Automated Algorithm Design of Tailored Metaheuristics for Photovoltaic Parameter Estimation in Single- and Double-Diode Models

Daniel F. Zambrano-Gutierrez*, Grecia C. Duque-Gimenez†, José C. Ortiz-Bayliss*, Itzel Aranguren‡, Diego Oliva‡, and Jorge M. Cruz-Duarte§

*Tecnologico de Monterrey, Monterrey, Mexico, E-mails: a00836756@exatec.tec.mx, jcobayliss@tec.mx

†Universidad Autónoma de Nuevo León, Monterrey, Mexico, E-mail: grecia.duqueg@uanl.edu.mx

‡Universidad de Guadalajara, Guadalajara, Mexico, E-mails: itzel.aranguren@academicos.udg.mx, diego.oliva@cucei.udg.mx

§University of Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, Lille, France, E-mail: jorge.cruz-duarte@univ-lille.fr

***Abstract*—Automated Algorithm Design and Configuration (AADC) reduces the manual effort required to select and tune population-based optimizers for engineering tasks. We apply AADC to photovoltaic parameter estimation under the Single-Diode (SD) and Double-Diode (DD) models via a two-stage pipeline that comprises a tailored metaheuristic and a subsequent refinement of its hyperparameters. In the design stage, a sampling-based hyper-heuristic with local search builds MHs by selecting and ordering search operators from a predefined heuristic space, and then tunes the selected structure using Optuna. Experiments on the R.T.C. France cell dataset show that DD renders a more stable design outcome than SD, while both models share the same best structure, a particle-swarm-based operator followed by a differential mutation. After fixing this structure, further tuning improves performance by lowering the median objective value and reducing dispersion across runs. The final SD and DD fits closely match the measured I–V and P–V characteristics, yielding physically plausible parameter estimates.**

***Index Terms*—Automated Algorithm Design; Automated Algorithm Configuration; Hyper-heuristic; Metaheuristic; Photovoltaic Parameter; Hyperparameter Tuning.**

I. Introduction

The current literature presents a colorful and vibrant palette of successful examples of challenging problems solved using Metaheuristics (MHs) [1], [2]. As a result, MHs have become a common solution approach across myriad scenarios. However, their use is not straightforward, since many MHs exist, each with multiple design options and hyperparameters to configure [3], [4]. Finding an effective heuristic-based algorithm and configuration for real-life engineering problems remains a challenging task [4], [5]. In practice, identifying effective component combinations and tuning typically depends on experience, trial-and-error experimentation, and available computational resources [3]. Hence, systematic and reproducible approaches to algorithm selection and configuration are increasingly needed. This challenge is particularly evident in practical calibration and identification problems, where the optimization process is only one component within a larger pipeline and must satisfy certain requirements, such as robustness across runs and transparent reporting of algorithmic settings [6], [7]. However, manual selection and tuning can lead to configurations that underperform or are overly tailored to a specific experimental setup, making them difficult to replicate [6]. In this context, Automated Algorithm Design and Configuration (AADC) addresses this challenge by automating key parts of algorithm composition and configuration, in that way reducing reliance on ad-hoc handcrafted tuning during algorithm development [3], [4], [8].

In this work, we explore how to apply AADC to MHs for Photovoltaic (PV) parameter estimation, a nonlinear inverse problem in which we infer model parameters from measured current-voltage data [9], [10]. Accurate parameter estimation is essential for reliable simulation, diagnostics, and control of PV devices [9], [11]. Although this problem has been widely studied using MHs [12], most prior work focuses on proposing or adapting a specific optimizer rather than automating the design of the optimizer itself. In our approach, we do not fix the optimizer in advance. Instead, we construct it automatically as we search for a solution.

From an AADC perspective, two coupled decisions must be addressed: the algorithmic structure, which components or operators to use and in what order these must be implemented; and the numerical hyperparameters of those components. Exploring both dimensions jointly can be beneficial, but it is often computationally demanding [3], [4], [13]. Hence, we adopt a pragmatic two-stage strategy. First, a Sampling-Based Hyper-Heuristic with Local Search (SBHH-LS) searches the space of operator sequences to identify compact MH structures that perform well under a fixed budget [14]. Then, after fixing the selected structure, Optuna is used to refine its hyperparameters in a controlled and repeatable manner [15]. This work makes three contributions. First, PV parameter estimation under the Single-Diode (SD) and Double-Diode (DD) models is formulated as a low-level optimization problem within the AADC paradigm. Second, we propose and instantiate a two-stage AADC pipeline in which SBHH-LS designs MH structures by selecting and ordering Search Operators (SOs), and Optuna then tunes the hyperparameters of the selected structure. Third, we provide an empirical analysis of the resulting structures and tuning effects, highlighting recurring operator patterns and their impact on robustness. The remainder of the paper is organized as follows. Section II provides the background that founds our approach. Section III presents the AADC pipeline and metric used in this work. Subsequently, Section IV describes the experimental setup, and Section V reports the results. Lastly, Section VI summarizes the insights of this work.

II. Theoretical Foundations

This section presents the key concepts underlying our approach, including Automated Algorithm Design and Configuration (AADC) for Metaheuristics (MHs), and the practical problem of Photovoltaic (PV) modeling.

A. Metaheuristics

MHs are optimization methods that iteratively guide the search toward regions of the decision space where better solutions are more likely to be found. In this work, an MH is modeled as a structured procedure that coordinates one or more Simple Heuristics (SHs) [16] to balance exploration and exploitation, which typically yields more consistent performance than applying isolated heuristics [1], [17]. Following [18], [19], SHs are categorized as Generative (h_g), Perturbative (h_p), and Collector (h_c). The generative heuristic produces the initial candidate solution(s). A perturbative approach, on its own, updates the current state to generate new candidates, potentially using contextual information available during the run (e.g., the incumbent's best value or population statistics). Lastly, a collector implements an acceptance rule and, in practice, is implemented as a selector h_s (e.g., greedy acceptance or random selection). We define Search Operators (SOs) as the composition of a perturbative heuristic with a selector, $h_o = (h_s \circ h_p)$, in accordance with [14]. Consequently, an MH can be defined as

$$\mathrm{MH}_o \triangleq \langle h_g, h_o, h_f \rangle = h_f(h_o) \circ h_g, \tag{1}$$

where h_f is a finalizer that controls or schedules the application of h_o throughout the search process [18]. In particular, h_o is the composition of ϖ search operators, i.e., $h_o = h_\varpi \circ h_{\varpi-1} \circ \cdots \circ h_1$, where each h_k is of the form $h_k = (h_s \circ h_p)$ and $\varpi \in \mathbb{Z}_{++}$ denotes the number of SOs.

