NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

MEASUREMENTS OF OPTICAL TURBULENCE AND ANALYSIS USING MACHINE LEARNING

by

Antonios Sklavounos

December 2021

Co-Advisors:	Joseph A. Blau
	Keith R. Cohn
Second Reader:	Amanda R. Coleman

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** December 2021	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE MEASUREMENTS OF OPTICAL TURBULENCE AND ANALYSIS USING MACHINE LEARNING	**5. FUNDING NUMBERS** RPP02
6. AUTHOR(S) Antonios Sklavounos	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research, Arlington, VA 22203-1995	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

Optical turbulence impacts the performance of laser weapons and laser communication by disrupting the focus of the laser beam. It is important to characterize the turbulence along the beam path in order to predict the performance of these systems. Unfortunately, the equipment needed to measure optical turbulence is delicate. A previous thesis found that the turbulence can be estimated using machine learning regression analysis trained on simple atmospheric measurements that can be made with more robust instruments. Machine learning regression analysis is a powerful tool to model complex phenomena with no clear analytical relationship, although extensive data sets are required to train the machine learning model. For this thesis, we measured optical turbulence and various atmospheric parameters (air temperature, humidity, solar flux, etc.) over many months. Using measured atmospheric parameters as inputs, we developed an ensemble of bagged trees regression model with optical turbulence as the response. Overall, this model showed good agreement with the measured values of turbulence. This indicates turbulence could be predicted using these more robust instruments coupled with a machine learning regression model.

14. SUBJECT TERMS turbulence, machine learning, laser energy, laser propagation	**15. NUMBER OF PAGES** 67
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

MEASUREMENTS OF OPTICAL TURBULENCE AND ANALYSIS USING MACHINE LEARNING

Antonios Sklavounos
Plotarhis, Hellenic Navy
BNS, Hellenic Naval Academy, 2007

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED PHYSICS

from the

NAVAL POSTGRADUATE SCHOOL
December 2021

Approved by: Joseph A. Blau
Co-Advisor

Keith R. Cohn
Co-Advisor

Amanda R. Coleman
Second Reader

Joseph P. Hooper
Chair, Department of Physics

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Optical turbulence impacts the performance of laser weapons and laser communication by disrupting the focus of the laser beam. It is important to characterize the turbulence along the beam path in order to predict the performance of these systems. Unfortunately, the equipment needed to measure optical turbulence is delicate. A previous thesis found that the turbulence can be estimated using machine learning regression analysis trained on simple atmospheric measurements that can be made with more robust instruments. Machine learning regression analysis is a powerful tool to model complex phenomena with no clear analytical relationship, although extensive data sets are required to train the machine learning model. For this thesis, we measured optical turbulence and various atmospheric parameters (air temperature, humidity, solar flux, etc.) over many months. Using measured atmospheric parameters as inputs, we developed an ensemble of bagged trees regression model with optical turbulence as the response. Overall, this model showed good agreement with the measured values of turbulence. This indicates turbulence could be predicted using these more robust instruments coupled with a machine learning regression model.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

GPGGA	Global positioning system fix data (time, position, fix type data)
GPR	Gaussian Process Regression
GPRMC	Recommended minimum specific GPS/Transit data
GPS	Global Positioning System
IRGASON	Integrated CO2/H2O Open-Path Gas Analyzer and 3D Sonic Anemometer
ML	Machine Learning
MSE	Mean Square Error
NAVSLaM	Navy Atmospheric Vertical Surface Model
NPS	Naval Postgraduate School
PSD	Power Spectral Density

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First and foremost, I would like to express my deep and sincere gratitude to my research advisors, Professors Joseph Blau and Keith Cohn, for their continuous support and guidance throughout my thesis research.

In addition, I am extremely grateful to my family for their love, caring and sacrifices for educating and preparing me for my future. I am very much thankful to my wife for her love, understanding, and continuing support to complete this research work.

Last but not least, I would like to express my gratitude to the Hellenic Navy for giving me the opportunity to study at the Naval Postgraduate School.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. PROBLEM STATEMENT

Modern militaries are increasingly using more advanced technologies on the battlefield. For example, laser-based communication systems and laser weapons to defend against surface and air targets (drones or missiles) are becoming more widely deployed.

Optical turbulence affects the performance of laser weapons and laser communication systems by disrupting the focus of the laser beam. Therefore, it is important to measure or reliably estimate the amount of turbulence along the beam path to predict the performance of these systems. Unfortunately, conventional methods for measuring optical turbulence demand expensive and delicate equipment, so in many cases these methods are not feasible to deploy. Validated physics-based models, such as the Navy Atmospheric Vertical Surface Model (NAVSLaM), already exist for characterizing turbulence in homogenous environments [1]. It is difficult, however, to model turbulence in a more complex environment like a ship where the turbulence is affected not only by the ocean and air temperatures but also by the heating of the ship's deck and airflow around the superstructure. In such cases, an alternate approach to physics-based models is to use machine learning regression analysis trained on simple atmospheric measurements from instruments that are more robust and easier to deploy [1], [2].

B. THESIS OVERVIEW

Machine learning (ML) regression analysis is a powerful tool to model phenomena with no clear analytical relationship, although extensive data sets are required to train the ML model. The purpose of this thesis is to explore the feasibility of ML regression methods using MATLAB, a mathematical programming environment published by MathWorks. For a ten-month period, the optical turbulence along with various atmospheric parameters (air temperature, humidity, solar flux, etc.) were recorded for use as a training set for various ML methods. These methods used the measured atmospheric parameters as inputs in order to predict the level of atmospheric turbulence. These predictions were then compared to data sets excluded from the training process to test the robustness of the ML models.

C. THESIS ORGANIZATION

This thesis is structured as follows. Chapter II begins with a brief explanation of optical turbulence and how it is measured. Chapter III presents the overall setup of the experiment describing each instrument that we used for the measurements. Chapter IV explains the methodology of the machine learning regression and in more detail the ensemble of bagged trees model that we used for this thesis. Chapter V presents the experimental results and analysis. Finally, conclusions and recommendations for future work are provided in Chapter VI.

II. TURBULENCE CHARACTERIZATION AND MEASUREMENTS

The atmosphere is a viscous fluid whose flow can be laminar or turbulent. Laminar flow means the fluid particles move in parallel layers with no mixing between the layers. The velocity profile features in laminar flow are uniform and smooth; i.e., they change in some regular fashion. As the fluid velocity increases, dynamic mixing can cause the velocity profile to lose its laminar features and acquire random sub-flows, producing swirling regions of air called turbulent eddies.

The non-dimensional Reynolds number predicts the transition from laminar to turbulent flow. This number is defined as $R_e = \frac{V l_f}{v_k}$, where V is the characteristic fluid velocity, l_f represents some characteristic size dimension of the flow, and v_k is the kinematic viscosity of the fluid. The higher the Reynolds number, the more likely the flow is to be turbulent.

If the Reynolds number is greater than some critical value that depends on the fluid properties and geometry, then the flow usually becomes turbulent. Near the surface, this critical Reynolds number is a few thousand [3]. Near the ground, the characteristic dimension l_f is a few meters, the wind velocity V is typically a few m/s, and the kinematic viscosity of air is $v_k \sim 1.5 \times 10^{-5}\ \mathrm{m^2/s}$ [4]. This yields a Reynolds number of the order of $R_e \sim 10^5$. This is much larger than the critical Reynolds number near the surface, so we can treat the airflow near the surface as fully turbulent.

To truly understand the structure and dynamics of a turbulent atmosphere, it is helpful to consider the Kolmogorov energy cascade theory of turbulence. Kinetic energy is injected into the turbulent eddies from wind shear and heat convection. The largest eddies tend to occur at sizes on the order of $L_0 \sim 10\ \mathrm{m}$ to $100\ \mathrm{m}$ near the surface, where L_0 is often called the outer scale of turbulence [4]. The outer scale depends on the weather conditions and the height above the ground.

These large eddies break apart into smaller and smaller eddies. Since the Reynolds number is large near the surface, the kinetic energy from large eddies gets transferred to

smaller eddies with negligible viscous losses. Nevertheless, this process does not continue forever since friction can no longer be neglected at small eddy sizes. At some characteristic inner scale size l_0, friction will dissipate the kinetic energy as heat; this inner scale is on the order of millimeters close to the surface.

Eddie sizes in between the outer and inner scales define the inertial subrange. The outer scale is usually assumed to grow linearly with the altitude near the surface, as shown in Figure 1 [4]. On the other hand, the inner scale is on the order of 1 mm to 10 mm near the ground or a few centimeters in the troposphere and stratosphere. The overall energy cascade within the inertial subrange is illustrated in Figure 1.

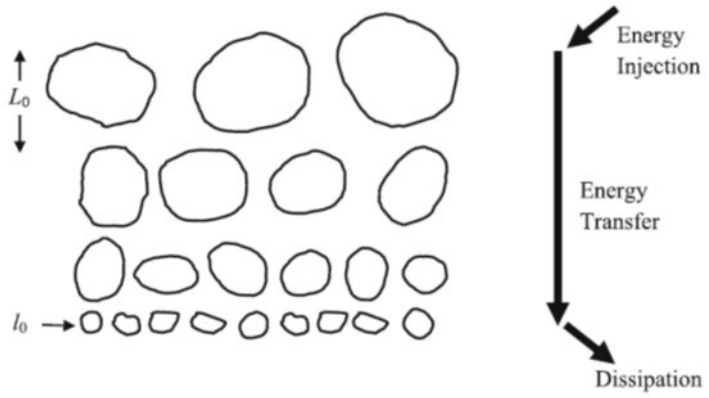


Figure 1. Kolmogorov cascade theory of turbulence. Denotes the outer scale and the inner scale. Eddies between the inner and outer scale form the inertial subrange. Source: [4].

Kolmogorov, using dimensional analysis, demonstrated that in the inertial subrange, the average of the square of the wind speed difference between two points separated by a distance R is given by [4]

$$D_{VV}(R) = \langle (V_1 - V_2)^2 \rangle = C_V^2 R^{2/3}, l_0 < R < L_0 \tag{1}$$

The parameter $D_{VV}(R)$ is called the structure-function for velocity; it represents a statistical measure of the wind speed fluctuations due to turbulence. The average is carried

out over all points R_2 a distance R from a point R_1. Kolmogorov argued that C_V^2 (called the velocity structure constant) is approximately a constant within the inertial subrange, $l_0 < R < L_0$. Thus, C_V^2 characterizes the amplitude of the wind speed fluctuations about neighboring points [5].

A. REFRACTIVE INDEX FLUCTUATIONS

Temperature fluctuations in the atmosphere produce density fluctuations, which in turn result in refractive index fluctuations. More precisely, the index of refraction at a point $\boldsymbol{r}$ is given by Equation (2) [4]:

$$n(\boldsymbol{r}) \cong 1 + 79 * 10^{-6} * \frac{P(\boldsymbol{r})}{T(\boldsymbol{r})}, \tag{2}$$

where P is the pressure (mbar) and T is the temperature (K). Since the pressure fluctuations are often negligible, then according to Equation (3), refractive index fluctuations are primarily due to temperature fluctuations. The fluctuations of the refractive index between eddies tend to be very small; however, as an optical beam passes through many eddies, the net result can cause the beam to scintillate or wander.

Kolmogorov's theory [4] for velocity fluctuations described previously can be extended to temperature fluctuations, resulting in a similar structure-function for temperature,

$$D_T(R) = \langle (T_1 - T_2)^2 \rangle = \begin{cases} C_T^2 {l_0}^{-4/3} R^2, R < l_0 \\ C_T^2 R^{2/3}, l_0 < R < L_0 \end{cases}, \tag{3}$$

where T_1 and T_2 are the temperatures of two arbitrary points, R is the distance between them, C_T^2 is the temperature structure constant, l_0 is the inner scale, and L_0 is the outer scale of the velocity fluctuations.

In full correspondence with the velocity and temperature structure functions, the index of refraction structure function can be defined according to Kolmogorov [4] as

$$D_n(R) = \begin{cases} C_n^2 {l_0}^{-4/3} R^2, & R < l_0 \\ C_n^2 R^{2/3}, & l_0 < R < L_0 \end{cases}, \tag{4}$$

where C_n^2 is the refractive index structure parameter (with SI units of $\mathrm{m}^{-2/3}$). Physically, C_n^2 corresponds to a statistical average of the magnitude of the refractive index fluctuations. The larger the value of the structure parameter, the stronger the turbulence. Typically, for weak turbulence, C_n^2 values are on the order of 10^{-16} $\mathrm{m}^{-2/3}$ or less, while for strong turbulence, C_n^2 is on the order of 10^{-13} $\mathrm{m}^{-2/3}$ or greater.

Since the temperature fluctuations are the primary drivers of index fluctuation, the value of C_n^2 can be estimated through the temperature structure constant C_T^2 using the Equation (5):

$$C_n^2 = \left(79 \times 10^{-6} \frac{P}{T^2}\right)^2 C_T^2, \tag{5}$$

where P is the pressure (mbar) and T is the temperature (K). Turbulence can vary along the propagation path of a laser beam and tends to be a strong function of the height above the ground. Thus, when we have a vertical or slant propagation path, C_n^2 often varies (often by orders of magnitude) as a function of the height above the ground, as we can see in Figure 2.

