Filtered Photodiodes for Laser Wavelength Metrology

Nils T. Otterstrom

A senior thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Bachelor of Science

Dallin S. Durfee, Advisor

Department of Physics and Astronomy

Brigham Young University

April 2015

ABSTRACT

Filtered Photodiodes for Laser Wavelength Metrology

Nils T. Otterstrom
Department of Physics and Astronomy, BYU
Bachelor of Science

Many physics laboratories rely on accurate wavelength meters for metrology applications. Unfortunately, traditional instruments for laser spectroscopy tend to be expensive and fragile. We demonstrate a method for laser wavelength metrology using a simple and inexpensive color sensor made of an array of filtered photodiodes. The photocurrents from photodiodes under four colors of filters are measured digitally using an external microprocessor. With this data, we calculate the wavelength of a laser using a set of calibrated spectral response curves and a least squares error algorithm. After analyzing and controlling thin film interference effects in the sensor, we report an accuracy of 0.0053 nm (on the range of 460.79 nm to 460.89 nm) and a stability of 0.0039 nm over 41.5 hours (at 460.85 nm).

Keywords: wavelength meter, laser spectroscopy

ACKNOWLEDGMENTS

First of all, I would like to thank the Brigham Young University Physics and Astronomy department for providing and promoting the opportunity for undergraduates such as myself to work on exciting research. Also, I want to thank Dr. Dallin Durfee for his constant effort to personally assist, direct, and inspire my work. He has been instrumental in my education and goals as a physicist. I would like to also thank other members of my research group: Tyler Jones, with whom I have collaborated for the entirety of the project and Jarom Jackson, who has made many contributions to this research project, especially the design of the algorithm. I also wish to recognize James Archibald for coming up with this idea as a research project.

I also gratefully acknowledge funding from the National Science Foundation (NSF grant number 1205736), the Brigham Young University College of Physical and Mathematical Sciences, and the Office of Research and Creative Studies.

Contents

List of Figures

Chapter 1

Introduction

1.1 Wavelength metrology: a filtered photodiode approach

Wavelength metrology plays a fundamental role in various fields, from analyzing the galaxy's chemical makeup to locking a laser [1, 2]. In atomic and optical physics laboratories, scientists use wavelength meters to characterize and monitor their lasers. Unfortunately, many commercially available scientific instruments for wavelength detection can be expensive, cumbersome, or fragile.

My research aims to develop a new method for laser wavelength detection that could reduce expenditures and maintain accuracy for applications in laser spectroscopy. Using an inexpensive color sensor which is overly-precise for its intended applications in consumer electronics [3], we demonstrate a robust, quick, and compact method for determining the wavelength of light.

1.2 Research trends

For the last several decades, diffraction grating spectrometers and interferometer wavelength meters have been the industry standard for commercial wavelength meters. Although these technologies have been refined and optimized, they can be expensive and have their respective limitations.

Michelson-Morley interferometric wavelength meters are relatively accurate (about 0.005 nm or better), but cost anywhere from $5,000 to $ 10,000 [4]. Also, they rely on moving parts, making them slow and fragile [5]. Diffraction grating spectrometers have far less resolution, but they enable full spectrum measurements. These instruments come in a wide range of costs and accuracy, but require careful alignment [6].

In light of present commercial limitations, precision laser wavelength detection is an exciting area of current research. In the recent past, optical physicists have achieved important milestones in this field. For example, a portion of the 2005 Nobel prize in physics was granted for the development of an optical frequency comb [7]. Using pulsed lasers, optical scientists produce a spectrum of equally spaced frequency standards — enabling absolute wavelength measurements by analyzing beat frequencies. This technology is now commercially available with an accuracy of 1 part in 100 trillion [8]. However, the instruments cost around $250,000 and are physically large and unwieldy [9].

Other scientists have recently developed a method for wavelength metrology using multimode optical fibers to create a wavelength-dependent speckle pattern. With a 20 meter long fiber, they can resolve two laser lines separated by 0.008 nm [10]. Still others are developing methods for building on-chip spectrometers [11].

1.3 Intended applications of the filtered photodiode approach

In the Matter-Wave Inteferometry Laboratory at Brigham Young University, we use wavelength meters to tune diode lasers within 10 GHz (about 0.01 nm) of an atomic transition. This is close enough to scan the laser to the doppler-broadened atomic transition without mode hopping. We can then use saturated absorption spectroscopy to get within 1 MHz, or about 1×10^{-16} m of the transition.

In the field of optical communications, scientists use a technique called wavelength-division-multiplexing (WDM) [6]. This method enables a single optical fiber to carry various signals simultaneously at multiple wavelengths. This technology requires being able to distinguish between light sources separated in frequency by approximately 100 GHz, or about 1 nm.

We set out to develop an inexpensive and robust method for wavelength detection using a simple array of filtered photodiodes with an initial accuracy goal of 0.01 nm — the accuracy needed for tuning lasers close enough to atomic transitions. However, we realized that if we could not reach this high standard, the method could be used for other applications that requirement far less accuracy, such as WDM.

1.4 Personal contribution and outline

This research represents a strong collaboration with my laboratory partner, Tyler Jones. Tyler and I individually worked on almost every aspect of this project, from data acquisition to instrumentation and algorithm design. We also frequently conversed and worked together. My primary contributions are the development of the least squares algorithm, analysis and utilization of the thin-film interference, design and implementation of electronics for data acquisition, and mathematical code for the Allan deviation and other analyses. Although Tyler and I worked together on this project, this thesis only reports data that I personally acquired and analyzed.

In the Chapter 2, I will first describe the color sensor and our adaptation for its use in laser wavelength detection. I then present a overview of the algorithm we designed to calculate the wavelength of single-frequency light. Chapter 3 gives an analysis of thin-film interference found in the sensor and also contains an experimental characterization of this method's accuracy and long term stability.

Chapter 2

Experimental and Computational Methods

2.1 TCS 3414: an array of filtered photodiodes

The TCS 3414 is a color sensor commonly found in many current consumer electronic devices — including cellular phones, HDTVs, and personal computers [3]. The sensor is designed to mimic the human eye and detect the general color of ambient light. This data is then sent to a personal computer, for example, to adjust the color display. However, the purpose of the sensor is not to measure the wavelength of single frequency light; the intended application is simply to detect the general color of ambient light.

There are two packages of the TCS 3414, namely the FN and the CS packages, and we will compare them in Chapter 3. As shown in Figure 2.1, both packages are relatively small — measuring less than 2 mm by 2 mm — and contain an array of 16 photodiodes with four colors of filters. There are four photodiodes underneath each color of filter. A protective sheet of glass lies on top of the sensors. Each set of photodiodes (with a given color of filter) is connected to a single analog to digital converter, which we will call a *channel*.

A photodiode is a p-n semiconductor junction. An incident photon on a photodiode becomes

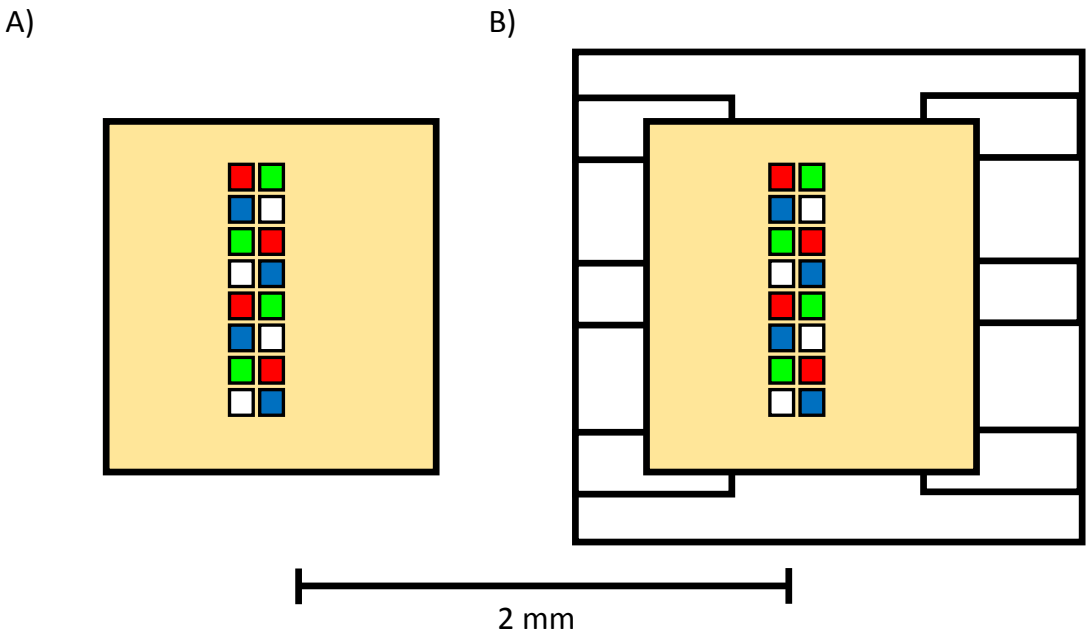


Figure 2.1 Scale diagram of both the A) CS and B) FN packages of the TCS 3414 Color Sensor [3].

an electron and electron-hole pair and can thus induce a photocurrent in a circuit. Photodiodes can make precise measurements of the intensity of incident light. Photodiodes also have intrinsic spectral response curves; their efficiency depends on the frequency, or the photon energy of the incident light. However, a plain array of photodiodes would be relatively ineffective as a wavelength meter; two or more identical photodiodes give no more spectral information than that provided by a single photodiode. To differentiate between different light sources, each channel of photodiodes must have its own spectral response curve. For this reason, the TCS 3414 uses different colors of filters. For example, red filters are designed to transmit red light and absorb other colors . Hence, with different colors of filters, each channel can have a unique spectral response curve.

