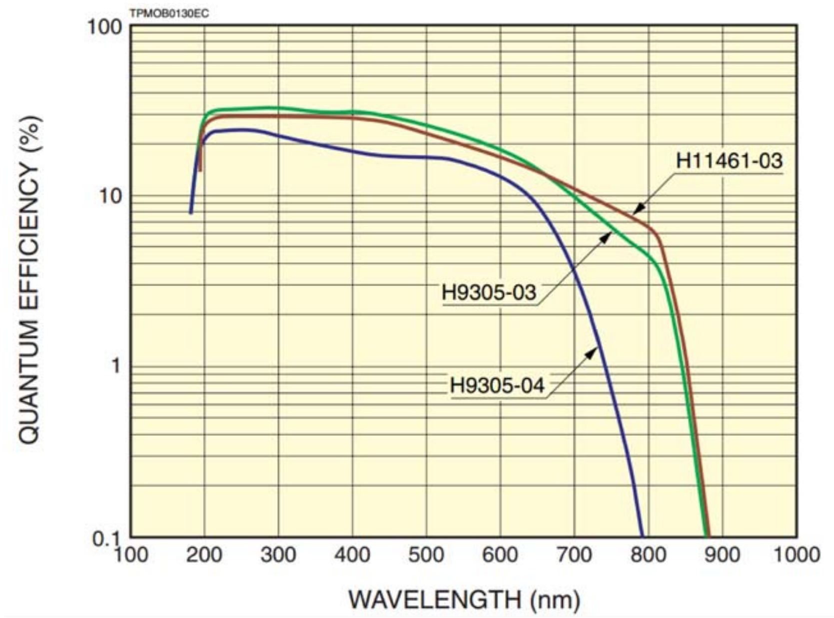
In[46]:=
```
AtmosphericTransmission = 0.983;
TelescopeReflectance = 0.94;
TelescopePlateTransmission = 0.98;
TelescopeFocus = Quantity[3048, "mm"];
NumberOfBEPlates = 1;
NumberOfBEMirrors = 2;
NumberOfBELenses = 1;
NumberOfBEFilters = 2;
TLFLtran =
  Import["C:\\Martin_Tauc\\Research\\Radiometric Analysis\\FL532-3.xlsx",
   {"Data", 1, All, {3, 4}}];
wlf = Drop[TLFLtran[[All, 1]], 1];
traf = Drop[TLFLtran[[All, 2]], 1];
transmissionFL = traf[[Flatten[Position[wlf, 532.]]]][[1]] / 100;
```

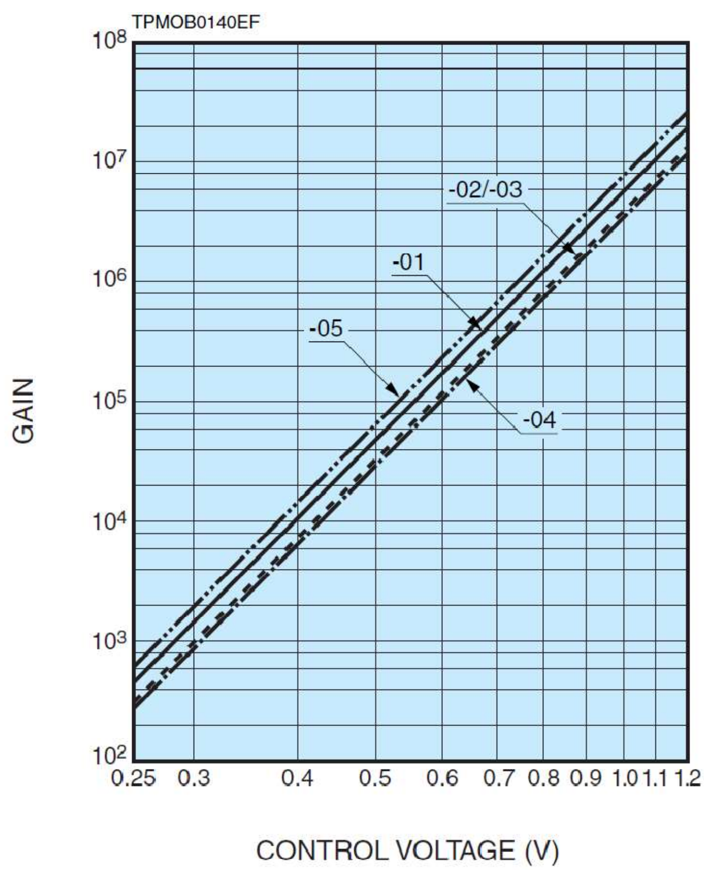
In[58]:=
```
DetectorArea = Quantity[3.7, "mm"] * Quantity[13, "mm"];
LambertianVegReflection = 0.15;
IrradianceSun = Quantity[10.83, "W/(m^2)"];
```

In[61]:=
```
TempOfSun = Quantity[5778, "Kelvins"];
AreaOfSun = Quantity[696 300, "km"]^2 * Pi;
DistEarthToSun = Quantity[149.6*^6, "km"];
PlancksConstant = Quantity[6.626*^-34, "m^2*kg/s"];
SpeedOfLight = Quantity[2.998*^8, "m/s"];
k = Quantity[1.38*^-23, "J/K"];
WaveLength = Quantity[532, "nm"];
DetectorArea = Quantity[3.7 * 13, "mm^2"];
```

Noise Quantities

In[69]:=  ;

In[70]:= 
TPMOB0130EC
QUANTUM EFFICIENCY (%)
WAVELENGTH (nm)
H11461-03
H9305-03
H9305-04
100
10
1
0.1
100
200
300
400
500
600
700
800
900
1000
;

In[71]:= 
TPMOB0140EF
GAIN
CONTROL VOLTAGE (V)
-02/-03
-01
-05
-04
10^8
10^7
10^6
10^5
10^4
10^3
10^2
0.25
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.1
1.2
;

In[72]:=
```
QuantumEfficiencyWL = {200, 300, 400, 500, 600, 700, 800};
QuantumEfficiencyPMT = {21.3, 22.3, 18.3, 16.6, 12.9, 3.5, 0.1};
interpPMT =
  Interpolation[Transpose[{QuantumEfficiencyWL, QuantumEfficiencyPMT}]];
PMTQE = interpPMT[532] / 100;
ElectronCharge = Quantity[1.602*^-19, "Coulombs"];
PMTGain = 4*^6;
LaserFreq = Quantity[9, "kHz"];
ElectricalBW = Quantity[100, "MHz"];
PMTDarkCurrent = Quantity[1, "nanoAmp"];
T = Quantity[300, "K"];
R = Quantity[1, "MegaOhm"];
```

■ Variable Quantities

In[83]:=
```
BeamStartingDiameter = Quantity[0.0508, "m"];
(*WingSize=UnitConvert[Quantity[ws,"mm^2"],"m^2"];
RangeAtInsect=Quantity[rng,"m"];
Reflectivity=Quantity[rlf,"DimensionlessUnit"];
DivergenceAngle=Quantity[da,"Radians"];
TelescopeOuterRadius=Quantity[tlr,"m"]/2;*)
```

In[84]:=
```
(*WingSize=UnitConvert[Quantity[25,"mm^2"],"m^2"]//N*)
(*Reflectivity=0.15;*)
(*RangeAtInsect=Quantity[100,"m"];*)
(*TelescopeOuterRadius=Quantity[0.3048,"m"]/2*)
```

Equations

■ laser power

In[85]:=
```
LidarOutputPower = Simplify[LaserPower * reflection8^NumberOfMirrorsAt8 *
   reflection45^NumberOfMirrorsAt45 * transmissionA^NumberOfLenses]
```

Out[85]=
```
4789.88 W
```

■ beam profile

In[86]:=
```
BeamDiameter =
 RangeAtInsect * 2 * Quantity[Tan[DivergenceAngle], "DimensionlessUnit"] +
  BeamStartingDiameter
BeamIrradiance = LidarOutputPower / (Pi * (BeamDiameter / 2) ^2);
ScatteredRadiance = BeamIrradiance * Reflectivity / Quantity[Pi, "sr"]
```

Out[86]= 0.0508 m + RangeAtInsect (2 Tan[DivergenceAngle])

Out[87]= (Reflectivity (1941.27 W/sr)) / (0.0508 m + RangeAtInsect (2 Tan[DivergenceAngle]))^2

Telescope Throughput

In[88]:=
```
TelescopeRatio =
 Simplify[(TelescopeOuterRadius) / TelescopeObstructionRadius]
EffectiveAperture = Simplify[Pi * (TelescopeOuterRadius)^2 -
   Pi * TelescopeObstructionRadius^2 - LensTubeObstruction]
EffectiveApertureDiameter = Simplify[
  2 * ((Pi * (TelescopeOuterRadius)^2 - Pi * TelescopeObstructionRadius^2 -
        LensTubeObstruction) / Pi)^(1 / 2)]
ProjectedSolidAngle = Simplify[Quantity[WingSize / RangeAtInsect^2, "sr"]]
Throughput = Simplify[EffectiveAperture * ProjectedSolidAngle]
ScatteredPower = Simplify[ScatteredRadiance * Throughput]
```

Out[88]= $\text{TelescopeOuterRadius}\left(19.685\ \text{per meter}\right)$

Out[89]= $\pi\,\text{TelescopeOuterRadius}^2 + -15\,206.6\ \text{mm}^2$

Out[90]= $2\sqrt{\text{TelescopeOuterRadius}^2 + -4840.42\ \text{mm}^2}$

Out[91]= $\frac{\text{WingSize}}{\text{RangeAtInsect}^2}\ \text{sr}$

Out[92]= $\left(\pi\,\text{TelescopeOuterRadius}^2 + -15\,206.6\ \text{mm}^2\right)\left(\frac{\text{WingSize}}{\text{RangeAtInsect}^2}\ \text{sr}\right)$

Out[93]= $\left(\text{Reflectivity}\left(\pi\,\text{TelescopeOuterRadius}^2 + -15\,206.6\ \text{mm}^2\right)\left(\frac{1941.27\ \text{WingSize}}{\text{RangeAtInsect}^2}\ \text{W}\right)\right) \Big/ \left(0.0508\ \text{m} + \text{RangeAtInsect}\left(2\,\text{Tan}[\text{DivergenceAngle}]\right)\right)^2$

Atmospheric and Receiver Transmission

In[94]:=

```
In[95]:= DetectedPower = Simplify[ScatteredPower * AtmosphericTransmission^2 *
            TelescopePlateTransmission^NumberOfBEPlates *
            TelescopeReflectance^NumberOfBEMirrors *
            transmissionA^NumberOfBELenses * transmissionFL^NumberOfBEFilters]
```

Out[95]= $\left(\text{Reflectivity}\left(\pi\,\text{TelescopeOuterRadius}^2 + -15\,206.6\,\text{mm}^2\right)\left(\frac{605.59\,\text{WingSize}}{\text{RangeAtInsect}^2}\,\text{W}\right)\right)\Big/\left(0.0508\,\text{m} + \text{RangeAtInsect}\left(2\,\text{Tan}[\text{DivergenceAngle}]\right)\right)^2$

Background Signal

```
In[96]:= ProjectedSolidAngleOverFilled = Simplify[DetectorArea / (TelescopeFocus)^2]
         RadianceBackground = Simplify[IrradianceSun * LambertianVegReflection / Pi]
         BackgroundPower = Simplify[
           RadianceBackground * EffectiveAperture * ProjectedSolidAngleOverFilled *
            AtmosphericTransmission * TelescopePlateTransmission^NumberOfBEPlates *
            TelescopeReflectance^NumberOfBEMirrors *
            transmissionA^NumberOfBELenses * transmissionFL^NumberOfBEFilters]
         SBR = Simplify[DetectedPower / BackgroundPower]
```

Out[96]= 5.17744×10^{-6}

Out[97]= $0.517094\,\text{W}/\text{m}^2$

Out[98]= $-1.29199\times10^{-8}\,\text{W} + \text{TelescopeOuterRadius}^2\left(2.66916\times10^{-6}\,\text{W}/\text{m}^2\right)$

Out[99]= $\left(\text{Reflectivity}\left(\pi\,\text{TelescopeOuterRadius}^2 + -15\,206.6\,\text{mm}^2\right)\left(\frac{605.59\,\text{WingSize}}{\text{RangeAtInsect}^2}\,\text{W}\right)\right)\Big/\left(\left(-1.29199\times10^{-8}\,\text{W} + \text{TelescopeOuterRadius}^2\left(2.66916\times10^{-6}\,\text{W}/\text{m}^2\right)\right)\left(0.0508\,\text{m} + \text{RangeAtInsect}\left(2\,\text{Tan}[\text{DivergenceAngle}]\right)\right)^2\right)$

Noise (For H9305-04)

In[100]:=
```
PMTCurrentSignal = PMTQE * ElectronCharge * (DetectedPower + BackgroundPower) *
   WaveLength / (PlancksConstant * SpeedOfLight)
PMTnoise = Sqrt[2 * ElectronCharge *
    (PMTCurrentSignal + PMTDarkCurrent) * ElectricalBW]
```

