Master's Thesis

Application of statistical techniques for the reduction of computation time in photovoltaic system simulations

Adrián Blanco Aguiar

Máster en Técnicas Estadísticas

Curso 2025-2026

Propuesta de Trabajo Fin de Máster

Título en galego: Aplicación de técnicas estatísticas para a redución do tempo de cómputo en simulacións de sistemas fotovoltaicos
Título en español: Aplicación de técnicas estadísticas para la reducción del tiempo de cómputo en simulaciones de sistemas fotovoltaicos
English title: Application of statistical techniques for the reduction of computation time in photovoltaic system simulations
Modalidad: Modalidad B
Autor/a: Adrián Blanco Aguiar, Universidade de Santiago de Compostela
Director/a: Brais González Rodríguez, Universidade de Vigo
Tutor/a: Miguel Sánchez de León Peque, ieco.io
Breve resumen del trabajo: En este trabajo se aplican diferentes técnicas estadísticas para reducir el tiempo necesario para realizar simulaciones de sistemas fotovoltaicos. Un ejemplo de estas técnicas es la utilización de un algoritmo de aprendizaje automático no supervisado (clustering) para la agrupación de los datos de entrada con el objetivo de reducir su dimensionalidad. El objetivo es reducir lo máximo posible el tiempo necesario para llevar a cabo las simulaciones, manteniendo controlado el error que se comete al realizar las simplificaciones.
Recomendaciones:
Otras observaciones:

Don Brais González Rodríguez, Profesor Distinguido de la Universidade de Vigo y don Miguel Sánchez de León Peque, fundador de ieco.io informan que el Trabajo Fin de Máster titulado

Application of statistical techniques for the reduction of computation time in photovoltaic system simulations

fue realizado bajo su dirección por don Adrián Blanco Aguiar para el Máster en Técnicas Estadísticas. Estimando que el trabajo está terminado, dan su conformidad para su presentación y defensa ante un tribunal. Además, Don Brais González Rodríguez y Don Adrián Blanco Aguiar

√ sí ☐ no

autorizan a la publicación de la memoria en el repositorio de acceso público asociado al Máster en Técnicas Estadísticas.

En Santiago de Compostela, a 05 de Enero de 2026.

El director:
Don Brais González Rodríguez

GONZALEZ RODRIGUEZ BRAIS - 44661382M
Digitally signed by GONZALEZ RODRIGUEZ BRAIS - 44661382M
Date: 2026.01.13 11:53:58 +01'00'

El tutor:
Don Miguel Sánchez de León Peque

05323619Q MIGUEL SANCHEZ (R: B02789733)
Digitally signed by 05323619Q MIGUEL SANCHEZ (R: B02789733)
Date: 2026-01-13T12:55:08.419681+01

El autor:
Don Adrián Blanco Aguiar

BLANCO AGUIAR ADRIAN - 49706851X
Firmado digitalmente por BLANCO AGUIAR ADRIAN - 49706851X
Fecha: 2026.01.13 12:01:03 +01'00'

Declaración responsable. Para dar cumplimiento a la Ley 3/2022, de 24 de febrero, de convivencia universitaria, referente al plagio en el Trabajo Fin de Máster (Artículo 11, Disposición 2978 del BOE núm. 48 de 2022), **el autor declara** que el Trabajo Fin de Máster presentado es un documento original en el que se han tenido en cuenta las siguientes consideraciones relativas al uso de material de apoyo desarrollado por otros/as autores/as:

- Todas las fuentes usadas para la elaboración de este trabajo han sido citadas convenientemente (libros, artículos, apuntes de profesorado, páginas web, programas,...)
- Cualquier contenido copiado o traducido textualmente se ha puesto entre comillas, citando su procedencia.
- Se ha hecho constar explícitamente cuando un capítulo, sección, demostración,... sea una adaptación casi literal de alguna fuente existente.

Y, acepta que, si se demostrara lo contrario, se le apliquen las medidas disciplinarias que correspondan.

Acknowledgments

I would like to express my deepest gratitude to my academic director, Brais González Rodríguez, for his invaluable guidance, expert mentorship, and continuous support throughout the development of this Master's thesis. His insights were fundamental to the completion of this work.

I extend my sincere thanks to Miguel Sánchez de León Peque and Clara Casas Castedo, founders of ieco.io, for their trust, for providing the necessary resources, and for the opportunity to conduct this research within a friendly and helpful environment.

Finally, I acknowledge the institutional and financial support that made this research possible:

This ieco.io research was supported by the project "Development of optimization algorithms and automation of the design of self-consumption photovoltaic installations, and analysis of the impact of local shadows on solar energy generation" of the company ieco.io, funded by the Xunta de Galicia through the Galician Innovation Agency (GAIN) under the program RECUPERACIÓN EXCELENCIA NEOTEC 2023-IN870A-006.

Contents

Abstract xi

Preface xiii

1 Introduction 1
- 1.1 Photovoltaic basics . . . 2
  - 1.1.1 Operating conditions data . . . 2
  - 1.1.2 Structure of photovoltaic systems . . . 4
  - 1.1.3 Current-voltage (I-V) curve and maximum power point (MPP) . . . 5
  - 1.1.4 Mismatch effect . . . 7
  - 1.1.5 Bypass diodes . . . 9
- 1.2 Implementation tools . . . 9
  - 1.2.1 Simulation scenarios . . . 10

2 Standard simulation methods 13
- 2.1 Simulation methods comparison . . . 13

3 Initial data aggregation methods 19
- 3.1 StraightForward Aggregation (SFA) . . . 21
- 3.2 Hierarchical Hourly Aggregation (HHA) . . . 23
- 3.3 Aggregation methods comparison . . . 26
- 3.4 SFA method for shading power losses . . . 28

4 Conclusions 31
- 4.1 Future work and limitations . . . 31

Bibliography 33

Abstract

English abstract

Accurate energy yield assessments (EYAs) for photovoltaic (PV) systems are critical for their financial viability, yet simulating the non-linear effects of partial shading over a full year remains computationally prohibitive. High-fidelity models, essential for capturing these mismatch losses, create a significant bottleneck in the rapid design and optimization of PV projects. This work introduces a novel data-centric framework to accelerate annual PV simulations by applying statistical aggregation, reducing the input dataset of operating conditions while preserving the high-resolution information necessary for accurate shading analysis. We propose and evaluate two methods based on k-means clustering, with results clearly demonstrating the superiority of one approach. A key advantage of this method is its tunability, enabling substantial reductions in simulation time while introducing only minimal, controllable error, thereby outperforming standard module-level simulations in both speed and accuracy. Importantly, when quantifying annual shading-induced power losses, the method drastically reduces computational overhead with negligible impact on accuracy, contrasting sharply with the significant underestimation of losses inherent to the module-level approach. This framework offers engineers a powerful and flexible tool for fast, reliable energy yield assessments without compromising simulation fidelity.

Resumen en español

Las evaluaciones precisas del rendimiento energético (EYA, por sus siglas en inglés) de los sistemas fotovoltaicos (FV) son fundamentales para su viabilidad financiera; sin embargo, simular los efectos no lineales del sombreado parcial durante un año completo sigue siendo computacionalmente prohibitivo. Los modelos de alta fidelidad, esenciales para capturar estas pérdidas por desajuste (mismatch), crean un cuello de botella significativo en el diseño y la optimización rápidos de los proyectos fotovoltaicos. Este trabajo presenta un novedoso marco de trabajo centrado en datos para acelerar las simulaciones anuales de sistemas FV mediante la aplicación de agregación estadística, reduciendo el conjunto de datos de entrada de las condiciones de operación y preservando al mismo tiempo la información de alta resolución necesaria para un análisis preciso del sombreado. Proponemos y evaluamos dos métodos basados en la agrupación por k-medias (k-means clustering), y los resultados demuestran claramente la superioridad de uno de los enfoques. Una ventaja clave de este método es su capacidad de ajuste, que permite reducciones sustanciales en el tiempo de simulación introduciendo a la vez un error mínimo y controlable, superando así a las simulaciones estándar a nivel de módulo tanto en velocidad como en precisión. Es importante destacar que, al cuantificar las pérdidas de potencia anuales inducidas por el sombreado, el método reduce drásticamente la carga computacional con un impacto insignificante en la precisión, lo que contrasta marcadamente con la significativa subestimación de las pérdidas inherente al enfoque a nivel de módulo. Este marco de trabajo ofrece a los ingenieros una herramienta potente y flexible para realizar evaluaciones del rendimiento energético rápidas y fiables sin comprometer la fidelidad de la simulación.

Preface

The company ieco.io focuses its activity in providing an online platform for photovoltaic (PV) system simulation. This is a crucial tool for engineers in order to design and dimension PV plants. The company has a range of simulation models, each one with different levels of detail. The more detailed the considered method, the more precise results are achieved. However, a high level of detail comes with an also high level of computational cost. At ieco.io, they focus their efforts in providing simulations that correctly estimate PV system performance when partial shadings are present. However, this can only be achieved with the highly detailed simulation methods, as the complex mismatching caused by those partial shadings are not captured with the less detailed methods. For this reason, the simulation method used for this purpose is very precise and detailed, but also very computationally demanding.

The simulation of a PV system performance is often done for a whole year. For this purpose, we need to feed meteorological data such as sun irradiance and position into the simulation. Even with hourly data (considered low resolution data in certain scenarios), we need to consider 8760 data points for a whole year (number of hours in a year), each one containing meteorological data at an specific date and hour stamp. For each one of these data points, we need to run a simulation, which turns into 8760 iterations of the simulation. This is computational prohibitive, as we are considering highly detailed simulation methods, that are very computationally demanding.

This issue motivates the study done in this work, in which we focus our efforts into reducing the number of data points that are fed into the simulation, while maintaining the information provided by the original data. This is done by reducing the 8760 hourly data points into a smaller set of representative points, containing a large amount of the information of the original data set. This is achieved by applying unsupervised machine learning algorithms for data segmentation into the data set. Different methods are studied, as well as its properties and limitations.

Chapter 1

Introduction

The widespread adoption of photovoltaic (PV) technology is crucial for the global transition to sustainable energy. Accurate energy yield assessments (EYAs) are fundamental to this adoption, underpinning both the financial viability and the design of PV projects (Milosavljevic et al. 2022). However, real-world conditions (particularly partial shading from obstructions such as buildings or clouds) can cause significant performance degradation due to non-linear electrical mismatch losses (Saeed et al. 2022). While high-fidelity simulations at the cell or submodule level can accurately capture these effects, they are computationally intensive, creating a substantial bottleneck for the annual performance analyses required for reliable EYAs. An annual simulation involves processing thousands of time steps, and the computational cost of detailed models renders tasks such as rapid design iteration and large-scale optimization impractical.

To address this challenge, ieco.io researchers have explored various acceleration strategies. One common approach is to reduce the input data by using clustering algorithms, such as k-means (Likas et al. 2003), to generate a set of "representative days" from a full year's weather data (Miraftabzadeh et al. 2023; Fahy et al. 2019). Other methods focus on simplifying the physical model itself, employing techniques like Model Order Reduction (MOR) (Gafurov et al. 2013) or replacing it entirely with machine learning (ML) surrogate models (Okif et al. 2025). While effective, existing "representative day" methods often aggregate 24-hour data profiles before clustering, a process that smooths out the instantaneous variations in sun position and irradiance. This loss of temporal resolution is critical, as the effects of partial shading are highly sensitive to the moment-to-moment geometry of the sun, array, and shading objects.

This work introduces a novel framework that accelerates simulations by applying statistical aggregation directly to the instantaneous operating conditions, thereby preserving the high-resolution data necessary for accurate shading analysis. We present and evaluate two distinct methods: StraightForward Aggregation (SFA), which applies k-means clustering directly to the multi-dimensional space of sun irradiance, elevation and azimuth; and Hierarchical Hourly Aggregation (HHA), which first segregates data by hour of the day before clustering. A key contribution of this work is the tunability of the SFA method, allowing users to explicitly define the desired data reduction percentage and predictably control the trade-off between simulation speed and accuracy. This study demonstrates that this data-centric approach provides a more robust and predictable performance improvement across various scenarios compared to simply using a less detailed physical model.

The contents of this work is an extension of the proceeding Blanco Aguiar et al. (2025), published at the EUPVSEC 2025 conference by ieco.io. However, as these conference proceedings must follow a specific format, the cited work is brief, in order match the extension requirements. We further develop the contents of it in here, exploring in more detail some aspects of the original work.

This document is organized to guide the reader from the fundamental physical principles to the implementation and evaluation of the proposed acceleration methods:

- **Chapter 1: Introduction.** This chapter establishes the context of the work. Following this

overview, we detail the fundamental physics of photovoltaics, the operating conditions (irradiance, temperature, sun position), and the hierarchical structure of PV systems (cells, submodules, modules, and arrays). We also describe the different tools we use to perform the simulations.

- **Chapter 2: Standard Simulation Methods.** Here, we describe the existing simulation framework used at ieco.io. We analyze the trade-offs between different levels of simulation granularity (cell-level, submodule-level, and module-level) and establish the submodule-level simulation as the optimal balance between accuracy and speed, which serves as the baseline for our experiments.
- **Chapter 3: Initial Data Aggregation Methods.** This chapter introduces the core contribution of the work. We detail the implementation of the k-means clustering algorithm and propose two distinct methodologies: StraightForward Aggregation (SFA), which clusters the entire dataset directly, and Hierarchical Hourly Aggregation (HHA), which applies clustering within hourly groups.
- **Chapter 4: Conclusions.** We summarize the findings, highlighting the superior performance of the SFA method, which defines a new Pareto frontier for the speed-accuracy trade-off. We also discuss the implications for estimating annual shading losses and suggest avenues for future research, as well as some limitations of this study.