2.2 Specifications

The TCS 3414 data sheets contains only two crucial specifications about its potential accuracy in wavelength metrology. First, for a semiconductor device, the sensor has a very small temperature coefficient — only 200 parts per million per degree Celsius [3]. Second, the sensor is equipped with 16-bit analog to digital converters (ADC) on each channel [3].

2.3 Data gathering set-up

The color sensor uses inter-integrated circuit (I^2C) protocol for input and output commands. In simple terms, I^2C is a language that enables communication between an external microprocessor and the color sensor. We program the microprocessor to externally determine the integration time on the analog to digital converters. All four channels simultaneously receive the same commands.

Upon integration, each channel records a specific value from 0 to 65536, which we will call *counts*. Dividing all channels by the clear counts mitigates experimental error in the microprocessor's timer and greatly attenuates classical intensity noise from power fluctuations in the laser.

The intensity ratios among the four sets of photodiodes have some spatial dependence due to the finite size of the sensor. For example, if we focus the light from a blue laser onto a single red filtered photodiode, the sensor records only red counts, and we could erroneously assume that the light source is red. Therefore, precise measurements require consistency in the spatial distribution of intensity. We promote this consistency by coupling laser light into a single-mode optical fiber. This way, only one intensity mode can traverse the fiber, enabling long-term repeatability and minimal dependence on the optical power entering the fiber.

We connect the output of the fiber to one side of an enclosed aluminum box and attached the surface-mount color sensor along with a portion of the electronics that control it to the opposite side. We use a fiber with a diameter of 125.0 ± 1.0 μm, and the distance from the end of the fiber

to the color sensor is about 16 cm. As light exits the fiber, it diverges in an approximate Gaussian profile. This footprint allows the Gaussian intensity distribution to expand such that the intensity across the sensor is fairly uniform and very stable. The enclosed box also serves to block any ambient light from the sensor; we record no dark counts on any of the channels with the laser turned off.

As shown in Figure 2.2, we install a thermoelectric cooler (TEC) and thermistor for temperature control behind the color sensor board. We place the TEC flush with the aluminum box such that the box serves as a heat sink. In between the TEC and the color sensor board we place a 1 mm thick sheet of copper. This layer enhances uniformity in the board's temperature. Using a proportional, integral, and derivative feedback (PID) controller we can set the temperature of the sensor to within $\pm 0.1^{\circ}$C.

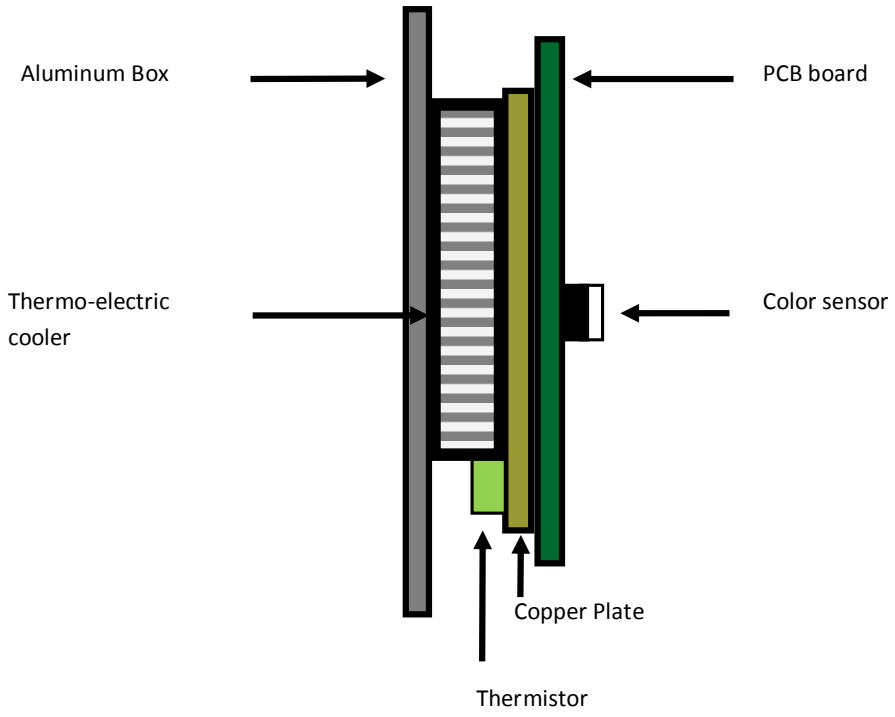


Figure 2.2 Termperature control set-up profile. Power distribution from the thermo-electric cooler (TEC) is evenly distributed using a sheet of copper. The thermistor provides the error signal for the feedback of the PID circuitry.

2.4 The least squares error (LSE) algorithm

After optical measurements are made, they must be processed to retrieve the spectral information of the light source. We designed an efficient algorithm that computes the wavelength of the single frequency light source using a set of calibrated response curves.

The least squares error (LSE) algorithm systematically and simultaneously compares a set of n experimental values to n functions in order to find the common argument. Each channel will record a value commensurate to the convolution of the source's spectrum with the spectral response curve of that channel. With line-widths on the order of a GHz or smaller, the laser light we measure can be sufficiently approximated by a delta function. The measured ratios among each channel for a given frequency will be unique, given that the source has a single frequency.

At the core of the LSE algorithm is the fitting error function. The fitting error Er is a function of wavelength λ and is defined as

$$\begin{aligned} Er(\lambda) = w_{\mathrm{r}}(\lambda)\left[1-\frac{R}{R_{\mathrm{c}}(\lambda)}\right]^2 + w_{\mathrm{b}}(\lambda)\left[1-\frac{B}{B_{\mathrm{c}}(\lambda)}\right]^2 \\ + w_{\mathrm{g}}(\lambda)\left[1-\frac{G}{G_{\mathrm{c}}(\lambda)}\right]^2, \end{aligned} \tag{2.1}$$

where R, B, G are the red, blue, and green counts of a given measurement, $R_{\mathrm{c}}(\lambda)$, $B_{\mathrm{c}}(\lambda)$, $G_{\mathrm{c}}(\lambda)$ are the response curves, and $w_{\mathrm{r}}(\lambda)$, $w_{\mathrm{b}}(\lambda)$, and $w_{\mathrm{g}}(\lambda)$ are weighting factors.

The error function serves to compare the measured value on each channel with the value of each calibrated spectral response curve at a specific wavelength. The algorithm records the error over a desired range of the spectrum and calculates the wavelength by minimizing the least squares error function.

We began by analyzing the spectral response curves given in the datasheet. We wrote a program to digitize the data exactly as was placed in the vector graphics format of the datasheet. We then interpolated the spectral response data to create response functions of wavelength. However, the spectral response data from the datasheet was recorded at 2 nm intervals using light with spectral half-widths of 17-47 nm. This lacked the resolution to make measurements with accuracy better than about 1 nm [3]. However, we achieved much greater accuracy by calibrating our own set of spectral response curves (see section 2.5).

We can optimize the algorithm by weighting the error function such that the error from each channel can be scaled according to the slope and magnitude of the response curves. The rationale for weighting the error according to the derivative of the spectral response curves is that a curve with a large slope contains more information about the wavelength than does a curve with a level slope. Weighting according to magnitude could be favorable because a channel that integrates a larger intensity will have less bit noise and more digits of precision.

As mentioned, the objective of the algorithm is to measure single frequency light. Howev-

er, it also can ascertain the peak frequency for light with relatively large line widths. We tested the algorithm numerically with various frequency distributions. A huge 5 nm FWHM Gaussian wavelength distribution centered at 500 nm causes only 0.057 nm of error. A symmetric triangle waveform centered at 500 nm with a base width of 5 nm causes only 0.027 nm of error. An asymmetric triangle waveform with a peak at 500 nm and a base of 1 nm causes an error of 0.335 nm. Hence, the algorithm is sufficiently robust to measure the peak wavelengths both small and large line-widths light sources.

2.5 Calibration

The LSE algorithm requires a calibrated set of spectral response curves. We calibrate the curves by simultaneously making measurements on both the color sensor and a commercial interferometer wavelength meter. We generate response curves as a function of the wavelength measured by the commercial device.

To calibrate the sensor, we record the ratios among the four channels at various wavelengths using the following systematic approach. The commercial laser controller provides an analog frequency modulation input. The frequency modulation input controls the angle and position of the diffraction grating on the external cavity of the diode laser. Altering the angle of the diffraction grating changes the frequency that the cavity feeds back to the diode. The input requires an analog dc signal between +3 V and -3 V.

The microprocessor we use does not have an analog output. However, it does provide a pulse width modulated (PWM) signal. Pulse width modulation allows one to convert a digital signal into an analog signal. During part of the cycle, the microprocessor outputs a high voltage (5 V) and during the rest, a low voltage (0 V). Two parameters determine the signal: the frequency and the duty cycle. As shown in Figure 2.3, the duty cycle determines what percentage of the cycle will be

high. For example, a duty cycle of 100 % is simply a dc voltage of 5 volts. The frequency we can control is simply one over the cycle period. We set our signal to a frequency of about 31 KHz.

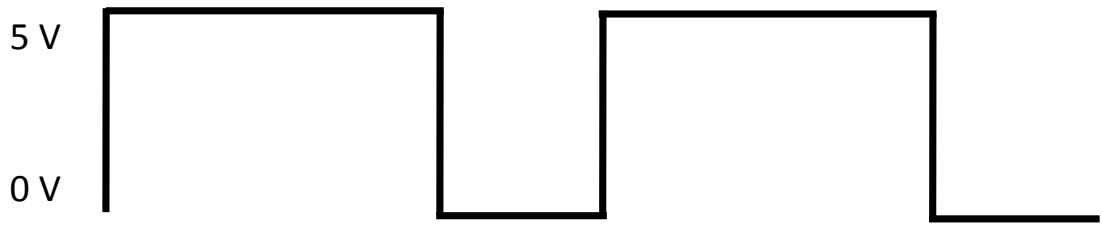


Figure 2.3 A pulse-width modulated signal with a duty cycle of approximately 70 %.