Out[100]=

$$\left(6.78333\times10^{-20}\ \mathrm{s^2C/(kg\,nm^2)}\right)\left(-1.29199\times10^{-8}\ \mathrm{W} + \mathtt{TelescopeOuterRadius}^2\left(2.66916\times10^{-6}\ \mathrm{W/m^2}\right) + \left(\mathtt{Reflectivity}\left(\pi\,\mathtt{TelescopeOuterRadius}^2 + -15\,206.6\ \mathrm{mm^2}\right)\left(\frac{605.59\ \mathtt{WingSize}}{\mathtt{RangeAtInsect}^2}\ \mathrm{W}\right)\right)\Big/\left(0.0508\ \mathrm{m} + \mathtt{RangeAtInsect}\left(2\ \mathrm{Tan}[\mathtt{DivergenceAngle}]\right)\right)^2\right)$$

Out[101]=

$$\sqrt{\left(\left(3.204\times10^{-17}\ \mathrm{C\,MHz}\right)\left(1\ \mathrm{nA} + \left(6.78333\times10^{-20}\ \mathrm{s^2C/(kg\,nm^2)}\right)\left(-1.29199\times10^{-8}\ \mathrm{W} + \mathtt{TelescopeOuterRadius}^2\left(2.66916\times10^{-6}\ \mathrm{W/m^2}\right) + \left(\mathtt{Reflectivity}\left(\pi\,\mathtt{TelescopeOuterRadius}^2 + -15\,206.6\ \mathrm{mm^2}\right)\left(\frac{605.59\ \mathtt{WingSize}}{\mathtt{RangeAtInsect}^2}\ \mathrm{W}\right)\right)\Big/\left(0.0508\ \mathrm{m} + \mathtt{RangeAtInsect}\left(2\ \mathrm{Tan}[\mathtt{DivergenceAngle}]\right)\right)^2\right)\right)\right)}$$

In[102]:= **SNR = PMTCurrentSignal / PMTnoise**

Out[102]=

$$\left(\left(6.78333\times10^{-20}\ \mathrm{s^2C/(kg\,nm^2)}\right)\left(-1.29199\times10^{-8}\ \mathrm{W} + \mathrm{TelescopeOuterRadius}^2\left(2.66916\times10^{-6}\ \mathrm{W/m^2}\right) + \left(\mathrm{Reflectivity}\left(\pi\,\mathrm{TelescopeOuterRadius}^2 + -15\,206.6\ \mathrm{mm^2}\right)\left(\frac{605.59\,\mathrm{WingSize}}{\mathrm{RangeAtInsect}^2}\ \mathrm{W}\right)\right)\Big/\left(0.0508\ \mathrm{m} + \mathrm{RangeAtInsect}\left(2\,\mathrm{Tan}[\mathrm{DivergenceAngle}]\right)\right)^2\right)\right)\Big/$$

$$\left(\sqrt{\left(\left(3.204\times10^{-17}\ \mathrm{C\,MHz}\right)\left(1\ \mathrm{nA} + \left(6.78333\times10^{-20}\ \mathrm{s^2C/(kg\,nm^2)}\right)\left(-1.29199\times10^{-8}\ \mathrm{W} + \mathrm{TelescopeOuterRadius}^2\left(2.66916\times10^{-6}\ \mathrm{W/m^2}\right) + \left(\mathrm{Reflectivity}\left(\pi\,\mathrm{TelescopeOuterRadius}^2 + -15\,206.6\ \mathrm{mm^2}\right)\left(\frac{605.59\,\mathrm{WingSize}}{\mathrm{RangeAtInsect}^2}\ \mathrm{W}\right)\right)\Big/\left(0.0508\ \mathrm{m} + \mathrm{RangeAtInsect}\left(2\,\mathrm{Tan}[\mathrm{DivergenceAngle}]\right)\right)^2\right)\right)\right)}\right)$$

Calculations with assumed values

```
In[103]:= assignments = {TelescopeOuterRadius -> Quantity[tlr, "m"] / 2,
            WingSize → UnitConvert[Quantity[ws, "mm^2"], "m^2"],
            RangeAtInsect → Quantity[rng, "m"],
            Reflectivity → Quantity[rlf, "DimensionlessUnit"],
            DivergenceAngle → Quantity[da, "Radians"]};
          vals = {tlr -> .3048, rlf -> .15, ws -> 25, da -> 0.0002, rng → 100};
```

```
In[104]:= UnitConvert[BeamIrradiance /. assignments /. vals, "kW/m^2"]
UnitConvert[ScatteredRadiance /. assignments /. vals, "kW/(m^2 sr)"]
UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"]
UnitConvert[ProjectedSolidAngle /. assignments /. vals, "sr"] // N
UnitConvert[Throughput /. assignments /. vals, "m^2*sr"]
UnitConvert[ScatteredPower /. assignments /. vals, "W"]
UnitConvert[DetectedPower /. assignments /. vals, "W"]
UnitConvert[BackgroundPower /. assignments /. vals, "microW"]
SBR /. assignments /. vals
UnitConvert[PMTCurrentSignal /. assignments /. vals, "microA"]
UnitConvert[SNR /. assignments /. vals, "DimensionlessUnit"]
UnitConvert[Sqrt[(4 * k * T * ElectricalBW / R)], "Amperes"]
UnitConvert[
  Sqrt[(10^5)^2 * 2 * ElectronCharge * (PMTCurrentSignal + PMTDarkCurrent) *
      ElectricalBW] /. assignments /. vals, "Amperes"] // ScientificForm
```

Out[104]= $739.713\ \mathrm{kW/m^2}$

Out[105]= $35.3187\ \mathrm{kW/(m^2 sr)}$

Out[106]= $577.593\ \mathrm{cm^2}$

Out[107]= $2.5\times10^{-9}\ \mathrm{sr}$

Out[108]= $1.44398\times10^{-10}\ \mathrm{m^2 sr}$

Out[109]= $5.09996\times10^{-6}\ \mathrm{W}$

Out[110]= $1.59096\times10^{-6}\ \mathrm{W}$

Out[111]= $0.0490734\ \mu\mathrm{W}$

Out[112]= 32.42

Out[113]= $0.111249\ \mu\mathrm{A}$

Out[114]= 58.6622

Out[115]= $1.28686\times10^{-9}\ \mathrm{A}$

Out[116]//ScientificForm=

$1.89643\times10^{-4}\ \mathrm{A}$

Detected Power and SNR

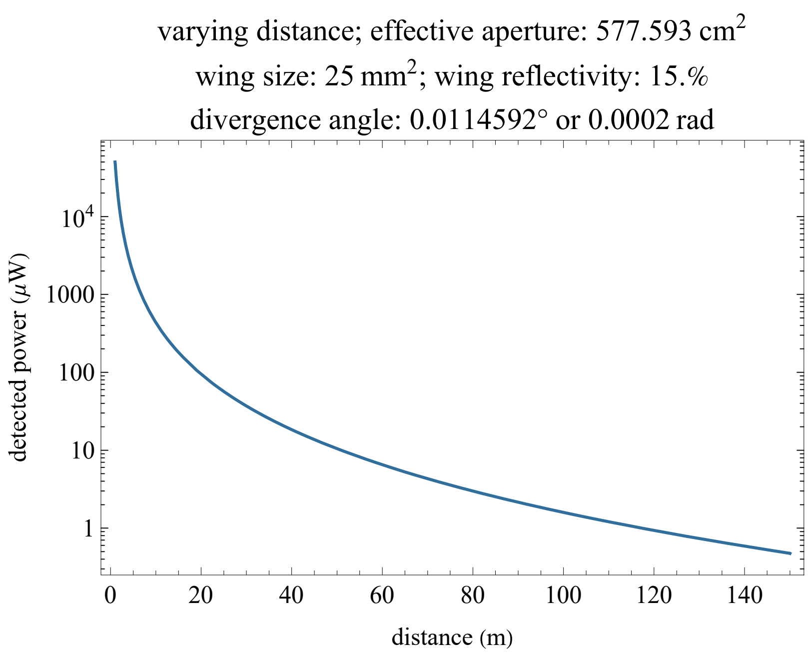
Plots

In[117]:=
```
rangelb = 1;
rangeub = 150;
SNRv = Quantity[10, "DimensionlessUnit"];
```

detected power vs distance

In[120]:=
```
assignments = {TelescopeOuterRadius -> Quantity[tlr, "m"] / 2,
   WingSize → UnitConvert[Quantity[ws, "mm^2"], "m^2"],
   RangeAtInsect → Quantity[rng, "m"],
   Reflectivity → Quantity[rlf, "DimensionlessUnit"],
   DivergenceAngle → Quantity[da, "Radians"]};
vals = {tlr -> .3048, rlf -> .15, ws -> 25, da -> 0.0002};
plt[1] = LogPlot[UnitConvert[
    DetectedPower /. assignments /. vals, "microWatts"], {rng, 1, 150},
   PlotLabel -> StringJoin["varying distance; effective aperture: ",
     ToString[UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"],
      TraditionalForm], "\nwing size: ",
     ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
      TraditionalForm], "; wing reflectivity: ",
     ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
     "%\ndivergence angle: ",
     ToString[UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
      TraditionalForm], " or ", ToString[UnitConvert[
       DivergenceAngle /. assignments /. vals, "rad"], TraditionalForm]],
   FrameLabel → {"distance (m)", "detected power (μW)"},
   PlotStyle → {c[1]}]
```

Out[121]=

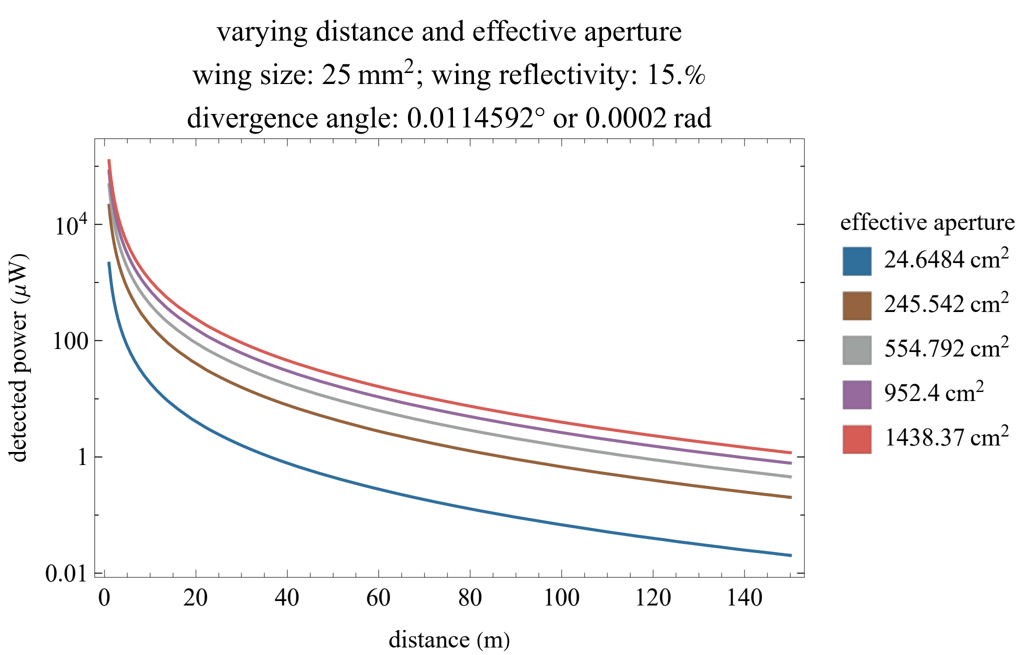


vary effective aperture

In[122]:=
```
ClearAll[tp, cv]
cv = 1;
vals = {da → .0002, rlf → .15, ws → 25, lb → 0.15, ub → .5, it → .075};
Do[{tp[cv] = LogPlot[UnitConvert[DetectedPower /. assignments /. vals,
        "microWatts"], {rng, rangelb, rangeub}, PlotStyle → {c[cv]}],
    cv = cv + 1}, {tlr, lb /. vals, ub /. vals, it /. vals}];
```

In[126]:=
```
plt[2] = Legended[Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All,
    FrameLabel → {"distance (m)", "detected power (μW)"}, PlotLabel ->
     StringJoin["varying distance and effective aperture\nwing size: ",
      ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
       TraditionalForm], "; wing reflectivity: ",
      ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
      "%\ndivergence angle: ", ToString[
       UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
       TraditionalForm], " or ", ToString[UnitConvert[
         DivergenceAngle /. assignments /. vals, "rad"], TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[
     ToString[UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"],
      TraditionalForm], {tlr, lb /. vals, ub /. vals, it /. vals}],
    LegendLabel → "effective aperture"]]
```