The following sections of this introduction provide the necessary theoretical background regarding the physics of photovoltaic energy conversion and the system components required to understand the subsequent chapters. The conceptual PV basics are mainly based on the PV EDUCATION website (Honsberg and Bowden 2019).

1.1 Photovoltaic basics

The conversion of sunlight into electrical energy via the photovoltaic (PV) effect is a process based on the principles of semiconductor physics and optics. The power generated by a photovoltaic system is not an intrinsic property of the device but is instead a dynamic response to the conditions of its operating environment. A comprehensive understanding of a PV system's performance, therefore, begins with an analysis of the primary external variables that dictate its electrical output: the quantity of incident sunlight and the operating temperature of the PV device itself. These factors define the potential of the system for energy generation.

1.1.1 Operating conditions data

The electrical output of a photovoltaic device is linked to two primary environmental factors: solar irradiance, which serves as the energy input, and operating temperature, which modulates the efficiency of the energy conversion process. When shading objects are present, sun position gives us information about the shadows cast onto the system. Therefore, a PV system simulation begins with the input of these operating conditions data.

Solar Irradiance

The fundamental fuel for any photovoltaic system is solar irradiance, defined as the power density of sunlight incident on a surface. It is typically measured in Watts per square meter (W/m^2). The light-generated current within a solar cell is directly proportional to the intensity of this incident light. While the mean solar irradiance outside Earth's atmosphere is approximately $1366 W/m^2$, the value at the terrestrial surface is variable, attenuated by atmospheric conditions and by the geometric relationship between the sun and the PV surface. In addition to this, near objects can cast shadows onto the system, removing some of its incident sunlight. In order to simulate this shadings, it is important to

take into account the sun position at each time in order to determine the shadows present in the 3D scene.

Sun Position

Sun position is a critical input for analyzing shadows in a 3D scene. It provides the necessary information to determine the shadows cast onto a PV system by nearby objects, relative to the modules' positions. These shadows cause a drop in incident sunlight, thereby reducing the system's power generation. Sun position is defined as a two-dimensional variable, composed of sun elevation and sun azimuth, both measured as angles in degrees. We will not get into the details on how these angles are calculated, as they are given by meteorological data platforms like PVGIS (Huld et al. 2012). However, it is worth mention that they depend on the location of the PV system in the earth's surface.

On one hand, sun elevation is the angle that form the line to the sun with the tangent line with earth's surface. It ranges from 0^{o} at sunrise to 90^{o} on certain hours, days and locations. Elevations lower than 0^{o} are not relevant, as there is no sunlight in those scenarios. We show a representation of this angle in Figure 1.1.

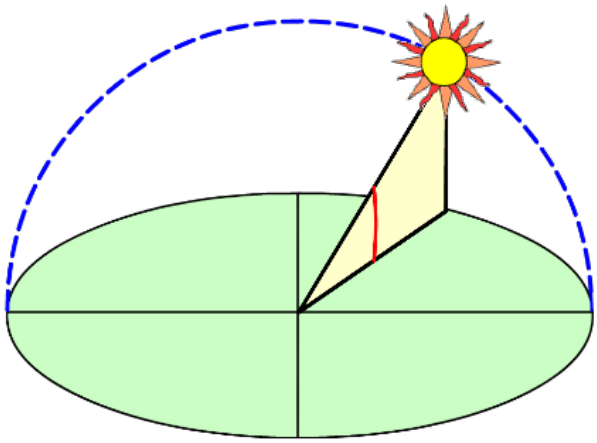

Figure 1.1: Sun elevation angle of the sun (in red) (Honsberg and Bowden 2019).

On the other hand, sun azimuth is the clockwise angle that form the line to the sun and the North compass direction line. It can range from 0^{o} to 360^{o} depending on the position of the system on the earth's surface. We show a representation of the angle in Figure 1.2.

Temperature

While irradiance provides the energy for conversion, the operating temperature of the solar cell is a critical, and predominantly negative, modulator of the conversion efficiency. As with all semiconductor devices, solar cells are highly sensitive to temperature. An increase in cell temperature directly impacts the material's electronic properties, producing a reduction in the power output. Even though temperature can be obtained from sun irradiance, it can be treated as an input variable.

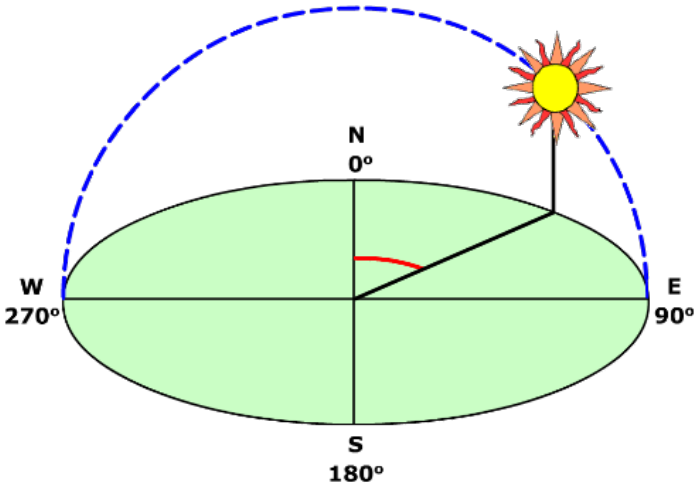

Figure 1.2: Azimuth angle of the sun (in red) (Honsberg and Bowden 2019).

1.1.2 Structure of photovoltaic systems

To generate useful levels of voltage and current for practical applications, individual solar cells must be interconnected into larger, more powerful structures. This assembly follows a well-defined hierarchy, scaling from a single cell to the expansive layout of a utility-scale power plant, conformed by several modules. Even though the structure of this union of solar cells can vary, we describe in this section the structure of the systems used for this work. However, many systems in the industry follow this structure or a similar one.

Cells

The solar cell is the fundamental building block of a PV system. It is an electronic device that directly converts the energy of photons in sunlight into electricity. Silicon cells are the most common ones in the current photovoltaic market, even though they are not the optimal in terms of sunlight energy absorption. However, the abundance of silicon and its wide spread in the semiconductor manufacturing industry make them the most used type of cells.

Submodules

A single crystalline silicon solar cell typically produces a low voltage. To achieve a voltage suitable for applications such as battery charging or grid connection, multiple cells are electrically connected in series. The first set of series-connected cells conform a half-submodule. Then, two half-submodules are connected in parallel, conforming a submodule. Each one of these submodules are protected by a bypass diode, that we will describe in following sections.

Module

Several submodules are connected in series in order to achieve a higher voltage. As already mentioned, each one of these submodules are protected by a bypass diode. Modules are the main way to scale a PV system, connecting several ones of them to achieve the desired power output. This connection of various modules conform an array.

Array

For applications requiring significant power output, multiple PV modules are interconnected to form a PV array. An array is the complete power-generating unit and can be configured to meet specific voltage and current requirements. A series connection of modules, referred to as a string, is used to increase the overall system voltage. To increase the total current output, multiple strings are then connected in parallel. This hierarchical series connection of components, from cell to submodule to module to string, makes the system adapt to any desired power output, making it scalable to more demanding use cases.

In our work, we will use different array configurations, conformed by JA Solar JAM72_S20_440 modules, with 144 cells, which features a twin half-cut cell architecture (Zhang et al. 2017). Their properties can be consulted in Shanghai JA Solar Technology Co. (2020). These cells are equally distributed in three submodules connected in series, each one protected by a bypass diode. Therefore, each submodule is composed of 48 cells. Each submodule, in turn, is composed of two parallel-connected half-submodules, each containing 24 cells in series. We show this PV system architecture in Figure 1.3.

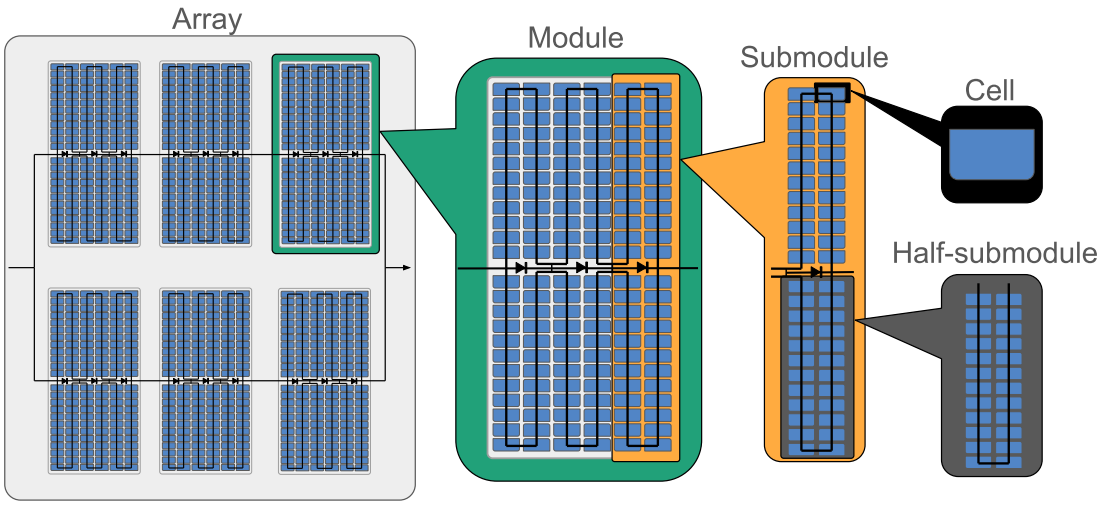

Figure 1.3: Layout of a PV system, from the cell (black) to submodule (orange) to module (green) to array (gray), from right to left. Half-submodule in bottom-right (dark gray).

1.1.3 Current-voltage (I-V) curve and maximum power point (MPP)

The complete electrical performance of a PV device under a given set of irradiance and temperature conditions is described by its current-voltage (I-V) characteristic. This curve serves as an essential diagnostic signature from which all key performance metrics are derived.

The I-V curve is a graphical representation of the relationship between the current (I) flowing through a solar cell and the voltage (V) across its terminals. In order to characterize it, the solar cell is modeled as an equivalent simple circuit. A common equivalent circuit model is the single diode model (Gray 2011), which is motivated on physical principles and characterized by the circuit in Figure 1.4.

This equivalent circuit model is mathematically described as the superposition of the exponential I-V curve of the semiconductor diode in the dark with the light-generated current, I_L (A), taking into

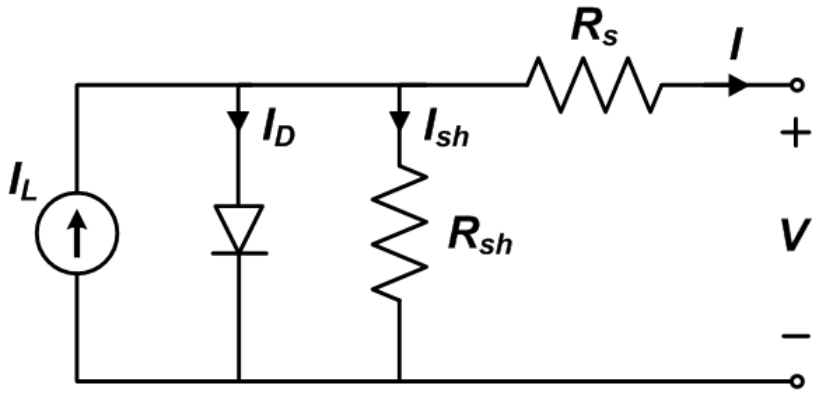

Figure 1.4: Single diode equivalent circuit, considering physical losses (Sandia National Laboratories 2025).

account some physical losses, derived from the series resistance, R_s (Ω), and the shunt resistance, R_{sh} (Ω). This superposition effectively shifts the I-V curve down into the fourth quadrant of the graph, the region where net electrical power can be extracted from the device. However, the curve is flipped into the first quadrant by convention. This results in the following I-V curve equation in the first quadrant:

$$I = I_L - I_0 \left[\exp\left(\frac{qV + qIR_s}{nkT}\right) - 1\right] - \frac{V + IR_s}{R_{sh}}, \tag{1.1}$$

where I_0 is the diode leakage current density in the absence of light (A), T is the absolute temperature (K), n is the diode ideality factor (unitless) and q and k are the absolute value of electron charge ($1.602 \times 10^{-19} C$) and the Boltzmann's constant ($1.381 \times 10^{-23} J/K$), respectively.

In Figure 1.4, the components of the Equation 1.1 are represented as $I_D := I_0 \left[\exp\left(\frac{qV+qIR_s}{nkT}\right) - 1\right]$ and $I_{sh} := \frac{V+IR_s}{R_{sh}}$, so that the equation can be written as $I = I_L - I_D - I_{sh}$, where the current of the cell is represented as the light-generated current, I_L, considering the voltage-dependent current losses, I_D, and the shunt resistance-dependent current losses, I_{sh}.

In an ideal cell, i.e., if $R_s = 0$ and $R_{sh} = \infty$, Equation 1.1 can be simplified as

$$I = I_L - I_0 \left[\exp\left(\frac{qV}{nkT}\right) - 1\right], \tag{1.2}$$

that can make the intuition for the following explanations easier to visualize, as it simplifies into an explicit equation, that makes it possible to express the voltage in terms of the current as

$$V = \frac{nkT}{q} \ln\left(\frac{I_l - I}{I_0}\right). \tag{1.3}$$

From Equation 1.3 we can extract that, when $I > I_L$ ($I - I_L < 0$), the logarithm becomes undefined. In reality, when this happens, the cell goes into reverse bias (negative voltage), forcing the cell to dissipate power. This causes a quick increase in the cells' temperature, provoking the so called "hot spots". If the cell is close to an ideal cell (high R_{sh}), it will be almost instantly destroyed in this situation, also affecting the nearby cells due to heating. A solution to this problem is described in Section 1.1.5.

