Machine Learning Methods for Fitting Ellipses from Differential Atomic Clock Experiments

by

Nico Ranabhat

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

Bachelor of Science

(Engineering Physics)

at the

UNIVERSITY OF WISCONSIN–MADISON

April 2023

Abstract

The precision and accuracy of atomic clocks has increased rapidly over the past decade, leading to improvements in technologies such as gravity gradiometers and GPS. However, in some cases, the methods of extracting information from optical atomic clock comparisons are imperfect and can be improved. This work introduces two methods for fitting elliptical data obtained from atomic clock experiments: a neural network and a maximum likelihood estimate. Despite extensive training, the neural network falls short of outperforming the current least-squares method. On the other hand, when given enough data, the maximum likelihood estimate is shown to produce more accurate fits than the least-squares method. Ideas for further improving these machine learning methods are discussed.

Contents

List of Figures

Chapter 1

Introduction

Ellipse fitting is relevant in many areas of study. For one, shape-based analysis plays an important role in image processing. It is often desirable to detect shapes like polygons or ellipses in images. Ellipse detection can be useful in applications such as pupil tracking, uncrewed aerial vehicle navigation, astronomical shape segmentation, and object detection [1]. The application of particular interest to this thesis is in atomic clock comparisons. Atomic clocks play an important role in society. They are used in a wide variety of applications such as the Global Positioning System (GPS), telecommunication, and financial transaction. GPS, for example, works by synchronizing atomic clocks in satellites to clocks on the ground to determine location. The precision of the atomic clocks is one of several factors that contribute to the overall precision of GPS. With more precise time-tracking, GPS could, for example, become accurate enough to measure small strains of the Earth's crust to predict earthquakes [1]. More accurate GPS could also be used in self-driving cars to help cars determine accurate positions of objects on the road. In general, as atomic clocks improve, technologies that depend on atomic clocks tend to also improve.

One way to improve atomic clocks is by enhancing the methods of extracting information from them. Some atomic clock experiments produce data that is

elliptical in shape, and information is extracted from these clocks via ellipse fitting. However, current methods of ellipse fitting perform poorly when applied to these experiments, and there exists a need to investigate new methods [1], [2]. This thesis introduces two novel machine learning methods for fitting ellipses: a neural network and a maximum likelihood estimate. It is shown that the neural network performs slightly worse than the current least-squares method, while the maximum likelihood estimate performs better than least-squares on ellipses with a sufficient number of points. Suggestions are provided for future work to improve upon the machine learning methods. These findings provide a new avenue for developing ellipse fitting algorithms and extracting information from atomic clocks. The results indicate that atomic clocks will continue to improve, and as such pave the way for advancements in a wide range of technologies.

Chapter 2

Atomic Clocks

2.1 Applications

Before introducing the algorithms developed for this thesis, it is important to understand why atomic clocks are relevant to society. Telecommunication systems and GPS navigation rely on extremely accurate timekeeping in order to function properly. Historically, whenever an improvement in time-tracking occurs, new technologies emerge, and current technologies improve. For one, more precise time-tracking may enable GPS to become accurate enough to measure small strains of the Earth's crust to predict earthquakes [1].

The application of atomic clocks also extends beyond uses in technology. Scientists have been attracted to the notion of being able to track the dynamics of simple quantum systems down to their most elementary steps. Some theories suggest that the fundamental forces, and thus the fundamental constants, may be time-dependent [3], [4]. Comparisons of precise atomic clocks could help uncover this time-dependence [5].

Additionally, precise frequency comparisons between clocks located in different places can be used to investigate the properties of dark matter. When a

dark-matter object passes through a network of clocks, the initially synchronized clocks may become desynchronized and exhibit a particular signature that reflects the spatial structure of the dark-matter object and how it interacts with atoms [6].

Results showing time-dependence of the fundamental constants or properties of dark-matter would open the door to an abundance of new physics and new engineering. Since many physical processes are time-dependent, advancements in the precision of clocks have seen an increase in science's ability to measure the dynamics of the physical world. Experiments that can measure the time variable to a high degree of certainty are able to infer strong conclusions.

2.2 Multiplexed Optical Lattice Clock

The enhancement of atomic clocks will likely lead to new and improved technologies, and possibly new physics. To address *how* to improve atomic clocks, a brief understanding of their physical mechanisms is needed.

Recently, a Multiplexed Optical Lattice Clock has been created to observe the dynamics of two atomic clocks [7]. Each clock in the multiplexed lattice consists of an ensemble of a few thousand strontium atoms. To interrogate the clocks, each ensemble of atoms is cooled to temperatures near absolute zero and prepared in a low energy state. Then, radiation of a specific energy and frequency is simultaneously shot at two clock ensembles, driving the ${}^1S_0 \leftrightarrow {}^3P_0$ frequency transition. Next, the number of atoms that have transitioned to the excited state is measured for each ensemble [7]. This same experimental procedure is conducted many times. By parametrically plotting the fraction of atoms in the excited state over the total number of atoms for each ensemble, one obtains a noisy ellipse, as shown in **Figure 2.1**.

Ellipse fitting can be used to extract the shift in periodicity between two

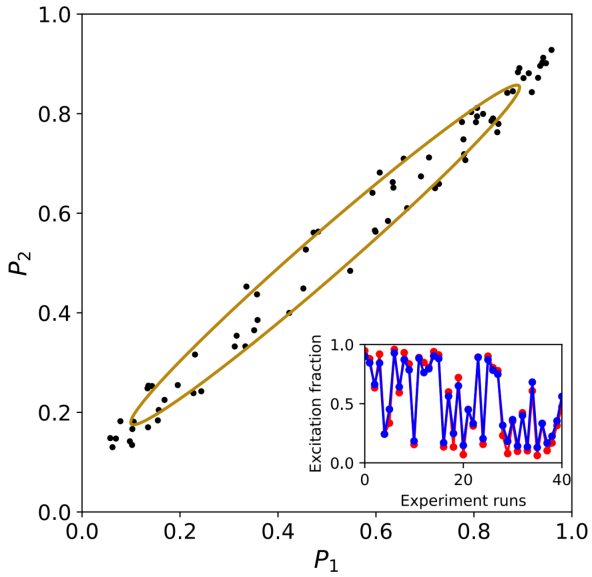

Figure 2.1: Data from the Multiplexed Optical Lattice Clock. P_1 and P_2 represent the excitation fraction of population 1 and 2, respectively. An excitation fraction of 1.0 indicates that all the atoms in an ensemble are found in the excited state. An excitation fraction of zero indicates all atoms are in the ground state. The excitation fractions are plotted parametrically and in time (inset) [7].

clocks ensembles. This phase shift is useful for diagnosing causes of phase difference due to several phenomena, including magnetic and electric fields, temperature gradients, and gravitational redshifts. The multiplexed clock is intentionally oriented in a vertical direction to quantify shifts in frequencies of the atom populations specifically due to gravitational time dilation. The device is able to measure the gravitational redshift due to general relativity at the sub-centimeter scale [7]. These capabilities are made possible mainly through analysis of the phase difference between ensembles. Improving methods of fitting atomic clock data and extracting differential phase thus directly impacts the precision to which these clocks can measure environmental conditions.

Chapter 3

Ellipse Fitting

The task of extracting the differential phase between two clocks boils down to fitting elliptical data (see **Figure 2.1**). This chapter details the form in which ellipses arise from clock data. In addition, the current least-squares method used for fitting clock data is described along with the method's shortcomings.

3.1 Phase Extraction

As described in [8], the normalized noise-free excitation fractions of two populations of atoms in two clock ensembles can be expressed as

$$\begin{aligned} x &= c_x \cos(\phi_c + \phi_d) + b_x \\ y &= c_y \cos(\phi_c - \phi_d) + b_y \end{aligned} \tag{3.1}$$

where x and y are the excitation fractions of each population, c_x and c_y are the normalized fringe contrasts, b_x and b_y are offsets, ϕ_c is the common mode phase, and ϕ_d is the differential phase. During a single clock comparison, the contrasts, offsets, and differential phase remain relatively constant, whereas the common mode phase changes from shot-to-shot. A parametric plot of equation 3.1 across variable

ϕ_c results in an ellipse.

