SPECTRAL SIMULATION FOR TAILORED LIGHT SOURCES USING EVOLUTIONARY COMPUTING STRATEGIES

by

NWAFOR CHINEDU KENNETH
(B.S., Computer Science)

A THESIS SUBMITTED FOR THE DEGREE OF

MASTER OF COMPUTING

in

ARTIFICIAL INTELLIGENCE

in the

GRADUATE DIVISION

of the

NATIONAL UNIVERSITY OF SINGAPORE

2021

Supervisors:
Professor WONG Limsoon, Main Supervisor
Professor NOVOSELOV Konstantin Sergeevich, Co-Supervisor

Examiners:
Associate Professor HUANG Zhiyong
Associate Professor NG Teck Khim

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

[illegible]

NWAFOR CHINEDU KENNETH

10 May 2021

To my family

Acknowledgments

I would like to thank my supervisor Prof. Wong Limsoon, who saw me through this project. Although we started working together during the COVID-19 lockdown and mostly had virtual meetings, he was very helpful with feedback, suggestions and general guidance on the project. My gratitude goes to Prof. Konstantin Novoselov (Kostya), under whose research group and grant this project was done. I have worked for Prof. Kostya from the first day of pursuing this degree and it has been a great honor and pleasure working for him. A Nobel laureate in Physics, he has taught me a lot despite my background in a different field. He co-supervised this project as well as other interesting projects I worked on in his lab. I am also grateful to Prof. Maciej Koperski in Prof. Kostya's research group under whose guidance much of the practical aspects of this project was done.

A big thanks goes to my parents and siblings for their immense support even when I live far away from them. Their emotional support has been very helpful in my academic endeavors. I want to thank Schaffhausen Institute of Technology for the scholarship which enabled me to pursue this degree.

Finally, my appreciation goes to my friends who have been close and supportive, making my stay in Singapore exciting and memorable even with the COVID-19 situation. Thank you all!

Contents

Abstract

Spectral Simulation for Tailored Light Sources using Evolutionary Computing Strategies

by

NWAFOR CHINEDU KENNETH

Master of Computing in Artificial intelligence

National University of Singapore

Lighting is an essential need and has become part and parcel of our daily life. Different forms of lighting are needed in different establishments for different purposes and technology has helped to improve these lighting systems. This work focuses on the development of light sources for tailored usage. We introduce a hybrid approach of light tuning and simulation that relies on the absorption properties of dyes and data analysis to approximate a light source with a desired spectrum. We also propose a new loss function for regression analysis which focuses on the shape as well as the alignment of signals and less on the euclidean distance between them. We built an experimental light filtering system which incorporates a range of optical elements to produce a target light. To optimize the tuning process, our method employs Genetic Algorithms with an objective based on the proposed loss function. Results show that the system is able to approximate the desired light sources with a minimal difference.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 A Brief History of Lighting

Lighting has been a necessity for humankind for as long as we have existed. In fact, it is an absolute necessity, not just for humans but for plants and animals as well. In the early periods of human existence, we only relied on natural light from the sun for lighting. Then several millennia ago, we discovered fire, which led to the invention of fire lamps. This improved the lighting system and meant that we could now have some light at night, but the methods of keeping the fire burning were not cheap and so we continued to seek for more ways of producing light. Fast forward to 1878 and famous inventor Thomas Edison developed the incandescent light bulb [1]. This signaled a new era of artificial light sources. Since then, we have invented many other sources of light including Fluorescent lights and Light Emitting Diodes (LED). It is not uncommon to now customize light sources to have specific properties for different functions such as growing plants [2]. These properties in part are characterized by the spectrum of the light source and hence the ability to produce light sources of specific spectra becomes important.

1.2 Background and Motivation

Visible light is electromagnetic radiation in the range from 400 nm to 780 nm that can be detected by the human eye. It is part of a much larger spectrum of electromagnetic radiation. Importantly, the perception of light does not only depend on the initial emission spectrum of the light source but also depends on

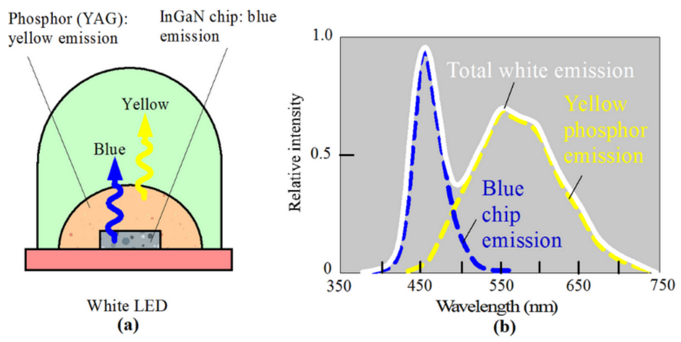


Figure 1.1: Absorption and emission of light in LED using Phosphor(source: chegg.com)

the absorption spectrum of the medium between the light source and the observer, as well as the absorption spectrum of the observer's eyes [3]. For example, color blind people only see certain colors. This is due to the cones in their retina only absorbing certain colors of light and not the others. In most variants of LED lights, phosphors are used as mediums or filters to absorb most of the luminous energy from the LED chips and emit only the visible light of specific range of wavelengths (green, yellow, etc) making it easy to customize the light [4]. When the need arises for the production of a specific kind of artificial light, we can design the spectrum based on the desired properties of the light. To match the desired spectrum we can use one or more filters to tune the original light.

1.3 Areas of Application

There is a myriad of areas where a tailored light source is useful. One is in growing plants, as earlier mentioned in § 1.1. Plants absorb most of blue light (400 nm to 500 nm) and red light(600 nm to 700 nm) but not much of green light(500 nm to 600 nm) [5]. Hence light sources for growing plants are tailored to match that spectrum as any green light component will be a waste. Another application is in the design of cleanrooms. A cleanroom is a controlled environment with a very low permissible levels of solid, liquid and gaseous pollutants. They are typically used in many manufacturing sectors including electronics, pharmaceuticals, medical equipment, optics, etc. Apart from keeping away particles, some also need to ensure that light of certain wavelengths are not permitted in, as it could interfere

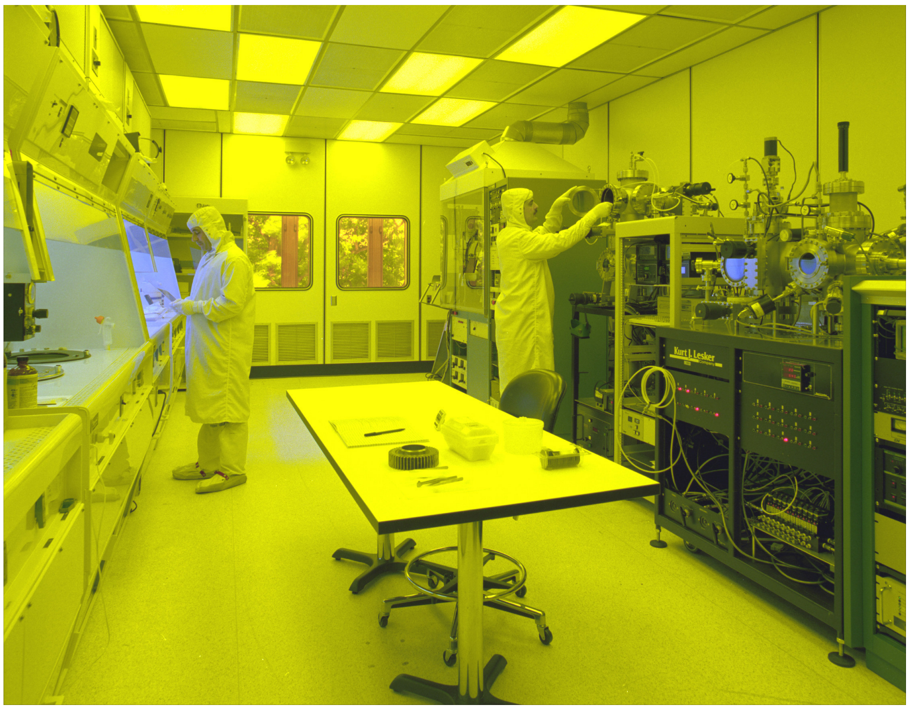


Figure 1.2: Cleanroom used for the production of microsystems. The yellow (red-green) lighting is necessary for photolithography, to prevent unwanted exposure of photoresist to light of shorter wavelengths.(source: Wikipedia)

with the manufacturing equipment. This can be the case for cleanrooms used for manufacturing optical products. In this case, specific mediums or filters could be designed to absorb and filter out specific wavelenghts of light that are considered harmful.

1.4 Evolutionary Computation

Evolutionary Computation(EC) is a class of algorithms for optimization inspired by biology and evolution. A major advantage of this class of algorithms is that they are agnostic of the properties of the optimization problem. As a result, they are used for global optimization [6]. This attractive property is one of the reasons we chose to use EC in this work. Furthermore, Evolutionary Computation has been used in many engineering systems, some of which we will discuss in Chapter 2.

1.5 Research Goals

This project was done in collaboration with a research group in the Material Science department. The goal is to design a system, such that given a desired spectrum of light, the system can output a solution that includes the light intensity and combination of filter components needed to achieve this spectrum.

1.6 Major Idea

As already mentioned earlier, the kind of light that the observer perceives can be altered by mediums which we called filters. So, first we design an experimental setup to measure the absorption spectra of these filters at different intensities of light. We will discuss in later chapters about the chemical components used as filters and their properties.

Having obtained the data, we design an algorithm based on Evolutionary Computation to find the best combination of these spectra and their corresponding intensities that will best fit the target spectrum. With this information, a light source can then be designed with the properties characterized by the target spectrum, using these components as filters.

1.7 Contribution

Research works on light tuning have primarily focused on the physics of the fluorescent materials to obtain the desired light. This study introduces the following:

1. A filter system that combines the physical properties of fluorescent materials with data analysis to obtain a dynamic light tuning system.

2. An effective data acquisition process that allows the objective function of our optimization algorithm to focus on the important properties to be predicted.

3. A loss function for regression analysis on signals which emphasizes on the shape and alignment of the signals instead of the euclidean distance between them.

4. An evaluation system based on simulated spectral data using a lighting and colorimetry toolbox.

1.8 Thesis Synopsis

The organization of the rest of this thesis is as follows: Chapter 2 reviews related work and commonly used algorithms for the processing and analysis of spectral data. In Chapter 3, we describe the experimental setup, data acquisition process and formulate the optimization problem. Chapter 4 describes our solution in detail, touching on our methodology and assumptions. In Chapter 5, evaluation will be done, with results shown and discussed.Chapter 6 will discuss possible improvements and future works, then bring a conclusion to the thesis.

Chapter 2

Literature Review

2.1 Light Simulation and Tuning

Due to varied uses of customized light sources as discussed in § 1.3, researchers have put in some work to devise ways to tune and tailor different light sources. These research endeavours are fuelled by the need to develop lighting technologies which are more efficient and consume less power, are environmentally friendly, and in some cases have sunlight-like emission [7]. Sunlight as a natural light is very suitable to human eyes due to our luminosity function which has in turn evolved to make optimal use of the light from the sun. Most of the research work focus on tuning LED lights which according to [7], is rapidly replacing older lighting technologies like the incandescent light bulbs, fluorescent lamps and high-intensity discharge lamps. As we briefly discussed in § 1.2, LED sources are powered by diodes (blue or ultraviolet) that send light through luminous materials called phosphors. A lot of research has focused on developing even better phosphors, including *nanophosphors* [8] which are nano-sized phospors. To tune the light, the intensity of the light from different diodes can be varied or multiple phosphors can be combined by means of a balanced spectral overlap and the relative ratio of their emission colors [7]. This principle of spectral combination is quite important and central to this work.

Silva et al. in [9] explored a method of generating with high accuracy, the three primary light colors simultaneously, which allowed visible light synthesis when combined. In their work, they developed a doped fluoroaluminate-based glass which controlled the light output. Doping with rare earth metals are often used on phosphores to enhance the luminosity [7]. In [9], the researchers use rare earth ions (Eu^{3+}, Tb^{3+} and Tm^{3+}) for the doping. These materials in their right proportions act as filters, absorbing some parts of the light and only emitting certain parts which resulted in light with the three primary colors. In [10], the authors condition the filters to generate red light, this time using Lead boro-telluro-phosphate glasses. Similarly, Zhou et al. in [11] developed color-tunable glasses by doping oxyfluoroborate glass with Eu^{3+}.

A common theme here is that researchers try to fine-tune the light generated from these sources by using luminescent materials which when used as filters, result in the corresponding colors of light. In essence, the research effort is mostly on developing materials with the desired photo-luminescent properties. However, in our case we do not only focus on the photo-luminescence of the material but also try to simulate and tune the light by finding optimal combinations algorithmically.

2.2 Spectral Processing and Analysis

To the best of our knowledge, no work has focused on the same exact task we explore in this project, but there are similar concepts related to spectral processing and analysis that have garnered a lot of research work over the years. Some of these we will discuss in the following subsections.

2.2.1 Definitions

A *filter* is a light absorbent medium that absorbs some of the light that passes through it and transmits the rest. Depending on the angle of the incident light, some of the light will also be reflected. In this work, we use dye solutions as our filters which are highly transparent, and so the reflectance is negligible. Ignoring biochemical noise, instrument error and reflectance, the amount of light absorbed by a filter A and the amount of light transmitted through T, sum up to the total amount of light Φ incident on the medium.

$$\Phi = A + T \tag{2.1}$$

With the output of multiple filters combined, the resulting light spectrum is simply a linear combination of the outputs of the individual filters [12].

2.2.2 Linear Deconvolution

Given an observed light spectrum S and a set of k reference spectra $f_1...f_k$, a common task is to fit the reference spectra in the observed spectra. Expressed mathematically, we arrive at the following equation:

$$S = c_1 f_1 + c_2 f_2 + ... + c_k f_k \tag{2.2}$$

This task has been called *Linear Deconvolution* [13]. The authors in [12] developed a framework called *Specter*, which performs a linear regression to solve this task. Their work aims to *deconvolve* a mixture of peptide precursors into individual precursors represented in a reference spectral library. Hence, it must select samples from the spectral library which give the best combination required to achieve the target mixture spectrum. Unlike in some other cases, the number of reference spectra to be used in the combination here is dynamic, resulting in a combinatorial complexity.

In [14] by Ciach et al., the authors employ an approach of computing the *Wasserstein* distance between the target spectrum and the combination of the reference spectra. The Wasserstein distance is a metric used to quantify the dissimilarity between two probability distributions [15]. It is measured as the minimal cost of transforming one distribution to another, one point at a time. Specifically, the authors view a spectrum as a multimodal distribution and use the following formulation to compute the Wasserstein distance:

Given a spectrum $S(x)$, defined as an intensity function of mass (their work uses a spectrum over mass rather than a spectrum over wavelengths), and similarly $F(y)$ which is a function that combines the reference spectra $f_1...f_k$ for any mass y, let $\gamma(x, y)$ be the amount of signal transported between x in spectrum S and y in combined spectrum F, then the corresponding Wassertein distance between S and F is given as:

$$W(S,F) = \min_{\gamma \in \Gamma} \sum_{x \in S, y \in F} |x - y| \gamma(x,y) \tag{2.3}$$

The function γ is referred to as the *transport plan* [15]. Having formulated this as an optimization problem, they solve it using linear programming to retrieve the optimal parameters for the combination. The Wassertein distance has also been employed in Deep Learning to stabilize the training of *Generative Adversarial Networks (GAN)* in a variant named *Wassertein GAN* [16].

Genetic Algorithms have also been employed in the more general deconvolution problem. As defined by [17], deconvolution is a process of extracting pure signals from the measured signals which usually contain distortions. Consider that we want to measure a time-dependent signal $o(t)$, then due to *convolution* in the measurement process, we instead obtain the signal:

$$i(\tau) = \int_{-\infty}^{\infty} o(t) s(\tau - t) dt \tag{2.4}$$

where $i(\tau)$ is the measured function and $s(\tau - t)$ is the distorting spread function. The authors in [17] attributed their choice of genetic algorithms for the deconvolution to the suitability of GA for constrained optimization. In this way, they make use of the prior knowledge of the shape of the actual signal $o(t)$ to constrain the search.

Some other works have also cited different, sometimes overlapping reasons for settling on GA for the task of deconvolution. In both [18] and [19], the authors cite the parallel nature of the algorithm which makes it easy to parallelize the search task while [19] also adds the simplicity of the algorithm and absence of assumptions about the search space.