We convert this PWM signal into an analog signal using a low pass filter. The output from the low pass filter passes to a differential amplifier and buffer, which convert the filtered dc signal ranging from 0 to 5 V into a signal ranging from -3 to 3 V (see Figure 2.4). This range of voltages corresponds to a wavelength range of about 0.12 nm.

We developed a program that simultaneously gathers data from the TCS 3414 and a commercial wavelength meter. The commercial wavelength meter uses a Michelson Morley interferometer. This instrument calculates the wavelength by measuring the fringe count ratio of the laser of interest to a reference laser [5, 12].

We calibrate the algorithm by tuning the laser to 100 random wavelengths within a 0.12 nm range and then gathering data from both the color sensor and the commercial wavelength meter. We divide the red, blue, and green channels by the clear channel and fit a second order polynomial to the blue to clear, red to clear, and green to clear ratios as functions of wavelength. We can then use these calibrated response curves in the algorithm for data processing. Figure 2.5 shows the general schematic for the calibration.

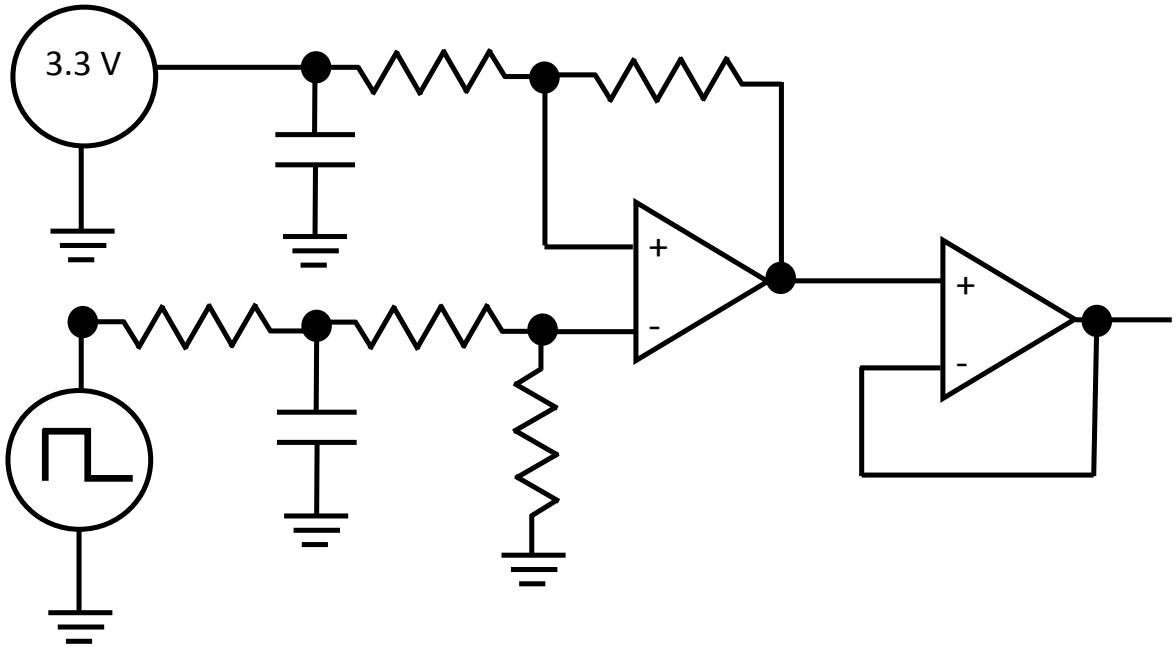


Figure 2.4 This schematic contains (from left to right) a low-pass filter, differential amplifier that converts a dc signal in the range from 0 to 5 V into a signal ranging from -3 to 3 V, and a buffer to provide enough current to the low impedance input on the laser controller.

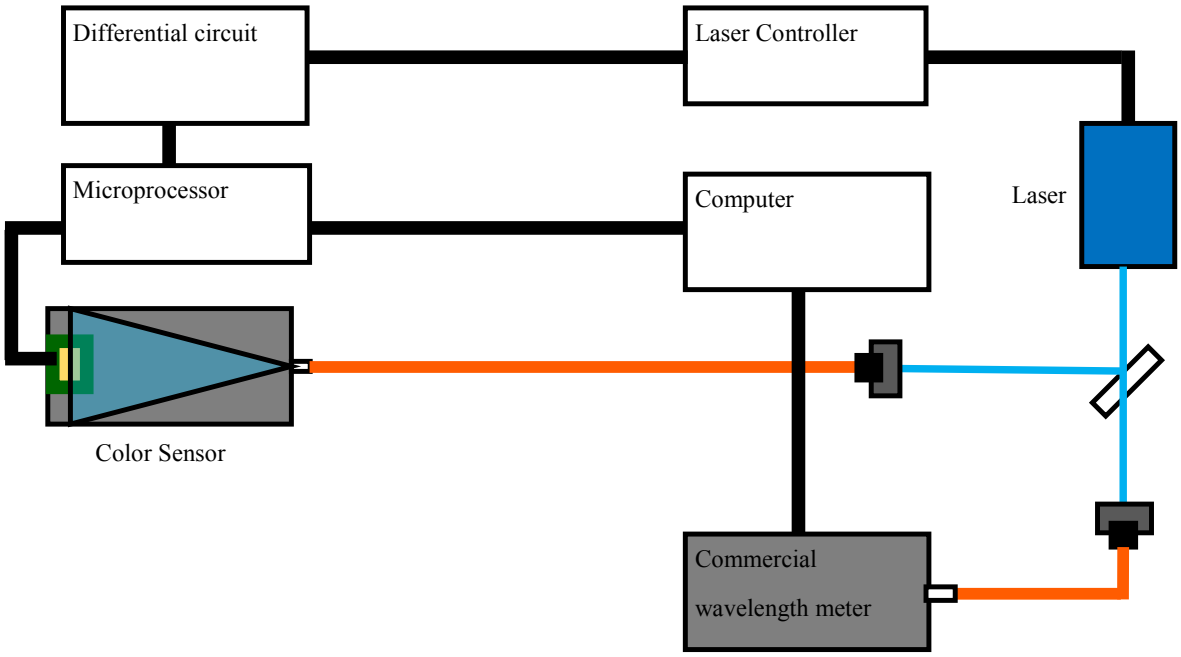


Figure 2.5 Experimental setup for color sensor calibration. The external microprocessor (Arduino) outputs a random pulse-width modulated (PWM) signal to the differential amplifier circuit. The circuit converts the pulse width modulation signal into a voltage between -3 V and 3 V. This voltage adjusts the piezos on the external cavity of the diode laser, which in turn changes the laser frequency. We simultaneously make measurements with the color sensor and the commercial wavelength meter.

Chapter 3

Experimental Results

In this chapter, we report an analysis of thin-film interference effects on the photodiodes' response and a characterization of the accuracy and long term stability of the filtered photodiode method.

3.1 Temperature dependence

To investigate the temperature dependence of the TCS 3414 in a reproducible manner, we put a heater plate on the top of the aluminum box to gradually warm up the color sensor. With the temperature control turned off, we measured the temperature using a 10 K thermistor from the temperature control set-up. The Arduino microprocessor that reads the data from the color sensor has several analog inputs which we used to measure the temperature of the device. Using a voltage divider with a 10 K resistor, we can digitally record the temperature data as we vary the temperature, as shown in Figure 3.1.

The TCS data sheet reports a temperature coefficient of 200 parts per million [3]. However, we observe much greater variability when measuring single frequency light. Both FN and CS packages exhibit a sinusoidal response as a function of temperature. We acquired data on two FN and two CS packages — one with the protective glass layer sanded and another not sanded for

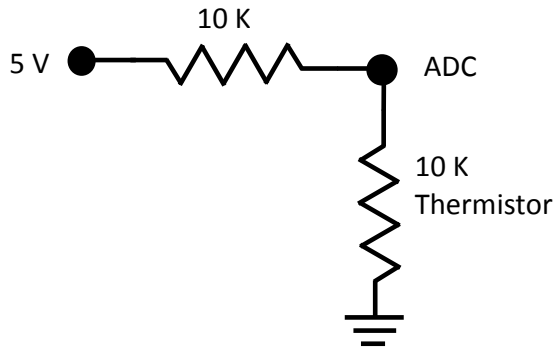


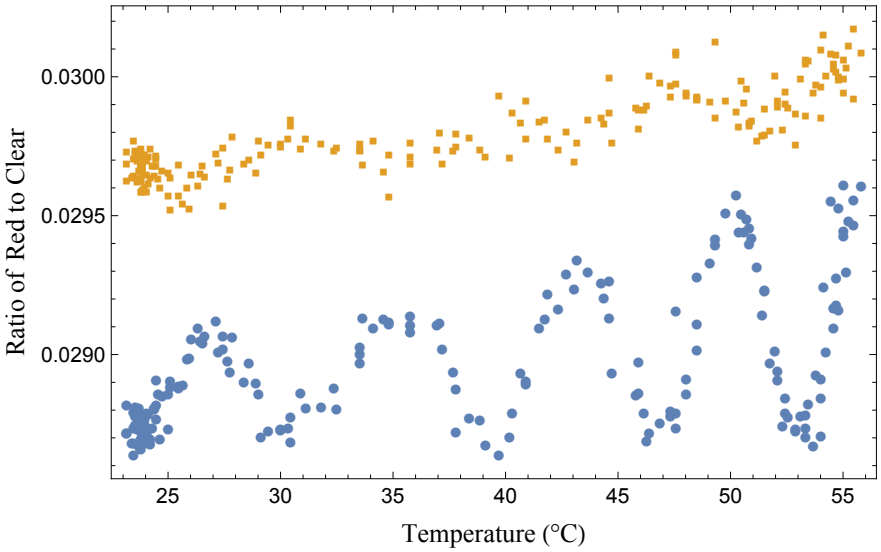
Figure 3.1 Schematic for temperature measurement set-up.

each package. The effect of sanding will be described shortly. Figures 3.2a and 3.2b show the temperature dependence of the FN and CS packages as we increase the temperature from 23°C to 55°C.