To fit data of this form to an ellipse, we first need to rewrite (x, y) in the conic form:

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0 \tag{3.2}$$

This can be done by re-scaling (x, y) as

$$\begin{aligned} x' &= (x - b_x)/c_x = \cos\phi_c \cos\phi_d - \sin\phi_c \sin\phi_d \\ y' &= (y - b_y)/c_y = \cos\phi_c \cos\phi_d + \sin\phi_c \sin\phi_d \end{aligned} \tag{3.3}$$

Then, by the Pythagorean trig identity, we see that

$$\left(\frac{x' - y'}{2\sin\phi_c}\right)^2 + \left(\frac{x' + y'}{2\cos\phi_c}\right)^2 - 1 = 0 \tag{3.4}$$

Finally, we can expand equation 3.4 using the relationships in equation 3.3 to find

$$\begin{aligned} &\frac{1}{c_x^2}x^2 - \frac{2\cos 2\phi_d}{c_x c_y}xy + \frac{1}{c_y^2}y^2 + \left(\frac{2b_y \cos 2\phi_d}{c_x c_y} - \frac{2b_x}{c_x^2}\right)x + \left(\frac{2b_x \cos 2\phi_d}{c_x c_y} - \frac{2b_y}{c_y^2}\right)y \\ &+ \left(\frac{b_x^2}{c_x^2} + \frac{b_y^2}{c_y^2} - \frac{2b_x b_y \cos 2\phi_d}{c_x c_y} - 4\cos^2\phi_d \sin^2\phi_d\right) = 0 \end{aligned} \tag{3.5}$$

Note that the coefficients in equation 3.5 are the same coefficients as in equation 3.2. Then, a quick substitution shows that the differential phase between the two clock ensembles can be calculated using equation 3.6:

$$\boxed{\boldsymbol{\phi_d} = \frac{1}{2}\cos^{-1}\left(\frac{-B}{2\sqrt{AC}}\right)} \tag{3.6}$$

where A, B, and C are coefficients of the fitted ellipse [8].

3.2 Least-Squares Algorithm

Currently, the least-squares optimization method is used to fit data to find coefficients A through F in equation 3.2 and extract the differential phase in atomic clock comparisons. The basic idea behind the method is outlined in [9], and a short description is provided here.

As outlined in the previous section, an ellipse is a special case of a conic section which can be described by a second order polynomial

$$F(x, y) = Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0 \tag{3.7}$$

with the ellipse-specific constraint

$$B^2 - 4AC < 0 \tag{3.8}$$

where A, B, C, D, E, F are coefficients of the polynomial and (x, y) are the coordinates of points that lie on the ellipse. The function $F(x, y)$ is the *algebraic distance* from the point (x, y) to the conic. By introducing vectors

$$\mathbf{a} = [A, B, C, D, E, F]^T$$

$$\mathbf{x} = [x^2, xy, y^2, x, y, 1]$$

the algebraic distance can be re-written as

$$F_{\mathbf{a}}(\mathbf{x}) = \mathbf{x} \cdot \mathbf{a} = 0 \tag{3.9}$$

Then, to find the conic with coefficients $\mathbf{a}$ that minimizes the sum of the squared algebraic distances from the points (x_i, y_i), $i = 1, ..., N$ to the conic, one can solve

the optimization problem:

$$\begin{aligned}\min_{\mathbf{a}} \sum_{i=1}^{N} F\left(x_i, y_i\right)^2 &= \min_{\mathbf{a}} \sum_{i=1}^{N} \left(F_{\mathbf{a}}\left(\mathbf{x_i}\right)\right)^2 \\ &= \min_{\mathbf{a}} \sum_{i=1}^{N} \left(\mathbf{x_i} \cdot \mathbf{a}\right)^2\end{aligned} \tag{3.10}$$

To ensure an ellipse-specific solution to equation 3.10, the constraint in equation 3.8 must be satisfied. Since conics with coefficients $\mathbf{a}$ and coefficients $\alpha \cdot \mathbf{a}$, $\alpha \neq 0$ represent the same conic, the constraint in equation 3.8 can be re-written as $B^2 - 4AC = 1$ with proper scaling.

Then, the ellipse-specific fitting problem can be reformulated as

$$\min_{\mathbf{a}} \|\mathbf{Da}\|^2 \quad \text{subject to} \quad \mathbf{a^T Ca} = 1 \tag{3.11}$$

where $\mathbf{D}$ is size N $\times$ 6 and the constraint matrix $\mathbf{C}$ is size 6×6.

$$\mathbf{D} = \begin{pmatrix} x_1^2 & x_1 y_1 & y_1^2 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_i^2 & x_i y_i & y_i^2 & x_i & y_i & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N^2 & x_N y_N & y_N^2 & x_N & y_N & 1 \end{pmatrix} \qquad \mathbf{C} = \begin{pmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The approach described in [10] uses Lagrangian multipliers to solve this optimization problem. One drawback, however, is that the matrix C is singular and thus the computation of the eigenvalues can be numerically unstable and produce wrong results [9].

The ellipse fitting method currently used in the multiplexed optical lattice

clock is the least-squares method described in [9], henceforth referred to as simply the 'least-squares' (LS) method. This method overcomes some drawbacks from the approach in [10] by splitting matrices D and C into four quadrants and solving separate equations corresponding to the quadratic and linear terms of matrix D. Although this method is more numerically stable, it too possesses some shortcomings such as bias.

3.3 Least-Squares Bias

When noise is present in data, the LS algorithm is bias towards smaller ellipses. This bias is demonstrated in **Figure 3.1**. An analysis was conducted where the LS algorithm was used to fit 12 sets of 20 points which represent different elliptical arcs of the same ellipse. The noise-to-signal ratio (SNR) was varied from 0.01 to 0.5, and it was found that as noise increases, the elliptical fits shrink in size and become less accurate [9].

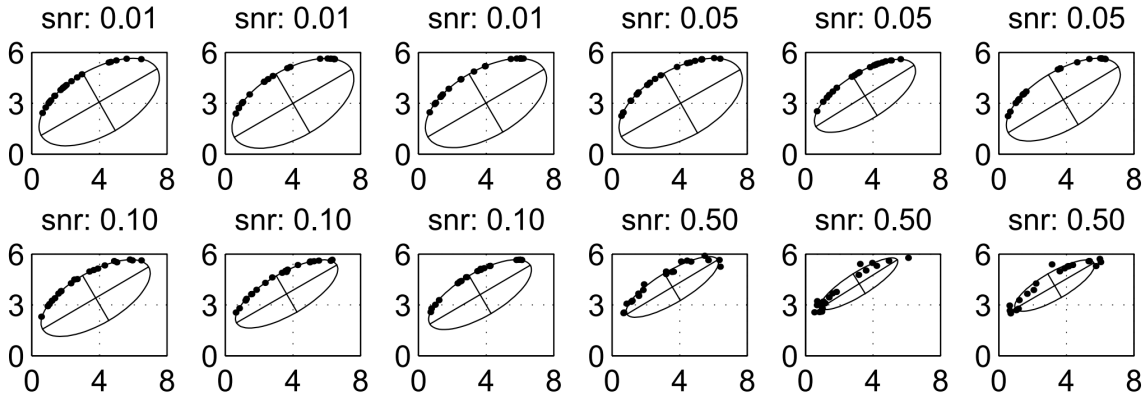

Figure 3.1: Least-squares algorithm applied to data with noise. 12 datasets describing different elliptical arcs of the ellipse centered at (4,3) with semiaxes (4,2) and tilt 30 degrees are plotted. As the noise-to-signal ratio (SNR) increases, the LS method produces smaller and less accurate fits [9].

In addition, the LS method is bias with respect to differential phase and produces inaccurate fits when ϕ_d is close to 0 or $\pi/2$ [11]. These ellipses are long

and skinny and have eccentricity close to unity. Conversely, the method performs well when the phase is close to $\pi/4$ and the ellipses are more circular in shape. This relationship is shown in **Figure 3.2**.

To create this plot, Monte-Carlo simulation were used to generate 500 ellipses with differential phase values evenly spaced between 0 and $\pi/2$. The contrast for each ellipse was 0.65, and each atom population had a constant 1000 atoms. The number of points for each ellipse was randomly selected between 5 and 500. More details about these simulations, such as the noise model for atomic clock comparisons, can be found in Chapter 4.2.1. The LS method was used to fit all 500 ellipses and extract the differential phase values. This whole process was repeated 100 times and averaged. The Phase Error $\epsilon = \bar{\phi}_{d,LS} - \phi_{d,true}$ is plotted against the true phase, where $\bar{\phi}_{d,LS}$ is the averaged LS differential phase estimate and $\phi_{d,true}$ is the true differential phase.