Out[126]=

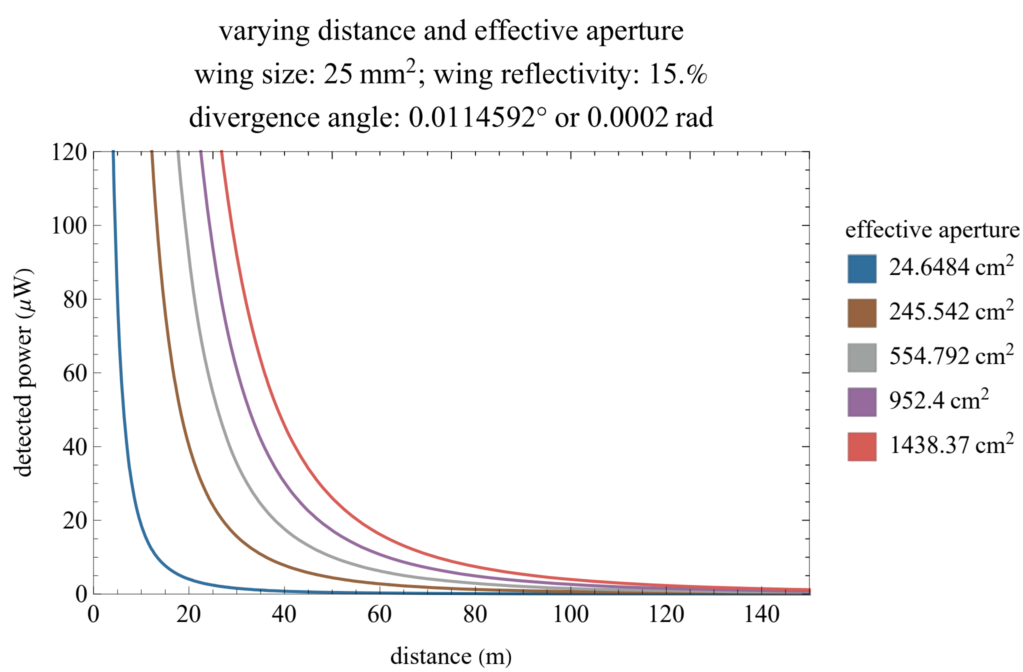


vary effective aperture

In[127]:=
```
ClearAll[tp, cv]
cv = 1;
vals = {da → .0002, rlf → .15, ws → 25, lb → 0.15, ub → .5, it → .075};
Do[{tp[cv] = Plot[UnitConvert[DetectedPower /. assignments /. vals,
      "microWatts"], {rng, rangelb, rangeub},
     PlotRange → {{0, 150}, {0, 120}}, PlotStyle → {c[cv]}],
   cv = cv + 1}, {tlr, lb /. vals, ub /. vals, it /. vals}];
```

In[131]:=
```
plt[22] = Legended[Show[Table[tp[h], {h, 1, cv - 1}],
   FrameLabel → {"distance (m)", "detected power (μW)"}, PlotLabel ->
    StringJoin["varying distance and effective aperture\nwing size: ",
     ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
      TraditionalForm], "; wing reflectivity: ",
     ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
     "%\ndivergence angle: ", ToString[
      UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
      TraditionalForm], " or ", ToString[UnitConvert[
       DivergenceAngle /. assignments /. vals, "rad"], TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[
    ToString[UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"],
     TraditionalForm], {tlr, lb /. vals, ub /. vals, it /. vals}],
   LegendLabel → "effective aperture"]]
```

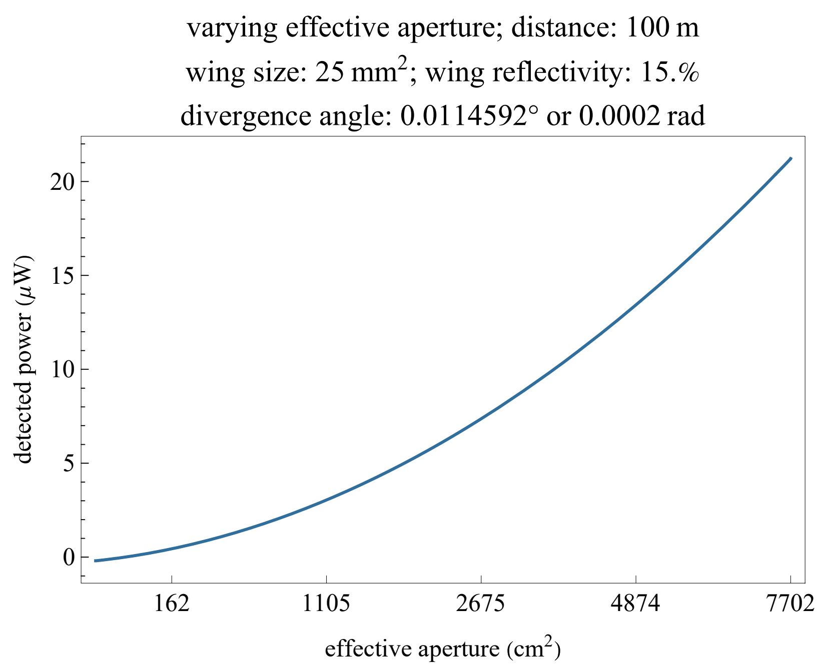
Out[131]=



■ detected power vs effective aperture

In[132]:=
```
assignments = {TelescopeOuterRadius -> Quantity[tlr, "m"] / 2,
   WingSize → UnitConvert[Quantity[ws, "mm^2"], "m^2"],
   RangeAtInsect → Quantity[rng, "m"],
   Reflectivity → Quantity[rlf, "DimensionlessUnit"],
   DivergenceAngle → Quantity[da, "Radians"]};
vals = {rng → 100, rlf -> .15, ws -> 25, da -> 0.0002};
plt[3] =
 Plot[UnitConvert[DetectedPower /. assignments /. vals, "microWatts"],
  {tlr, .1016, 1}, Frame → {{True, True}, {True, True}},
  FrameTicks -> {{All, None},
    {Table[{tlr, Round[QuantityMagnitude[UnitConvert[EffectiveAperture /.
           assignments, "cm^2"]]]}, {tlr, .2, 1, .2}], None}},
  PlotLabel -> StringJoin["varying effective aperture; distance: ",
    ToString[UnitConvert[RangeAtInsect /. assignments /. vals, "m"],
     TraditionalForm], "\nwing size: ",
    ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
     TraditionalForm], "; wing reflectivity: ",
    ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
    "%\ndivergence angle: ",
    ToString[UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
     TraditionalForm], " or ", ToString[UnitConvert[
      DivergenceAngle /. assignments /. vals, "rad"], TraditionalForm]],
  FrameLabel → {"effective aperture (cm²)", "detected power (μW)"},
  PlotStyle → {c[1]}]
```

Out[133]=

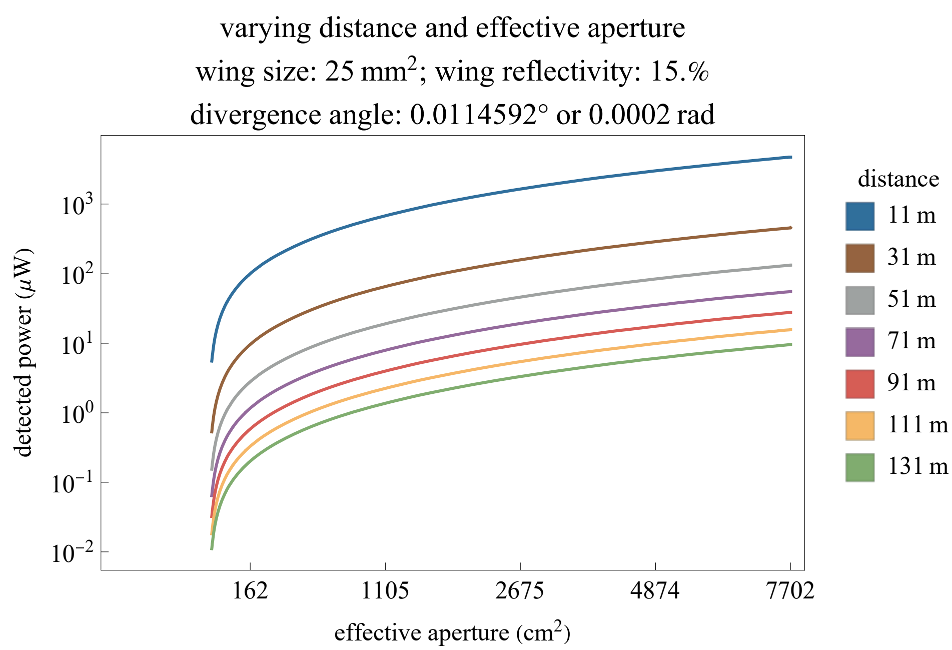


vary distance

In[134]:=
```
ClearAll[tp, cv]
cv = 1;
vals = {da → .0002, rlf → .15, ws → 25, lb → 11, ub → 141, it → 20};
Do[{tp[cv] = Plot[Log10[QuantityMagnitude[
        UnitConvert[DetectedPower /. assignments /. vals, "microWatts"]]],
      {tlr, .1016, 1}, PlotStyle → {c[cv]}], cv = cv + 1},
  {rng, lb /. vals, ub /. vals, it /. vals}];
```

In[138]:=
```
plt[4] = Legended[Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → Automatic,
   FrameLabel → {"effective aperture (cm²)", "detected power (μW)"},
   FrameTicks -> {{{#, Superscript[10, #]} & /@ (Table[x, {x, -2, 3}]), None},
     {Table[{tlr, Round[QuantityMagnitude[UnitConvert[EffectiveAperture /.
            assignments, "cm^2"]]]}, {tlr, .2, 1, .2}], None}}, PlotLabel ->
    StringJoin["varying distance and effective aperture\nwing size: ",
     ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
      TraditionalForm], "; wing reflectivity: ",
     ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
     "%\ndivergence angle: ", ToString[
      UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
      TraditionalForm], " or ", ToString[UnitConvert[
       DivergenceAngle /. assignments /. vals, "rad"], TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[
    ToString[UnitConvert[RangeAtInsect /. assignments /. vals, "m"],
     TraditionalForm], {rng, lb /. vals, ub /. vals,
     it /. vals}], LegendLabel → "distance"]]
```

Out[138]=

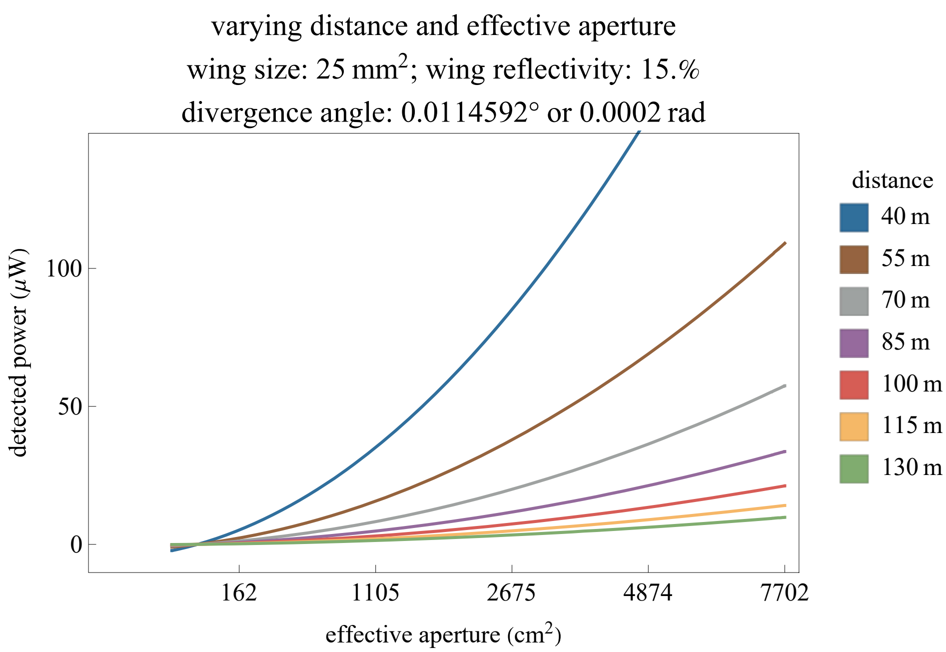


vary distance

In[139]:=
```
ClearAll[tp, cv]
cv = 1;
vals = {da → .0002, rlf → .15, ws → 25, lb → 40, ub → 130, it → 15};
Do[{tp[cv] = Plot[UnitConvert[DetectedPower /. assignments /. vals,
       "microWatts"], {tlr, .1016, 1}, PlotStyle → {c[cv]}],
    cv = cv + 1}, {rng, lb /. vals, ub /. vals, it /. vals}];
```

