
Autonomous University of Querétaro

Faculty of Engineering

Master of Science Program

THESIS

Submitted as part of the requirements to obtain the degree of
Master of Science

Presented by:

**Ing. Rafael Rojas Galván**

Advisor:

**Dr. Juvenal Rodríguez Reséndiz**

COMMITTEE

Dr. Juvenal Rodríguez Reséndiz
President

Dr. José Román García Martínez
Co-Advisor

Dr. José Manuel Álvarez Alvarado
Member

Dr. Marcos Romo Avilés
Substitute

Dr. Edson Eduardo Cruz Miguel
Substitute

Querétaro, QRO
México.
December 2025

Autonomous University of Querétaro


Faculty of Engineering
Master of Science Program

# Implementation of an optimization algorithm for MPPT prediction using a regression model

Thesis

Submitted as part of the requirements to obtain the degree of
Master of Science

*Presented by:*
Ing. Rafael Rojas Galván
*Advisor:*
Dr. Juvenal Rodríguez Reséndiz
*Co-Advisor:*
Dr. José Román García Martínez
*Member*
Dr. José Manuel Álvarez Alvarado
*Committee Member:*
Substitute: Dr. Marcos Romo Avilés
Substitute: Dr. Edson Eduardo Cruz Miguel

Querétaro, Qro., December 2025

# Abstract

This thesis presents the development and implementation of a predictive model for Maximum Power Point Tracking (MPPT) in photovoltaic systems under partial shading conditions. The approach integrates an Artificial Neural Network (ANN) with the Grey Wolf Optimizer (GWO), a bio-inspired metaheuristic used for hyperparameter tuning. The methodology comprised five stages: data acquisition from experimental sensors, preprocessing to simulate shading conditions, ANN model training, optimization through GWO, and validation under real operating conditions. Experimental evaluation confirmed that the optimized ANN–GWO model significantly improved prediction accuracy, achieving a determination coefficient ($R^2$) above 0.99 and a prediction accuracy greater than 98.9%, while maintaining feasible computational costs for embedded implementation. The model was successfully deployed on a Raspberry Pi 4, demonstrating its practical applicability in real-time MPPT prediction for photovoltaic systems. This research highlights the potential of combining machine learning and bio-inspired optimization to enhance solar energy efficiency and contribute to sustainable power generation.

**Keywords:** Photovoltaic systems, MPPT, Artificial Neural Networks, Grey Wolf Optimizer.

# Resumen

Este trabajo presenta el desarrollo e implementación de un modelo predictivo para el seguimiento del punto de máxima potencia (MPPT) en sistemas fotovoltaicos bajo condiciones de sombreado parcial. La propuesta integra una Red Neuronal Artificial (RNA) con el Grey Wolf Optimizer (GWO), un algoritmo bioinspirado empleado para el ajuste de hiperparámetros. La metodología se estructuró en cinco etapas: adquisición de datos mediante sensores experimentales, preprocesamiento para simular escenarios de sombreado, entrenamiento del modelo de RNA, optimización con GWO y validación en condiciones reales de operación. La evaluación experimental confirmó que el modelo RNA–GWO optimizado mejoró significativamente la precisión de las predicciones, alcanzando un coeficiente de determinación ($R^2$) superior a 0.99 y una exactitud mayor al 98.9%, manteniendo al mismo tiempo un costo computacional adecuado para su implementación en sistemas embebidos. El modelo fue implementado exitosamente en una Raspberry Pi 4, demostrando su viabilidad práctica para la predicción en tiempo real de MPPT en sistemas fotovoltaicos. Esta investigación resalta el potencial de combinar aprendizaje automático y optimización bioinspirada para mejorar la eficiencia de la energía solar y contribuir a la generación sustentable.

**Palabras clave:** Sistemas fotovoltaicos, MPPT, Redes neuronales artificiales, Optimizador del lobo gris.

*Dedico este trabajo a quienes, con su ejemplo constante y su esfuerzo silencioso, me enseñaron que la verdadera grandeza se construye día a día, sin necesidad de reconocimientos visibles.*
*Es también un homenaje a la perseverancia y a la fuerza discreta de quienes avanzan con determinación, demostrando que el compromiso sostenido transforma cualquier meta en una realidad posible.*
*De manera más íntima, a Rafael del pasado, que con convicción eligió este camino y sostuvo con firmeza la voluntad de llegar hasta aquí; a Rafael del presente, que reconoce en este logro la confirmación de que la disciplina y la constancia son la base de todo avance; y a Rafael del futuro, como recordatorio de que cada etapa cumplida abre nuevos senderos, y que el esfuerzo mantenido será siempre la brújula que marque el rumbo.*

# Agradecimientos

Los logros académicos, aunque se escriban con un solo nombre, son siempre fruto de un esfuerzo compartido. Este trabajo refleja la suma de muchas voluntades, palabras de aliento y apoyos que acompañaron mi trayecto.

A mi familia, por su respaldo incondicional, por la paciencia en los momentos de mayor exigencia y por ser el motor constante de mi perseverancia.

A mi director de tesis y al cuerpo académico de la institución, por su guía exigente, su orientación crítica y la generosidad de compartir conocimientos que van más allá de estas páginas.

A mis compañeros y colegas, por las conversaciones, los debates y las experiencias que enriquecieron no solo este proyecto, sino también mi visión profesional y personal.

Y, finalmente, a todas aquellas personas e instituciones que, de manera visible o silenciosa, contribuyeron a este proceso. Este trabajo no es un punto final, sino la apertura de nuevas posibilidades; un recordatorio de que cada logro académico es también una responsabilidad hacia el futuro.

# Contents

# List of Figures

# List of Tables

CHAPTER 1

# Introduction

Solar radiation is among the most abundant renewable energy resources; however, its efficient exploitation is significantly constrained by environmental and technological factors. The process of energy conversion is directly influenced by irradiance intensity and the operating temperature of photovoltaic (PV) modules. Standard PV systems typically achieve a conversion efficiency of about 15%, which makes it essential to understand module behavior to evaluate solar plant performance under different operating conditions. Variations in irradiance and temperature generate nonlinear power–voltage characteristics, especially under partial shading. Therefore, operating PV modules at their Maximum Power Point (MPP) becomes crucial to maximize energy extraction, a task accomplished through Maximum Power Point Tracking (MPPT) strategies, each with particular advantages and limitations [3].

Recent developments in PV technology have responded to the increasing global demand for clean energy. The depletion of fossil fuels and rising concerns about environmental pollution have accelerated the adoption of photovoltaic solutions [4]. Nevertheless, challenges such as high material costs and relatively low conversion efficiencies persist. The incorporation of MPPT-enabled power converters is therefore a key step toward enhancing PV system performance and reliability.

Conventional MPPT algorithms work effectively under uniform irradiance. Yet, their accuracy decreases in scenarios with rapid weather fluctuations or partial shading, often failing to identify the true global MPP. To address these issues, advanced methods—such as stochastic optimization techniques and artificial intelligence—have emerged, offering superior accuracy and faster convergence when multiple local peaks are present [5].

Classical methods like Perturb and Observe (P&O) or Incremental Conductance (INC) rely on heuristic adjustments and mathematical approximations of the MPP. Although widely used, their effectiveness diminishes under highly dynamic conditions. By contrast, mathematical approaches such as curve fitting and beta MPPT achieve higher precision. The former requires knowledge of module characteristics and explicit equations, whereas the latter estimates the MPP rapidly using a variable coefficient.

Other families of methods include measurement- and comparison-based strategies, such as lookup tables and current sweep techniques. The lookup table method requires prior knowledge of PV specifications and weather conditions, alongside large storage resources, while the current sweep approach reconstructs the I–V curve iteratively, at the expense of computational complexity and slower operation. Simpler parameter-based strategies like FOCV and FSCC rely on linear approximations but suffer from periodic energy losses [6].

Trial-and-error approaches—including gradient descent or variable inductance—along with evolutionary optimization techniques like genetic algorithms, can reduce accuracy due to oscillations and slow response. Intelligent methods such as artificial neural networks (ANNs) and fuzzy logic controllers instead learn patterns from data, predicting optimal operating points based on training and validation. Similarly, machine learning algorithms (MLA) are evaluated with performance indices such as RMSE, SSE, and $R^2$ [7].

The global expansion of PV energy reflects both its environmental benefits and long-term cost advantages. However, enhancing system efficiency remains a pressing challenge. Robust MPPT algorithms are essential for dynamically adjusting voltage and current to keep PV systems operating at their MPP under continuously changing environmental factors like temperature and irradiance. Ongoing research on MLA-based MPPT approaches shows strong potential for improving efficiency, reliability, and sustainability in photovoltaic power generation [8].

## 1.1 Justification

The assessment of photovoltaic (PV) system efficiency and the advancement of maximum power point tracking (MPPT) strategies rest on several key considerations. Solar energy represents a promising alternative to reduce the environmental footprint of conventional energy sources such as coal and natural gas. According to the International Energy Agency (IEA), wider adoption of solar power could lower global carbon dioxide emissions by up to 30% by the year 2030 [9]. Moreover, the performance of PV modules is highly sensitive to environmental factors, especially temperature and irradiance. For instance, an increase of just 1°C can diminish panel efficiency by nearly 0.5% [10]. Maximizing this efficiency is therefore essential to strengthen the profitability and competitiveness of solar energy, particularly in climates with high variability.

Ensuring that PV systems operate at their Maximum Power Point (MPP) is fundamental for extracting maximum energy. MPPT algorithms provide the mechanisms to dynamically adjust system operation under varying weather conditions. A wide spectrum of approaches exists, ranging from classical heuristic and model-based methods to modern machine learning techniques. The selection of an appropriate method has a direct impact on system stability and overall efficiency. For example, neural network–based approaches have demonstrated improvements of 10–15% in tracking accuracy compared to conventional techniques [11].

Optimizing PV operation not only improves energy yield but also reduces maintenance costs and prolongs equipment lifetime. Studies suggest that advanced MPPT algorithms can cut energy losses by up to 25% and extend PV system lifespan by 20–30% [12]. These benefits apply across scales, from residential systems to large commercial plants. As an illustration, optimized MPPT integration in industrial PV installations can lead to savings of approximately $50,000 annually in operational expenses [13].

Despite technological progress, considerable challenges persist in ensuring reliable and efficient PV operation. Ongoing research in MPPT development, especially through the integration of machine learning models, is expected to improve both the accuracy and resilience of tracking under diverse and dynamic operating conditions. Notably, machine learning–based MPPT solutions have been reported to enhance overall system efficiency by 5–10% when exposed to fluctuating irradiance [14].

## 1.2 Description of the problem

The deployment of photovoltaic (PV) solar energy systems faces multiple challenges that must be addressed to ensure efficiency. PV modules naturally exhibit non-linear power–voltage (P–V) characteristics as a result of temperature fluctuations and varying irradiance, which complicates the consistent extraction of maximum power under diverse environmental conditions. These non-linearities highlight the need for advanced strategies to improve energy conversion efficiency [15].

For optimal performance, PV systems must operate at their Maximum Power Point (MPP). Nonetheless, locating and maintaining this operating point becomes particularly difficult in dynamic or non-uniform irradiance scenarios. Conventional Maximum Power Point Tracking (MPPT) algorithms, such as Perturb and Observe (P&O) and Incremental Conductance (INC), often demonstrate reduced effectiveness when exposed to rapid environmental changes or partial shading, conditions frequently encountered in real-world applications. These limitations typically result in decreased accuracy and unstable operation [16].

Recent advances, especially those incorporating machine learning, show great promise for improving MPP tracking by adapting to variable conditions. However, their adoption entails significant challenges, including the demand for extensive datasets, rigorous training, and the management of algorithmic complexity to ensure practical implementation [17, 18].

The wide range of MPPT techniques available creates both opportunities and difficulties. While traditional approaches are well established and broadly applied, their performance may be suboptimal under specific scenarios. On the other hand, advanced methods—particularly those based on artificial intelligence—can provide superior accuracy and adaptability but increase system complexity and require comprehensive validation [19, 20].

Environmental variability, such as fluctuations in irradiance and temperature across different times of day and geographical locations, further complicates PV system performance. To maintain optimal efficiency, MPPT algorithms must be capable of adapting to these dynamic conditions. Studies demonstrate that non-uniform irradiance and temperature distributions considerably affect both the efficiency and the reliability of PV systems, underscoring the necessity for more resilient and adaptive control strategies [21, 15].

Moreover, the adoption of advanced MPPT methods is influenced by their cost and implementation complexity. While they can significantly improve system performance, they may also increase the financial and technical burden of PV installations. Therefore, achieving a balance between performance gains and cost-effectiveness is critical for ensuring widespread adoption and practical viability of solar energy solutions [21, 16].

Ultimately, the ongoing scientific debate centers on whether to refine traditional algorithms or to pursue entirely new approaches. Although conventional methods are dependable, innovation is required to address their limitations under dynamic operating conditions. Advanced techniques hold great potential but demand careful consideration regarding cost, scalability, and validation. Tackling these challenges is essential to improve the efficiency and robustness of PV systems, paving the way for broader implementation and enhanced environmental benefits [18, 20].

## 1.3 Hypothesis

Implementing a multi-objective optimization algorithm using a regression-based MPPT system has at least 90% of precision prediction considering the partial shade condition in a real scenario.

## 1.4 General Objective

To predict the MPPT in a photovoltaic system using a improved machine learning-based regression algorithm with data from the environmental sensors in partial shade conditions.

## 1.5 Specific Goals

- To collect data from the photovoltaic panel and environmental sensors for preprocessing.
- To develop an improved machine learning-based regression model for predicting the optimal operating points based on collected data.
- To integrate the trained model into the photovoltaic system for MPPT prediction.
- To evaluate implemented model performance in terms of accuracy in partial shade conditions.

CHAPTER 2

# Theoretical Background

En esta sección se definirán los conceptos científicos necesarios para comprender los métodos que utilizarán para este trabajo.

### Photovoltaic Solar Energy Systems

Photovoltaic technology is widely seen as a cornerstone for achieving sustainable energy systems in the future across many nations. Significant financial resources are allocated to its research, development, and demonstration, with governments launching large-scale market introduction initiatives and industries expanding their production capacities. Among renewable energy technologies, photovoltaics stands out as the most publicly and politically favored, with growing support from the industrial and financial sectors [22, 23]. This is especially noteworthy given that photovoltaic (PV) electricity is still often viewed as being significantly more costly than conventional grid electricity. The widespread enthusiasm for photovoltaics is likely due to its numerous advantageous features as a method of converting solar energy into electricity.