B. Automated Algorithm Design and Configuration

Designing optimization algorithms often requires navigating a large space of structural choices and parameter settings, a process that is still commonly performed by manual trial-and-error. AADC has gained relevance as a systematic alternative to this practice [20], [21], particularly in response to the steady proliferation of new MHs that are sometimes introduced without explicit design principles or formal justification [22]. In the traditional workflow, practitioners iteratively assemble algorithmic components and tune their parameters based on experience [5]. While this can render satisfactory results, it typically explores only a small fraction of the configuration space and may produce subcompetitive configurations. An AADC problem can be formulated within the General Combinatorial Optimization Problem (GCOP) framework [16], [23], specialized for the synthesis of heuristic-based optimization algorithms. The goal is to select an MH configuration from a predefined heuristic space that maximizes a performance measure on a target optimization problem. This can be expressed as

$$\mathrm{MH}_* = \underset{\mathrm{MH} \in \mathfrak{H}^\varpi}{\operatorname{argmax}} \left\{ Q(\mathrm{MH} \mid \mathfrak{X}) \right\}, \tag{2}$$

where $\mathfrak{H}$ denotes the heuristic space, ϖ is the MH cardinality, and $Q(\mathrm{MH} \mid \mathfrak{X})$ evaluates the performance of a candidate MH on the optimization problem $(\mathfrak{X}, f)$, with $\mathfrak{X}$ representing the decision space and $f : \mathfrak{X} \to \mathbb{R}$ the objective function. In this work, the GCOP is addressed through Hyper-Heuristics (HHs), which search the heuristic space to automatically construct candidate solution strategies [24], [25]. We adopt a model-free approach in which MHs are built by composing available SOs [26]. Once an MH structure is fixed, a model-based strategy can then tune its hyperparameters to maximize performance [13].

C. Photovoltaic Models

This section summarizes the two standard equivalent-circuit PV models considered in this work: the Single-Diode (SD) and Double-Diode (DD) representations [9], [10]. Let V_t [V] and I_t [A] denote the terminal voltage and current, respectively. In both models, define the voltage across the parallel network formed by the diode branch and the shunt resistance as

$$V_d = V_t + R_s I_t, \tag{3}$$

where R_s [Ω] is the series resistance.

To model the steady-state I-V characteristic of the PV cell, either the SD or the DD equivalent-circuit model can be employed. Using (3), the corresponding implicit I-V relationships are stated next, starting with SD and then its DD extension.

1) Single-Diode Model: The SD model uses a single diode, which is given by

$$I_t = I_{ph} - I_{sd}\left[\exp\left(\frac{q\,V_d}{nkT}\right) - 1\right] - \frac{V_d}{R_{sh}}, \tag{4}$$

where I_{ph} [A] is the photo-generated current, I_{sd} [A] is the diode saturation current, n is the (effective) diode ideality factor, and R_{sh} [Ω] is the shunt resistance. In the Shockley diode equation, $q = 1.602 \times 10^{-19}$ C is the elementary charge, $k = 1.380 \times 10^{-23}$ J/K is the Boltzmann constant, and T [K] is the cell temperature [9], [10].

2) Double-Diode Model: The DD model extends the SD representation by using two diodes, and is written as

$$\begin{aligned} I_t = I_{ph} &- I_{sd1}\left[\exp\left(\frac{q\,V_d}{n_1 kT}\right) - 1\right] \\ &- I_{sd2}\left[\exp\left(\frac{q\,V_d}{n_2 kT}\right) - 1\right] - \frac{V_d}{R_{sh}}, \end{aligned} \tag{5}$$

where I_{sd1} and I_{sd2} are the saturation currents of the first and second diode. The ideality factors n_1 and n_2 correspond to the diffusion and recombination diodes, respectively [9], [10].

III. Proposed Approach

The high-level problem defined by GCOP in (2) is addressed using the Sampling-based Hyper-Heuristic with Local Search (SBHH-LS), which explores alternative SO configurations for the low-level problem. The method employs a heuristic space $\mathfrak{H}$ that contains the available SOs and samples heuristic sequences to generate candidate MHs, as defined in (1). The performance of each candidate MH is estimated as follows,

$$Q(\mathrm{MH}_o \,|\, \mathfrak{X}) \;=\; \Big(\mathrm{med}(F_h) \;+\; \mathrm{iqr}(F_h)\Big), \qquad (6)$$

where $F_h = \{f(\vec{x}_{r,*}) \mid \forall\, \vec{x}_{r,*} \in X_*\}$, and $f(\vec{x}_{r,*})$ is the objective value at the best solution $\vec{x}_{r,*}$ returned by MH_o in the r^{th} independent run. The set $X_* = \{\vec{x}_{1,*}, \ldots, \vec{x}_{N_r,*}\}$ collects the best solutions obtained over N_r runs, while med and iqr denote the median and interquartile range of the values in F_h.

Pseudocode 1 describes SBHH-LS as an iterative construct-and-improve scheme executed for T HH steps. At each step, a candidate sequence of SOs is sampled from the heuristics space ($\mathfrak{H}$) using RandomConstruction. In particular, the sequence length ϖ is drawn uniformly from $\{1, \ldots, \varpi_{\max}\}$, and then ϖ distinct SOs are selected uniformly from $\mathfrak{H}$ without replacement to build $h_o^{(t,0)}$. This candidate is evaluated over N_r independent runs to obtain $Q^{(t,0)}$. Subsequently, a local search generates a neighboring sequence $h_o^{(t,\mathrm{LS})}$ by applying either `Swap` (exchange the positions of two randomly chosen SOs) or `Replace` (substitute one SO with another uniformly sampled from $\mathfrak{H}$ that is not already present in the sequence). The neighbor is evaluated in the same manner to obtain $Q^{(t,\mathrm{LS})}$. The best of the two candidates is selected, and the global best record is updated whenever an improvement is observed. The final output is the best MH found, along with its associated performance measurement.

Pseudocode 1 Sampling-based Hyper-heuristic with Local Search (SBHH-LS).