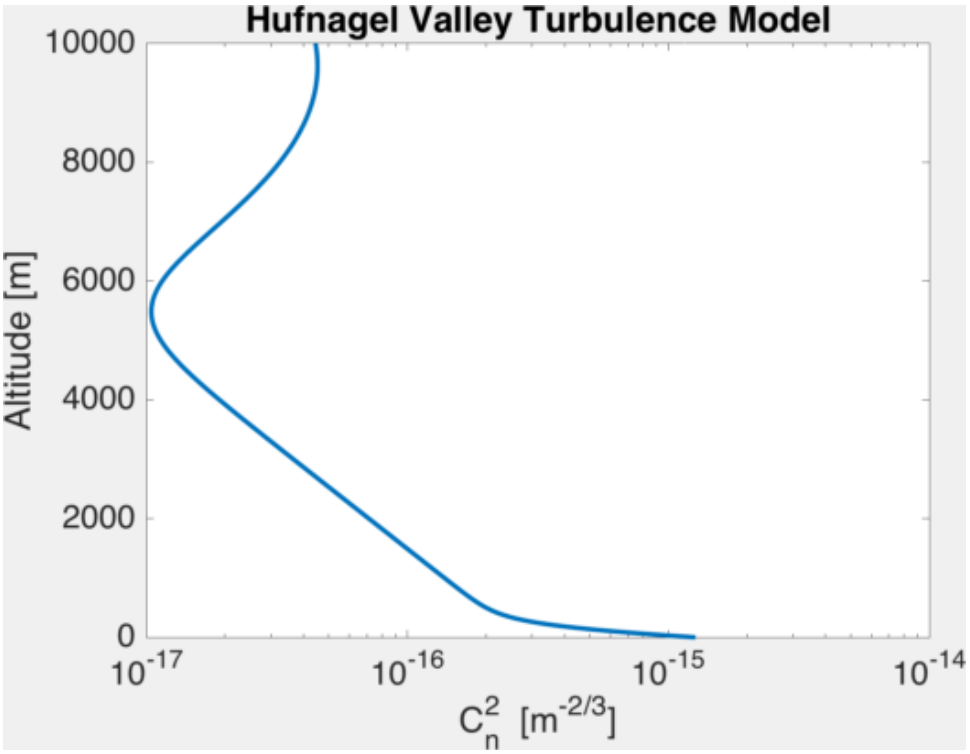


Figure 2. Altitude variation of the refractive-index structure parameter for the Hufnagel Valley model. Source: [6]

The plot in Figure 2 depicts the Hufnagel-Valley 5/7 model; it is a commonly used empirical model derived from measurements taken over the land [4], [7]. The most important feature is that C_n^2 drops dramatically (by an order of magnitude or so) in the first few hundred meters. There are a couple of physical reasons why the turbulence is much more significant close to the surface. First, the temperature differentials between the ground and air directly above create strong temperature gradients, producing variations in the density and thus the index of refraction between the air cells; these air cells tend to mix and smooth out their differences as the altitude increases. Moreover, if there is significant wind shear at the ground level, it can stir up the turbulent cells resulting in larger random fluctuations. These larger fluctuations imply larger values of C_n^2.

The Fried coherence length for spherical waves is given by the Equation (6):

$$r_0 = 2.1 * \left[1.46k^2 \int_0^R C_n^2(z)\left(1 - \frac{z}{R}\right)^{5/3} dz\right]^{-3/5}. \quad (6)$$

The Fried parameter is one of the most important parameters for characterizing turbulence's effects on an optical system [7].

The second term inside the integral $\left(1 - \frac{z}{R}\right)^{5/3}$ is a path weighting function; this term will be smaller as one gets closer to the target (i.e., as z gets closer to R), which means C_n^2 closer to the target has less impact on r_0. This makes sense since turbulence near the source will have a greater impact on the beam than turbulence near the target. From the equation just given, note that r_0 is smaller if C_n^2 is larger; so stronger turbulence leads to a smaller r_0.

Specifically, the Fried parameter is a measure of the transverse coherence; it expresses how well the laser beam wavefront stays in phase throughout a transverse cross-section of the beam. When the Fried parameter/coherence length becomes on the order of the original beam director size or smaller, then the beam will no longer maintain coherence in the transverse direction; hence, the focus of the beam will be significantly impacted by turbulence.

B. MEASUREMENTS OF TURBULENCE USING SONIC ANEMOMETERS

A type of sensor that we use to measure point values of $C_n^2 C_n^2$ is a sonic anemometer (Figure 3). Turbulent eddies have different temperatures and, therefore, different densities, that, as we discussed before, lead to fluctuations in the index of refraction. A sonic anemometer measures very rapid (~50 Hz) temperature fluctuations of the eddies that are passing between its sonic transducers (Figure 4).

Figure 3. Sonic Anemometer. Source: [5].

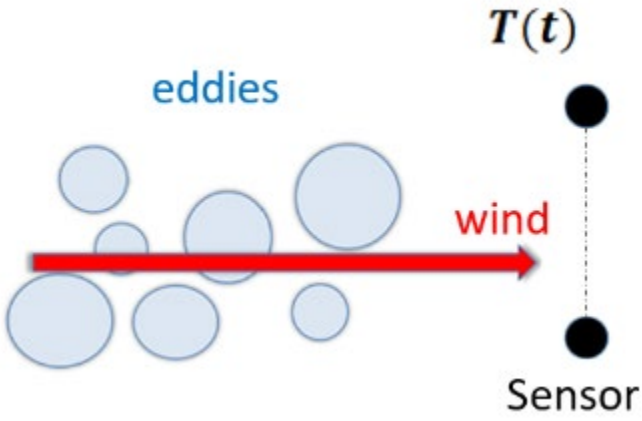


Figure 4. Eddies of various sizes, temperatures, and densities pass over a temperature sensor such as a sonic anemometer. Source: [5].

In order to measure the temperature fluctuations, the sonic anemometers used for this study have three pairs of transducers that emit ultrasonic pulses between them (Figure 5).

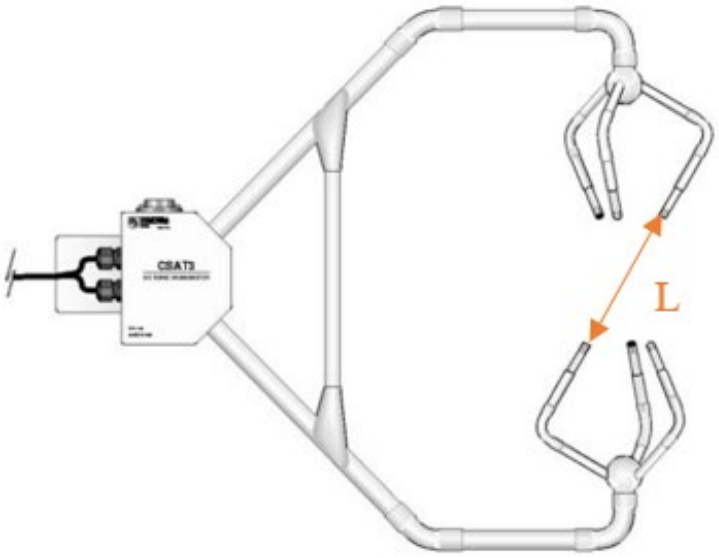


Figure 5. Sonic Anemometer. Source: [5].

Knowing the distance between the transducers and measuring the time it takes for the pulse to travel from one transducer to the other, the sonic anemometer calculates six transit times (back-and-forth for each pair of transducers). The combination of the three pairs of transducers' results gives us, with great precision, the wind's direction and speed in three dimensions.

Equation (7) gives the time of flight, which is measured by the sonic anemometer:

$$t_{1,2} = \frac{L}{c \pm u_a}, \tag{7}$$

where $t_{1,2}$ is the inbound/outbound time of flight of a transmitted pulse between one pair of transducers, L is the distance between transducer pairs, c is the speed of sound, and u_a is the wind speed. The unknown variables are the speed of sound, which depends on temperature, and the wind speed, which are calculated respectively by Equations (8) and (9):

$$c = \frac{L}{2}\left(\frac{1}{t_1} + \frac{1}{t_2}\right), \tag{8}$$

$$u_a = \frac{L}{2}\left[\frac{1}{t_1} - \frac{1}{t_2}\right] \tag{9}$$

Having calculated the speed of sound, we can then calculate the sonic virtual temperature (T_s) by using Equation (10):

$$T_s = \frac{c^2}{\gamma R}, \tag{10}$$

where $\gamma = 1.4$ and $R = 287 \frac{\mathrm{J}}{\mathrm{K \cdot kg}}$ is the universal gas constant [8].

The plot in Figure 6 shows an example of sonic virtual temperature fluctuations versus time measured by a sonic anemometer.

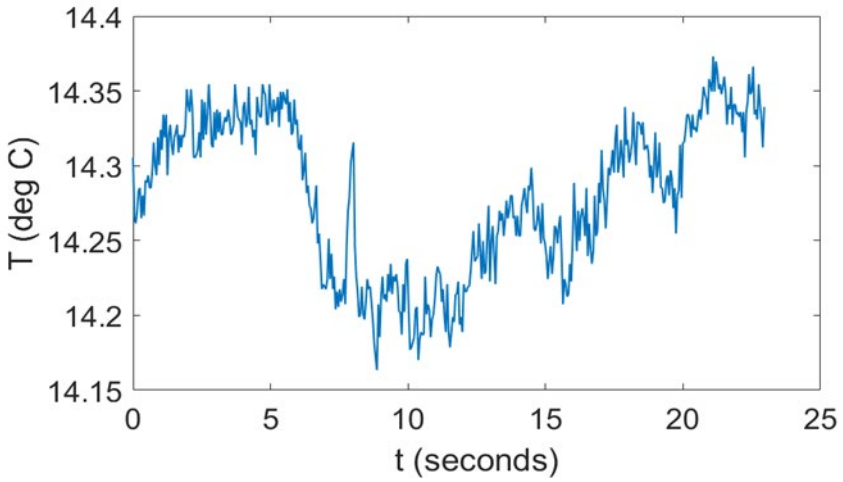


Figure 6. Sonic virtual temperature fluctuations versus time measured by a sonic anemometer. Source: [5].

As we can notice from Figure 6, there are very rapid but also low amplitude fluctuations of the temperature, on the order of 0.01 degrees C or less. These fluctuations correspond to small eddies passing over the sensor. There are also low frequency, high amplitude fluctuations on the order of 0.1 degrees C; these correspond to larger eddies passing over the sensor. Thus, there is a direct relationship between the amplitude and frequency of the sonic temperature fluctuations. Kolmogorov developed a mathematical theory of this

relationship [4], [5]. The plot in Figure 7 is a log-log plot of the Power Spectral Density (PSD), which is the square of the amplitude of the temperature oscillations versus frequency.

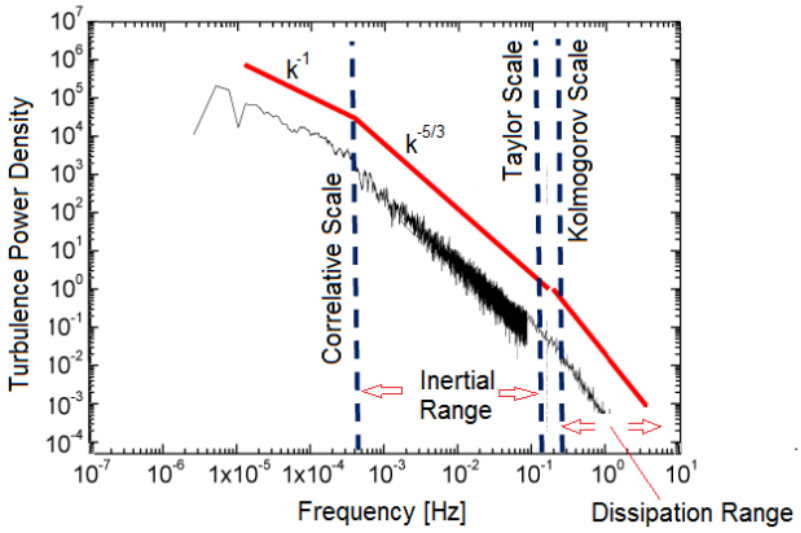


Figure 7. Power Spectral Density (PSD) versus frequency.
Source: [5].

According to Kolmogorov's theory [4], [5], within the inertial subrange, the amplitude of the power spectral density PSD should be related to the frequency f according to:

$$PSD(f) \propto C_T^2 f^{-5/3}.$$

Therefore, if we plot this on a log-log plot as in Figure 7, we should get a -5/3 slope [1]. By fitting a line to the PSD, we can get the value of C_T^2. Then, as mentioned before, the value of C_n^2 can be estimated through the temperature structure constant C_T^2 using Equation (5), repeated here:

$$C_n^2 = \left(79 \times 10^{-6} \frac{P}{T^2}\right)^2 C_T^2,$$

where P is the pressure (mbar) and T is the temperature (K).

III. EXPERIMENTAL SETUP

For this thesis, an experiment was designed and set up on the roof of Spanagel Hall at the Naval Postgraduate School (Figure 8). The whole system was set with a northwest orientation (facing Monterey Bay). This allowed onshore winds from the bay to flow over the apparatus, but the building blocked winds from the southeast. The experimental setup, shown in Figure 9, consisted of a tripod with four devices: an IRGASON (Integrated CO_2/ H_2O Open-Path Gas Analyzer and 3D Sonic Anemometer), an infrared radiometer, a net radiometer, and a global positioning system (GPS) receiver. Data were recorded with a data logger onto a Secure Digital (SD) card and were collected nearly continuously from the end of January 2021 to the end of November 2021. The data included measurements of air temperature, ground temperature, solar flux, wind speed, and humidity.