### Features of Photovoltaic Energy Conversion

Photovoltaic systems serve two main purposes: powering off-grid professional devices and systems (such as telecommunications equipment or solar home systems) and generating electricity on a large scale to replace or supplement current unsustainable energy sources. For large-scale applications, the global potential of photovoltaic electricity is particularly crucial. The theoretical potential does not account for limitations such as land use restrictions, conversion efficiency, or storage needs. However, the technical potential should not be mistaken for short-term economic feasibility, as market conditions and investment requirements for large-scale deployment are not factored in [24].

Photovoltaic energy conversion aligns closely with the principles of sustainable energy production. It operates without releasing harmful emissions or pollutants, producing noise, or generating by-products. The process itself is a straightforward and efficient one-step conversion of sunlight into electricity, avoiding the need for conventional thermodynamic or mechanical intermediaries. Nevertheless, as with any industrial manufacturing process, the production of photovoltaic modules and system components involves material use and waste generation. Therefore, it is critical to develop photovoltaic technologies that adhere to environmentally friendly production practices [25].

### Embedded Systems

An embedded system is a special-purpose computer system designed to execute one or a few dedicated functions, typically with real-time computing constraints. It combines hardware and software in a tightly coupled way, optimized for specific tasks rather than general-purpose use. According to Wang et al. (2017), "An embedded system is generally defined as a special-purpose computer, which is based on application, computer technology, software and hardware can be cut and adapted to the application system, and the functions, reliability, cost, volume and power are strictly required" [26]. Similarly, other authors define it as "a programmed controlling and operating system with a dedicated function within a larger mechanical or electrical system, often with real-time computing constraints" [27]. These definitions highlight that embedded systems must deliver predictable performance under resource limitations of memory, power, and computational capacity, which makes them essential for applications in industrial control, biomedical devices, automotive, and renewable energy systems.

### Solar Collectors

A solar collector is a device specifically designed to capture solar radiation and convert it into useful thermal energy, usually transferring the absorbed heat to a working fluid such as water or air. Ahmadi et al. (2020) define it as "equipment which is used to gather sun-rays and absorb sunlight thermal energy and deliver it to a working fluid, mostly air or water" [28]. Likewise, scientific references emphasize that solar collectors are central components in solar thermal systems, where their efficiency depends on design features such as glazing, absorber material, insulation, and the type of collector employed (flat-plate, evacuated tube, or concentrating) [29]. For example, flat-plate collectors generally consist of an absorber plate, transparent glazing, insulation, and embedded pipes, while concentrating collectors employ mirrors or lenses with tracking systems to achieve higher temperatures. These devices play a crucial role in renewable energy systems for water heating, space heating, and even industrial applications.

### MPPT

Solar radiation is one of the most abundant renewable energy sources, but its practical utilization is highly dependent on environmental and technological conditions. Specifically, the energy conversion process is strongly influenced by the irradiance levels and the temperature of the photovoltaic module. Additionally, it is widely recognized that the conversion efficiency of standard installations remains relatively low, approximately 15%. Understanding the operational characteristics of the module is crucial for evaluating the solar plant's efficiency under various conditions and identifying the operating point where the maximum power is generated. This is achieved through the Maximum Power Point Tracker (MPPT), which optimizes the power output of a photovoltaic system by adjusting to the given radiation and temperature conditions. The MPPT relies on a central algorithm that continuously searches for and maintains operation near the Maximum Power Point (MPP) [21].

### Machine Learning

Machine learning focuses on deriving meaningful insights from data and operates at the crossroads of statistics, artificial intelligence, and computer science. It is often referred to as predictive ana-

lytics or statistical learning. In recent years, machine learning has become a fundamental part of various everyday technologies. Examples include systems that recommend movies, suggest food options, enable online music personalization, and even identify individuals in photographs. Numerous modern websites and smart devices rely on machine learning algorithms as an integral component of their operation [30, 31, 32].

### Why Machine Learning?

In earlier versions of "intelligent" systems, decision-making processes relied heavily on manually designed rules based on "if-else" logic. Consider a spam filter that aims to categorize emails into spam or non-spam. A rule-based system could involve creating a blacklist of specific keywords to identify spam emails. While this approach demonstrates how expert-designed rules can create basic intelligent applications, it has significant limitations:

- The rules are narrowly tailored to specific tasks or domains, requiring complete re-engineering for even slight changes to the task.

- Developing rules necessitates a detailed understanding of the problem and expert insights on decision-making.

An example of the shortcomings of rule-based systems is facial detection in images. For instance, modern smartphones easily detect faces in photographs, but this task remained unresolved until 2001. The core issue lies in the fact that the representation of pixels in an image, as processed by a computer, vastly differs from how humans perceive a face. This disparity makes it impractical for humans to devise effective rules for defining a face in digital images [33].

### Challenges Machine Learning Can Address

Machine learning algorithms excel at automating decision-making processes by identifying patterns from previously observed data. This paradigm, known as supervised learning, involves training an algorithm using input-output pairs provided by the user. The algorithm then learns to generate the desired output for a given input, including inputs it has not encountered before, without human intervention.

For instance, in spam email classification, the user supplies a dataset of emails (inputs) along with their corresponding labels, indicating whether they are spam (desired outputs). Based on this data, the algorithm predicts whether a new email belongs to the spam category. Algorithms that learn from such paired data are called supervised learning algorithms, as the desired outputs act as a form of supervision. Although creating labeled datasets can be a time-intensive process, supervised learning algorithms are well understood, and their effectiveness is straightforward to measure. If your problem can be formulated within this framework and an appropriate dataset can be constructed, machine learning is likely to provide a viable solution [34, 35].

### Regression Model

Regression is a fundamental concept in statistics and machine learning used to establish and quantify the relationship between variables. It seeks to understand how one or more independent variables (predictors) influence a dependent variable (response). The primary purpose of regression

is to predict the dependent variable based on given predictors. Regression methods can be categorized broadly into linear regression, which assumes a proportional relationship, and nonlinear regression, which accounts for more intricate patterns and dynamics [36, 37].

### Supervised Learning

Supervised learning is a machine learning framework where algorithms are trained using data with known labels. Each input in the dataset is paired with a corresponding output, enabling the model to learn the mapping between them. The goal is to make accurate predictions for new, unseen inputs. Tasks within supervised learning include classification, where the outputs are categorical, and regression, where the outputs are continuous. Regression tasks, in particular, focus on predicting numerical values by analyzing the relationship between features and outcomes [38, 39].

### Evaluation of Regression Models

Evaluating the performance of a regression model is critical to understanding its ability to generalize to unseen data. Several metrics are commonly used to assess the quality of regression models:

- **Coefficient of Determination ($R^2$)**: Indicates the fraction of variability in the dependent variable that can be explained by the independent ones. Its range is from 0 to 1, where values closer to 1 imply stronger explanatory power [37].

- **Mean Absolute Error (MAE)**: Represents the average of the absolute differences between predicted and observed values, ignoring the direction of the error. Smaller values denote higher predictive accuracy [40].

- **Mean Squared Error (MSE)**: Refers to the mean of squared differences between predictions and actual values. Since larger errors are squared, this metric is more sensitive to significant deviations [40].

- **Root Mean Squared Error (RMSE)**: Defined as the square root of MSE, it reports the error in the same units as the output variable, making interpretation more intuitive [37].

These performance indicators offer an overall assessment of the model's effectiveness, enabling researchers and practitioners to determine the most suitable approach for their particular application.

### Pareto Optimality

In multi-objective optimization, a solution is Pareto optimal if improving one objective results in the deterioration of at least one other. The Pareto front is a collection of such optimal solutions and serves as a visual tool to explore the trade-offs among objectives. Decision-makers can use this front to select solutions that align with their preferences and priorities.

## Approaches to Multi-Objective Optimization

There are various strategies to tackle multi-objective optimization problems, each offering distinct advantages:

- **Weighted Sum Method**: Combines all objectives into a single scalar function by assigning weights to each one. This method requires predefining weights, which may not always capture the complexity of trade-offs [41].

- **Pareto-based Methods**: Algorithms like Non-dominated Sorting Genetic Algorithm II (NSGA-II) and Strength Pareto Evolutionary Algorithm (SPEA2) focus on identifying a diverse set of solutions along the Pareto front using evolutionary techniques [42, 43].

- **Constraint Method**: This approach transforms a multi-objective optimization problem into a single-objective formulation by treating all objectives except one as constraints, each bounded by predefined limits [41].

- **Goal Programming**: Sets specific targets for each objective and minimizes the deviation from these targets, making it ideal for problems where goals are clearly defined [44].

- **Multi-Criteria Decision Analysis (MCDA)**: Methods like Analytical Hierarchy Process (AHP) and Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) allow decision-makers to prioritize and evaluate objectives based on personal or organizational preferences [45, 46].

By considering multiple objectives simultaneously, these approaches enable a structured and balanced way to evaluate and solve complex problems [47].

## Optimization Algorithms

Optimization is the process of finding the best possible values for a set of variables that influence an objective function. This process can be carried out with or without constraints. While any combination of variable values will generate an output, the optimal combination produces the most favorable result. Objectives and constraints, if present, are typically defined using mathematical equations or functions. The optimization process is guided by an objective function, which is formulated to either maximize or minimize the desired outcome, depending on the specific requirements of the problem [1].

Figure 2.1 illustrates a general optimization framework, highlighting the interaction between input variables and constraints to achieve the objective function's optimal value. The function, $f(X)$, depends on a set of input variables represented as $X = [x_1, x_2, x_3, x_4]$.

The optimization process typically begins by defining the problem, such as reducing costs, energy usage, or resource consumption, or increasing quality or profit. Once the problem is clearly stated, the objective function is developed, indicating whether the goal is to minimize or maximize the desired outcome. The next step involves identifying constraints, which may be expressed as equalities or inequalities. Parameters influencing the objective function and constraints must be specified, along with their allowable ranges. The ultimate goal is to find the best solution by exploring a defined search space where these parameters interact.

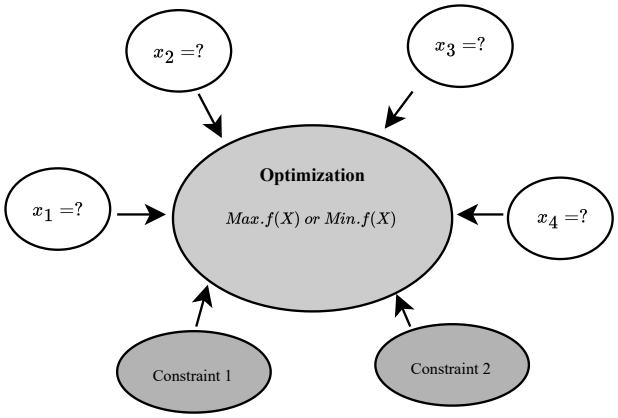



Figure 2.1: General representation of an optimization problem [1]

### Nature-Inspired Optimization Algorithms

Nature-inspired optimization algorithms are metaheuristic methods that draw inspiration from natural phenomena such as biological evolution, collective behavior in swarms, and physical or chemical processes [48]. These algorithms fall under the category of bioinspired computational intelligence techniques, as they incorporate intelligent mechanisms into their design.

### Traditional Optimization Algorithms versus Nature-Inspired Algorithms

Optimization algorithms are mathematical tools developed to identify the best solution while adhering to defined constraints. Traditional optimization methods often struggle to find the global optimum, as their performance heavily relies on initial conditions. Since these algorithms are deterministic, they produce the same outcome if initialized at the same starting point. Classical methods, which often depend on derivatives, are typically unsuitable for problems involving discontinuities, or for those that are highly complex, non-linear, or feature multiple local optima [49].

Nature-inspired algorithms, on the other hand, take inspiration from biological systems and natural processes to tackle challenges that are difficult to solve with traditional methods. These approaches do not require the calculation of derivatives, making them gradient-free and adaptable to a wide range of problems. Additionally, nature-inspired methods are stochastic in nature, meaning that repeated runs starting from the same initial point can yield different results, enhancing their capacity to explore diverse solution spaces.

### Gray Wolf Optimizer

The Gray Wolf Optimizer (GWO), introduced in 2014, is a nature-inspired metaheuristic that mimics the social hierarchy and hunting patterns of gray wolves [50, 51]. In a pack, the alpha ($\alpha$) wolves lead the decisions, betas ($\beta$) assist, deltas ($\delta$) follow their guidance, and omegas ($\omega$) take the lowest rank.

The algorithm represents solutions according to this hierarchy: the best candidate is $\alpha$, followed by $\beta$ and $\delta$, while the rest are $\omega$. The hunting process is modeled through equations that simulate how wolves approach the prey:

$$\vec{D} = \left| \vec{C} \cdot \vec{x}_p(t) - \vec{x}(t) \right| \tag{2.1}$$

$$\vec{x}(t+1) = \vec{x}_p(t) - \vec{A} \cdot \vec{D} \tag{2.2}$$

where $t$ is the current iteration, $\vec{x}_p(t)$ the prey's position, and $\vec{x}(t)$ the wolf's position. The control vectors are defined as:

$$\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a}, \qquad \vec{C} = 2\vec{r}_2 \tag{2.3}$$

with $\vec{r}_1, \vec{r}_2 \in [0, 1]$ and $\vec{a}$ decreasing from 2 to 0 during the iterations, balancing exploration and exploitation.

The position of each wolf is updated considering the three leaders:

$$\vec{x}(t+1) = \frac{\vec{x}_1 + \vec{x}_2 + \vec{x}_3}{3} \tag{2.4}$$

where $\vec{x}_1, \vec{x}_2, \vec{x}_3$ are calculated from the distances to $\alpha$, $\beta$, and $\delta$.

Through this mechanism, the population gradually converges toward the optimal solution, effectively imitating the cooperative hunting behavior of wolf packs.

**Algorithm 1** Pseudocode of the GWO procedure [51]

Generate an initial population of grey wolves $X_i$ $(i = 1, 2, \ldots, n)$
Set the control parameters $a$, $A$, and $C$
Evaluate the fitness value of each candidate solution
Identify $X_\alpha$ as the best solution
Identify $X_\beta$ as the second best solution
Identify $X_\delta$ as the third best solution
**while** ($t <$ maximum iterations) **do**
    **for** each candidate solution **do**
        Update its position according to Eq. (2.2)
    **end for**
    Recalculate $a$, $A$, and $C$
    Re-evaluate the fitness of all candidate solutions
    Update the leaders: $X_\alpha$, $X_\beta$, and $X_\delta$
    Increment iteration counter: $t = t + 1$
**end while**
**return** the best solution $X_\alpha$

Table 2.1 summarizes different optimization algorithms that have been applied to photovoltaic systems, highlighting their optimized metrics, reported precision, operating conditions, and tuning parameters. It can be observed that most algorithms, such as PSO, GWO, SSA, SHO, and hybrid approaches, achieve very high accuracy levels, often exceeding 97% in terms of efficiency or power output prediction. These results demonstrate the potential of metaheuristic optimization in addressing the nonlinear and dynamic nature of photovoltaic maximum power point tracking (MPPT).