A particularly interesting spectral analysis very close to ours is done by Carlevaro et al. in [20]. The analysis involved fitting a target gamma spectrum with generated reference spectra of single nuclides simulated with the *Monte Carlo* method. The aim was to identify and quantify the nuclides present in the target gamma spectrum. The task was formulated as thus: given a target gamma spectrum T with n number of channels (wavelength dimension), let $X_1 ... X_N$ be a set of vectors, each of dimension n, representing the simulated reference nuclide spectra, where N is the number of such spectra, then:

$$T = \sum_{i=1}^{N} c_i X_i \tag{2.5}$$

Where c_i is the coefficient of X_i in the mixture. The task is now to find the vector c with components $c_i ... c_N$ which best approximates the target spectrum T. For this, the authors chose to use GA. In GA, each solution is represented as an individual in the population with a genotype g which is part of the population's genotype space Φ. They use encoding and decoding functions to transform the vector c into genotype space and back, such that:

$$Enc(c) : \mathbb{R}^2 \rightarrow \Phi \tag{2.6}$$

$$Dec(g) : \Phi \rightarrow \mathbb{R}^2 \tag{2.7}$$

An important part of a GA is the *fitness function* $f(g)$ which measures how well the solution encoded by an individual fits the task. With the $f(g)$ well defined, the optimization objective of Genetic Algorithms is to find $\hat{g}$, such that :

$$\hat{g} = \arg\min_{g \in \Phi} f(g) \tag{2.8}$$

Hence, it is very important to choose a good fitness function as the choice of the function can greatly affect the result. The fitness function is equivalent to the loss function used in machine learning algorithms. A common loss function used in regression problems is the *Mean Squared Error (MSE)* but this metric is very sensitive to outliers, hence the authors chose the following fitness function:

$$f(g) = \sum_{j=1}^{n} \frac{(\sum_{i=1}^{N} c_i X_{i,j} - T_j)^2}{\sum_{i=1}^{N} c_i X_{i,j} + T_j} \tag{2.9}$$

Where c_i is i-th element of the solution to c, encoded in the genotype g, $X_{i,j}$ is the j-th wavelength value of the reference spectrum X_i and T_j is correspondingly the j-th wavelength value of the target spectrum T. In this way, each squared difference is normalized separately by a corresponding value before being summed. We will explore the choice of fitness functions in more details in later chapters.

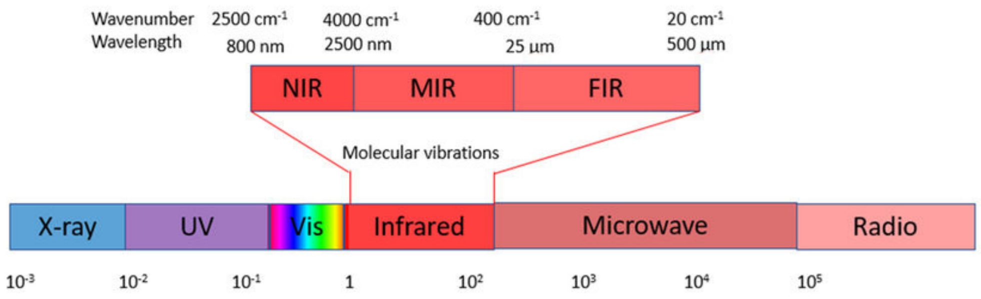


Figure 2.1: The NIR region(source: [21])

For evaluation, they both used samples of measured and simulated target spectra in order to discard the effects of measurement noise. The errors were less for the simulated case with the largest deviation at 2% while it was 17% for the measured target spectra. This difference can be attributed to measurement noise and other forms of interference involved in physical measurements compared to the Monte Carlo simulation.

2.2.3 Adulterant Quantification

Another common procedure which aims to decompose spectra of mixtures is adulterant detection and quantification. Given the spectrum M of a substance, the first task is usually to determine if the substance is pure or has been adulterated. The second task is to determine the percentage of adulteration. Depending on the nature of the task, it could also be required to identify the adulterant. The task can be expressed mathematically as this:

$$M = pS + qA \tag{2.10}$$

where S is the pure substance, A is the adulterant and p, q are their respective percentages in M.

It is important to note that most works on adulterant quantification and food quality analysis make use of Near-Infrared(NIR) spectra which has a range between 800 nm - 2500 nm, just after the visible light region as well as spectra from the nearby Mid-Infrared(MIR) region.

Subramanian et al. in [22] explained that the frequent use of spectra in the NIR region for food analysis is due to the simplicity and non-destructive effect of the measurement process which often applies Fourier spectroscopy. In addition, NIR spectra are quite characteristic of the chemical composition of the substance as the absorbance spectrum is mostly dictated by the stretching vibration overtones and combination modes of hydrogen bonds such as *C-H, O-H, S-H* and *N-H* [23].

Another common theme found in adulterant quantification is the use of the *Partial Least Squares (PLS)* regression method and its derivatives. Researchers argue that PLS is well suited for data with numerous and highly correlated independent variables also called predictors [24], [25]. Arguments for PLS also mention its suitability for data with a higher number of predictors than data samples, a major issue encountered with the *Ordinary Least Squares (OLS)* regression method [26]. Since this is the case for most data used in chemometrics, PLS has quickly become the standard.

PLS can be summarized as follows: Let $X \in \mathbb{R}^{N\times k}$ be a matrix of k-dimensional predictors and $Y \in \mathbb{R}^{N\times m}$ be the matrix of m-dimensional independent or response variables. PLS then projects both X and Y into different dimensions often smaller than their original dimension. To do this, a similar decomposition used in *Principal Component Analysis(PCA)* is used such that:

$$X = TP^{\mathsf{T}} + E \tag{2.11}$$

$$Y = UC^{\mathsf{T}} + G \tag{2.12}$$

Where P is a matrix known as the loadings or summaries of X and E is a matrix of residuals. Similarly, C and G are the loadings and residuals of Y. T and U are the projections of X and Y in latent space. The decomposition is performed using the *NIPALS* algorithm[27] until U and T converge. Having obtained the projections, a linear regression is then performed between U and T, so that $U = \beta T$, to obtain the regression model.

Sun et al. [28] use PLS-DA, a classification variant of PLS to detect adulteration in Chinese hawthorn fruits powder and then proceed to used PLSR, the regression variant to quantify the amount of adulteration. Similarly, the authors in [29] also employed PLS in their work for detection and quantification of adulterants in gasoline. Only this time, they use PCA to determine if the gasoline is pure (binary classification into classes *adulterated* and *unadulterated*), then use PLS-DA to both ascertain the adulterant out of four possible adulterants and quantify the percentage of adulteration in 1% steps from 5 to 40%. To select the number of PLS components with the best result, they use *cross-validation* technique. They obtain quite a good result of 97 - 100% classification accuracy and 1.7 - 0.66 *Root Mean Squared Error of Prediction (RMSEP)* on the quantification. RMSEP is part of a family of Root Mean Squared Error based metrics commonly used in chemometrics. The other members of the family are: *Root Mean Squared Error of Calibration (RMSEC)* and *Root Mean Squared Error of Cross Validation (RMSECV).* Their computation is essentially the same and the names are only used to differentiate the different stages that the metrics are computed.

Other authors have also approached the adulterant quantification task using MIR spectra as mentioned ealier. In [30], the authors used PLS to quantify the adulteration of butter oil by soybean oil using both MIR and NIR spectra and reported good and similar results for both, asserting that any of them can be used to accurately solve the task.

Though PLS based techniques are the predominant techniques so far for adulterant quantification, other methods are being used as well. *Artificial Neural Networks (ANNs)* have gradually gained prominence among machine learning algorithms, since their re-discovery in the turn of the century. Architectures built on this neural biology inspired technique have continuously outperformed older machine learning techniques in a variety of tasks such as image recognition [31], natural language processing [32] and lots more. *Deep Learning* is a class of machine learning algorithms that use ANNs with many layers of nonlinear computational units for learning data representations [33]. It is precisely this ability of ANNs to learn salient representation from data, that makes them so useful because this reduces the need for hand-crafted features. In signal processing, *Convolutional Neural Networks (CNNs)* have been applied to electrocardiogram signals (ECG) [34], mineral spectrum classification [35] and audio spectral data [36], with significant improvements reported. Signals often have local relationships between neighboring points and this is well exploited by the CNN convolution operation. Specifically for adulteration analysis, Neto et al. [37] compare a CNN based architecture with *ensemble learners* and regression based techniques including PLS, for the detection of adulteration in milk using infrared spectra. On the part of the CNN, as usual, they build a binary classifier that detects that the milk has been adulterated. Then, they build a second multi-class classifier to select the adulterant from a list of six adulterants. They report that the CNN model outperformed both the ensemble methods and PLS based methods, but most importantly, the CNN model did not require the preprocessing technique applied to the other methods to perform well. This attests to the ability of neural networks to learn latent features on their own which could be better than hand crafted ones.

Apart from ANNs, Genetic Algorithms have also been used either as alternatives for PLS based methods or as a complimentary method for selecting features that are now fed into the other algorithms. Research works that have applied GA to adulteration detection and quantification include [38] where a fusion of GA and *Linear Discriminant Anaysis (LDA)* termed GA-LDA, is used to detect and quantify adulterants in biodesel, and [39] where GA is used to select the best components after PLS has been applied for adulteration quantification in honey.

The researchers in [38] argue that the computation of the *Mahalanobis distance* used in LDA requires that the variables are not collinear, as this affects inverse matrix computation, hence the need for variable selection using a GA. Each variable is encoded as a binary chromosome where 1 means it should be selected and 0 means it should not. The length of each individual's genotype is the same as the number of variables which is equivalent to the number of wavelengths in the spectrum. As already discussed earlier, the fitness function used in a GA is very important and has to be chosen carefully. To improve the performance, the authors designed their fitness function to fit these two objectives:

- minimize the classification error
- minimize the number of variables

That is, the second objective penalized solutions with more variables. The authors reported that this approach achieved better results compared to only using the classification error as the objective. The result shows that the mono-objective case selects about 20 variables on average with an average of 14% classification error rate while the multi-objective variant selects only 7 variables on average with an error rate of 11%.

2.3 Conclusion

In conclusion, many techniques have been applied in the processing and analysis of spectral data. Depending on the task, specific algorithms could perform better than the others and the final result strongly depends on the dataset, loss function and training procedure as well as the hyperparameters used for training. We have chosen to use Genetic Algorithms in this work because of the simplicity and other features which we shall explain in later chapters.

Chapter 3

Problem Statement

In the last chapter, we reviewed common tasks and methods used for spectral processing and analysis. In this chapter, we go into details about the specific task we wish to solve in this project, first by explaining in full details, the experimental setup and data collection process, then proceed to formulate the analytical problem. We also discuss the preprocessing techniques we employed to treat the data before analysis.

3.1 Research Question

Before we proceed to describe the experiment, it is important to again pose the question that led to the experiment. As already discussed in previous chapters, the output of a light source can be customized with the help of filters that absorb some portions of the light and transmit the rest. This allows us to effectively fine-tune the light to our needs. On the level of Physics, this fine-tuning of a light source can be reduced to a task of fine-tuning the light spectrum. The question then becomes:

> Given a light beam and n chemical samples as filters, how can the output of the beam, after it passes through each filter, be tuned so that summed together, they approximate a target spectrum T?

While one can attempt to answer this question analytically if the absorption properties of the filters are well known, real life data is often messy due to uncertainties in the measurement instruments, so that a solution computed solely analytically cannot generalize to real data. Hence, to answer this question, we decided to take an empirical approach and designed an experimental setup which we shall describe in the following sections.

3.2 Experiment

For the purpose of this experiment, we use five dyes as filters due to their absorption properties. The setup consists of a light source, beamsplitters, the dye samples, motorized rotation stages and other components all assembled on a breadboard, most of these having been acquired from Thorlabs Inc., an optical equipment manufacturer. Optical fibres are used to collect the transmitted light outputs and combined with fibre couplers, with the final output connected to the spectrometer. See the component diagram of the entire setup in Figure 3.1.

Below we describe some of the major components of the setup and how they interact with each other.

3.2.1 Light Source

The initial beam is generated by a broadband light source which spans the visible and infrared wavelength regions (360 nm - 2600 nm). The intensity can be controlled by shutters mounted on the breadboard. Currently, we use two shutters to control the beam energy before they get to the dye samples. Due to the positioning of the light source relative to the samples, a mirror is placed to reflect the light at 45° so that it is accurately redirected to the samples.

3.2.2 Beamsplitters

There are five dye samples which means the beam has to be split to get to each sample. The Thorlabs *Cube-Mounted Pellicle Beamsplitters* are mounted in such a way that each one splits the beam into two equal beams. One part of the beam is reflected again at 45° to a sample while the other passes through the splitter to the next splitter. Subsequently, this is done till the last splitter which is a complete mirror and reflects all the light it receives to the sample. Notice that this implies that the amount of light each sample receives is different: the first receives half of the light, the second receives a 4th, etc. To balance the beam intensity, the output from

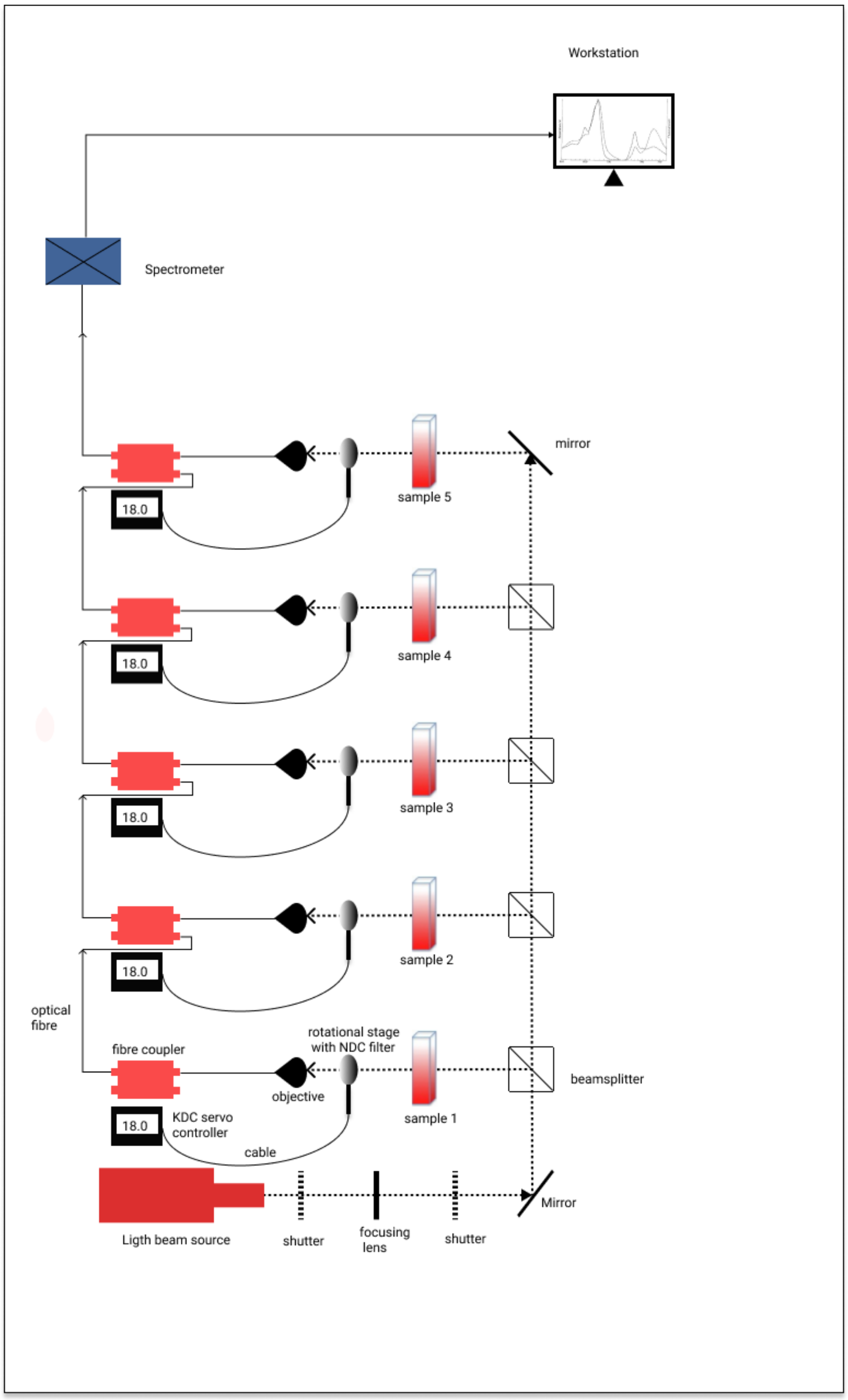


Figure 3.1: Component diagram of the experiment setup

the samples are passed through couplers which in turn split the output intensity in such way as to make them equal before passing the final output to the spectrometer.