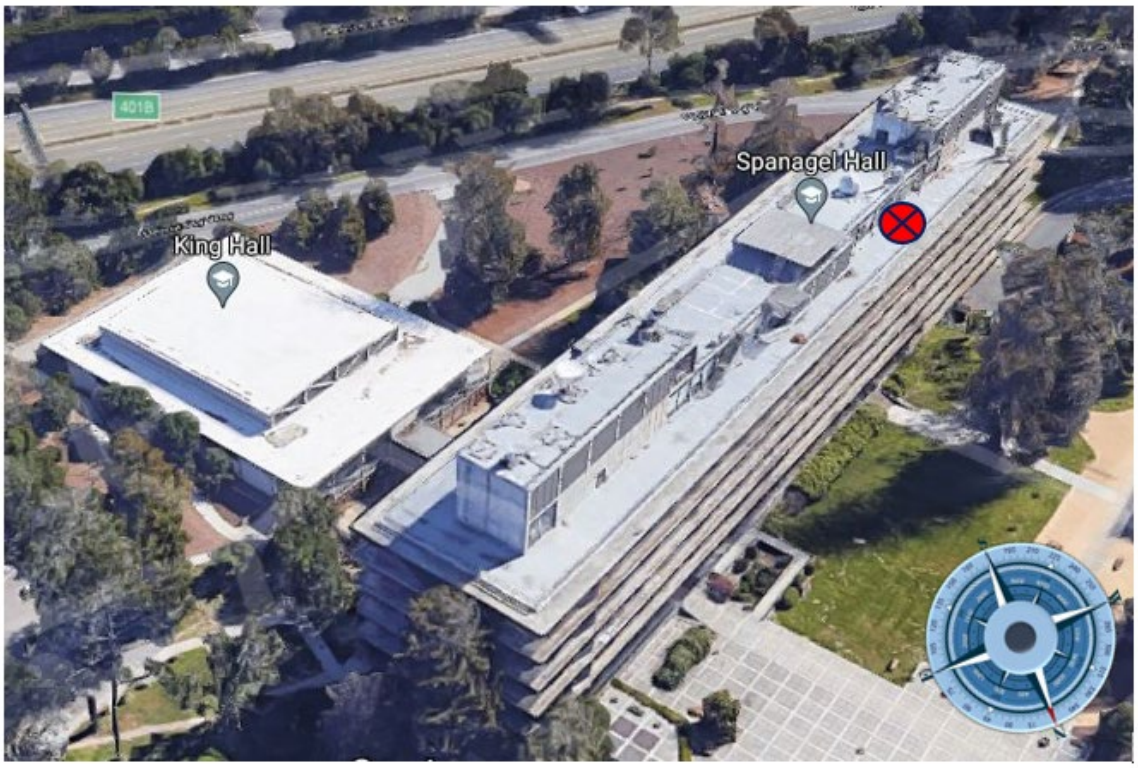


Figure 8. View of Spanagel Hall's roof with the exact location of the system for the experiment.

Figure 9. The experimental setup on the roof of Spanagel Hall.

A. INTEGRATED CO2/H2O OPEN-PATH GAS ANALYZER AND 3D SONIC ANEMOMETER (IRGASON)

The IRGASON (Figure 10) is a three-dimensional sonic anemometer with a built-in optical gas analyzer [9]. The gas analyzer determines the concentrations of carbon dioxide and water vapor, while the sonic anemometer determines three-dimensional wind velocity and sonic temperature. Additionally, the IRGASON includes an air temperature probe (with a solar shield) and a barometer. The EC100 electronics monitor and measure the IRGASON, synchronizing gas, wind, pressure, and temperature data that can then be used to estimate optical turbulence values. The machine learning (ML) model is trained using the logarithm (base 10) of the $C_n^2 C_n^2$ values estimated from the IRGASON measurements.

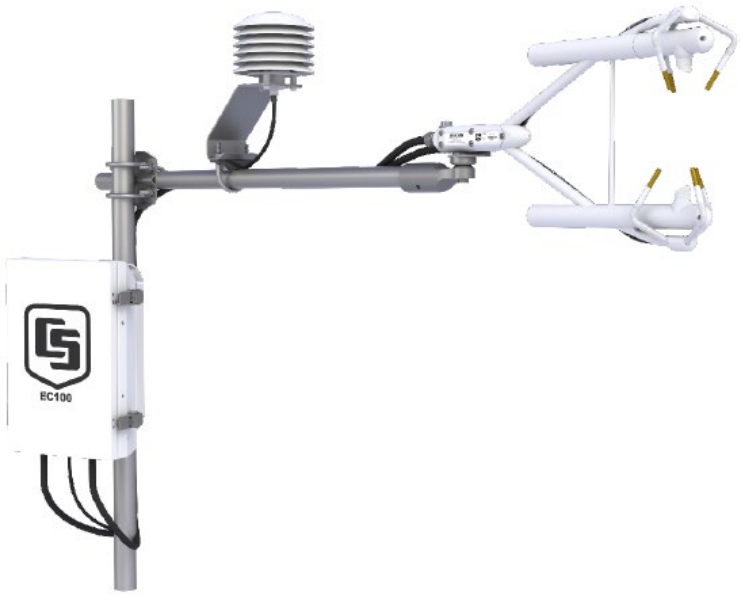


Figure 10. IRGASON system (Integrated CO2/H2O Open-Path Gas Analyzer and 3D Sonic Anemometer). Source: [9]

As shown in Figure 10, the main parts of the IRGASON system are the sonic anemometer, the temperature probe, and the EC100 electronics front panel. The standard outputs from the IRGASON appear in Table 1, later in this chapter.

B. INFRARED RADIOMETER

The infrared radiometer (Figure 11) is a sensor that measures infrared radiation, which was used to determine the surface temperature of Spanagel's roof remotely [10].

Figure 11. Apogee infrared radiometer. Source: [10].

The infrared radiometer determines surface temperature by measuring the blackbody radiation from the surface. It consists of a thermopile detector, a germanium filter, a precise thermistor (for measurement of the reference temperature of the detector), and a circuit for signal processing, all of which are fitted inside an anodized aluminum tube. The sensor outputs the temperature of the target by accounting for the infrared radiation hitting the detector and the temperature of the sensor. Ground temperature differences with the air immediately above the ground are one driver of turbulence.

C. NET RADIOMETER

The SN500SS net radiometer (Figure 12) is a four-piece instrument that contains individual pyranometers that look up toward the sky and down toward the ground [11]. A thermopile detector and filter are housed in an anodized aluminum housing on each radiometer. Each radiometer is heated to prevent dew, frost, snow, and ice from forming on the filter and sensor head. Each radiometer's analog signals are calculated and converted to outputs using an onboard voltmeter.

Figure 12. Apogee net radiometer. Source: [11].

The sun emits primarily shortwave radiation (between 280 and 4000 nm), a portion of which is reflected from the surface of the Earth. Atmospheric and terrestrial molecules absorb the solar light and reradiate it as longwave thermal radiation (between 4 and 100 μm). The difference between inbound (downwelling) and outbound (upwelling) radiation is called net radiation flux. The differences between the inbound and the outbound

shortwave radiation give information about the amount of solar radiation that the ground absorbs. The difference between the temperatures of the ground and the sky is related to differences in the inbound and outbound longwave radiation. As a result of the Earth's shifting location relative to the sun and weather conditions, the net radiation flux varies spatially and temporally. The inbound shortwave radiation flux varies throughout the day as the Sun moves across the sky; it is essentially zero at night. Sunlight is responsible for heating both the ground and the air; this strongly influences the development of turbulence and thus is an important parameter to measure.

D. GPS RECEIVER

The GPS16X-HVS is a global positioning system (GPS) receiver (Figure 13) that provides information about position, velocity, and time [12]. It includes the GPS receiver and antenna with a power supply cable and communication cables in the same housing. The data from the receiver is transmitted to the datalogger at a 38400 baud rate; the GPRMC and GPGGA sentences that contain the time and telemetry data are produced once a second (but only recorded by the datalogger once every five seconds). The main objective of the GPS receiver is to serve as a clock for this experiment since the datalogger's clock was set to synchronize with the GPS data automatically.

Figure 13. Garmin GPS receiver GPS16X-HVS. Source: [12].

E. DATA ORGANIZATION

During the experiment, collected data from the IRGASON, infrared radiometer, net radiometer, and GPS receiver were automatically stored in hourly files. Each day the collected files were merged and transformed into a MATLAB data file. Then this data file was used to calculate the experimental value of $C_n^2 C_n^2$. The data in the MATLAB file and the experimental value of $C_n^2 C_n^2$ were then used in the ML phase. The output elements are described in Table 1.

Table 1. Output elements

Data Element	Description	Units	Frequency
IRGASON (Integrated CO2/H2O Open-Path Gas Analyzer and 3D Sonic Anemometer)			
1	Ux	m/s	50Hz
2	Uy	m/s	50Hz
3	Uz	m/s	50Hz
4	Sonic Temperature	°C	50Hz
5	CO2 Density	$mg \cdot m^{-3}$	50Hz
6	H2O Density	$g \cdot m^{-3}$	50Hz
7	Air Temperature	°C	50Hz
8	Air Pressure	kPa	50Hz
9	CO2 Density from Fast-response Temperature	$mg \cdot m^{-3}$	50Hz
10	Source Housing Temperature	°C	50Hz
11	Detector Housing Temperature	°C	50Hz
Infrared Radiometer			
12	Ground Temperature	°C	0.2Hz
Net Radiometer			
14	Absorbed Solar Radiation (As described in Chapter III.A, it reports shortwave and longwave fluxes in the up and down directions)	W/m^2	0.2Hz
GPS Receiver			
15	Time Synchronization	N/A	0.2Hz

IV. MACHINE LEARNING REGRESSION METHODOLOGY

Validated models for characterizing turbulence in homogenous environments already exist [1], but it is difficult to model turbulence in a more complex environment. For example, on a ship, the turbulence is affected not only by the ocean and air temperatures but also by the heating of the ship's deck and airflow around the superstructure. An alternative approach to physics-based models is to use machine learning (ML) regression analysis, which is a powerful tool to model phenomena with no clear analytical relationship. MATLAB has built-in tools to facilitate ML regression; this section describes the process we applied to predict turbulence based on the experimental data.

A. OVERVIEW OF REGRESSION ANALYSIS

Regression analysis is a method for estimating the response (the value that we are trying to predict) based upon the predictors (other variables that can affect the response) [13]. One common type of regression analysis is linear regression, in which one tries to find a linear relationship between the predictors and the response that best fits the data according to a set of mathematical criteria. In many cases, however, the relationship between the predictors and the response is nonlinear or even non-analytic. In many such cases, ML regression methods can be used where simple linear regression may fail.

In a supervised ML regression method, which we use in this thesis, training data that consists of a set of predictors and known responses are used to build the regression model [14]. A data set is split into a training set and a test set; the training data consists of a random portion of the collected data, and then the remainder of the data is used to test the model [14]. The main goal of ML regression is to build a model using the training data to make accurate predictions for untrained data (i.e., data that was not used to create the model).

B. REGRESSION METHOD PERFORMANCE METRICS

One challenge of data fitting is to capture general trends in the data without overemphasizing variations due to noise in the data. In the following paragraphs, we

explain the meaning of bias and variance and how they affect the mean square error (MSE), which is a metric used by MATLAB for determining the best regression method.

Suppose that we have a training data set with n observations $D = \{(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), (\boldsymbol{x_3}, y_3), \dots, (\boldsymbol{x_n}, y_n)\}$, where $\boldsymbol{x_i}$ are the predictors and y_i are the measured responses that we are trying to model. Also, we assume that there is a mapping between $\boldsymbol{x}$ and $\boldsymbol{y}$ that can be represented mathematically as

$$\boldsymbol{y} = f(\boldsymbol{x}) + \boldsymbol{\varepsilon}, \tag{11}$$

where $\boldsymbol{\varepsilon}$ represents noise in the data (and is assumed to have a mean of zero and a variance of σ^2). Our goal is to use ML regression to find a function $\tilde{f}(\boldsymbol{x}; D)$ that is built using the training data D and approximates the true function $f(\boldsymbol{x})$ while minimizing the effect of the noise ε. We can quantify the performance of $\tilde{f}(\boldsymbol{x}; D)$ for a data set by using the MSE:

$$MSE = E\left[\left(\boldsymbol{y} - \tilde{f}(\boldsymbol{x}; D)\right)^2\right], \tag{12}$$

where $E[\cdots]$ represents the expected value over different choices of the training set D. There is a temptation to choose $\tilde{f}(\boldsymbol{x}; D)$ to make the MSE go to zero over the training data set; however, since the data contains noise, this would represent an overfit to the noise. On the other hand, if $\tilde{f}(\boldsymbol{x}; D)$ exactly matches $f(\boldsymbol{x})$, then $MSE = \sigma^2 > 0$ (the variance of the noise). The problem is that σ^2 is unknown, which makes it difficult to find the optimum $\tilde{f}(\boldsymbol{x}; D)$.

We can expand Equation (12) to illustrate this point further. Inserting Equation (11) into Equation (12), applying the square, and then collecting terms results in

$$MSE = \underbrace{\left\{E\left[\tilde{f}(\boldsymbol{x}; D)\right] - f(\boldsymbol{x})\right\}^2}_{\left(\text{Bias}\left[\tilde{f}(\boldsymbol{x};D)\right]\right)^2} + \underbrace{E\left\{\left(\tilde{f}(\boldsymbol{x}; D) - E\left[\tilde{f}(\boldsymbol{x}; D)\right]\right)^2\right\}}_{\text{Var}\left[\tilde{f}(\boldsymbol{x};D)\right]} + \sigma^2. \tag{13}$$

The first term on the right side of Equation (13) is the square of the bias. Bias is defined as the difference between the average predictions of a model and the "true" measured values that we are trying to predict [15]. If this term is large, then the model is failing to capture important general features in the data. The second term in Equation (13) is the variance, which quantifies the spread of our model predictions for a given set of inputs. If this term

is large, then the model might be too complex, overfitting the data by following the noise (see bottom right of Figure 14). Theoretically, the bias and variance would both be zero, which means the minimum desired MSE is equal to σ^2, as described before.

In practice, it is difficult or impossible to eliminate bias and variance simultaneously; instead, a trade-off in model complexity is sought to minimize the sum of both bias and variance—the so-called bias-variance tradeoff (see Figure 14). The lower plots of Figure 14 illustrate this bias-variance tradeoff. The plots show three models with different characteristics that are fit to the same data. The one on the left has low variance and high bias; it is underfit to the response since it does not follow the general trends. On the right, the model is overfit in that it has a low bias but high variance; it follows both the overall trends (low bias) but also the noise (high variance). The middle plot represents a balance between these two extremes [16].