However, it is important to note that many of these works remain at the simulation stage. In most cases, the evaluation of optimization algorithms is carried out using standard test conditions (STC), MATLAB/Simulink environments, or synthetic datasets that do not fully represent the variability and unpredictability of real operating conditions. While these approaches provide valuable theoretical insights, their direct applicability to physical photovoltaic systems is limited.

Simulated scenarios often do not capture the complete spectrum of real-world phenomena such as rapid irradiance fluctuations, partial shading caused by passing clouds, or temperature variations at the module surface. As a result, although the precision values appear remarkably high, their robustness in real environments remains uncertain. This gap highlights the need for experimental validation and field testing of optimization-based ANN models.

The methodology proposed in this thesis addresses this limitation by not only training models on public datasets but also validating them experimentally using a one-degree-of-freedom solar collector under real irradiance conditions. In this way, the proposed approach goes beyond simulation, ensuring that the developed ANN-GWO model can be reliably implemented in real photovoltaic applications.

Table 2.1: Comparative summary of optimization algorithms applied to PV systems.

| **Reference** | **Algorithm(s)** | **Objective** | **Performance** | **Conditions** |
|---|---|---|---|---|
| [50] | HHO, CS, GWO, PSO | Eficciency/Power | PSO: 99.32%; GWO: 98.37% | Uniform conditions and partial shading |
| [52] | PSO | Power output | 93.31% | Partial shading |
| [53] | SHO, PSO | Eficiency | SHO: 99.81%; PSO: 98.75% | Shading and temperature variation |
| [54] | MPSO, GWO | Power / Efficiency | MPSO: 98.42%; GWO: 98.76% | Variable weather |
| [55] | SSA | Power / Efficiency | 97.65% | Dynamic partial shading |
| [56] | Firefly, EAS | Power output | Firefly: 97.6%; EAS: 96.8% | Fluctuating irradiance |
| [57] | CS, PSO, ANN, WOA | Power / Efficiency | WOA: 98.72%; PSO: 97.55% | Smart grid environment |

CHAPTER 3

# Methodology

The methodology proposed in this work is structured into five main stages, as illustrated in Figure 3.1. These stages establish a systematic workflow for the development, optimization, and validation of the regression-based MPPT prediction model.

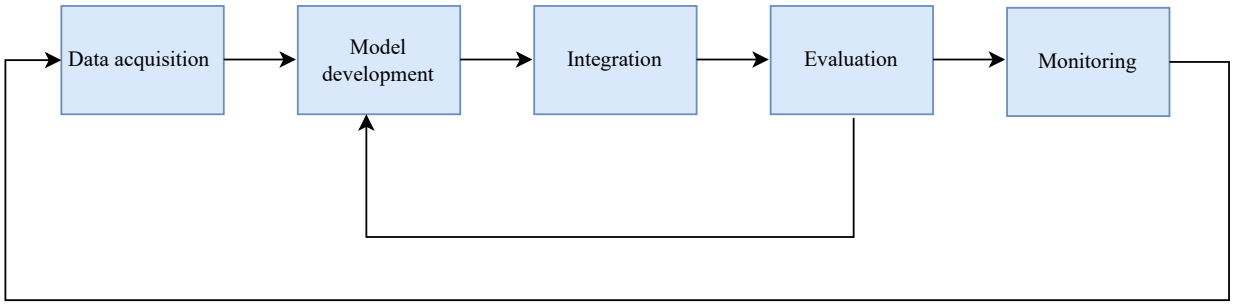



Figure 3.1: Methodology flowchart

**Stage 1: Data Acquisition**

In the first stage, data are collected from photovoltaic repositories, including measurements of irradiance, temperature, and electrical characteristics. To reproduce realistic operating conditions, perturbations are applied to $V_{mp}$ and $I_{mp}$ to simulate partial shading scenarios. The dataset is then cleaned, structured, and normalized for model training.

**Stage 2: Model Development**

Here, the predictive model is defined using an artificial neural network (`MLPRegressor`) with two hidden layers. Hyperparameters such as activation function, optimizer, learning rate, and epochs are specified to ensure proper convergence. The dataset is divided into training and testing subsets to evaluate generalization performance, resulting in a baseline ANN capable of estimating MPPT values.

**Stage 3: Integration**

In this step, the validated ANN is combined with the Grey Wolf Optimizer (GWO) to adjust hyperparameters, mainly the number of neurons per hidden layer, minimizing the mean squared error (MSE). Once optimized, the ANN–GWO model is deployed on a one-degree-of-freedom solar collector, enabling real-time prediction in a hardware environment.

**Stage 4: Evaluation**

The optimized ANN–GWO model is assessed using regression metrics such as MSE, MAE, and $R^2$, along with convergence curves to validate the stability of the optimization. These results quantify prediction accuracy and computational efficiency.

**Stage 5: Monitoring**

Finally, the model is continuously monitored under real operating conditions, where predicted values are compared against actual measurements under varying irradiance and temperature. This monitoring stage confirms the robustness, reliability, and lightweight computational cost of the ANN–GWO approach, validating its feasibility for embedded real-time MPPT applications.

## 3.1 Pre-Process

**Analysis EDA**

The dataset used in this study was obtained from [58]. To simulate partial shading conditions, four perturbations of different magnitudes were introduced, directly modifying the voltage and current values of the photovoltaic panel.

During this stage, adjustments were carried out on the voltage ($V_{mp}(V)$) and current ($I_{mp}(A)$). These modifications were obtained by scaling the respective columns with a reduction factor across predefined ranges.

For each specified range, the perturbations applied to $V_{mp}(V)$ and $I_{mp}(A)$ are defined in Equations (3.1) and (3.2):

$$V_{mp(V)perturbed} = Vmp(V) \times factor \tag{3.1}$$

$$I_{mp(A)perturbed} = Imp(A) \times factor \tag{3.2}$$

where:

- $V_{mp(V)}$ denotes the initial voltage measured at the maximum power point.
- $I_{mp(A)}$ corresponds to the initial current recorded at the MPP.
- $factor$ represents the reduction parameter applied within the chosen interval.

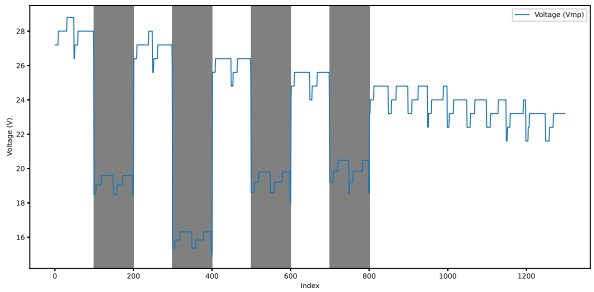
Once the perturbations are introduced, the *Power* column is recomputed following Equation (3.3).

$$Power_{perturbed} = V_{mp(V)_{perturbed}} \times I_{mp(A)_{perturbed}} \tag{3.3}$$

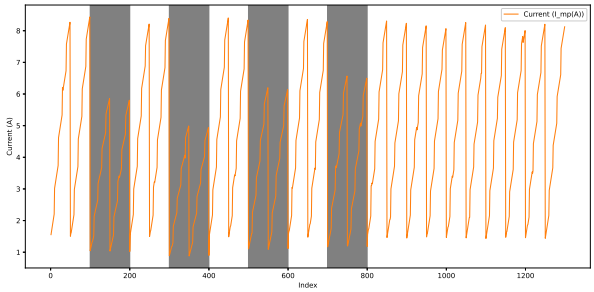
The sequence of steps followed in this procedure is summarized as:

1. Create the initial *Power* column by multiplying $V_{mp}(V)$ and $I_{mp}(A)$.
2. Define the perturbation ranges and assign the corresponding reduction factors.
3. Apply the perturbations to $V_{mp}(V)$ and $I_{mp}(A)$ across the defined segments.
4. Recalculate the *Power* once the perturbations have been applied.
5. Export the updated *DataFrame* as a new Excel file.

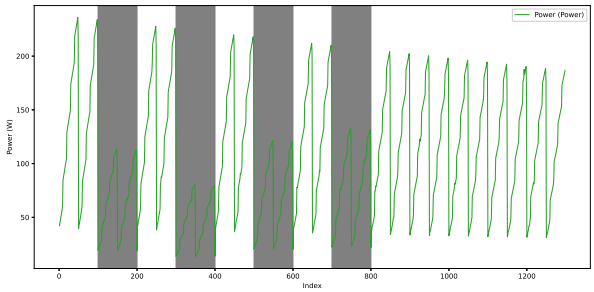
As shown in Figure 3.1, these modifications alter the voltage and current characteristics, thereby simulating the performance of the PV system under partial shading conditions.




(**a**)




(**b**)




(**c**)

Figure 3.2: Results of the perturbations: (**a**) voltage versus index, (**b**) current versus index, and (**c**) power versus index.

Figure 3.3 presents the correlation matrix, which was constructed to analyze the relationships among the variables. Figure 3.3 presents a correlation matrix created to analyze the relationships among the variables. The matrix was visualized as a heat map to emphasize overall trends and interactions within the dataset. In this representation, lighter shades denote positive associations between variables, whereas darker shades indicate negative correlations, meaning that as one variable increases, the other tends to decrease. The results reveal that multicollinearity is not present, since none of the input variables show excessively high correlations with each other. Nevertheless, several significant relationships with the target variable (Power) were identified:

- **Temperature vs. Voltage (Vmp):** A pronounced inverse correlation (-0.94) indicates that higher temperatures considerably reduce voltage levels.
- **Irradiance vs. Voltage (Vmp):** A modest direct correlation (0.28) suggests that voltage

increases slightly as irradiance rises.

- **Current ($\mathbf{I}_{mp(A)}$) vs. Power:** An almost perfect direct correlation (0.99) confirms that current is the primary factor influencing power generation.
- **Irradiance vs. Power:** A high positive association (0.90) demonstrates that greater irradiance levels directly improve power output.

The results indicate that Temperature, Irradiance, Current, and Voltage are the key variables with the greatest impact on the power output of the photovoltaic system.

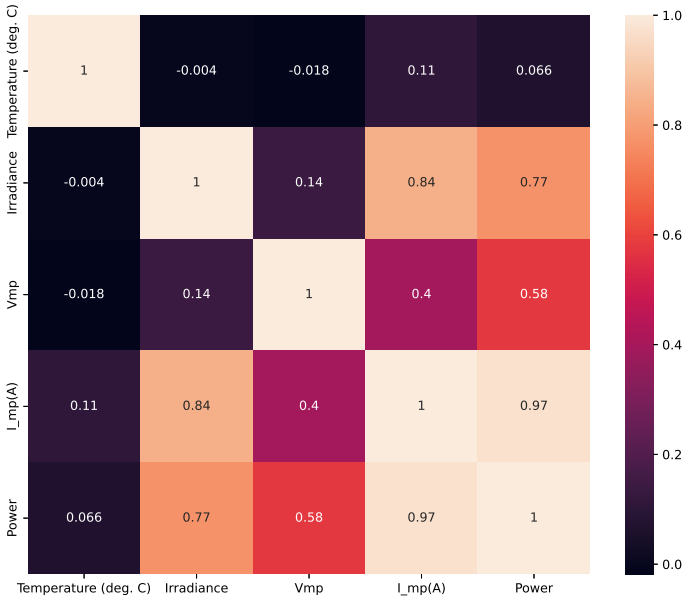



Figure 3.3: Heatmap of variables.

The exploratory data analysis (EDA) highlights meaningful dependencies among temperature, irradiance, voltage, current, and power. The principal findings are summarized below:

- Voltage decreases as temperature increases, showing a strong negative relationship.
- Irradiance has a direct and positive influence on both voltage and the power output.
- Current exhibits a positive correlation with power, acting as the main contributor to its variation.

Such outcomes are fundamental for understanding the operational dynamics of the photovoltaic system reflected in the dataset, while also providing a platform for advanced analyses and the formulation of optimization methods to maximize energy yield under diverse environmental circumstances.

## 3.2 Model Development and Parameter Adjustment

To normalize the input features, the *MinMaxScaler* function from the *scikit-learn* library was employed, scaling all variables to the [0, 1] interval.

- The scaler was first fitted to the dataset and subsequently applied to transform the feature values.

For model validation, the dataset was partitioned into training and testing sets.

- Using the `train_test_split` function, 80% of the data was assigned for training and 20% for testing.
- A random seed of 42 was set to ensure reproducibility.

An *MLPRegressor* from *scikit-learn* was adopted to capture the relationship between the input variables and the target power output.

- The neural network architecture was composed of two hidden layers containing 64 and 32 neurons, respectively.
- The ReLU activation function was chosen to account for nonlinearities.
- Training was carried out using the Adam optimizer, with a maximum of 200 epochs.
- The random state was again fixed at 42 to ensure experimental consistency.

The ANN was trained on the training set as follows:

- The *fit* method was applied to the training data.

Once trained, the model was evaluated to assess its predictive capability.

- Predictions were obtained using the *predict* method.
- Performance was assessed using the following error metrics:
  - Mean Squared Error (MSE),
  - Mean Absolute Error (MAE),
  - and the Coefficient of Determination ($R^2$).

The evaluation results were presented both numerically and graphically:

- Comparative plots were generated to contrast predicted values with actual ones.
- Metric results were also displayed in the console.

This procedure guarantees that the ANN is effectively trained and validated under partial shading scenarios, providing a reliable framework for forecasting photovoltaic power generation.

Figure 3.4 shows the schematic diagram of the developed ANN for MPPT prediction in a PV system. The architecture integrates the input parameters (Temperature, Irradiance, $V_{mp}$, $I_{mp}$), two hidden layers with 64 and 32 neurons, and an output layer dedicated to estimating the generated power.

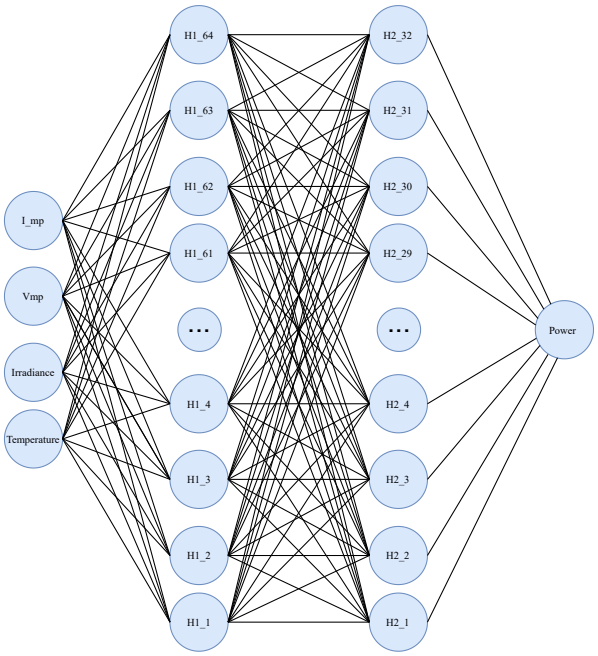



Figure 3.4: Conceptual representation of the proposed ANN architecture.