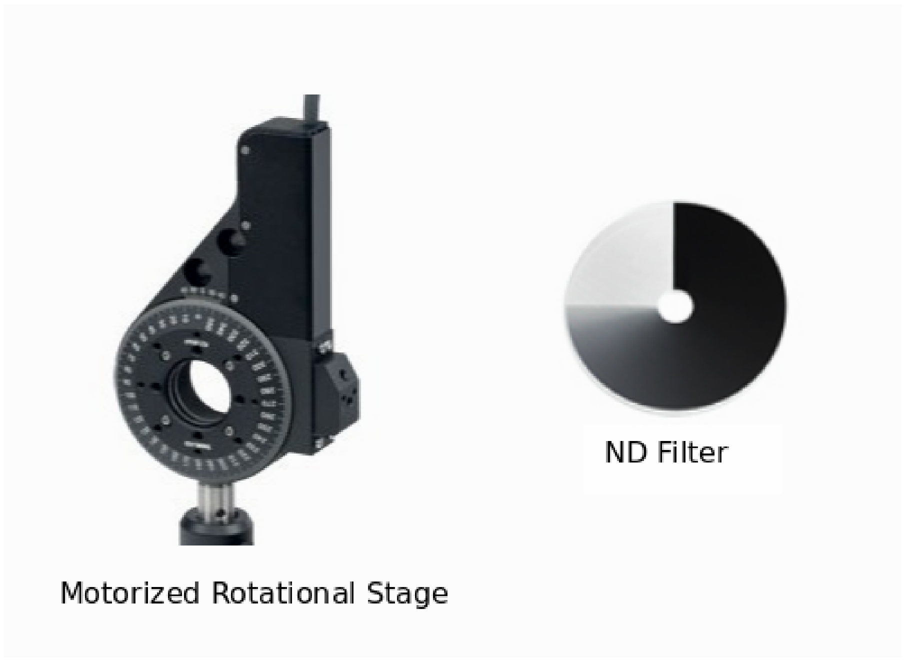


Figure 3.2: Thorlabs motorized rotational stage and Neutral Density Filter

3.2.3 Rotational Stages

The rotational stages are motorized components with a lens fitted that are used to control the intensity of the light output transmitted through each dye solution. The lens is a variable *Neutral Density (ND)* filter with optical density of 0.04 - 2.0 from Thorlabs. The round lens starts out opaque at 0° and becomes completely transparent at 270°. In this project we only rotate from 0 to 180°. The rotational stage is controlled from the Thorlabs *KDC Servo Motor Controller* which has controls to rotate the angles of the stage. The controls of the KDC controller can also be automated via APIs from Thorlabs which have bindings for several languages and software platforms such as C#, Visual Basic and LabVIEW. In this project, we use the C# bindings to automate the controller and rotate the stage and filter to take measurements. Figure 3.2 shows the motorized stage and ND filter. The stage is connected by cable to the KDC controller as shown in Figure 3.1.

3.2.4 Microscope Objectives

The objectives are used to focus the transmitted light from the ND filter into the optical fibres. For this, we use the Thorlabs *Olympus Plan Achromat* objectives which are designed for a tube lens focal length of 180 mm. An optical fibre is connected to each objective to collect the focused light and send to the spectrometer. However, the collected beams are first combined using the coupler before getting to the spectrometer.

3.2.5 Fibre Couplers

The couplers are used to combine the output beams collected by the different optic fibres. Apart from that, it serves another important purpose in our setup. Recall that in § 3.2.2, we mentioned that the samples receive beams of different intensities due to the beamsplitters. The couplers have two inputs and two outputs. Each input beam is halved with the halves sent to each of the two outputs. If only the beam from one output is collected, the original input is effectively halved. We use this mechanism to balance the output intensity of the light transmitted through each sample so that they are proportional across all samples.

After all the output optical fibres are passed through a cascade of couplers, the final output is obtained and sent to the spectrometer via another optical fibre.

3.2.6 Spectrometer

The spectrometer is an *AvaSpec SensLine* spectrometer from Avantes, a spectroscopy instruments developer. The spectrometer has a usable range of 497 nm - 1027 nm covering parts of the visible light and infrared regions. For this project we take readings of 1000 points across the range 497 nm - 770 nm. The product ships with a software through which the spectrometer can be operated as well as a driver with language bindings. Here, we also use the C# bindings available for the Microsoft .NET framework to enable us automate the measurement and incorporate it with the rest of the setup. The optical fibre from the last coupler is connected to the spectrometer to take the readings. Readings are stored in spreadsheet files in the machine from where the spectrometer is operated.

3.3 The Dyes

The dyes are absorption mediums or filters we adopted for the experiment. There are five different dyes: *DCM, Pyridine-1, Pyridine-2, Styryl-8* and *Styryl-9*, each

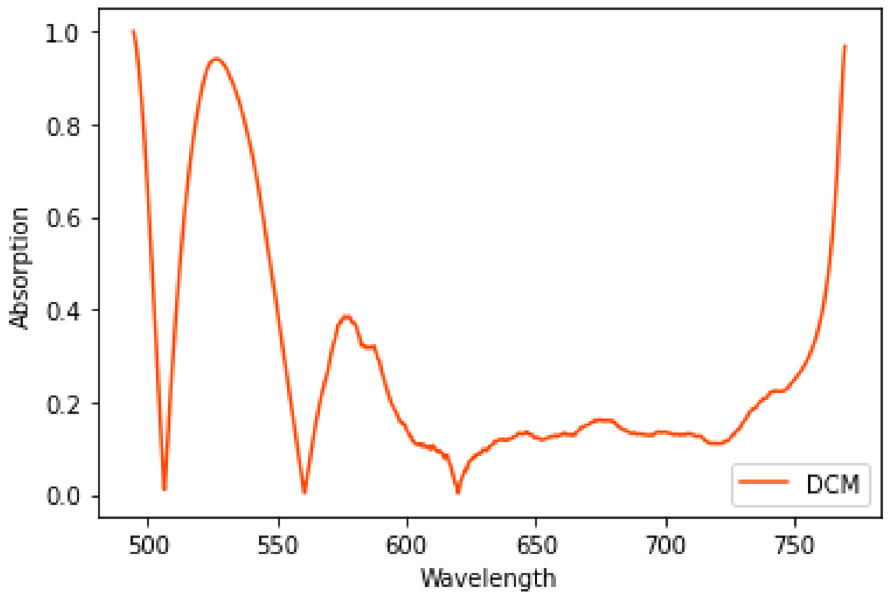


Figure 3.3: Absorption spectrum of DCM at 10^{-4} mol/L in Ethanol

of them an organic compound. To obtain the solution, each dye is dissolved in ethanol in different concentrations. Each solution is placed in a 4 ml cuvette which is attached to a cuvette holder fastened on the breadboard. Below, we discuss each dye and their spectral properties.

3.3.1 DCM

The organic compound *4-dicyanomethylene-2-methyl-6-(p-(dimethylamino)styryl)-4H-pyran* shortened DCM has a luminescent activity in the 600 nm – 700 nm region and is widely used in laser spectroscopy [40]. According to Bondarev et al. in [41], the spectral-luminescent and electrophysical properties of DCM are largely determined by charge transfer between terminal electron-donating (dimethylamino group) and electron-accepting (pyran ring with two cyano groups) substituents. The dye is reddish in color and hence has absorption peaks in other regions of the visible spectrum with low absorption in the red region. See Figure 3.3 for the absorption spectrum we obtained for this dye in our experiment.

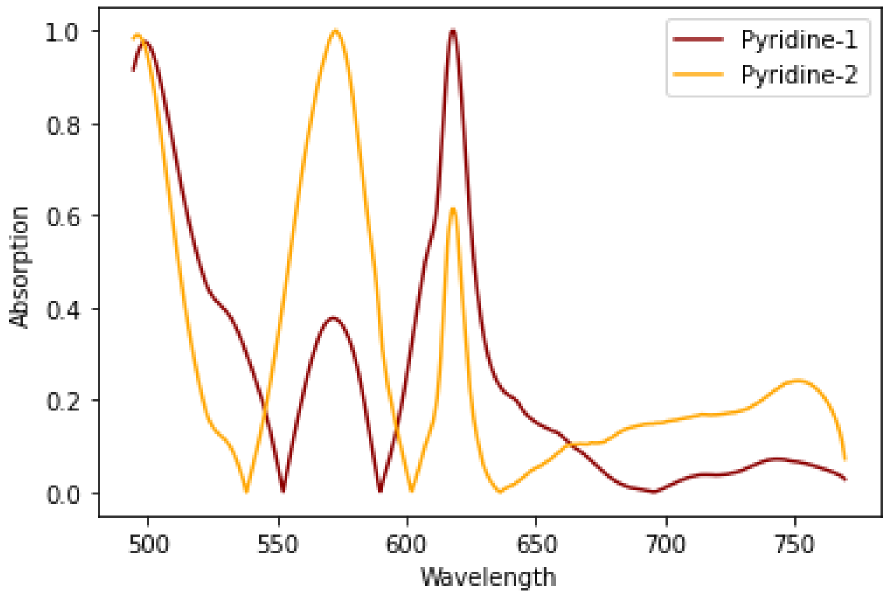


Figure 3.4: Absorption spectra of Pyridine-1 at 10^{-4} mol/L in Ethanol and Pyridine-2 at 10^{-3} mol/L in Ethanol

3.3.2 Pyridine Dyes

Pyridine is a family of disperse dyes with many derivatives commonly used for the synthesis of azo dyes for polyester fabrics [42]. These dyes have a range of colors and Al-Etaibi et al. in [42] cite that the color differences result from changes in substituents on the diazonium part of the dyes. In our experiment we use two derivatives:

- *2-[4-[4-(dimethylamino)phenyl]-1,3-butadienyl]-1-ethylpyridinium monoperchlorate* or Pyridine-1 with chemical formula $C_{19}H_{23}ClO_4$
- *4-[4-[4-(dimethylamino)phenyl]-1,3-butadienyl]-1-ethyl-pyridinium perchlorate* or Pyridine-2 with chemical formula $C_{19}H_{23}N_2ClO_4$

The derivatives used in the experiment showed luminescent activity in the 650 nm - 800 nm region with distinct absorption peaks in the green region.

3.3.3 Styryl Dyes

Styryl dyes derive their name from the univalent radical $C_6H_5 - CH = CH-$ which is derived from *styrene* and called styryl [43]. Their use ranges from application

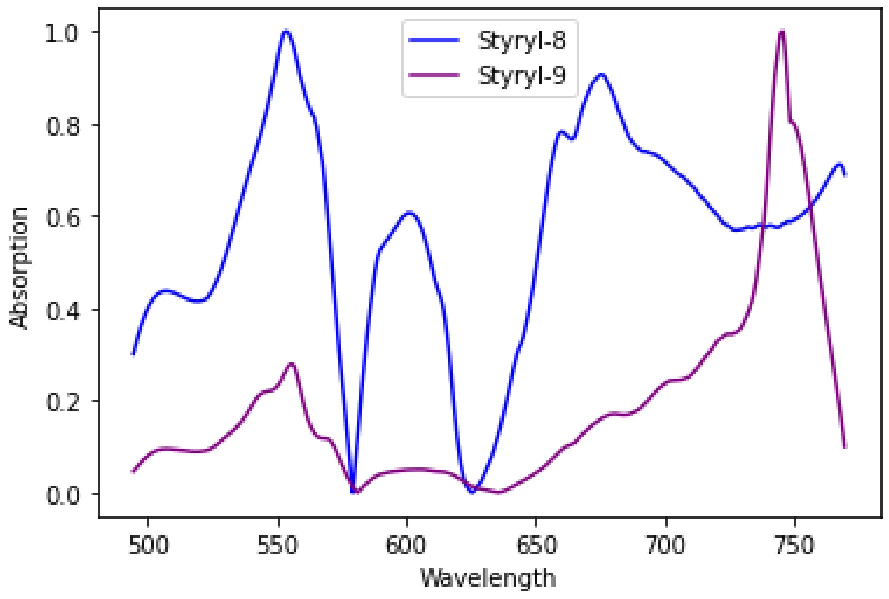


Figure 3.5: Absorption spectra of Styryl-8 at 5×10^{-6} mol/L in Ethanol and Styryl-9 at 10^{-5} mol/L in Ethanol

as colourants in polyacrylic fibres to applications as fluorescent materials due to their good spectral and photo-luminescent properties [43]. In this work, we use two variants of styryl dyes:

- *2-4-[4-(dimethylamino)phenyl]buta-1,3-dien-1-yl-3-ethyl-1,3-benzothiazol-3-ium perchlorate* or styryl-8 with chemical formula $C_{21}H_{23}ClN_2O_4S$
- *2-[6-[4-(dimethylamino)phenyl]-1,3,5-hexatrienyl]-3-ethyl-benzothiazolium perchlorate* or Styryl-9 with Chemical formula $C_{23}H_{25}N_2SClO_4$

The styryl dyes showed distinct peaks of luminescent activity across the spectrum. Styryl-8 had luminescent peaks at 580 nm and 630 nm while Styry-9 had a longer range between 580 nm and 700 nm.

3.4 Problem Statement

In § 3.1, we defined the task as that of finding the correct intensity of the output beams from each sample that would give the best approximation for a target spectrum T. From the setup, this intensity is controlled by the neutral density filters attached on the rotational stages as described in § 3.2.3. So we can further reduce the task to that of finding the optimal angle of rotation for each stage, such that the optimal intensities are achieved. With this in mind, we now redefine our analytical task as follows: let $X_1..X_N$ be vectors representing the output beam intensities after absorption by the samples and $\alpha_1...\alpha_N$ are the angles of the rotational stages, where N is the number of samples and stages, which is 5 in our case. Also, we define Φ as the ND filter function which takes a beam X_i and angle α_i and produces an output beam which is then combined with other beams and passed to the spectrometer function S. Then we need to find the best approximate for a target spectrum T, so that:

$$T \approx S(\sum_{i=1}^{N} \Phi(X_i, \alpha_i)) \tag{3.1}$$

For brevity, we can omit the spectrometer function S which leads to:

$$T \approx \sum_{i=1}^{N} \Phi(X_i, \alpha_i) \tag{3.2}$$

From Equation 3.2, it follows that we need to minimize the loss L, such that:

$$L = \min |\sum_{i=1}^{N} \Phi(X_i, \alpha_i) - T| \tag{3.3}$$

Though the optical properties of the ND filters are known and can theoretically be used to estimate Φ, it did not generalize well in our experiments, so we assume the function Φ is unknown and has to be learned. What we do know is that it is a monotonically increasing function in our setup for both X_i and α_i.

We constrain the values of the angles in the range $[0°, 180°]$. The vectors $X_1...X_N$ are of dimension n where n is the number of wavelengths which is 1000 as mentioned in § 3.2.6. One way to solve this is to collect enough data with different combinations of angles, then build a machine learning regression model on the data and proceed to predict new angles for any given target spectrum T.

3.5 Data Acquisition process

Having described the major components that make up the setup for the experiment and the analytical task we wish to solve, we now explore the data acquisition process.

To collect a single spectrum from the setup is simple and involves fixing the angles of the rotational stages using the corresponding KDC controllers, then clicking on a button on the spectrometer software to take the reading. But then, we need to take many measurements enough to train a model on. This is why we automated the process in code by interfacing with the hardware drivers to work as follows:

1. Connect to the KDC controller and set the required angle on all controllers.
2. Trigger the spectrometer to take a reading.
3. Obtain the reading asynchronously from the spectrometer and save to a memory buffer which is flushed to the file system at savepoints.
4. Repeat steps 1-3 until the number of spectra needed are collected.

A big issue here is the wide range of angles $[0°, 180°]$. For the 5 samples, collecting data at 18° intervals (0, 18,..,180) already requires $11^5 = 161,051$ datapoints to be collected. And this still makes the dataset sparse with respect to the angles because the training data will not have spectra with angles in-between such as $3°, 37°, 46°$ etc. This also implies that many target spectra whose optimal angles would be in that range cannot be accurately approximated. For the first experiment, we used 22.5° intervals, measuring only the angles [0, 22.5, 45, 67.5, 90, 112.5, 135, 157.5, 180] and acquired about 59,000 spectra. That is 59k rows of data each with 1000 columns, so that is quite a lot of data to process. This took approximately five days to collect due to the delays during the rotation of the stages and during the spectrometer readings.

Though what gets measured is the spectrum of the transmitted light, we are interested in the absorption spectrum. For this, we first obtain a reference transmission spectrum T_R using the empty cuvettes in place of the samples. Then after obtaining the transmission T, we compute the absorption spectrum A as follows:

$$A = \frac{|T_R - T|}{T_R} \tag{3.4}$$

We already established in Equation 2.1 that $A = T_R - T$. The denominator normalizes the value so that it is in the range [0, 1]. The absolute operator in the numerator ensures the intensity values are positive. In some wavelengths, luminescence is triggered through excitation, giving higher total light output than the reference spectrum which results in a kind of *negative absorption.* By Kirchhoff's law, the fluorescent emission is proportional to the absorption at those wavelengths, thus allowing us to use the absolute value in such cases.

With this data, a model can be built to minimize the objective for every spectrum T

$$\arg\min_{\alpha_p \in A} |\alpha_m - \alpha_p| \tag{3.5}$$

Where α_p is the vector of angles predicted to approximate spectrum T, α_m is the vector of angles used in the measurement to produce T and A is the angle space. Note that during inference, α_m is unknown. We of course use the ℓ_1 loss here for illustration of the objective, but other losses can be used as well.

3.5.1 Spectral Library

As already mentioned, there are issues with the data acquisition process described above, namely:

1. It takes a lot of time to collect the data.

2. The data is sparse, with many combinations of angles not represented.

3. The solution of the best combination of angles is not necessarily unique, so the optimization procedure built on the data should be designed to get any set of angles that best approximates the target data. However, with the current dataset, the loss can only be computed in terms of the single set of angles measured instead of the actual spectrum because there is no way to retrieve the spectrum for arbitrary angles not already in the data except by ad-hoc measurement which is difficult to factor into into a learning algorithm and will

be extremely slow. There should be a robust way to allow the model search for other equally good points by defining the loss in terms of the spectrum and not the angles.