In[143]:=
```
plt[21] = Legended[Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → Automatic,
    FrameLabel → {"effective aperture (cm²)", "detected power (μW)"},
    PlotRange → {{0, 1}, {0, 300}},
    FrameTicks -> {{{#, #} & /@ (Table[x, {x, 0, 300, 50}]), None},
      {Table[{tlr, Round[QuantityMagnitude[UnitConvert[EffectiveAperture /.
            assignments, "cm^2"]]]}, {tlr, .2, 1, .2}], None}}, PlotLabel ->
     StringJoin["varying distance and effective aperture\nwing size: ",
      ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
       TraditionalForm], "; wing reflectivity: ",
      ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
      "%\ndivergence angle: ", ToString[
       UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
       TraditionalForm], " or ", ToString[UnitConvert[
        DivergenceAngle /. assignments /. vals, "rad"], TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[ToString[UnitConvert[
       RangeAtInsect /. assignments /. vals, "m"], TraditionalForm],
     {rng, lb /. vals, ub /. vals, it /. vals}], LegendLabel → "distance"]]
```

Out[143]=

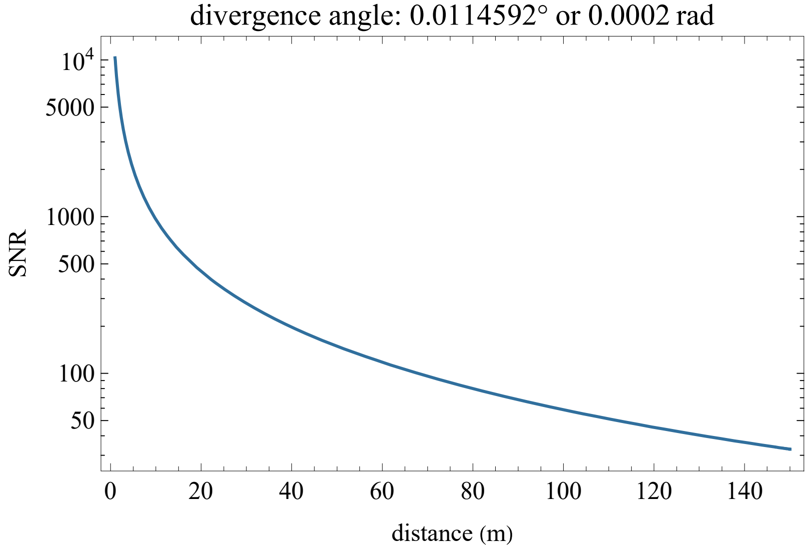


SNR vs distance

In[144]:=
```
assignments = {TelescopeOuterRadius -> Quantity[tlr, "m"] / 2,
   WingSize → UnitConvert[Quantity[ws, "mm^2"], "m^2"],
   RangeAtInsect → Quantity[rng, "m"],
   Reflectivity → Quantity[rlf, "DimensionlessUnit"],
   DivergenceAngle → Quantity[da, "Radians"]};
vals = {tlr -> .3048, rlf -> .15, ws -> 25, da -> 0.0002};
plt[5] = LogPlot[
   UnitConvert[SNR /. assignments /. vals, "DimensionlessUnit"], {rng, 1, 150},
   PlotLabel -> StringJoin["varying distance; effective aperture: ",
     ToString[UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"],
      TraditionalForm], "\nwing size: ",
     ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
      TraditionalForm], "; wing reflectivity: ",
     ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
     "%\ndivergence angle: ",
     ToString[UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
      TraditionalForm], " or ", ToString[UnitConvert[
       DivergenceAngle /. assignments /. vals, "rad"], TraditionalForm]],
   FrameLabel → {"distance (m)", "SNR"}, PlotStyle → {c[1]}]
```

Out[145]=

varying distance; effective aperture: 577.593 cm^2
wing size: 25 mm^2; wing reflectivity: 15.%
divergence angle: 0.0114592° or 0.0002 rad

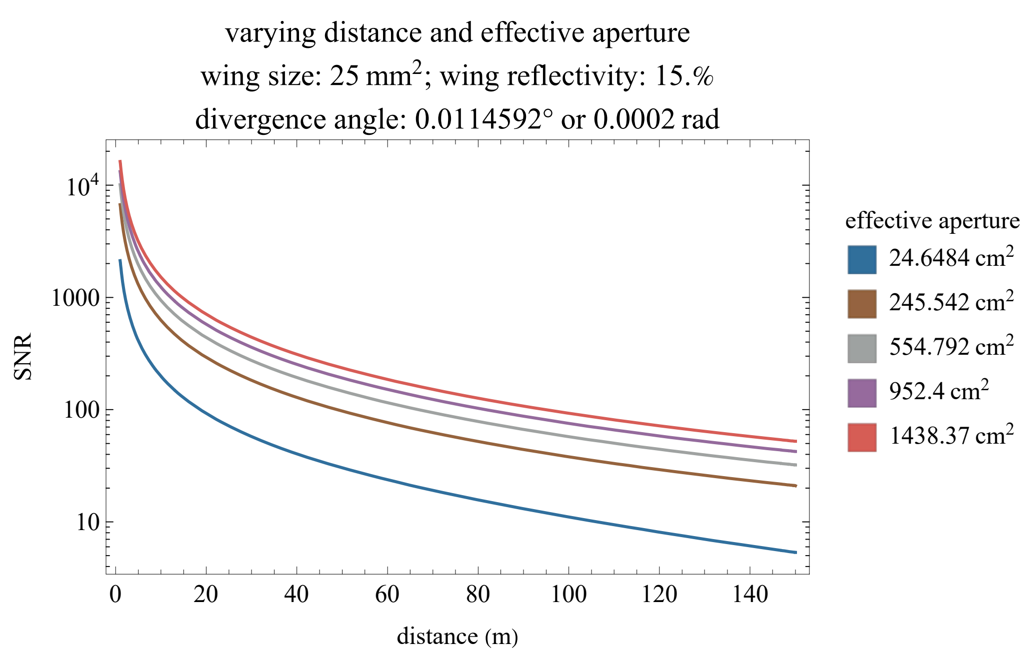


vary effective aperture

In[146]:=
```
ClearAll[tp, cv]
cv = 1;
vals = {da → .0002, rlf → .15, ws → 25, lb → 0.15, ub → .5, it → .075};
Do[{tp[cv] = LogPlot[UnitConvert[SNR /. assignments /. vals,
      "DimensionlessUnit"], {rng, rangelb, rangeub}, PlotStyle → {c[cv]}],
   cv = cv + 1}, {tlr, lb /. vals, ub /. vals, it /. vals}];
```

In[150]:=
```
plt[6] = Legended[Show[Table[tp[h], {h, 1, cv - 1}],
   PlotRange → All, FrameLabel → {"distance (m)", "SNR"}, PlotLabel ->
    StringJoin["varying distance and effective aperture\nwing size: ",
     ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
      TraditionalForm], "; wing reflectivity: ",
     ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
     "%\ndivergence angle: ", ToString[
      UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
      TraditionalForm], " or ", ToString[UnitConvert[
       DivergenceAngle /. assignments /. vals, "rad"], TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[
    ToString[UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"],
     TraditionalForm], {tlr, lb /. vals, ub /. vals, it /. vals}],
   LegendLabel → "effective aperture"]]
```

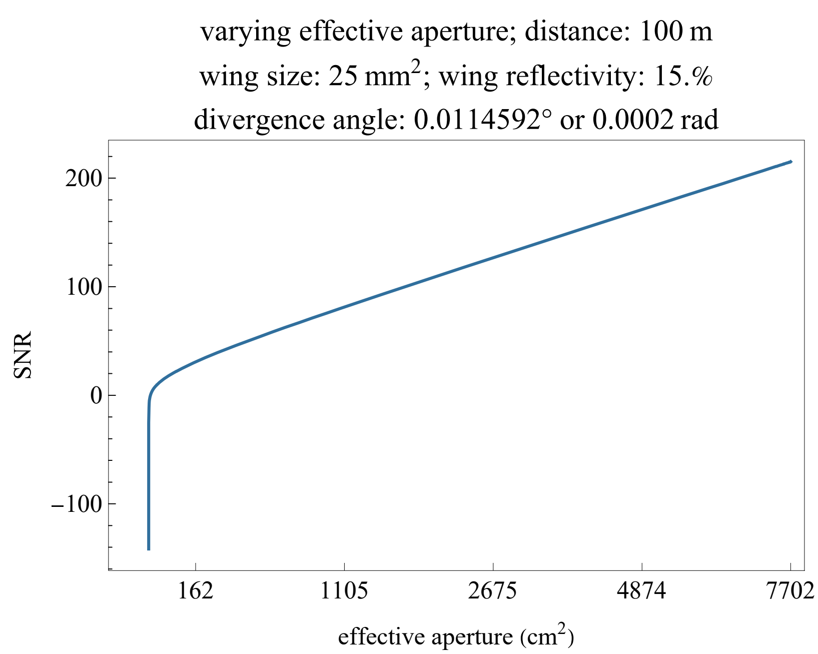
Out[150]=



■ SNR vs effective aperture

```
In[151]:= assignments = {TelescopeOuterRadius -> Quantity[tlr, "m"] / 2,
    WingSize → UnitConvert[Quantity[ws, "mm^2"], "m^2"],
    RangeAtInsect → Quantity[rng, "m"],
    Reflectivity → Quantity[rlf, "DimensionlessUnit"],
    DivergenceAngle → Quantity[da, "Radians"]};
  vals = {rng → 100, rlf -> .15, ws -> 25, da -> 0.0002};
  plt[7] = Plot[UnitConvert[SNR /. assignments /. vals, "DimensionlessUnit"],
    {tlr, .1016, 1}, FrameTicks -> {{All, None},
      {Table[{tlr, Round[QuantityMagnitude[UnitConvert[EffectiveAperture /.
              assignments, "cm^2"]]]}, {tlr, .2, 1, .2}], None}},
    PlotLabel -> StringJoin["varying effective aperture; distance: ",
      ToString[UnitConvert[RangeAtInsect /. assignments /. vals, "m"],
       TraditionalForm], "\nwing size: ",
      ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
       TraditionalForm], "; wing reflectivity: ",
      ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
      "%\ndivergence angle: ",
      ToString[UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
       TraditionalForm], " or ", ToString[UnitConvert[
        DivergenceAngle /. assignments /. vals, "rad"], TraditionalForm]],
    FrameLabel → {"effective aperture (cm²)", "SNR"},
    PlotStyle → {c[1]}]
```

Out[152]=

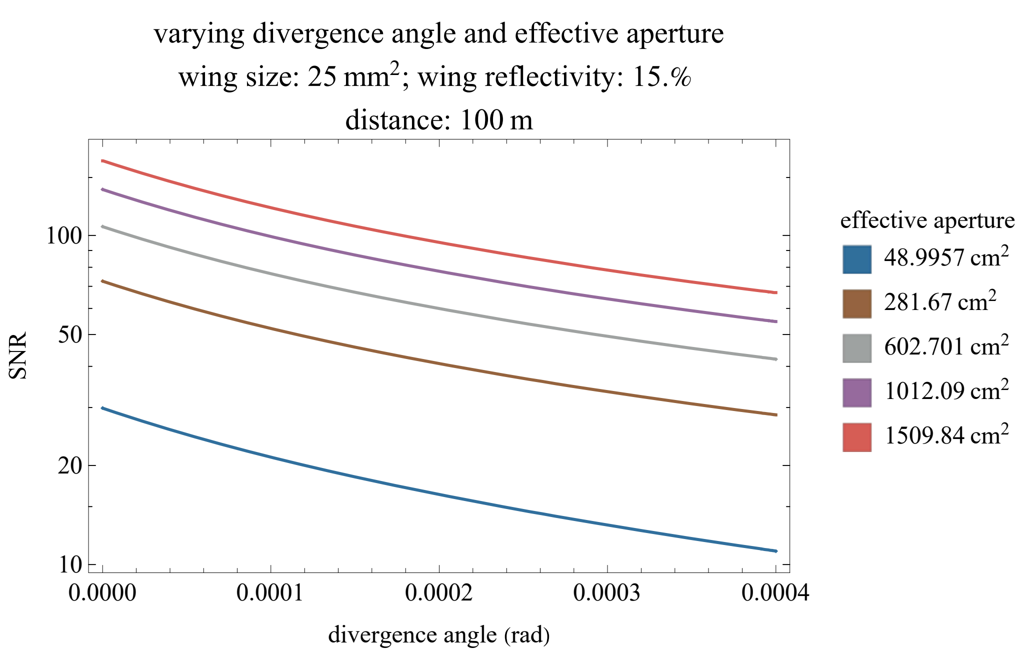


SNR vs divergence angle

In[153]:=
```
ClearAll[tp, cv]
cv = 1;
vals = {rng → 100, rlf → .15, ws → 25, lb → .16, ub → .5, it → .075};
Do[{tp[cv] = LogPlot[UnitConvert[SNR /. assignments /. vals,
      "DimensionlessUnit"], {da, 0, .0004}, PlotStyle → {c[cv]}],
   cv = cv + 1}, {tlr, lb /. vals, ub /. vals, it /. vals}];
```