The Artificial Neural Network (ANN) architecture was configured with predefined hyperparameters to ensure optimal performance. Unlike trainable parameters, which are updated iteratively during the learning phase, hyperparameters are fixed prior to training and directly influence the convergence behavior and generalization capability of the model [59]. Proper hyperparameter tuning is therefore critical, as it determines the configuration that minimizes error and enhances predictive accuracy [60].

In this implementation, the network architecture included two hidden layers composed of 64 and 32 neurons, respectively, allowing the model to capture complex nonlinear relationships between the input variables and the output. The Rectified Linear Unit (ReLU) activation function was selected because of its effectiveness in deep networks and its capacity to reduce the vanishing gradient issue. Model training was performed using the Adam optimizer, which employs adaptive learning rates to speed up convergence and enhance stability, with a training limit of 200 epochs. To guarantee reproducibility, the random seed was fixed at 42. The source code for this configuration is presented in Annex .1.

## 3.3 Optimization

Designing the architecture of a neural network is crucial for achieving high predictive accuracy. A key element of this design involves selecting the number of neurons in each hidden layer, since this factor directly influences the model's capacity to learn nonlinear and complex patterns in the data. Bio-inspired optimization methods offer an effective approach to explore wide search spaces by dynamically adjusting the neuron count per layer. In this way, such algorithms determine optimal configurations that enhance network performance through the systematic expansion or reduction of hidden layer sizes, enabling the architecture to adapt to the particular problem being addressed.

The selection of hidden layer sizes is particularly relevant because these layers largely dictate how the ANN encodes, transforms, and generalizes information. Focusing the optimization process on this aspect is both practical and advantageous for several reasons:

- **Learning Capacity:**
  - *Representational Complexity:* The number of neurons defines the network's ability to approximate functions and capture complex data structures. Larger layers increase representational capacity, enabling the ANN to learn more sophisticated features.
  - *Capacity–Generalization Trade-off:* Adjusting hidden layer sizes allows balancing expressiveness with generalization, ensuring that the network is neither underfitted nor excessively complex, thereby reducing the risk of overfitting.
- **Computational Efficiency:**
  - *Parameter Control:* The size of hidden layers directly influences the number of trainable parameters, which affects memory usage and computational cost. This is especially important when working with limited hardware resources.
  - *Simplification of the Search Space:* Concentrating on hidden layer sizes narrows the optimization scope, allowing computational resources to be directed towards fine-tuning this critical component rather than exploring the entire architecture.
- **Influence on Learning Dynamics:**
  - *Gradient Propagation:* Properly sized hidden layers improve the flow of gradients during backpropagation, reducing the likelihood of training instabilities such as vanishing.
  - *Adaptability to Data:* Adjusting the number of neurons enables the ANN to adapt its representational power to the complexity of the input data, improving the accuracy of predictions.
- **Flexibility and Adaptation:**
  - *Ease of Experimentation:* Adjusting the size of hidden layers offers a straightforward but powerful method to explore different configurations and assess their effect on performance.
  - *Context-Specific Optimization:* When the overall architecture is already adequate, tuning hidden layer sizes provides a precise mechanism for refining performance without altering the entire design.

- **Reduction of Design Complexity:**
  - *Streamlined Decision-Making:* Limiting the optimization to hidden layer sizes decreases the number of architectural variables, simplifying the design and enabling a more targeted and efficient optimization process.

In conclusion, tuning the size of hidden layers represents a targeted and efficient approach that can significantly enhance the performance of ANNs, especially when the overall network topology is already well-suited to the problem. The primary goal of this optimization is to reduce prediction errors through a composite loss function that combines MSE, MAE, and $R^2$. By lowering the first two error measures while maximizing $R^2$, the model not only minimizes discrepancies between predicted and actual values but also effectively captures the variability in the data, leading to more accurate and dependable predictions. The optimization task is mathematically expressed in Equation (3.4):

$$\min_{\mathbf{h}} \text{Objective}(\mathbf{h}) = \text{MSE}(\mathbf{h}) \tag{3.4}$$

where:

- $\mathbf{h} = (h_1, h_2)$ denotes the hyperparameter vector, indicating the number of neurons assigned to each hidden layer of the ANN.
- $\text{MSE}(\mathbf{h}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i(\mathbf{h}))^2$ corresponds to the Mean Squared Error, adopted as the cost function minimized by the GWO algorithm.
- $y_i$ represents the actual power measured for the $i$-th sample in the test dataset.
- $\hat{y}_i(\mathbf{h})$ is the predicted power obtained from the ANN configured with hyperparameters $\mathbf{h}$.
- $n$ is the number of test instances used in the evaluation.

The optimization process seeks the most appropriate hidden layer configuration (i.e., neuron allocation per layer) by means of an iterative routine aimed at minimizing MSE and MAE, while maximizing $R^2$. The procedure can be described as follows:

1. **Initialization:**
   - A population of candidate solutions is generated, each one corresponding to a different hidden layer setup $\mathbf{h}$.
2. **Evaluation:**
   - Each candidate $\mathbf{h}$ is used to train the ANN with the training dataset.
   - Model quality is assessed on the test set by calculating the objective function based on the *MSE*.
3. **Selection:**
   - The performance of all candidates is compared, and the configuration $\mathbf{h}$ yielding the lowest error is retained for that iteration.

4. **Update:**
    - Based on evaluation results, the algorithm generates a new population by modifying the hidden layer sizes **h**.
5. **Iteration:**
    - The cycle of evaluation, selection, and update is repeated until a maximum number of iterations is reached or no further improvements are observed.
6. **Convergence:**
    - After several iterations, the procedure converges to an optimal solution $\mathbf{h}^*$ that minimizes prediction errors while maximizing $R^2$.

At the maximum power point (MPP), voltage and current values are obtained from the PV module performance curve. The ANN, trained with inputs such as irradiance and temperature, predicts these operating points. Once optimized, the model provides reliable estimates of voltage and current at the MPP, ensuring maximum efficiency of the photovoltaic system.

## 3.4 Configuration bioinspired algorithm

The GWO was employed to optimize the key hyperparameters of the ANN, focusing mainly on the hidden layer sizes. The corresponding implementation can be found in Annex .2.

- The algorithm begins with a population of 25 wolves.
- The hyperparameter search range is set between [10, 10] and [100, 100].
- A maximum of 100 iterations is executed.

*Optimization Workflow*
The GWO optimization process can be summarized in the following stages:

- **Initialization:** Wolves are randomly positioned within the predefined search limits.
- **Fitness Evaluation:** Each candidate solution is evaluated based on the Mean Squared Error (MSE) of ANN predictions. The ANN is trained using the hidden layer sizes proposed by each wolf, and the resulting MSE on the test data is computed.
- **Position Updating:** Wolf positions are adjusted according to the alpha, beta, and delta leaders, integrating the best solutions identified so far:
    - Calculate the distances relative to the alpha, beta, and delta wolves.
    - Update each wolf's position by combining these distances.
- **Exploration–Exploitation Trade-off:** The parameter $a$, which decreases linearly with iterations, regulates the balance between global search and local refinement.
- **Convergence Tracking:** The optimization is monitored using MSE, MAE, and $R^2$, ensuring stability and accuracy throughout the process.

*Evaluation of the Optimized Model*

After completing the optimization, the ANN configured with the best hyperparameters found by GWO is trained and validated:

- The network is trained using the optimal hidden layer configuration and the training dataset.
- Predictions are generated for the test dataset.
- Model performance is assessed using MSE, MAE, and the $R^2$ metric.

## 3.5 Experimental Plant and Solar Collector

To generate real-world data and validate the proposed methodology, an existing experimental platform was used, consisting of a photovoltaic (PV) module mounted on a one-degree-of-freedom (1-DoF) solar collector. This plant was originally designed and constructed by [2], and has been adapted in this research to integrate the proposed MPPT prediction model. The platform provides a controlled yet realistic environment to evaluate the ANN model optimized with GWO.

### Photovoltaic Module

The PV module corresponds to a commercial crystalline silicon panel, selected for its availability and representativeness of small-scale solar applications.

### One-Degree-of-Freedom Solar Collector

The 1-DoF solar collector allows tilting about a single axis (altitude) to emulate solar tracking or impose controlled tilt angles during experiments. Its main features include:

- A mechanical structure with adjustable inclination.
- A motor and driver system for smooth and precise actuation.
- Position feedback via encoder or potentiometer, enabling repeatable configurations.
- Mounts for environmental and electrical sensors to measure irradiance, temperature, voltage, and current simultaneously.

This design makes it possible to reproduce different daily solar conditions and to introduce controlled perturbations such as partial shading.

Figure 3.5: One-degree-of-freedom solar collector used in the experimental plant [2].

The experimental solar collector was installed at the *Centro Universitario* of the Universidad Autónoma de Querétaro, located in Santiago de Querétaro, México. Its exact geographical coordinates are latitude **20.590530** and longitude **-100.411805**, which allow precise identification of the site where the measurements were carried out. Figure 3.6 shows the location of the solar collector within the university facilities.

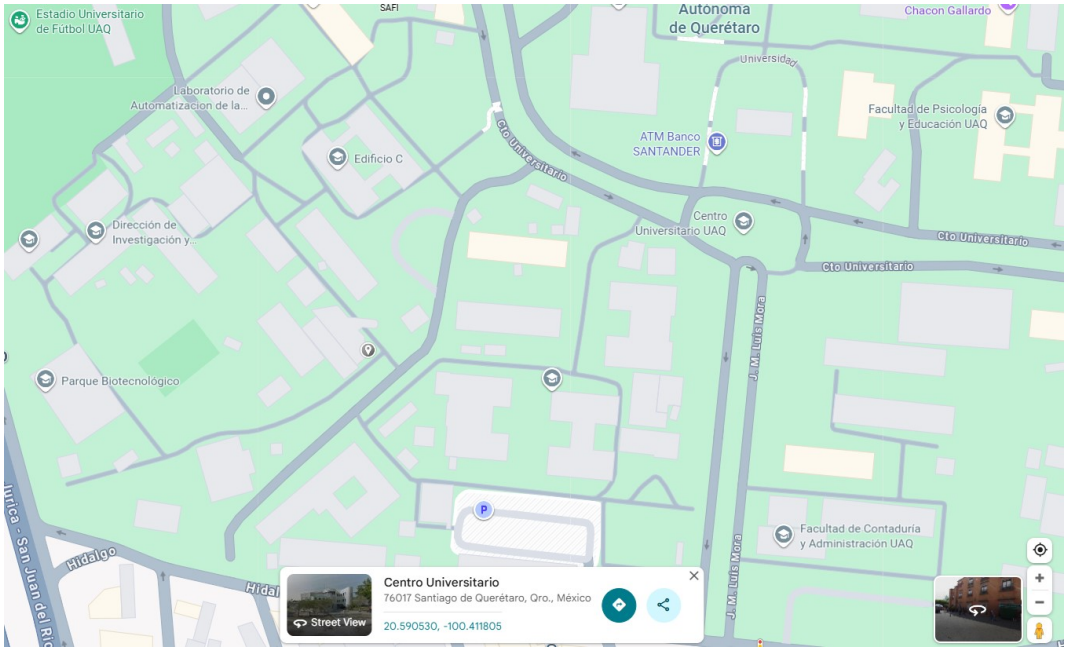



Figure 3.6: Location of the solar collector

**Instrumentation and Data Acquisition**

The platform incorporates sensors for measuring irradiance, module temperature, ambient humidity, voltage, and current. Table 3.1 summarizes their specifications.

Table 3.1: Specifications of sensors used in the experimental setup.

| **Parameter** | **Solar Radiation** | **Ambient Temp. and Humidity** | **Current, Voltage, and Power** |
|---|---|---|---|
| Nomenclature | R | $T_a$, $H_a$ | I, P |
| Units | [W/m$^2$] | [°C], [%] | [A], [W] |
| Instrument | Pyranometer | DHT22 | INA219 |
| Operating Voltage | 7–30 VDC | 3–6 VDC | 3–5.5 VDC |
| Range | 0–1800 W/m$^2$ | 40–80 °C; 0–100 % | 0–3.2 A |
| Accuracy | ±5 % | ±0.5 °C; ±2 % | ±1 % |
| Resolution | 1 W/m$^2$ | 0.1 °C; 0.1 % | 0.01 A; 0.1 W |
| Response Time | 10 s | 2 s | ¡ 1 s |
| Protocol | ADC, 0–5 VDC | Single-bus (bidirectional) | I2C |
| Signal Type | Digital | Digital | Digital |

### 3.5.1 Sample Selection and Data Acquisition

The acquisition window was defined from 09:00 to 17:59, covering the period of effective solar radiation at the experimental site. This range ensures that the dataset includes the complete daily irradiance profile, from the morning increase to the midday peaks and the afternoon decline. According to [61], the evaluation of solar resources is based on the concept of Sun Peak Hours (SPH), which represent the equivalent number of hours per day with a constant irradiance of 1000 W/m$^2$. Although SPH is a standardized metric, in practice it corresponds to the portion of the day when the irradiance remains within usable levels for photovoltaic generation. Therefore, restricting measurements to the 09:00–17:59 interval guarantees the capture of the effective solar resource while excluding periods of low-angle irradiance (early morning and late evening) that contribute minimally to energy production.

The monitoring campaign generated approximately **41,000 data points**, structured into **8,200 rows** and **5 columns**, corresponding to the variables: solar irradiance, module temperature, ambient humidity, PV voltage, and PV current. Measurements were acquired at a rate of one sample per minute, and the dataset was subsequently aggregated and averaged across multiple days to produce representative daily radiation curves. This comprehensive database forms the basis for training and validating the proposed ANN model optimized with GWO.

## 3.6 Model development and training with real data

After the acquisition and structuring of the dataset, the next stage of the methodology consisted of designing and training a predictive model based on an Artificial Neural Network (ANN). This process was carried out systematically, ensuring reproducibility and alignment with the objectives of the study.

### Data Preparation

The dataset was first reviewed to eliminate inconsistencies such as missing values. The variables were divided into two groups: input features and output labels. The inputs consisted of temperature, solar irradiance, voltage, and current, while the output corresponded to the power delivered by the photovoltaic module. To ensure uniformity in scale and facilitate convergence during training, the input variables were normalized within a fixed range using a feature-scaling technique.