To solve these issues, we referred back to the linear deconvolution task we reviewed in § 2.2.2, which makes the assumption that the final spectrum we measure, is simply a linear combination of the spectrum of the individual signals from the samples. This essentially means we can modify Equation 3.1 by pushing the spectrometer function within the sum as follows:

$$T \approx \sum_{i=1}^{N} S(\Phi(X_i, \alpha_i)) \tag{3.6}$$

With the summation outside the spectrometer function, this makes the system more flexible and dynamic so that we can now build a library of spectra for a much denser set of angles for each sample or filter, then generate any desired number of combinations for the training data. We call the spectra in this library the base spectra. We can solve the above outlined issues using this method as follows:

1. It will take much shorter time to collect the data as we only need to collect the individual sample spectra. We chose a precision or interval of 0.1° (0.1, 0.2...180), so we just need 1801 base spectra for each sample and a total of $1801 \times 5 = 9,005$ base spectra for all samples. This is much less than the 59k spectra collected before and with a much better precision.

2. The data is no longer sparse as we can represent any angle up to a precision of 0.1° in the dataset. Of course, we cannot add all possible combinations of angles with this precision as the dataset size will simply be too large. But we can remedy this by sampling 50k random combinations which should generate a more representative distribution than the data with equally spaced and sparse intervals.

3. Since we now have a base spectral library, we can define the optimization objective in terms of the spectral intensities instead of angles. Given a vector of predicted angles α_p, we will lookup the spectral values in the library and use them to compute the predicted spectrum. In this way, even if the model

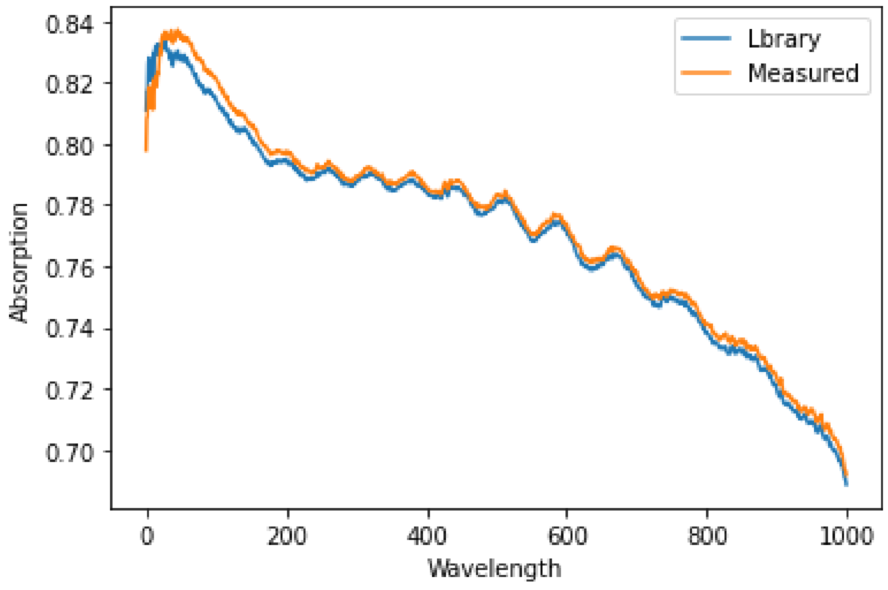


Figure 3.6: Absorption spectra: measured vs computed from library

predicts a different set of angles from the ones measured, the loss will still be minimal if the angles give a good approximation of the target spectrum.

Having identified that acquiring the data in this way is more useful, we conducted an experiment to test the assumption of linear deconvolution before we proceed to apply it. We collected separately the spectra of the samples at angles $180°, 60°, 0°, 0°, 120°$ respectively. Then we measured the spectrum with all the samples together at the same angles and got the result in Figure 3.6.

For a more extensive evaluation, we collected 200 measured sample spectra and the corresponding base library spectra for their angles. For each of the sample spectra, we computed the corresponding spectrum using the base library spectra and studied the distribution of the Mean Absolute Error (MAE) between the measured and computed spectra. The experiment was done without any preprocessing on the spectra to avoid any influence on the result. The distribution is shown in Figure 3.7. The error distribution shows that the bulk of the computed spectra only differed by less than 0.01 (1%) from the measured spectra, with the maximum difference at 1.5%. The difference can be attributed to measurement noise and time batch effects.

After confirming that this property indeed holds in our experiments, we collected the spectra to build the full spectral library as described above. We also collected

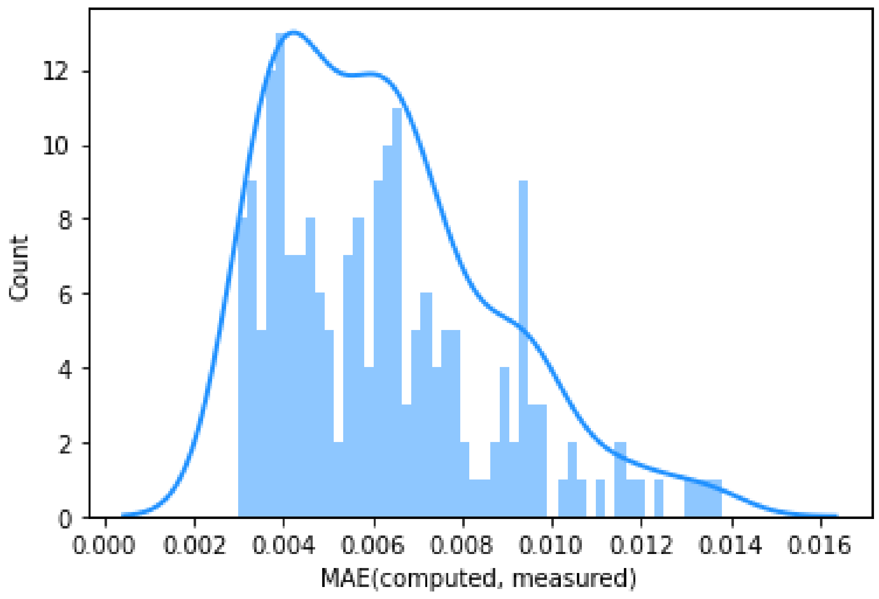


Figure 3.7: Distribution of MAE between computed and measured spectra

separate reference spectra for each of the five sample positions to ensure that position specific noise is reflected in the reference spectra as well.

3.6 Preprocessing

As outlined by Gholizadeh et al. in [44], the major categories of preprocessing and treatment of spectral data include *Baseline Removal, Scatter Correction* and *Smoothing*. These techniques are applied to remove some of the instrument and measurement errors that accrue in the course of the spectra acquisition.

3.6.1 Baseline Removal

Background noise during measurements introduces a background signal that pushes the final output signal away from zero (the base). Due to the undesirable effect this has on the peaks of the signal, it is important to remove this signal to obtain the correct data with peaks in tact.

Different techniques have been proposed to deal with baseline removal [45] and in this work, we use a method introduced in [46] which uses a modified least-squares polynomial curve fitting called *ModFit*, for the estimation of the baseline.

Different degrees of polynomials can be used for the fitting and this is supplied as a hyperparameter to the preprocessing pipeline.

3.6.2 Scatter Correction

According to [47], the shape of a spectrum is affected by these factors:

1. Absorption by the sample is different across different wavelengths of the incident light. This is essentially the signal we want to measure.
2. Particle sizes of the sample material vary and will cause a reflection of light at different angles across different wavelengths. This causes an undesirable *scattering* effect which we want to get rid of. The scattering could be both additive and multiplicative.
3. There arise differences in optical path length between the different samples caused by irregularities in the positioning of the sample and the sample surface itself. This also introduces deviations in the final signal.

The different methods of scatter correction seek to remove these undesirable scattering effects introduced into the signal. The common methods of achieving this are: *Multiplicative Scatter Correction (MSC)* and *Standard Normal Variate (SNV)*. MSC approaches the correction of a spectrum x by performing a regression to solve the following equation:

$$x = a + b\hat{x} \tag{3.7}$$

where $\hat{x}$ is a reference spectrum without scattering effects and a and b are the regression coefficients to be computed. Since it is difficult to obtain a good reference spectrum without scattering effects, the average of all the spectra is often used as the reference.

SNV performs the correction simply by centering and scaling each individual spectrum using their mean $\bar{x}$ and standard deviation σ.

$$x = \frac{x - \bar{x}}{\sigma} \tag{3.8}$$

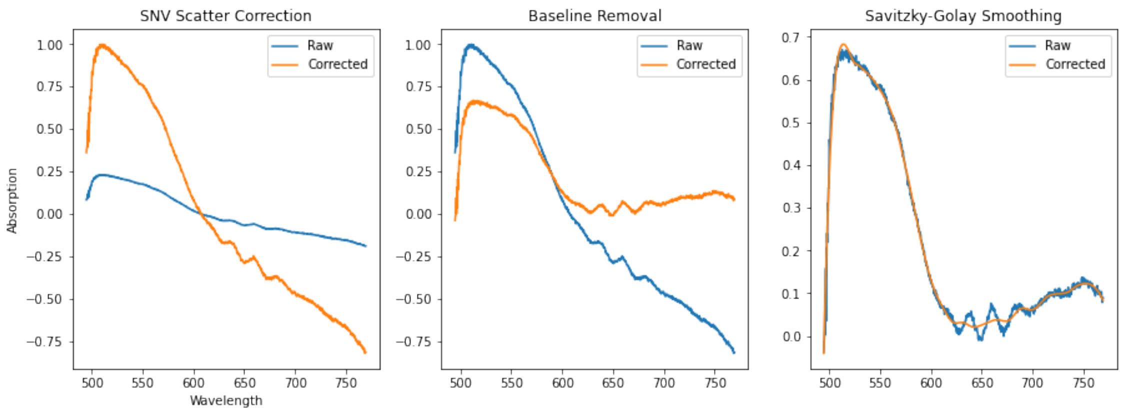


Figure 3.8: The effects of different stages of preprocessing

After the operation, each spectrum will have a mean of zero and a standard derivation of 1.

Both MSC and SNV perform well for scatter correction [47] and so we chose SNV first for its simplicity and secondly because it does not require a reference spectrum.

3.6.3 Smoothing

This category of preprocessing techniques is used to remove the random noise in the measurement that results in a noisy spectrum that is hard to work with. The commonly used techniques for spectral smoothing is the Savitzky-Golay Smoothing [48] which again fits a least squares polynomial to different segments of the spectrum, obtaining curves of best fit which are used to in of the original spectrum.

These preprocessing steps are applied to the spectral library so that it is not done during spectrum generation to reduce computation overhead. Figure 3.8 shows effects of the different stages of preprocessing on a sample spectrum. For demonstration, the preprocessing steps in Figure 3.8 are done after generating the spectrum and converting to absorption as it more clearly shows the effects of the treatment.

3.7 Conclusion

In this chapter we have described in details the system we designed for the light filtering and tuning experiment. The problem statement and analytical task were defined and we looked at how the data will be collected in order to solve the task. We also discussed the preprocessing techniques employed to treat the collected spectra. In the next chapter, we will introduce the methodology we have chosen to solve the analytical task posed here and describe its implementation details.

Chapter 4

Methodology

In Chapter 2, we looked at some of the learning algorithms used for analysis and processing of spectral data. We mentioned the Evolutionary Algorithms family and stated our choice of Genetic Algorithms, which is a member of this family, for our work. In this chapter, we will go into details about these evolution inspired algorithms and the specifics of our implementation.

4.1 Evolutionary Algorithms

Evolutionary Algorithm (EA) is a population-based optimization algorithm inspired by the natural evolution process. In EA, each individual is a solution to the underlying task. Each solution is gradually optimized by the evolutionary processes of natural selection mutation, and crossover. As already mentioned earlier, a key advantage of EA is the absence of assumptions about the problem landscape, such as is assumed by some other optimization algorithms and this makes it a global optimization method. Another advantage as cited by Xu in [49], is that having a population of candidate solutions in this family of search algorithms encourages an early exploration of different regions in the decision space, which helps to avoid getting trapped in local optima. EA is a large family of algorithms with many variations of each algorithm. Some of the popular members of this family are Genetic Algorithms (GA), Genetic Programming (GP), Differential Evolution (DE), Evolution Strategy (ES) and Evolutionary Programming (EP). Slowik et al. in [50] explored the taxonomy of the EA family of algorithms and came up with the hierarchy in Figure 4.1. Since our solution is based on GA, we will focus on it in the following sections.

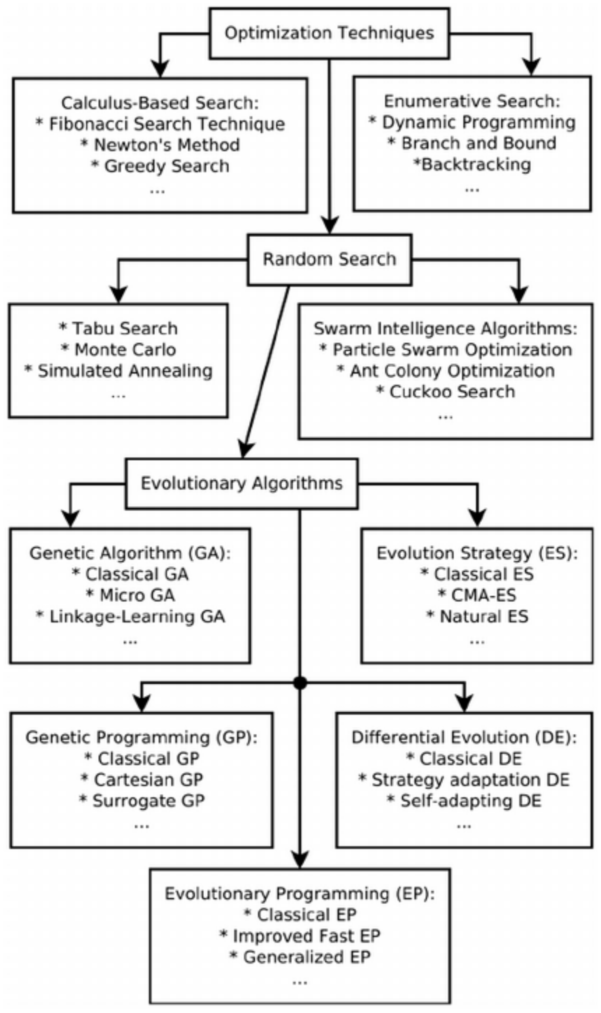


Figure 4.1: Taxonomy of nature-inspired optimization methods (source: [50])

4.2 Genetic Algorithms

The Genetic Algorithm is one of the most popular variants of EA. The individuals in the population here are represented by their genetic materials which are called chromosomes, analogous to their biological counterparts. A biological chromosome consists of genes made up of DNA. Each gene encodes specific traits such as eye color, hair color, etc. Likewise, the chromosomes in GA are used to encode the parts of the solution. In a traditional GA, the solution is encoded as a binary string, so that each gene is a bit with values of 1 or 0. Figure 4.2 shows a representation using binary encoding for the chromosomes. This is suitable for solving discrete problems, however, continuous problems can employ an encoding allowing integers and floating point numbers. The implementation we use in this project uses integer and floating point genes for the chromosome encoding. There are 5 main steps involved in the algorithm, namely: initialization, fitness assignment, selection, crossover and mutation.

4.2.1 Initialization

The first step is to define and initialize the population. The population size is usually a hyperparameter supplied to the algorithm. In this work, it is typically in the range 100 - 300. A chromosome is created for each member of the population with each gene given an initial value. The initialization procedure varies based on the encoding process. Of course, the constraints of the underlying problem also has to be taken into account so that the initial values lie in a valid range. In this work, the value of each gene represents the angle being predicted which we earlier on established to be in the range $0° - 180°$.

Studies have shown that the initialization of a GA greatly affects the convergence. Common techniques used include *Random Initialization* and *Heuristic Initialization*. With random initialization, the population is diverse, which as we observed in § 4.1, is a good advantage that allows early exploration of large portions of the decision space. However, this could also lead to a population with very poor fitness and may take more time to converge. On the other hand, the initialization can be done with a heuristic exploiting the nature of the underlying problem. This could speed up the convergence but presents a risk of reduced diversity in the population and could even lead to premature convergence where the solutions get stuck in local optima, as observed by Goldberg in [51]. We use random initialization for its simplicity, sampling the values from a uniform distribution in the specified range.