Another property that we can extract from the equation is that the curve is bounded by two critical operating points:

- **Short-Circuit Current (I_{SC}):** The maximum current that can be drawn from the cell. This occurs at zero voltage. The value of I_{SC} is nearly identical to the light-generated current I_L and is directly proportional to the incident solar irradiance.
- **Open-Circuit Voltage (V_{OC}):** The maximum voltage that appears across the cell. This occurs at zero current. The value of V_{OC} has a logarithmic dependence on irradiance and is strongly and inversely dependent on temperature.

The PV system will aim to work under the conditions that generate the highest power output, extracted from the I-V curve as the maximum $P = I \times V$ value. This maximum value is know as the Maximum Power Point (MPP) of the system. Therefore, the objective of the simulation will be to extract this MPP for any given system I-V curve, extracted from the operating conditions of the system. We show in Figure 1.5 an example of the I-V curve of a cell, remarking some of its relevant points.

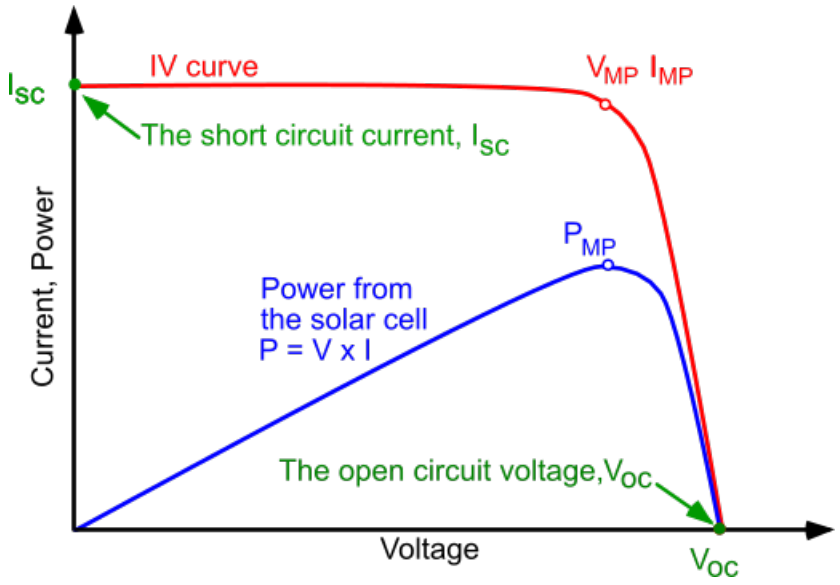

Figure 1.5: Current voltage (I-V) curve of a solar cell and some of its relevant points, such as I_{SC}, V_{OC} and, most importantly, MPP, denoted as P_{MP} (Honsberg and Bowden 2019).

The I-V curve shown corresponds to an ideal uniform illuminated system scenario. In reality, when partial shadings are present, bypass diodes activate as described in Section 1.1.5. In those cases, fluctuations in the I-V curve appear, making it harder to find the MPP, as multiple local maximums are present in the power curve. However, the consideration of these fluctuations is necessary, as neglecting them in the simulation would provoke errors on the results according to the reality.

1.1.4 Mismatch effect

In an ideal scenario, our whole module could be completely lit by direct sunlight. In that case, we just need to compute the I-V curve of the entire module as a whole and extract the MPP from it. However, in a more realistic scenario, the individual cells from the module could not behave all the same. This can be caused by diverse factors, being partial shading the most important for this work. This effect must be taken into account, as the power output of the entire PV system can be determined by a single solar cell with low output. This can lead to dissipate the power generated by the entire module in a single solar cell, causing it to overheat and provoke a hot spot, provoking irreversible damage to the module. We show a solution for hot spots in Section 1.1.5.

Even if the hot spot problem is solved, we need to take into account the mismatch effect into the simulation, as it plays an important role even in scenarios where hot spots do not occur. For this, we need to check each cell individually to know how is it behaving. After that, we can join the cells considering the disposition of them in the module. This is important, as mismatch does not behave the same when occurring on cells connected in series as in cells connected in parallel.

When cells are combined in series, the total voltage is the sum of the individual cell voltages. However, current must be the same, so it is equal to the lowest current between both cells. This provoke a drop in current if one cell connected in series has a I-V curve with low current. This is shown in Figure 1.6.

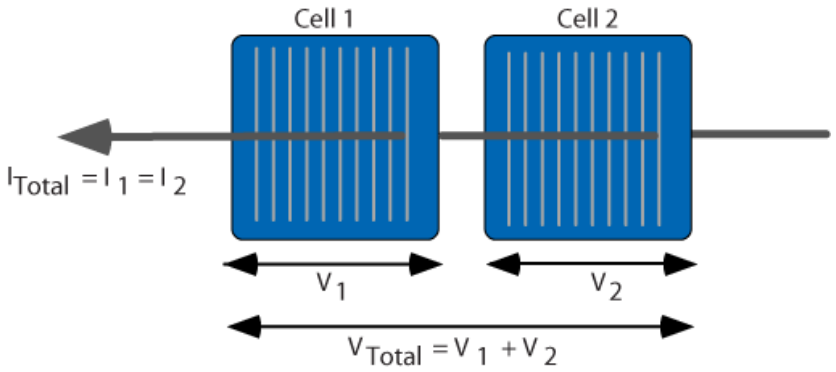

Figure 1.6: Connection between two cells in series. Voltages are added, but current must be the same, so it is equal to the lowest current (Honsberg and Bowden 2019).

Once we get the I-V curve from the cells connected in series, we need to connect them in parallel. In that case, we get the opposite effect. The total intensity is the sum of the cells intensities, while voltage must be the same and forced to be the lowest voltage between both cells. This makes voltage to be lower than expected in mismatch scenarios. This is shown in Figure 1.7.

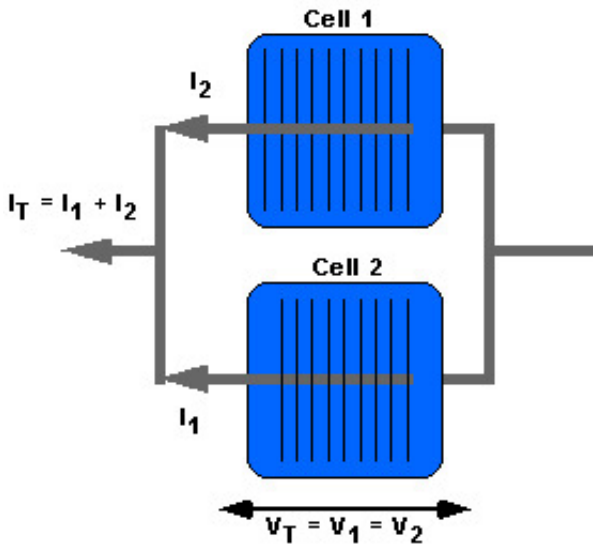

Figure 1.7: Connection between two cells in parallel. Intensities are added, but voltage must be the same, so it is equal to the lowest voltage (Honsberg and Bowden 2019).

Even though the explanation is made for a two-cell connection, the same applies to a larger number

of cells connection. For example, in our module architecture case, the procedure is the following:

1. Connect two cells in series and get the I-V curve of the connection.
2. Treat that two-cell connection as one and connect it with the next cell in series.
3. Repeat the procedure for all the series connected cells for each half-submodule.
4. Connect each pair of half-submodule in parallel, that will conform the submodules I-V curves.
5. Finally, connect the three submodules in series as before (if more modules are present in the array, connect each module as desired depending on the array architecture).

Mismatch effect forces us to compute the I-V curve for each cell of the module and combine them as corresponds. This can lead to high computational cost in large systems simulations, as we need to compute a lot of curves and combinations of them. In addition to that, mismatch can cause the final power-voltage curve of the system to present several local maximums, making it harder to determine the MPP. Lastly, we have to solve the problem of hot spots, that can cause irreversible damage to the system. We show a solution for this in the next section.

1.1.5 Bypass diodes

Given the vulnerabilities introduced by series connections and harsh environmental conditions, incorporating protective components into the PV module is essential for ensuring long-term reliability and safety. A bypass diode is a protective semiconductor device connected in parallel across, most commonly, a submodule. Its function is to mitigate the destructive effects of current mismatch caused by partial shading or other non-uniformities.

Under normal, uniform illumination, all cells in a module are generating power. In this state, the bypass diodes remain inactive. However, if a cell or submodule becomes shaded, its ability to generate current plummets. The other fully illuminated cells in the same string will attempt to force their higher current through the shaded section. This causes the shaded cells to dissipate significant power and leading to a rapid temperature rise known as "hot-spot" heating, which can cause irreversible damage to the module materials.

The bypass diode prevents this catastrophic failure. When a certain threshold of cell shading occurs in a submodule, the diode activates and provides a low-resistance alternative path for the current. This allows the current from the healthy submodules to "bypass" the underperforming one, limiting the reverse voltage across it and preventing thermal runaway. The bypass diode transforms the inherent vulnerability of a series-connected system into a manageable and non-destructive event, thereby ensuring the system's long-term operational viability. This behavior is represented in a graphical way in Figure 1.8.

1.2 Implementation tools

In order to simulate the performance of the PV system, ieco.io developed a codebase to perform the necessary simulations to estimate this power output. The codebase is written in Python (Python Software Foundation 2024) and based on the pvlib open source library (Anderson et al. 2023). This grants us the necessary functions and classes to perform the simulation of a PV system performance.

The codebase comprises several Python modules, totaling approximately 3,300 lines of source code. The structural organization is detailed below:

- *simulation.py*: Encapsulates the *Simulation* class, which contains the core logic for executing various simulation methods (1,463 lines).
- *utils.py*: Houses auxiliary functions for intermediate calculations, such as aggregating I-V curves in series or parallel and deriving single-cell I-V characteristics (566 lines).

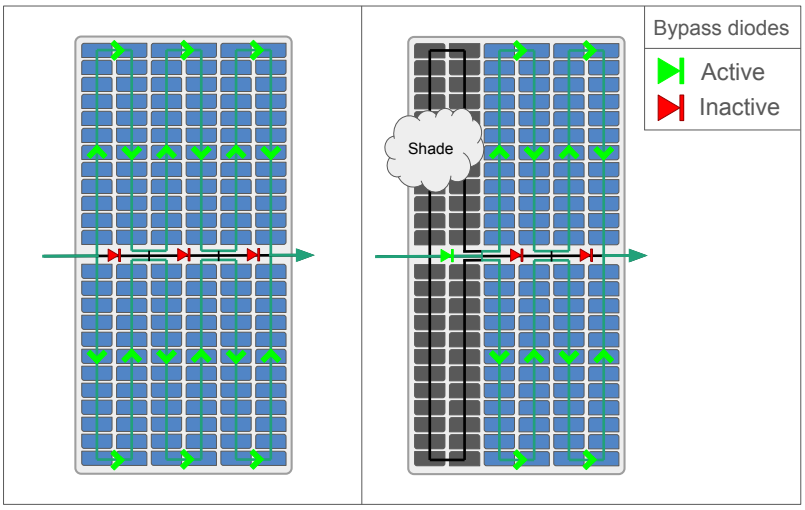

Figure 1.8: Representation of bypass diodes behavior under a partial shading scenario.

- *cell.py*: Defines the *Cell* class to establish the physical and electrical properties of individual solar cells (98 lines).
- *module.py*: Outlines the *Module* class, setting the parameters for the specific PV module under study (523 lines).
- *array.py*: Specifies the *Array* class, which defines the configuration of the photovoltaic array (41 lines).
- *shading.py*: Provides the algorithms necessary for shading calculations, including the determination of shading factors at both the system and individual cell levels (622 lines).

Complementary Jupyter Notebooks (Granger and Pérez, 2021) were utilized to generate the figures presented in this work. The long-term objective in ieco.io is the integration of this codebase into its commercial platform to enhance simulation accuracy under partial shading conditions. Furthermore, plans are in place to release the software as an open-source tool, thereby facilitating reproducibility and community access.

Although the primary research objective was the development of aggregation methods, significant effort was dedicated to computational optimization. This ensures reduced execution time without compromising simulation fidelity.

1.2.1 Simulation scenarios

The performance of the methods presented in this work is tested in a purely empirical way. For this, we consider various PV system scenarios, shown in Table 1.1. The scenarios considered range in size from small systems, corresponding to typical residential installations, to larger commercial-scale systems. Common shading objects are also considered to measure the accuracy of the methods when it comes to partial shading effects. We show a more visual representation of the simulation scenarios in Figure 1.9.