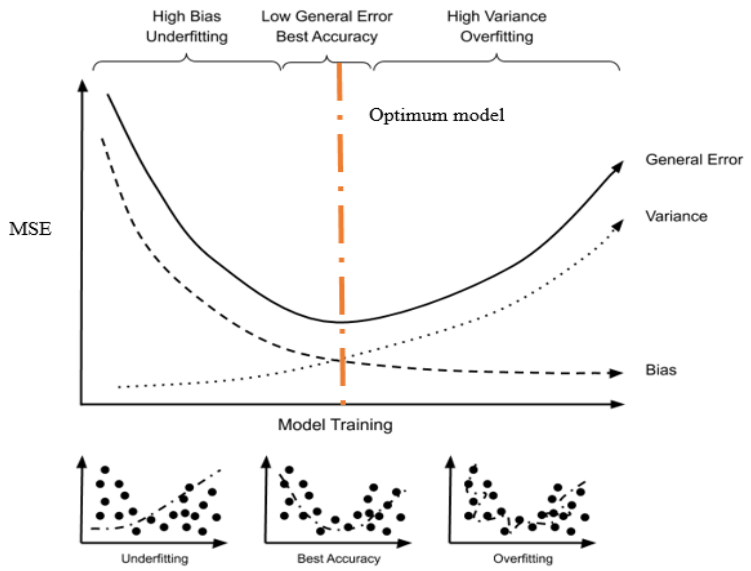


Figure 14. Bias and variance as a function of model complexity. Source: [17].

To prevent overfitting, the training data used to build the model is often partitioned into k subsets called "folds" for cross-validation. One fold is used to validate the model trained on the other $k - 1$ folds. This process is repeated k times so that each fold is used exactly once for validation. Finally, the average error over all the folds is used to estimate the

model's overall performance. The goal is to generate a final model that produces the lowest average error. This model is then applied to the test set (the untrained data). If the MSE of the training data is significantly lower than the MSE of the test data, then the model is likely overfitting to the noise in the training data.

C. MATLAB REGRESSION LEARNER APP

The MATLAB Regression Learner App provides an easy interface to test many of these supervised ML regression methods and determine which one is optimal for a given data set. To select the best regression method, we imported a subset of the experimental data into the MATLAB Regression Learner App, as shown in Figure 15, to serve as training data for the model. Then the App applied many different methods (see Figure 16) to this training set in parallel and reported the root mean square error (RMSE) between the model predictions and the actual response, which allows comparison between the various methods. The RMSE is the square root of the MSE defined in Equation (12). Through this process (the details of which are described in the next chapter), an ensemble of bagged decision trees was determined the optimum for our data.

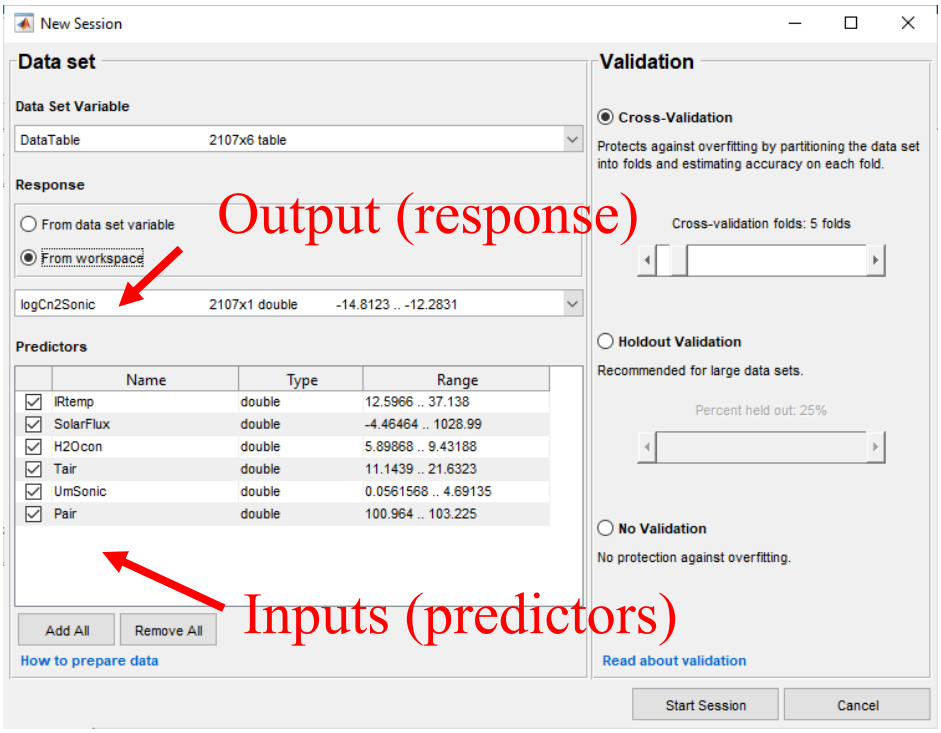


Figure 15. Regression Learner App interface. Data set variables and response

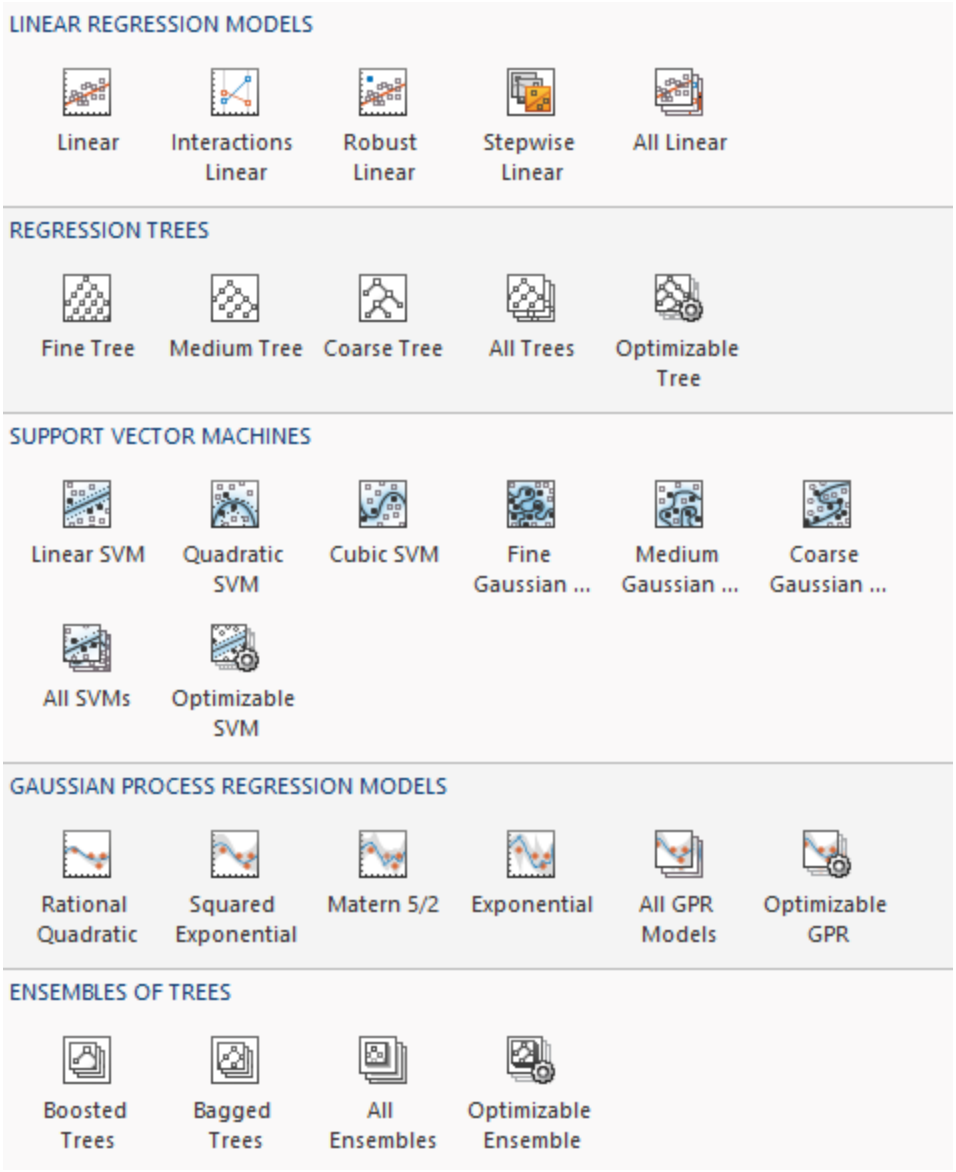


Figure 16. Regression learner models

D. OVERVIEW OF DECISION TREE METHODS

Decision trees are supervised learning algorithms that form the basis for our selected method. A decision tree is a flowchart-like algorithm that makes choices based upon the predictors; it consists of nodes connected by branches (Figure 17). The tree is built starting with all the training data (both predictors and known responses) in the root node. Then the data is split into subsets using certain criteria based upon the values of the predictors, as described in the following section. Each node in the tree where splitting

occurs is referred to as a decision node (red box in Figure 17). The final nodes at the end of all the branches are called leaf nodes (green boxes in Figure 17) since they do not lead to further branches. The predicted response $\hat{y}_i$ for each leaf L_i is the mean $\langle y \in L_i \rangle$ of all the responses for the subset of the training data that remains in that leaf. The resulting decision tree can then be applied to new data by following the decision tree from the root to a leaf node based upon the values of the predictors in the new data.

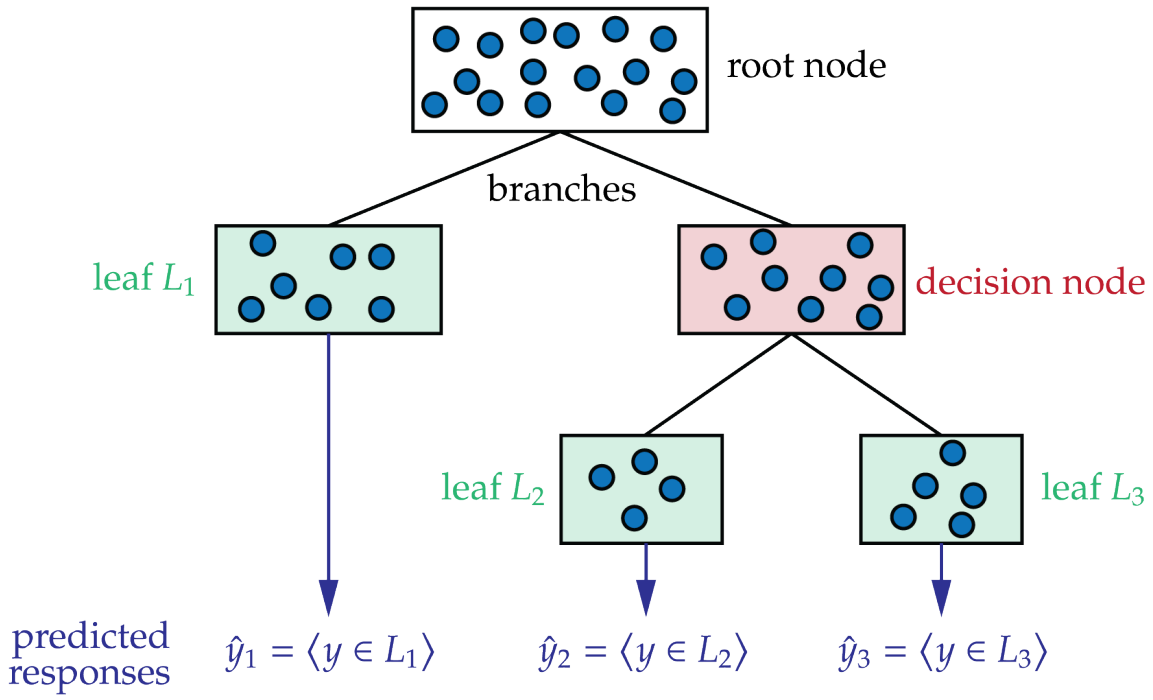


Figure 17. Example of decision tree

1. Splitting Criteria

In a decision tree, the model is built using the training data by recursively splitting the data into subgroups based upon the values of the predictors. There are different criteria that can be used to split the data. The criterion described in this section is the default used by MATLAB [18].

To determine the optimum split criterion at each decision node, the weighted variance for a given node t of the responses is calculated using the Equation (14):

$$\sigma_t^2 = \sum_{j \in T} w_j \left(y_j - \langle y_t \rangle\right)^2, \tag{14}$$

where T is the set that contains all the training observation indices in node t, w_j is the weight of observation j, y_j is the value of the response for observation j, and $\langle y_t \rangle$ is the mean value of the response for node t. The weighting factor w_j allows certain observations to be emphasized over others; if all observations are equally weighted (as we assume in our case), then $w_j = 1/n$, where n is the number of observations in set T.

For each node t, a candidate predictor and split value is tested by splitting the training data into two subsets T_L and T_R . Then the variance for each subset is calculated in the same manner as Equation (14). The probability for an observation to be in node t is given by

$$P(T) = \sum_{j \epsilon T} w_j, \tag{15}$$

and likewise for the subsets T_L and T_R. The *reduction* in weighted variance for the current splitting candidate is given by the Equation (16):

$$\Delta I = P(T)\sigma_t^2 - P(T_L)\sigma_{t_L}^2 - P(T_R)\sigma_{t_R}^2. \tag{16}$$

Finally, ΔI is calculated for each possible choice of predictor and split value; the split candidate that maximizes the reduction ΔI in weighted variance is chosen for that node.

2. Stopping Criteria

The splitting process is repeated recursively along each branch until a stopping condition is reached. In regression trees, a common stopping criterion is when the number of training observations in a node reaches a threshold called the minimum leaf size; this is a parameter that can be adjusted in MATLAB. Another criterion is to limit the maximum number of splits to a preset value; this can also be set in MATLAB by selecting Fine, Medium, or Coarse Tree in the Regression Learner App, as shown in Figure 16 [18], [19].