In[157]:=
```
plt[8] = Legended[Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All,
   FrameLabel → {"divergence angle (rad)", "SNR"}, PlotLabel -> StringJoin[
     "varying divergence angle and effective aperture\nwing size: ",
     ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
      TraditionalForm], "; wing reflectivity: ",
     ToString[(Reflectivity /. assignments /. vals) * 100, TraditionalForm],
     "% \ndistance: ", ToString[UnitConvert[
       RangeAtInsect /. assignments /. vals, "m"], TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[
    ToString[UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"],
     TraditionalForm], {tlr, lb /. vals, ub /. vals, it /. vals}],
   LegendLabel → "effective aperture"]]
```

Out[157]=



Set SNR Plots

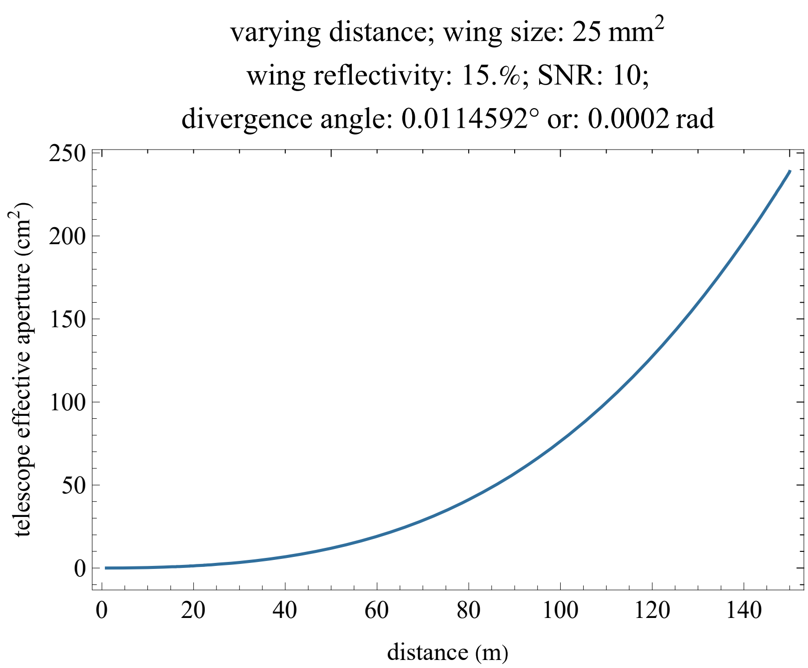
Solve for TelescopeOuterRadius

```
In[158]:= assignments = {WingSize → UnitConvert[Quantity[ws, "mm^2"], "m^2"],
            RangeAtInsect → Quantity[rng, "m"],
            Reflectivity → Quantity[rlf, "DimensionlessUnit"],
            DivergenceAngle → Quantity[da, "Radians"]};

In[159]:= sol = Solve[SNR == SNRv /. assignments, TelescopeOuterRadius][[4]];
```

vary distance

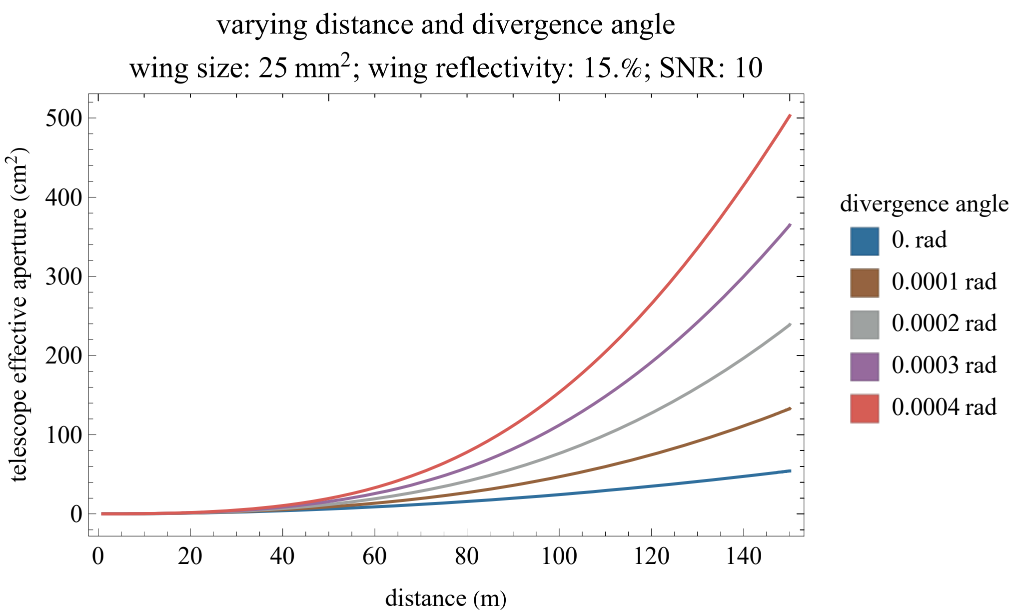
```
vals = {rlf -> .15, ws -> 25, da → .0002, cv → 1};
plt[9] = Plot[UnitConvert[EffectiveAperture /. sol /. vals, "cm^2"],
  {rng, rangelb, rangeub},
  PlotLabel -> StringJoin["varying distance; wing size: ",
    ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
     TraditionalForm], "\nwing reflectivity: ",
    ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
    "%; SNR: ", ToString[SNRv], ";\ndivergence angle: ",
    ToString[UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
     TraditionalForm], " or: ",
    ToString[DivergenceAngle /. assignments /. vals, TraditionalForm]],
  PlotStyle → {c[cv /. vals]}, FrameLabel →
   {"distance (m)", "telescope effective aperture (cm²)"}]
```



vary divergence angle

```
ClearAll[tp, cv]
cv = 1;
vals = {rlf → .15, ws → 25, lb → 0, ub → .0004, it → .0001};
Do[{tp[cv] = Plot[UnitConvert[EffectiveAperture /. sol /. vals, "cm^2"],
     {rng, rangelb, rangeub}, PlotStyle → {c[cv]},
     FrameLabel → {"distance (m)", "telescope effective aperture (cm²)"}],
   cv = cv + 1}, {da, lb /. vals, ub /. vals, it /. vals}];
```

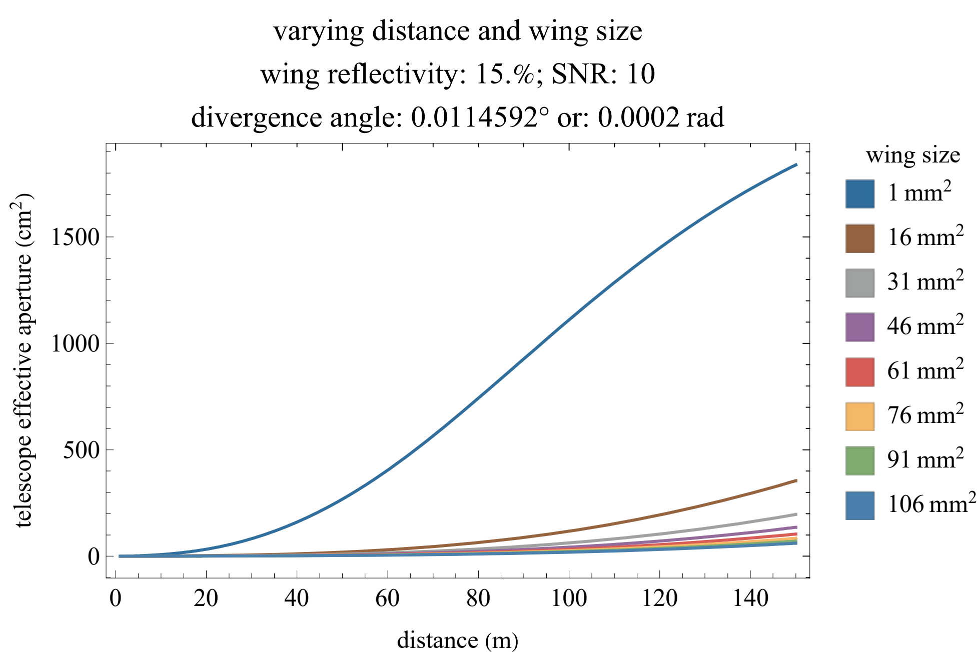
```
plt[10] =
 Legended[Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All, PlotLabel ->
    StringJoin["varying distance and divergence angle\nwing size: ",
     ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
      TraditionalForm], "; wing reflectivity: ",
     ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
     "%; SNR: ", ToString[SNRv]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[
    ToString[UnitConvert[DivergenceAngle /. assignments /. vals, "rad"],
     TraditionalForm], {da, lb /. vals, ub /. vals, it /. vals}],
   LegendLabel → "divergence angle"]]
```



vary wing size

```
ClearAll[tp, cv]
cv = 1;
vals = {rlf → .15, da → .0002, lb → 1, ub → 106, it → 15};
Do[{tp[cv] = Plot[UnitConvert[EffectiveAperture /. sol /. vals, "cm^2"],
     {rng, rangelb, rangeub}, PlotStyle → {c[cv]},
     FrameLabel → {"distance (m)", "telescope effective aperture (cm²)"}],
   cv = cv + 1}, {ws, lb /. vals, ub /. vals, it /. vals}];
```

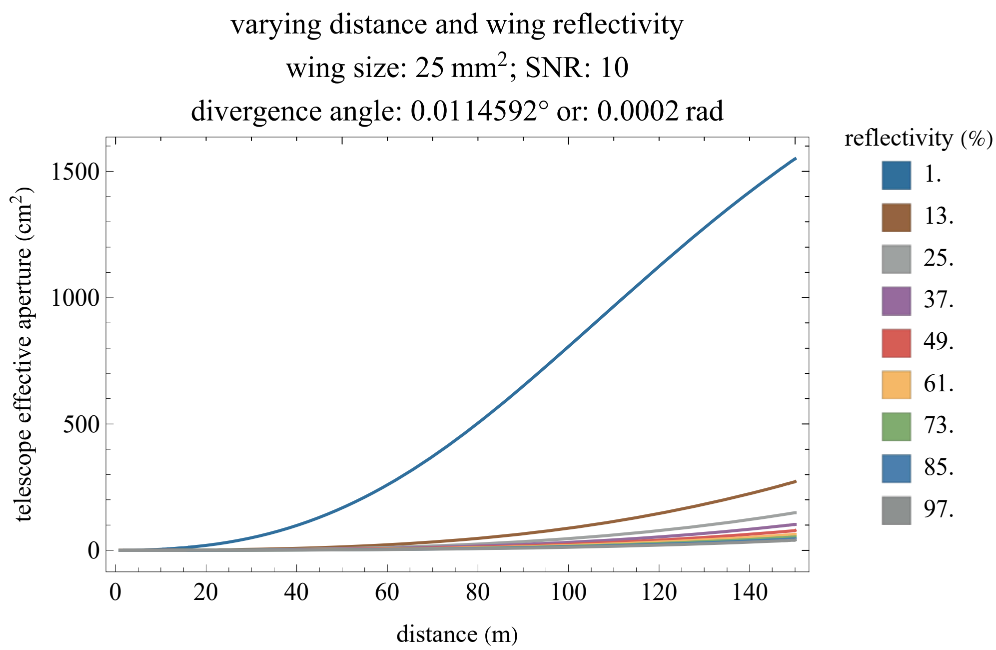
```
plt[11] =
 Legended[Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All, PlotLabel ->
    StringJoin["varying distance and wing size\n", "wing reflectivity: ",
     ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
     "%; SNR: ", ToString[SNRv], "\ndivergence angle: ",
     ToString[UnitConvert[DivergenceAngle /. assignments /. vals,
       "degrees"], TraditionalForm], " or: ",
     ToString[DivergenceAngle /. assignments /. vals, TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[ToString[
     UnitConvert[WingSize /. assignments /. vals, "mm^2"], TraditionalForm],
    {ws, lb /. vals, ub /. vals, it /. vals}], LegendLabel → "wing size"]]
```



varying reflectivity

```
ClearAll[tp, cv]
cv = 1;
vals = {ws → 25, da → .0002, lb → .01, ub → 1.0, it → .12};
Do[{tp[cv] = Plot[UnitConvert[EffectiveAperture /. sol /. vals, "cm^2"],
     {rng, rangelb, rangeub}, PlotStyle → {c[cv]},
     FrameLabel → {"distance (m)", "telescope effective aperture (cm²)"}],
   cv = cv + 1}, {rlf, lb /. vals, ub /. vals, it /. vals}];
```

```
plt[12] =
 Legended[Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All, PlotLabel ->
    StringJoin["varying distance and wing reflectivity\n", "wing size: ",
     ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
      TraditionalForm], "; SNR: ", ToString[SNRv], "\ndivergence angle: ",
     ToString[UnitConvert[DivergenceAngle /. assignments /. vals,
       "degrees"], TraditionalForm], " or: ",
     ToString[DivergenceAngle /. assignments /. vals, TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}],
   Table[ToString[UnitConvert[(Reflectivity /. assignments /. vals) * 100,
      "DimensionlessUnit"], TraditionalForm], {rlf, lb /. vals,
     ub /. vals, it /. vals}], LegendLabel → "reflectivity (%)"]]
```