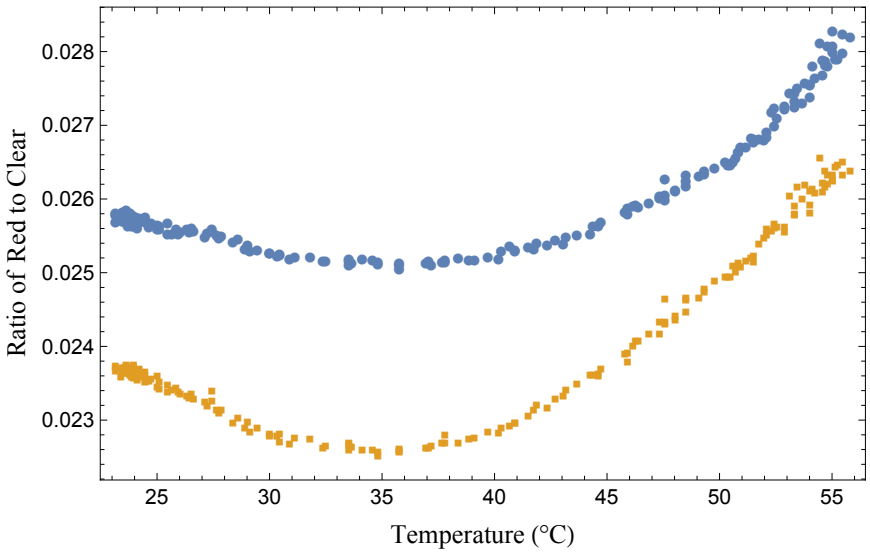
3.2 Thin film interference effects

The observed sinusoidal response indicates thin-film interference. As the system increases in temperature, the transparent protective material expands. With broadband light — the light that the color sensor is intended to characterize — the thermal expansion of the thin film has little effect on the recorded intensity. With a single frequency light source, however, the protective layer of glass can form an optical cavity. Hence, thermal expansion changes the length of the cavity or *etalon*, resulting in temperature dependent fringes.

We observe that sanding the first protective glass surface of the FN package mitigates fringes while sanding the CS package minimally affects the response (see Figures 3.2a and 3.2b). This suggests that the dominant thin-film interference in the FN package originates from the thin film



(a) Ratio of red to clear of the sanded (red) and not sanded (blue) FN package as a function of temperature. Sanding the protective surface dramatically mitigates the etalon response in the FN package.



(b) Ratio of red to clear of the sanded (red) and not sanded (blue) CS package as a function of temperature. Sanding the protective surface has little effect on the period and amplitude of the fringes.

Figure 3.2 Spectral responsivity of the CS and FN packages as a function of temperature.

of glass covering the filters while a secondary thin film is responsible for the etalon in the CS package. This comes as a surprise because the packages are similar in design and functionality.

We also took measurements at various wavelengths to confirm that the periodic variation in temperature was due to etaloning. Figures 3.3a and 3.3b show the red to clear ratios of the FN and CS packages.

An analysis of the free spectral range (FSR) — or the change in frequency from one peak to another — confirms that the etaloning inducing thin-film is different for the two packages. The FSR is given by

$$\Delta\lambda_{FSR} = \frac{\lambda^2}{2nd_0 cos\theta}, \tag{3.1}$$

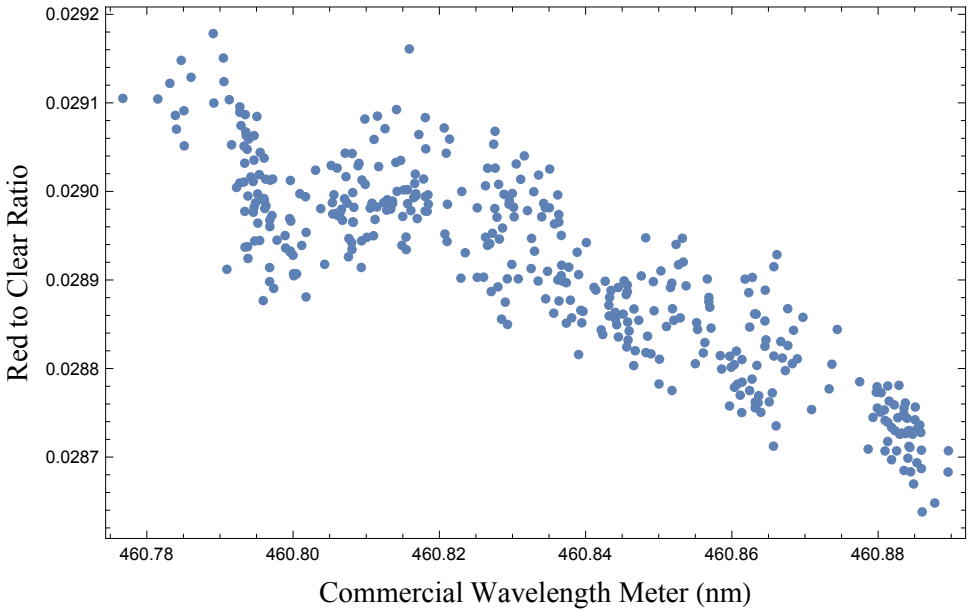
where n is the index of refraction, d is the length of the optical cavity, θ is the angle of incidence, and λ is the wavelength.

The free-spectral range of the FN package is about 0.04 nm (see Figure 3.3a). The datasheet indicates an index of refraction of 1.53 for the glass protective layer, suggesting a thin film thickness of about 1.7 mm [3]. The etalon in the CS package originates from an inner thin film — either from the layer of epoxy or the filters themselves. For the CS package, the observed free spectral range is about 0.2 nm, indicating a 0.35 mm layer (see Figure 3.3). The larger period in the temperature response (see Fig. 3.2b) of the CS package corroborates that the film is much thinner than the glass of the FN package.

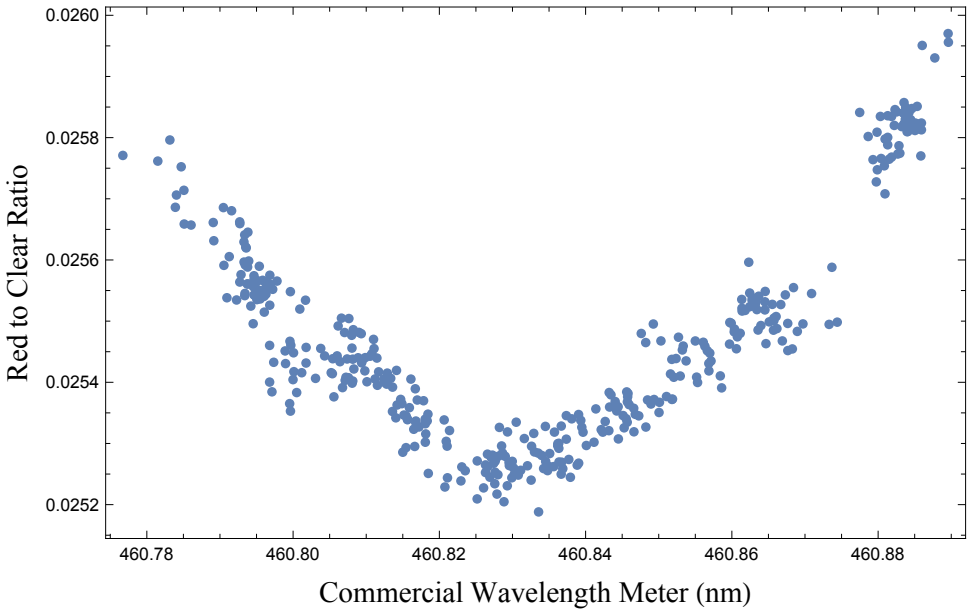
In an etalon, the coefficient of finesse F determines the amplitude and width of the fringes, and coefficient of finesse is determined by the reflectivity of the first and second interfaces:

$$F_s = \frac{4\sqrt{R_s^{1->0} R_s^{1->2}}}{(1-\sqrt{R_s^{1->0} R_s^{1->2}})^2}, \tag{3.2}$$

where $R_s^{1->0}$ is the reflectivity of the first interface and $R_s^{1->2}$ is the reflectivity of the second interface. As the reflectivity of the first interface goes to zero, so does the coefficient of finesse.



(a) Ratio of red to clear of the FN package (not sanded) as a function of wavelength.



(b) Ratio of red to clear of the CS package (not sanded) as a function of wavelength.

Figure 3.3 Spectral responsivity of the CS and FN packages as a function of wavelength.

Hence, decreasing the reflectivity of the first interface of the etalon inducing film will attenuate the fringes and consequently the dramatic temperature instability. While roughening the surface of a thin film does not eliminate reflections on a microscopic level, it randomizes the reflection angles and macroscopically mitigates spatially coherent reflections. Hence, by roughening the first reflective surface of the FN package we reduce the macroscopic reflectivity and consequently the fringes due to thin-film interference (see Figure 3.2a).

3.3 Accuracy and calibration deviation

Without sufficient temperature stability, thin-film interference can impair the reliability of the filtered photodiode method. However, thin-film interference can actually enhance the precision of the photodiode method if the temperature of the chip is controlled. The thin film introduces relatively steep features in the ratios among channels, allowing the algorithm to more accurately distinguish between different wavelengths in a smaller range.