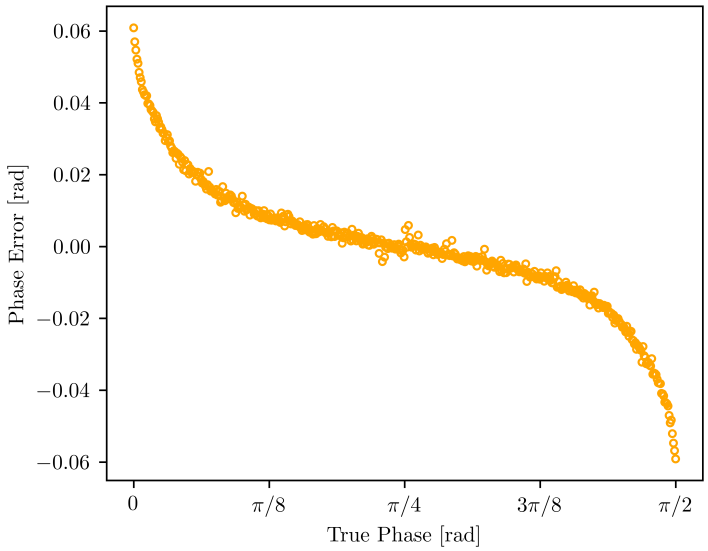

Figure 3.2: Least-Squares Phase Bias. 100 batches of 500 ellipses with differential phase values evenly spaced between 0 and $\pi/2$ were created via Monte-Carlo simulation. The LS method was used to extract estimates of each differential phase, and the estimates were averaged over all 100 runs. At ϕ_d near 0, LS overestimates ϕ_d, while at ϕ_d near $\pi/2$, LS underestimates ϕ_d.

It is clear that at true phase near 0, the least-squares method overestimates the true phase, while at ϕ_d near $\pi/2$, it underestimates ϕ_d. Luckily, the nature of this bias is fairly structured and can be compensated for via bias correction.

3.4 Least-Squares Bias Correction

To correct for the LS bias, we first fit the curve in **Figure 3.2**. The curve follows the general shape of the tangent function, so we fit to a function of the form

$$\epsilon = a_1 \tan[a_2(\phi_{d,true} + a_3)] + a_4 \tag{3.12}$$

where ϵ is the residual, $\phi_{d,true}$ is the true differential phase value, and a_1 through a_4 are the coefficients of the fit. We find the following values for the coefficients:

Parameter	Value
a_1	-0.01053
a_2	1.77000
a_3	-0.78614
a_4	-0.00001

Table 3.1: Table of Least-Squares phase bias fit values a_1-a_4 from equation 3.12.

The resulting fit is shown in **Figure 3.3**. Note that by symmetry, $a_4 = 0$ and $a_3 = -\pi/4$. By inspection, $a_2 \approx \sqrt{\pi}$. It is likely the case that $a_2 = \sqrt{\pi}$ as the number of simulations approaches infinity, but moving forward the values in **Table 3.1** are used. The high accuracy of the fit indicates that the function could be analytically derived, but we leave this derivation for future work.

This regression fit can be used to describe the bias in the LS algorithm and subsequently be used to correct for bias when fitting with the LS algorithm. While the bias-corrected LS algorithm is relatively accurate, an even more accurate method for fitting elliptical clock data is desirable. The next section outlines a neural net-

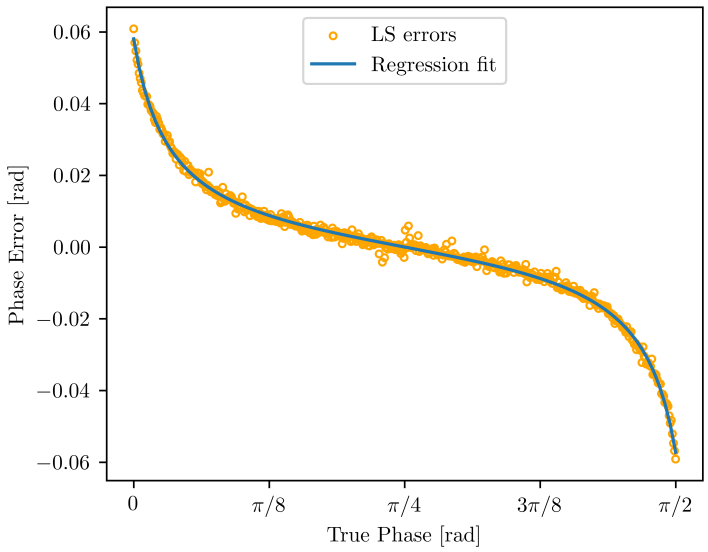

Figure 3.3: Fit of the Least-Squares Phase Bias. The Least-Squares phase bias shown in Figure 3.2 is fit to a funciton of the form $a_1 \tan[a_2(\phi_{d,true} + a_3)] + a_4$.

work algorithm designed specifically for this task.

Chapter 4

Neural Network

Neural networks have become a powerful tool in the field of machine learning, enabling us to solve complex problems that were once thought to be beyond the capabilities of computers. At their core, neural networks are inspired by the structure and function of the human brain, using layers of interconnected nodes to learn from data and make predictions. This ability to learn from data makes neural networks a highly effective tool for tasks such as pattern recognition, classification, and regression.

A novel neural network used to fit ellipses in the multiplexed clock and other atomic clocks could produce more accurate fits than the current LS method [7]. Machine learning is appealing because it has been implemented for ellipse fitting outside the application of atomic clocks and has outperformed numerical methods in these applications [12], [13]. Furthermore, a neural network algorithm is promising because the noise present in physical clock comparisons can be accurately modeled and a vast training data set that closely resembles real experiments can be simulated to train the new algorithm.

In this chapter, we review previous work that highlights the success of neural networks in fitting elliptical data. We then introduce a novel neural network

specifically engineered to extract differential phase from atomic clock comparisons.

4.1 Past Work

Machine learning is used in many fields and is a popular technique for fitting elliptical data. It is widely used in computer vision—convolutional neural networks have been developed to infer clustered and heavily occluded elliptical objects from images. Examples include fruits hanging from trees shown in **Figure 4.1a** [13], [14]. Another real-world application involves an ellipse fitting Lagrange Programming Neural Network (LPNN) that was developed to fit ellipses with different types of noise, shown in **Figure 4.1b** [12].

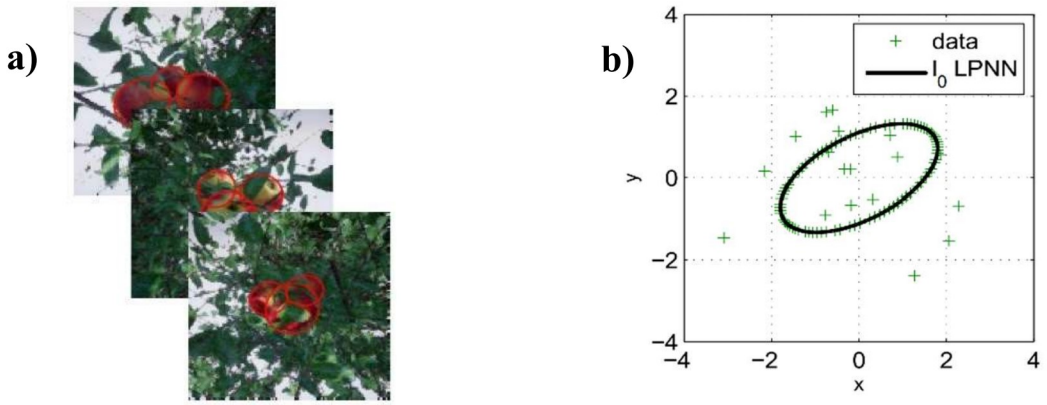

Figure 4.1: Real-world applications of machine learning for fitting ellipses. a) Ellipses being fit to occluded fruit in trees. b) Lagrange programming neural network (LPNN) fitting to noisy data. In both applications, the machine learning method outperformed non-machine learning methods [12], [13].

The neural network used in **Figure 4.1b** was also used for edge detection of various images, including those of the human eye, space probes, and plankton. This particular neural network has demonstrated superior performance in fitting elliptical data when compared to multiple state-of-the-art algorithms [12]. In addition, the approach outlined in **Figure 4.1a** has also outperformed other non-machine learning methods. Although the LPNN and other machine learning techniques used in

computer vision are impressive, their relevance to atomic clock data is limited. Nevertheless, given the success of machine learning methods in the applications shown above, it is reasonable to assume that a novel neural network could perform well in fitting atomic clock data.

4.2 Neural Network

Here, we introduce the novel Neural Network (NN) that is engineered to fit atomic clock data. The first section details how the noise in differential clock comparisons can be modeled. Arguably the most important quality of a neural network is the data it is trained on, so it is important that our noise model accurately mimics the physical dynamics. The next section highlights how the training data is simulated. Then, the motivation for the structure of the network is detailed along with a procedure followed for training.