Input: Domain $\mathfrak{X}$ and objective $f(\vec{x})$; heuristic space $\mathfrak{H}$; initializer h_i; finalizer h_f; population size N; maximum cardinality $\varpi_{\max}$; runs N_r; HH iterations T.
Output: $\mathrm{MH}_o^{\text{best}}$ and $Q(\mathrm{MH}_o^{\text{best}} \mid \mathfrak{X}, f)$.

$\mathrm{MH}_o^{\text{best}} \leftarrow \emptyset$, $Q^{\text{best}} \leftarrow \infty$
for $t = 1, \ldots, T$ **do**
  $h_o^{(t,0)} \sim \textsc{RandomConstruction}(\mathfrak{H}, \varpi_{\max})$
  $Q^{(t,0)} \leftarrow \textsc{EvalPerformance}(h_o^{(t,0)})$
  $h_o^{(t,\mathrm{LS})} \leftarrow \textsc{LocalSearch}(h_o^{(t,0)})$
  $Q^{(t,\mathrm{LS})} \leftarrow \textsc{EvalPerformance}(h_o^{(t,\mathrm{LS})})$
  $k_* \leftarrow \mathrm{argmin}_{k \in \{0,\mathrm{LS}\}}\{Q^{(t,k)}\}$
  $h_o^{(t,\mathrm{sel})} \leftarrow h_o^{(t,k_*)}$, $Q^{(t,\mathrm{sel})} \leftarrow Q^{(t,k_*)}$
  if $Q^{(t,\mathrm{sel})} < Q^{\text{best}}$ **then**
    $\mathrm{MH}_o^{\text{best}} \leftarrow \langle h_i, h_o^{(t,\mathrm{sel})}, h_f \rangle$, $Q^{\text{best}} \leftarrow Q^{(t,\mathrm{sel})}$
return $\mathrm{MH}_o^{\text{best}}$, Q^{best}

function $\textsc{EvalPerformance}(h_o)$
  $X_* \leftarrow \emptyset$
  for $r = 1$ to N_r **do**
    $\vec{x}_{r,*} \leftarrow \textsc{EvalMH}(h_o)$, $X_* \leftarrow X_* \cup \{\vec{x}_{r,*}\}$
  return $\mathrm{perf}(X_*)$ ▷ *Determined using* (6)

function $\textsc{EvalMH}(h_o)$
  $X \leftarrow \{h_i\{\mathfrak{X}\} \mid n = 1, \ldots, N\}$, $F \leftarrow \{f(\vec{x}) \mid \vec{x} \in X\}$
  $\vec{x}_* \leftarrow \arg\min_{\vec{x} \in X} F$
  while $h_f\{\vec{x}_*\}$ **do**
    for $o = 1$ to ϖ **do**
      $(h_p, h_s) \leftarrow h_o[o]$
      $X, F \leftarrow h_p\{X\}$, $\vec{x}_* \leftarrow h_s\{X\}$
  return $\vec{x}_*$

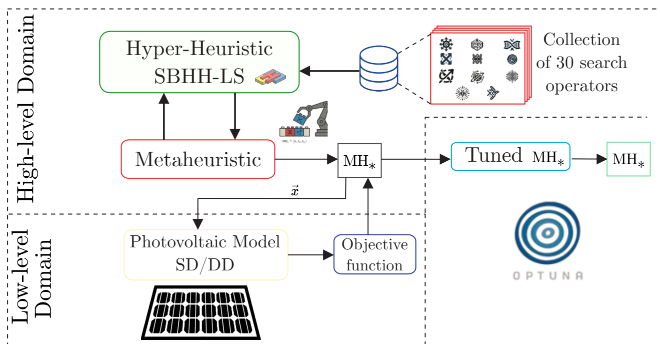


Fig. 1: Overview of the proposed HH-based methodology for PV parameter estimation: MH design via SBHH-LS and subsequent hyperparameter tuning via Optuna.

IV. Methodology

This section reports the methodological setup adopted in this study. The procedure follows a two-stage pipeline in which the MH structure is first designed at a high level using HH and then refined through hyperparameter tuning. The low-level component solves the PV parameter-estimation problem for the SD and DD models under a fixed evaluation budget. The experimental setup and evaluation protocol, including the number of replicas, trials, and logging strategy, are detailed in Subsection IV-A. Fig. 1 provides an overview of the resulting high-level and low-level interactions.

A. Experimental Setup

All experiments were run in Python 3.9 on an ASUS TUF Gaming F17 (AMD Ryzen™ 7 5700G, 8 cores; 16 GB RAM; Windows 11 64-bit). For each PV model (SD and DD), SBHH-LS was executed in 30 independent HH replicas. In each replica, MH candidates were evaluated over T HH steps using Q as defined in (6), and the best and worst MH candidates were retained. For every evaluated MH, we logged its SO sequence and the corresponding Q value, including the samples used to compute Q. The final structure for each model corresponded to the best MH_* across the 30 replicas (i.e., the lowest Q); the worst MHs were preserved for comparison. Given the selected structure, we tuned its operator hyperparameters with Optuna using 50 trials [15]. This tuning stage was repeated 30 times independently. In each tuning replica, every trial was scored with Q by running the instantiated MH 30 times under the same low-level budget (cf. Section IV-B). We stored the tested hyperparameters and the resulting Q, including the samples used to compute it. Finally, the best-tuned MH was applied to the SD and DD parameter-estimation tasks to report the fitted solutions.

B. Low-Level Domain

The low-level optimization task is PV parameter estimation from measured I-V data. We use the widely adopted benchmark dataset of a commercial silicon solar cell from the R.T.C. Company of France (57 mm diameter), measured under 1 sun (i.e., 1000 W/m^2) at 33°C ($T = 306.15$ K). The dataset contains 26 measured I-V points, with $V \in [-0.2057, 0.5900]$ V and $I \in [-0.2100, 0.7640]$ A [10]. For the SD model, the design vector is $\vec{x} = (R_s, R_{sh}, I_{ph}, I_{sd}, n)^\intercal \in \mathfrak{X} \subseteq \mathbb{R}^5$ with $R_s \in [0, 0.5]$ Ω, $R_{sh} \in [0, 100]$ Ω, $I_{ph} \in [0, 1]$ A, $I_{sd} \in [0, 1]$ μA, and $n \in [1, 2]$. For the DD model, $\vec{x} = (R_s, R_{sh}, I_{ph}, I_{sd1}, I_{sd2}, n_1, n_2)^\intercal \in \mathbb{R}^7$ with the same bounds and $n_1, n_2 \in [1, 2]$. Hence, the parameters are estimated by minimizing the objective function F_{obj}, defined as the Root Mean Square Error (RMSE) over the K measured samples,

$$F_{\text{obj}}(\vec{x}) = \text{RMSE}_{I\text{–}V}(\vec{x}) = \sqrt{\tfrac{1}{K} \textstyle\sum_{k=1}^{K} r_k(\vec{x})^2}, \quad (7)$$

where $r_k(\vec{x})$ is the residual at (V_k, I_k) obtained by evaluating the SD or DD model I-V equation (cf. (4) and (5)). For reporting purposes only, fitted I-V and P-V curves are reconstructed by solving the implicit I-V equation for $\widehat{I}(V_k; \vec{x})$ via Newton iterations with continuation in voltage, where the converged solution at V_{k-1}, namely $\widehat{I}(V_{k-1}; \vec{x})$, initializes the solver at V_k to compute $\widehat{I}(V_k; \vec{x})$ [27].