We continuously gathered data from a CS package sensor (not sanded) at randomized wavelengths to determine the accuracy and calibration drift. Wavelengths ranging from 460.78 nm to 460.89 nm were continuously selected at random to ensure a proper representation of the wavelength dependence of the sensor. Temperature was controlled with a stability of $\pm 0.15^{\circ}$C. Table 3.1 shows the accuracy (average discrepancy photodiode measured value and the measurement of the commercial wavelength meter) of the method as a function of time. The initial accuracy of our method is 0.0053 nm. Figure 3.4 shows data at random wavelengths as recorded by the commercial wavelength meter and the filtered photodiode method.

Accuracy over time: CS package	
Time after calibration (hrs)	Accuracy (nm)
0-1	0.00534302
10-11	0.00863695
23-24	0.0102975
30-31	0.0119349
70-71	0.0118203

Table 3.1 Deviation from initial calibration of filtered photodiode method over a period of 71 hours. Spectral response curves were calibrated using the first 20 minutes of data. Wavelengths ranging from 460.78 nm to 460.89 nm were continuously selected at random to ensure a proper representation of the wavelength dependence of the sensor. Temperature was controlled with a stability of $\pm 0.15^{\circ}$C.

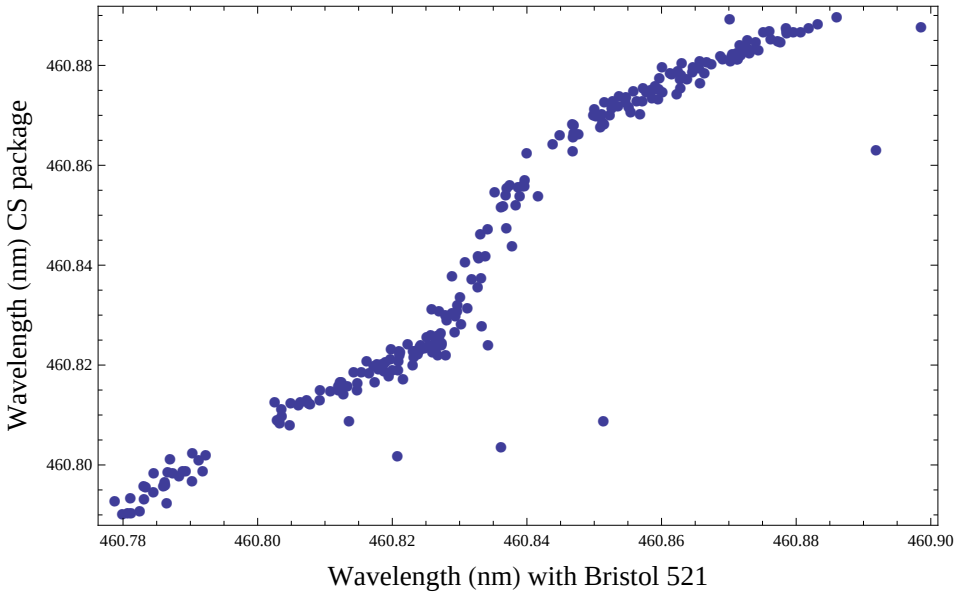


Figure 3.4 Data at random wavelengths as recorded by the commercial wavelength meter and the filtered photodiode method. We took this particular data set 30 hours after the initial calibration. Wavelengths ranging from 460.78 nm to 460.89 nm were continuously selected at random to ensure a proper representation of the wavelength dependence of the sensor. Temperature was controlled with a stability of $\pm 0.15^{\circ}$C. The average accuracy for this set of data was 0.0119 nm.

3.4 Laser stability detection

To test the photodiode method's potential for use in atomic physics, we left the laser unlocked while taking data and then locked the laser to an atomic resonance of strontium. Figure 3.5 shows that the CS package can accurately detect changes in wavelength as small as 0.05 nm while the FN package fails to detect changes in wavelength of this order of magnitude.

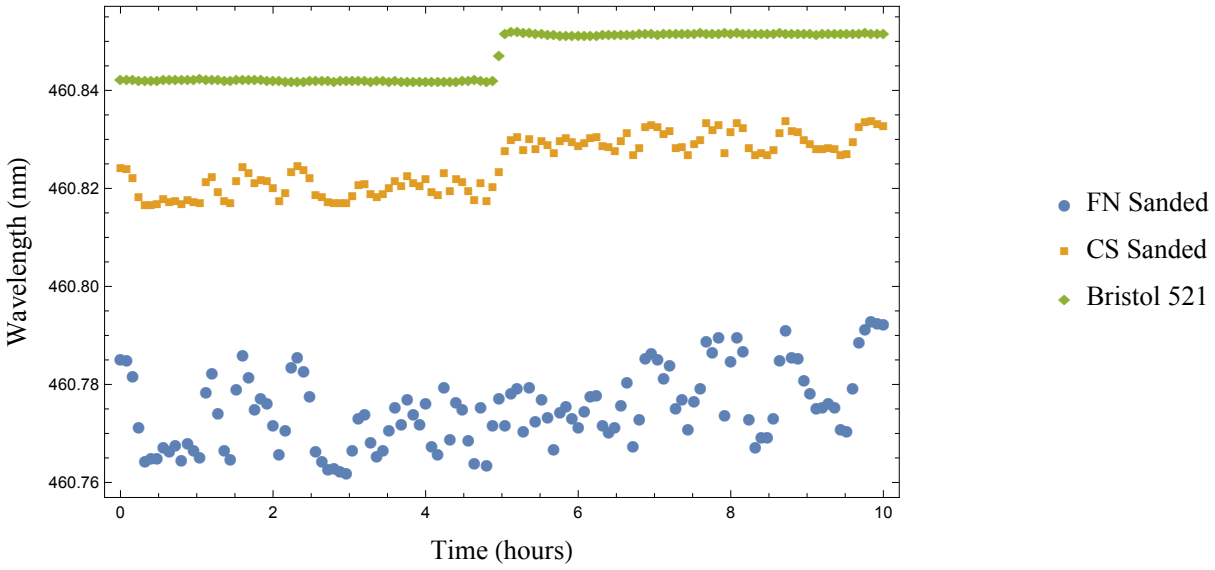


Figure 3.5 Continuous measurements on with an unlocked (first half) vs a locked laser (second half). Data were averaged over periods of about 5 minutes. The CS package can accurately detect changes in wavelength as small as 0.05 nm. The FN package fails to detect changes in wavelength of this order of magnitude.

3.5 Allan deviation

To accurately characterize the long term drift we locked a laser using saturated absorption spectroscopy(see Appendix A.1). We and gathered data continuously (3 data points per second on each channel of each of the four sensors) for 8 days. Figure 3.6 shows how the measurements deviate with time.

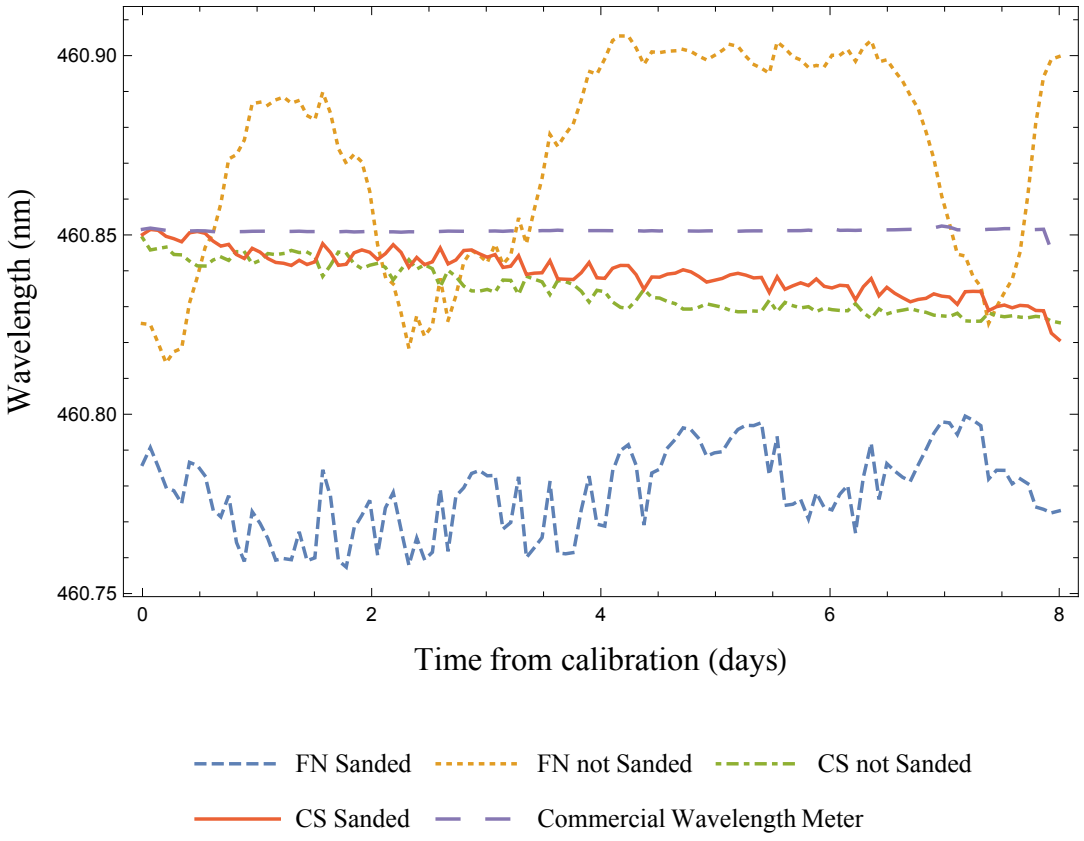


Figure 3.6 Wavelength as measured by the FN and CS color sensor packages and commercial interferometer based wavelength meter. We locked a 461 nm diode laser using saturated absorption spectroscopy in a hot strontium vapor cell. While locked, data was gathered continuously from the four sensors and wavelength meter over a period of 8 days. Data were averaged over periods of 1 hour.