E. REGRESSION TREE ENSEMBLE USING BOOTSTRAP AGGREGATION (BAGGING)

A regression tree ensemble combines many regression trees in a weighted combination to improve predictive performance and reduce overfitting. For ensemble methods, the training data is split into multiple subsets, as described in the following sections. Then a decision tree is built using each subset, thus creating an ensemble of decision trees. Then, the overall response of the ensemble is a weighted mean of the responses of the trees comprising the ensemble [20].

One method to create an ensemble of regression trees is by bootstrap aggregation (or bagging). For example, assume we have a training dataset *D* with *n* observations, each containing *f* predictors. The bagging process can be broken down into three stages: bootstrap sampling, base modeling, and aggregation, as shown in Figure 18.

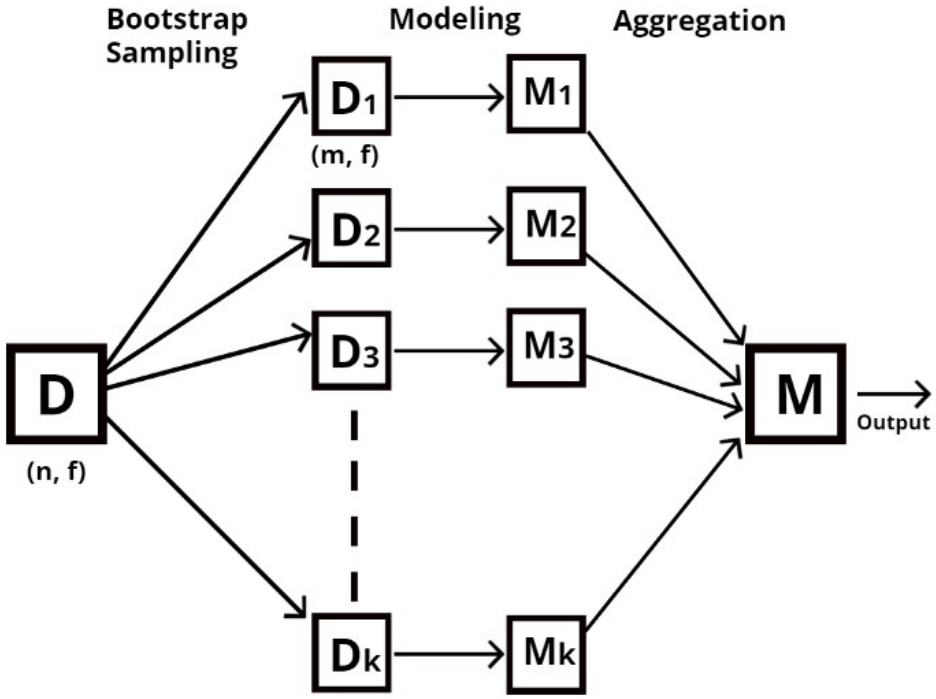


Figure 18. Three steps of bagging – bootstrap sampling, modeling, aggregation. Source:[20]

1. Bootstrap Sampling

The first step in bagging process is to create k smaller subsets $D_1, D_2, \cdots, D_k$, each with the same number of predictors f, and the number of observations $m<n$, by sampling with replacement from the full training set D. A single observation can appear in multiple subsets, as seen in Figure 19. In this example, the full training data set D, which consists of ten observations, is split into k subsets, each with five observations randomly sampled from D; notice that some of the observations appear in multiple subsets.

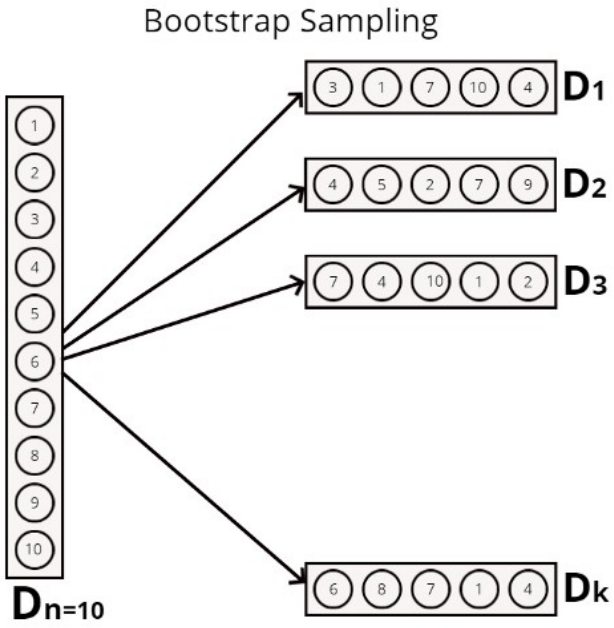


Figure 19. Bootstrap sampling for a sample dataset of ten observations. Source: [20].

2. Modeling

The second step in the bagging process is modeling, where each of the subsets is trained using ML algorithms. Decision tree algorithms, for example, can be used as base models; each model can have different stopping criteria. Base models (also called weak learners, M_1 to M_k in Figure 18) are the models that are trained on each bootstrap dataset. Typically, these base models are too simple for complex data sets, either because of strong

bias or variance. Nonetheless, their combined results often outperform any single base model.

3. Aggregation

By aggregating the k different base models, using cross-validation as described earlier, a final, robust model is created. This allows for the reduction of bias and/or variance, increasing the model accuracy [20]. Because the base models are trained on different subsets, each model's predictions may differ. Depending on the problem statement, different aggregation techniques are used. In our case, we take the mean of all the models' predicted responses.

THIS PAGE INTENTIONALLY LEFT BLANK

V. MODEL TRAINING AND PERFORMANCE ANALYSIS

To analyze the data, we first had to prepare the data, then we selected the optimum ML regression method. Then we trained the model using that method, and finally we applied the model to make predictions using test data that was not used during the training process. These steps are discussed in the following sections.

A. DATA PREPARATION

As discussed in Chapter III, we collected measurements of air temperature, ground temperature, wind velocity, sonic temperature, water vapor concentration, and solar flux over many months; these variables were used as the predictors in our model. The wind velocity and sonic temperatures from the IRGASON were used to estimate C_n^2 using Kolmogorov theory [4], [5], as described in Chapter III. To do this, we took the power spectral density (PSD) of the sonic temperatures over ~40-second intervals (using 2048 observations acquired at 50 Hz) to calculate C_n^2; average values for all the other measurements were calculated over the same time interval. This yielded approximately 2000 observations per day; thus, we obtained a total of approximately 600,000 observations over the entire ten-month experiment. The data for each day was compiled into a single text file that was then imported into MATLAB for analysis. For our model, the response parameter is the logarithm of C_n^2, since C_n^2 can vary over several orders of magnitude. A screenshot of the data prepared in this manner is shown in Figure 20.

B. MODEL SELECTION AND OPTIMIZATION

To select the optimum regression method, we imported all the data into the Regression Learner App in MATLAB for 12 days that were approximately equally dispersed from January to June. Since the training process takes a significant amount of time, it was impractical to run all the methods listed in Figure 16 for the whole data collection period; hence, only a few days were used for this step. To protect against overfitting, cross-validation with five folds was used (as described in Chapter IV). Then the App ran all 19 available regression methods in parallel and reported the RMSE of each model; when the training process was completed, MATLAB highlighted the method with

the lowest RMSE, indicating the model with the best performance using this metric. Also, for each model it produced several plots that can be used to visually observe the model's performance.

	A	B	C	D	E	F	G	H
	CompiledDataWithDates1							
	Time	IRtemp	SolarFlux	H2Ocon	Tair	UmSonic	Pair	LogCn2
	Datetime	Number	Number	Number	Number	Number	Number	Number
1	Time	IRtemp	SolarFlux	H2Ocon	Tair	UmSonic	Pair	LogCn2
2	2021-01-25 13:14:27.760	8.44944	139.1896	5.5980477333069	9.8876037597656	6.57673374956306	100.91347503662	-13.4905217039395
3	2021-01-25 13:15:17.320	8.57144	137.172	5.6556401252747	9.9233093261719	8.61285216661356	100.93549346924	-13.490310484843
4	2021-01-25 13:15:58.280	8.20096	113.6728	5.6733436584473	9.8648986816406	8.32412176776651	100.89084625244	-13.5405685303498
5	2021-01-25 13:16:39.240	7.890232	92.9712	5.681661605835	9.8077087402344	7.94680768561356	100.89817810059	-13.510603829679
6	2021-01-25 13:17:20.200	7.65496	77.964	5.6498284339905	9.7831726074219	8.7826904462672	100.93984222412	-13.568194231324
7	2021-01-25 13:18:01.160	7.410184	81.9232	5.6475081443787	9.7581176757812	10.1183136511654	100.90132141113	-13.6038062697654
8	2021-01-25 13:18:42.120	7.481784	85.8488	5.3649749755859	9.6886291503906	10.6091599257892	100.91076660156	-13.5069545424546
9	2021-01-25 13:19:23.080	7.458936	73.96496	5.423198223114	9.6455383300781	8.8235065286447	100.92832946777	-13.44189517264
10	2021-01-25 13:20:04.040	7.347624	66.434	5.4899516105652	9.5977172851562	7.2994613190028	100.9164352417	-13.5643259946258
11	2021-01-25 13:20:45.000	7.477	65.33	5.4850606918335	9.6667785644531	7.65432159188189	100.92169189453	-13.6734438302071
12	2021-01-25 13:21:25.960	7.255072	60.98632	5.5082564353943	9.6712951660156	8.12244059675682	100.90001678467	-13.45085394941
13	2021-01-25 13:22:06.920	7.40344	71.99168	5.5282559394836	9.5531311035156	7.41343648383338	100.92886352539	-13.4481630128566
14	2021-01-25 13:22:47.880	7.608904	89.564	5.5832538604736	9.5701293945313	6.1564548659033	100.89109802246	-13.6412592784629
15	2021-01-25 13:23:28.840	7.54868	93.248	5.5691776275635	9.6105651855469	7.25509304875661	100.91357421875	-13.5805695518479
16	2021-01-25 13:24:09.800	7.34728	68.2628	5.2806992530823	9.6349182128906	9.8412071499557	100.88565063477	-13.4436745794294
17	2021-01-25 13:24:50.760	7.160512	64.1736	5.470591545105	9.6006774902344	8.98506212592072	100.9122467041	-13.464935361143
18	2021-01-25 13:25:31.720	7.155944	63.11152	5.569100856781	9.6168518066406	6.64056334390585	100.91075134277	-13.7543035319295
19	2021-01-25 13:26:12.680	7.311192	59.83384	5.6290788650513	9.7476806640625	5.42844175138954	100.90881347656	-13.8863366115919
20	2021-01-25 13:26:53.640	7.196896	57.18936	5.6762828826904	9.8019104003906	7.12343642949639	100.90494537354	-13.5084254050644
21	2021-01-25 13:27:34.600	7.29896	60.9756	5.7001194953918	9.7119140625	7.66710135387536	100.90341186523	-13.4820866948813
22	2021-01-25 13:28:15.560	7.302064	78.50856	5.5548176765442	9.7233581542969	6.9637182272791	100.91725158691	-13.657597664798

Figure 20. Data from daily measurements. Columns B through G served as predictors for the regression model; the last column ($logC_n^2$) was the response used for training and validation.

For example, Figure 21 and Figure 22 show the results for one randomly chosen day (using all the observations for that day). Figure 21 indicates that for this data set, the ensemble of bagged trees method has the lowest RMSE (0.23811). Also shown is a plot of the response ($logC_n^2$) for the true (measured) values (blue dots) and the predicted values (orange dots); the horizontal axis is, effectively, the time axis over a 24-hour period. Figure 22 plots the predicted response versus the true (measured) response for the same data set; the diagonal black line corresponds to perfect prediction.

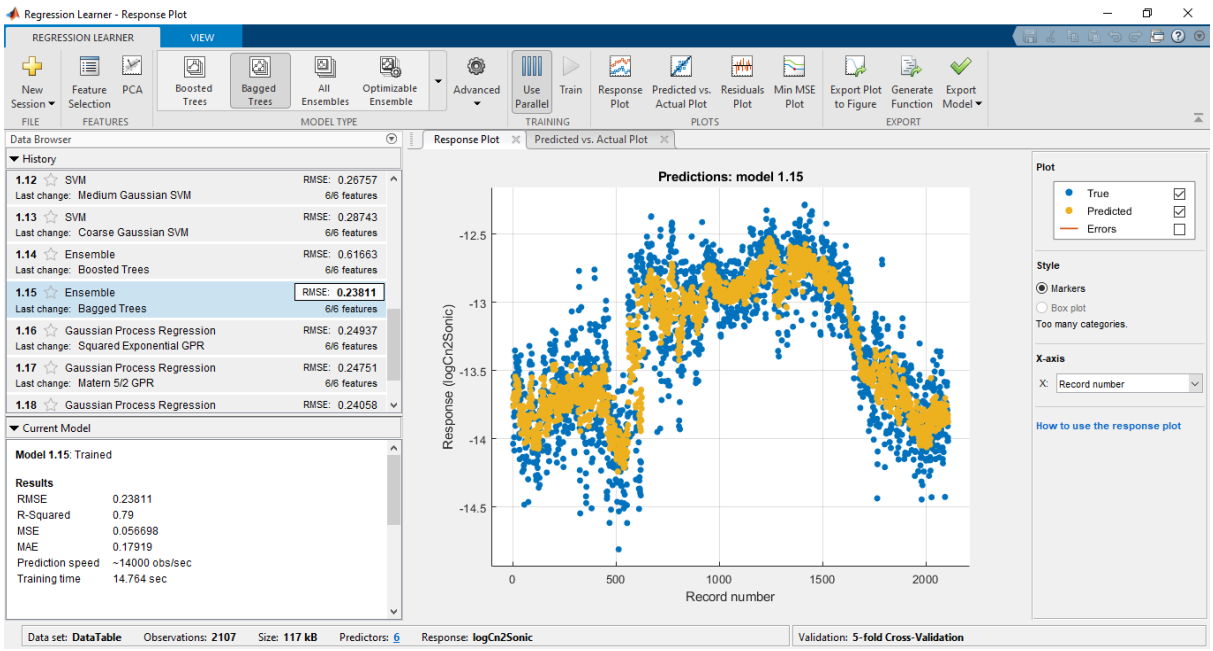


Figure 21. An example of output from the Regression Learner App, showing that the bagged trees method (highlighted in blue on the left) has the lowest RMSE for this case. On the right is a plot of the response ($logC_n^2$) for this method.