4.2.2 Fitness Assignment

This is another critical step in the algorithm. Having initialized the chromosomes of the individuals, the the next step is to access their fitness. The fitness function should take the chromosome of an individual in the population and return a real-valued fitness usually computed from the values of the genes. Since the mutation,

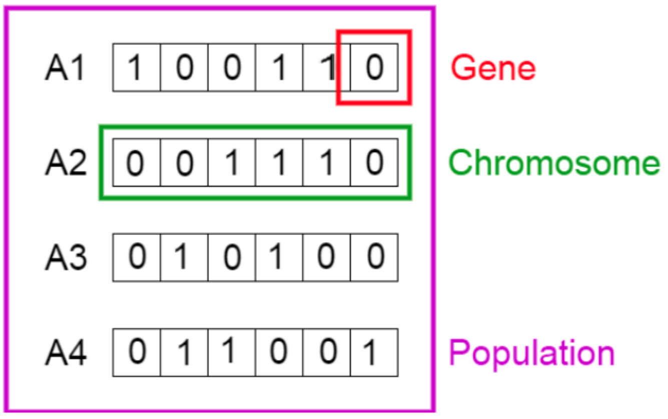


Figure 4.2: Representation of individuals in a GA (source: Analytics Vidhya Blog)

selection and crossover steps heavily rely on the fitness of the individuals, it is very important to select a good fitness function that best suits the task at hand. Additionally, the computation of the function should also be very fast as it could quickly become a bottleneck and greatly slow down the algorithm.

The function is inversely related to loss functions in Machine Learning algorithms. In traditional implementations, the algorithm tries to select the best solutions by maximizing the fitness. In the implementation we use, the fitness is represented by a loss which the algorithm tries to minimize, hence, from here on, we will refer to the fitness function and loss function interchangeably.

In typical regression tasks, loss functions like the Absolute Loss or Mean Squared Error are commonly used but these functions have their downsides. We will discuss the loss functions we use in more details in § 4.3.

4.2.3 Selection

The next step in the algorithm is the parent selection. This selects the parents who can *mate* to produce offspring for the next generation. This again closely resembles the process of natural selection, only this time, there is a very limited number of individuals who can be selected as parents and it is controlled by a fixed ratio. This ratio is usually supplied as a hyperparameter to the algorithm. In nature, selection occurs through *survival of the fittest* as enshrined in Darwinian evolutionary theory. However, this is not exact as a few less fit individuals survive too and some more fit ones do not. This encourages diversity in the population and this also transfers to Genetic Algorithms. If the top fit individuals are simply selected, rendering the population less diverse, the algorithm will not explore diverse parts of the decision space as mentioned in § 4.2.1. An elitist variant of GA, which we use, selects a small proportion of parents greedily from the individuals purely based on their fitness, then uses a selection method for the rest of the parents. This ensures that if the selection method does not select good individuals, there will still be a few fit parents for reproduction. Different methods for the parent selection have been proposed over the years as this could greatly affect the performance of the algorithm. We explore some of these in the next few sections.

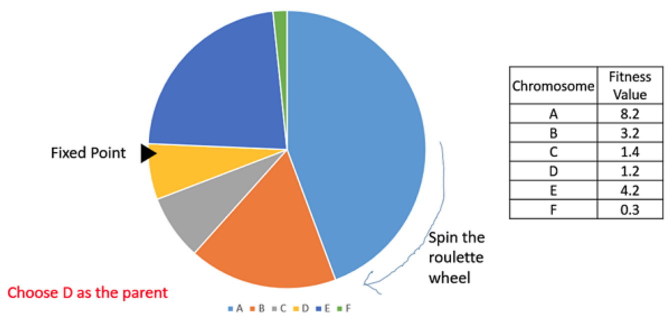
4.2.3.1 The Roulette Wheel Selection (RWS)

This is one of the most common methods and the one used in our implementation. The major characteristic of this selection method is that each individual gets the chance of being selected, proportional to their fitness [52]. For each individual i in a population with size n and a fitness function f, the probability of getting selected p_i, can be expressed as:

$$p_i = \frac{f(i)}{\sum_{j=1}^{n} f(j)} \tag{4.1}$$

To make the selection, the cumulative probability distribution is computed and at each selection step, a random number is sampled and the individual whose probability lies in the region of the random number is selected. It works very much like a physical roulette wheel as shown in Figure 4.3. Jebari et al. in [52] note that there is still a risk of premature convergence due to possibility of a dominant individual getting selected most of the times, thereby reducing diversity and leading to local optimum.

Many variants of the RWS have been proposed to solve this issue, such as the Stochastic Universal Sampling (SUS) which tries to select all individuals in the same step and the Linear Rank Selection (LRS) which uses a rank of the individuals derived from their fitness instead of the fitness directly.



Chromosome	Fitness Value
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

Figure 4.3: Roullete Wheel Selection (source: Tutorialspoint)

4.2.3.2 Tournament Selection

In Tournament selection, a number of individuals are first selected at random irrespective of their fitness. Then the selected individuals are then ranked with the fittest individual selected to be added to the final parent list. This is repeated in each step till the parent list is complete. The diversity of this approach is derived in the stochastic first stage selection where individuals are selected at random. The second stage then ensures that some level of fitness is maintained to avoid poor solutions that do not converge. There is a similar method named Random selection, the difference being that it only involves the initial selection stage so that the individual is totally selected at random. This is a naive method which though it encourages diversity and exploration, does not encourage fitness and exploitation and hence will often lead to poor solutions.

4.2.3.3 Truncation Selection

This is a highly exploitative approach where the individuals are ranked and the parent list is simply selected from the top individuals. This is the opposite of the random approach and can produce solutions with very poor diversity, leading to premature convergence.

4.2.4 Crossover

Crossover emulates the natural reproduction process. Two parents are chosen at random at each step and points are established on their chromosomes where swapping of genes occur. Not all selected parents eventually pass their genes to the next generation because there is a crossover probability which is a hyperparameter usually around 0.5. With this probability, the number of parents to cross over are chosen. Additionally, at each crossover iteration, two parents within the already reduced parent list are selected at random. This means that some parents in the list may never get selected. This whole arrangement increases the stochasticity of the system.

There are different methods of choosing the points where gene swapping occur as this affects the result of the search algorithm. We discuss the common ones below.

4.2.4.1 One Point Crossover (OPC)

In OPC, a single point on the chromosome is chosen at random, then all genes to the right of that point are swapped for both parents to obtain two new offspring. Using this method in our angle prediction task, lets say the first parent encodes a solution [10, 20, 30, 40, 50], while the second parent encodes a solution [5, 15, 25, 35, 45] and let the randomly chosen crossover point be position 2. Then the result of the crossover would be the children with chromosomes: [10, 20, 25, 35, 45] and [5, 15, 30, 40, 50]. The motivation is that both parents can pass favorable genes or parts of their solution to the children. There is of course a possibility for them to pass their worse genes to the offspring but then these offspring will likely be filtered out in the next generation if they end up poorly fit.

4.2.4.2 Two Point Crossover (TPC)

This is similar to OPC but allows the swapping to happen in a range between the two chosen points instead of taking place from one point to the end. Using the same example in OPC and a second point chosen at position 4, the two offspring would be: [10, 20, 25, 35, 50] and [5, 15, 30, 40, 45].

4.2.4.3 Uniform Crossover

Uniform crossover allows for crossover to take place uniformly along the length of the chromosome. For each gene position in the chromosomes, a coin flip is used to decide whether the gene will be swapped between parents or kept as is.

4.2.5 Mutation

In natural evolution, offspring often develop new changes called *mutations* in their DNA which were not inherited from parents. This usually occurs as a result of environmental factors or errors during replication but also aids to keep the population genetically diverse. Analogous to this, an important step of GA is the mutation of individuals in every generation. The mutation only occurs with a little probability and should affect only very few individuals in the population. Care has to be taken on the selection of the mutation probability as high values lead to random search [53] and low values lead to less diversity in the population.

Many methods of performing mutations have been studied, a few of which we will cover here.

4.2.5.1 Random Resetting

In this mutation operator, with a small probability p, each gene in a chromosome is removed and replaced with a random gene from the gene space. In the setting of this project, the gene angle would be replaced with a random angle in the range of $0° - 180°$, if a randomly sampled number is less than the mutation probability p. This is the mutation operator we employ.

4.2.5.2 Swap Mutation

Swap mutation selects two random genes in the chromosome and swaps their values, also with probability p.

4.2.5.3 Scramble Mutation

This operator selects a range of genes in the chromosome and shuffles their values. The range selection is done for every gene position in the chromosome with probability p.

4.2.5.4 Inversion Mutation

Inversion mutation works similar to scramble mutation. They both select a range of the chromosome but this time, the selected range is inverted.

4.3 Loss Functions

Loss functions are central to any learning algorithm as they present the objective of learning. As already mentioned, tasks involving the prediction of real-valued dependent variables often employ MSE or other closely related loss functions. MSE is known to be susceptible to outliers and modifications like the *Huber loss* have been proposed to mitigate this.

However, all these loss functions mostly focus on the Euclidean distance between the true and predicted values. In this work, an even more important property is the shape of the spectra produced. For approximating a target spectrum t, we may be able to match closely the shape but not the same intensity due to the beam not having enough initial intensity. Another reason could be due to the spectrum not being in the same exact range even after normalization. But this is not an issue as the intensity of the beam source can simply be adjusted later on. So it is much more important to match the shape of the target spectrum than matching the absolute values of the peaks.

Using the spectral library data discussed in § 3.5.1, the loss function can access not only the predicted angles but also the constructed spectrum from the angles. Hence, it should optimize for the shape of the constructed spectra and not simply the Euclidean distance between the points. This is where MSE and related loss functions fall short.

A metric has been developed to solve a similar issue for temporal sequences, called *Dynamic Time Warping (DTW)*. DTW is an algorithm developed by *Richard E. Bellman* [54] back in the late 1950s which has been extensively applied to areas such as speech recognition, time series, human pose tracking among others [55]. The algorithm creates a mapping between two temporal sequences by minimizing the warping distance between them. The said warping distance gives a similarity measure between the two signals with much better emphasis on shape compared to simple Euclidean metrics.

However, the warping distance computed by DTW does not exactly fit into our use case because two signals with the same shape but slightly shifted in time (or along the wavelength in our case) will result in a high similarity by DTW and this is undesirable. So, we do not just desire a similarity in shape but in alignment as

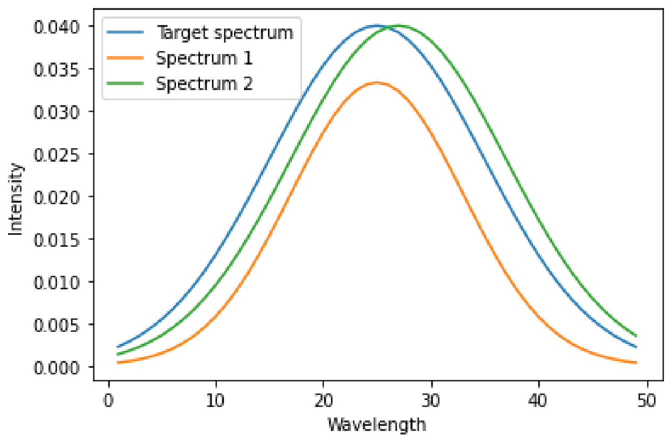


Figure 4.4: Comparison of spectra by shape and alignment

well. Consider the spectra represented by Gaussian distributions in Figure 4.4. By Euclidean metrics, *Spectrum 1* best matches the target spectrum and yet *Spectrum 2* is a better match as they have very similar shapes and are aligned. While *Spectrum 1* also has a similar shape, it is not aligned. The MSE between *Spectrum 1* and the target spectrum is 4.8×10^{-5} while that of *Spectrum 2* is 1.1×10^{-5}. Similarly, DTW gives a distance of 0.08 for *Spectrum 1* and 0.003 for *Spectrum 2*.

Using any of these distance measures in our loss function will make our GA prefer solutions that produce *Spectrum 2* over those that produce *Spectrum 1*. Hence, there is need to develop another distance metric with more emphasis on shape and alignment.

4.3.1 Shape Loss

We developed a simple metric which we call *Shape Loss* to measure the shape dissimilarity between two spectra. The central idea is that the shapes of two curves are similar if they maintain a near constant distance between each other at all points. If the mean distance between the curves is d, then large deviations from d will denote large dissimilarity while small deviations from d will denote small dissimilarity. We can gauge the amount of deviations from the mean distance d using the standard deviation of the distance between the two curves. The algorithm is quite simple and can be summarized in two steps. Given two spectra x and y:

1. compute the residuals between x and y: $d = x - y$

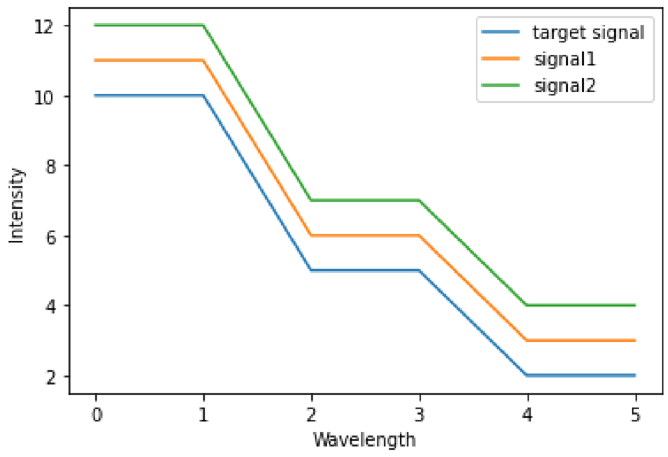


Figure 4.5: Aligned signals at different intensities

2. find the standard deviation of the residuals:

$$SL = \sqrt{\frac{\sum_{i=1}^{n} |d_i - \bar{d}|^2}{n}} \tag{4.2}$$

   where n is the length of d and $\bar{d}$ is the mean of d. SL is the shape loss.

In principle, other distance measures can be used to compute the residuals in step 1, but it is important to keep the signs instead of an absolute value so that the alignment reflects in the calculated shape loss.

In the Gaussian spectra in Figure 4.4, the proposed shape loss gave a distance of 0.002 for *Spectrum 1* and 0.003 for *Spectrum 2*. With this, we have a better objective for the GA so that the predicted spectrum can better fit the shape and alignment of the target spectrum.

Shape Loss is simple with a linear time complexity which again suits the GA ecosystem which requires fast fitness functions for speed. In comparison, DTW has a quadratic complexity with some slightly faster implementations. One issue with Shape Loss is that it focuses entirely on the shape and does not take into account the relative distance between the signals.

Consider the signals in Figure 4.5. Both signals are perfectly aligned with the target signal but *signal1* is closer in intensity to the target signal. Their shape losses are both 0 but we would prefer *signal1* in this case. So, we have to somehow account for the difference in intensities in Shape Loss as well.

We therefore need to adjust the Shape Loss to incorporate the intensity differences. One way to do that could be to compute the MSE or other Euclidean distance as well and at the moment of comparison of the individual in GA, we first compare the Shape Loss and if it is equal, we then compare the distance measure. While this is simple, it will require restructuring the GA a bit.

A second option is to sum Shape Loss and the distance metric. While easier, the summation needs to be done in such a way that more emphasis still lies on the Shape Loss. We employed this approach and made modifications to the Shape Loss computation as follows:

1. Compute the Shape Loss SL as described above.
2. Compute the MAE between the signals.
3. Compute the maximum intensity of both signals m.
4. Scale both the SL and MAE by dividing by m.
5. The final loss is computed as follows:

$$loss = \alpha SL + (1 - \alpha)MAE \tag{4.3}$$

Where alpha is a control parameter for the Shape Loss with values between 0 and 1 which controls the amount of attention given to each component of the loss.

Using this modified version with very high α will result in a soft approximation of the first suggestion for double comparison. With an α of 0.9, the loss for the signals in Figure 4.5 are 0.009 for and 0.016 for *signal1* and *signal2* respectively. For the spectra in Figure 4.4, it is 0.065 for *Spectrum 1* and 0.083 for *Spectrum 2*.

More comprehensive comparisons of the loss functions we have discussed here will be made in the next chapter.

4.4 Implementation Details

For our experiments, we use a GA implementation based on the open source Python library *geneticalgorithm*(`https://github.com/rmsolgi/geneticalgorithm`).

The popular Python data analysis libraries *Numpy, Scipy, Maplotlib* and *Scikit-Learn* are also extensively used.

A preprocessing pipeline was built to treat the collected spectra using the preprocessing techniques discussed in § 3.6. The next step is the GA configuration.

4.4.1 GA Chromosome Encoding

The chromosome of our GA population contains 10 genes: 5 for the angles and 5 binary genes. The binary genes act as switches that predict if a component of the spectral library should be added to the computation or omitted. Consider a target spectrum T that is proportional to the first component of the spectrum X_1 at an angle α_1. Since at any given angle, the absorption value of each of the other components is not zero, then we cannot accurately approximate T by just predicting the angles. In the physical system, a component can be switched off by blocking the light from getting to the optic fibres. We thus simulate this component switching mechanism into our algorithm using the binary flags. In the scenario above, the GA could then predict $[\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5]$ for the first five genes and [1, 0, 0, 0, 0] for the last five genes. The values of $\alpha_2 - \alpha_5$ do not matter in this case as the spectrum simulator will exclude them from the computation.

4.4.2 Spectrum Simulation

As described in the previous chapter, we built a spectral library to easily generate the spectrum for predicted angles. A component of our analytical system which we call the spectrum simulator takes care of simulating the spectrum given the angles and the component switching flags. The work of the simulator can be summarized as follows:

1. For each angle α_i, lookup the corresponding base spectrum X_i from the spectral library.

2. Compute the spectrum s using the binary component switches $w_1 ... w_5$ as weights.

$$s = \sum_{i=1}^{5} w_i X_i \tag{4.4}$$

3. Convert s to an absorption spectrum using Equation 3.4.

During fitness assignment at each generation of the GA, the loss function gets the chromosome representation for each individual in the population, generates the spectrum using the simulator, then computes the loss between the target spectrum and predicted spectrum s. This loss is used as the measure of fitness.