We characterize the long term stability using the Allan deviation. The Allan deviation is defined

by

$$\sigma(\tau) = \sqrt{\frac{1}{2}\overline{(\overline{y_{n+1}} - \overline{y_n})^2}}, \tag{3.3}$$

where $\overline{y_{n+1}}$ and $\overline{y_n}$ are the average values of adjacent groups in time. Essentially, the Allan deviation divides the data in groups of length time τ. The average value of each data group is compared to the averages of adjacent groups. The Allan variation is the average of squared differences between adjacent groups, and the Allan deviation is the square root of the Allan variation. In simple terms, Allan deviation computes how the measurement deviates on a specific time scale.

Figure 3.7 shows the Allan deviation of the FN and CS color sensor packages and the commercial interferometer based wavelength meter. The CS package (not sanded) achieves a stability of 0.0039 nm over 41.5 hours. The CS packages exhibits less deviation than the commercial wavlenegth meter on the short term, but drifts over longer time scales. While sanding minimally affects the CS package performance, it significantly influences the FN package.

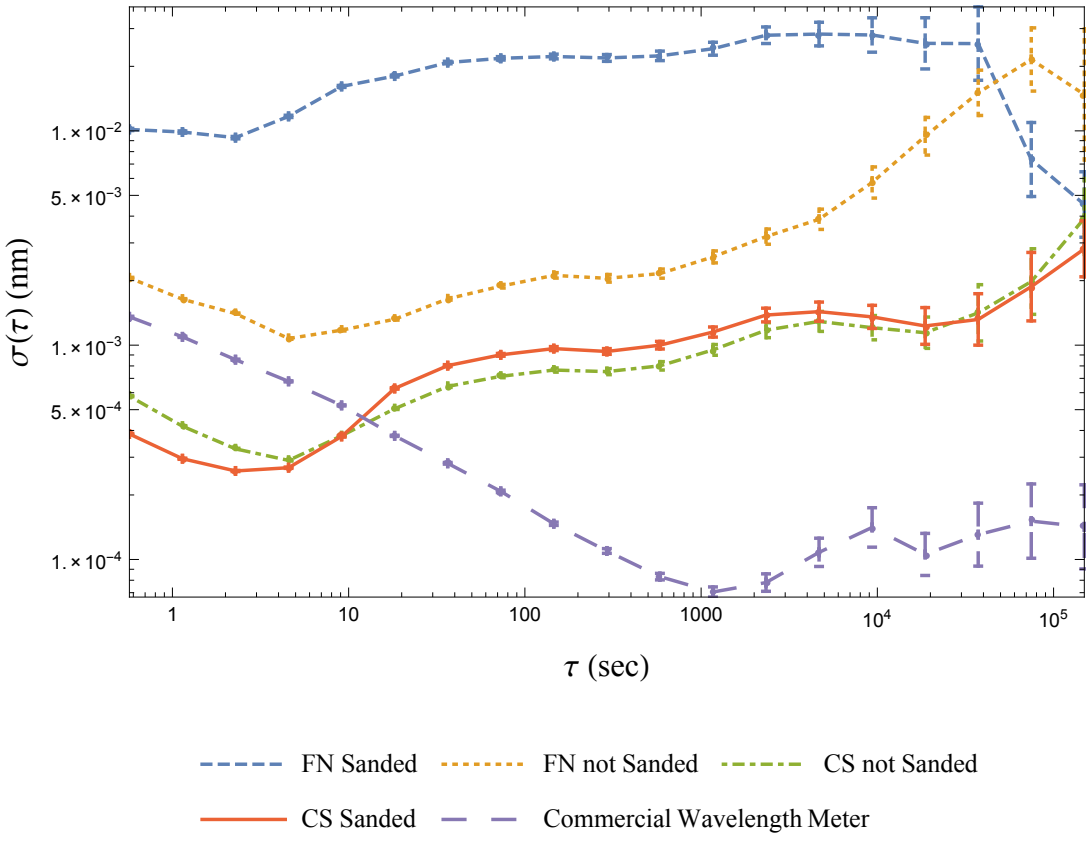


Figure 3.7 Allan Deviation of the FN and CS color sensor packages and commercial interferometer based wavelength meter. We locked a 461 nm diode laser using saturated absorption spectroscopy in a hot strontium vapor cell. While locked, data was gathered continuously from the four sensors and wavelength meter over a period of 8 days. The Allan Deviation shows the average drift over a specified time interval τ.

3.6 Conclusion

The Allan deviation reveals both the short term precision and the long term, temperature independent drift of the filtered photodiode method. As a proof of principle, this research establishes an inexpensive, compact, and robust method for laser wavelength detection using an array of filtered photodiodes. We also report superior performance of the CS package in comparison with the FN package. Using an unsanded CS package, the method achieves an accuracy of 0.0053 nm (on the

range of 460.79 nm to 460.89 nm) and a stability of 0.0039 nm over 41.5 hours (at 460.85 nm). Within a couple of days from calibration, we can use this method to tune lasers close to atomic resonances or detect the instability of a laser lock. On short time scales, the $3 photodiode array compares favorably with commercial, interferometer based methods. With a greater understanding of compensation for the long-term drift, the filtered photodiode method has potential as an accurate and stable wavelength meter.

Appendix A

Experimental Techniques

A.1 Saturated absorption spectroscopy

We used a saturated absorption lock to determine the long term behavior of the color sensor. Due to Doppler broadening in atomic vapors, there is a limit to how precise one can lock to an atomic transition. However, using a technique called saturated absorption spectroscopy, one can lock a resonance peak that is limited only by the lifetime of the state [13].

Saturated absorption spectroscopy involves a hot (gas) ensemble of atoms, a strong pump field, and a weak counter propagating probe field, as shown in Figure A.1. The atoms moving toward the field experience a blue shift in the resonance while those traveling with the field are red shifted. Hence the only ensemble of atoms that both the pump and the probe field can address are the atoms with no component of their velocity in the direction of laser propagation.

We scan the laser frequency over the resonance and observe the absorption of the probe field as a function of frequency. In the center of the resonance there is a dip in the absorption (see Fig. A.2. This occurs because the pump laser saturates a large portion of the atoms that the probe field is also trying to address. Using homodyne detection, we take the derivative of this absorption spectrum,

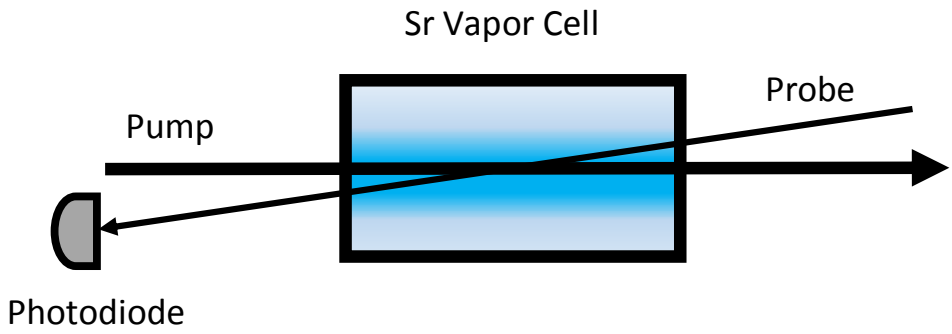


Figure A.1 Experimental setup for saturated absorption spectroscopy. Pump and probe fields counter-propagate in a hot atomic vapor.

resulting in a narrow very steep dispersion-like curve. We can then use a PID controller to the lock the laser to this narrow feature.

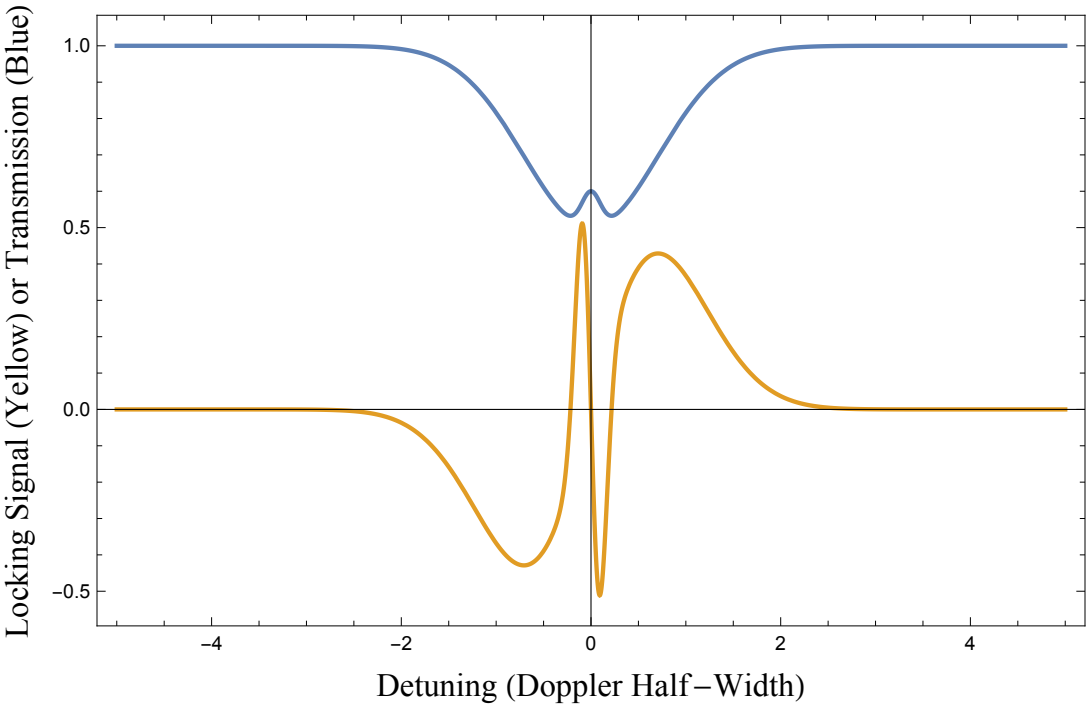


Figure A.2 (Blue) Mathematical representation of probe signal as a function of frequency. This graph gives the general shape of the transmission spectrum (Blue) along with its derivative (Yellow). When the probe and pump are exactly on resonance, they interact with the same ensemble of atoms. Hence, the pump saturates the absorption medium, increasing the probe's transmitted power. (Yellow) Lock signal for the saturated absorption spectroscopy lock. The lock signal is the derivative of the transmitted probe signal.