Solve for WingSize

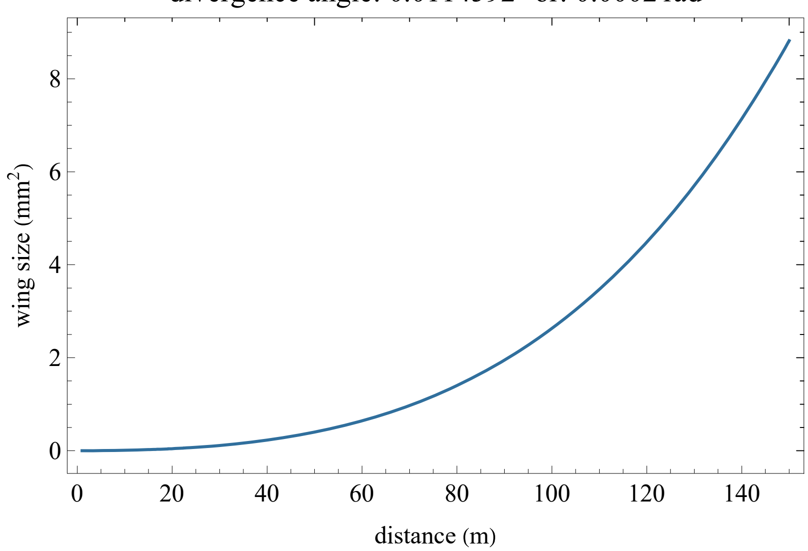
```
assignments = {TelescopeOuterRadius -> Quantity[tlr, "m"] / 2,
   RangeAtInsect → Quantity[rng, "m"],
   Reflectivity → Quantity[rlf, "DimensionlessUnit"],
   DivergenceAngle → Quantity[da, "Radians"]};

sol = Solve[SNR == SNRv /. assignments, WingSize][[2]];
```

varying distance

```
vals = {da → .0002, rlf → .15, tlr → .3048};
plt[13] =
 Plot[UnitConvert[WingSize /. sol /. vals, "mm^2"], {rng, rangelb, rangeub},
  PlotLabel -> StringJoin["varying distance; effective aperture: ",
    ToString[UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"],
     TraditionalForm], "\nwing reflectivity: ",
    ToString[(100 * Reflectivity /. assignments /. vals), TraditionalForm],
    "%; SNR: ", ToString[SNRv], ";\ndivergence angle: ",
    ToString[UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
     TraditionalForm], " or: ",
    ToString[DivergenceAngle /. assignments /. vals, TraditionalForm]],
  PlotStyle → {c[1]}, FrameLabel → {"distance (m)", "wing size (mm²)"}]
```

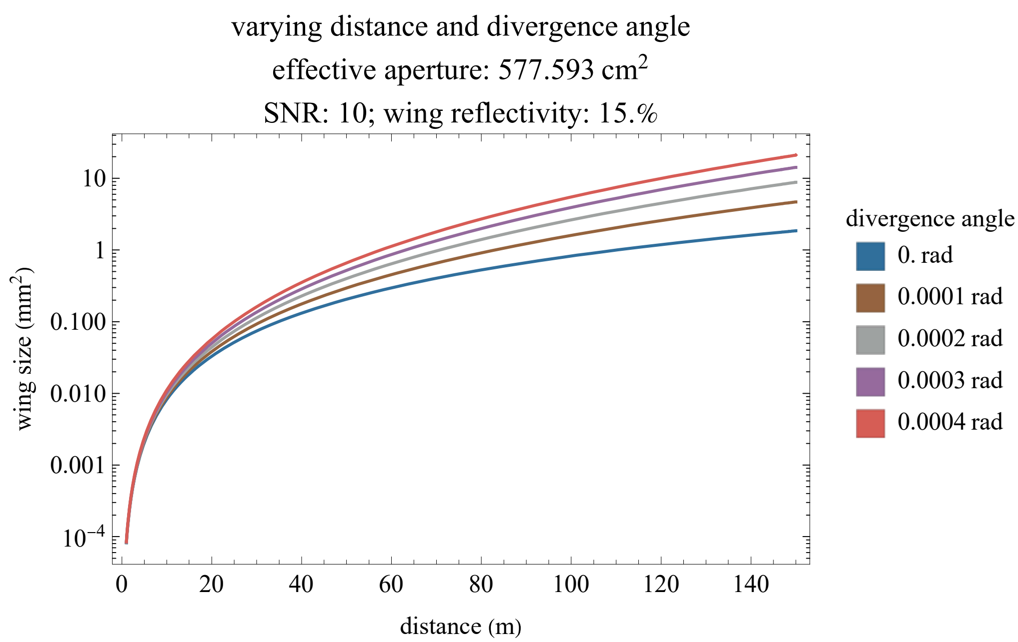
varying distance; effective aperture: 577.593 cm^2
wing reflectivity: 15.%; SNR: 10;
divergence angle: 0.0114592° or: 0.0002 rad



varying divergence angle

```
ClearAll[tp, cv]
cv = 1;
vals = {tlr → .3048, rlf → .15, lb → .0, ub → .0004, it → .0001};
Do[{tp[cv] = LogPlot[UnitConvert[WingSize /. sol /. vals, "mm^2"],
     {rng, rangelb, rangeub}, PlotStyle → {c[cv]},
     FrameLabel → {"distance (m)", "wing size (mm²)"}],
   cv = cv + 1}, {da, lb /. vals, ub /. vals, it /. vals}];
```

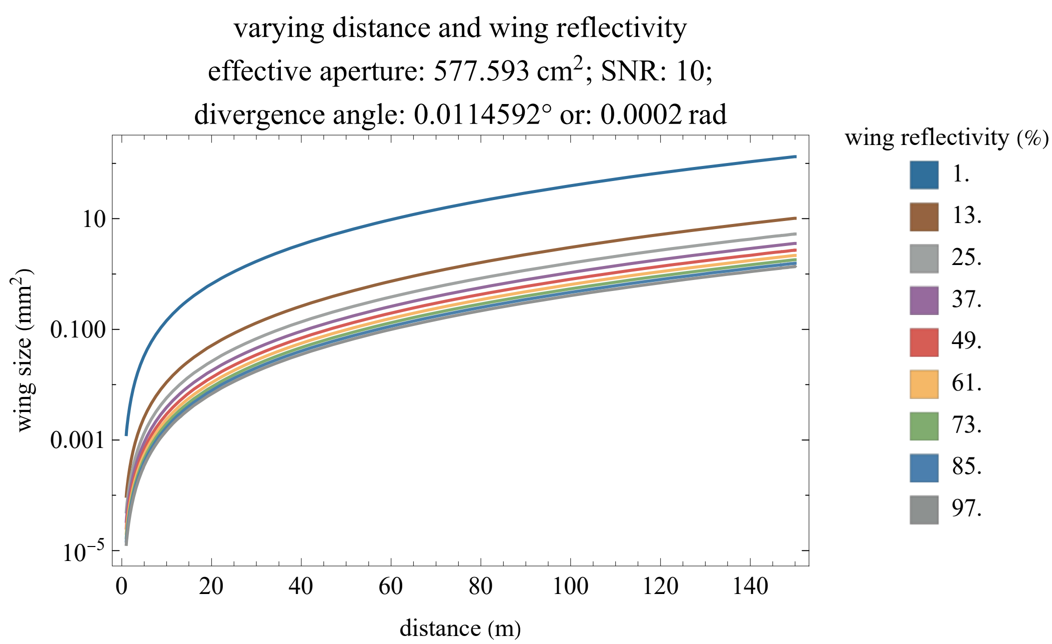
```
plt[14] = Legended[
  Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All, PlotLabel -> StringJoin[
     "varying distance and divergence angle\n", "effective aperture: ",
     ToString[UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"],
      TraditionalForm], "\nSNR: ", ToString[SNRv], "; wing reflectivity: ",
     ToString[(Reflectivity /. assignments /. vals) * 100, TraditionalForm],
     "%"]], SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[
    ToString[UnitConvert[DivergenceAngle /. assignments /. vals, "Radians"],
     TraditionalForm], {da, lb /. vals, ub /. vals, it /. vals}],
   LegendLabel → "divergence angle"]]
```



varying reflectivity

```
ClearAll[tp, cv]
cv = 1;
vals = {tlr → .3048, da → .0002, lb → .01, ub → 1.0, it → .12};
Do[{tp[cv] = LogPlot[UnitConvert[WingSize /. sol /. vals, "mm^2"],
     {rng, rangelb, rangeub}, PlotStyle → {c[cv]},
     FrameLabel → {"distance (m)", "wing size (mm²)"}],
   cv = cv + 1}, {rlf, lb /. vals, ub /. vals, it /. vals}];
```

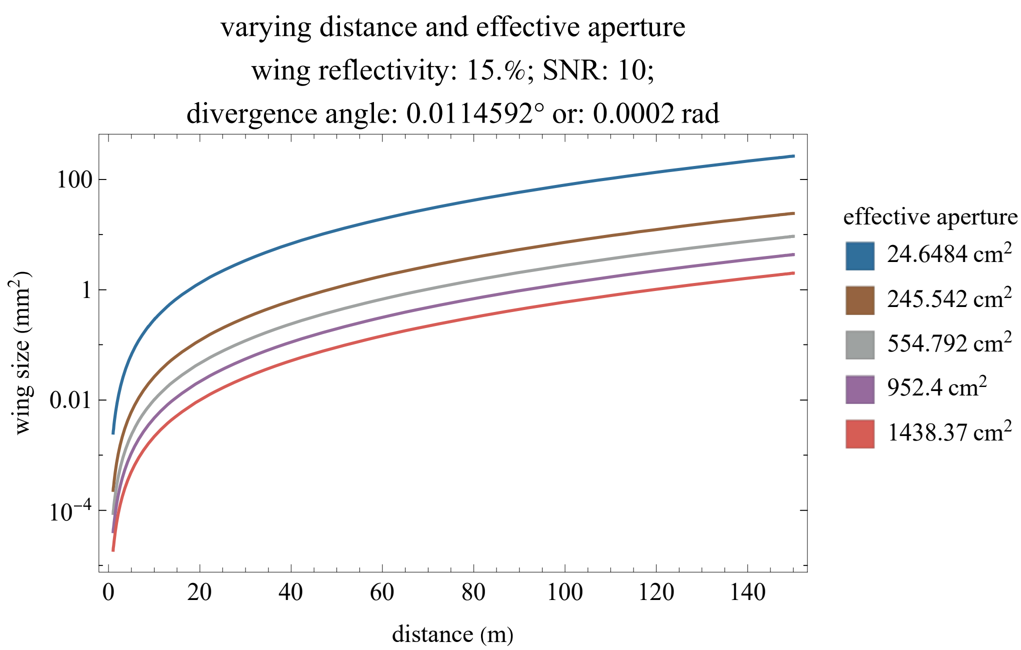
```
plt[15] = Legended[
  Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All, PlotLabel -> StringJoin[
     "varying distance and wing reflectivity\n", "effective aperture: ",
     ToString[UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"],
      TraditionalForm], "; SNR: ", ToString[SNRv], ";\ndivergence angle: ",
     ToString[UnitConvert[DivergenceAngle /. assignments /. vals,
       "degrees"], TraditionalForm], " or: ",
     ToString[DivergenceAngle /. assignments /. vals, TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[ToString[UnitConvert[
       (Reflectivity /. assignments /. vals) * 100, "DimensionlessUnit"],
     TraditionalForm], {rlf, lb /. vals, ub /. vals, it /. vals}],
   LegendLabel → "wing reflectivity (%)"]]
```



varying effective aperture

```
ClearAll[tp, cv]
cv = 1;
vals = {rlf → .15, da → .0002, lb → .15, ub → .5, it → .075};
Do[{tp[cv] = LogPlot[UnitConvert[WingSize /. sol /. vals, "mm^2"],
     {rng, rangelb, rangeub}, PlotStyle → {c[cv]},
     FrameLabel → {"distance (m)", "wing size (mm²)"}],
   cv = cv + 1}, {tlr, lb /. vals, ub /. vals, it /. vals}];
```

```
plt[16] = Legended[
  Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All, PlotLabel -> StringJoin[
     "varying distance and effective aperture\nwing reflectivity: ",
     ToString[UnitConvert[(Reflectivity /. assignments /. vals) * 100,
       "DimensionlessUnit"], TraditionalForm],
     "%; SNR: ", ToString[SNRv], ";\ndivergence angle: ",
     ToString[UnitConvert[DivergenceAngle /. assignments /. vals,
       "degrees"], TraditionalForm], " or: ",
     ToString[DivergenceAngle /. assignments /. vals, TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[ToString[
     UnitConvert[(EffectiveAperture /. assignments /. vals), "cm^2"],
     TraditionalForm], {tlr, lb /. vals, ub /. vals, it /. vals}],
   LegendLabel → "effective aperture"]]
```