### Dataset Partitioning

Once preprocessed, the dataset was split into two subsets: a training set used to adjust the model parameters, and a testing set reserved for evaluating the model's performance on unseen data. This separation guarantees an unbiased assessment of the generalization capability of the network.

### Neural Network Architecture

A Multi-Layer Perceptron (MLP) was employed as the prediction model. Its structure consisted of an input layer aligned with the feature set, several hidden layers designed to capture nonlinear patterns, and a single output layer to approximate the target value. Rectified Linear Unit (ReLU) activations were applied in the hidden layers to enhance convergence, while the output layer used a linear activation to generate continuous outputs.

### Compilation and Training

The ANN was compiled using an optimization algorithm suitable for gradient-based learning. A loss function was defined to measure the difference between the predicted and actual outputs, while additional metrics were included to monitor the learning progress. The training process consisted of iterative weight updates over several epochs, using mini-batches of data to balance computational efficiency and stability. A fraction of the training data was reserved for validation, enabling the detection of overfitting and the adjustment of hyperparameters when necessary.

### Evaluation and Model Storage

At the end of the training phase, the model was evaluated using the test set. Predictions were compared against actual values through performance metrics that quantify error magnitude and explanatory power. Finally, the trained ANN was exported and stored in a standardized format, making it available for integration into the experimental platform for real-time validation.

### Embedded System Deployment

To validate the feasibility of the proposed approach in real-time applications, the trained ANN model was deployed on a Raspberry Pi 4 platform. This embedded system was selected due to its low cost, portability, and sufficient computational capacity to execute neural network inference efficiently. The exported model was integrated into a Python-based application running on the Raspberry Pi 4, which managed data acquisition from the photovoltaic module, preprocessing of the signals, and execution of predictions in real time. The embedded deployment demonstrated low latency and acceptable resource consumption (CPU and RAM), confirming the practicality of using the Raspberry Pi 4 for predictive tasks in renewable energy environments.

CHAPTER 4

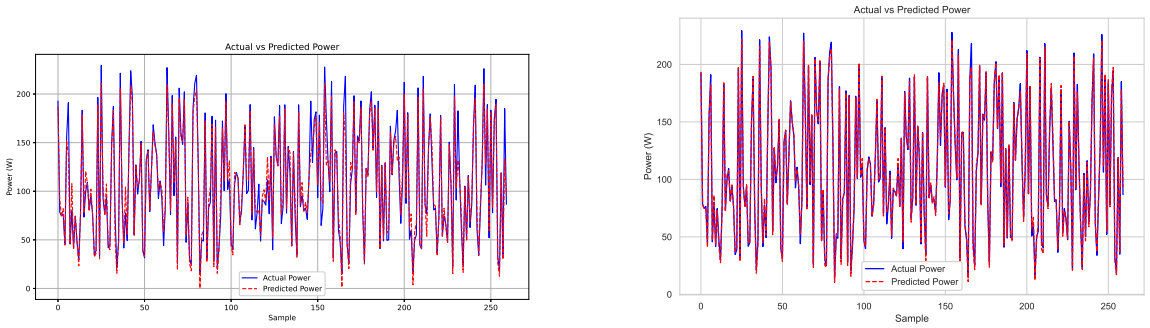
# Results and discussion

Table 4.1 summarizes the experimental configuration. The ANN was first defined as the baseline model, and then the GWO was applied to optimize the number of neurons in the hidden layers. This approach allowed a straightforward comparison between the baseline ANN and the optimized version.

Figure 4.1 displays the comparison between measured and predicted photovoltaic power. Subfigure 4.1a depicts the outputs from the baseline ANN, whereas Subfigure 4.1b illustrates the predictions obtained with the GWO-enhanced ANN.

Table 4.1: Comparison of hyperparameter settings between the baseline ANN and the ANN optimized with GWO.

| **Hyperparameter** | **Baseline ANN** | **GWO-Optimized ANN** |
|---|---|---|
| $n$ (population size) | – | 25 |
| $lb$ (search lower bound) | – | [10, 10] |
| $ub$ (search upper bound) | – | [100, 100] |
| Maximum iterations | – | 50 |
| $hls$ (hidden layer structure) | (64, 32) | Adaptive |
| Activation function | ReLU | ReLU |
| Solver | Adam | Adam |
| Epochs | 200 | 200 |
| Random seed | 42 | 42 |




(a) Predictions using ANN

(b) Predictions using GWO-optimized ANN

Figure 4.1: Comparison of real photovoltaic power values against predictions from the base ANN and the GWO-enhanced ANN.

The GWO was employed to optimize the number of neurons in each hidden layer of the ANN for power prediction. This section provides a thorough evaluation of its performance, considering error metrics, convergence behavior, prediction reliability, and computational cost.

## 4.1 Performance Metrics

Table 4.2 reports the performance measures obtained with the ANN optimized through GWO.

Table 4.2: Performance indicators of the ANN optimized with GWO.

| **Metric** | **Value** |
|---|---|
| MSE | 11.9487 |
| MAE | 2.4552 |
| $R^2$ | 0.9964 |
| Neurons (Layer 1) | 66 |
| Neurons (Layer 2) | 100 |
| Execution Time | 1198.99 s |

The resulting model exhibited very low mean squared error (MSE) and mean absolute error (MAE), while the coefficient of determination ($R^2$) confirmed an excellent agreement between the predicted and observed values. The chosen neural architecture provided a good compromise between accuracy and generalization, and the execution time remained reasonable given the iterative nature of the optimization process.

## 4.2 Convergence Behavior and Computational Effort

The progression towards convergence in GWO optimization is illustrated in the following figures, each one representing the behavior of a specific performance metric or computational indicator.

Figure 4.2 shows the convergence of the MAE. It can be observed that the MAE decreases rapidly during the first iterations and then stabilizes at a constant low value, confirming that the optimizer effectively minimized local deviations between predicted and actual outputs.

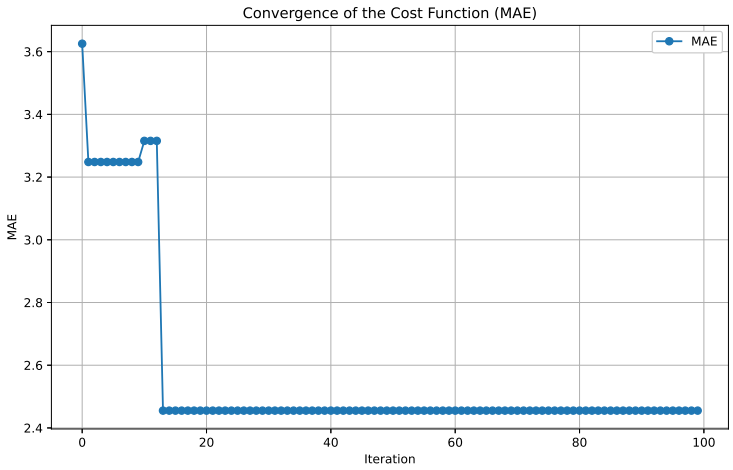



Figure 4.2: Convergence of MAE during GWO optimization.

Figure 4.3 presents the convergence of the mean squared error (MSE). Similar to the MAE, the MSE shows a sharp reduction at the early stages of optimization, followed by stabilization. This confirms that the model avoids large deviations and penalizes outliers effectively.

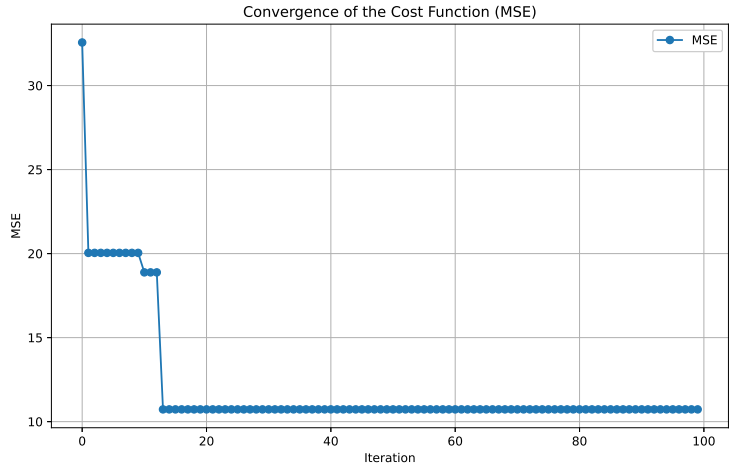



Figure 4.3: Convergence of MSE during GWO optimization.

Figure 4.4 illustrates the evolution of the $R^2$. The metric steadily increases with each iteration, approaching values close to unity. This demonstrates the model's ability to capture the variance of the data and achieve a strong fit with the measured outputs.

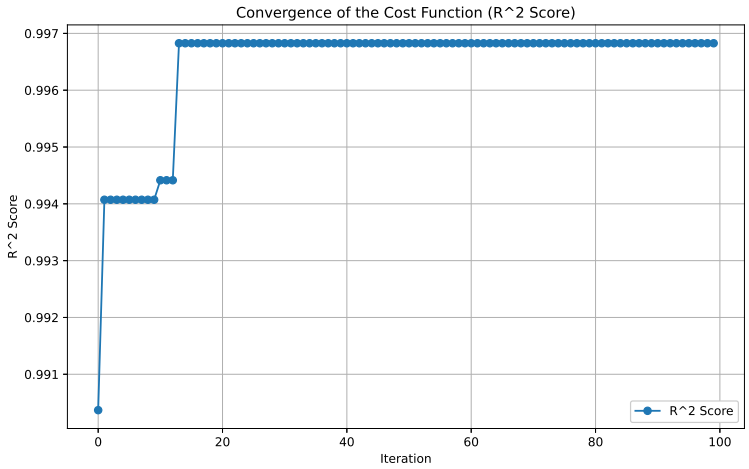



Figure 4.4: Convergence of $R^2$ during GWO optimization.

In addition, Figure 4.5 shows the computational effort of the optimization process as a function of iterations. The execution time grows linearly with the number of iterations, but the increase is smooth and consistent, confirming the algorithm's efficiency and scalability for practical applications.

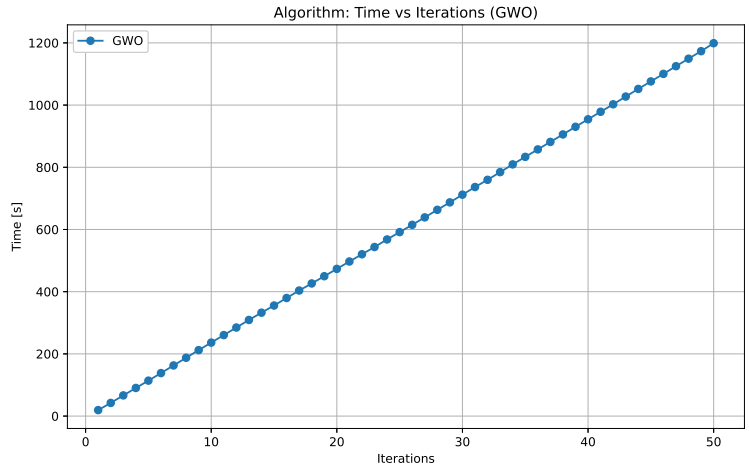



Figure 4.5: Execution time vs. iterations during GWO optimization.

Finally, Figure 4.6 illustrates how the GWO algorithm adjusts the number of neurons in the hidden layers during optimization. Both layers initially fluctuate as the algorithm explores the search space, but they quickly stabilize into an optimal configuration. This reflects the balance between exploration and exploitation inherent to GWO, ensuring that the network achieves high predictive performance without unnecessary architectural complexity.

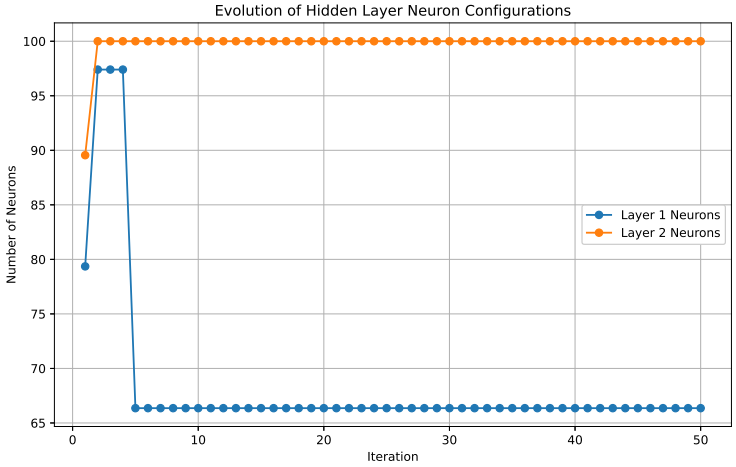



Figure 4.6: Evolution of ANN hidden layer structures under GWO-based optimization.

Overall, the convergence analysis demonstrates that the Grey Wolf Optimizer provides a stable, reliable, and computationally feasible approach for tuning the ANN model. The combined trends of MAE, MSE, and $R^2$ validate the accuracy of the predictions, while the neuron evolution and execution time confirm the practicality of the algorithm for real-world implementations.

Figure 4.7 presents the cumulative average radiation profiles for the months of July and August. The results highlight the dynamic nature of irradiance throughout the day, with fluctuations caused by environmental factors such as passing clouds and partial shading. Peaks are especially evident around midday, when the solar altitude and panel orientation favor maximum energy incidence, while the morning and afternoon periods capture the ascending and descending phases of irradiance.

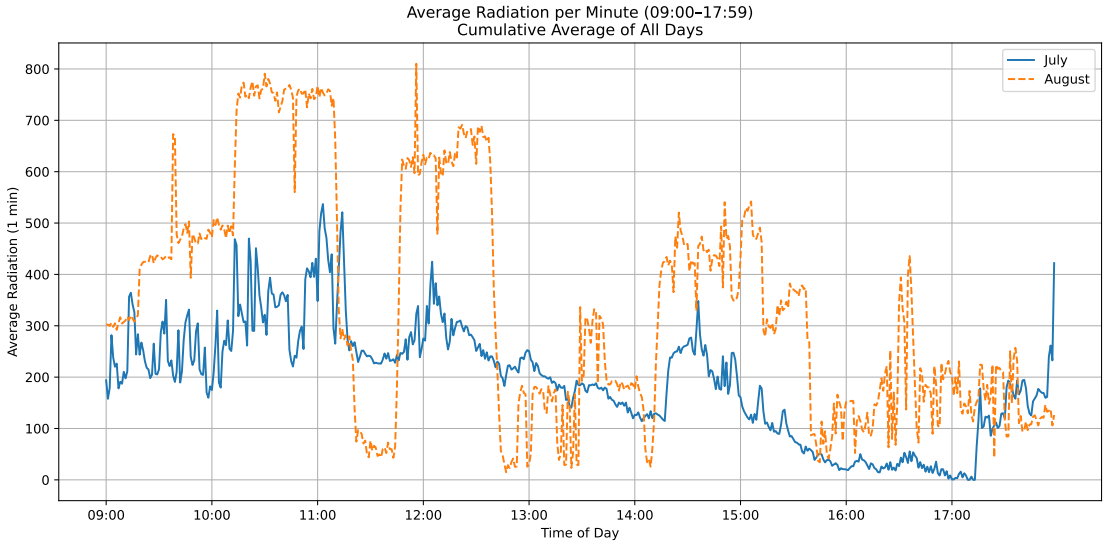



Figure 4.7: Average radiation per minute (09:00–17:59) for July and August, cumulative average across all days.