Below, we show examples of applying the system for approximation of a spectrum drawn from a dataset created from the spectral library and another spectrum not taken from the experiment.

4.4.3 Approximating a Spectrum from Experiment

Here we sample a spectrum generated from the spectral library to test the working of the GA. We initialize the GA with these control parameters:

- Number of generations: 500
- Population size: 200
- Early stopping: 100 generations without improvement
- Mutation Probability: 0.1
- Elite ratio: 0.01
- Parents ratio: 0.3

With the large population size and number of generations, the algorithm ran for 22 seconds. Figure 4.6 shows the loss through generations of the search. The algorithm stops after about 200 generations or iterations due to the early stopping criterion. The search was done with Shape Loss which achieved a minimum loss of 0.13. It is possible that allowing a higher number of generations without the early stopping condition will eventually result in finding the the very angles that were used to generate the target spectrum but the approximation is already good enough

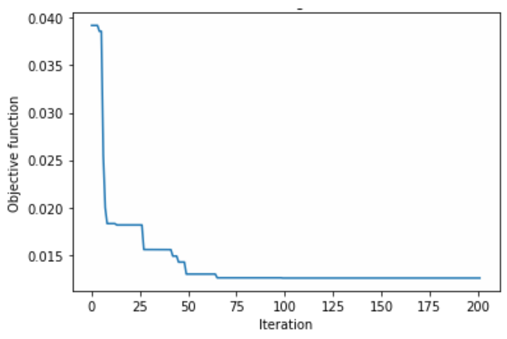


Figure 4.6: Loss function

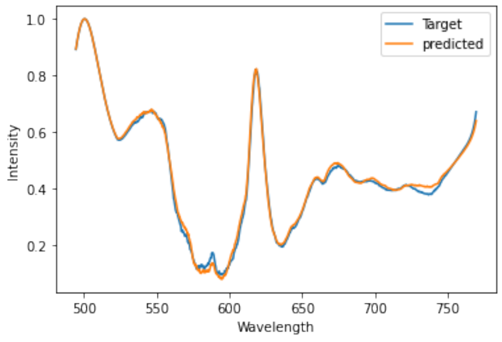


Figure 4.7: Prediction

as can be seen in Figure 4.7. Searching for the global optimum could prove time and resource consuming. The angles used to generate the target spectra are [163.5 108.5, 112.1, 111.5, 17.4] while the search predicted [46.4, 119.7, 116.8, 14.9, 77.9], including all components in the generation. This is inline with the discussion in § 3.5.1, where we asserted that the optimal combination of angles to approximate a given target spectrum is not necessarily unique and hence it is important to design the loss against the predicted spectrum instead of the predicted angles.

4.4.4 Approximating other spectra

Approximating a spectrum generated from the experimental data is a good gauge of the algorithm but the ultimate goal is to approximate other spectra of interest from outside the experiment. Here we have to ensure that the target spectrum fits the wavelengths that are covered in our experiment. We do this by interpolating the target spectrum values for the 1000 wavelength points used in the experiment. The next step is to normalize the intensity of the spectrum to be in the range 0 - 1 to also be in the same intensity range as the experiment data.

The spectral data that we use for fitting are absorption spectra, so if the target spectrum is not an absorption spectrum but an emission spectrum, then it needs to be converted to an absorption spectrum by subtracting from one. This is after normalizing to 0 - 1. Using the system as a filtering system as already described in previous chapters, the task would be to fit an absorption spectrum that will result in the final transmitted spectrum captured in the spectrometer fitting the target emission spectrum.

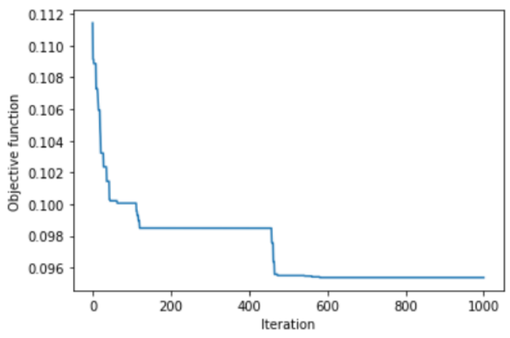


Figure 4.8: Loss function

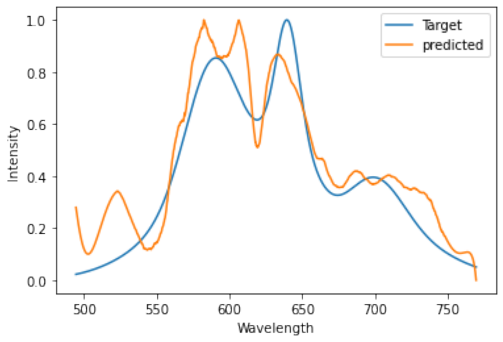


Figure 4.9: Prediction

The spectra that can be approximated also need to have similar peaks to the spectra of the samples we used. As an example here, we use a phosphor type LED emission spectrum. First we normalize and convert to absorption spectrum. After fitting, we convert both the target and predicted spectra back from absorption to emission and transmission spectra. Figure 4.9 shows a comparison of the final spectra after all the conversions.

The GA control parameters are the same as those used previously in § 4.4.3 only the number of generations is increased to 1000 and the early stopping criterion was removed. This is because approximating an external spectrum is more difficult and so the algorithm is given more time to search. As can be expected, the result is not as good as fitting a spectrum generated from the same pool as there is no guarantee that a good fit exists for it. However, the peaks and general trend of the spectrum are roughly approximated and the light generated should be similar as well.

4.5 Conclusion

In this chapter, we explored the details of genetic algorithms and their implementation. We also discussed the specifics of our GA implementation and how it is used in our system for the approximation of a given target spectrum. In the next chapter, we will cover an extensive evaluation of the system and show results.

Chapter 5

Evaluation and Results

In this chapter, we go into detailed evaluation of the analytical system built in the previous chapter. First, we perform an internal evaluation on the data generated from the experiments, then we evaluate external spectral data.

The evaluation process relies heavily on the work of Kevin A.G Smet [56] in which they developed an open source Python toolbox for lighting and color science called *LuxPy*[1]. LuxPy supports several common lighting and colorimetric calculations which we use in this section.

A key function of the package is the simulation of different light sources including monochromatic lights based on the work of Yoshiro Ohno [57] and phosphor LED-types based on Smet et al. in [58]. It was used to simulate the light spectrum in § 4.4.4 which was a LED-type with two phosphores.

5.1 Light output evaluation

In this section we evaluate the output of the approximated light source using our filter system by comparing it with the target light source in color space. First, we simulate a LED light target spectrum using LuxPy with two phosphores having their peaks in the yellow (between green and red) region of the visible spectrum. Then we construct a GA to approximate the spectrum using our filter system with the control parameters in Table 5.1.

The result is shown in Figure 5.1 with a Shape Loss of 0.151. The approximation is a rough one and it is not obvious how close the actual light output will be. One way to compare light outputs is to compare their colors. So we convert the light

[1] https://github.com/ksmet1977/luxpy

Generations	400
Population size	500
Early Stopping	100
Mutation probability	0.1
Crossover probability	0.5
Parents portion	0.3

Table 5.1: GA control parameters

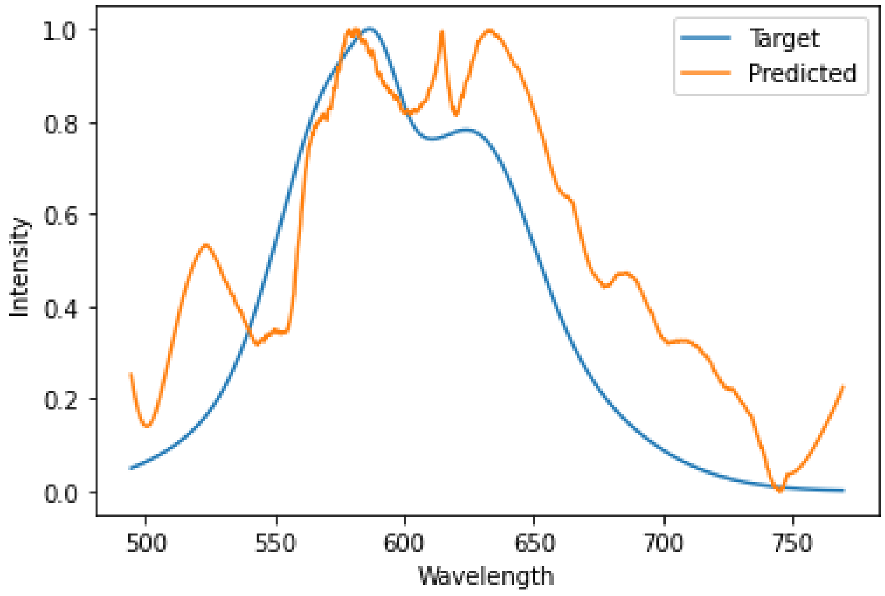


Figure 5.1: Simulated target light spectrum vs approximated light spectrum

spectra to the CIE 1931 XYZ color space and compute their distance apart using the CIEDE2000 [59] colour difference metric.

We use colorimetric tools in LuxPy for the conversion and distance computations and obtain a distance of 1.85. For an interpretation of this number, we use the observation of Béla Török in [60] which outlined the following thresholds for the colour CIE **Δ**E metrics:

- 0 - 0.5: No color difference.
- 0.5 - 1.0: Difference only perceivable for experienced observers.
- 1.0 - 2.0: Minimal color difference.
- 2.0 - 4.0: Perceivable color difference.

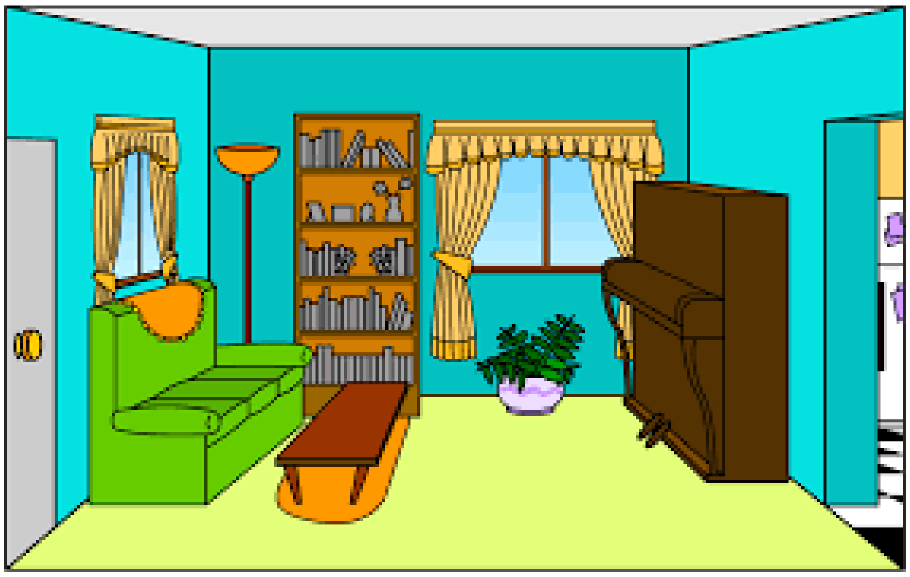


Figure 5.2: Original image in white light

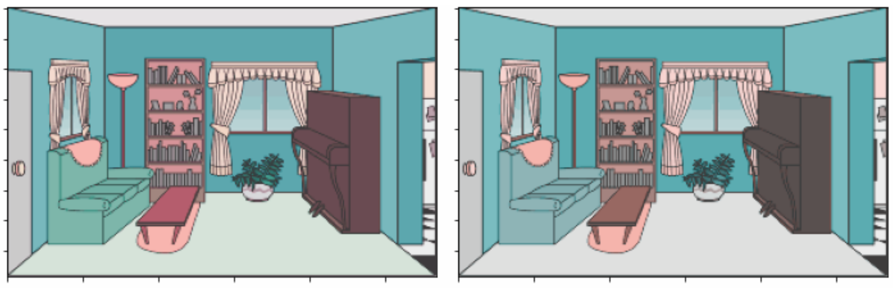


Figure 5.3: Left: Rendered with the predicted light; Right: Rendered with the target light

- 4.0 - 5.0: Significant color difference.
- > 5.0: Different colors

Using the thresholds above, the color difference of the target light and approximated light is minimal. To further investigate this, we use a rendering tool provided by LuxPy to render the colors of both lights on an image and compare their outputs.

Figure 5.2 shows the original image in white light. The result of the rendering is shown in Figure 5.3 with the predicted light on the left and the target light on the right. An observable difference exists but minimal. The predicted light tends to be brighter with more red components and this can be attributed to the predicted spectrum having more activity in the red region compared to the target light as can be observed in Figure 5.1. Nevertheless, the result agrees with the minimal color difference of 1.85 obtained above. We use the color difference metric for the rest of the evaluation.

The evaluation will be done in two main stages. The first stage is done on the internal data generated on the experiment to establish the best objective and control parameters and the second stage is on external data simulated with LuxPy.

5.2 Evaluation with Internal Data

In § 4.4.3, we showed the stages involved in approximating one of the absorption spectra generated from the spectral library. We use the same steps for the evaluation, only this time on 50 different such spectra. We study the effects of the different loss functions we already discussed as well as different control parameters of the genetic algorithm.

5.2.1 Comparison of Loss Functions

For the evaluation experiments we use MSE, Huber loss and the proposed Shape Loss as objectives for the GA. Since we established that the CIEDE2000 color difference metric is good enough to quantify the difference in the light as well, it is tempting to use it as a search objective as well. However, while light of very close spectra may have very similar colors, having similar colors does not translate to similarity of the light source due to color *metamerism*.

As described by Schmitt in [61], metamerism is a perception of different spectral power distributions as the same color. As already mentioned, the human eye contains cone cell receptors that absorb light and account for the color we observe. These eye cones respond to the cumulative energy along a wide range of wavelengths, making it possible for different combinations of light across these wavelengths to produce an equivalent receptor responses and the same color sensation. This is the cause of metametarism.

To demonstrate this, we created a search objective using DIEDE2000 and fed it to the GA to approximate a simulated target spectrum. The search ended with a DIEDE2000 loss of 0.058. Using the loss thresholds above, this should be interpreted as having no color difference. Even the color rendition on Figure 5.5 shows no

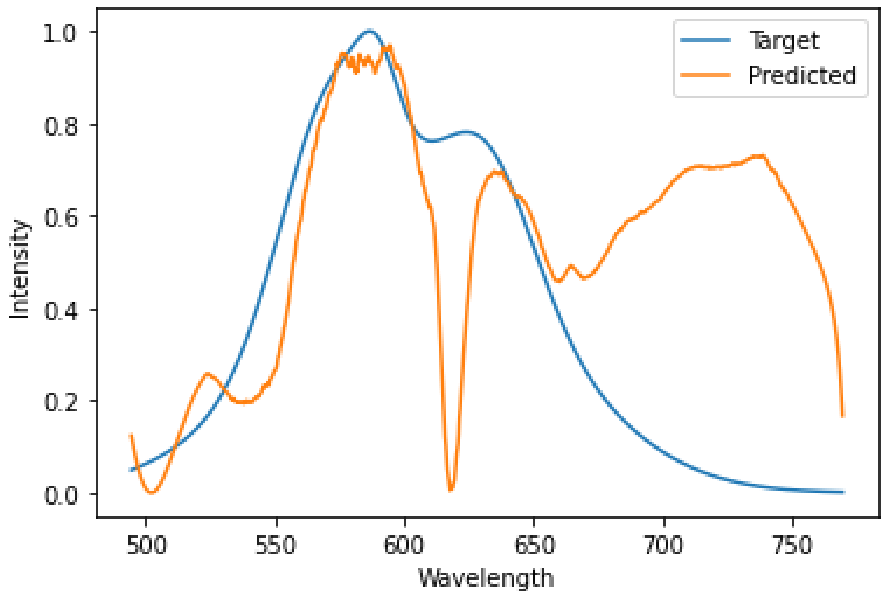


Figure 5.4: Simulated target light spectrum vs approximated light spectrum using CIEDE200 as objective

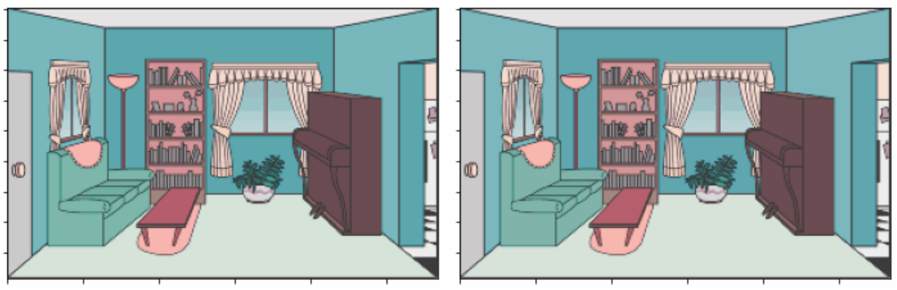


Figure 5.5: Left: Rendered with light predicted with CIEDE2000 objective; Right: Rendered with the target light

noticeable difference. However, the predicted spectrum is far from well fitted to the target spectrum as can be seen on Figure 5.4.