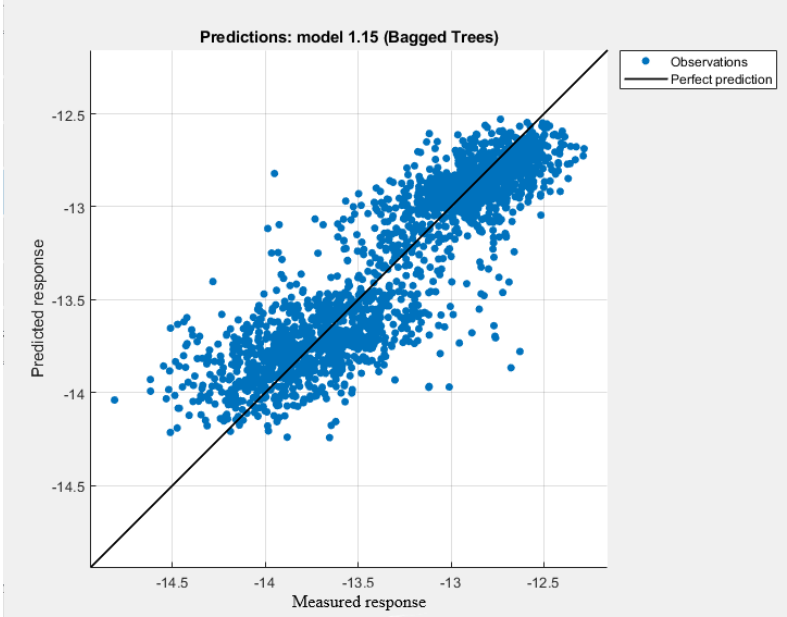


Figure 22. An example of predicted vs. measured response of $logC_n^2$ for the bagged trees method.

From this process, we concluded that the methods that consistently produced the lowest RMSE were the regression tree ensemble using bagging and two types of Gaussian Process Regression: exponential and rational quadratic [21], [22]. For example, Figure 23–Figure 25 show the results for one of these days (June 15, 2021) using these three methods. The figures all look similar, but the bagged trees method gives the lowest RMSE in this case. Note that for all three methods, the plots in Figure 23–Figure 25 show that the models tend to underestimate large values of $logC_n^2$ and overestimate smaller values of $logC_n^2$; however, this is expected behavior if the extremes in the measured response data are due to noise.

Results for all 12 days for these three methods are included in the appendix; the results presented there show that of these three types, the ensemble bagging method had more days with the lowest RMSE. Thus, the ensemble bagging method was selected to proceed with the training using a data set sampled from all the data collected over the entire ten-month observation period.

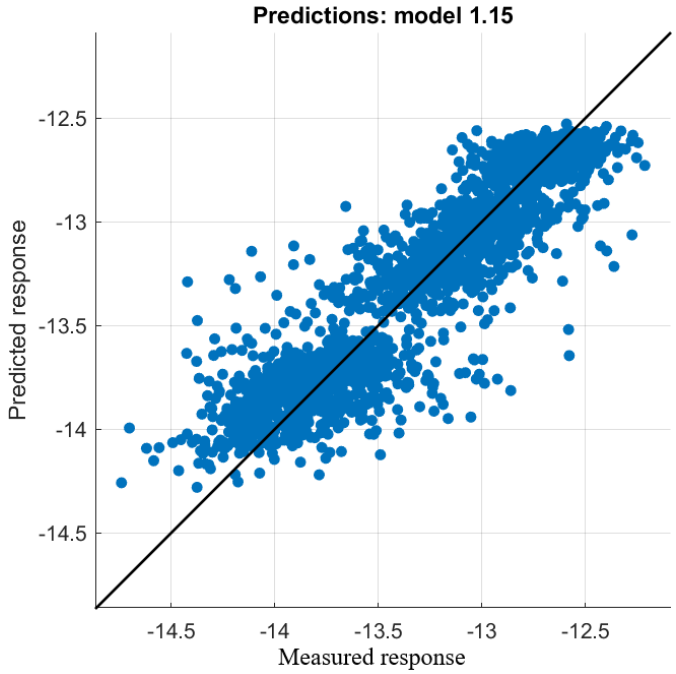


Figure 23. Predicted vs. measured response, $logC_n^2$, for the bagged tree method with RMSE = 0.21784 (Date: June 15, 2021).

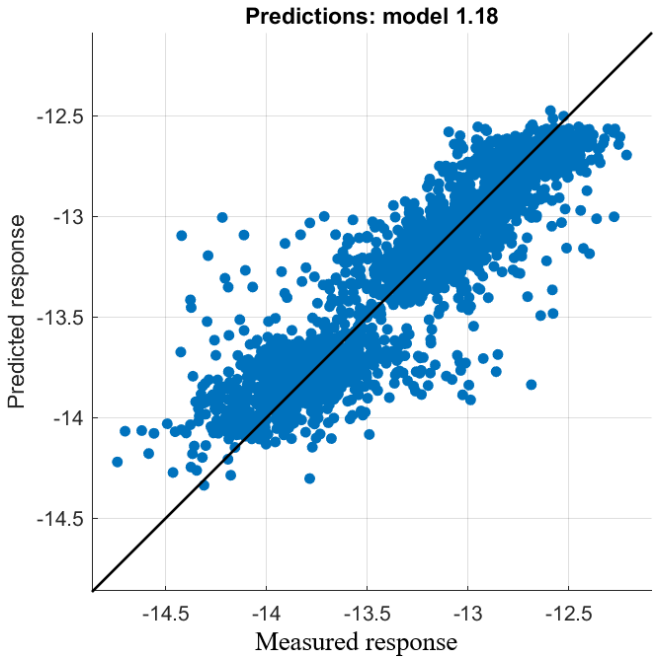


Figure 24. Predicted vs. measured response of $logC_n^2$. Gaussian process regression method: Exponential GPR with RMSE = 0.22566 (Date: June 15, 2021).

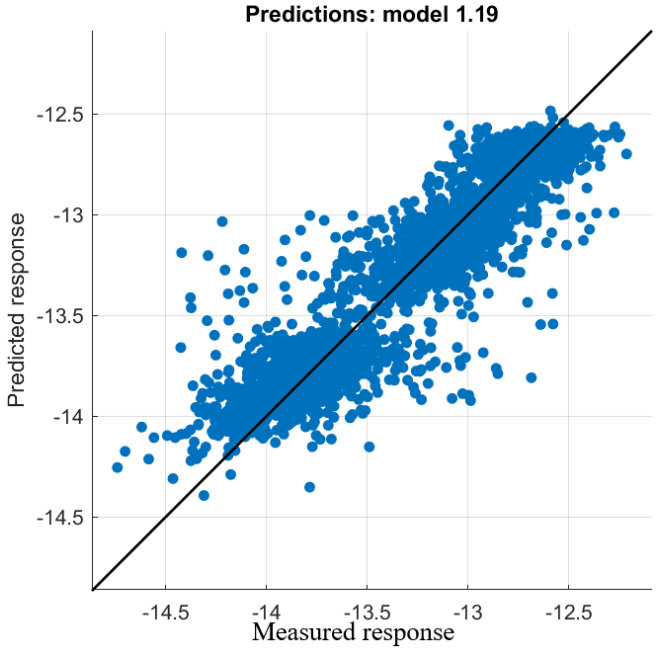


Figure 25. Predicted vs. measured response of $logC_n^2$. Gaussian process regression method: rational quadratic GPR with RMSE = 0.22525 (Date: June 15, 2021).

After we selected the ensemble of bagged trees as the best method for our data, we optimized some of the numerical parameters for this method using several months of data. MATLAB tried many combinations of minimum leaf sizes and number of ensemble base models (or weak learners) to find the optimum response. Minimum leaf size refers to the smallest number of observations allowed per tree leaf. If the number of the observations is less than the minimum leaf size, then the tree is not split any further beyond that leaf node [18]. After this optimization step, a minimum leaf size of 1 and number of ensemble base models equal to 498 produced the lowest RMSE. These values were chosen for building the model using the larger data set.

C. MODEL TRAINING

We then trained the model using a randomly selected 70% portion of all the data for the entire ten-month period (i.e., approximately 420,000 out of the 600,000 total points). Then the remaining ~180,000 observations were used to test the robustness of the model. After the model was trained using the ensemble of bagged trees method, we obtained the plot shown in Figure 26. The blue dots represent the true (measured) values of ($logC_n^2$), while the orange dots represent the predicted values. The horizontal axis corresponds to time throughout the year. The RMSE for the training data is equal to 0.2633. Note there was a gap in the data during April due to an issue with the memory card in the datalogger. Figure 27 zooms in on the same data for a two-week period; the predicted and measured values follow the same trends, with the typical diurnal pattern of stronger turbulence during the day and weaker turbulence at night.

Figure 28 plots the predicted versus measured values of $logC_n^2$ for the same data; the black diagonal line corresponds to perfect agreement. This figure shows that the model overestimates the small values of $logC_n^2$ and underestimates the large values, but the overall agreement is good, as seen by the clustering of the observations near the diagonal line.

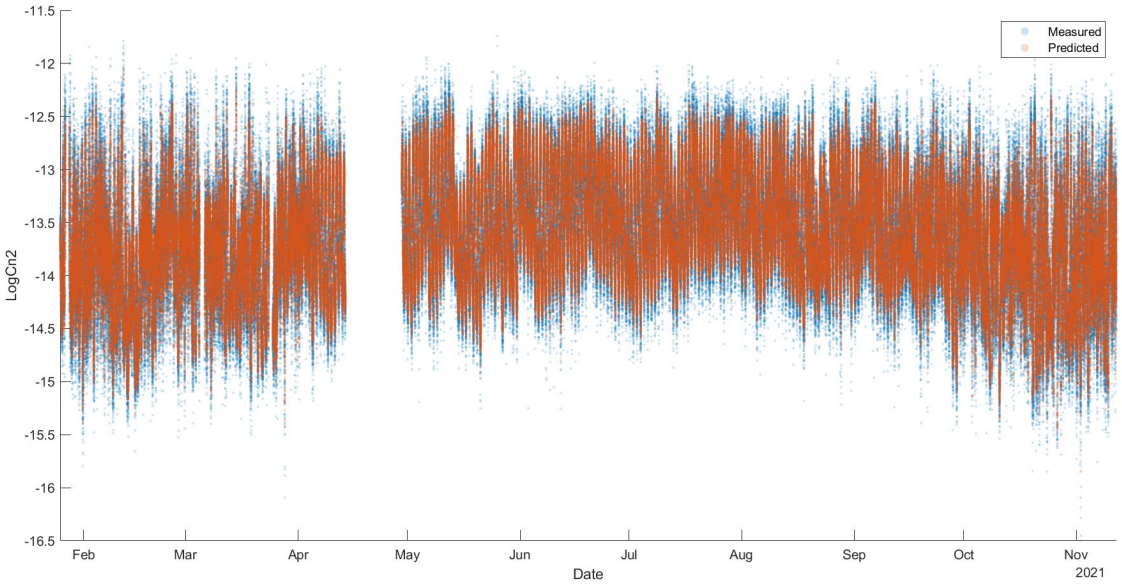


Figure 26. Predicted and measured values of $logC_n^2$ vs. time using the training data set for the entire ten-month period of the experiment. The RMSE is 0.2633 for this data set.

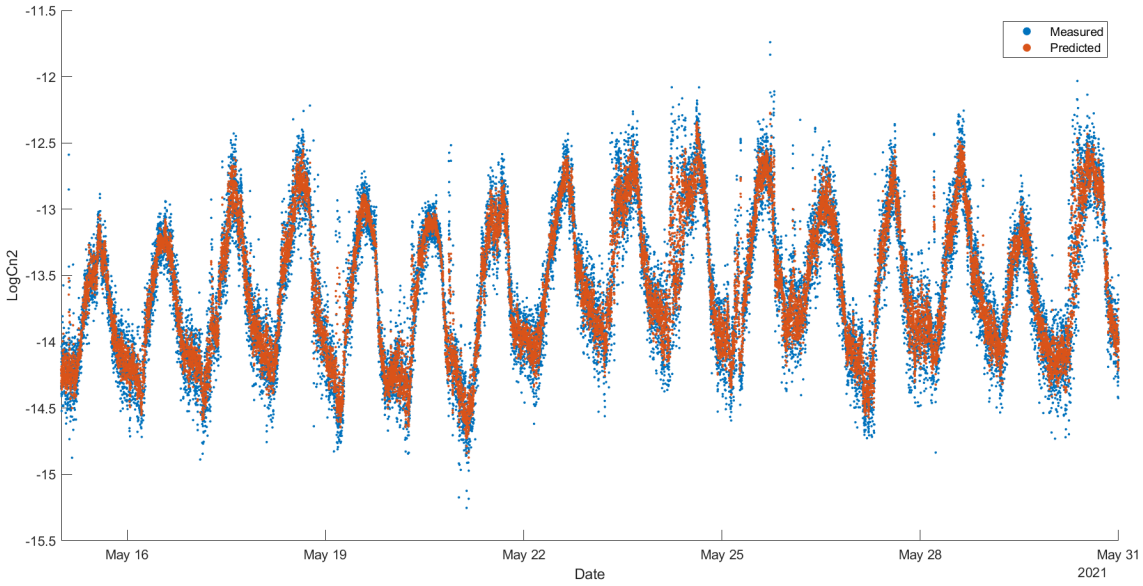


Figure 27. Predicted and measured values of $logC_n^2$ vs. time using the training data set from May 15 through May 31, 2021.