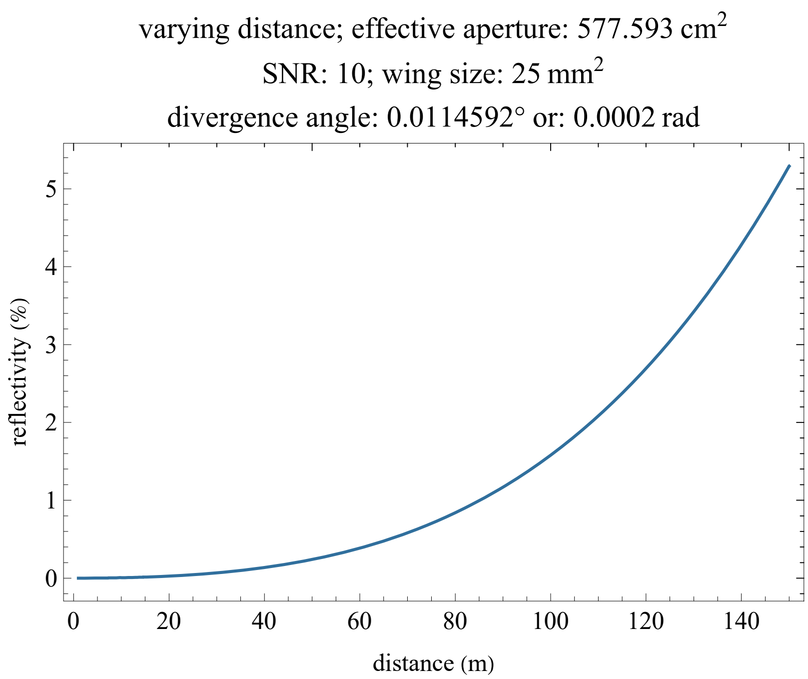
Solve for Wing Reflectivity

```
assignments = {TelescopeOuterRadius -> Quantity[tlr, "m"] / 2,
   RangeAtInsect → Quantity[rng, "m"],
   DivergenceAngle → Quantity[da, "Radians"],
   WingSize → UnitConvert[Quantity[ws, "mm^2"], "m^2"]};

sol = Solve[SNR == SNRv /. assignments, Reflectivity][[2]];
```

varying distance

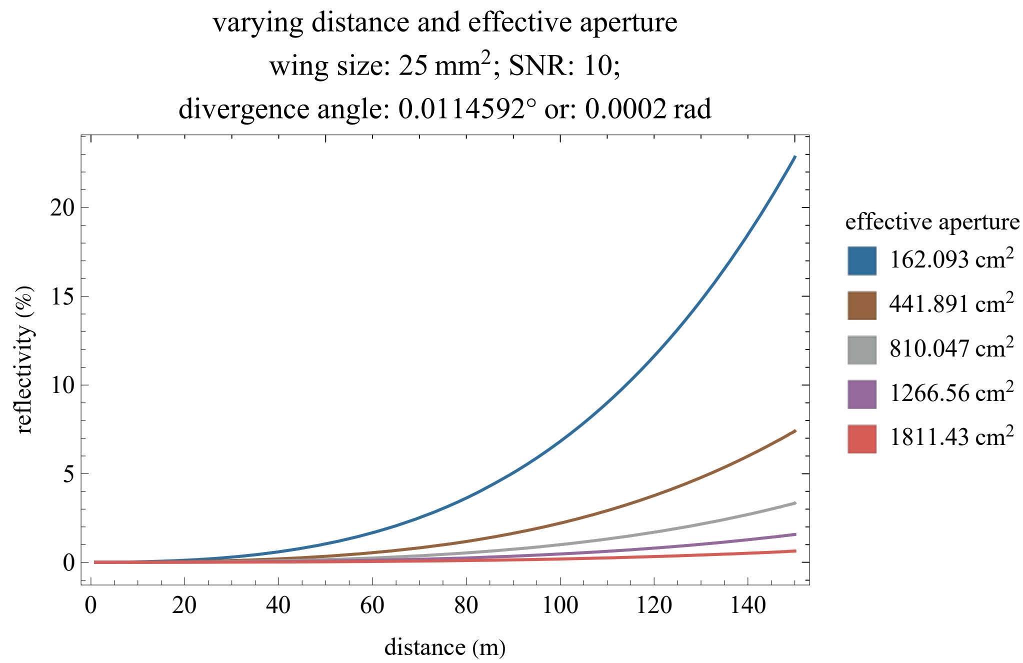
```
vals = {ws → 25, da → .0002, tlr → .3048};
plt[17] = Plot[UnitConvert[(Reflectivity /. sol /. vals) * 100,
    "DimensionlessUnit"], {rng, rangelb, rangeub},
   PlotLabel -> StringJoin["varying distance; effective aperture: ",
     ToString[UnitConvert[EffectiveAperture /. assignments /. vals, "cm^2"],
      TraditionalForm], "\nSNR: ", ToString[SNRv], "; wing size: ",
     ToString[UnitConvert[WingSize /. assignments /. vals, "mm^2"],
      TraditionalForm], "\ndivergence angle: ",
     ToString[UnitConvert[DivergenceAngle /. assignments /. vals, "degrees"],
      TraditionalForm], " or: ",
     ToString[DivergenceAngle /. assignments /. vals, TraditionalForm]],
   PlotStyle → {c[1]}, FrameLabel → {"distance (m)", "reflectivity (%)"}]
```



varying effective aperture

```
ClearAll[tp, cv]
cv = 1;
vals = {ws → 25, da → .0002, lb → .2, ub → .5, it → .075};
Do[{tp[cv] = Plot[UnitConvert[(Reflectivity /. sol /. vals) * 100,
      "DimensionlessUnit"], {rng, rangelb, rangeub}, PlotStyle → {c[cv]},
     FrameLabel → {"distance (m)", "reflectivity (%)"}],
   cv = cv + 1}, {tlr, lb /. vals, ub /. vals, it /. vals}];
```

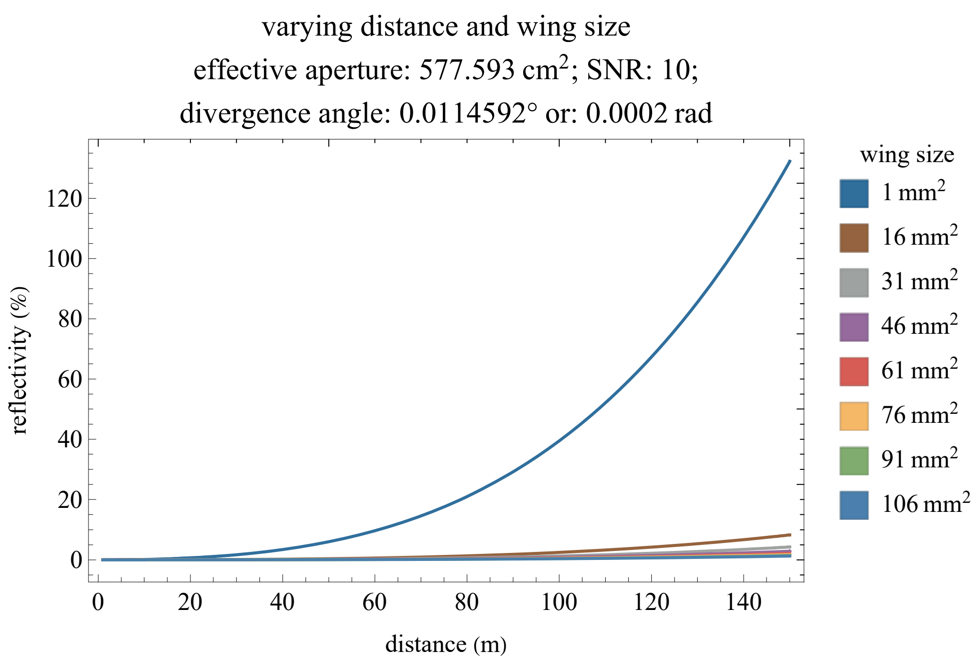
```
plt[18] =
 Legended[Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All, PlotLabel ->
    StringJoin["varying distance and effective aperture\nwing size: ",
     ToString[UnitConvert[(WingSize /. assignments /. vals), "mm^2"],
      TraditionalForm], "; SNR: ", ToString[SNRv], ";\ndivergence angle: ",
     ToString[UnitConvert[DivergenceAngle /. assignments /. vals,
       "degrees"], TraditionalForm], " or: ",
     ToString[DivergenceAngle /. assignments /. vals, TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[ToString[
     UnitConvert[(EffectiveAperture /. assignments /. vals), "cm^2"],
     TraditionalForm], {tlr, lb /. vals, ub /. vals, it /. vals}],
   LegendLabel → "effective aperture"]]
```



varying wing size

```
ClearAll[tp, cv]
cv = 1;
vals = {tlr → .3048, da → .0002, lb → 1, ub → 106, it → 15};
Do[{tp[cv] = Plot[UnitConvert[(Reflectivity /. sol /. vals) * 100,
      "DimensionlessUnit"], {rng, rangelb, rangeub}, PlotStyle → {c[cv]},
     FrameLabel → {"distance (m)", "reflectivity (%)"}],
   cv = cv + 1}, {ws, lb /. vals, ub /. vals, it /. vals}];
```

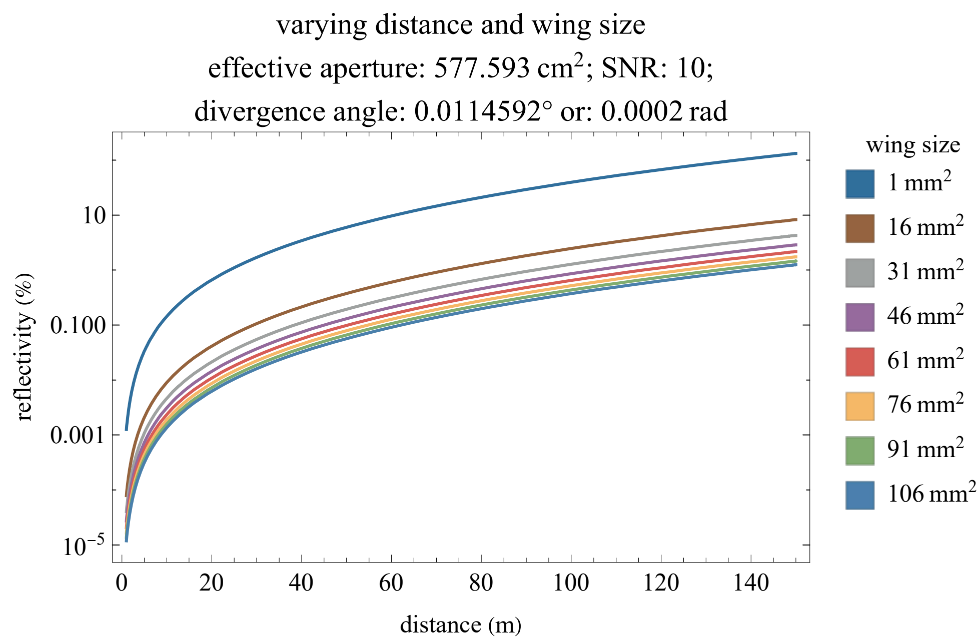
```
plt[19] = Legended[
  Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All, PlotLabel -> StringJoin[
     "varying distance and wing size\neffective aperture: ", ToString[
      UnitConvert[(EffectiveAperture /. assignments /. vals), "cm^2"],
      TraditionalForm], "; SNR: ", ToString[SNRv], ";\ndivergence angle: ",
     ToString[UnitConvert[DivergenceAngle /. assignments /. vals,
       "degrees"], TraditionalForm], " or: ",
     ToString[DivergenceAngle /. assignments /. vals, TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[ToString[UnitConvert[
       (WingSize /. assignments /. vals), "mm^2"], TraditionalForm],
     {ws, lb /. vals, ub /. vals, it /. vals}], LegendLabel → "wing size"]]
```



varying wing size log

```
ClearAll[tp, cv]
cv = 1;
vals = {tlr → .3048, da → .0002, lb → 1, ub → 106, it → 15};
Do[{tp[cv] = LogPlot[UnitConvert[(Reflectivity /. sol /. vals) * 100,
      "DimensionlessUnit"], {rng, rangelb, rangeub}, PlotStyle → {c[cv]},
     FrameLabel → {"distance (m)", "reflectivity (%)"}],
   cv = cv + 1}, {ws, lb /. vals, ub /. vals, it /. vals}];
```