Hence if it is used as an objective, we could get light spectra that do not match the target but produce similar colors. Another reason why it will not be suitable is that it will try to match the absolute intensity as well which we have discussed before as not very necessary in our experiment. And yet a third reason is speed. An objective function based on the color difference metric needs to convert the spectrum to the XYZ color space first, then compute the CIEDE2000 distance both of which are not very fast algorithms and this will greatly slow down the GA search. For the test above, the GA ran for approximately 3 times the duration of a similar GA built on Shape Loss.

Huber loss is a hybrid of the squared loss and absolute loss which attempts to remedy the outlier sensitivity of squared loss by using the absolute loss when the difference exceeds a given threshold δ. Given two signals x and y and a threshold δ, Huber loss is computed piecewise as follows:

$$L_\delta(x,y) = \begin{cases} \frac{0.5\times(x-y)^2}{\delta} & \text{if } |x-y| < \delta \\ |x-y| - 0.5\times\delta & \text{otherwise} \end{cases} \tag{5.1}$$

Then we simply use the mean value of $L_\delta(x,y)$ as the loss.

Table 5.2 shows the control parameters used for the GA. Notice that the number of generations and early stopping criterion are smaller here compared to previous experiments due to the larger number of spectra. Consequently, the result will not be as good but just good enough to compare the different loss functions.

Generations	200
Population size	200
Early Stopping	50
Mutation probability	0.1
Crossover probability	0.5
Parents portion	0.3

Table 5.2: GA control parameters

Table 5.3 shows the results from the experiment. The GA trained with the Shape Loss objective achieved the best result in the CIEDE2000 color difference. We also

Objective	Shape Loss (mean)	CIEDE2000	Iterations (mean)	Running Time (minutes)
Shape Loss ($\alpha = 0.8$)	**0.0113**	**0.232**	152	15
MSE	0.0134	0.358	139	13
Huber loss ($\beta = 0.5$)	0.0126	0.324	165	15

Table 5.3: Result of different GA objectives

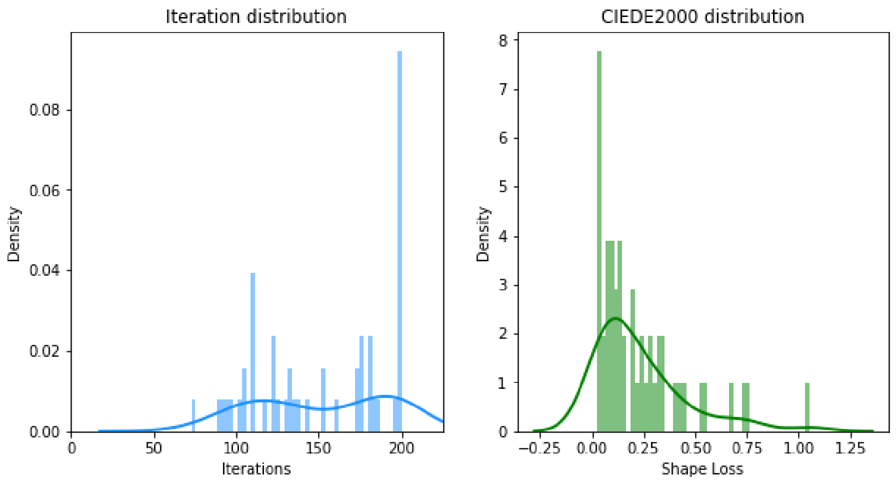


Figure 5.6: Distribution of color distance and iterations with Shape Loss objective

included the Shape Loss as an evaluation metric since we have established that it fits well into the optimization objective. It is however, no surprise that the GA with Shape Loss achieved a better Shape Loss compared to the other objectives. The results also shows that the Huber loss based GA performed better than the MSE. Hoewever, MSE searches faster than both but hit the early stopping condition faster as well resulting in a lesser number of average iterations. The maximum number of generations is 200 with an early stopping criterion of 50 iterations. Stopping earlier means the algorithm is no longer able to find better solutions than it already has. This could be interpreted as both a good feature in terms of speed and a bad feature in terms of getting stuck and unable to explore better parts of the search space.

Generally, the CIEDE2000 performance of all objectives were good given that the internal data was easily approximated and hence the distance still lies in the zone of no difference in colors. As the table only shows point estimates, we show distributions of the iterations and the color distance as well.

The iteration distributions show that the search kept improving till the end of the 200 generations most of the time for all objectives but more so for the Huber loss. From this we can conclude that extending the number of generations will likely improve the results at the expense of time and resources.

Given the result, we use Shape Loss as the objective for all experiments going forward.

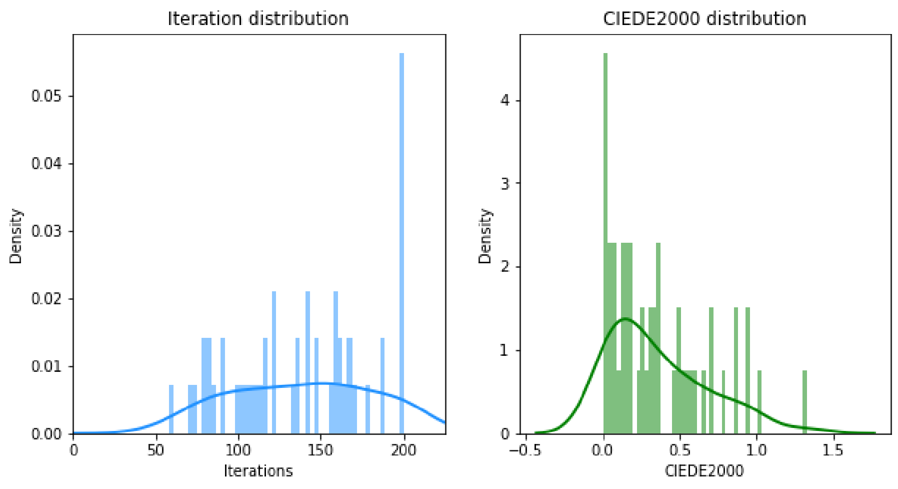


Figure 5.7: Distribution of color distance and iterations with MSE objective

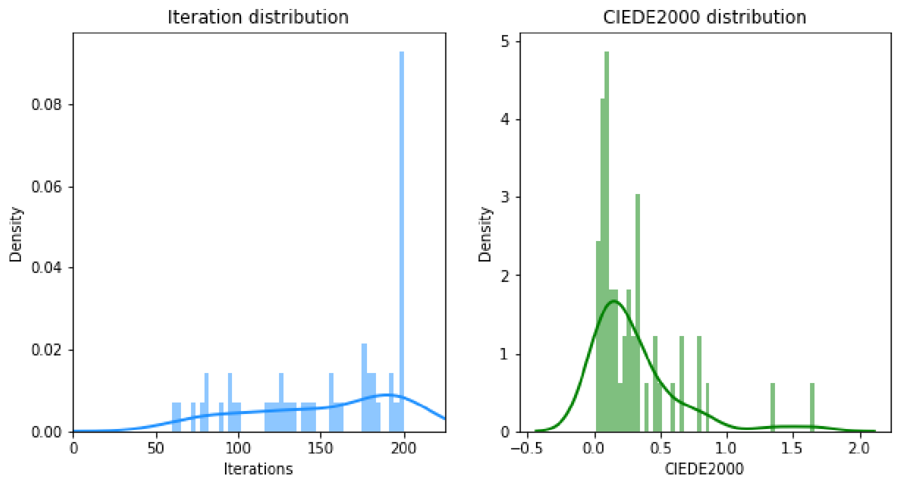


Figure 5.8: Distribution of color distance and iterations with Huber loss objective

5.2.2 Comparison of GA control parameter values

Genetic algorithms have many control parameters which we already discussed. In the previous experiment, we used 200 generations and 200 iterations with the internal data generated from the experiments. As already noted, the performance on the internal data from the same distribution is much better than the performance on external data. This means we have to tune up values of the search parameters to encourage more exploration of the search space. However, will increasing the number of generations result in a better performance than increasing the population

Parameter	Shape Loss (mean)	CIEDE2000	Running Time (minutes)
Generation	0.0113	0.263	16
Population	**0.0106**	**0.223**	24

Table 5.4: Result of increasing the number of generations vs population size

size? We conduct an experiment once more on the internal data in this section to answer that.

The GA control parameters here are exactly the same as in Table 5.2, except that in one experiment, we increase the number of generations to 300 while in the second experiment, the population gets increased to 300. We use Shape Loss with $\alpha = 0.8$ as the GA objective and we measure both the Shape Loss and CIEDE2000 of the predictions.

Table 5.4 shows the result of the experiments. It shows that increasing the population to 300 performs better than increasing the number of generations to 300 while taking more time to run. Intuitively, the experiment can be thought of as a search depth vs breadth comparison. Increasing the number of generations increases the depth of the exploration of the search space while increasing the population size increases the search breadth.

Another point to note is that the generation increment does not show a significant improvement over the result in Table 5.3, however, the increase in population does. Given the highly stochastic nature of the GA algorithm, this is not guaranteed to always be the case but a good pointer to the tweaks that could result in a better performance. The performance improvement from both parameters likely taper off at some point as their values are increased. Figure 5.9 shows the CIEDE2000 distributions for the experiment.

From this experiment, we conclude that the population size should be increased more than the number of generations, to cater for the difficulty of approximating external data in the next section.

5.3 Evaluation with External Data

Phosphore-based LED lights often incorporate phosphore of different colors to produce different kinds of light with varying range of colors. We developed a dataset

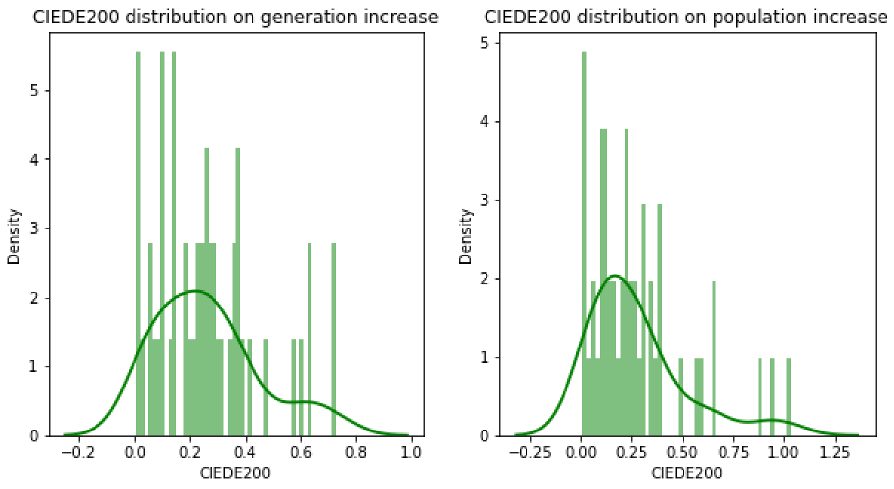


Figure 5.9: Color distance distributions for control parameters experiment

around this concept to evaluate the performance of the algorithm on different number of phosphores. As LuxPy allows only 1 or 2 phosphores, we only created datasets for one-phosphore light spectra and two-phosphore light spectra. Each dataset contains 50 different light spectra with phosphore peaks randomly sampled from a range of the visible spectrum.

Due to the relative difficulty of working with other spectra not from the experiment, we increased the number of generations and population size to improve the search performance. Based on the result of evaluation on the GA control parameters, the population size was increased more than the number of generations. The early stopping condition was also doubled relative to the previous section to encourage longer search before termination. Table 5.5 shows the values of the control parameters used for this experiment.

Generations	300
Population size	400
Early Stopping	100
Mutation probability	0.1
Crossover probability	0.5
Parents portion	0.3

Table 5.5: GA control parameters

The results are documented in Table 5.6. The first thing to notice is the relative

Dataset	Shape Loss (mean)	CIEDE2000	Iterations (mean)	Running Time (minutes)
One-Phosphore	0.151	3.801	224	34
Two-Phosphore	**0.129**	**2.467**	246	37

Table 5.6: Result on phosphore datasets

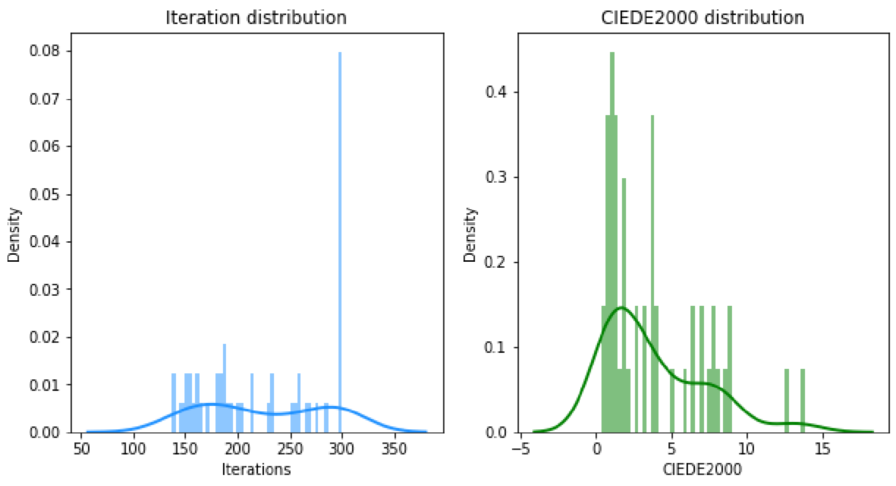


Figure 5.10: Distribution of color distance and iterations for One-Phosphore dataset

size of the metric values compared to the results on internal data in the previous section. This attests to the relative difficulty of approximation between the spectra generated from the experiment and external data.

The search algorithm achieved better performance on the two-phosphore dataset while running for longer. However, the CIEDE2000 color distance for both datasets lie in the same threshold of perceivable color difference. Depending on the application, the approximation may not be suitable but this can be improved by using a larger GA.

The distribution of the number of iterations shows that the search terminated much earlier in most of the cases for the one-phosphore dataset resulting in the faster search time. On the other hand, the loss spanned a wider range for this dataset with more outliers.

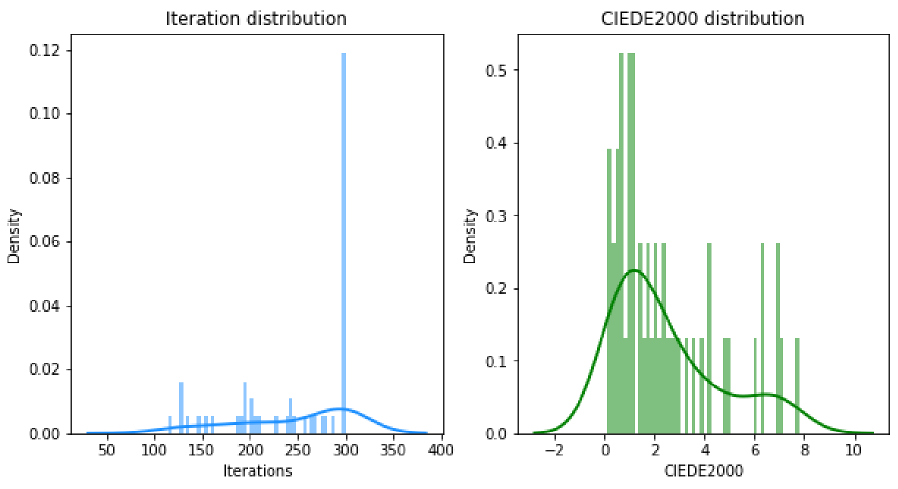


Figure 5.11: Distribution of color distance and iterations for Two-Phosphore dataset

5.4 Conclusion

In this section, we evaluated the light filtering system using the CIEDE2000 color difference metric as a guage on different datasets. The results show that approximating spectra from the experiment is much easier than spectra from other sources. The phosphore-type LED light simulated data was approximated with some perceivable difference but this can be improved by tuning the GA paramaters. Another interesting observation is that the proposed Shape Loss closely follows the trend of the CIEDE2000 in all experiments showing a strong correlation between the two. The next chapter discusses possible improvements to the projects, then brings a conclusion to the thesis.

Chapter 6

Conclusion

In this thesis, we designed a dynamic system for filtering light to obtain a desired light with a given spectrum. Typically for tuning lighting, the physical properties of the underlying fluorescent material is the focus to obtain the desired light. However, we took a hybrid approach in this work that incorporates the physics of the materials with data analysis for a more dynamic tuning. However, the properties of the materials still have a big influence on what kind of light spectrum can be approximated and how well that can be done. The results showed a sharp contrast between approximating the kind of light seen in the experiment and approximating other kinds of light. The spectral library proposed reduced the work of collecting data and improved the objective by allowing the optimization algorithm to focus on the difference between the light spectra instead of the angles.