In this setting, the achieved MH candidate at the high level is the component that directly interacts with the PV domain, searching over $\mathfrak{X}$ with a population size of 30 for 100 iterations.

C. High-Level Domain

At the high level, the proposed HH controls the generation of candidate MH structures by operating on sequences of SOs defined over the collection, as shown in TABLE I. TABLE I lists the SOs collection used by SBHH-LS, reporting only the variation parameters and selector for each heuristic instantiation, while operator definitions and baseline hyperparameters follow [14]. SBHH-LS (cf. Pseudocode 1) is executed with $T = 10$ HH steps and maximum cardinality $\varpi_{\max} = 2$. At each HH step, the local-search neighbor is generated by applying either `Swap` or `Replace` (selected uniformly at random), which are HH-level structure-edit operations used to propose alternative MH candidates while preserving feasibility without repeating SOs. Each candidate MH is scored using Q in (6) by running it $N_r = 30$ times with different random seeds, with the low-level budget fixed as described in Section IV-B.

V. Experimental Results

TABLE II summarizes the MH sequences obtained by HH over 30 independent replicas for SD and DD, reporting in each replica the best configuration, lowest Q in (6), and the worst configuration observed in the same run, including their performance values. For DD, the best results concentrate on only three distinct sequences: $[h_{27}, h_3]$ appears in 23/30 replicas (and $[h_3, h_{27}]$ in 6/30), matching the compact best-performance distribution in Fig. 2a (median 1.98×10^{-3}, IQR 0.15×10^{-3}). In contrast, SD exhibits a broader set of best configurations and a wider distribution in Fig. 2a (median 2.67×10^{-3}, IQR 0.82×10^{-3}), with a larger upper range. The globally best MH is the same for both models, $[h_{27}, h_3]$, achieving 1.807×10^{-3} in SD (replica 16) and 1.68×10^{-3} in DD (replica 30). Regarding unfavorable outcomes, Fig. 2b shows similar medians for SD and DD (279.44×10^{-3} and 274.89×10^{-3}, respectively), but higher dispersion for DD (IQR 186.68×10^{-3} vs. 120.73×10^{-3}). In SD, the most frequent worst sequences are $[h_{14}, h_0]$ and $[h_0, h_{14}]$ (3/30 each), and the overall worst case is $[h_1, h_0]$ with 357.54×10^{-3} (replica 25). In DD, $[h_1, h_{14}]$ is the most frequent worst sequence (3/30), and the overall worst case is $[h_0, h_1]$ with 408.65×10^{-3} (replica 26).

For SD, the operator usage patterns in the first position (SO$_1$) show a clear preference for swarm-based search when HH selects the best-performing MHs. In particular, the most frequent SO$_1$ choice among best MHs is h_{26} (`swarm_dynamic`, selector `all`), appearing in 6/30 replicas, followed by h_{27} (`swarm_dynamic`, selector `greedy`) in 5/30 replicas; `differential_mutation` (h_3) also appears recurrently as SO$_1$ (4/30 replicas). In contrast, the worst-selected MHs are more often initiated

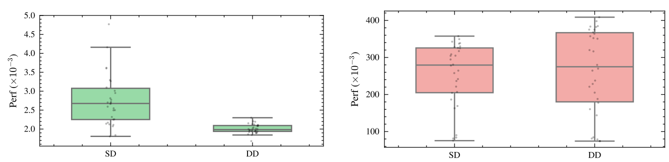
TABLE I: Search Operators (SOs) collection used during the HH process to compose the tailored MH$_*$, reporting their perturbation heuristic, associated variation parameters, and selection heuristic.

SOs	Perturbation Heuristic	Variation Parameters	Selection Heuristic
h_0	`central_force_dynamic`	–	all
h_1	`central_force_dynamic`	–	greedy
h_2	`differential_mutation`	MS^{b1}	all
h_3	`differential_mutation`	MS^{b1}	greedy
h_4	`differential_mutation`	MS^{b2}	all
h_5	`differential_mutation`	MS^{b2}	greedy
h_6	`firefly_dynamic`	pdf^{a2}	all
h_7	`firefly_dynamic`	pdf^{a2}	greedy
h_8	`genetic_crossover`	PS^{c1}, CM^{d3}	all
h_9	`genetic_crossover`	PS^{c1}, CM^{d3}	greedy
h_{10}	`genetic_crossover`	PS^{c3}, CM^{d3}	all
h_{11}	`genetic_crossover`	PS^{c3}, CM^{d3}	greedy
h_{12}	`genetic_mutation`	pdf^{a2}	all
h_{13}	`genetic_mutation`	pdf^{a2}	greedy
h_{14}	`gravitational_search`	–	all
h_{15}	`gravitational_search`	–	greedy
h_{16}	`local_random_walk`	pdf^{a1}	all
h_{17}	`local_random_walk`	pdf^{a1}	greedy
h_{18}	`random_sample`	–	all
h_{19}	`random_sample`	–	greedy
h_{20}	`random_search`	pdf^{a2}	greedy
h_{21}	`random_search`	pdf^{a2}	all
h_{22}	`spiral_dynamic`	–	all
h_{23}	`spiral_dynamic`	–	greedy
h_{24}	`swarm_dynamic`	pdf^{a2}, VA^{e1}	all
h_{25}	`swarm_dynamic`	pdf^{a2}, VA^{e1}	greedy
h_{26}	`swarm_dynamic`	pdf^{a2}, VA^{e2}	all
h_{27}	`swarm_dynamic`	pdf^{a2}, VA^{e2}	greedy
h_{28}	`random_flight`	pdf^{a3}	all
h_{29}	`random_flight`	pdf^{a3}	greedy

VA: `velocity_approach`, MS: `mutation_scheme`,
PS: `pairing_scheme`, CM: `crossover_mechanism`,
pdf: `probability density function`.
Options: VA $\in$ {^{e1}Inertial, ^{e2}Constriction};
MS $\in$ {^{b1}current-to-best, ^{b2}rand-to-best-and-current};
PS $\in$ {^{c1}Cost, ^{c3}Tournament}; CM $\in$ {^{d3}Uniform};
pdf $\in$ {^{a1}Gaussian, ^{a2}Uniform, ^{a3}Lévy}.