We run the simulations on each one of these scenarios with the different simulation methods presented in this work and compare the results to get an empirical measurement of the characteristics of these methods. However, the performance results must be seen as purely empirical metrics that may not cover all the real life scenarios. To achieve a better representation of the performance of the methods, many more scenarios should be considered, preferable based on existing real life PV

Scenario ID	Tilt (°)	Azimuth (°)	Obstruction(s)	Parameters (dist/height [m], x-range)
1 Module (small residential)				
1M_T20_A0_1IW	20	0	1 infinite wall	1.2 / 1.35
1M_T0_A50_1IW	0	50	1 infinite wall	1.2 / 1.35
1M_T30_A-10_1C	30	-10	1 chimney	2 / 2, $x \in [3, 4]$
1M_T30_A-10_2C	30	-10	2 chimneys	#1: 1.5 / 2, $x \in [-5, -4]$; #2: 2 / 2, $x \in [3, 4]$
8 Modules (4 series, 2 parallel, medium residential)				
8M_T20_A0_1IW	0	0	1 infinite wall	1.2 / 1.35
8M_T20_A50_1IW	0	50	1 infinite wall	4 / 2
8M_T20_A0_2W	0	0	2 walls	#1: 1.2 / 1.35, $x \in [-7, -2]$; #2: 1.2 / 1.35, $x \in [2, 7]$
8M_T30_A-10_1C	30	-10	1 chimney	2 / 2, $x \in [3, 4]$
8M_T30_A-10_2C	30	-10	2 chimneys	#1: 1.5 / 2, $x \in [-5, -4]$; #2: 2 / 2, $x \in [3, 4]$
16 Modules (4 series, 4 parallel, large residential)				
16M_T30_A-10_1C	30	-10	1 chimney	2 / 2, $x \in [3, 4]$
16M_T30_A-10_2C	30	-10	2 chimneys	#1: 1.5 / 2, $x \in [-5, -4]$; #2: 2 / 2, $x \in [3, 4]$
64 Modules (8 series, 8 parallel, commercial)				
64M_T0_A0_1B	0	0	1 box	1 / 2, $x \in [3, 6]$
64M_T0_A0_1B1C	0	0	Box and chimney	Box: 1 / 2, $x \in [3, 6]$; Chimney: 1 / 2, $x \in [-4, -3]$

Table 1.1: Configuration of simulation scenarios organized by PV system size.

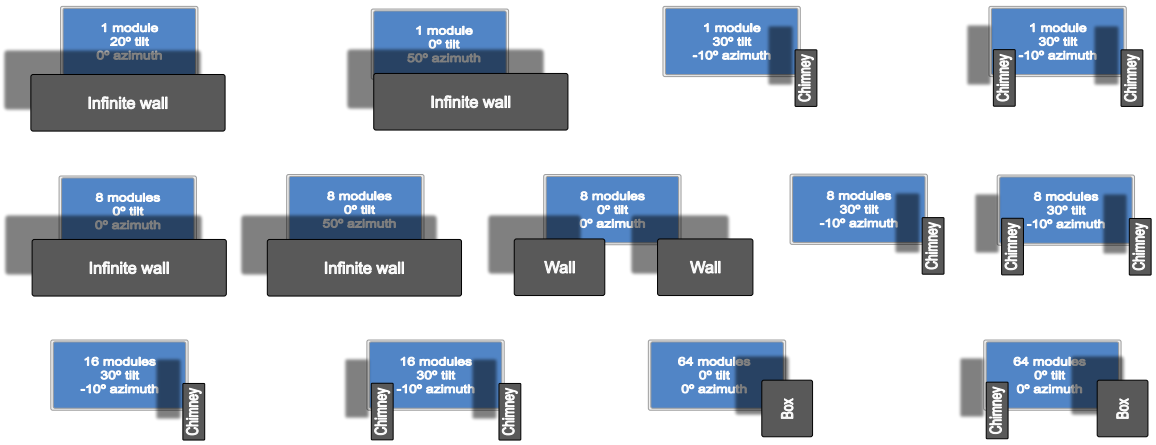

Figure 1.9: Visual representation of the simulation scenarios considered to measure the accuracy of the different methods presented in this work. Simulation scenarios range in size and contain different shading objects.

systems. However, the more scenarios we consider, the more computational resources we need. For the purpose of this work, we find these scenarios to cover a sufficient range of PV systems variations while maintaining a reasonable amount of computational resources needed.

Chapter 2

Standard simulation methods

As already mentioned, ieco.io developed a custom Python simulation framework built upon the pvlib library. The simulation first computes the system's current-voltage (I-V) curve, from which the power output is determined by identifying the maximum power point (MPP), as described in Section 1.1.3. To model the I-V curve of a cell, module or array, we use the single-diode model, described in that same section.

Our simulation framework supports multiple methods for computing PV system power output. Since this work focuses on partial shading, a cell-level simulation is the most accurate approach, as it explicitly models the electrical mismatch between individual cells, modeling the bypass diodes as described in Section 1.1.5. However, this high detailed simulation method is computationally intensive for large systems. It requires calculating the I-V curve for each cell and then aggregating these curves while accounting for mismatch due to shading and the effects of bypass diodes. The high computational cost motivates the adoption of faster, albeit less precise, simulation methods.

One such alternative is a submodule-level simulation. This approach applies the single-diode model to each submodule as a whole, calculating an aggregate I-V curve that incorporates the effects of shading across that submodule. These submodule curves are then combined, taking bypass diodes into account. This method provides a good approximation of the power output while requiring significantly fewer computational resources. For those reasons, we use the submodule-level simulation as a benchmark method to evaluate the performance of the novel approaches introduced in this paper.

Finally, we consider a module-level simulation method that disregards the mismatch caused by partial shading. In the following section, the performance of all three approaches (cell-, submodule-, and module-level) is evaluated under these partial shading scenarios and compared with the simplest approach that ignores shading.

2.1 Simulation methods comparison

The precision of the simulation methods described above depends on their level of granularity. Less detailed methods tend to overestimate the system's power output, as they do not account for the electrical mismatch losses caused by partial shading as described in Section 1.1.4.

To illustrate this behavior, we present the simulated power output from the different methods for a specific scenario: a single-module system shaded by two chimneys (Scenario 1M_T30_A-10_2C in Table 1.1) on November 3rd. We show these results in Figure 2.1, which plots the MPP in Watts as a function of the hour of the day (from 6:00 to 19:00). We represent each simulation method by a different color: the ideal, unshaded simulation (red), the module-level simulation (orange), the submodule-level simulation (blue), and the cell-level simulation (green).

As established above, less detailed simulation methods overestimate the system's power output because they neglect the mismatch losses caused by partial shading. This overestimating behavior

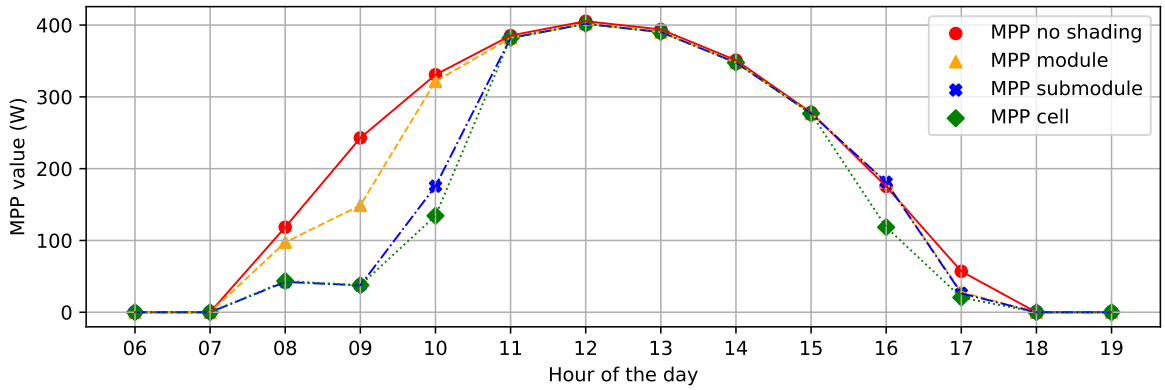

Figure 2.1: MPP output on November 3rd from 6 to 19 hours for different simulation methods in a scenario with a single module shaded by two chimneys.

does not occur as an isolated case for November 3rd as shown on Figure 2.1. The less detailed method consistently overestimates the MPP output for all the scenarios and date stamp.

In order to compare the different simulation methods in this work, we will use relative metrics, as we are considering simulation scenarios with distinct scales, so we will need some sort of scaling in the comparison metrics. With this in mind, we will present the results of a simulation method relative to a reference method. Those metrics quantify the discrepancies of both methods in terms of MPP output and computational time. This grants us the results needed to compare the different methods.

To quantify the accuracy of the different methods presented in this work, we compute the error of a simulation method relative to a reference one for a given scenario as

$$RelativeError = \frac{\sum_{t=1}^{T} |MPP^{(t)}_{new\_method} - MPP^{(t)}_{reference\_method}|}{\sum_{t=1}^{T} MPP^{(t)}_{reference\_method}}, \tag{2.1}$$

where $t = 1, ..., T$ correspond to the different time steps considered in the simulation (if we consider hourly input data and we want to perform the simulation of MPP values for the whole year, we have $T = 8760$, that corresponds to the hours present in a year) and $MPP^{(t)}_{new\_method}, MPP^{(t)}_{reference\_method}$ the MPP output of the method to compare and the one of the reference method, respectively, at the time step t.

To compare the long-term performance estimated by a simulation method relative to a reference one for a given scenario, we will also use the metric

$$\begin{aligned} RelativeDifference &= \frac{\sum_{t=1}^{T} (MPP^{(t)}_{new\_method} - MPP^{(t)}_{reference\_method})}{\sum_{t=1}^{T} MPP^{(t)}_{reference\_method}} \\ &= \frac{\sum_{t=1}^{T} MPP^{(t)}_{new\_method}}{\sum_{t=1}^{T} MPP^{(t)}_{reference\_method}} - 1, \end{aligned} \tag{2.2}$$

which calculates the relative difference in annual power output between any two methods being compared.

The utilization of the metrics (2.1) and (2.2) will depend on the use case and the characteristics of the results that we want to compare. For instance, if we are interested in punctual precision, we will make use of the relative error (2.1), as it will compute the summation of all the absolute punctual discrepancies of the two methods to compare, all relative to the reference one. However, relative difference (2.2) will be useful if we want to compare the estimations of total power output of the

system throughout the entire simulation period, as it is computed as the ratio between the summation of all MPP outputs on the simulation period for both methods, minus one. For example, if we get a ratio greater than one, we will have a positive relative difference, indicating that the new method estimates a greater power output of the system in the simulation period relative to the reference one. However, when it comes to punctual precision measurement, this last metric is not useful, as overestimation of MPP in some time stamps can be compensated by underestimation in other time stamps. This can be easily extracted from the first expression in equation (2.2), where negative terms of the summation can be canceled by other positive ones.

Even though this two metrics are different in general, there are some cases where they are equivalent or opposite. From equation (2.1) and the first expression of equation (2.2), we can easily extract that

$$\begin{aligned} &\text{if } (MPP^{(t)}_{new\_method} - MPP^{(t)}_{reference\_method}) > 0, \forall t = 1, ..., T, \\ &\text{then } RelativeError = RelativeDifference, \end{aligned} \tag{2.3}$$

and

$$\begin{aligned} &\text{if } (MPP^{(t)}_{new\_method} - MPP^{(t)}_{reference\_method}) < 0, \forall t = 1, ..., T, \\ &\text{then } RelativeError = -RelativeDifference. \end{aligned} \tag{2.4}$$

In other words, (2.3) stands that if we consistently overestimate the MPP with a simulation method in comparison with a reference one for the entire simulation period, then both metrics are equivalent and (2.4) stands that if we consistently underestimate it, then they are equal in magnitude but opposite in sign.

This is the case in our simulation methods, as less detailed methods consistently overestimate the power output of the system. With this in mind, if we take the less detailed simulation method (unshaded simulation) as the reference one, and compare the other ones relative to it, we can make use of (2.4) to conclude that it is equivalent to consider relative errors or differences in our case (taking into account the corresponding sign). We will make use of the relative differences to compare the methods, as it grants us more information in this case due to the sign, that emphasizes the underestimation of the more detailed method relative to the unshaded simulation.

With this motivation, Figure 2.2 displays the distribution of the relative differences in power output for each simulation method, calculated with (2.2) using the ideal (unshaded) simulation method as reference. Each boxplot (McGill et al. 1978) in the figure summarizes the results across all the scenarios described previously.

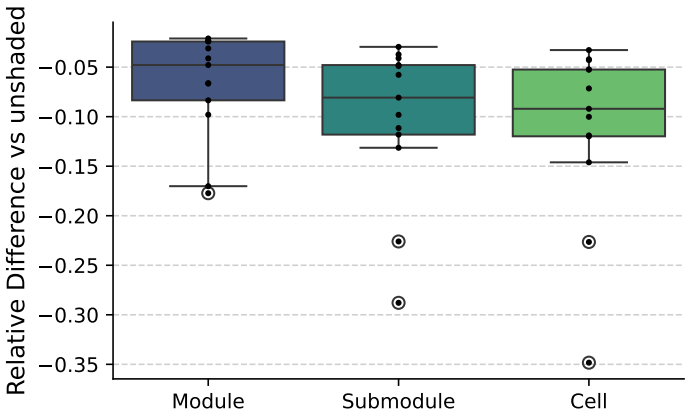

Figure 2.2: Distribution of the relative differences of the power output throughout a year for every simulation scenario for each simulation method with the unshaded simulation as the reference.

The results show a clear trend: as the model detail of the simulation method increases, the estimated annual power output decreases. This reduction, which quantifies the estimated power loss from partial shading, reaches up to over 30% for the cell-level simulation in scenarios with significant mismatch, making this most detailed cell-level simulation method crucial in these scenarios with high partial shading presence if we want precise results.