4.2.1 Modeling Quantum Projection Noise

As outlined in chapter 3.1, the normalized noise-free excitation fractions of two populations of atoms in two clock ensembles can be expressed as

$$\begin{aligned} x &= c_x \cos\left(\phi_c + \phi_d\right) + b_x \\ y &= c_y \cos\left(\phi_c - \phi_d\right) + b_y \end{aligned} \tag{4.1}$$

where x and y are the excitation fractions of each population, c_x and c_y are the normalized fringe contrasts, b_x and b_y are offsets, ϕ_c is the common mode phase, and ϕ_d is the differential phase. Note that equations 4.1 do not account for noise.

The main source of noise in clock experiments is Quantum Projection Noise (QPN). This noise arises when quantifying the excitation fraction of each clock ensemble. During a clock interrogation in the multiplexed clock [7], all atoms in an

ensemble are prepared in the $|^1S_0\rangle$ (ground) state. Then, lasers are shot at the atom populations to excite some of the atoms to the $|^3P_0\rangle$ (excited) state. The energy level of each atom must collapse to one of these two states. The excitation fraction is simply the fraction of atoms measured in the excited state over the total number of atoms. Since only a finite number of atoms can be probed, there will always be some uncertainty in the true superposition of the quantum mechanical state. This uncertainty in the measured excitation fraction is called quantum projection noise.

To model QPN, consider an ensemble of N atoms with a probability p_a to be in state a ($|^3P_0\rangle$ for example). Then, the likelihood of measuring N_a atoms in that state is given by the binomial distribution [15]

$$\mathcal{P}\left(N_a, N, p_a\right) = \frac{N}{N_a!\left(N - N_a\right)!} p_a^{N_a} \left(1 - p_a\right)^{N - N_a} \tag{4.2}$$

Now, consider simulating ellipse data using equations 4.1. Point (x_i, y_i) will have some QPN associated with each coordinate. To account for QPN in coordinate x_i, x_i can be updated by sampling from a binomial distribution $X \sim \mathrm{Bin}(N, x_i)$ where N is the number of atoms in each ensemble (trials) and x_i is the excitation fraction (probability of success). This will return the number of successes over the N trials. After normalizing over N, we obtain an updated x_i that accounts for QPN. This process is repeated for coordinate y_i and the rest of the points on the ellipse.

4.2.2 Creating Training Data with Monte-Carlo Simulations

After modeling QPN, the next step is to generate training and testing data via Monte-Carlo simulation. However, the complexity of the data needs to be carefully chosen. Each simulated ellipse can be described by nine variables: contrasts and offsets for each axis (c_x, c_y, b_x, b_y), differential phase (ϕ_d), common phase (ϕ_c), number of atoms in each ensemble (N_x, N_y), and the number of points. It is worth

noting that due to the latency between measurements in the lab, the contrasts and the number of atoms in each ensemble can also vary from point-to-point [7].

To simplify the data and make it easier for the neural network identify patterns, the degrees of freedom in the data are reduced by making certain assumptions for each ellipse. For example, all offsets are set to $b_x = b_y = 0.5$, and differential phase is held constant across points. Additionally, it is assumed that contrasts and the number of atoms in each ensemble are constant from point-to-point ($c_x = c_y, N_x = N_y$). In lab, ϕ_c is essentially randomly varied for each point; thus, it is also varied randomly in simulation.

As a result of these assumptions, the degrees of freedom of the data set is reduced from nine to three for each ellipse: ϕ_d, contrasts, and number of points are the only varying parameters. This simplification not only makes it easier for the neural network to identify patterns, but also helps reduce overfitting and improve the generalization of the model.

As detailed in Chapter 6.1, the data can be further simplified by excluding QPN, variable contrast, and variable number of points, leaving ϕ_d as the only changing parameter. This primitive data set serves as an initial training set before progressing to more complex data sets that incorporate QPN, variable contrast, and variable number of points. By starting with this simpler data set, the neural network is able to learn the underlying physics and patterns of the system before dealing with the more complex data sets.

Overall, the process of generating and simplifying the simulated ellipse data is crucial for training an accurate and reliable neural network. By reducing the degrees of freedom and carefully choosing the complexity of the data, we can improve the generalization of the model and reduce overfitting, leading to better predictive power and a deeper understanding of the underlying physics.

4.2.3 Structure and Hyperparameters

The architecture and hyperparameters of the neural network are arguably more important than the quality of the training data. Neural networks are a class of machine learning models inspired by the structure and function of the human brain. They are composed of interconnected "neurons" that receive input signals, apply a nonlinear function called an "activation function", and produce an output signal. These neurons are organized into layers, with the first layer receiving input data and subsequent "hidden" layers transforming the data to produce an output prediction.

The neural network used in this study has a specific structure consisting of 1000 input neurons, with the capacity to process up to 500 ellipse points. The input layer is designed in such a way that the first neuron represents the x-coordinate of the first ellipse point, while the 501^{st} neuron represents the y-coordinate. The input layer is filled sequentially in this way until all the points on the ellipse are accounted for. The remaining neurons in the input layer are set to 0 and have no impact on the output. This technique is known as "zero padding," and it allows for the efficient processing of varying-length input sequences. The output layer contains only one neuron that represents the differential clock phase, ϕ_d.

During training, the network weights are initialized to random values. The value of each neuron in the first hidden layer is obtained by taking a weighted linear combination of each neuron from the input layer and applying the Rectified Linear Unit (ReLU) activation function, which enables the network to capture nonlinear relationships between input and output. The values for the remaining neurons in the network, including the output, are calculated in this same way. The difference between the predicted output and the true output is then calculated using a loss function. The main loss function used in this work is the Mean Squared Error (MSE)

described by equation 4.3:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\phi_{d_i} - \hat{\phi}_{d_i})^2 \tag{4.3}$$

where ϕ_{d_i} is the true differential phase of the i'th ellipse and $\hat{\phi}_{d_i}$ is the estimated differential phase.

The goal during training is to adjust the weights to minimize the MSE loss function and produce an algorithm that can predict phase close to the true value. This is done via, backpropagation, an algorithm that computes the gradient of the loss function with respect to each weight and updates the weights to minimize the loss function[16].

In designing the NN, we also need to choose hyperparameters that determine the optimization process. A summary of some of the important hyperparameters and their functionalities is provided below.

- **Epochs**: The number of times the entire training dataset is passed through the network during training.
- **Batch size**: The number of training examples processed in each forward and backward pass before updating the weights during training.
- **Optimizer**: The algorithm used to update the network weights based on the gradients computed during backpropagation. The two main optimizers used in this work are Stochastic Gradient Descent (SGD) and Adam.
- **Scheduler**: The technique used to adjust the learning rate during training to improve convergence and prevent overfitting.
- **Second layer size**: The number of neurons in the second (hidden) layer of the network.

- **Starting learning rate**: The initial value of the learning rate used during training, which determines the step size for weight updates.
- **Dropout**: A regularization technique used to prevent overfitting by randomly dropping out (setting to zero) some fraction of the neurons in a layer during training.

In summary, the architecture and hyperparameters of a neural network are paramount in producing a reliable and precise algorithm. However, selecting the right combination of hyperparameters can be a daunting task as there is no established protocol for doing so. To address this challenge, we have leveraged the capabilities of Weights and Biases (W&B), a popular platform for hyperparameter optimization.

4.2.4 Hyperparameter Sweeps

The platform Weights and Biases (W&B) is used to sweep over ranges of hyperparameters to systematically explore the space of possible architectures and hyperparameters, and to identify those that produce the best results. The platform also provides helpful information about system metrics such as GPU percent utilization, power usage, temperature, percent memory allocation, and more. Knowledge of these metrics helps to identify any computing inefficiencies that occur during training. For this work, W&B is used to track training and testing loss along with several hyperparameter values.

Figure 4.2 shows a 'sweep'; a visual representation of 116 neural networks trained with different hyperparameters and their resulting loss. Each curve represents a single network with a unique set of hyperparameters. The color of the curve corresponds to the loss. In total, 126 sweeps were conducted to train a total of 2489 different neural networks.