Appendix B

Code for data analysis

B.1 Calibration and Allan deviation

Data Analysis Code

Setup

In[1]:=
```
SetDirectory[NotebookDirectory[]]
Needs["PlotLegends`"]
Needs["ErrorBarPlots`"]


FindFits[input_, divide_, select_, graph_] :=
 Module[{data, red, green, blue, BluebyClear, GreenbyClear, RedbyClear,
   BluebyClear4, GreenbyClear4, RedbyClear4, BluebyClear2, GreenbyClear2,
   RedbyClear2, BluebyClear3, GreenbyClear3, RedbyClear3, length,
   listblue = {}, listgreen = {}, listred = {}, listblue2 = {}, listgreen2 = {},
   listred2 = {}, listblue3 = {}, listgreen3 = {}, listred3 = {}, listblue4 = {},
   listgreen4 = {}, listred4 = {}, Wavelength = {}, plotb, plotg, plotr,
   mag = {}, selectlistBlue = {}, selectlistRed = {}, selectlistGreen = {}},
  data = Drop[Import[input, "Table"], 1];
  length = Length[data] / divide;
  BluebyClear = #[[4]] / #[[1]] & /@ data // N;
  GreenbyClear = #[[3]] / #[[1]] & /@ data // N;
  RedbyClear = #[[2]] / #[[1]] & /@ data // N;
  BluebyClear2 = #[[8]] / #[[5]] & /@ data // N;
  GreenbyClear2 = #[[7]] / #[[5]] & /@ data // N;
  RedbyClear2 = #[[6]] / #[[5]] & /@ data // N;
  BluebyClear3 = #[[12]] / #[[9]] & /@ data // N;
  GreenbyClear3 = #[[11]] / #[[9]] & /@ data // N;
  RedbyClear3 = #[[10]] / #[[9]] & /@ data // N;
  BluebyClear4 = #[[16]] / #[[13]] & /@ data // N;
  GreenbyClear4 = #[[15]] / #[[13]] & /@ data // N;
  RedbyClear4 = #[[14]] / #[[13]] & /@ data // N;

  mag = #[[1]] & /@ data // N;
  Wavelength = #[[17]] & /@ data;
  Do[{AppendTo[listblue, {Wavelength[[i]], BluebyClear[[i]]}],
    AppendTo[listred, {Wavelength[[i]], RedbyClear[[i]]}],
    AppendTo[listgreen, {Wavelength[[i]], GreenbyClear[[i]]}],
    AppendTo[listblue2, {Wavelength[[i]], BluebyClear2[[i]]}],
    AppendTo[listred2, {Wavelength[[i]], RedbyClear2[[i]]}],
    AppendTo[listgreen2, {Wavelength[[i]], GreenbyClear2[[i]]}],
    AppendTo[listblue3, {Wavelength[[i]], BluebyClear3[[i]]}],
    AppendTo[listred3, {Wavelength[[i]], RedbyClear3[[i]]}],
    AppendTo[listgreen3, {Wavelength[[i]], GreenbyClear3[[i]]}],
```

```
    AppendTo[listblue4, {Wavelength[[i]], BluebyClear4[[i]]}],
    AppendTo[listred4, {Wavelength[[i]], RedbyClear4[[i]]}],
    AppendTo[listgreen4, {Wavelength[[i]], GreenbyClear4[[i]]}]}, {i, length}];
  selectlistBlue = {listblue, listblue2, listblue3, listblue4};
  selectlistRed = {listred, listred2, listred3, listred4};
  selectlistGreen = {listgreen, listgreen2, listgreen3, listgreen4};
  plotb = ListPlot[selectlistBlue[[select]]];
  plotg = ListPlot[selectlistGreen[[select]]];
  plotr = ListPlot[selectlistRed[[select]]];

  If[graph > 0,
   Print[GraphicsGrid[{{plotb, plotg, plotr}}, ImageSize → {600, 300}]]];
  {selectlistBlue[[select]], selectlistGreen[[select]], selectlistRed[[select]]}
 ]
Calibrate[list_] := Module[{values, express, bluesol, greensol, redsol},
  express = a + b x + c x^2;
  bluesol = FindFit[list[[1]], express, {a, b, c}, x];
  greensol = FindFit[list[[2]], express, {a, b, c}, x];
  redsol = FindFit[list[[3]], express, {a, b, c}, x];
  {express /. bluesol, express /. greensol, express /. redsol}]

SelectFiles[file_, number_, graph_] :=
 Module[{hello, list, results, plotb, plotg, plotr},
  list = FindFits[file, 1, number, graph];
  results = Calibrate[list];
  plotb = Plot[results[[1]], {x, 460.8, 460.88}];
  plotg = Plot[results[[2]], {x, 460.8, 460.88}];
  plotr = Plot[results[[3]], {x, 460.8, 460.88}];

  If[graph > 0,
   Print[GraphicsGrid[{{plotb, plotg, plotr}}, ImageSize → {600, 300}]]];

  results]
```

FindFits, Calibrate, and SelectFiles are functions that collectively fit and plot 2nd order polynomials to the calibration data.

In[16]:=

```
weighting[λ_, amountd_, amountm_] :=
 Module[{cderiv, rderiv, gderiv, bderiv, cmag, rmag,
   gmag, bmag, totald, totalm, factord, factorm, wc, wr, wg, wb},
  cderiv = Abs[clearf'[λ]];
  rderiv = Abs[redf'[λ]];
  gderiv = Abs[greenf'[λ]];
  bderiv = Abs[bluef'[λ]];
  cmag = clearf[λ];
  rmag = redf[λ];
  gmag = greenf[λ];
  bmag = bluef[λ];
  totald = cderiv + rderiv + gderiv + bderiv;
  factord = 1 / totald;
  totalm = cmag + rmag + gmag + bmag;
  factorm = 1 / totalm;
  wc = 1 + (cderiv * factord) * amountd + (cmag * factorm) * amountm;
  wr = 1 + (rderiv * factord) * amountd + (rmag * factorm) * amountm;
  wg = 1 + (gderiv * factord) * amountd + (gmag * factorm) * amountm;
  wb = 1 + (bderiv * factord) * amountd + (bmag * factorm) * amountm;
  {wc, wr, wg, wb}
 ]
```

This weighting function allows the user to weight the error according to the derivative and magnitude of the interpolated functions. The function calls from the amountd and the amountm, which refer to the amount we desire to weight acording to the magnitude of the derivative or the magnitude of the function itself.

Our rationale for biasing the error according to the magnitude was that function with a greater magnitude at a certain wavelength would enable us to find the optimal scaling factor with the greatest accuracy. Once the scaleFactor was found, we conjectured that functions with derivatives of greater magnitudes would allow us to more effectively hone down on the correct wavelength. This is because a small change in wavelength would result in a large change in intensity.

However, we found that found our purposes, we obtained better results by leaving the error function unweighted in order to find the scale factor and then putting a minute negative bias on the derivatives and larger positive bias on the magnitudes. The optimalerror functions uses the following weighting after the scalefactor and offset have been found.

In[4]:=
```
bluef1[x_] := 74937.66841042477` - 325.19589539246186` x + 0.35280459011931553` x^2;
bluef2[x_] := -323252.4645447568` + 1403.1000384090917` x - 1.5225600328909292` x^2;
bluef3[x_] := 1.2892216958920173`*^6 - 5595.3617646355915` x + 6.071122492496138` x^2;
bluef4[x_] := 214296.3177164328` - 929.649735015795` x + 1.0082433592664468` x^2;
greenf1[x_] := 32849.578432942784` - 142.57150846603213` x + 0.15469546640741025` x^2;
greenf2[x_] := -46412.570576115075` + 201.4809895362766` x - 0.21866085183060144` x^2;
greenf3[x_] := 71200.84199716614` - 308.98412173884395` x + 0.33521861367800887` x^2;
greenf4[x_] := 333191.70818404824` - 1446.1692749564977` x + 1.5692215274209804` x^2;
redf1[x_] := -4647.438270157217` + 20.169782172841934` x - 0.021883965464137838` x^2;
redf2[x_] := -3939.7850467125736` + 17.10173531564185` x - 0.018558575552595703` x^2;
redf3[x_] := 40390.70180728409` - 175.2948784086074` x + 0.19019422954599277` x^2;
redf4[x_] := 69561.41425509405` - 301.90161605290064` x + 0.3275688694604616` x^2;
```