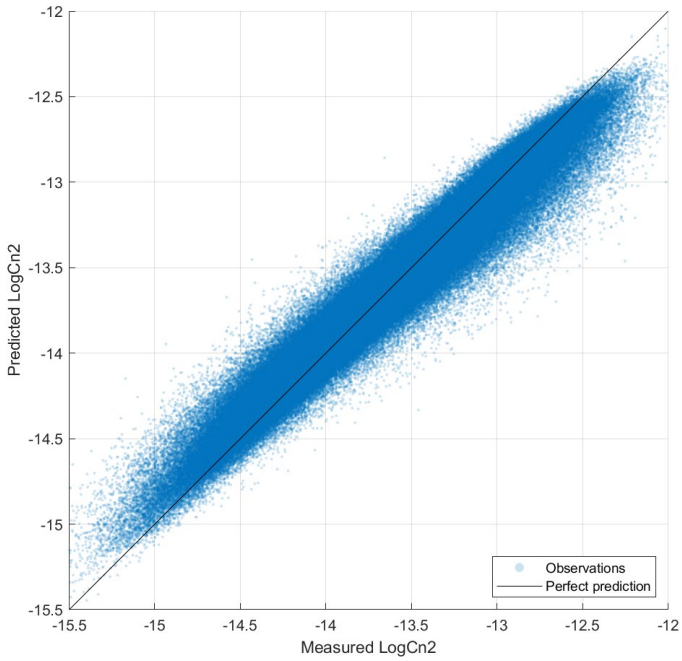


Figure 28. Predicted vs. measured values of $logC_n^2$ using the training data set from the entire ten-month period of the experiment.

D. TESTING THE MODEL WITH THE UNTRAINED DATA

To further test the model, we applied the model to the remaining 30% of the collected data that was not used for training; the results are shown in Figure 29. Again, the blue dots represent the true (measured) values of ($logC_n^2$), while the orange dots represent the predicted values. The horizontal axis corresponds to each observation (specific day and time) in the data set. Like the fit to the training data set, the model overestimates the small values of $logC_n^2$ and underestimates the large values, but the overall trends are well followed, as seen in Figure 30. The RMSE for the untrained data is equal to 0.2560, nearly identical to the RMSE for the training data set. This is evidence that we are not overfitting to the training data set.

Figure 31 again plots the predicted versus measured response of $logC_n^2$ for the untrained data set. The central core of the observations closely follows the idealized diagonal line; however, in this case there are more measured values that fall slightly further away from the predicted values than we saw with the training data set, although these values are not numerous enough to affect the overall RMSE.

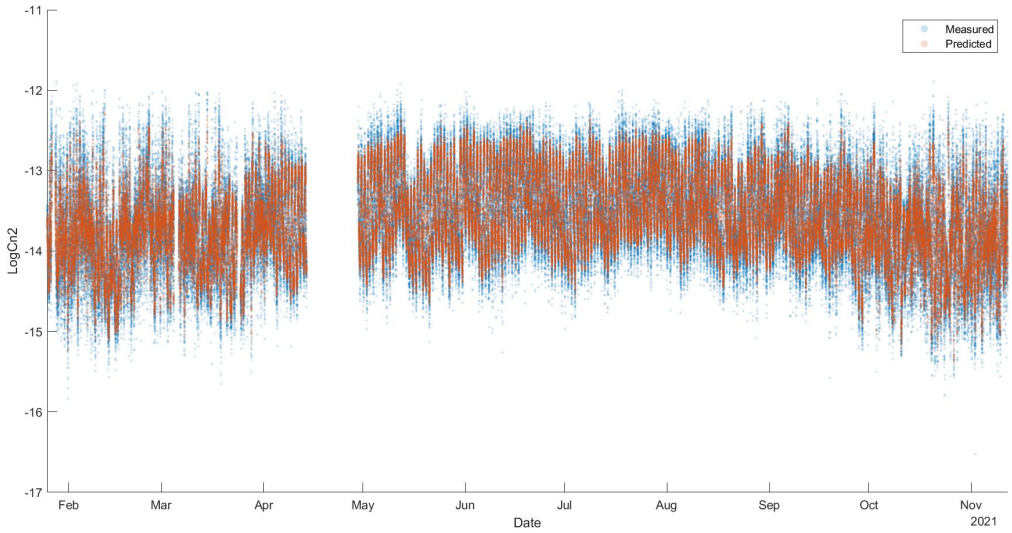


Figure 29. Predicted and measured values of $logC_n^2$ vs. time using the untrained data set for the entire ten-month period of the experiment. The RMSE is 0.2560 for this data set.

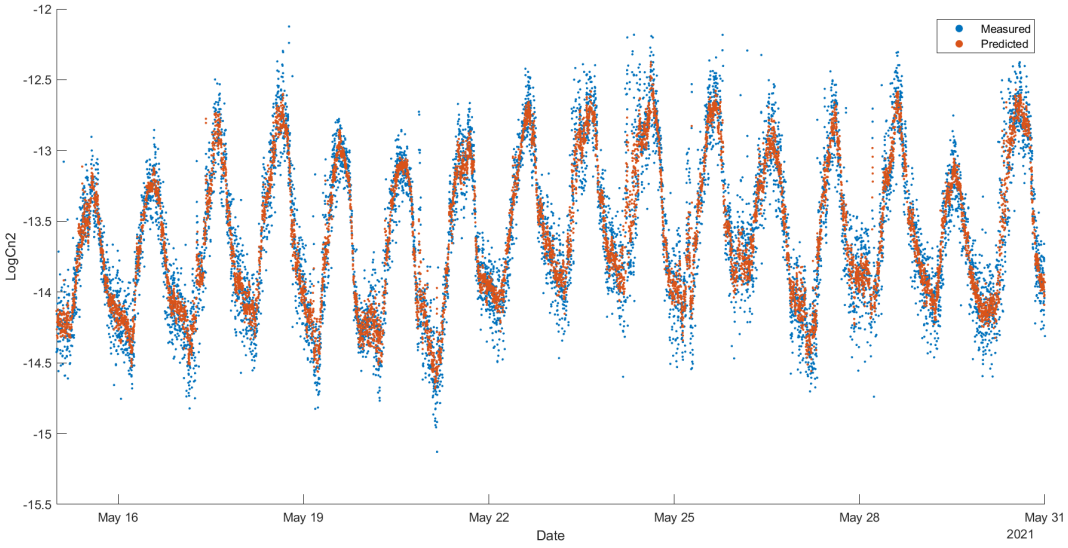


Figure 30. Predicted and measured values of $logC_n^2$ using the untrained data set from May 15 through May 31, 2021.

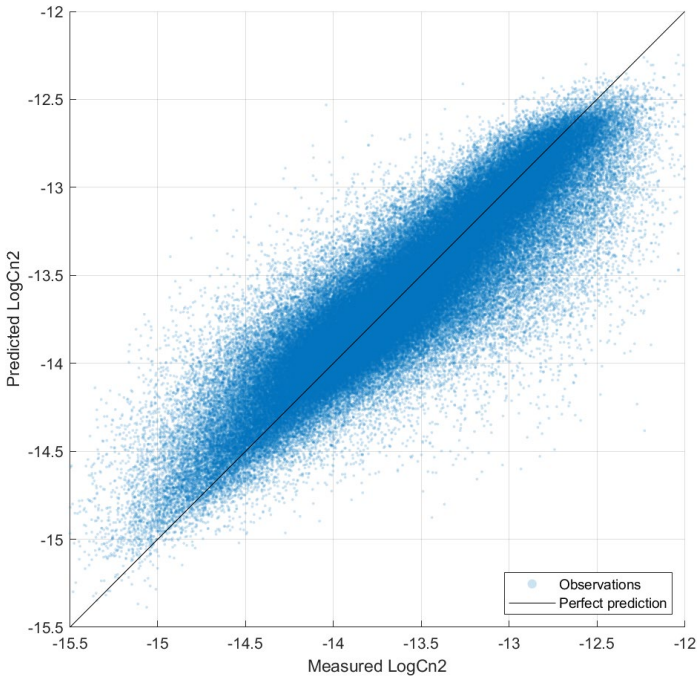


Figure 31. Predicted vs. measured values of $logCn^2$ using the untrained data set for the entire ten-month period of the experiment.

VI. CONCLUSION

A. SUMMARY

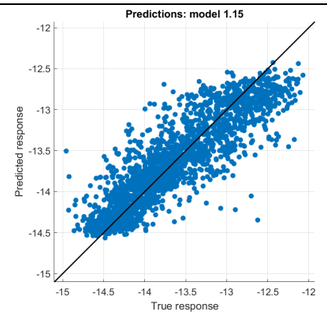
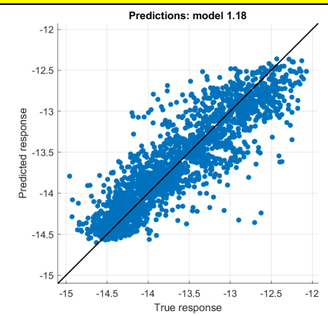
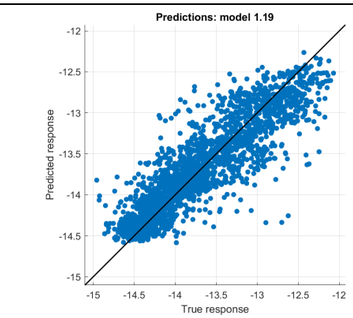
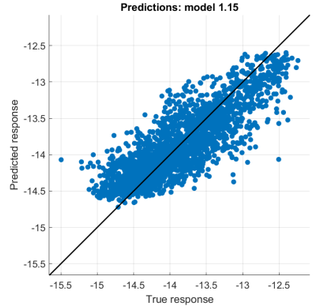
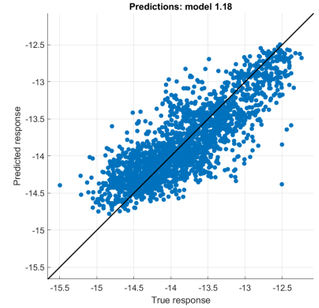
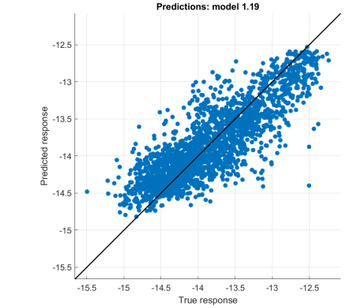
The main purpose of this thesis was to explore whether machine learning regression can be used for the prediction of optical turbulence. According to our results and analysis, we conclude that ML regression seems viable for turbulence prediction using simple atmospheric measurements from robust equipment. For our recorded data taken over a ten-month experiment at the Naval Postgraduate School (NPS) in Monterey, the ensemble of bagged trees method seemed to provide the best predictions with the lowest overall RMSE for all the ML methods that we tried. The model showed strong correlation between predicted and measured values of turbulence with high noise resistance. The results of this research could be used for better predictions of optical turbulence for modeling the performance of laser weapons and laser communication systems.

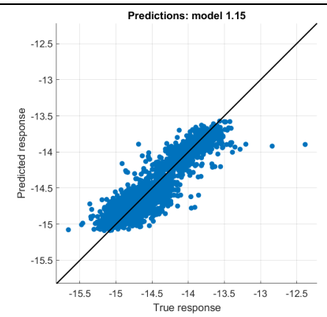
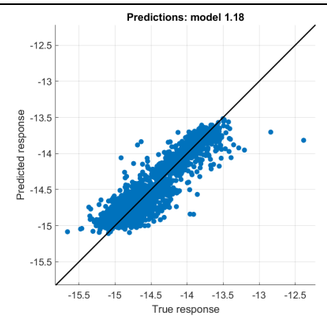
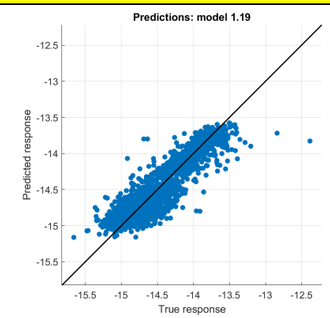
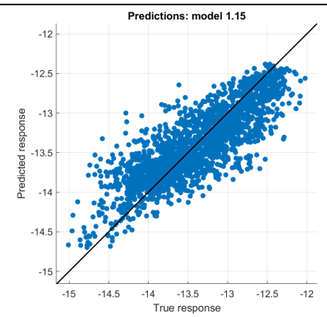
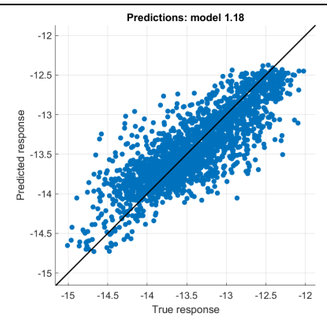
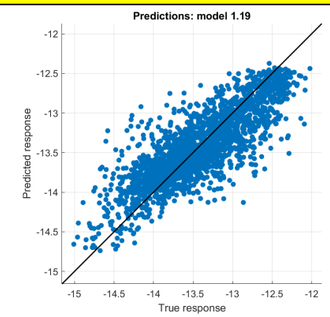
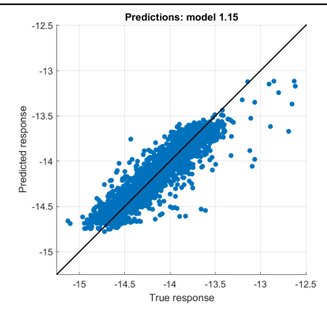
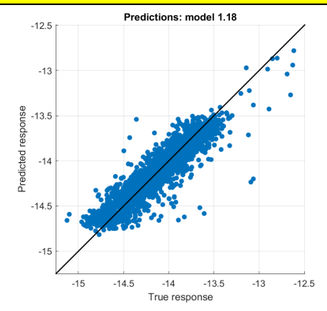
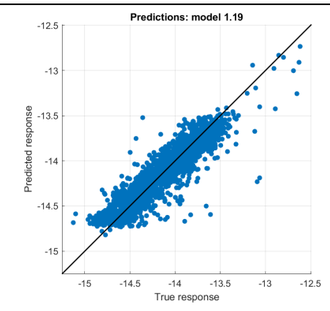
B. FUTURE WORK