However, this precision comes at a high computational cost. To quantify this trade-off, we compare the execution time of the methods by calculating a relative time metric, analogous to (2.2), as

$$RelativeTime = \frac{time_{new\_method} - time_{reference\_method}}{time_{reference\_method}} = \frac{time_{new\_method}}{time_{reference\_method}} - 1. \tag{2.5}$$

We represent in Figure 2.3 the relative execution times for each simulation method, calculated with (2.5) using the unshaded simulation method as the reference. Each boxplot show the distribution of the relative times between all simulation scenarios. The findings highlight the substantial computational overhead of the cell-level method, which is on average 2000% slower (21 times the reference time) and, in the worst case, up to 8000% (81 times the reference time), in comparison with the unshaded simulation method. In contrast, the submodule-level simulation is far more efficient, showing a mean slowdown of 200% (3 times the reference time) and a maximum of 600% (7 times the reference time).

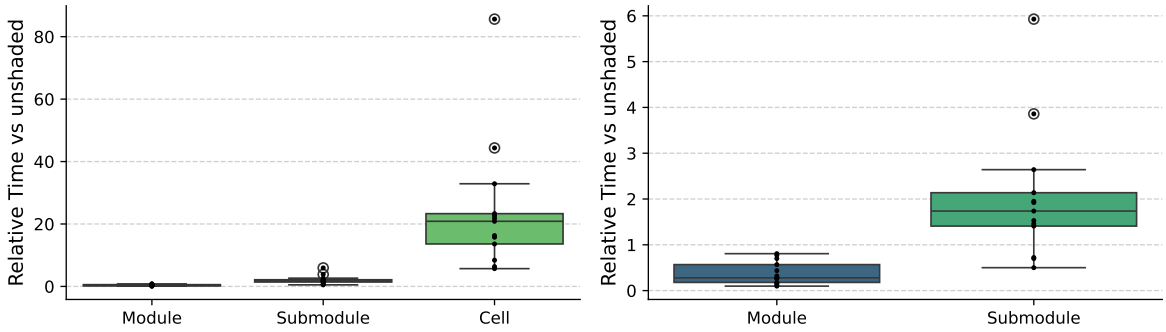

Figure 2.3: Relative times of compute needed for every simulation scenario for each simulation method with the unshaded simulation as the reference. The right graph is the same as the left one without the simulation at the cell level results for cleaner visualization.

Taking this speed/accuracy trade-off into account, if we recall Figure 2.2 results, we can see that the submodule-level simulation seems similar to the cell-level simulation in terms of discrepancies with the unshaded method. This similarity, added to the enormous advantage in speed shown in Figure 2.3, makes this simulation method the baseline method used in the industry as a good approximation to reality while not being computationally prohibitive. This is shown in Figure 2.4, where cell-level simulation method results in terms of relative differences and times relative to the submodule-level simulation are represented. From it, we can extract that submodule-level simulation makes for a good approximation of cell-level simulation for most cases, not surpassing the 2% relative different mark for all simulation scenarios except for one with high partial shading presence, that present a 8% of relative difference, that is acceptable in most cases. However, the improvement in computational time makes this submodule-level simulation method more suitable in the practice, as it is 600% (7 times) faster in average than the cell-level simulation.

We also show the comparison of the results of the other mentioned methods (unshaded and module-level) with the submodule-level simulation as the reference in Figure 2.5. This shows us that, even though this low-detail methods are faster than the submodule-level simulation, they tend to generate high errors in power output estimation, specially in scenarios with a lot of mismatch effect. This makes these methods unsuitable for our use case, as we need to take into account as much as possible the losses caused by partial shadings. However, they may be useful for other use cases, where precision is

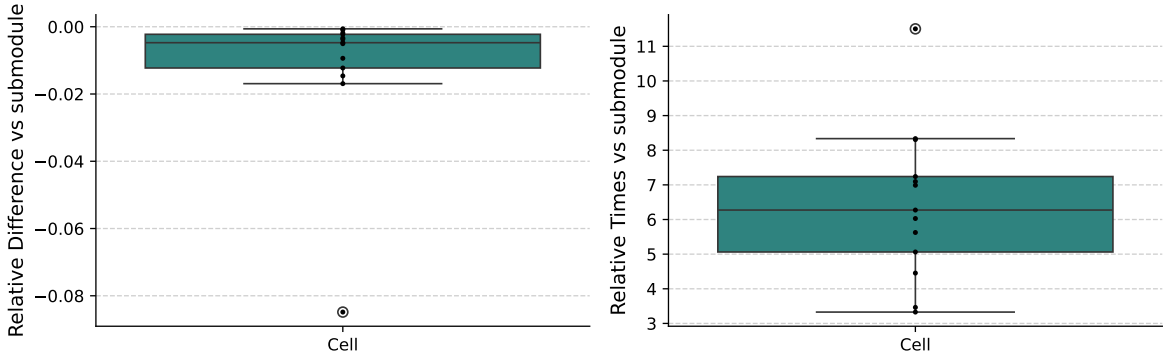

Figure 2.4: Relative differences and times for every simulation scenario for the cell-level simulation method with the submodule-level simulation as the reference.

not as necessary as in our case and fast results are needed. Moreover, in the following chapter we use the module-level simulation method as a baseline to compare the errors and computational time saved by applying the statistical methods over the submodule-level simulation in order to approximate the results.

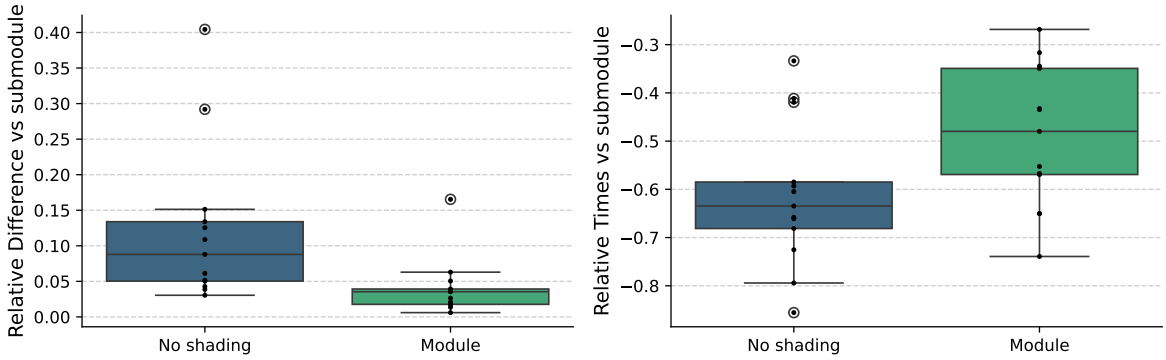

Figure 2.5: Relative differences and times for every simulation scenario for the unshaded and module-level simulation method with the submodule-level simulation as the reference.

The results presented in this section demonstrate that the submodule-level simulation provides an effective trade-off between accuracy and computational cost. It delivers a reliable power output estimation, while remaining computationally efficient. Therefore, we selected this simulation method as the basis for applying and compare the aggregation methods described in the following chapter.

Chapter 3

Initial data aggregation methods

Simulating the annual power generation of a PV system requires full year of operating condition data. In this work, we use an hourly dataset (8760 total data points) that provides solar irradiance, sun position (azimuth and elevation) and temperature. However, the methods presented here are also suitable for higher resolution data. To improve efficiency, we exclude data points where the system generates no power, such as during nighttime or when solar irradiance is zero. For the location studied in here (Madrid, Spain), this filtering reduces the simulation dataset by approximately half, to roughly 4000 relevant data points.

The high computational cost of using detailed simulation methods with large, year-long datasets presents a significant challenge. We address this challenge by developing methods to reduce the size of the operating conditions dataset used for simulation, with the goal of minimizing execution time while controlling the impact on accuracy.

The proposed approaches reduces data redundancy by aggregating similar operating conditions. For instance, solar conditions at a specific hour, such as 11:00, are often nearly identical across consecutive clear-sky days. Instead of simulating each of these points individually, we can use a single representative point (such as their mean value) to obtain a good approximation of the power output for that period, thus reducing the size of the input dataset.

To automatically group similar operating conditions, we employ the k-means clustering algorithm, a widely used unsupervised machine learning technique. This algorithm partitions the original data into a predefined number of clusters based on similarity. We then select the centroid of each resulting cluster as a single representative data point. The collection of these centroids constitutes the new, reduced dataset for our simulation. Figure 3.1 illustrates this data reduction concept, showing how a large dataset can be effectively summarized by just four cluster centroids.

Our simulation framework implement the k-means algorithm by making use of the scikit-learn library (Pedregosa et al. 2011). Its documentation describe the procedure to execute the algorithm in a data set. As already mentioned intuitively, in our case the algorithm divides a dataset of T data points $x_1, ..., x_T$, conformed by the operating conditions mentioned before, into k disjoint clusters. These clusters are described by the mean $\mu_j, j = 1, ..., k$, of the data points inside each cluster. These mean representative points for each cluster are commonly called "centroids". With this in mind, the k-means algorithm aims to choose these k centroids as the ones that minimize the inertia (or within-cluster sum-of-squares, $WCSS$), defined as

$$WCSS = \sum_{t=0}^{T} \min_{\mu_j}(||x_t - \mu_j||^2). \tag{3.1}$$

The steps the algorithm follow for determining the centroids are the following:

1. The algorithm is initialized by selecting k initial centroids. A simple way is to select k random points from the dataset. However, scikit-learn implements a smarter way of selecting the initial

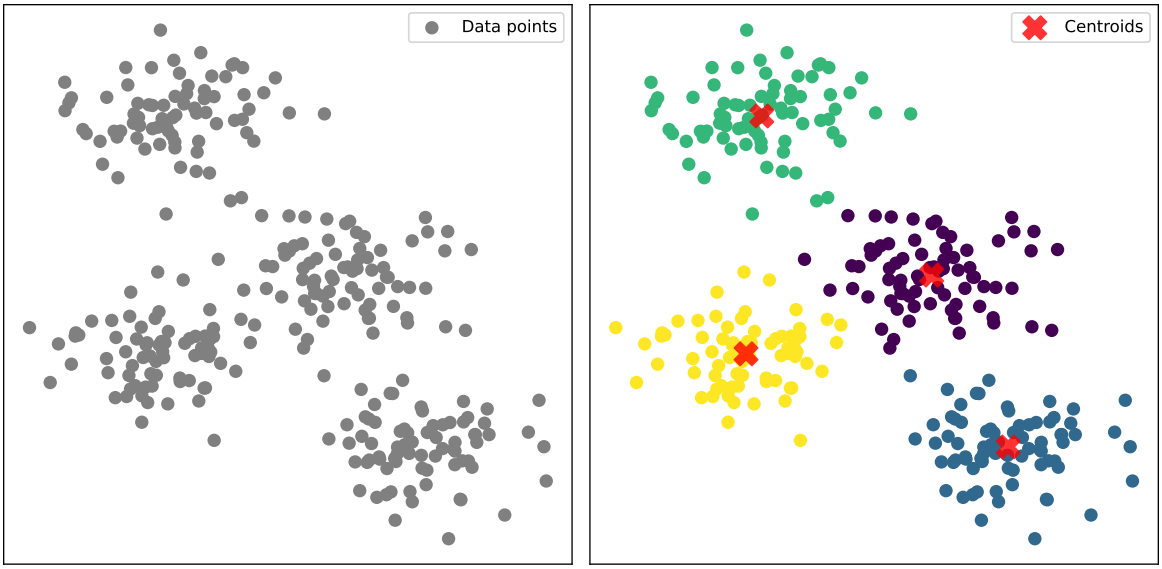

Figure 3.1: K-means algorithm intuition. A vast dataset (gray dots in the left graph) is summarized by just four cluster centroids (red crosses in the right graph).

centroids to speed up convergence (Arthur and Vassilvitskii 2007).

2. With those k initial cluster, the algorithm assign each data point to its nearest centroid, conforming a cluster associated with each one of those centroids.

3. For each one of the cluster formed, the mean value of all data points assigned to that cluster is computed. The centroids are actualized to be these mean values.

4. If there is a significant variation in the position of this new centroids with respect to the previous ones, the algorithm returns to step 2. If the centroids do not move significantly, the algorithm ends.

This algorithm always converge. However, it can get stuck into a local minimum. The smart selection of the initial centroids implemented in scikit-learn generally solves this problem, as it selects the centroids in a way they are distant from each other.

Even though this is the base algorithm for aggregating similar operating conditions, it can be applied following different procedures. In this work, we propose and evaluate two distinct approaches for this aggregation: StraightForward Aggregation (SFA) and Hierarchical Hourly Aggregation (HHA). As a brief overview, the SFA method applies the k-means algorithm in a more direct way, while the HHA method first aggregates the dataset into groups by the corresponding hour of the day, and after that it applies the k-means algorithm inside each hour group. This second method needs a dynamic way of selecting the number of clusters for each hour of the day group, as we will describe in the following sections. That makes this second method more complex than the SFA one.

After aggregating the original dataset and obtaining a new reduced dataset of representative operating conditions, we can feed this last one to the simulation and assign the representative results to their corresponding original data points. However, this simulation can be performed using any method from the ones described in Chapter 2. In order to be able to measure the impact in the accuracy and speed of the approximation through the aggregation methods, we use in this step the same method as the one we use as the precise non-aggregated baseline. We already discussed the selection of the submodule-level simulation method for this purpose, so this is the one we use.

Aggregating the data and performing the simulation on the reduced dataset using the sumodule-level simulation method produces faster results. However, this speed up comes with an accuracy cost. This behavior recalls us the discussion made in Section 2.1 about the selection of the granularity of the physical model used in the simulation. We considered three levels of granularity for the simulations that takes into account shading: module-level, submodule-level and cell-level. If we want faster results than the ones from the submodule-level simulation, we are forced to reduce the granularity to the module-level simulation. However, this is a great jump in accuracy loss, especially for scenarios with high partial shading presence.