Figure 4.8 provides a complementary view of the irradiance distribution by hour for July and August. The results clearly show that the dispersion of the data was greater in August compared to July. This behavior can be attributed to the distinct climatic conditions observed during the monitoring period. July was characterized by several cloudy and rainy days, which led to more stable but generally lower irradiance values. In contrast, August presented longer periods of clear skies and intense solar radiation, resulting in higher irradiance peaks and greater variability across the measured hours.

The boxplot analysis also highlights how the irradiance behavior differs throughout the day. During the morning hours (09:00–11:00), both July and August show relatively moderate radiation values, but the spread in August is significantly larger, reflecting the presence of sunny days with sharp increases in irradiance. At midday (12:00–14:00), the effect of solar altitude is evident, as the distributions in August reach higher medians and wider interquartile ranges, confirming stronger and more variable radiation. Finally, during the afternoon (15:00–17:00), the decreasing solar angle reduces the irradiance levels, although August still shows a higher dispersion compared to July.

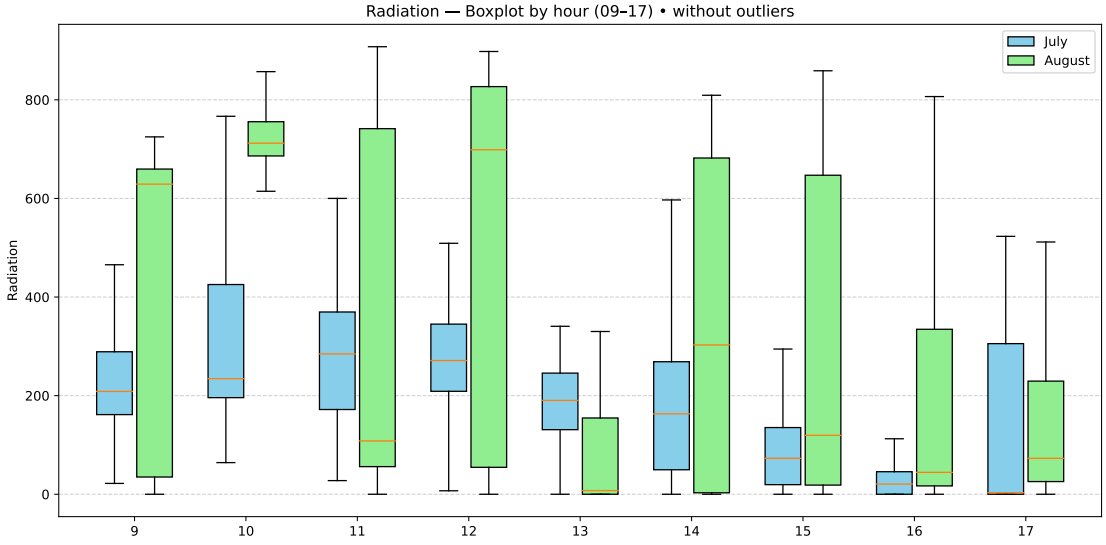



Figure 4.8: Boxplot of hourly radiation values (09:00–17:00) for July and August.

These observations emphasize the importance of incorporating data from both months into the predictive model. The contrasting conditions between July and August provide a diverse and realistic dataset that challenges the ANN model and ensures that the optimization process with GWO captures not only stable but also highly variable scenarios. As a result, the robustness of the predictive framework is enhanced for real-world photovoltaic applications.

In addition to irradiance analysis, the monitoring of all sensed variables is shown in Figure 4.9. The plots provide a time-series representation of solar radiation, module temperature, current, and power, allowing for a clear visualization of their temporal behavior during the effective solar hours.

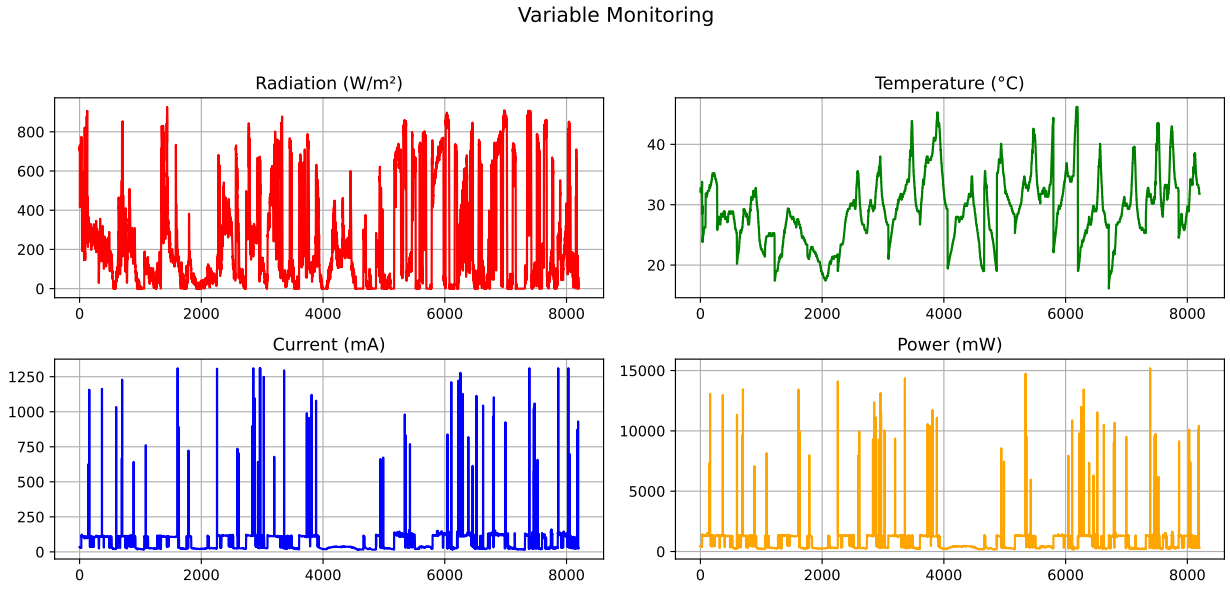



Figure 4.9: Monitoring of experimental variables: radiation, temperature, current, and power.

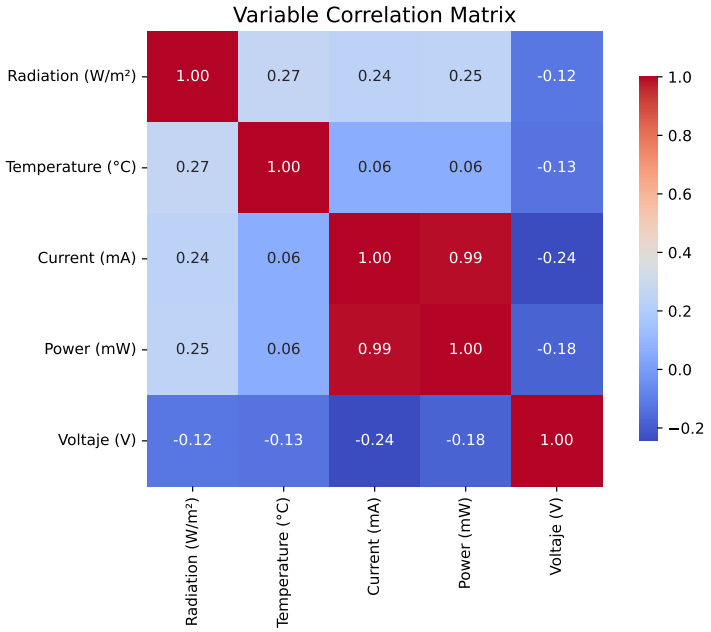



Figure 4.10: Correlation matrix of the measured variables.

To further analyze the relationships among the measured variables, a correlation matrix was computed. As illustrated in Figure 4.10, irradiance and current exhibit a strong positive correlation with power, while temperature shows a negative correlation with voltage. These insights confirm the physical interdependence between environmental conditions and the electrical behavior of the

PV module.

This acquisition method provides a realistic representation of solar availability under real operating conditions. By recording continuously over the effective solar hours, the dataset incorporates natural variability and environmental perturbations, ensuring that the ANN model optimized with GWO is trained and validated with data representative of actual photovoltaic operation.

### 4.2.1 Preconfigured ANN Model Tested with Real Data (PC Evaluation)

Once the artificial neural network was defined with its preconfigured architecture of two hidden layers (66 and 99 neurons), the model was trained and evaluated using the experimental dataset obtained from the solar collector. This evaluation was essential to validate the capacity of the preconfigured ANN to generalize under real operating conditions, moving beyond controlled simulations or public datasets.

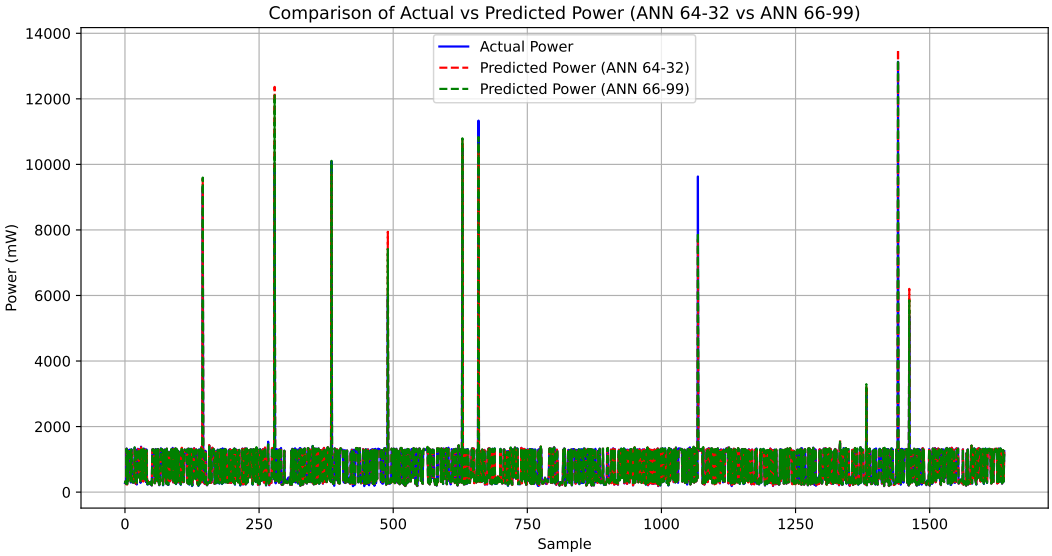



Figure 4.11: Comparison of measured and predicted power obtained from the baseline ANN (64–32 neurons) and the tuned ANN (66–99 neurons), both evaluated on real data using a PC.

Figure 4.11 shows the superimposed predictions of the baseline ANN and the preconfigured ANN compared with the actual power measured during the experimental campaign. The baseline ANN (64–32 neurons) was able to reproduce the general trend of the power curve, but discrepancies became evident in periods of high variability, particularly during sudden irradiance fluctuations. In contrast, the preconfigured ANN (66–99 neurons) demonstrated much closer alignment with the actual power, accurately following both smooth variations and abrupt peaks. This highlights the benefit of increasing the network's representational capacity, as the larger number of neurons allows the model to better capture the nonlinear behavior of the photovoltaic system under real operating conditions.

The numerical results in Table 4.3 further confirm this improvement. The preconfigured ANN reduced the MSE by 19.8% (from 9549.50 to 7656.68) and the MAE by 23.6% (from 17.80 to 13.61)

compared to the baseline ANN. Similarly, the $R^2$ increased from 0.9862 to 0.9889, and the overall prediction accuracy improved from 98.62% to 98.89%.

Table 4.3: Performance metrics of the baseline ANN (64–32 neurons) and the preconfigured ANN (66–99 neurons) tested with real data on PC.

| **Metric** | **ANN (64–32)** | **ANN (66–99)** |
|---|---|---|
| MSE | 9549.50 | 7656.68 |
| MAE | 17.80 | 13.61 |
| $R^2$ | 0.9862 | 0.9889 |
| Prediction Accuracy | 98.62% | 98.89% |

### 4.2.2 Preconfigured ANN Model Deployed on Raspberry Pi 4 (Embedded Evaluation)

Beyond offline testing on a PC, the models were also embedded and executed on a Raspberry Pi 4 platform to evaluate their feasibility under hardware constraints. The results confirmed that both architectures maintained high accuracy, with the optimized ANN again outperforming the baseline ANN.

Table 4.4: Performance metrics of the baseline ANN (64–32 neurons) and the preconfigured ANN (66–99 neurons) tested on Raspberry Pi 4.

| **Metric** | **ANN (64–32)** | **ANN (66–99)** |
|---|---|---|
| MSE | 9575.08 | 7121.51 |
| MAE | 16.31 | 14.73 |
| $R^2$ | 0.9891 | 0.9919 |
| Prediction Accuracy | 98.91% | 99.19% |

The embedded evaluation revealed that the preconfigured ANN achieved lower MSE and MAE values, along with improvements in $R^2$ and overall accuracy compared to the baseline model. Importantly, the findings show that the proposed approach is not only effective in offline computational experiments but also performs reliably on embedded hardware with constrained resources. This confirms the feasibility of applying the ANN–GWO framework for real-time MPPT prediction in photovoltaic systems.

Several key outcomes were identified. The optimized ANN, configured with 66–99 neurons, reduced the MSE by approximately 25.6% relative to the baseline ANN (decreasing from 9575.08 to 7121.51). This reduction reflects the model's ability to better minimize large deviations between predicted and actual values, which is especially relevant under dynamic irradiance conditions that produce sudden peaks. Likewise, the MAE dropped by nearly 9.7% (from 16.31 to 14.73), indicating an improvement in point-by-point predictive accuracy.

The $R^2$ coefficient further supports this progress. While the baseline ANN already achieved a strong value of 0.9891, the optimized ANN increased it to 0.9919. Though the numerical difference (0.28%) appears small, it signifies a notable gain in the proportion of variance captured by the

model, particularly given the evaluation constraints of a Raspberry Pi 4. Prediction accuracy also improved consistently, rising from 98.91% to 99.19%.