TABLE II: Best and worst MH sequences obtained per HH replica across 30 runs, comparing the Single-Diode (SD) and Double-Diode (DD) PV models.

	SD				DD			
Rep	Best Seq	Perf ($\times 10^{-3}$)	Worst Seq	Perf ($\times 10^{-3}$)	Best Seq	Perf ($\times 10^{-3}$)	Worst Seq	Perf ($\times 10^{-3}$)
1	$[h_{25}, h_{26}]$	2.26	$[h_8, h_1]$	204.30	$[h_3, h_{27}]$	2.08	$[h_8, h_{14}]$	279.91
2	$[h_{15}, h_0]$	4.77	$[h_{14}, h_0]$	317.17	$[h_{27}, h_3]$	2.00	$[h_{11}, h_1]$	160.44
3	$[h_{27}, h_1]$	2.59	$[h_{16}, h_{14}]$	327.82	$[h_{27}, h_3]$	1.95	$[h_1, h_0]$	374.90
4	$[h_7, h_{22}]$	2.72	$[h_{21}, h_0]$	241.62	$[h_{27}, h_3]$	1.89	$[h_{14}, h_{17}]$	316.43
5	$[h_3, h_{17}]$	3.29	$[h_8, h_9]$	280.64	$[h_{27}, h_3]$	1.97	$[h_1, h_{14}]$	382.20
6	$[h_{15}, h_{27}]$	2.49	$[h_0, h_{14}]$	324.88	$[h_{27}, h_3]$	1.93	$[h_{21}, h_{18}]$	81.19
7	$[h_0, h_{26}]$	3.60	$[h_0, h_8]$	163.00	$[h_3, h_{27}]$	2.00	$[h_9, h_0]$	319.87
8	$[h_{17}, h_{13}]$	11.6	$[h_{17}, h_{11}]$	221.27	$[h_{27}, h_3]$	1.95	$[h_1, h_{10}]$	205.05
9	$[h_7, h_{22}]$	4.16	$[h_2, h_1]$	90.02	$[h_{26}, h_3]$	2.13	$[h_4, h_7]$	74.44
10	$[h_{26}, h_{23}]$	2.68	$[h_{10}, h_8]$	278.24	$[h_{27}, h_3]$	2.12	$[h_{14}, h_1]$	384.7
11	$[h_1, h_{26}]$	2.55	$[h_{11}, h_{14}]$	236.44	$[h_{27}, h_3]$	1.90	$[h_1, h_{19}]$	78.79
12	$[h_5, h_{27}]$	2.31	$[h_0, h_{14}]$	338.67	$[h_{27}, h_3]$	2.04	$[h_2, h_{12}]$	242.64
13	$[h_{26}, h_1]$	2.12	$[h_{21}, h_0]$	168.98	$[h_{27}, h_3]$	1.86	$[h_0, h_{21}]$	220.95
14	$[h_{25}, h_{26}]$	2.14	$[h_{14}, h_{17}]$	325.93	$[h_3, h_{27}]$	2.11	$[h_{14}, h_8]$	375.12
15	$[h_{27}, h_3]$	1.84	$[h_0, h_1]$	308.28	$[h_3, h_{27}]$	2.02	$[h_{14}, h_7]$	177.55
16	$[h_{27}, h_3]$	1.80	$[h_1, h_0]$	349.43	$[h_{27}, h_3]$	1.96	$[h_{16}, h_0]$	143.87
17	$[h_{16}, h_{26}]$	2.49	$[h_{14}, h_0]$	304.46	$[h_{27}, h_3]$	2.01	$[h_{16}, h_{14}]$	352.13
18	$[h_{26}, h_{25}]$	2.25	$[h_{10}, h_0]$	206.84	$[h_{27}, h_3]$	1.84	$[h_{17}, h_{14}]$	264.48
19	$[h_{22}, h_{13}]$	3.27	$[h_{14}, h_8]$	294.55	$[h_{27}, h_3]$	1.94	$[h_0, h_8]$	228.52
20	$[h_3, h_0]$	3.62	$[h_{15}, h_{14}]$	329.99	$[h_{27}, h_3]$	1.93	$[h_{14}, h_{12}]$	367.85
21	$[h_3, h_{15}]$	2.11	$[h_{18}, h_9]$	81.33	$[h_{27}, h_3]$	1.96	$[h_1, h_{14}]$	383.39
22	$[h_{15}, h_{22}]$	2.70	$[h_5, h_4]$	75.45	$[h_{27}, h_3]$	1.89	$[h_8, h_9]$	269.87
23	$[h_{27}, h_5]$	2.17	$[h_1, h_{14}]$	304.81	$[h_{27}, h_3]$	1.92	$[h_1, h_{14}]$	383.39
24	$[h_3, h_{21}]$	3.09	$[h_0, h_{14}]$	337.25	$[h_3, h_{27}]$	2.29	$[h_{14}, h_8]$	363.81
25	$[h_{25}, h_{15}]$	3.00	$[h_1, h_0]$	357.54	$[h_{27}, h_3]$	2.09	$[h_8, h_0]$	187.96
26	$[h_{25}, h_{15}]$	2.81	$[h_{15}, h_1]$	186.45	$[h_3, h_{27}]$	2.15	$[h_0, h_1]$	408.65
27	$[h_{27}, h_{22}]$	2.66	$[h_4, h_{13}]$	83.88	$[h_{27}, h_3]$	2.07	$[h_{14}, h_{12}]$	357.27
28	$[h_{26}, h_{25}]$	2.07	$[h_{14}, h_0]$	308.52	$[h_{27}, h_3]$	2.20	$[h_3, h_{10}]$	84.45
29	$[h_{26}, h_{22}]$	2.77	$[h_0, h_{21}]$	257.78	$[h_{27}, h_3]$	2.19	$[h_0, h_{14}]$	397.94
30	$[h_{26}, h_{19}]$	2.95	$[h_{16}, h_{14}]$	264.45	$[h_{27}, h_3]$	1.68	$[h_1, h_9]$	237.60