With the methods presented in this work, we aim to improve the accuracy of the module-level simulation, while maintaining (or reducing in the best case) the computational time. For this, we show the results of applying the aggregation methods over the submodule-level simulation and compare it with the baseline of the module-level simulation to check if we improve its performance.

3.1 StraightForward Aggregation (SFA)

As its name suggests, the StraightForward Aggregation (SFA) method applies the k-means algorithm directly to the entire dataset of approximately 4000 operating conditions (once non-relevant operating conditions are removed). In order to determine the number of clusters to perform, so that the method is versatile enough to adapt to datasets of different sizes, it uses a parameter for the percentage of data reduction to perform. For example, to achieve a 80% reduction, we set the number of clusters (k) to 20% of the original dataset size, resulting in

$$k = dataset\_size \times (1 - reduction) = 4000 \times (1 - 0.80) = 4000 \times 0.20 = 800 \text{ clusters.}$$

We then perform the simulation only on the centroids of these k clusters, that will conform our new input dataset of representative operating conditions. To reconstruct the full annual time series of MPP values, we assign the power output calculated for each centroid to all original data points within that centroid's corresponding cluster.

This clustering process can be made in any set of features from the dataset. As we already mentioned, our dataset contains the irradiance, sun position and temperature of the system. As temperature can be obtained from irradiance (and they are directly proportional), we found it not relevant to include it in the clustering process. To determine the optimal set of features for clustering, we evaluated various combinations of the input variables: solar irradiance, sun elevation and sun azimuth. The brief analysis concluded that aggregating the data based on all three variables simultaneously yields the most accurate results. Consequently, the methods described in this work perform clustering in this 3-dimensional feature space. However, the analysis of this variable selection can be explored more deeply, even though initial guesses indicate that aggregating on all 3 variables is the best choice.

We show the performance of the SFA method across a range of data reduction percentages for all simulation scenario from Table 1.1 in Figure 3.2. The top plot shows the relative errors of the SFA method, calculated using (2.1), with the submodule-level simulation performed on the complete, non-aggregated dataset as the reference. The bottom plot shows the corresponding relative execution times, calculated using (2.5). For comparison, we also include the performance of the simpler module-level simulation as a baseline. The mean error and times for this baseline are indicated by dashed lines, while the minimum and maximum values across all scenarios are shown as dotted lines. This allows a direct comparison of the SFA method's accuracy and speed against the next faster, but less detailed, simulation approach.

The results reveal the expected trade-off: higher data reduction percentages lead to faster execution times at the cost of increased error. For instance, an 80% data reduction reduces computation time by approximately 80% while introducing a mean relative error of about 2.5%.

Notably, an optimal range for the SFA method appears between 60% and 80% reduction. Within this "sweet spot", the SFA method not only maintains higher accuracy than the module-level simulation

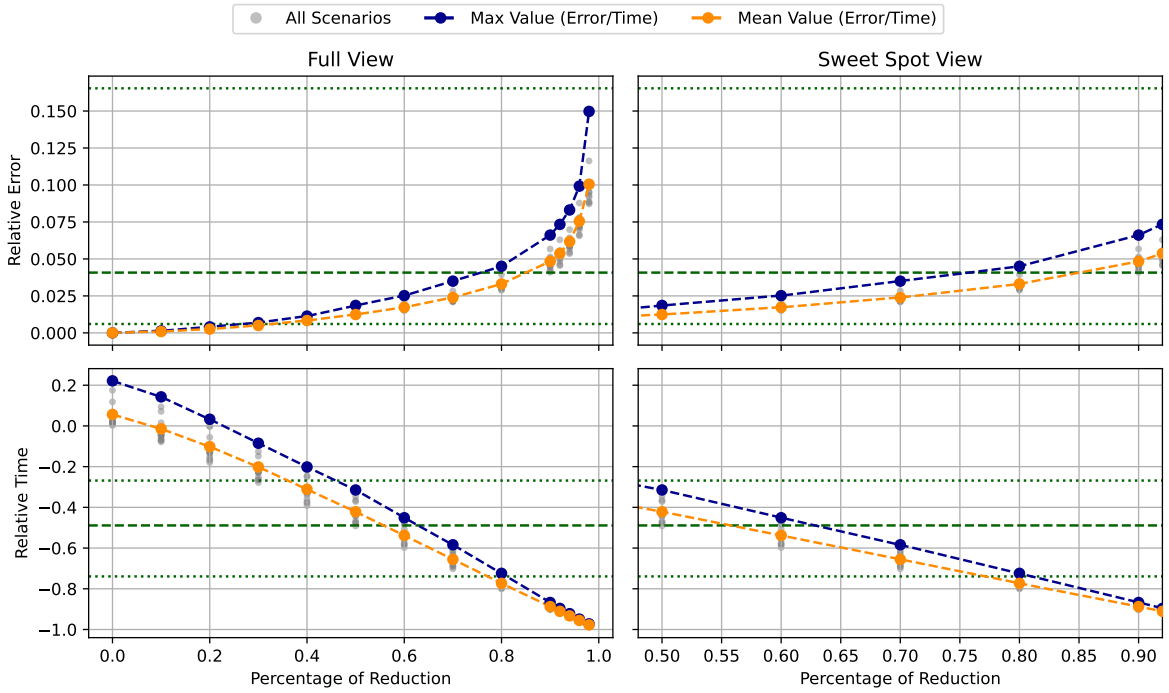

Figure 3.2: Relative errors (top) and execution times (bottom) of the SFA method for each simulation scenario for a grid of data reduction percentages, using the submodule-level simulation as the reference. The performance of the module-level simulation is plotted as dotted lines (minimum and maximum values) and dashed lines (mean values across all simulation scenarios). The right-hand graphs show a zoomed-in view of the x-axis to highlight the "sweet spot".

but also exceeds its computational speed. This demonstrates a key advantage, providing a solution that is both faster and more precise than the next simplest modeling approach.

To further analyze these results, Figure 3.3 directly compares the module-level simulation with the SFA method using an 80% data reduction. The figure displays the distributions of relative errors (left) and relative execution times (right), using the full submodule-level simulation as the reference for both.

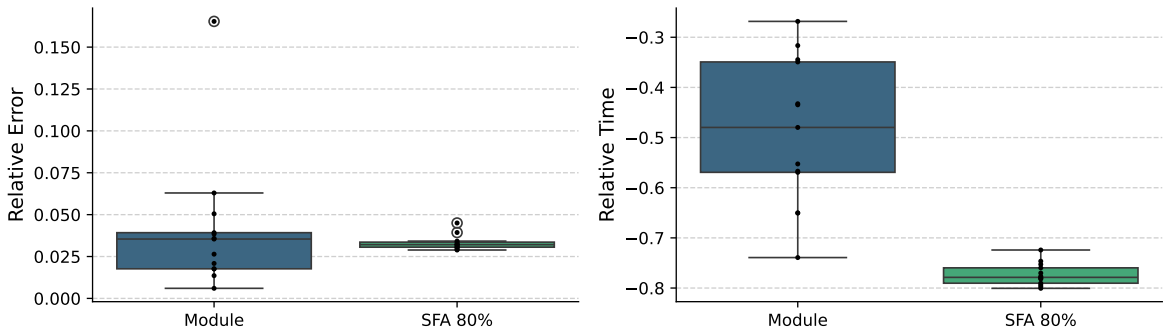

Figure 3.3: Relative errors (left) and execution times (right) of the simulation at the module level and the one performing the SFA with 80% of reduction.

In terms of accuracy, the SFA method achieves a mean error comparable to the module-level

simulation. However, the distribution of SFA errors is significantly less dispersed. This indicates that the SFA method is more robust and provides more consistent results, especially in scenarios with high partial shading, making it's accuracy loss more predictable in those scenarios.

Regarding execution time, the SFA method not only achieves greater speedup (approximately 80% reduction) but also demonstrates more predictable performance. The execution time of the module-level simulation varies considerably depending on the specific scenario, whereas the SFA method's runtime remains far more consistent.

A key advantage of the SFA method is its configurability. Based on the performance curves in Figure 3.2, users can select a specific data reduction percentage to achieve a desired trade-off between computational speed and simulation accuracy. This flexibility allows the method to adapt to different application requirements.

While any reduction level is possible, the most compelling results occur for parameters in the 60% to 80% reduction range. As previously noted, this window provides a solution that is not only faster but also more accurate than the baseline module-level simulation, representing the optimal operational "sweet spot" for this approach.

3.2 Hierarchical Hourly Aggregation (HHA)

An alternative approach, the Hierarchical Hourly Aggregation (HHA) method, works in a hierarchical way, as it first partitions the data before applying clustering. In the initial step, we divide the dataset into 24 distinct groups, one for each hour of the day. We then apply k-means clustering independently within each of these hourly groups.

This hierarchical structure introduces a new challenge: determining the optimal number of clusters (k) for each hourly group, as intra-group variability is not uniform. For example, all data points for nighttime hours (e.g., 03:00) can be summarized by a single centroid ($k = 1$), since solar irradiance is consistently zero. In contrast, conditions at midday (e.g., 12:00) exhibit significant seasonal variation, with much higher irradiance in summer than in winter, thus requiring a larger k to represent them accurately. We show this in Figure 3.4, where we represent the dispersion of the irradiance values of Madrid, Spain, for each hour group. The plot exposes the previous intuition, where midday irradiance values present more variability than the ones at the last or first hours of the day.

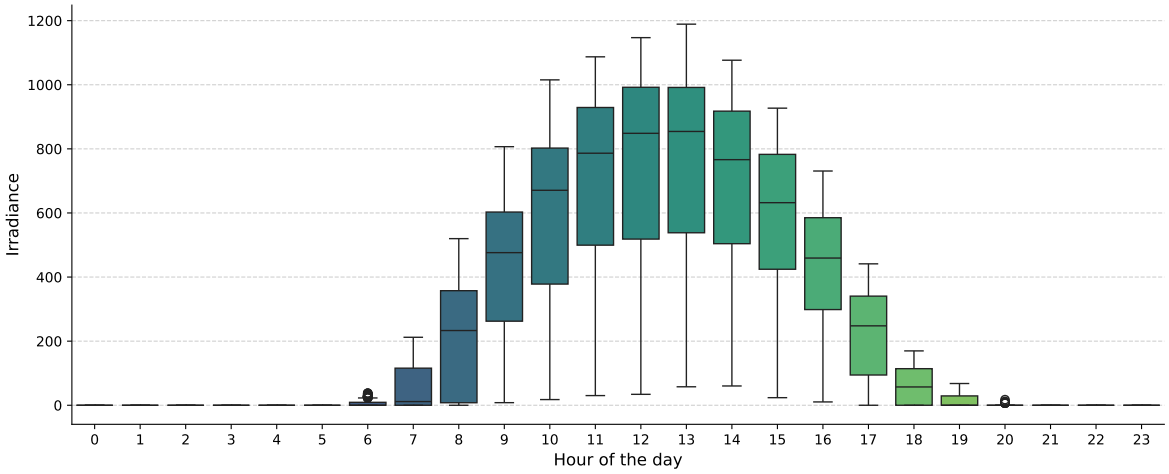

Figure 3.4: Irradiance distribution for each hour in Madrid, Spain.

To dynamically determine the number of clusters for each hourly group, we introduce a new hyper-parameter: a target for the percentage of explained variability, or precision per hour. This approach

selects the minimum number of clusters (k) required to account for a specified portion of the data's variance within each hourly subgroup. For instance, if this target is set to 99%, the algorithm determines, for each hour, the number of clusters needed to explain 99% of that group's internal variability. This ensures that hours with high operating conditions variance (e.g., midday) receive more clusters than hours with low variance (e.g., nighttime).

We base this method on the procedure detailed in Caliński and Harabasz (1974). Given a number of clusters k over a dataset $x_1, ..., x_T$, with centroids $\mu_1, ..., \mu_k$, we can compute the explained variability of the clusters obtained by the k-means algorithm using

$$Explained\ Variability = \frac{BCSS}{TSS} = \frac{TSS - WCSS}{TSS} = 1 - \frac{WCSS}{TSS}, \tag{3.2}$$

where TSS (Total Sum of Squares) is the sum of squared distances between each point and the overall mean of the data,

$$TSS = \sum_{t=1}^{T} ||x_t - \mu||^2, \tag{3.3}$$

with $\mu = \frac{\sum_{t=1}^{T} x_t}{T}$ the overall mean, $WCSS$ (Within-Cluster Sum of Squares or inertia) is the sum of squares between each point and its assigned cluster centroid, previously presented in (3.1) as

$$WCSS = \sum_{t=0}^{T} \min_{\mu_j}(||x_t - \mu_j||^2).$$

and $BCSS$ (Between-Cluster Sum of Squares) represents the variance between clusters and is obtained as

$$BCSS = TSS - WCSS. \tag{3.4}$$

With these definitions in mind, the algorithm follows these steps:

1. Select a desired percentage of explained variability (precision per hour) and group the data into hourly groups.
2. Loop from the first hourly group to the last one.
3. Loop from $k = 1$ clusters to a maximum number of clusters $k = k_{max}$.
4. For that hourly group, determine the explained variability using (3.2) of applying the k-means algorithm to obtain k clusters. This steps needs to apply the k-means algorithm for k clusters in the hourly group.
5. If that explained variability exceeds the desired one, then $k_{opt} = k$ is the desired amount of clusters for this hourly group, so this last aggregation in k_{opt} clusters is stored and the loop from step 3 is broken, if not, the loop continues.
6. Continue the loop from step 2 until finishing the hourly groups.