Given the emphasis of most loss functions on the distance between the signals, we proposed a new loss that focuses on the shape and alignment of the signals and results showed that this achieved better results compared to the traditional regression loss functions. While there are no similar works to compare our system with, we created an evaluation procedure using a lighting toolbox, where the metric was based on the color difference between the produced lights of the target and predicted spectra.

6.1 Future Work

While the proposed loss function showed significant improvements over the traditional regression losses, the loss computation can be improved. The computation of the initial distance between the curves relies on the vertical distance between the individual points on the curve. Consider the signals on Figure 6.1. Even though the

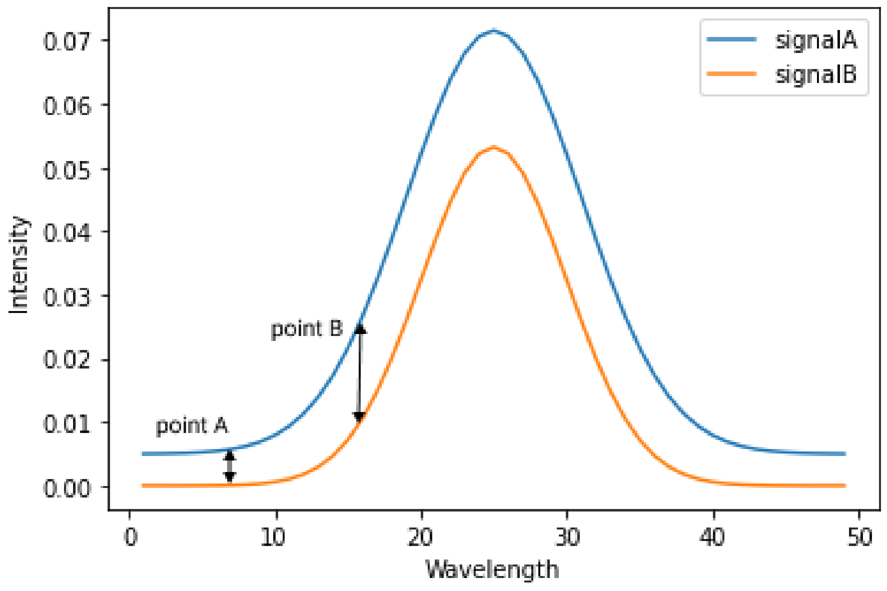


Figure 6.1: Vertical distance between curves

signals maintain a similar shape, the vertical distance between the points will show a strong deviation from each other due to the difference in gradients.

To take this into account into account, the signals can be divided into different segments and the area of each segment computed. The shape loss could then be computed as the standard deviation of the computed area for each segment. This is more computationally intensive and difficult to implement but should yield better results.

We primarily employed the Genetic Algorithm in the analytical part which proved good but other algorithms could be explored as well. Neural networks have become central to many regression problems and can be used here as well. We use the spectral library as an easy way to generate the spectra given any set of angles without the need of obtaining fresh measurements, but learning the spectral distribution with the aid of architectures such as Generative Adversarial Networks (GANs) could be explored. With enough spectral data fed into the GAN for training, conditioned on the angles, it could be used to generate a good approximation of a spectrum, given the angles. This then will eliminate the need for the spectral library albeit at a higher computational cost for the GAN training.

We relied on simulated data for the evaluation of the system in the absence of standard datasets. While this is a good gauge,the use of real life light spectra will give a better metric to assess the system for real applications. Getting such data is currently not easy from the best of our knowledge but can be attempted in future works.

In conclusion, this project serves as a proof of concept for a more dynamic way of light tuning by using the filtering and absorption system we have built. It can be adapted to any form of spectrum approximation outside the visible light range as well. With the rising need for tailoring light sources for different needs and the limitations and rigidity of relying solely on the physical properties of the fluorescent materials, we believe this is a research area with a great potential for a lot of work in the near future.

Bibliography

[1] C. Sulzberger, “First edison lights at sea: The ss columbia story, 1880-1907 [history]”, *IEEE Power and Energy Magazine*, vol. 13, no. 1, pp. 92–101, 2015.

[2] L. Bayat, M. Arab, S. Aliniaeifard, M. Seif, O. Lastochkina, and T. Li, “Effects of growth under different light spectra on the subsequent high light tolerance in rose plants”, *AoB Plants*, vol. 10, no. 5, ply052, 2018.

[3] “How do artificial lights work?” `https://ec.europa.eu/health/scientific_committees/opinions_layman/artificial-light/en/l-3/2-technologies.htm`.

[4] N. Narendran, Y. Gu, J. Freyssinier-Nova, and Y. Zhu, “Extracting phosphor-scattered photons to improve white led efficiency”, *physica status solidi (a)*, vol. 202, no. 6, R60–R62, 2005.

[5] D. M. Gates, H. J. Keegan, J. C. Schleter, and V. R. Weidner, “Spectral properties of plants”, *Applied optics*, vol. 4, no. 1, pp. 11–20, 1965.

[6] T. Back, “Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms”, Oxford university press, 1996.

[7] M. Cesaria and B. Di Bartolo, “Nanophosphors-based white light sources”, *Nanomaterials*, vol. 9, no. 7, p. 1048, 2019.

[8] M. Haase and H. Schäfer, “Upconverting nanoparticles”, *Angewandte Chemie International Edition*, vol. 50, no. 26, pp. 5808–5829, 2011.

[9] J. E. C. Silva, G. F. de Sa, and P. A. Santa-Cruz, “Red, green and blue light generation in fluoride glasses controlled by double excitation”, *Journal of alloys and compounds*, vol. 323, pp. 336–339, 2001.

[10] S. Selvi, K. Marimuthu, N. S. Murthy, and G. Muralidharan, “Red light generation through the lead boro- telluro- phosphate glasses activated by eu3+ ions”, *Journal of Molecular Structure*, vol. 1119, pp. 276–285, 2016.

[11] D. Zhou, R. Wang, X. He, J. Yi, Z. Song, Z. Yang, X. Xu, X. Yu, and J. Qiu, "Color-tunable luminescence of eu3+ in pbf2 embedded in oxyfluoroborate glass and its nanocrystalline glass", *Journal of Alloys and Compounds*, vol. 621, pp. 62–65, 2015.

[12] R. Peckner, S. A. Myers, J. D. Egertson, R. S. Johnson, J. G. Abelin, S. A. Carr, M. J. MacCoss, and J. D. Jaffe, "Specter: Linear deconvolution as a new paradigm for targeted analysis of data-independent acquisition mass spectrometry proteomics", *BioRxiv*, p. 152 744, 2017.

[13] R. Peckner, S. A. Myers, A. S. V. Jacome, J. D. Egertson, J. G. Abelin, M. J. MacCoss, S. A. Carr, and J. D. Jaffe, "Specter: Linear deconvolution for targeted analysis of data-independent acquisition mass spectrometry proteomics", *Nature methods*, vol. 15, no. 5, p. 371, 2018.

[14] M. A. Ciach, B. Miasojedow, G. Skoraczynski, S. Majewski, M. Startek, D. Valkenborg, and A. Gambin, "Masserstein: Linear regression of mass spectra by optimal transport", *Rapid Communications in Mass Spectrometry*, 2020.

[15] C. Villani, "Optimal transport: old and new", Springer Science & Business Media, 2008, vol. 338.

[16] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks", in *International conference on machine learning*, PMLR, 2017, pp. 214–223.

[17] L. Bengi, B. Kovács, M. Bezdek, and E. Keszei, "Model-free deconvolution of transient signals using genetic algorithms", *Ramirez Muñoz, A. Garza Rodriguez, I.(eds.) Handbook of Genetic Algorithms: New Research. Nova Science Publishers: New York*, pp. 41–59, 2012.

[18] J.-C. Hung and B. Chen, "Combining a genetic algorithm and simulated annealing to design a fixed-order mixed h 2/h deconvolution filter with missing observations", *JCSE-Journal of Control Science and Engineering*, vol. 2008, p. 22, 2008.

[19] M. Garcia-Talavera and B. Ulicny, "A genetic algorithm approach for multiplet deconvolution in γ-ray spectra", *Nuclear Instruments and Methods in Physics*

Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, vol. 512, no. 3, pp. 585–594, 2003.

[20] C. Carlevaro, M. Wilkinson, and L. Barrios, "A genetic algorithm approach to routine gamma spectra analysis", *Journal of Instrumentation*, vol. 3, no. 01, P01001, 2008.

[21] G. Fox, "The brewing industry and the opportunities for real-time quality analysis using infrared spectroscopy", *Applied Sciences*, vol. 10, Jan. 2020.

[22] A. Subramanian and L. Rodriguez-Saona, "Chapter 7 - fourier transform infrared (ftir) spectroscopy", in *Infrared Spectroscopy for Food Quality Analysis and Control*, D.-W. Sun, Ed., San Diego: Academic Press, 2009, pp. 145–178, ISBN: 978-0-12-374136-3. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780123741363000079.

[23] L. Ma, Y. Peng, Y. Pei, J. Zeng, H. Shen, J. Cao, Y. Qiao, and Z. Wu, "Systematic discovery about nir spectral assignment from chemical structural property to natural chemical compounds", *Scientific reports*, vol. 9, no. 1, pp. 1–17, 2019.

[24] S. Wold, M. Sjöström, and L. Eriksson, "Pls-regression: A basic tool of chemometrics", *Chemometrics and intelligent laboratory systems*, vol. 58, no. 2, pp. 109–130, 2001.

[25] A. Sharifi, "Partial least squares-regression (pls-regression) in chemometrics",, Jun. 2016.

[26] H. A. Farahani, A. Rahiminezhad, L. Same, *et al.*, "A comparison of partial least squares (pls) and ordinary least squares (ols) regressions in predicting of couples mental health based on their communicational patterns", *Procedia-Social and Behavioral Sciences*, vol. 5, pp. 1459–1463, 2010.

[27] H. Wold, "Nonlinear iterative partial least squares (nipals) modelling: Some current developments",, 1973.

[28] X. Sun, H. Li, Y. Yi, H. Hua, Y. Guan, and C. Chen, "Rapid detection and quantification of adulteration in chinese hawthorn fruits powder by near-infrared spectroscopy combined with chemometrics", *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, vol. 250, p. 119 346, 2021.

[29] G. Mendes and P. J. Barbeira, “Detection and quantification of adulterants in gasoline using distillation curves and multivariate methods”, *Fuel*, vol. 112, pp. 163–171, 2013.

[30] C. G. Pereira, A. I. N. Leite, J. Andrade, M. J. V. Bell, and V. Anjos, “Evaluation of butter oil adulteration with soybean oil by ft-mir and ft-nir spectroscopies and multivariate analyses”, *Lwt*, vol. 107, pp. 1–8, 2019.

[31] N. O’Mahony, S. Campbell, A. Carvalho, S. Harapanahalli, G. V. Hernandez, L. Krpalkova, D. Riordan, and J. Walsh, “Deep learning vs. traditional computer vision”, in *Science and Information Conference*, Springer, 2019, pp. 128–144.

[32] H. Li, “Deep learning for natural language processing: Advantages and challenges”, *National Science Review*, 2017.

[33] H. Chen, O. Engkvist, Y. Wang, M. Olivecrona, and T. Blaschke, “The rise of deep learning in drug discovery”, *Drug discovery today*, vol. 23, no. 6, pp. 1241–1250, 2018.

[34] D. Li, J. Zhang, Q. Zhang, and X. Wei, “Classification of ECG signals based on 1D convolution neural network”, in *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*, IEEE, 2017, pp. 1–6.

[35] J. Liu, M. Osadchy, L. Ashton, M. Foster, C. J. Solomon, and S. J. Gibson, “Deep convolutional neural networks for raman spectrum recognition: A unified solution”, *Analyst*, vol. 142, no. 21, pp. 4067–4074, 2017.

[36] A. Kumar, M. Khadkevich, and C. Fügen, “Knowledge transfer from weakly labeled audio using convolutional neural network for sound events and scenes”, in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2018, pp. 326–330.

[37] H. A. Neto, W. L. Tavares, D. C. Ribeiro, R. C. Alves, L. M. Fonseca, and S. V. Campos, “On the utilization of deep and ensemble learning to detect milk adulteration”, *BioData mining*, vol. 12, no. 1, pp. 1–13, 2019.

[38] L. Ribeiro, A. Soares, T. de Lima, C. Jorge, R. Da Costa, R. Salvini, C. Coelho, F. Federson, and P. Gabriel, “Hr multi-objective genetic algorithm for variable selection in multivariate classification problems: A case study in verification of biodiesel adulteration”, *Procedia. Comput. Sci*, vol. 51, pp. 346–355, 2015.

[39] J. Peng, W. Xie, J. Jiang, Z. Zhao, F. Zhou, and F. Liu, “Fast quantification of honey adulteration with laser-induced breakdown spectroscopy and chemometric methods”, *Foods*, vol. 9, no. 3, p. 341, 2020.

[40] P. Hammond, “Self-absorption of molecular fluorescence, the design of equipment for measurement of fluorescence decay, and the decay times of some laser dyesa”, *The Journal of Chemical Physics*, vol. 70, no. 8, pp. 3884–3894, 1979.

[41] S. Bondarev, V. Knyukshto, V. Stepuro, A. Stupak, and A. Turban, “Fluorescence and electronic structure of the laser dye dcm in solutions and in polymethylmethacrylate”, *Journal of Applied Spectroscopy*, vol. 71, no. 2, pp. 194–201, 2004.

[42] A. M. Al-Etaibi and M. A. El-Apasery, “A comprehensive review on the synthesis and versatile applications of biologically active pyridone-based disperse dyes”, *International Journal of Environmental Research and Public Health*, vol. 17, no. 13, p. 4714, 2020.

[43] T. Deligeorgiev, A. Vasilev, S. Kaloyanova, and J. J. Vaquero, “Styryl dyes–synthesis and applications during the last 15 years”, *Coloration Technology*, vol. 126, no. 2, pp. 55–80, 2010.

[44] A. Gholizadeh, L. Borůvka, M. M. Saberioon, J. Kozak, R. Vašát, and K. Němeček, “Comparing different data preprocessing methods for monitoring soil heavy metals based on soil spectral features”, *Soil and Water Research*, vol. 10, no. 4, pp. 218–227, 2015.

[45] G. Schulze, A. Jirasek, M. Marcia, A. Lim, R. F. Turner, and M. W. Blades, “Investigation of selected baseline removal techniques as candidates for automated implementation”, *Applied spectroscopy*, vol. 59, no. 5, pp. 545–574, 2005.

[46] C. A. Lieber and A. Mahadevan-Jansen, “Automated method for subtraction of fluorescence from biological raman spectra”, *Applied spectroscopy*, vol. 57, no. 11, pp. 1363–1367, 2003.

[47] D. Pelliccia, “Two scatter correction techniques for nir spectroscopy in python”, `https://nirpyresearch.com/two-scatter-correction-techniques-nir-spectroscopy-python/`, 2018.

[48] A. Savitzky and M. J. Golay, “Smoothing and differentiation of data by simplified least squares procedures.” *Analytical chemistry*, vol. 36, no. 8, pp. 1627–1639, 1964.

[49] X. WEINAN, “Differential evolution for solving constrained and large-scale optimization problems”, 2019.

[50] A. Slowik and H. Kwasnicka, “Evolutionary algorithms and their applications to engineering problems”, *Neural Computing and Applications*, pp. 1–17, 2020.

[51] D. E. Goldberg, “Genetic algorithms in search”, *Optimization, and Machine-Learning*, 1989.

[52] K. Jebari, M. Madiafi, and A. Elmoujahid, “Parent selection operators for genetic algorithms”, *International Journal of Engineering*, vol. 2, 2013.

[53] A. Hassanat, K. Almohammadi, E. Alkafaween, E. Abunawas, A. Hammouri, and V. Prasath, “Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach”, *Information*, vol. 10, no. 12, p. 390, 2019.

[54] R. Bellman and R. Kalaba, “On adaptive control processes”, *IRE Transactions on Automatic Control*, vol. 4, no. 2, pp. 1–9, 1959.

[55] P. Senin, “Dynamic time warping algorithm review”, *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA*, vol. 855, no. 1-23, p. 40, 2008.

[56] K. A. Smet, “Tutorial: The luxpy python toolbox for lighting and color science”, *Leukos*, 2019.

[57] Y. Ohno, “Spectral design considerations for white led color rendering”, *Optical Engineering*, vol. 44, no. 11, p. 111 302, 2005.

[58] K. Smet, W. R. Ryckaert, M. R. Pointer, G. Deconinck, and P. Hanselaer, “Optimal colour quality of led clusters based on memory colours”, *Optics Express*, vol. 19, no. 7, pp. 6903–6912, 2011.

[59] M. R. Luo, G. Cui, and B. Rigg, “The development of the cie 2000 colour-difference formula: Ciede2000”, *Color Research & Application*, vol. 26, no. 5, pp. 340–350, 2001.

[60] B. Török, “Human eye color change sensibility in cielab units”, Jun. 2014.

[61] F. J. Schmitt, “A method for the treatment of metamerism in colorimetry”, *JOSA*, vol. 66, no. 6, pp. 601–608, 1976.