(a) Best-selected MHs

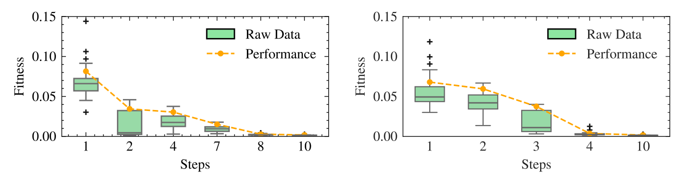
(b) Worst-selected MHs

Fig. 2: Performance distributions of the best and worst selected MHs over 30 independent HH runs for the Single-Diode (SD) and Double-Diode (DD) PV models.

by h_{14} (gravitational_search, selector all), which is the most frequent SO_1 choice in the worst cases (5/30 replicas), with h_1 (central_force_dynamic, selector greedy) following at 3/30 replicas. A similar trend is observed for SO_2 in the worst-selected MHs, where h_{14} dominates (8/30 replicas), while h_1 is the next most frequent (4/30 replicas). Overall, these counts indicate that, for SD calibration, the best HH-selected MHs most often start with swarm_dynamic in SO_1, while gravitational_search appears disproportionately in the worst configurations, especially when placed in SO_1 and/or repeated in SO_2 within the same HH run.

For DD, the operator usage patterns are even more concentrated, indicating a highly stable structure among the best MHs returned by HH. The dominant SO_1 choice is h_{27} (swarm_dynamic, selector greedy), which appears in 23/30 replicas, while the dominant SO_2 choice is h_3 (differential_mutation), selected in 24/30 replicas. This strong recurrence suggests that, under the DD calibration landscape and the considered evaluation budget, HH consistently favors a two-step strategy in which an initial swarm-driven search provides rapid global progress and maintains a competitive population spread, followed by a differential update that introduces directed, population-informed perturbations to refine candidate solutions. In terms of practical guidance, it is also noteworthy that the globally best configuration identified for SD coincides with the dominant DD best sequence, namely $[h_{27}, h_3]$, reinforcing this pairing as a robust default across both PV models in the present setup. In contrast, the worst-selected MHs in DD are more frequently associated with h_{14} (gravitational_search, selector all), which is the most common operator in both SO_1 and SO_2 among poor configurations (7/30 replicas in each position). As in the SD case, this pattern is consistent with attraction-driven dynamics that reduce diversity too early and increase the likelihood of stagnation.

Fig. 3b shows the HH process in the best-selected replica for SD (replica 16) and DD (replica 30). Each HH step evaluates a candidate MH across 30 independent runs; boxplots summarize the resulting fitness distributions, and the overlaid curve shows the incumbent's best at each step. Early steps exhibit wider dispersion, indicating higher run-to-run variability, whereas later steps show reduced spread and a steadily improving incumbent best, consistent with iterative refinement. For both SD (Fig. 3a) and DD (Fig. 3b), the lowest fitness is reached at step 10, which corresponds to the selected best MH. Each boxplot shows the distribution of final fitness values obtained from 30 independent executions of the candidate MH at that HH step.



(a) Single-Diode (SD)

(b) Double-Diode (DD)

Fig. 3: Fitness evolution along the HH procedure for the best-achieved MH (selected among 30 HH replicas) in the SD and DD models.

The tailored MH (MH_*) selected from the HH process is the best-performing configuration among the 30 replicas; in this study, the same sequence is obtained for both SD and DD, namely $[h_{27}, h_3]$, whose variation parameters and hyperparameters are reported in TABLE III. Building on this structure, Fig. 4 compares the performance distributions of the best MHs produced during HH with those achieved after tuning MH_* (i.e., preserving $[h_{27}, h_3]$ fixed and adjusting its hyperparameters). For SD, tuning shifts the distribution downward, reducing the median from 2.6755×10^{-3} to 1.2214×10^{-3} and the IQR from 0.8245×10^{-3} to 0.2486×10^{-3}. For DD, the median decreases from 1.9856×10^{-3} to 1.3196×10^{-3}, with tightly clustered tuned runs (IQR 0.2251×10^{-3}) and best tuned outcomes reaching 1.0185×10^{-3}.

Across the 30 independent tuning replicas, the hyperparameter adjustment preserves the dominant structural choices identified by HH, most notably pdf = Uniform for

TABLE III: Variation parameters and hyperparameters of each SO composing MH$_*$ obtained during the HH process.

SOs	Simple Heuristic	Variation Parameters	Hyperparameters
h_{27}	Swarm Dynamic	VA = Constriction `pdf` = Uniform	$\chi = 1.0$, $\phi_1 = 2.54$, $\phi_2 = 2.56$
h_3	Differential Mutation	MS = current-to-best	$M = 1$, $a_1 = 1.0$

VA: `velocity_approach`, MS: `mutation_scheme`, `pdf`: probability density function.
Ranges: $\phi_1, \phi_2 \in]0, 4]$, $\chi \in]0, 1]$, $M \in \{1, 2, 3\}$, $a_m \in]0, 3]$.
Options: VA $\in$ {Inertial, Constriction}, MS $\in$ {rand, best, current, current-to-best, rand-to-best, rand-to-best-and-current}, pdf $\in$ {Uniform, Gaussian, Levy}.

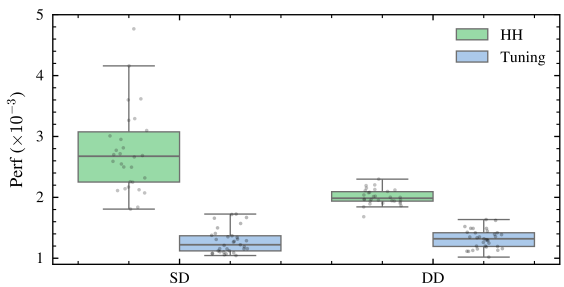


Fig. 4: Performance distributions for SD and DD comparing the best MHs obtained during the HH process against the tuned MH$_*$ (fixed structure $[h_{27}, h_3]$ with adjusted hyperparameters), over 30 HH replicas per model.