With these steps, we get an aggregation of data points inside each hourly group. However, the process of applying the k-means algorithm for every individual value of the number of clusters k until achieving the desired explained variability can be computationally intensive. A solution to this could be exploring the amount of clusters incrementing the value of k by more then one on each iteration of the loop in step 3. However, for the purposes of this work, we will explore it one by one. We show the performance of the HHA method in Figure 3.5 in the same way we presented the ones of the SFA method.

As expected, setting a higher target for the explained variability results in improved accuracy but also increases the computational time. When compared to the previous SFA method (Figure 3.2),

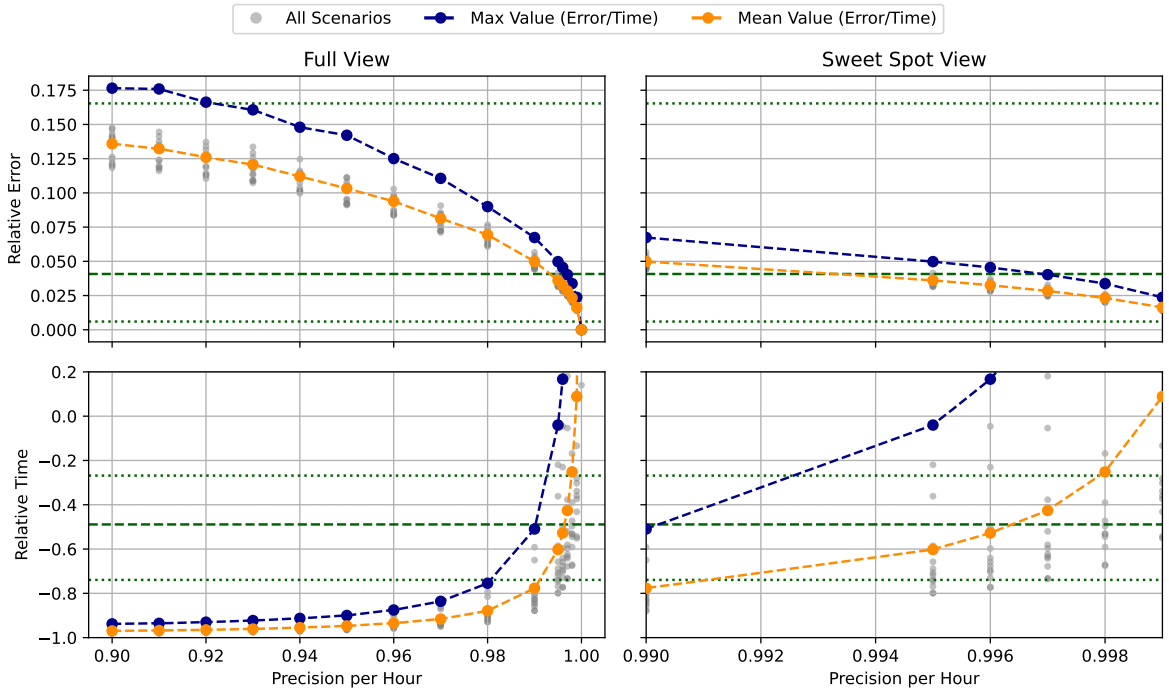

Figure 3.5: Relative errors (top) and execution times (bottom) of the HHA method for each simulation scenario for a grid of precision per hour percentages, using the submodule-level simulation as the reference. The performance of the module-level simulation is plotted as dotted lines (minimum and maximum values) and dashed lines (mean values across all simulation scenarios). The right-hand graphs show a zoomed-in view of the x-axis to highlight the "sweet spot".

the HHA approach achieves a similar level of accuracy and robustness across the different scenarios. However, it is computationally inferior. The execution time is not only longer, but its variability between scenarios is also significantly greater. This reduced efficiency is attributed to the computational overhead required to dynamically determine the optimal number of clusters for each of the 24-hourly groups. This selection process can be time-consuming, and the cost is not always offset by accuracy gains, particularly for smaller simulation scenarios.

The results indicate an optimal performance range for the HHA method when the explained variability target is set between 99.5% and 99.7%. To examine this in detail, Figure 3.6 compares the HHA method (using a 99.6% target) against the module-level simulation, using the non-aggregated submodule-level simulation as the reference.

In terms of accuracy, the HHA method offers comparable, or even superior, performance, demonstrating greater robustness across different scenarios. However, the opposite is true for computational time. This highlights the previously discussed issue: the method's significant overhead makes it ill-suited for smaller scenarios which show the highest relative execution times. If we exclude these specific scenarios from the analysis, the average performance of the HHA method would more closely resemble that of the SFA approach. We show the same results from Figure 3.6, but without the scenarios formed by one single module in Figure 3.7. We can see that, excluding the small scenarios, the HHA method performs as well as the SFA one. However, the bad behavior of this method in small scenarios makes the SFA method more interesting in general.

Although the HHA method is outperformed by the SFA one in this study, further refinement could improve its performance. As already discussed, the current implementation uses a linear search to determine the number of clusters k per hour, incrementing k by one until it meets the target for

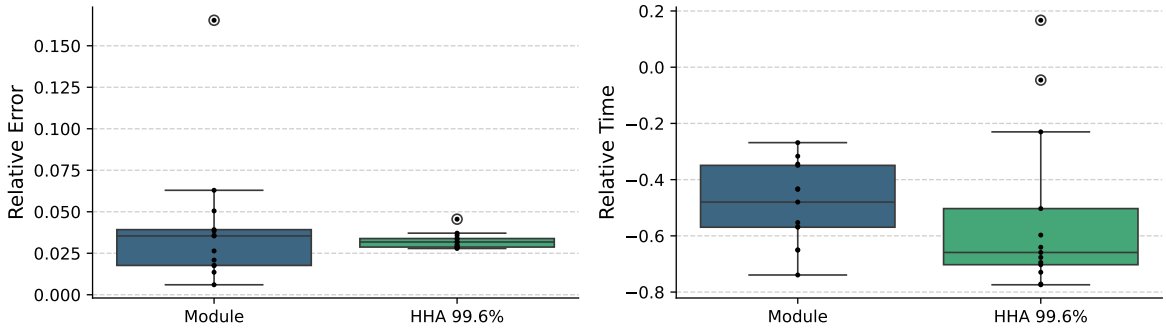

Figure 3.6: Relative errors (left) and execution times (right) of the simulation at the module level and the one performing the HHA with 99.6% of precision per hour.

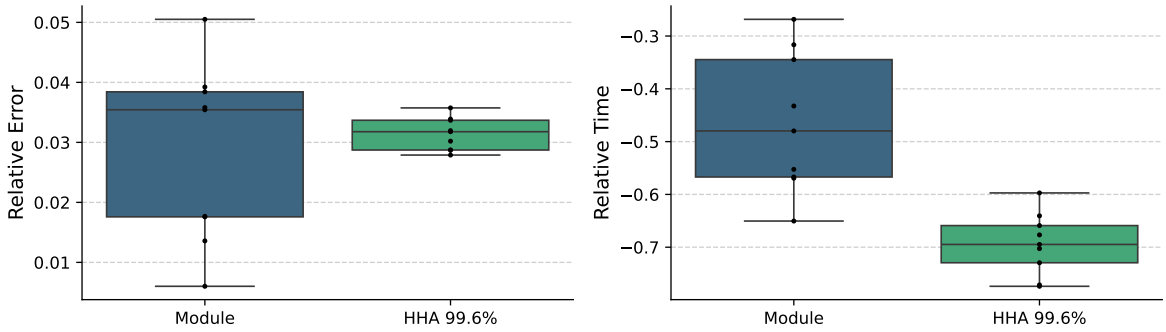

Figure 3.7: Relative errors (left) and execution times (right) of the simulation at the module level and the one performing the HHA with 99.6% of precision per hour for scenarios with more than one module.

explained variability. This process could be significantly accelerated by using a larger step size, thereby reducing the method's computational overhead. As the SFA method already provided excellent results, we did not pursue this optimization further in this work. Nonetheless, we present the HHA concept here as a promising alternative that, with such modifications, could become a competitive approach for future research.

3.3 Aggregation methods comparison

To directly compare the two aggregation methods, we need a metric that can describe both methods in the same plot. For this, from Figure 3.2 and Figure 3.5, we can extract the orange lines for each method, corresponding to the average performance of them. We compare the methods according to the average trade-off between speed and accuracy using those values. However, we can also extract the blue lines corresponding to the maximum values, that represent the worst case scenario for the methods, with the highest error and lowest computational time saved.

We plot the mean relative errors against the mean relative execution time for both the SFA and HHA approaches in Figure 3.8. Each point on the plot represents the performance of a method at a specific parameter setting, averaged across all simulation scenarios. The plot focuses on negative relative times, as these values indicate a computational speedup. For comparison, we also show the fixed performance of the baseline module-level simulation, which achieves an average time reduction of approximately 50% at a mean error of about 4%. This visualization provides a clear comparison of the

average accuracy-speed trade-off for both proposed methods relative to each other and to a standard, less detailed approach.

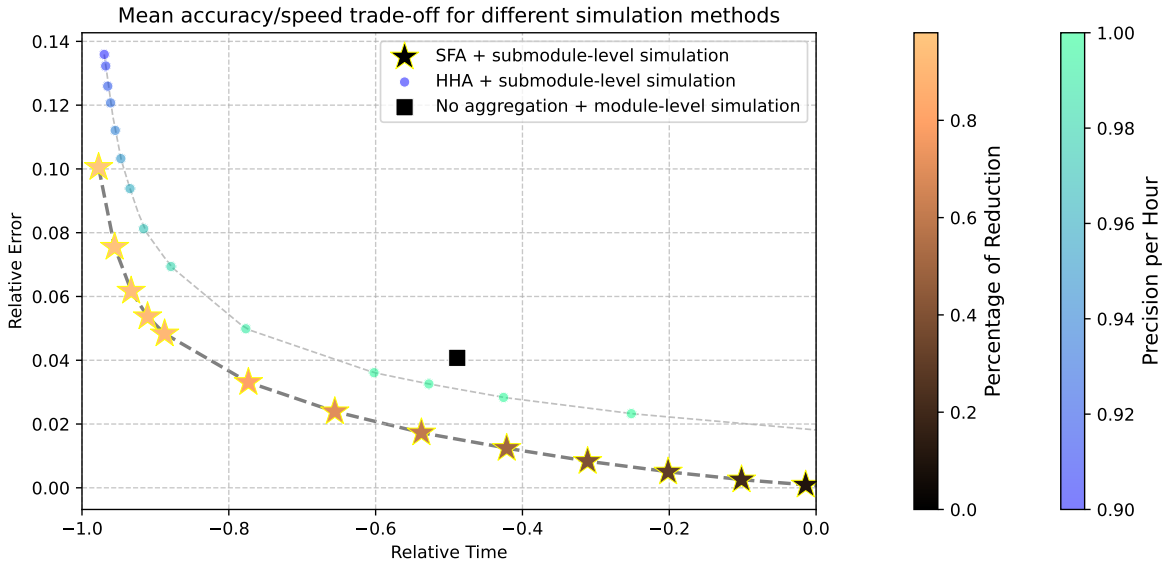

Figure 3.8: Mean relative times versus mean relative errors for the SFA and HHA methods for a range of parameters.

The results demonstrate a clear superiority of the StraightForward Aggregation (SFA) method over both the HHA and the submodule-level simulation in terms of average performance. The SFA method consistently defines the Pareto Frontier (Lotov and Miettinen 2008) for the average speed-accuracy trade-off, providing the best possible accuracy for any given level of computational speedup. This behavior does not only occur for average performance. We show a similar plot of the maximum accuracy-speed trade-off in Figure 3.9. The plot is similar to the previous one, but the values correspond to the maximum errors and times extracted from the blue lines from the figures previously mentioned. In other words, they serve as a worst case scenario in terms of speed-accuracy trade-off.

We can see the same trend as in the previous figure, demonstrating the SFA method even higher dominance in this metric. This is caused by the previously discussed phenomenon: module-level simulation produces a high error in worst case scenarios (high partial shading), while HHA method behaves similar to the SFA one in terms of errors, but with significantly lower computational time reduction in worst case scenarios (small systems).

The results presented in this section demonstrate the clear superiority of the SFA aggregation method over the submodule-level simulation, which outperforms the performance of reducing the detail of the physical model of the simulation to the module level. Furthermore, this approach offers a versatile mechanism to tune this trade-off as needed. For instance, focusing on Figure 3.8, users can achieve an 80% reduction in computational time while introducing a relative error of only about 3%, all in average performance terms. This same principle can be applied to target any desired balance between computational cost and precision, highlighting the method's practical flexibility. This characteristic contrasts with the unique trade-off value offered by the module-level simulation, that do not provide any way to tune its performance to match different use cases.