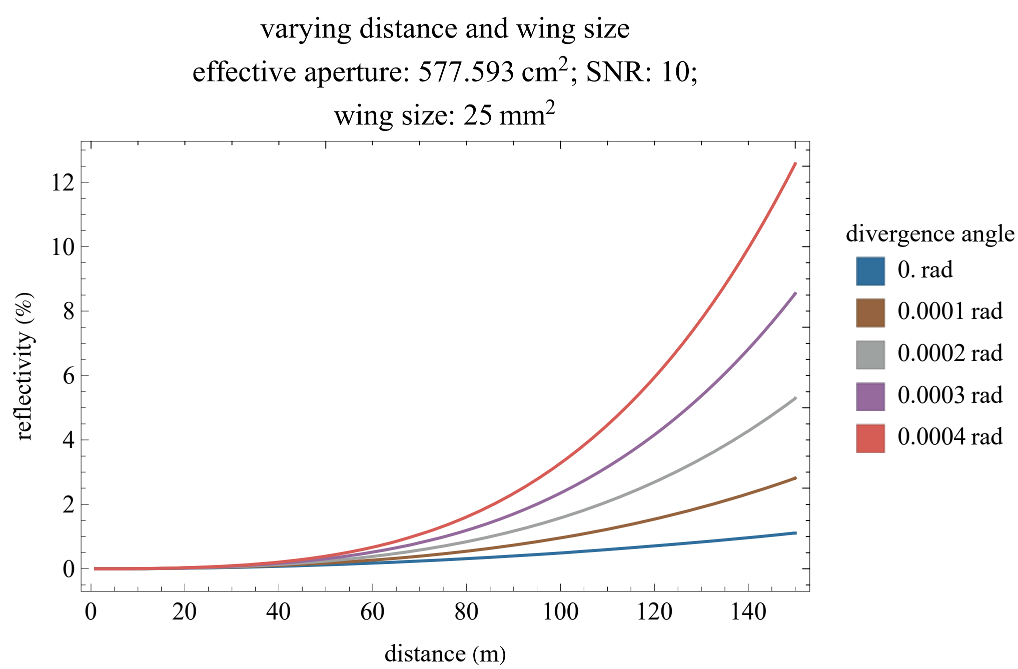
```
plt[23] = Legended[
  Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All, PlotLabel -> StringJoin[
     "varying distance and wing size\neffective aperture: ", ToString[
      UnitConvert[(EffectiveAperture /. assignments /. vals), "cm^2"],
      TraditionalForm], "; SNR: ", ToString[SNRv], ";\ndivergence angle: ",
     ToString[UnitConvert[DivergenceAngle /. assignments /. vals,
       "degrees"], TraditionalForm], " or: ",
     ToString[DivergenceAngle /. assignments /. vals, TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[ToString[UnitConvert[
      (WingSize /. assignments /. vals), "mm^2"], TraditionalForm],
    {ws, lb /. vals, ub /. vals, it /. vals}], LegendLabel → "wing size"]]
```



varying divergence angle

```
ClearAll[tp, cv]
cv = 1;
vals = {tlr → .3048, ws → 25, lb → 0, ub → .0004, it → .0001};
Do[{tp[cv] = Plot[UnitConvert[(Reflectivity /. sol /. vals) * 100,
      "DimensionlessUnit"], {rng, rangelb, rangeub}, PlotStyle → {c[cv]},
     FrameLabel → {"distance (m)", "reflectivity (%)"}],
   cv = cv + 1}, {da, lb /. vals, ub /. vals, it /. vals}];
```

```
plt[20] =
 Legended[Show[Table[tp[h], {h, 1, cv - 1}], PlotRange → All, PlotLabel ->
    StringJoin["varying distance and wing size\neffective aperture: ",
     ToString[UnitConvert[(EffectiveAperture /. assignments /. vals),
       "cm^2"], TraditionalForm], "; SNR: ",
     ToString[SNRv], ";\nwing size: ", ToString[UnitConvert[
       WingSize /. assignments /. vals, "mm^2"], TraditionalForm]]],
  SwatchLegend[Table[c[x], {x, 1, cv - 1}], Table[ToString[
     UnitConvert[(DivergenceAngle /. assignments /. vals), "radians"],
     TraditionalForm], {da, lb /. vals, ub /. vals, it /. vals}],
   LegendLabel → "divergence angle"]]
```



Eye Safety

```
assignments = {TelescopeOuterRadius -> Quantity[tlr, "m"] / 2,
   RangeAtInsect → Quantity[rng, "m"],
   Reflectivity → Quantity[rlf, "DimensionlessUnit"],
   DivergenceAngle → Quantity[da, "Radians"],
   WingSize → Quantity[ws, "mm^2"]};
vals = {tlr → .3048, ws → 25, rng → 100, rlf → .15, da → .0002};

{tlr → 0.3048`, ws → 25, rng → 100, rlf → 0.15`, da → 0.0002`}

{tlr → 0.3048, ws → 25, rng → 100, rlf → 0.15, da → 0.0002}
```

```
pulsereprate = Quantity[9, "kHz"]
```

9 kHz

```
Time = Quantity[0.25, "s"]
```

0.25 s

```
MPE = Quantity[1.8 * Time^(3 / 4) / Quantity[1, "s^(3/4)"] * 10^-3, "J/cm^2"]
```

$0.000636396\ \mathrm{J/cm^2}$

```
n = UnitConvert[pulsereprate * Time, "DimensionlessUnit"]
```

2250.

```
Cp = n^(-1 / 4)
```

0.145196

```
MPEwCp = ScientificForm[MPE * Cp]
```

$9.24021 \times 10^{-5}\ \mathrm{J/cm^2}$

```
MPEwCW = MPE / n // N
```

$2.82843 \times 10^{-7}\ \mathrm{J/cm^2}$

```
ES = UnitConvert[
  LaserEnergy / (((BeamDiameter /. assignments /. vals) / 2)^2 * Pi), "J/cm^2"]
```

$4.63297 \times 10^{-8}\ \mathrm{J/cm^2}$

Note that ES < MPEwCW ⇒ eye safety

Overlap

■ Telescope FOV

```
In[57]:= x = 3.7;
         y = 13.0;
         EPL = 4 + 635 + 631 + 660 + 17.391 + 11.740 + 20.0 + 5.34 + 46.112
Out[59]= 2030.58
```

In[60]:=
```
FOVx = ArcTan[(x / 2) / EPL]
FOVy = ArcTan[(y / 2) / EPL]
FOVd = ArcTan[Sqrt[(x / 2) ^2 + (y / 2) ^2] / EPL]
```

Out[60]= 0.000911068

Out[61]= 0.00320104

Out[62]= 0.00332817

In[63]:=
```
50.8 * Tan[Quantity[Pi / 2 - FOVd, "radians"]]
```

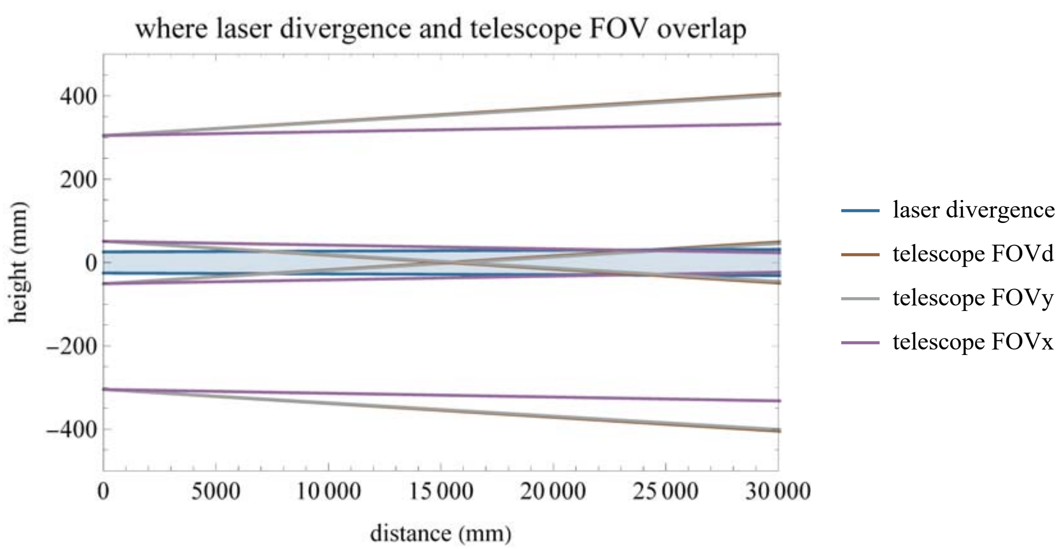
Out[63]= 15 263.6

In[64]:=
```
25.4 / (Tan[Quantity[FOVd, "radians"]] + Tan[Quantity[.0002, "radians"]])
```

Out[64]= 7199.18

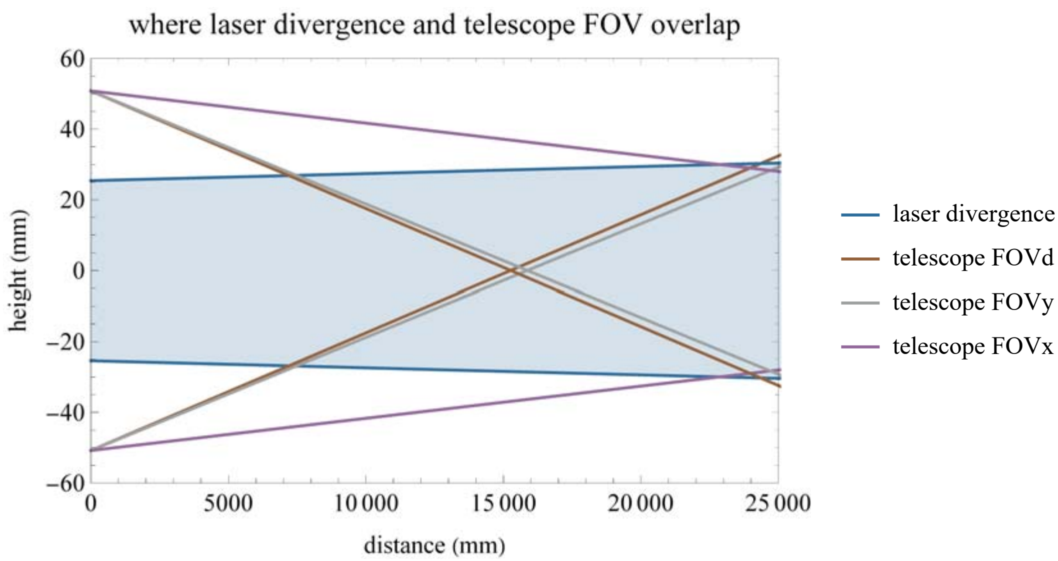
In[65]:=
```
plt[25] = Plot[
  {(Tan[-0.0002] * xt) - 25.4, (Tan[FOVd] * xt) - 50.8, (Tan[FOVy] * xt) - 50.8,
   (Tan[FOVx] * xt) - 50.8, (Tan[0.0002] * xt) + 25.4, (Tan[-FOVd] * xt) + 50.8,
   (Tan[-FOVy] * xt) + 50.8, (Tan[-FOVx] * xt) + 50.8, (Tan[-FOVd] * xt) - 304.8,
   (Tan[-FOVy] * xt) - 304.8, (Tan[-FOVx] * xt) - 304.8,
   (Tan[FOVd] * xt) + 304.8, (Tan[FOVy] * xt) + 304.8, (Tan[FOVx] * xt) + 304.8},
  {xt, 0, 50000}, PlotStyle → {c[1], c[2], c[3], c[4], c[1],
    c[2], c[3], c[4], c[2], c[3], c[4], c[2], c[3], c[4]},
  FrameLabel → {"distance (mm)", "height (mm)"}, Filling → {1 → {5}},
  PlotRange → {{0, 30000}, {-500, 500}}, PlotLegends -> {"laser divergence",
    "telescope FOVd", "telescope FOVy", "telescope FOVx"},
  PlotLabel → "where laser divergence and telescope FOV overlap"]
```

Out[65]=



In[66]:=
```
plt[24] = Plot[{(Tan[-0.0002] * xt) - 25.4, (Tan[FOVd] * xt) - 50.8,
    (Tan[FOVy] * xt) - 50.8, (Tan[FOVx] * xt) - 50.8, (Tan[0.0002] * xt) + 25.4,
    (Tan[-FOVd] * xt) + 50.8, (Tan[-FOVy] * xt) + 50.8, (Tan[-FOVx] * xt) + 50.8,
    (Tan[-FOVd] * xt) - 304.8, (Tan[-FOVy] * xt) - 304.8,
    (Tan[-FOVx] * xt) - 304.8, (Tan[FOVd] * xt) + 304.8,
    (Tan[FOVy] * xt) + 304.8, (Tan[FOVx] * xt) + 304.8}, {xt, 0, 50000},
  PlotStyle → {c[1], c[2], c[3], c[4], c[1], c[2], c[3], c[4], c[2],
    c[3], c[4], c[2], c[3], c[4]}, PlotRange → {{0, 25000}, {-60, 60}},
  FrameLabel → {"distance (mm)", "height (mm)"}, Filling → {1 → {5}},
  PlotLegends -> {"laser divergence",
    "telescope FOVd", "telescope FOVy", "telescope FOVx"},
  PlotLabel → "where laser divergence and telescope FOV overlap"]
```

Out[66]=



save

```
(*Do[Export[fullfilecld<>"plt"<>ToString[x]<>".png",plt[x]],{x,1,25,1}];
Do[Export[fullfileloc<>"plt"<>ToString[x]<>".png",plt[x]],{x,1,25,1}];*)
```