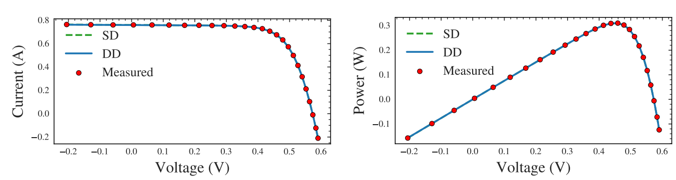
h_{27} (selected in 29/30 SD replicas and 30/30 DD replicas), while refining the mutation scheme of h_3 and the numerical coefficients of both operators. In each tuning replica, Optuna runs 50 trials and returns the best hyperparameter configuration (i.e., the configuration with the lowest Q) for that replica; the frequencies reported here are computed across the 30 best-per-replica configurations. For SD, the most frequent selections favor MS = `rand-to-best` (9/30 replicas) and $M = 2$ (12/30 replicas), whereas the best tuned SD configuration attains 1.0443×10^{-3} with MS = `rand-to-best-and-current` and $M = 3$. For DD, MS = `rand-to-best` (14/30 replicas) and $M = 1$ (17/30 replicas) are the most recurrent choices, and the best-tuned DD configuration reaches 1.0185×10^{-3} with MS = `best` and $M = 1$. Overall, once MH$_*$ is fixed, hyperparameter tuning provides an additional gain beyond structural selection, lowering the central tendency and, particularly for SD, reducing run-to-run variability; the best-tuned settings obtained for each PV model are summarized in TABLE IV.

Finally, the MH$_*$ tuned for PV parameter identification produced best-fit SD and DD solutions that closely match the measured operating points and yield physically plausible parameter magnitudes, as shown by the I-V and P-V overlays in Fig. 5 (top and bottom panels, respectively). For the SD model, the identified parameters were $R_s = 3.63 \times 10^{-2}\,\Omega$,

TABLE IV: Best tuned hyperparameters for the selected MH$_*$ structure $[h_{27}, h_3]$ in SD and DD.

Model	SO	Variation Parameters	Hyperparameters
SD	h_{27}	VA = Constriction `pdf` = Uniform	$\chi = 0.218741$, $\phi_1 = 3.993975$, $\phi_2 = 0.023545$
	h_3	MS = rand-to-best-and-current	$M = 3$, $a_1 = 0.101992$
DD	h_{27}	VA = Constriction `pdf` = Uniform	$\chi = 0.701305$, $\phi_1 = 3.231502$, $\phi_2 = 2.353836$
	h_3	MS = best	$M = 1$, $a_1 = 0.767774$

$R_{sh} = 57.1\,\Omega$, $I_{ph} = 7.61 \times 10^{-1}$ A, $I_s = 3.34 \times 10^{-7}$ A, and $n = 1.484$, achieving a global I-V curve error of $\text{RMSE}_{I\text{-}V} = 1.05 \times 10^{-3}$ A and reproducing the maximum-power condition with negligible deviation ($P_{\max} = 0.30995$ W vs. 0.31006 W measured, relative error $\approx 3.5 \times 10^{-2}\%$), which is also reflected by the near-coincident P-V peak in Fig. 5b. The corresponding Fill Factor (FF), defined as $FF = P_{\max}/(V_{oc} I_{sc})$, remained essentially unchanged (measured 0.7119, SD 0.7120). The DD solution, with $R_s = 3.57 \times 10^{-2}\,\Omega$, $R_{sh} = 70.7\,\Omega$, $I_{ph} = 7.60 \times 10^{-1}$ A, $I_{s1} = 8.59 \times 10^{-7}$ A, $I_{s2} = 2.71 \times 10^{-7}$ A, $n_1 = 2.0$, and $n_2 = 1.470$, preserved the same operating-point indicators ($P_{\max} = 0.30977$ W, $FF = 0.7122$) while yielding a lower curve error of $\text{RMSE}_{I\text{-}V} = 7.93 \times 10^{-4}$ A, consistent with the tighter agreement with the measured points in Fig. 5a. Therefore, both fits imply low series losses ($R_s \approx 0.036\,\Omega$) and limited shunt leakage (tens of Ω for R_{sh}) over the measured range. Notwithstanding, the DD fit saturates at n_1's upper bound, suggesting that some of the added flexibility is absorbed by parameter limits rather than consistently improving the fit.



(a) I-V fitting (b) P-V curves

Fig. 5: Best solutions obtained by running the tuned MH on the PV parameter extraction problem. SD and DD model fits are compared with I-V and P-V measurements.

VI. Conclusions

This work presented an Automated Algorithm Design and Configuration (AADC) approach for Photovoltaic (PV) parameter estimation based on a Sampling-Based Hyper-Heuristic with Local Search (SBHH-LS). SBHH-LS constructs Metaheuristics (MHs) by selecting and ordering Search Operators (SOs) from a predefined heuristic space, and refines the selected MH through Optuna-based hyperparameter tuning. Overall, the proposed pipeline yields compact, interpretable MH designs and improves them via systematic parameter refinement. For the considered PV calibration task, the Double-Diode (DD) model exhibited a more stable HH outcome than the Single-Diode (SD) model. In DD, the best MHs concentrate on only three sequences, dominated by $[h_{27}, h_3]$ (`swarm_dynamic`, `differential_mutation`) in 23/30 replicas, with a median best performance of 1.98×10^{-3} and an IQR of 0.15×10^{-3}. In contrast, SD renders a broader set of optimal configurations and greater dispersion (median 2.67×10^{-3}, IQR 0.82×10^{-3}). Despite this difference, both models share the same globally best HH structure, $[h_{27}, h_3]$, achieving 1.80×10^{-3} in SD and 1.68×10^{-3} in DD. Once $\mathrm{MH}_* = [h_{27}, h_3]$ is fixed, tuning provides an additional gain beyond structural selection, reducing the SD median from 2.6755×10^{-3} to 1.2214×10^{-3} and the DD median from 1.9856×10^{-3} to 1.3196×10^{-3}. DD also exhibits lower curve error than SD, although the additional degrees of freedom do not materially improve operating-point indicators. The study's results highlight the strong performance of swarm-based operators, particularly when coupled with differential-mutation-based updates, resulting in a robust two-step design that consistently delivers competitive solutions for PV parameter extraction. In contrast, gravitational-search-based operators (e.g., h_{14}) are repeatedly associated with the worst results, which indicates premature loss of diversity and stagnation in this calibration setting. These conclusions depend on the adopted computational budget and may vary across different evaluation settings.