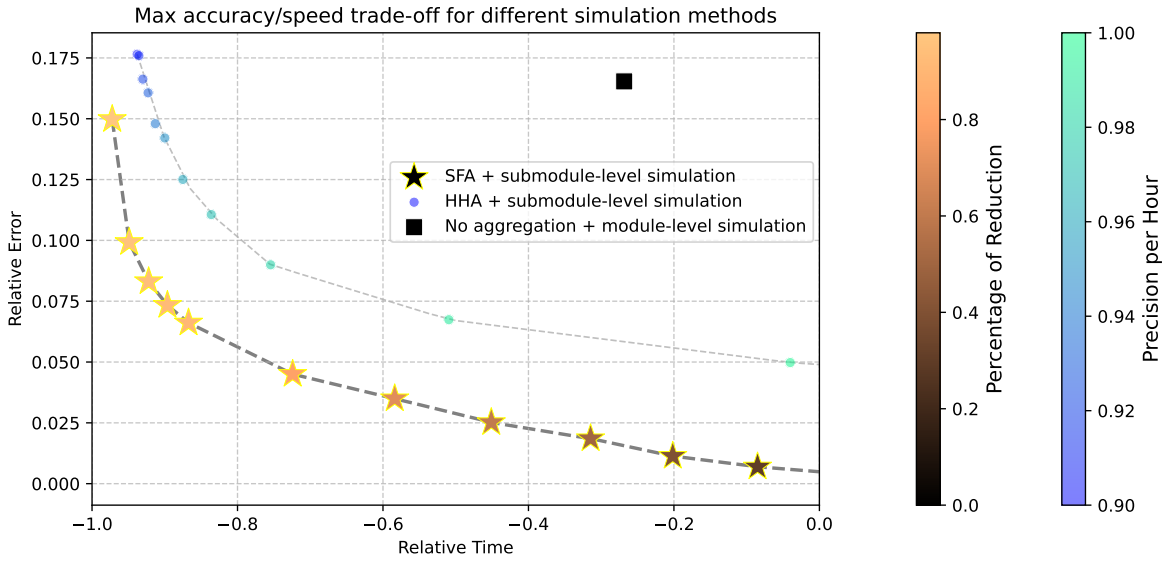

Figure 3.9: Maximum relative times versus mean relative errors for the SFA and HHA methods for a range of parameters.

3.4 SFA method for shading power losses

A primary application of detailed PV system simulations, and a central focus of research at ieco.io, is to quantify the annual energy losses caused by partial shading. The aggregation methods developed in this work are highly effective for this task, as they significantly reduce the required computational time while maintaining the accuracy of the submodule-level simulation. To formally analyze these losses, we define the hourly energy loss due to partial shading with

$$MPP_{loss} = MPP_{no\_shading} - MPP_{shading}. \tag{3.5}$$

To evaluate how accurately the method estimates the annual power loss from shading, we adapted the metric from (2.2). For this analysis, the error is calculated by substituting the total hourly power outputs with the hourly shading loss values (MPP_{loss}) using the full submodule-level simulation as the reference. The use of relative difference instead of relative error is motivated on the discussion made when these metrics were presented. As we aim to compare the accuracy on the power output of the system for the whole year, we will not need a metric that compares the hourly precision of the simulation, as we are just concerned about the long-term accuracy of the method. Relative difference is the appropriate metric for this. The results are presented in a similar way to Figure 3.2, but showing relative differences instead of relative errors, in Figure 3.10. This figure compares the performance of the SFA method in estimating shading losses against the module-level simulation, which serves a a performance baseline.

The presented results confirm that the SFA method significantly outperforms the standard module-level simulation for estimating shading losses. On average, the module-level simulation underestimates the annual energy loss by approximately 35% compared to the reference full submodule-level simulation, while only being twice as fast. In contrast, the SFA method shows negligible differences in loss estimation even with data reduction percentages as high as 90%, while drastically reducing computational time. This demonstrates that the SFA approach provides a far superior balance of speed and accuracy for this application.

In summary, these results demonstrate that the proposed data aggregation method serves as a highly effective tool for quantifying annual energy losses from partial shading in a fast way. The

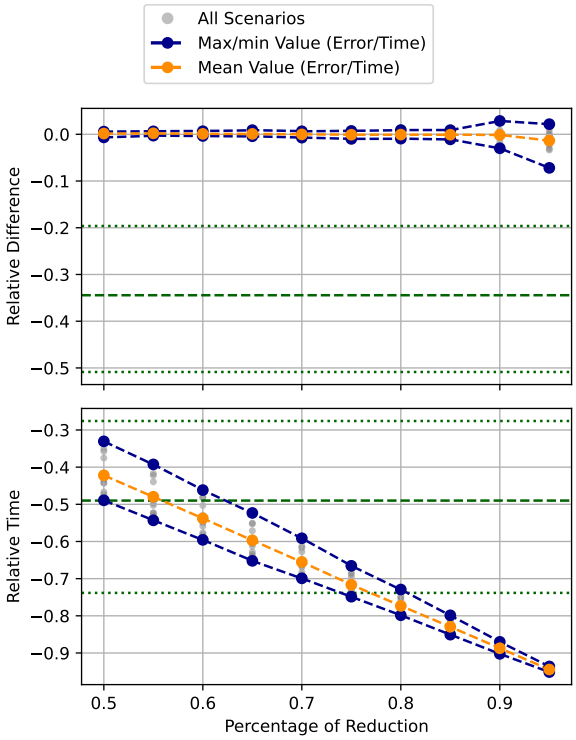

Figure 3.10: Relative differences (top) and execution times (bottom) of the SFA method for each simulation scenario for a grid of data reduction percentages, using the submodule-level simulation as the reference. The performance of the module-level simulation is plotted as dotted lines (minimum and maximum values) and dashed lines (mean values across all simulation scenarios). The right-hand graphs show a zoomed-in view of the x-axis to highlight the "sweet spot".

approach reduces computational time by up to 90% while having negligible impact on the simulation's accuracy.

Chapter 4

Conclusions

This work addressed the significant computational challenge of performing accurate, year-long energy yield assessments for photovoltaic systems, particularly under the complex conditions of partial shading. We demonstrated that a key bottleneck arises from the sheer volume of time-step data required for an annual simulation.

To overcome this, we introduced and evaluated two novel statistical aggregation methods, Straight-Forward Aggregation (SFA) and Hierarchical Hourly Aggregation (HHA), designed to reduce the size of the input dataset while preserving the high-resolution information critical for accurate shading analysis. Both methods leverage k-means clustering to group similar instantaneous operating conditions, defined by solar irradiance, elevation, and azimuth, and use the resulting cluster centroids as representative points for the simulation.

Our findings clearly establish the superiority of the SFA method. This approach not only proved more computationally efficient than the HHA method but also consistently outperformed the standard, less-detailed module-level simulation across all evaluated metrics. The SFA method successfully defines a new Pareto frontier for the speed-accuracy trade-off, providing the best possible accuracy for any given level of computational speedup. A key advantage of SFA is its tunability; for example, users can achieve an 80% reduction in simulation time while introducing a mean relative error of only about 3%, but also achieve a 70% reduction in simulation time introducing a mean relative error of about 2%. This delivers robust, predictable performance that remains consistent across different scenarios, outperforming simpler modeling approaches.

Crucially, the SFA method proved exceptionally effective for quantifying annual energy losses due to shading, a primary application of detailed PV simulations and a central focus of research at ieco.io. While the module-level simulation underestimated these losses by an average of 35%, our SFA approach reduces computational time by up to 90% with negligible impact on accuracy.

In conclusion, this work presents a data-centric framework that significantly accelerates PV simulations without compromising the fidelity required to model non-linear shading effects. The SFA method offers a practical, flexible, and powerful tool for engineers and researchers, enabling rapid design iterations and reliable financial assessments for PV projects in ieco.io and in any environment.

4.1 Future work and limitations

This work present two data-centric aggregation methods in order to accelerate the traditional PV system simulation methods. However, the methods were made in a simple way, presenting two different basic methods for aggregating initial data. The methods can be developed in a more sophisticated way, considering other variable, steps or some other type of procedures such as another hierarchical structure as in the HHA method. In addition to this, the presented results of the performance of the methods were computed using the submodule level simulation as the reference one. Even though this

is the most used simulation method, other methods such as the module-level or cell-level simulation can better suit specific scenarios. For example, if we need a highly precise simulation, a cell level simulation is mandatory, specially in scenarios with a lot of partial shading.

With these considerations, some possible future work to further develop the methods presented in this work are the following:

- Measure the performance of the present methods using another simulation method as the base method, such as the cell-level simulation. This can be crucial to consider in scenarios that require high fidelity results.
- Measure the performance of the methods using higher resolution meteorological data. This can make the initial data aggregation more relevant, as increasing the resolution of the input data means increasing its size, making the simulation computationally more expensive. Moreover, with more data points, there is a higher chance of getting a greater amount of similar operating conditions, making the aggregation methods more effective.
- Modify the aggregation methods in some way, making them perform better. The modifications can be done in a lot of ways, such as changing the aggregation algorithm, the parameters of it or even the steps followed or variable used. The methods presented in this work were made as simple as possible in order to serve as a base to their further development.
- Measure the precision of the methods in other simulation scenarios. In this work, we considered just thirteen simple simulation scenarios. However, real-life scenarios are more complex. It would be interesting to measure the precision of the methods in complex real-life scenarios, where a lot of shading objects are considered.

As we present a simple initial study of these aggregation methods, some further improvements of them could make them more effective and interpretable. This work aimed to show some simple initial data aggregation methods that work very well in certain situations even though they are not very sophisticated. These methods can make the building block for more complex and effective initial data aggregation methods or work as a simple way to reduce the computational time while being aware of the precision loss.

Bibliography

[1] Alrahim Shannan NMA, Yahaya NZ, Singh B (2013) Single-diode model and two-diode model of PV modules: A comparison. IEEE International Conference on Control System, Computing and Engineering, Penang, Malaysia, pp 210-214.

[2] Anderson K, Hansen C, Holmgren W, Jensen A, Mikofski M, Driesse A (2023) pvlib python: 2023 project update. Journal of Open Source Software 8:5994.

[3] Arthur D, Vassilvitskii S (2007) k-means++: the advantages of careful seeding. En: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, Philadelphia, pp 1027-1035.

[4] Blanco Aguiar A, González Rodríguez B, Martínez Barbeito M, Sánchez de León Peque M (2025) Accelerating Photovoltaic System Simulations via Statistical Data Aggregation. 42nd European Photovoltaic Solar Energy Conference and Exhibition. doi: 10.4229/EUPVSEC2025/4AO.9.5.

[5] Caliński T, Harabasz JA (1974) A Dendrite Method for Cluster Analysis. Communications in Statistics - Theory and Methods 3:1-27.

[6] Fahy K, Stadler M, Pecenak ZK, Kleissl J (2019) Input data reduction for microgrid sizing and energy cost modeling: Representative days and demand charges. Journal of Renewable and Sustainable Energy 11:065301.

[7] Gafurov T, Prodanovic M, Usaola J (2013) PV system model reduction for reliability assessment studies. 4th IEEE/PES Innovative Smart Grid Technologies Europe (ISGT Europe 2013), pp 1-5.

[8] Granger BE, Pérez F (2021) Jupyter: Thinking and Storytelling With Code and Data. Computing in Science & Engineering 23:7-14.

[9] Gray JL (2011) The Physics of the Solar Cell. En: Luque A, Hegedus S (ed) Handbook of Photovoltaic Science and Engineering. John Wiley and Sons, Chichester, pp 82-129.

[10] Honsberg CB and Bowden SG (2019) Photovoltaics Education Website, www.pveducation.org. Accessed 2 January 2026.

[11] Huld T, Müller R, Gambardella A (2012) A new solar radiation database for estimating PV performance in Europe and Africa. Solar Energy 86:1803-1815.

[12] Likas A, Vlassis N, Verbeek JJ (2003) The global k-means clustering algorithm. Pattern Recognition 36:451-461.

[13] Lotov AV, Miettinen K (2008) Visualizing the Pareto Frontier. En: Branke J, Deb K, Miettinen K, Slowiński R (ed) Multiobjective Optimization. Lecture Notes in Computer Science, vol 5252. Springer, Berlin, Heidelberg.

[14] McGill R, Tukey JW, Larsen WA (1978) Variations of Box Plots. The American Statistician 32:12-16.

[15] Milosavljevic D, Kevkić T, Jovanovic S (2022) Review and validation of photovoltaic solar simulation tools/software based on case study. Open Physics 20:431-451.

[16] Miraftabzadeh SM, Colombo C, Longo M, Foiadelli F (2023) K-Means and Alternative Clustering Methods in Modern Power Systems. IEEE Access 11:1-1.

[17] Okif M, Meena S, Lal S, Prajapati R, Meena A (2025) Machine Learning-Based Performance Prediction Model For Solar PV Systems Using Meteorological Inputs. International Journal of Environmental Sciences:872-885.

[18] Pedregosa F, et al. (2011) Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12:2825-2830.

[19] Python Software Foundation (2024) Python Language Reference, version 3.13. http://www.python.org.

[20] Sandia National Laboratories (2025) PV Performance Modeling Collaborative (PVPMC). https://pvpmc.sandia.gov/. Accessed 22 December 2025.

[21] Saeed F, Tauqeer HA, Gelani HE, Yousuf MH, Idrees A (2022) Numerical modeling, simulation and evaluation of conventional and hybrid photovoltaic modules interconnection configurations under partial shading conditions. EPJ Photovoltaics 13:10.

[22] Shanghai JA Solar Technology Co., Ltd (2020) Mono 465W MBB Half-Cell Module JAM72S20 440-465/MR/1000V Series. https://www.jasolar.com/uploadfile/2020/0619/20200619040220997.pdf. Accessed 10 November 2025.

[23] Zhang HX, Zhuang H, Gou XF, Huang QS, Jiang LK, Chen ZY (2017) Study on the Benefit of Half-Cut Cells towards Higher Cell-To-Module Power Ratio. Power and Electrical Engineering, 978-1.