These outcomes have important implications. First, they show that GWO-based optimization enhances not only offline model performance but also real-time robustness when implemented on embedded platforms with limited processing power. This demonstrates that the ANN–GWO solution is computationally efficient enough to operate in real-world conditions without sacrificing precision. Second, the reductions in MSE and MAE, together with the improvement in $R^2$, confirm the optimized model's stronger generalization under variable environmental conditions, a critical factor for reliable MPPT tracking.

In conclusion, the embedded evaluation validates that the optimized ANN model preserves superior accuracy and robustness even when executed on cost-effective embedded devices. This reinforces its practical potential for photovoltaic controllers, narrowing the gap between algorithmic design and deployment in operational field scenarios.

CHAPTER 5

# Conclusions

In this section, the impacts, conclusions, and key points of this project are described, providing a comprehensive perspective on its contributions and implications.

The results of this research lead to several key conclusions. First, the acquisition of experimental data using the solar collector produced a robust and representative dataset that captured the real behavior of photovoltaic systems, including the variability caused by irradiance fluctuations and partial shading. This dataset was essential to validate the predictive models under conditions that extend beyond the idealized assumptions of simulation environments.

Second, the baseline ANN model confirmed the feasibility of applying artificial intelligence techniques for MPPT. However, its predictive capacity was constrained by the manual selection of hyperparameters, limiting its ability to adapt to the nonlinear and highly dynamic nature of real operating conditions.

The integration of the GWO with the ANN represented a decisive advancement. By automating hyperparameter tuning, the model achieved significant improvements in both accuracy and stability. The convergence analysis confirmed that GWO successfully balances exploration and exploitation, leading to architectures that are sufficiently complex to capture system dynamics without introducing unnecessary computational overhead.

Most importantly, the optimized ANN model not only reached a high level of predictive accuracy—with an $R^2$ close to 0.99 and an accuracy above 98.8%—but also demonstrated robustness when confronted with real, noisy, and variable data. In addition, its deployment on a Raspberry Pi 4 confirmed that the proposed approach is not only accurate in offline computational environments but also efficient and reliable when embedded in low-cost hardware with limited resources. This demonstrates its capability for real-time MPPT prediction, effectively linking algorithmic design with practical deployment in the field.

In general, this study shows that integrating neural networks with bio-inspired methods algorithms such as GWO provides a powerful strategy to advance maximum power point prediction. Beyond numerical improvements, the proposed methodology contributes to the broader objective of making renewable energy systems more efficient, intelligent, and reliable. In this sense, the research not only enriches the state of the art in computational intelligence applied to renewable energies but also lays the foundation for practical applications that can accelerate the transition toward sustainable energy systems.

Finally, it can be concluded that the central hypothesis of this thesis has been fully confirmed: the integration of bio-inspired optimization with neural networks significantly improves the predic-

tive capacity and robustness of MPPT models when applied to real photovoltaic data. Unlike other studies in the literature, which often remain limited to simulation or controlled test conditions, this work validated the methodology through both experimental implementation and embedded deployment. This contrast reinforces the originality and scientific and technological contribution of the research, showing that the proposed approach not only achieves competitive accuracy compared to state-of-the-art methods but also proves its effectiveness under real operating conditions, thereby bridging the persistent gap between simulation and real-world deployment.

## 5.1 Future Work

Even though the objectives of this thesis were successfully achieved, several opportunities for further research remain open. One of the most relevant directions is to extend the embedded implementation. In this work, the proposed ANN optimized with GWO was successfully deployed on a Raspberry Pi 4, confirming the feasibility of running the model in real time on low-cost hardware. Future research could expand this effort by testing the model on other platforms such as FPGAs, microcontrollers, or AI-dedicated processors, where computational efficiency, energy consumption, and scalability can be more rigorously evaluated. These studies would provide valuable insights into the trade-offs between performance and resource requirements, which are crucial for practical MPPT controllers.

Another promising avenue is the expansion of the experimental dataset. While the current database captures variability in irradiance, temperature, and partial shading during the monitoring campaign, future datasets should extend to longer time horizons, different seasons, and multiple geographic locations. Incorporating additional environmental variables such as wind speed, dust accumulation, or solar incidence angle would enrich the input space and enhance the robustness and generalization capacity of the predictive model.

It would also be valuable to broaden the optimization framework. In a previous publication by the author, GWO was benchmarked against Particle Swarm Optimization (PSO), Cuckoo Search (CS), and the Salp Swarm Algorithm (SSA), consistently demonstrating superior convergence and accuracy. Nevertheless, future work could explore newer or hybrid metaheuristic algorithms, including swarm-based and evolutionary approaches, to identify combinations that accelerate convergence, reduce computational costs, or improve stability under highly variable conditions.

Hybrid and ensemble learning methods represent another line of investigation. By integrating multiple neural network architectures or combining optimization strategies, it may be possible to design MPPT predictors that adapt more flexibly to sudden irradiance fluctuations, outperforming single-model approaches in highly dynamic scenarios.

Finally, testing should be extended beyond the current experimental setup. Hardware-in-the-loop simulations and direct deployment in grid-connected photovoltaic plants would provide stronger validation of the methodology under industrial-scale conditions. Such implementations would close the gap between algorithmic development and real-world adoption, accelerating the transition toward intelligent MPPT controllers that are efficient, adaptive, and reliable in diverse photovoltaic environments.

## 5.2 Contributions

This section outlines the most relevant impacts generated by the project, offering an integrated view of its scientific, technological, social, and economic contributions.

### 5.2.1 Scientific contribution

This work contributes to the renewable energy field through the application of bio-inspired algorithms for the optimization of artificial neural networks in maximum power point tracking (MPPT) prediction of photovoltaic systems. The integration of the Grey Wolf Optimizer (GWO) with ANN-based regression models advances the state of the art in hyperparameter optimization and intelligent control applied to clean energy systems.

### 5.2.2 Technological contribution

The development of an optimized predictive model represents a tangible technological contribution, as it was successfully embedded and executed on a Raspberry Pi 4 platform. This implementation confirms the feasibility of deploying the proposed ANN–GWO approach in real-time environments with limited computational resources. By running directly on embedded hardware, the model demonstrates its potential to be integrated into MPPT control systems for photovoltaic plants, improving energy conversion efficiency and enabling controllers to be more accurate and adaptable under varying irradiance and partial shading conditions.

Beyond validating the practicality of the proposed methodology, this achievement establishes a solid foundation for the design of intelligent, low-cost, and portable embedded systems for renewable energy applications. It also opens the possibility of scaling the approach to other platforms, such as microcontrollers or FPGA-based systems, paving the way toward next-generation MPPT controllers that combine computational intelligence with real-time adaptability.

### 5.2.3 Social contributions

The use of advanced prediction and optimization techniques in solar systems supports the promotion of sustainable energy sources and reduces dependency on fossil fuels. By improving solar energy utilization, the project contributes to the transition toward more sustainable environments, leading to direct societal benefits such as reduced pollutant emissions and improved access to clean energy in communities with limited conventional supply.

### 5.2.4 Economic contributions

Improving photovoltaic system efficiency through optimized models can reduce operating costs and increase the profitability of solar projects. In the long term, the implementation of this type of solution contributes to making solar energy more competitive compared to other energy sources, enabling broader accessibility and adoption by both companies and end users.

CHAPTER 6

# Ethical Considerations

## 6.1 Ethical considerations

It is mentioned that this research work does not involve the participation of human subjects, nor Experimental animals will be used and good laboratory practices will be followed, such as:

- To use machines and/or tools you must have appropriate clothing, that is, gown, closed shoes and safety glasses.
- All requests must be made in writing.
- The use of the facilities for personal work is prohibited.
- All signs must be respected as well as due diligence maintained at all times. respect, order and discipline within the laboratory.
- Smoking and eating drinks and/or food are strictly prohibited inside the laboratory.
- It is the user's obligation to always keep the work area clean and leave it in the same conditions upon completion of your work or internship.

## 6.2 Description of Resources Used in the Research

1. Equpment:
    - **Name**: Solar collector.
    - **Proper Use**:
        - Converts solar energy into electrical energy.
        - Safety measures: Install in a location with minimal shading and secure properly to withstand environmental conditions.
    - **Waste Management**:
        - Follow local regulations for recycling or disposing of photovoltaic panels at the end of their life cycle.

2. Renewables:
    - Flora
        - Not applicable.
    - Fauna
        - Not applicable.
3. Does your research require information obtained from human beings as a source of information? NO
    - **Description**:
        - The research does not require information obtained from human beings.
    - **Criteria for inclusion and exclusion implemented for the selection of participants**:
        - Not applicable.
    - **Attach informed consent document**:
        - Not applicable.
    - Does the research discriminate against the participation of individuals or include differential treatment among participants based on gender, race or ethnic group, age, religion, economic income, disadvantage or disability, illness, or any similar classification? NO
        - Not applicable.
    - Does the research include the participation of socially or physically vulnerable individuals (men and women under age, elderly, people with disabilities, etc.) or legally restricted or isolated groups, or could the inadequate use of information affect the integrity of individuals in any way? NO
        - Not applicable.

# Bibliography


[1] A. Vasuki, *Nature-Inspired Optimization Algorithms*. Chapman & Hall, 2020.

[2] D. A. Ruíz, "Integración de estructura iot a un concentrador de disco parabólico para mejorar la eficiencia de un horno solar," master's thesis, Universidad Autónoma de Querétaro, Facultad de Ingeniería, Querétaro, México, 2024. Maestría en Ciencias en Instrumentación y Control Automático.

[3] A. R. P. Venkata Mahesh, S. Meyyappan, "Maximum power point tracking with regression machine learning algorithms for solar pv systems," *International Journal of Renewable Energy Research-IJRER*, vol. 12, no. 3, p. 12, 2022.

[4] N. Kamarzaman and C. W. Tan, "A comprehensive review of maximum power point tracking algorithms for photovoltaic systems," *Renewable and Sustainable Energy Reviews*, vol. 37, p. 585–598, 09 2014.

[5] C. Mai, L. Zhang, X. Chao, *et al.*, "A novel mppt technology based on dung beetle optimization algorithm for pv systems under complex partial shade conditions," *Scientific Reports*, vol. 14, p. 6471, 2024.

[6] V. R. Kota and M. Bhukya, "A simple and efficient mppt scheme for pv module using 2-dimensional lookup table," 02 2016.

[7] K. K. Sreedhar R, "Machine learning based cascaded ann mppt controller for erratic pv shading circumstances," *International Journal of Power Electronics and Drive Systems*, p. 10, 2023.

[8] O. D.Gomez, F. Aznar, "Mppt algorithm based on multiple linear regression model for solar pv systems," *ECTI Transactions on Electrical Engieering Electronics and Communications*, p. 13, Jun 2022.

[9] I. E. Agency, "Net zero by 2050: A roadmap for the global energy sector," *IEA Publications*, 2023.

[10] J. Smith, "Effects of temperature on solar panel efficiency," *Journal of Renewable Energy Research*, vol. 12, pp. 345–356, 2018.

[11] L. Chen, "Improving mppt accuracy with neural networks," *International Journal of Solar Energy*, vol. 15, pp. 123–134, 2019.

[12] X. Liu, "Optimizing pv systems with advanced mppt algorithms," *Solar Energy Advances*, vol. 22, pp. 567–578, 2020.

[13] M. Garcia, "Cost benefits of optimized mppt in commercial pv installations," *Renewable Energy Economics*, vol. 10, pp. 89–100, 2021.

[14] Q. Zhang, "Enhancing pv system efficiency with machine learning-based mppt algorithms," *Energy Conversion and Management*, vol. 58, pp. 987–996, 2022.

[15] J. Konstantaras and M. Vrachopoulos, "Impact of non-uniform irradiance and temperature distribution on the performance of photovoltaic generators," *Energies*, vol. 16, no. 17, p. 6322, 2023.

[16] G. Farahani, *Effects of PV Module Shading on the Efficiency of the PV Array*, pp. 25–35. Singapore: Springer, 2020.

[17] A. Ziouh and A. Abbou, "Comparative study of fuzzy logic, ripple correlation control and pilot cell methods for maximum power point tracking," in *2016 International conference on electrical sciences and technologies in Maghreb (CISTEM)*, IEEE, 2016.

[18] J. Ahmed and Z. Salam, "A modified po maximum power point tracking method with reduced steady-state oscillation and improved tracking efficiency," *IEEE Transactions on Sustainable Energy*, vol. 7, no. 4, pp. 1506–1515, 2016.

[19] M. A. Husain *et al.*, "Comparative assessment of maximum power point tracking procedures for photovoltaic systems," *Green Energy and Environment*, vol. 2, no. 1, pp. 5–17, 2017.

[20] S. Bhattacharyya *et al.*, "Steady output and fast tracking mppt (soft mppt) for po and inc algorithms," *IEEE Transactions on Sustainable Energy*, vol. 12, no. 1, pp. 293–302, 2021.

[21] J. Gosumbonggot and G. Fujita, "Partial shading detection and global maximum power point tracking algorithm for photovoltaic with the variation of irradiation and temperature," *Energies*, vol. 12, p. 2154, 2019.

[22] U. D. of Energy, "Photovoltaic research and development," 2021.

[23] U. D. of Energy, "Solar research and development funding programs," 2022.

[24] P. Research and Development, "Department of energy," 2022.

[25] U. D. of Energy, "Photovoltaic research and development (pvrd)," 2023.

[26] Y. Wang, Z. Liu, and Y. Zhang, "Research on embedded system based on arm," in *Proceedings of the 2017 2nd International Conference on Education, Sports, Arts and Management Engineering*, pp. 258–262, Atlantis Press, 2017.

[27] S. Kumar *et al.*, "An embedded system—a review paper," *International Journal of Advanced Scientific Research & Engineering Trends (IJASRET)*, vol. 2, no. 3, 2015.

[28] M. Ahmadi, M. Ghazvini, M. Sadeghzadeh, and et al., "Review of solar collectors: Types, design, and performance," *International Journal of Ambient Energy*, vol. 41, no. 14, pp. 1645–1658, 2020.

[29] S. Topics, "Solar collector." https://www.sciencedirect.com/topics/agricultural-and-biological-sciences/solar-collector, 2023.

[30] A. C. Müller and S. Guido, *Introduction to Machine Learning with Python: A Guide for Data Scientists*. Sebastopol, CA: O'Reilly Media, 2016.

[31] G. Ciaburro, "Knowledge extraction from data using machine learning," *Data*, vol. 7, no. 3, pp. 110–123, 2022.

[32] O. Maimon, L. Rokach, and E. Shmueli, *Data Science and Knowledge Discovery Using Machine Learning Methods*, pp. 1–19. Springer, 2023.

[33] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[34] J. Friedman, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer, 2001.

[35] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer, 2nd ed., 2009.

[36] D. A. Freedman, *Statistical Models: Theory and Practice*. New York: Cambridge University Press, 2009.