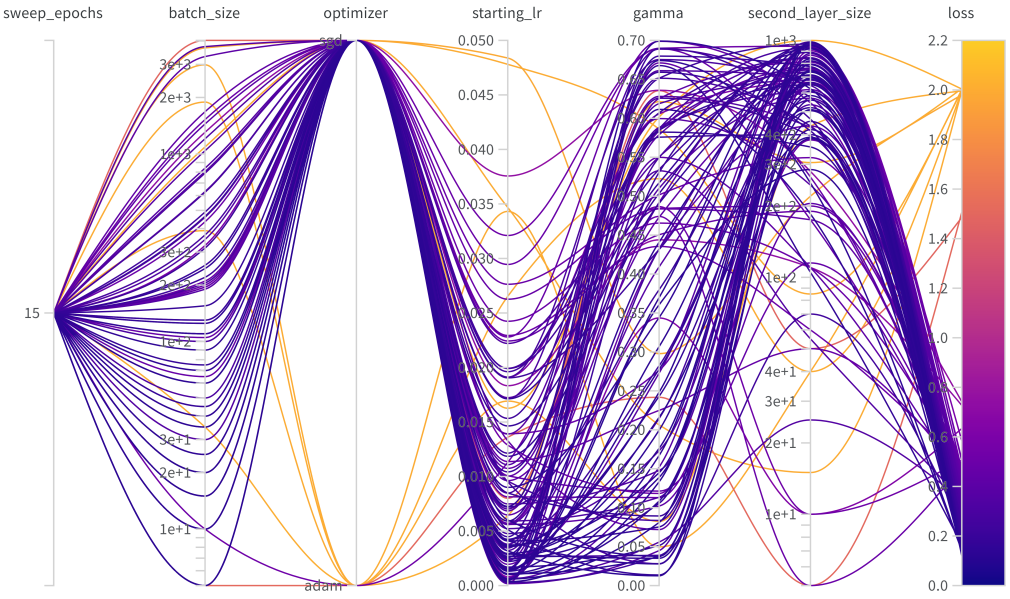

Figure 4.2: Sweep of hyperparameters. 116 different neural networks are trained to find optimal hyperparameter values. Each curve represents a neural network with a unique set of hyperparamters. The networks are trained and loss is computed and shown as a heatmap on the right of the figure.

Most of the sweeps were conducted using the W&B Bayesian sweep, which works by creating a probabilistic model of the loss function and suggesting new sets of hyperparameters to evaluate, based on the expected improvement of the loss function. The sweep balances exploration (trying out new, potentially better hyperparameters) with exploitation (using the information gained from previous evaluations to suggest the most promising hyperparameters).

The performance of the NN is compared to that of the LS algorithm in chapter 6.1.

Chapter 5

Maximum Likelihood Estimate

In addition to the neural network detailed in Chapter 4, a maximum likelihood estimate (MLE) algorithm was developed to fit data from clock comparisons. This method was primarily developed by Matt Cambria to estimate differential phase, and extend by myself (Nico Ranabhat) to estimate clock contrasts. MLE algorithms have been used in the past to successfully fit ellipse data from differential clock comparisons [17] and are widely used in various fields such as economics, engineering, and biology.

5.1 Theory

MLE is a method used to estimate the parameters of a statistical model based on observed data, by maximizing the likelihood function. The likelihood function is defined as the probability of the observed data, given the model parameters. By maximizing this function, we obtain the parameter values that are most likely to have generated the observed data.

Mathematically, the MLE is obtained by solving the following optimization problem:

$$\hat{\theta}_{\text{MLE}} = \underset{\theta}{\text{argmax}}, \mathcal{L}(\theta; x_1, x_2, ..., x_n) \tag{5.1}$$

where $\hat{\theta}_{\text{MLE}}$ is the maximum likelihood estimate of the parameter θ, $\mathcal{L}(\theta; x_1, x_2, ..., x_n)$ is the likelihood function, and $x_1, x_2, ..., x_n$ are the observed data [18].

In order to obtain the MLE, we need to find the value of θ that maximizes the likelihood function. When the likelihood function is explicitly known, this can be achieved by taking the derivative of the likelihood function with respect to θ, setting it equal to zero, and solving for θ. The resulting value of θ is the maximum likelihood estimate.

For atomic clock data, we define θ to consist of two parameters: differential phase (ϕ_d) and contrast (c). That is, we want to find the values for ϕ_d and c that best describe the given points of an ellipse. Note that this MLE algorithm could be further extended to include N, the number of atoms in each ensemble. The algorithm iterates through these steps:

1. Estimate initial values for ϕ_d and c based on the correlation and range of the data.

2. Construct an ellipse using the initial guess values for ϕ_d and c, and generate 1000 points evenly spaced on this ellipse.

3. Generate bivariate Gaussian distributions centered at each of these 1000 points to approximate the bivariate binomial distribution produced by quantum projection noise (see Chapter 4.2.1).

4. Evaluate the probability density for each experimental data point with respect to all of the generated Gaussian distributions.

5. Sum all of the probability densities to determine the likelihood of observing the data given the initial parameter guess.

6. Adjust the parameters (using the scipy.optimize package in Python) until the likelihood function is maximized.

In step 3, the bivarite Gaussian distribution is used to approximate the bivariate binomial distribution produced by QPN. This conversion from discrete to continuous distributions is a necessary step for the scipy optimization package. This approximation is valid when $Np > 10$ and $N(p-1) > 10$ where N is the number of atoms in each ensemble and p is the excitation fraction [18]. In our analysis, we use 1000 atoms in each ensemble and only consider excitation fractions between $0.175 < p < 0.825$, so the approximation is valid.

5.2 Bias

The same procedure outlined in Chapter 3.4 is used to classify the bias of the MLE algorithm. To recap, Monte-Carlo simulation is used to generate 500 ellipses, each with differential phase values evenly spaced between 0 and $\pi/2$. The contrast and number of atoms in each ensemble for each ellipse is set to 0.65 and 1000, respectively. The number of points varies randomly between 5 and 500. Then, the MLE and LS algorithms are used to fit each ellipse to obtain a differential phase estimate. This whole process is repeated 100 times and averaged.

Figure 5.1 shows the errors of the phase estimate plotted against ϕ_d for the LS and MLE algorithms. Unlike the LS errors, the MLE errors are seemingly independent of ϕ_d, indicating that the MLE algorithm is not bias for this particular simulated set. Thus, we do not apply a bias correction to the algorithm.

The bias of MLE could be further investigated by considering data at phase specifically near 0 or $\pi/2$, or data with larger amounts of noise. However, we omit this analysis in this work.

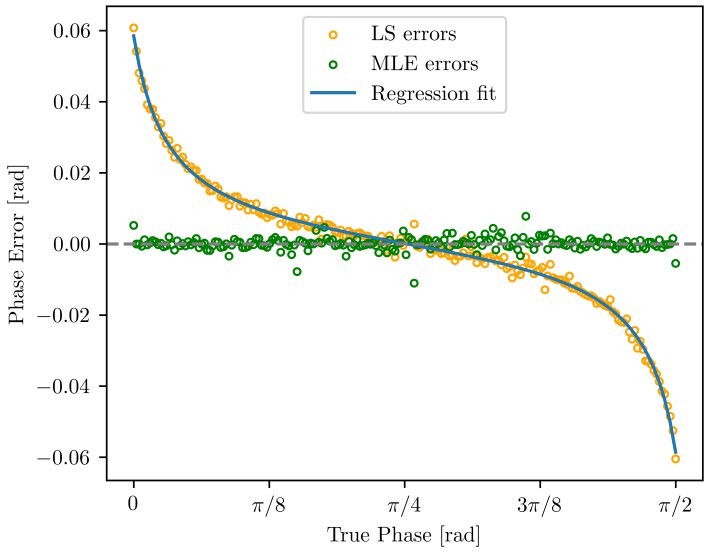

Figure 5.1: MLE Phase Bias. 100 batches of 500 ellipses with differential phase values evenly spaced between 0 and $\pi/2$ were created via Monte-Carlo simulation. The MLE method was used to extract estimates of each differential phase, and the estimates were averaged over all 100 runs. Unlike LS, the MLE errors are independent of true phase for this simulated set. A regression fit of the LS errors is included along with a dashed line through the origin for reference.

Chapter 6

Results and Discussion

The performance of the neural network and maximum likelihood estimate is compared against the least squares algorithm. Each algorithm is used to fit various ellipses that are simulated to mimic differential atomic clock experiments. In this analysis, we use 'loss' to quantify the closeness of an estimate to the true value. The loss function used is the mean squared error (MSE):

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\phi_{d_i} - \hat{\phi}_{d_i})^2 \tag{6.1}$$

where N is the number of ellipses being evaluated (typically 100), ϕ_d is the true differential phase and $\hat{\phi}_d$ is the estimated differential phase.

6.1 Neural Network

Initially, the neural network was trained on a simple dataset with 30 points per ellipse and no noise, and after achieving reasonable results, it was gradually adapted and trained on increasingly complex datasets. The objective of this development process was to create a neural network that could surpass the performance of the LS algorithm on a testing set containing diverse ellipses with a multitude of degrees

of freedom.