In[17]:=
```
optimalerror4channels[λ_?NumericQ, clearin_, redin_, greenin_, bluein_, if_] :=
 Module[{scaleFactor, offset, error1, error2, error3, error4, ans, clear, red, blue1,
   green1, red1, blue2, green2, red2, blue3, green3, red3, blue4, green4, red4, wc, wr,
   wg, wb, weight, totalweightd, totalweightm, finalerror, optimaloffset, answer},
  blue1 = bluein[[1]] / clearin[[1]] // N;
  red1 = redin[[1]] / clearin[[1]] // N;
  green1 = greenin[[1]] / clearin[[1]] // N;
  blue2 = bluein[[2]] / clearin[[2]] // N;
  red2 = redin[[2]] / clearin[[2]] // N;
  green2 = greenin[[2]] / clearin[[2]] // N;
  blue3 = bluein[[3]] / clearin[[3]] // N;
  red3 = redin[[3]] / clearin[[3]] // N;
  green3 = greenin[[3]] / clearin[[3]] // N;
  blue4 = bluein[[4]] / clearin[[4]] // N;
  red4 = redin[[4]] / clearin[[4]] // N;
  green4 = greenin[[4]] / clearin[[4]] // N;
  clear = 1;
  error1 = ((blue1) / bluef1[λ] - 1)^2 +
    ((red1) / redf1[λ] - 1)^2 + ((green1) / greenf1[λ] - 1)^2;
  error2 = ((blue2) / bluef2[λ] - 1)^2 + ((red2) / redf2[λ] - 1)^2 +
    ((green2) / greenf2[λ] - 1)^2;
  error3 = ((blue3) / bluef3[λ] - 1)^2 + ((red3) / redf3[λ] - 1)^2 +
    ((green3) / greenf3[λ] - 1)^2;
  error4 = ((blue4) / bluef4[λ] - 1)^2 + ((red4) / redf4[λ] - 1)^2 +
    ((green4) / greenf4[λ] - 1)^2;
  If[if == 1, answer = error1];
  If[if == 2, answer = error2];
  If[if == 3, answer = error3];
  If[if == 4, answer = error4];
  If[if == 5, answer = error4 + error3];
  answer

 ]
```

This optimalerror function optimizes a scaling factor and offset in order to output the lowest possible least squares error for each possible wavelength. Hence, our assumption is that with a certain set of

data, the correct wavelength will be the wavelength that produces the least amount of error.Because we do not know which sensor is the most reliable, we choose to weight them equally.
We found the FindMinimum funcion to be much faster than alternatives such as NMinimize.
You may note that the final error is divided by 4 times the square of the optimal scaling factor and then squarerooted. This is because in our error function squares the product of the scaleFactor and each sensor value. We divided by four because we wanted the average error produced by each clear, red, green, and blue values.
The ?NumericQ allows this module to be used as a normal function (in oder to be plotted etc.)

```
AllanDev4[filename_, taustart_, number_, approximate_, wd_, wm_, range_, amount_] :=
 Module[{size, data5 = {}, Clear1, Red1, Green1, Blue1, Clear2, Red2, Green2,
   Blue2, Clear3, Red3, Green3, Blue3, Clear4, Red4, Green4, Blue4, bristol,
   index, tau, ans, error = {}, location = {}, locationold = {}, results = {},
   totalvariance = {}, allanvar1 = {}, allanvar2 = {}, allanvar3 = {}, allanvar4 = {},
   allanvarb = {}, resultsb = {}, locationb, locationbold, errorb, plotS1, plotS2,
   plotS3, plotS4, plotB, itera, error4list, allanvar5 = {}, allanvar6 = {}},
  Do[data5 = Join[data5, Drop[Import[filename[[i]], "Table"], 1]],
   {i, Length[filename]}];
  size = Length[data5];
  Print[size];
  data5 = Reverse[data5];
  Clear1 = #[[1]] & /@ data5 // N;
  Red1 = #[[2]] & /@ data5 // N;
  Green1 = #[[3]] & /@ data5 // N;
  Blue1 = #[[4]] & /@ data5 // N;
  Clear2 = #[[5]] & /@ data5 // N;
  Red2 = #[[6]] & /@ data5 // N;
  Green2 = #[[7]] & /@ data5 // N;
  Blue2 = #[[8]] & /@ data5 // N;
  Clear3 = #[[9]] & /@ data5 // N;
  Red3 = #[[10]] & /@ data5 // N;
  Green3 = #[[11]] & /@ data5 // N;
  Blue3 = #[[12]] & /@ data5 // N;
  Clear4 = #[[13]] & /@ data5 // N;
  Red4 = #[[14]] & /@ data5 // N;
  Green4 = #[[15]] & /@ data5 // N;
  Blue4 = #[[16]] & /@ data5 // N;



  bristol = #[[17]] & /@ data5 // N;

  tau = taustart;
  Print[tau];
  Do[{index = 1,
    results = {},
    Do[{location = {},
      Do[If[q == 6, AppendTo[location, Mean[bristol[[index ;; index + tau - 1]]]],
```

```
    {ans = FindMinimum[optimalerror4channels[x,
        {Mean[Clear1[[index ;; index + tau - 1]]], Mean[Clear2[[index ;;
            index + tau - 1]]], Mean[Clear3[[index ;; index + tau - 1]]],
         Mean[Clear4[[index ;; index + tau - 1]]]}, {Mean[Red1[[index
            ;; index + tau - 1]]], Mean[Red2[[index ;; index + tau - 1]]],
         Mean[Red3[[index ;; index + tau - 1]]], Mean[Red4[[index ;;
            index + tau - 1]]]}, {Mean[Green1[[index ;; index + tau - 1]]],
         Mean[Green2[[index ;; index + tau - 1]]], Mean[Green3[[index
            ;; index + tau - 1]]], Mean[Green4[[index ;; index + tau - 1]]]},
        {Mean[Blue1[[index ;; index + tau - 1]]], Mean[Blue2[[index ;;
            index + tau - 1]]], Mean[Blue3[[index ;; index + tau - 1]]],
         Mean[Blue4[[index ;; index + tau - 1]]]}, q], {x, approximate,
        approximate - range, approximate + range}, Method → "PrincipalAxis"],
     AppendTo[location, ans[[2, 1, 2]]]}], {q, 1, 6}],


   If[index > 1, {error = (location - locationold)^2, AppendTo[results, error]}],
   locationold = location,
   index = index + tau}, {If[(size / tau) > amount, amount, Floor[size / tau]]}],
  AppendTo[allanvar1, {{tau * 0.57, (Sqrt[Mean[#[[1]] & /@ results // N] / 2])},
   ErrorBar[(Sqrt[StandardDeviation[#[[1]] & /@ results // N] / 2]) /
     Sqrt[Length[#[[1]] & /@ results // N]]]}],
  AppendTo[allanvar2, {{tau * 0.57, (Sqrt[Mean[#[[2]] & /@ results // N] / 2])},
   ErrorBar[(Sqrt[StandardDeviation[#[[2]] & /@ results // N] / 2]) /
     Sqrt[Length[#[[2]] & /@ results // N]]]}],
  AppendTo[allanvar3, {{tau * 0.57, (Sqrt[Mean[#[[3]] & /@ results // N] / 2])},
   ErrorBar[(Sqrt[StandardDeviation[#[[3]] & /@ results // N] / 2]) /
     Sqrt[Length[#[[3]] & /@ results // N]]]}],
  AppendTo[allanvar4, {{tau * 0.57, (Sqrt[Mean[#[[4]] & /@ results // N] / 2])},
   ErrorBar[(Sqrt[StandardDeviation[#[[4]] & /@ results // N] / 2]) /
     Sqrt[Length[#[[4]] & /@ results // N]]]}],
  AppendTo[allanvar5, {{tau * 0.57, (Sqrt[Mean[#[[5]] & /@ results // N] / 2])},
   ErrorBar[(Sqrt[StandardDeviation[#[[5]] & /@ results // N] / 2]) /
     Sqrt[Length[#[[5]] & /@ results // N]]]}],
  AppendTo[allanvar6, {{tau * 0.57, (Sqrt[Mean[#[[6]] & /@ results // N] / 2])},
   ErrorBar[(Sqrt[StandardDeviation[#[[6]] & /@ results // N] / 2]) /
     Sqrt[Length[#[[6]] & /@ results // N]]]}],

  tau = tau * 2},

 {number}];
plotS1 = ErrorListLogLogPlot[{allanvar1, allanvar2, allanvar3, allanvar4,
   allanvar5, allanvar6}, Joined → True, ImagePadding → 60, Frame → True,
  PlotStyle → Thick, FrameLabel → {Style["τ", 20], Style["σ(τ)", 20]}];

Print[plotS1];
{allanvar1, allanvar2, allanvar3, allanvar4, allanvar5, allanvar6}]
```

This function computes the Allan deviation.

Bibliography

[1] T. G. Phillips and J. Keene, "Submillimeter astronomy [heterodyne spectroscopy]," Proceedings of the IEEE **80,** 1662–1678 (1992).

[2] R. Drever, J. L. Hall, F. Kowalski, J. Hough, G. Ford, A. Munley, and H. Ward, "Laser phase and frequency stabilization using an optical resonator," Applied Physics B **31,** 97–105 (1983).

[3] *Color Sensor TCS 3414*, ams AG, 2011.

[4] *Bristol 521 Series*, Bristol, 2014.

[5] P. Potuluri, M. Gehm, M. Sullivan, and D. Brady, "Measurement-efficient optical wavemeters," Optics Express **12,** 6219–6229 (2004).

[6] S. Roy, S. Chaudhuri, and C. Unnikrishnan, "A simple and inexpensive electronic wavelength-meter using a dual-output photodiode," American Journal of Physics **73,** 571–573 (2005).

[7] D. J. Jones, S. A. Diddams, J. K. Ranka, A. Stentz, R. S. Windeler, J. L. Hall, and S. T. Cundiff, "Carrier-envelope phase control of femtosecond mode-locked lasers and direct optical frequency synthesis," Science **288,** 635–639 (2000).

[8] *FC1500-250-WG*, Menlo Systems Inc., 2015.

[9] J. Reeves, personal communication.

[10] B. Redding, S. M. Popoff, and H. Cao, "All-fiber spectrometer based on speckle pattern reconstruction," Optics Express **21,** 6584–6600 (2013).

[11] B. Gao, Z. Shi, and R. W. Boyd, "Design of flat-band superprism structures for on-chip spectroscopy," Optics Express **23,** 6491–6496 (2015).

[12] J. Ishikawa, N. Ito, and K. Tanaka, "Accurate wavelength meter for cw lasers," Applied Optics **25,** 639–643 (1986).

[13] D. W. Preston, C. E. Wieman, and K. M. Siegbahn, "Doppler free-saturated absorption spectroscopy: Laser spectroscopy," American Journal of Physics **64,** 1432–1436 (1996).

Index