[37] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*. Hoboken, NJ: John Wiley & Sons, 2012.

[38] J. Friedman, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer, 2001.

[39] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[40] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International Journal of Forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

[41] K. Miettinen, *Nonlinear Multiobjective Optimization*. Boston: Springer, 1999.

[42] K. Deb, *A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II*, vol. 6. 2002.

[43] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[44] A. Charnes and W. W. Cooper, "Goal programming and multiple objective optimization," *European Journal of Operational Research*, vol. 1, pp. 39–54, 1955.

[45] T. L. Saaty, *The Analytical Hierarchy Process*. New York: McGraw-Hill, 1990.

[46] C. L. Hwang and K. Yoon, *Multiple Attribute Decision Making: Methods and Applications*. Berlin: Springer, 1981.

[47] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*. New York: Springer, 2007.

[48] I. Fister Jr., X.-S. Yang, I. Fister, J. Brest, and D. Fister, "A brief review of nature-inspired algorithms for optimization," *Elektrotehniški Vestnik*, vol. 80, pp. 1–7, July 2013.

[49] X.-S. Yang, ed., *Nature Inspired Algorithms and Applied Optimization.* Studies in Computational Intelligence, Springer, 2018.

[50] H. Ahmed, A. Abid, and A. Obed, "Four bioinspired optimization techniques in PV MPPT under uniform and non-uniform shading," in *Proceedings of the 2023 International Conference on Power Electronics and Applications (ICPEA).*

[51] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[52] I. Sajid, A. Gautam, A. Sarwar, M. Tariq, H.-D. Liu, S. Ahmad, C.-H. Lin, and A. Sayed, "Optimizing photovoltaic power production in partial shading conditions using dandelion optimizer (DO)-based MPPT method," *Processes*, vol. 11, p. 2493, 2023.

[53] E. Ranganathan and R. Natarajan, "Spotted hyena optimization method for harvesting maximum PV power under uniform and partial-shade conditions," *Energies*, vol. 15, p. 2850, 2022.

[54] N. Deghfel, A. Badoud, F. Merahi, M. Bajaj, and I. Zaitsev, "A new intelligently optimized model reference adaptive controller using GA and WOA-based MPPT techniques for photovoltaic systems," *Sci. Rep.*, vol. 14, p. 6827, 2024.

[55] M. Jamaludin, M. Tajuddin, J. Ahmed, A. Azmi, S. Azmi, N. Ghazali, T. Babu, and H. Alhelou, "An effective salp swarm based MPPT for photovoltaic systems under dynamic and partial shading conditions," *IEEE Access*, vol. 9, pp. 34570–34589, 2021.

[56] Y. Zhang, Y.-J. Wang, H. Li, J.-B. Chang, and J.-Q. Yu, "A firefly algorithm and elite ant system-trained elman neural network for mppt algorithm of pv array," *Int. J. Photoenergy*, vol. 2022, p. 5700570, 2022.

[57] B. Jegajothi, S. Arumugam, N. Shukla, I. Kathir, P. Yamunaa, and M. Digra, "An efficient MPPT tracking in solar PV system with smart grid enhancement using CMCMAC protocol," *Comput. Syst. Sci. Eng.*, vol. 47, pp. 2417–2437, 2023.

[58] R. Sharmin, S. S. Chowdhury, F. Abedin, and K. M. Rahman, "Implementation of an mppt technique of a solar module with supervised machine learning," *Frontiers in Energy Research*, vol. 10, p. 932653, 2022.

[59] E. Elgeldawi, A. Sayed, A. Galal, and A. Zaki, "Hyperparameter tuning for machine learning algorithms used for arabic sentiment analysis," *Informatics*, vol. 8, no. 4, p. 79, 2021.

[60] S. Perez-Rodriguez, J. Alvarez-Alvarado, J.-A. Romero-Gonzalez, M. Aviles, A. Mendoza-Rojas, C. Fuentes-Silva, and J. Rodriguez-Resendiz, "Metaheuristic algorithms for solar radiation prediction: A systematic analysis," *IEEE Access*, vol. 12, pp. 100134–100151, 2024.

[61] J. M. Álvarez-Alvarado, J. G. Ríos-Moreno, E. J. Ventura-Ramos, G. Ronquillo-Lomeli, and M. Trejo-Perea, "An alternative methodology to evaluate sites using climatology criteria for hosting wind, solar, and hybrid plants," *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, vol. 43, no. 23, pp. 2974–2991, 2020.

CHAPTER 7

# Annexes

## .1 Python code for baseline ANN model

The following script shows the implementation of the baseline Artificial Neural Network (ANN) for photovoltaic power prediction. This model uses a fixed architecture with two hidden layers (64 and 32 neurons) and is trained with the Adam optimizer.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import matplotlib.pyplot as plt

# Load dataset
df = pd.read_excel('data_base_with_multiple_perturbations.xlsx')
df_cleaned = df.dropna()

# Split features and target
X = df_cleaned[['Temperature (deg. C)', 'Irradiance', 'Vmp', 'I_mp(A)']]
y = df_cleaned['Power']

# Normalize features
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
)

# Define ANN model
model = MLPRegressor(hidden_layer_sizes=(64, 32),
                     activation='relu', solver='adam',
                     max_iter=200, random_state=42)

# Train model
model.fit(X_train, y_train)

# Generate predictions
```

```
y_pred = model.predict(X_test)

# Evaluation metrics
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error (MSE): {mse}')
print(f'Mean Absolute Error (MAE): {mae}')
print(f'R^2 Score: {r2}')

# Plot actual vs predicted power
plt.figure(figsize=(12, 6))
plt.plot(y_test.values, label='Actual Power', color='blue')
plt.plot(y_pred, label='Predicted Power', color='red', linestyle='--')
plt.xlabel('Sample')
plt.ylabel('Power (W)')
plt.title('Actual vs Predicted Power (Baseline ANN)')
plt.legend()
plt.grid(True)
plt.savefig('predictions_vs_actual_nn.svg', format='svg')
plt.savefig('predictions_vs_actual_nn.eps', format='eps')
plt.show()
```

Listing 1: Python script implementing the baseline ANN model

## .2 Python code for ANN optimization with GWO

The following script shows the implementation of the Artificial Neural Network (ANN) optimized using the Grey Wolf Optimizer (GWO). The code includes preprocessing, model definition, optimization, evaluation metrics, and visualization of the results.

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import matplotlib.pyplot as plt
import time

# Load dataset
df = pd.read_excel('data_base_with_multiple_perturbations.xlsx')
df_cleaned = df.dropna()

# Split features and target
X = df_cleaned[['Temperature (deg. C)', 'Irradiance', 'Vmp', 'I_mp(A)']]
y = df_cleaned['Power']

# Normalize features
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
)

# ANN evaluation function
def evaluate_ann(params):
    hidden_layer_sizes = tuple(params.astype(int))
    model = MLPRegressor(hidden_layer_sizes=hidden_layer_sizes,
                         activation='relu', solver='adam',
                         max_iter=200, random_state=42)
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    return mse, mae, r2

# Grey Wolf Optimizer class
class GreyWolfOptimizer:
    def __init__(self, n, lb, ub, max_iter):
        self.n = n
        self.lb = lb
        self.ub = ub
        self.max_iter = max_iter
        self.positions = np.random.uniform(lb, ub, (n, len(lb)))
        self.alpha, self.beta, self.delta = np.zeros(len(lb)), np.zeros(len(lb)),
            np.zeros(len(lb))
        self.alpha_score = self.beta_score = self.delta_score = np.inf
        self.mse_history, self.mae_history, self.r2_history = [], [], []
        self.hidden_layer_history = []

    def optimize(self, fitness_function):
        start_time = time.time()
        for iteration in range(self.max_iter):
            for i in range(self.n):
                fitness = fitness_function(self.positions[i])
                if fitness < self.alpha_score:
                    self.delta_score, self.delta = self.beta_score, self.beta.copy
                        ()
                    self.beta_score, self.beta = self.alpha_score, self.alpha.copy
                        ()
                    self.alpha_score, self.alpha = fitness, self.positions[i].copy
                        ()
                elif fitness < self.beta_score:
                    self.delta_score, self.delta = self.beta_score, self.beta.copy
                        ()
                    self.beta_score, self.beta = fitness, self.positions[i].copy()
                elif fitness < self.delta_score:
                    self.delta_score, self.delta = fitness, self.positions[i].copy
                        ()

            a = 2 - iteration * (2 / self.max_iter)
            for i in range(self.n):
                for j in range(len(self.lb)):
                    r1, r2 = np.random.rand(), np.random.rand()
                    A1, C1 = 2 * a * r1 - a, 2 * r2
```

```
                    D_alpha = abs(C1 * self.alpha[j] - self.positions[i, j])
                    X1 = self.alpha[j] - A1 * D_alpha
                    r1, r2 = np.random.rand(), np.random.rand()
                    A2, C2 = 2 * a * r1 - a, 2 * r2
                    D_beta = abs(C2 * self.beta[j] - self.positions[i, j])
                    X2 = self.beta[j] - A2 * D_beta
                    r1, r2 = np.random.rand(), np.random.rand()
                    A3, C3 = 2 * a * r1 - a, 2 * r2
                    D_delta = abs(C3 * self.delta[j] - self.positions[i, j])
                    X3 = self.delta[j] - A3 * D_delta
                    self.positions[i, j] = (X1 + X2 + X3) / 3
                self.positions[i] = np.clip(self.positions[i], self.lb, self.ub)

            mse, mae, r2 = evaluate_ann(self.alpha)
            self.mse_history.append(mse)
            self.mae_history.append(mae)
            self.r2_history.append(r2)
            self.hidden_layer_history.append(self.alpha.copy())
            print(f"Iteration {iteration+1}/{self.max_iter}, Best MSE: {self.
                alpha_score:.4f}")

        print(f"Total Optimization Time: {time.time() - start_time:.2f} seconds")
        return self.alpha, self.alpha_score

# Define bounds and run optimizer
lb, ub = np.array([10, 10]), np.array([100, 100])
gwo = GreyWolfOptimizer(n=25, lb=lb, ub=ub, max_iter=50)
best_params, best_fitness = gwo.optimize(lambda x: evaluate_ann(x)[0])
print(f'Best parameters: {best_params}, Best MSE: {best_fitness}')
```

Listing 2: Python script implementing ANN optimization with Grey Wolf Optimizer (GWO)

## .3 Python code for ANN implementation in Keras

The following script shows the implementation of an Artificial Neural Network (ANN) using TensorFlow and Keras. The model uses two hidden layers (66 and 99 neurons), is trained with the Adam optimizer, and stored in HDF5 format for later deployment.

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import Adam

# Load dataset
df = pd.read_excel('data_base_with_multiple_perturbations.xlsx')
df_cleaned = df.dropna()

# Split features and target
X = df_cleaned[['Temperature (deg. C)', 'Irradiance', 'Vmp', 'I_mp(A)']]
y = df_cleaned['Power']

```

```
# Normalize features
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
)

# Define ANN model in Keras
keras_model = Sequential([
    Dense(66, input_dim=4, activation='relu'),  # Hidden layer 1
    Dense(99, activation='relu'),               # Hidden layer 2
    Dense(1)                                    # Output layer
])

# Compile model
keras_model.compile(optimizer=Adam(), loss='mse', metrics=['mae'])

# Train model
keras_model.fit(X_train, y_train,
                epochs=200, batch_size=32,
                validation_split=0.1, verbose=1)

# Save model in HDF5 format
keras_model.save('mlp_regressor_model_clean.h5', include_optimizer=False)

print("Model saved as 'mlp_regressor_model_clean.h5'")
```

Listing 3: Python script implementing ANN model in Keras/TensorFlow

## .4 Products Achieved

One of the most significant outcomes of this research was the publication of a peer-reviewed scientific article in the journal *Biomimetics*, entitled *"Performance Comparison of Bio-Inspired Algorithms for Optimizing an ANN-Based MPPT Forecast for PV Systems"*. This article, shown in Figure 1, presents a comparative analysis of different metaheuristic algorithms—GWO, PSO, CS, and SSA—for optimizing the hyperparameters of an artificial neural network used in maximum power point tracking (MPPT) prediction.

The publication validates the scientific contribution of this thesis to the academic community and situates the work within the broader context of bio-inspired optimization and renewable energy research. Moreover, it demonstrates that the methodological framework developed in this thesis has undergone rigorous peer review and has been recognized as relevant for advancing the state of the art in intelligent photovoltaic systems.

This product not only provides visibility to the research but also serves as a foundation for future studies, as it establishes a benchmark for comparing optimization strategies in MPPT forecasting. It highlights the superiority of GWO as an optimization algorithm and reinforces the decision to adopt it as the central method in this thesis for experimental implementation.

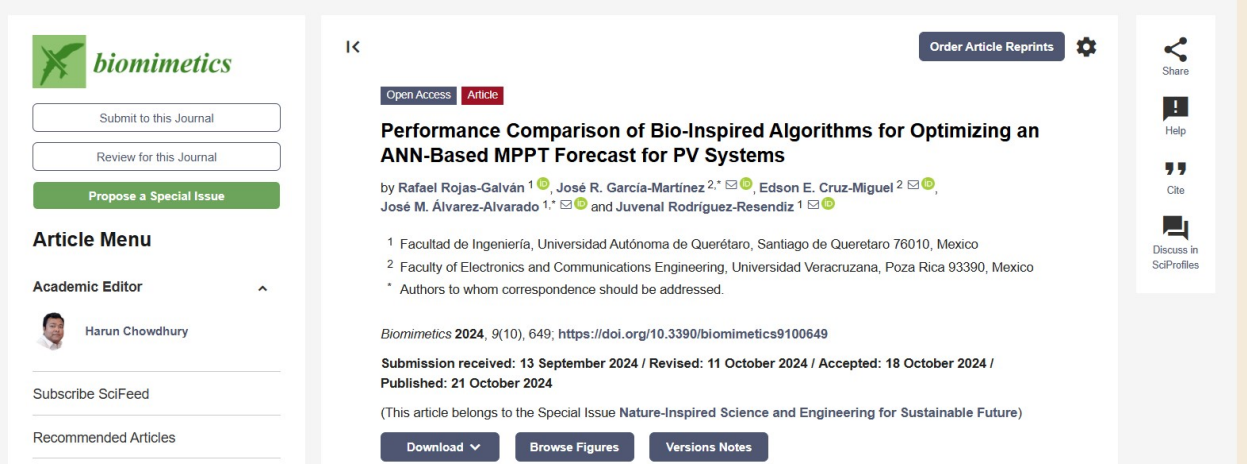



Figure 1: A scientific paper published in *Biomimetics* was derived from this research, presenting a comparative analysis of bio-inspired optimization techniques applied to ANN-based MPPT prediction.