Figure 6.1 shows results from two early networks trained on datasets with 30 points per ellipse, with and without noise. P1 and P2 are the excitation fractions of ensemble 1 and ensemble 2, respectively. The neural network fit is in blue and the 'truth' is plotted in black. The nine ellipses in each plot are randomly selected ellipses from a testing set of 100 ellipses. Additionally, the average loss (from equation 6.1) for the NN and LS algorithms are included above each plot. Smaller loss indicates a more accurate algorithm. Note that we do not apply the least-squares bias correction from Chapter 3.4 for these results.

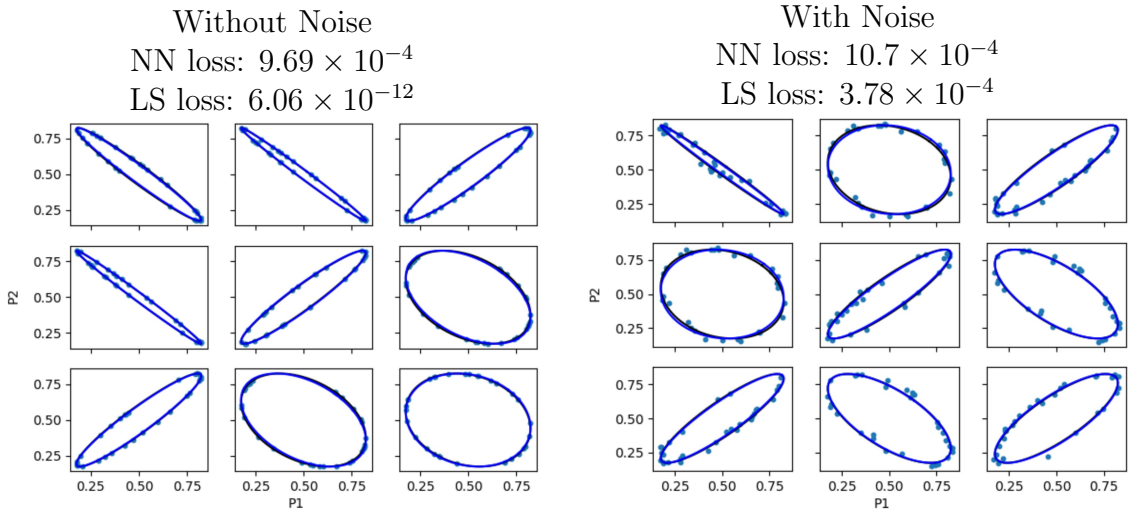

Figure 6.1: Initial Neural Network (NN) results. These plots show fits for nine randomly selected ellipses from a test set of 100 ellipses, where P1 and P2 are the excitation fraction of each ensemble. Two neural network are trained on data with (left) and without (right) quantum projection noise. The average loss over the testing set for the NN and the least-squares (LS) methods is included above each figure. The blue curve is the NN fit and the black curve is the true ellipse.

The neural network loss is similar for both datasets, while the LS algorithm performs much better on noise-free data. The two algorithms have similar loss on the data with noise (note, however, that we use the LS algorithm without bias correction for the loss calculations in Figure 6.1). This indicates that changes in complexity of the data affect the LS algorithm more than the NN.

Next, the network was modified through zero padding techniques, as outlined in Chapter 4.2.3, to enable processing of up to 500 points, as opposed to the previous fixed number of 30. Following this modification, the network was trained on a training set featuring ellipses with varying numbers of points, ranging from 5 to 500. It is important to note that this dataset only consisted of ellipses with ϕ_d on the interval $[0, 0.15]$ and $[\frac{\pi}{2} - 0.15, \frac{\pi}{2}]$ radians. This interval is where the LS method is most bias could potentially be outperformed by the NN. Additionally, the LS-bias correction was applied to the LS algorithm results in **Figure 6.2**. The loss for the NN, LS, and MLE algorithms are included below the plot.

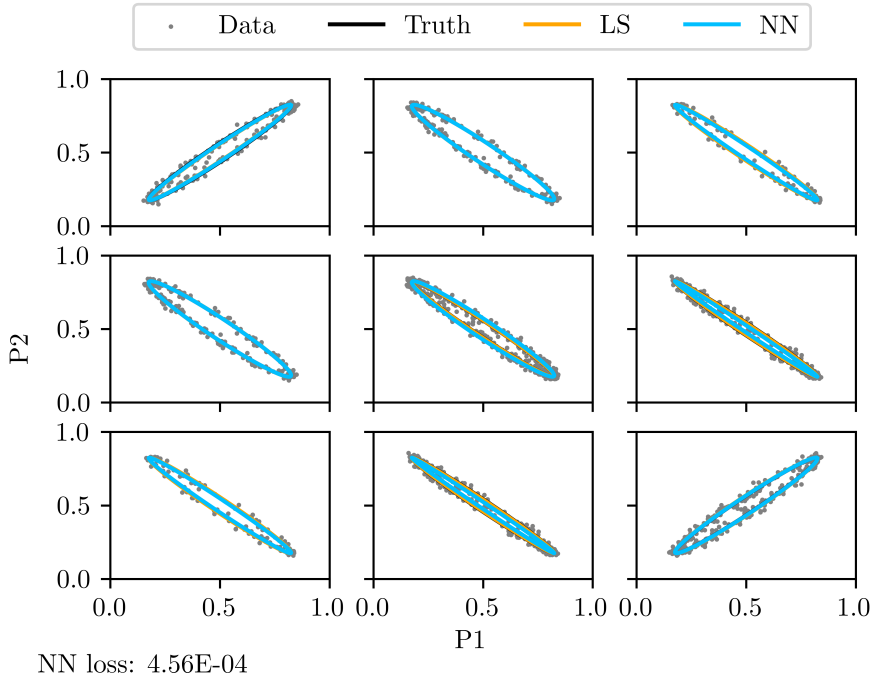

Figure 6.2: Additional Neural Network results. This figure shows nine randomly selected ellipses from a test set of 100 ellipses. Each ellipse has a random number of points between 5 and 500. P1 and P2 are the excitation fraction of each ensemble. The average loss over the testing set for the NN, LS, and MLE methods is provided below the plot. The NN fit is plotted in blue, LS in orange, and the truth in black.

This NN has loss roughly an order of magnitude larger than the LS and MLE

loss. This network was trained on roughly one-million testing ellipses for 200+ epochs. After extensive training, the network was unable to produce loss lower than the LS method.

To investigate bias, phase errors are plotted against true phase, shown in **Figure 6.3**. The NN errors are in blue, bias-corrected LS in orange, and MLE in green.

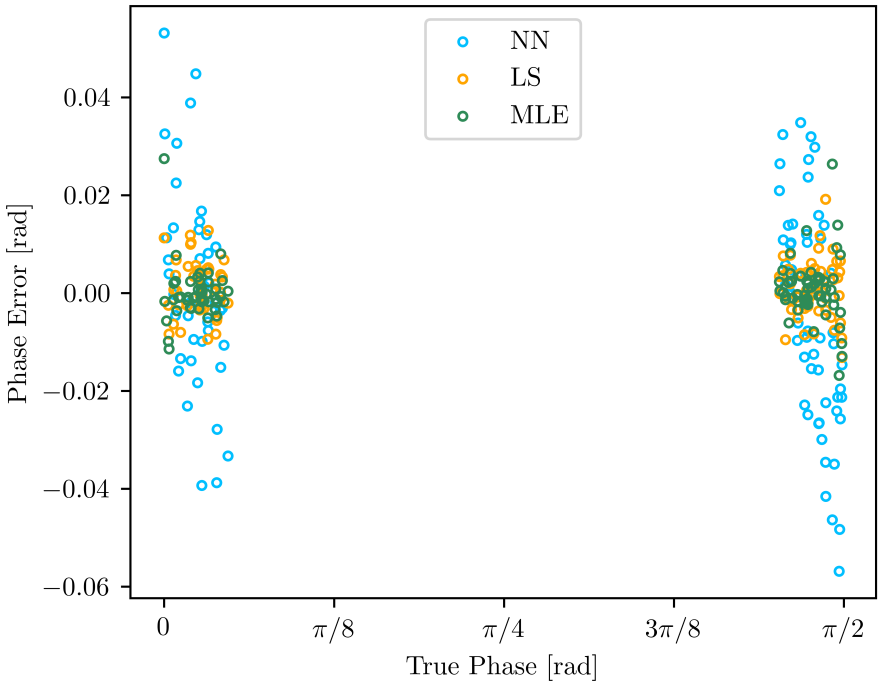

Figure 6.3: NN Phase Bias. The phase errors for each sample in the testing set are plotted for the NN, bias-corrected LS, and MLE algorithms. The NN errors are seemingly independent of true phase.

The NN does not seem to have strong phase bias for this particular testing set, and a bias-correction could not be successfully applied to the NN in the same way it was for LS. Future work could investigate the bias of the NN for data with high-noise or few shots, and a bias correction could potentially be applied for these regimes.