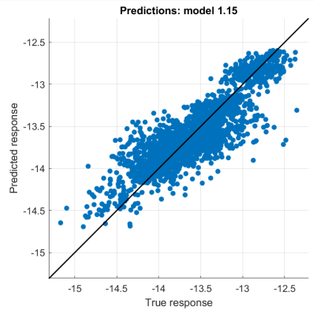
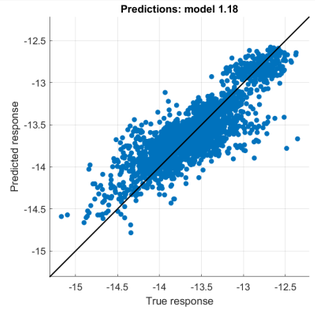
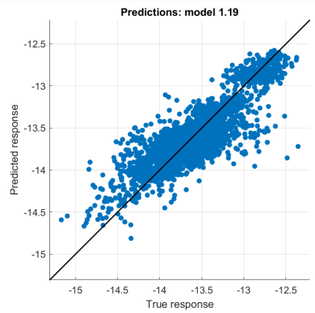
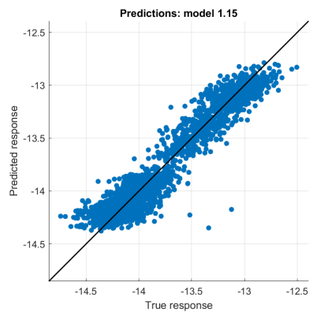
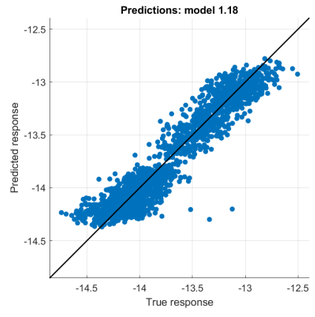
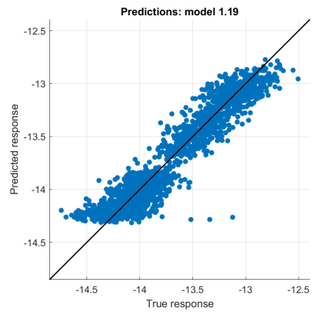
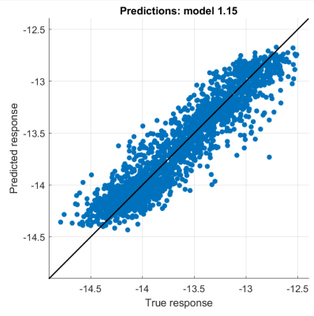
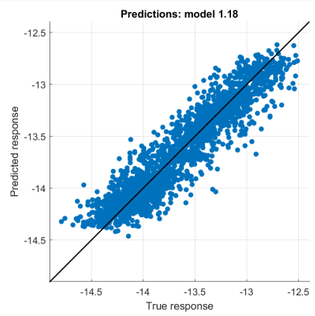
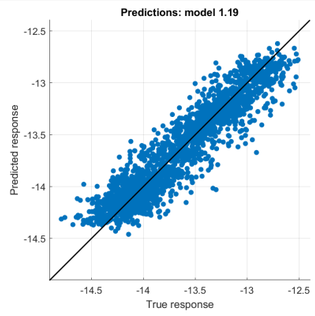
The results of this thesis highlight the need for further tests of this model at different locations, since the experimental data was collected from the roof of Spanagel Hall at NPS and the model was trained based only on this data. Additionally, further analysis is needed to determine which of the atmospheric parameters have greater influence on the turbulence predictions. Different cross-validation techniques, splitting criteria, and stopping criteria could be explored to guard against overfitting. Also, neural network regression is another type of supervised ML that can be used to examine the collected data; a colleague in the Hellenic Navy (LCDR Antonios Lionis) is looking into this approach using the data we collected. We will compare his findings to ours when they become available. Finally, the performance of ML regression methods should be compared with the corresponding predictions of NAVSLaM, which is a physics-based turbulence model that also uses simple atmospheric measurements to predict turbulence.

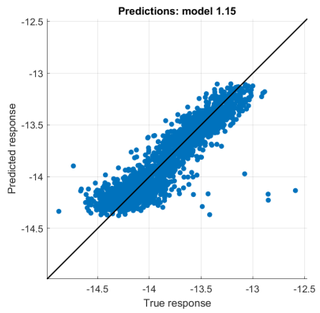
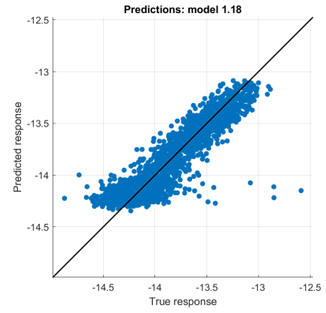
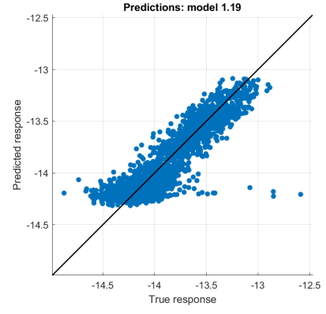
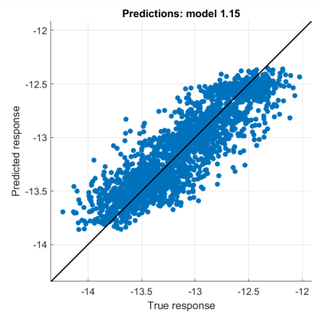
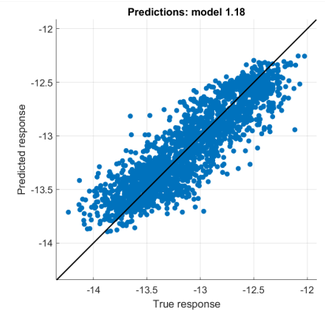
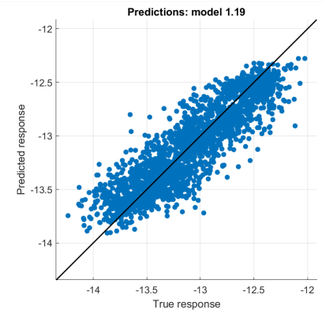
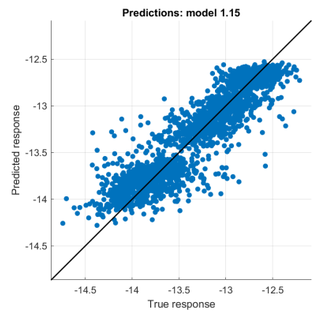
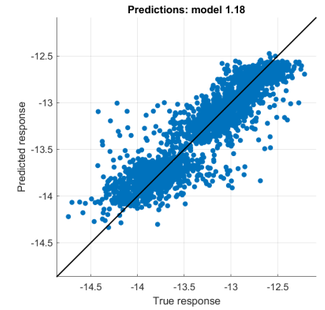
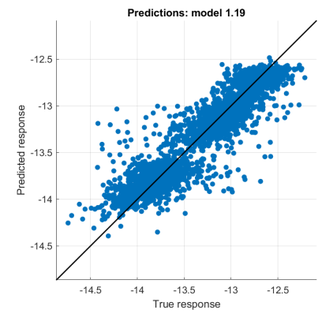
THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX. COMPARISON BETWEEN THREE REGRESSION METHODS FOR SELECTED DAYS

January 26, 2021	Ensemble Bagged Trees 0.30217	Gaussian Process Regression 0.29455 Exponential GPR	Gaussian Process Regression 0.29977 Rational Quadratic GPR
			
February 1, 2021	Ensemble Bagged Trees 0.33474	Gaussian Process Regression 0.34251 Exponential GPR	Gaussian Process Regression 0.34139 Rational Quadratic GPR
			

February 15, 2021	Bagged Tree 0.17653	Gaussian Process Regression 0.17664 Exponential GPR	Gaussian Process Regression 0.17474 Rational Quadratic GPR
			
March 1, 2021	Ensemble Bagged Trees 0.32134	Gaussian Process Regression 0.32092 Exponential GPR	Gaussian Process Regression 0.32071 Rational Quadratic GPR
			
March 15, 2021	Ensemble Bagged Trees 0.15512	Gaussian Process Regression 0.14926 Exponential GPR	Gaussian Process Regression 0.1494 Rational Quadratic GPR
			

April 1, 2021	Ensemble Bagged Trees 0.24238	Gaussian Process Regression 0.24877 Exponential GPR	Gaussian Process Regression 0.24688 Rational Quadratic GPR
			
April 13, 2021	Ensemble Bagged Trees 0.14064	Gaussian Process Regression 0.14239 Exponential GPR	Gaussian Process Regression 0.14498 Rational Quadratic GPR
			
May 1, 2021	Ensemble Bagged Trees 0.17112	Gaussian Process Regression 0.17141 Exponential GPR	Gaussian Process Regression 0.17241 Rational Quadratic GPR
			

May 15, 2021	Ensemble Bagged Trees 0.15311	Gaussian Process Regression 0.15491 Exponential GPR	Gaussian Process Regression 0.15874 Rational Quadratic GPR
			
June 1, 2021	Ensemble Bagged Trees 0.20807	Gaussian Process Regression 0.20267 Exponential GPR	Gaussian Process Regression 0.20389 Rational Quadratic GPR
			
June 15, 2021	Ensemble Bagged Trees 0.21784	Gaussian Process Regression 0.22566 Exponential GPR	Gaussian Process Regression 0.22525 Rational Quadratic GPR
			

LIST OF REFERENCES

[1] P. A. Frederickson, S. Hammel, and D. Tsintikidis, "Measurements and modeling of optical turbulence in a maritime environment," in *Atmospheric Optical Modeling, Measurement, and Simulation II*, Sep. 2006, vol. 6303, pp. 71–80. doi: 10.1117/12.683017.

[2] G. Chaskakis, "Experimental analysis and modeling of optical turbulence in the maritime environment and its potential effect on laser communications," Naval Postgraduate School, Monterey, California USA, 2020.

[3] G. M. Anderson, "Development Of A Standard Maritime Cn2 Profile Using Satellite Measurements," Air Force Institute Of Technology, Ohio USA, 2015. [Online]. Available: https://apps.dtic.mil/sti/citations/ADA615226

[4] L. C. Andrews and R. L. Phillips, *Laser Beam Propagation through Random Media*, 2nd ed. SPIE, 2005.

[5] J. Blau and K. Cohn, "Measuring Optical Turbulence in the Atmosphere," Naval Postgraduate School, Monterey CA, Jan. 12, 2020.

[6] J. Blau and K. Cohn, "Atmospheric Propagation," Naval Postgraduate School, Monterey CA, Jan. 12, 2020.

[7] S. Frederick G., "The Infrared and Electro-Optical Systems Handbook," in *Atmospheric Propagation of Radiation*, vol. 2, SPIE OPTICAL ENGINEERING PRESS, 1993, pp. 157–232.

[8] Campbell Scientific, Inc., *Three-Dimensional Sonic Anemometer*. Logan, UT USA.

[9] Campbell Scientific, Inc. *IRGASON® Integrated CO2/H2O Open-Path Gas Analyzer and 3D Sonic Anemometer*. Logan, UT USA, 2021. [Online]. Available: https://s.campbellsci.com/documents/us/manuals/irgason.pdf

[10] Apogee Instruments, Inc., *Infrared Radiometer*. 721 West 1800 North, Logan, Utah 84321, USA, 2018. [Online]. Available: http://www.jetec.com.tw/pdf/6/manual/JSI100_E_manual.pdf

[11] Campbell Scientific, Inc., *Net Radiometer SN500SS*. Logan, UT USA, 2021. [Online]. Available: https://s.campbellsci.com/documents/us/manuals/sn500ss.pdf

[12] Campbell Scientific, InC., *GPS16X-HVS GPS Receiver*. Logan, UT USA, 2020. [Online]. Available: https://s.campbellsci.com/documents/us/manuals/gps16x-hvs.pdf

[13] "Regression analysis," *Wikipedia*. Nov. 08, 2021. Accessed: Nov. 16, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Regression_analysis&oldid=1054153996

[14] G. Lawton, "Comparing Supervised vs. Unsupervised Learning," *SearchEnterpriseAI*, Mar. 22, 2021. https://searchenterpriseai.techtarget.com/feature/Comparing-supervised-vs-unsupervised-learning (accessed Oct. 13, 2021).

[15] "Bias-Variance Trade off - Machine Learning," *GeeksforGeeks*, Feb. 03, 2020. https://www.geeksforgeeks.org/ml-bias-variance-trade-off/ (accessed Oct. 13, 2021).

[16] J. Brownlee, "Overfitting and Underfitting With Machine Learning Algorithms," *Machine Learning Mastery*, Mar. 20, 2016. https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/ (accessed Oct. 13, 2021).

[17] "Bias-Variance Tradeoff." https://www.ml-science.com/bias-variance-tradeoff (accessed Oct. 13, 2021).

[18] "Fit binary decision tree for regression - MATLAB fitrtree." https://www.mathworks.com/help/stats/fitrtree.html#butl1ll_head (accessed Nov. 01, 2021).

[19] A. Yadav, "Decision Trees," *Medium*, Jan. 11, 2019. https://towardsdatascience.com/decision-trees-d07e0f420175 (accessed Oct. 26, 2021).

[20] S. Kumar, "Improving the Performance of Machine Learning Model using Bagging," Jul. 02, 2020. https://towardsdatascience.com/improving-the-performance-of-machine-learning-model-using-bagging-534cf4a076a7 (accessed Oct. 13, 2021).

[21] H. Sit, "Quick Start to Gaussian Process Regression," *towardsdatascience*, Oct. 13, 2021. https://towardsdatascience.com/quick-start-to-gaussian-process-regression-36d838810319 (accessed Oct. 13, 2021).

[22] "Gaussian process," *Wikipedia*. Accessed: Oct. 13, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Gaussian_process&oldid=1032214476

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California