To extend and strengthen this framework, three directions merit immediate pursuit. First, a landscape-aware configuration, derived from exploratory landscape analysis, would facilitate instance-specific MH adaptation without substantial methodological extension. Second, transfer learning across diode model variants offers a pragmatic approach to reducing computational overhead while validating the generalizability of operator sequences. Third, dynamic hyperparameter adjustment during the search could enhance robustness and adaptability by leveraging the robust two-stage pipeline established in this work.

References

[1] K. Hussain, M. N. M. Salleh, S. Cheng, and Y. Shi, "Metaheuristic research: a comprehensive survey," *Artif. Intell. Rev.*, vol. 52, no. 4, pp. 2191–2233, 2019.

[2] A. E. Ezugwu, A. K. Shukla, R. Nath, A. A. Akinyelu, J. O. Agushaka, H. Chiroma, and P. K. Muhuri, "Metaheuristics: a comprehensive overview and classification along with bibliometric analysis," *Artif. Intell. Rev.*, vol. 54, no. 6, pp. 4237–4316, aug 2021.

[3] T. Stützle and M. López-Ibáñez, "Automated design of metaheuristic algorithms," in *Handbook of metaheuristics*. Springer, 2019, pp. 541–579.

[4] C. Huang, Y. Li, and X. Yao, "A Survey of Automatic Parameter Tuning Methods for Metaheuristics," *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 201–216, 2020.

[5] C. L. Camacho-Villalón, T. Stützle, and M. Dorigo, "Designing new metaheuristics: manual versus automatic approaches," *Intelligent Computing*, vol. 2, p. 0048, 2023.

[6] J. Carrasco, S. García, M. Rueda, S. Das, and F. Herrera, "Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review," *Swarm Evol. Comput.*, vol. 54, no. April 2019, p. 100665, may 2020.

[7] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 Special Session on Real Parameter Optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.

[8] D. F. Zambrano-Gutierrez, J. M. Cruz-Duarte, J. G. Avina-Cervantes, J. C. Ortiz-Bayliss, J. J. Yanez-Borjas, and I. Amaya, "Automatic design of metaheuristics for practical engineering applications," *IEEE Access*, vol. 11, pp. 7262–7276, 2023.

[9] A. R. Jordehi, "Parameter estimation of solar photovoltaic (pv) cells: A review," *Renewable and Sustainable Energy Reviews*, vol. 61, pp. 354–371, 2016.

[10] D. Oliva, M. Abd El Aziz, and A. E. Hassanien, "Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm," *Applied energy*, vol. 200, pp. 141–154, 2017.

[11] D. S. Pillai and N. Rajasekar, "Metaheuristic algorithms for pv parameter identification: A comprehensive review with an application to threshold setting for fault detection in pv systems," *Renewable and Sustainable Energy Reviews*, vol. 82, pp. 3503–3525, 2018.

[12] M. N. Akhter, S. Mekhilef, H. Mokhlis, and N. Mohamed Shah, "Review on forecasting of photovoltaic power generation based on machine learning and metaheuristic techniques," *IET Renewable Power Generation*, vol. 13, no. 7, pp. 1009–1023, 2019.

[13] M. López-Ibáñez, J. Dubois-Lacoste, L. P. Cáceres, M. Birattari, and T. Stützle, "The irace package: Iterated racing for automatic algorithm configuration," *Operations Research Perspectives*, vol. 3, pp. 43–58, 2016.

[14] J. M. Cruz-Duarte, I. Amaya, J. C. Ortiz-Bayliss, S. E. Conant-Pablos, H. Terashima-Marín, and Y. Shi, "Hyper-heuristics to customise metaheuristics for continuous optimisation," *Swarm and Evolutionary Computation*, vol. 66, p. 100935, 2021.

[15] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2623–2631.

[16] J. Cruz-Duarte, J. Ortiz-Bayliss, I. Amaya, and N. Pillay, "Global optimisation through hyper-heuristics: Unfolding population-based metaheuristics," *Applied Sciences*, vol. 11, no. 12, p. 5620, jun 2021.

[17] A. Alorf, "A survey of recently developed metaheuristics and their comparative analysis," *Engineering Applications of Artificial Intelligence*, vol. 117, p. 105622, 2023.

[18] J. Cruz-Duarte, I. Amaya, J. Ortiz-Bayliss, S. Conant-Pablos, H. Terashima-Marín, and Y. Shi, "Hyper-heuristics to customise metaheuristics for continuous optimisation," *Swarm Evol. Comput.*, vol. 66, p. 100935, 2021.

[19] D. F. Zambrano-Gutierrez, J. M. Cruz-Duarte, J. G. Avina-Cervantes, J. C. Ortiz-Bayliss, J. J. Yanez-Borjas, and I. Amaya, "Automatic design of metaheuristics for practical engineering applications," *IEEE Access*, vol. 11, pp. 7262–7276, 2023.

[20] A. J. Nebro, M. López-Ibáñez, J. García-Nieto, and C. A. Coello Coello, "On the automatic design of multi-objective particle swarm optimizers: experimentation and analysis," *Swarm Intelligence*, pp. 1–35, 2023.

[21] P. Alfaro-Fernández, R. Ruiz, F. Pagnozzi, and T. Stützle, "Automatic algorithm design for hybrid flowshop scheduling problems," *European Journal of Operational Research*, vol. 282, no. 3, pp. 835–845, 2020.

[22] K. Sörensen, "Metaheuristics—the metaphor exposed," *International Transactions in Operational Research*, vol. 22, no. 1, pp. 3–18, 2015.

[23] R. Qu, G. Kendall, and N. Pillay, "The general combinatorial optimisation problem: Towards automated algorithm design," *IEEE Comput. Intell. Mag.*, vol. 15, no. 2, pp. 14–23, 2020.

[24] E. K. Burke, M. R. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward, *A Classification of Hyper-Heuristic Approaches: Revisited*. Springer, 2019, pp. 453–477.

[25] N. Pillay and R. Qu, *Hyper-Heuristics: Theory and Applications*. Springer, 2018.

[26] Q. Zhao, Q. Duan, B. Yan, S. Cheng, and Y. Shi, "Automated design of metaheuristic algorithms: A survey," *arXiv preprint arXiv:2303.06532*, 2023.

[27] T. Easwarakhanthan, J. Bottin, I. Bouhouch, and C. Boutrit, "Nonlinear minimization algorithm for determining the solar cell parameters with microcomputers," *Int. Journal of Solar Energy*, vol. 4, p. 12, 1986.