The bias-corrected LS algorithm outperformed all neural network models de-

veloped in terms of minimizing loss. Despite various network architectures and training datasets being implemented and tested, the performance of the neural network could not surpass that of the LS algorithm. While this outcome may be discouraging, it is important to note that the neural network may still have potential for further improvement. For instance, training for more epochs or conducting more in-depth hyperparameter sweeps could potentially enhance the performance of the network. Next, we analyzse the performance of the MLE algorithm.

6.2 Maximum Likelihood Estimate

The MLE algorithm performs significantly better than the neural network and is able to accurately fit ellipses with ϕ_d in the full range of $[0, \frac{\pi}{2}]$. **Figure 6.4** shows the performance of MLE tested on 100 different ellipses with varying ϕ_d. The loss for the MLE and LS algorithms on this test set is included on the top right of the figure.

Although the phase error plot does not provide a clear indication of which algorithm performs better overall, calculating the loss provides a quantitative measure of algorithm accuracy. However, the loss for both algorithms is comparable, making it challenging to distinguish their performance based on this metric alone.

To compare MLE to LS further, we plotted the loss against the number of points on the ellipse for both algorithms. As the number of points increased, the loss decreased for both algorithms, which intuitively makes sense. More points mean more information, which leads to a better fit. However, the rate of loss convergence differed between the two algorithms. To demonstrate this, we simulated 20 sets of 95 ellipses, with each set having a unique ϕ_d value ranging between 0 and $\pi/2$. Within each set, one ellipse had five points, another six, and so on up to 100 points. All ellipses were simulated with 1500 atoms in each ensemble. **Figure 6.5** displays the

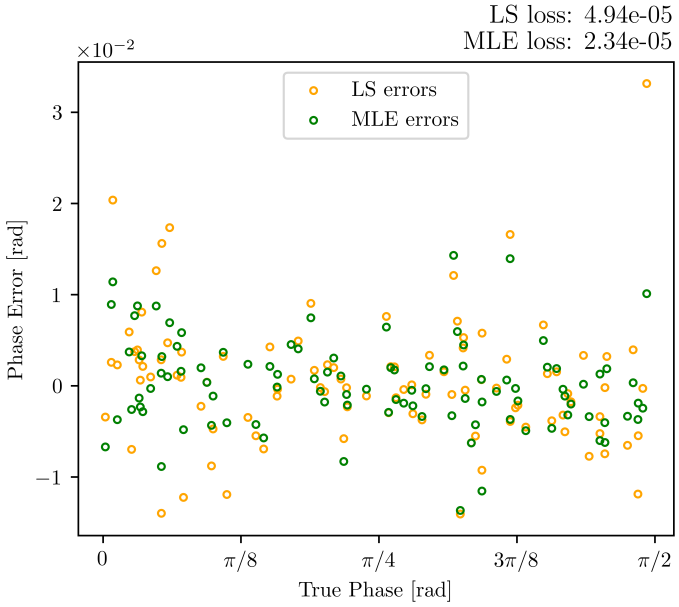

Figure 6.4: MLE Phase Errors. 100 different ellipses spanning the range of ϕ_d were simulated and fit using MLE and LS. Phase error is plotted against true phase. The MLE loss is lower than the LS loss by roughly a factor of 2.

relationship between the average loss and the number of points for both algorithms.

The LS algorithm outperforms MLE in fitting ellipses with fewer points, as indicated by its lower average loss at low shot numbers. However, as the number of shots increases, the MLE algorithm outperforms LS with lower loss. This trend is demonstrated in **Figure 6.6**, which analyzes the same data from Figure 6.5 but for ellipses with more than 55 points.

The results indicate that LS is a preferable algorithm when dealing with a small number of data points. On the other hand, MLE has a faster rate of loss convergence and is the more favorable option when there are a large number of points on an ellipse (for this data, roughly more than 40 points).

The algorithms are further compared through the metric of noise. The same procedure to construct Figure 6.5 is followed to create six batches of data with varying number of atoms per ensemble. Fewer atoms results in more noise. The results are presented in **Figure 6.7**, which illustrates the relationship between noise

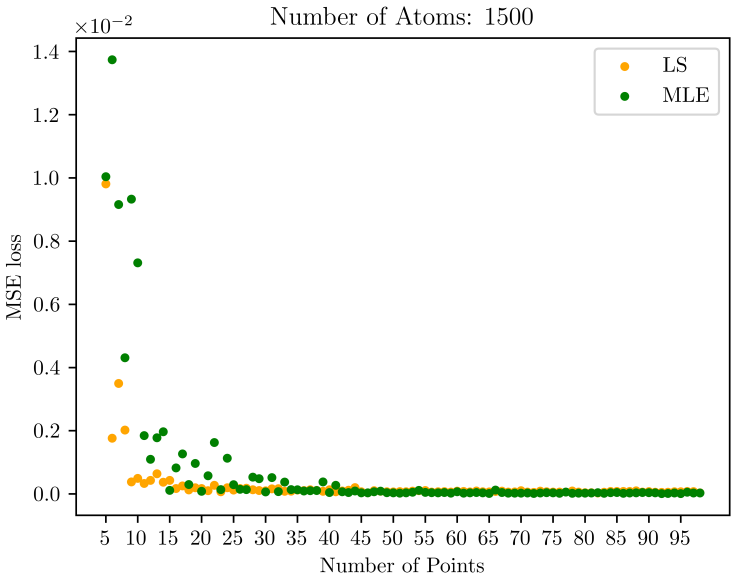

Figure 6.5: Comparing MLE to LS for variable shots (0-100). 20 sets of ellipses that span different ϕ_d values between 0 and $\pi/2$ were simulated. Each set of ellipses had a variable number of points between 5 and 100. The MLE and LS loss is calculated for each set and plotted against number of shots.

and algorithm accuracy.

As expected, the accuracy of both algorithms decreases as the noise level increases. This is intuitive, as more noise means that the measurements are less reliable, and therefore fitting an accurate ellipse becomes more difficult.

However, the rate at which the algorithms' accuracy decreases with increasing noise is different. The LS algorithm is much more sensitive to noise, and its accuracy decreases much more rapidly as the noise level increases. This is likely due to the fact that the LS algorithm is based on minimizing the sum of squared errors, which can be heavily influenced by outliers or noise. On the other hand, the MLE algorithm is based on maximizing the likelihood of the data, which is a more robust method that is less affected by outliers or noise.

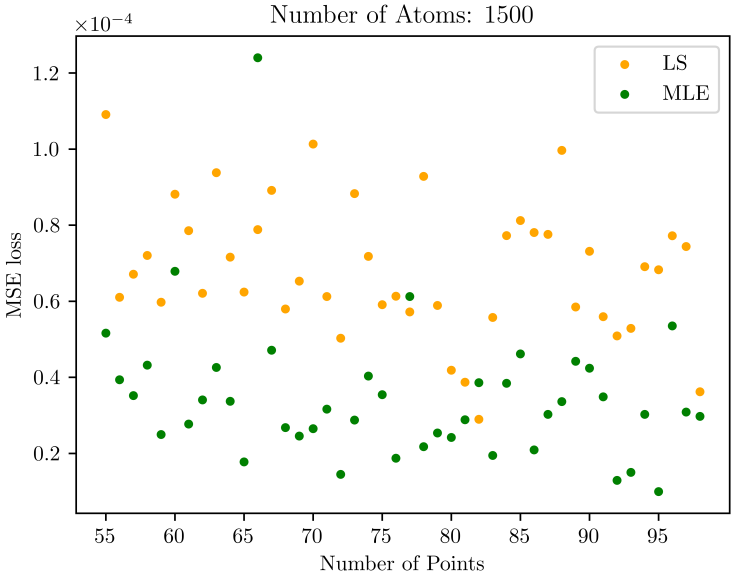

Figure 6.6: Comparing MLE to LS for variable shots (55-100). Data from Figure 6.5 but zoomed in on number of points greater than 55. The MLE average loss is lower than the LS loss in this regime.

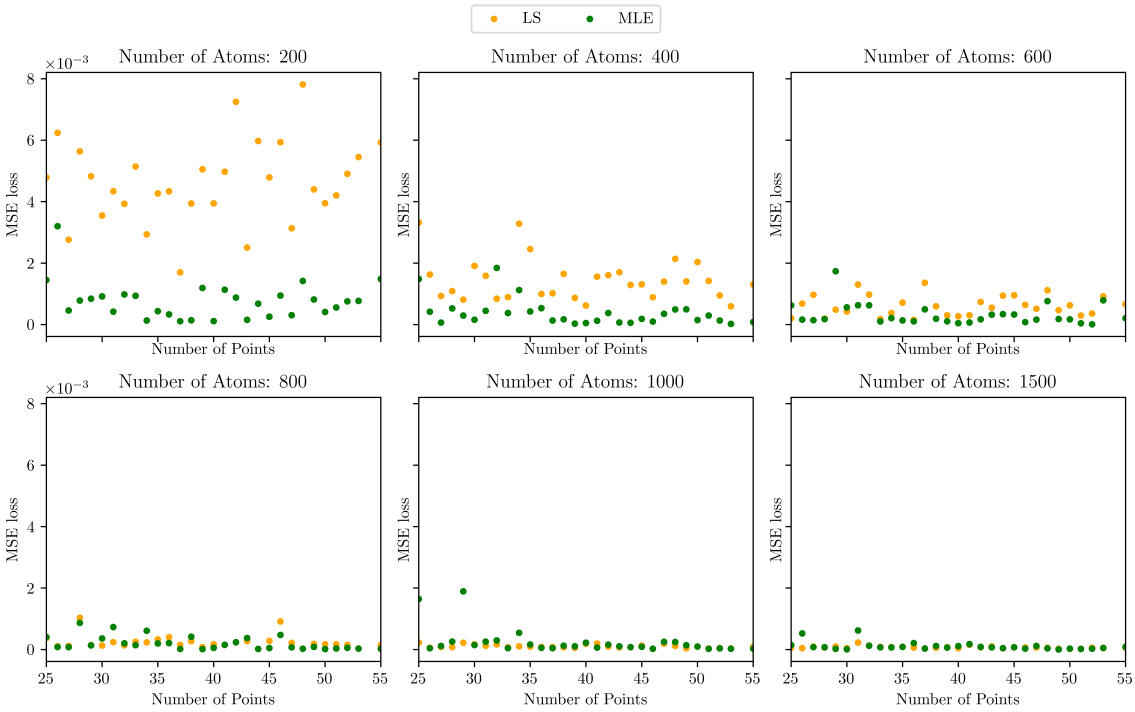

Figure 6.7: Comparing MLE to LS for variable noise. 20 sets of 95 ellipses, with each set having a unique ϕ_d value ranging between 0 and $\pi/2$ are simulated. Within each set, one ellipse had five points, another six, and so on up to 100 points. The average loss is plotted against number of points for six different batches with each batch having a different level of noise. The LS algorithm is more affected by noise than MLE, but converges as noise decreases.

Chapter 7

Conclusions

Ellipse fitting is a common tool used in science and engineering [7], [19], [20]. Generally, the task of fitting ellipses to data is not straightforward. For differential clock comparisons in the multiplexed clock, the current least-squares method is known to have bias and perform poorly when fitting data with high levels of noise [2], [7].

This work introduces two algorithms to address the challenges of ellipse fitting in atomic clock experiments: a neural network and a maximum likelihood estimate. These algorithms are compared to the current bias-corrected least-squares algorithm.

The bias-corrected LS algorithm outperformed all neural network models developed in terms of minimizing loss, and despite various network architectures and training datasets being implemented and tested, the performance of the neural network could not surpass that of the LS algorithm. However, the neural network may still have potential for further improvement through training for more epochs or conducting more in-depth hyperparameter sweeps.

The maximum likelihood estimate performs much more competitively with the least-squares algorithm. To analyze the discrepancies between the two algorithms, their performance was evaluated in relation to the number of points on an

ellipse and the amount of noise present. For ellipses with few points, the LS algorithm outperformed MLE. However, as the number of points on an ellipse increased, the MLE algorithm surpassed the LS algorithm. Furthermore, the LS algorithm performed worse than the MLE algorithm on data with high noise.

In summary, the results indicate that the MLE algorithm is preferable when dealing with data that has many points or high noise. Else, the bias-corrected LS algorithms performs best.

These findings provide important insights into the performance of different ellipse fitting methods and can help guide the development of future methods. Ultimately, the improved accuracy of fitting methods will lead to significant advancements in a range of technological fields, making it a worthwhile area of study for researchers in science and engineering.

Bibliography

[1] A. D. Ludlow, M. M. Boyd, J. Ye, *et al.*, "Optical atomic clocks," *Reviews of Modern Physics*, vol. 87, no. 2, 2015. DOI: 10.1103/RevModPhys.87.637.

[2] R. Halır̆ and J. Flusser, "Numerically Stable Direct Least Squares Fitting Of Ellipses," *Proc. of Sixth Intl Conf. Computer Graphics and Visualization*, 1998.

[3] C. J. A. P. Martins, "The status of varying constants: A review of the physics, searches and implications," *Reports on Progress in Physics*, vol. 80, no. 12, 2017. DOI: 10.1088/1361-6633/aa860e.

[4] W. J. Marciano, "Time variation of the fundamental "constants" and kaluza-klein theories," *Physical Review Letters*, vol. 52, no. 7, 1984. DOI: 10.1103/PhysRevLett.52.489.

[5] R. Lange, N. Huntemann, J. M. Rahm, *et al.*, "Improved Limits for Violations of Local Position Invariance from Atomic Clock Comparisons," *Physical Review Letters*, vol. 126, no. 1, 2021. DOI: 10.1103/PhysRevLett.126.011102.

[6] A. Derevianko and M. Pospelov, "Hunting for topological dark matter with atomic clocks," *Nature Physics*, vol. 10, no. 12, 12 2014. DOI: 10.1038/nphys3137.

[7] X. Zheng, J. Dolde, V. Lochab, *et al.*, "Differential clock comparisons with a multiplexed optical lattice clock," *Nature*, vol. 602, no. 7897, 2022. DOI: 10.1038/s41586-021-04344-y.

[8] B. V. Estey, "Precision Measurement in Atom Interferometry Using Bragg Diffraction," *UC Berkeley Electronic Theses and Dissertations*, 2016.

[9] A. Fitzgibbon, M. Pilu, and R. Fisher, "Direct least squares fitting of ellipses," in *Proceedings of 13th International Conference on Pattern Recognition*, vol. 1, 1996. DOI: 10.1109/ICPR.1996.546029.

[10] W. Gander, "Least squares with a quadratic constraint," *Numerische Mathematik*, vol. 36, no. 3, 1980. DOI: 10.1007/BF01396656.

[11] A. W. Young, W. J. Eckner, W. R. Milner, *et al.*, "Half-minute-scale atomic coherence and high relative stability in a tweezer clock," *Nature*, vol. 588, no. 7838, 2020. DOI: 10.1038/s41586-020-3009-y.

[12] H. Wang, C.-S. Leung, H. C. So, *et al.*, "Robust real-time ellipse fitting based on lagrange programming neural network and locally competitive algorithm," *arXiv:1806.00004 [eess]*, 2018.

[13] W. Dong, P. Roy, C. Peng, *et al.*, "Ellipse r-CNN: Learning to infer elliptical object from clustering and occlusion," *IEEE Transactions on Image Processing*, vol. 30, 2021. DOI: 10.1109/TIP.2021.3050673.

[14] C. Hu, G. Wang, K. C. Ho, *et al.*, "Robust ellipse fitting with laplacian kernel based maximum correntropy criterion," *IEEE Transactions on Image Processing*, vol. 30, 2021. DOI: 10.1109/TIP.2021.3058785.

[15] W. M. Itano, J. C. Bergquist, J. J. Bollinger, *et al.*, "Quantum projection noise: Population fluctuations in two-level systems," *Physical Review A*, vol. 47, no. 5, 1993. DOI: 10.1103/PhysRevA.47.3554.

[16] E. Alpaydin, *Introduction to machine learning* (Adaptive computation and machine learning), 2nd ed. Cambridge, Mass: MIT Press, 2010.

[17] G. E. Marti, R. B. Hutson, A. Goban, *et al.*, "Imaging Optical Frequencies with 100 Hz Precision and 1.1 m Resolution," *Physical Review Letters*, vol. 120, no. 10, 2018. DOI: 10.1103/PhysRevLett.120.103201.

[18] R. Peck, C. Olsen, and J. L. Devore, *Introduction to Statistics and Data Analysis*, 3rd ed. Australia ; Belmont, CA: Thomson Brooks/Cole, 2008.

[19] M. Takamoto, I. Ushijima, N. Ohmae, *et al.*, "Test of general relativity by a pair of transportable optical lattice clocks," *Nature Photonics*, vol. 14, no. 7, 2020. DOI: 10.1038/s41566-020-0619-8.

[20] B. Stray, A. Lamb, A. Kaushik, *et al.*, "Quantum sensing for gravity cartography," *Nature*, vol. 602, no. 7898, 2022. DOI: 10.1038/s41586-021-04315-3.