

PVMSim: A MATLAB app for reproducible double-diode parameter extraction from measured photovoltaic current–voltage curves

Liomnis Osorio^{a,b,c,*}, Mailyn Moreno-Espino^{d,e,**}, Laurent Duchêne^c, Víctor Tuninetti^f, Carlos Zalazar^b, Rodrigo Irrarrázaval^b, Yoalbys Retirado-Mediaceja^{g,h}, Marco Rivera^{i,j}

^a Department of Industrial Processes, Faculty of Engineering, Universidad Católica de Temuco, Temuco, Chile

^b Doctoral Program in Engineering, Macro-Faculty of Engineering (UFRO–UBB–UTALCA Consortium), Temuco, Chile

^c ArGenCo Department, MSM Team, University of Liège, Liège, Belgium

^d Facultad de Informática, Universidad Complutense de Madrid, Madrid, Spain

^e Instituto de Tecnología del Conocimiento, Universidad Complutense de Madrid, Madrid, Spain

^f Department of Mechanical Engineering, Universidad de La Frontera, Temuco, Chile

^g Centro de Estudio de Energía y Tecnología Avanzada de Moa, Facultad de Metalurgia y Electromecánica, Universidad de Moa Dr. Antonio Núñez Jiménez, Moa, Cuba

^h Sección de Ciencias Técnicas, Academia de Ciencias de Cuba, La Habana, Cuba

ⁱ Laboratory of Energy Conversion and Power Electronics, Universidad de Talca, Curicó, Chile

^j Power Electronics, Machines and Control Research Institute, Faculty of Engineering, University of Nottingham, Nottingham, United Kingdom

ARTICLE INFO

Keywords:

Double-diode model
Photovoltaic parameter extraction
Measured I–V curves
MATLAB
Reproducible workflow
Hybrid optimization

ABSTRACT

PVMSim is a MATLAB application for traceable parameter extraction of the double-diode photovoltaic model from measured current–voltage (I–V) curves. The software addresses a reproducibility gap in photovoltaic parameter-identification studies, where input provenance, run settings, and diagnostic evidence are often reported inconsistently. It provides graphical and command-line execution modes for staged parameter extraction from measured I–V data using module definitions from a configuration library. Each run exports a structured package with configuration, logs, summaries, result files, and integrity hashes, enabling traceable reruns and controlled comparisons across datasets and computational settings.

Metadata

Nr	Code metadata description	Metadata
C1	Current code version	v0.1.0
C2	Permanent link to code/repository used for this code version	https://github.com/pvmsim/PVMSim
C3	Permanent link to reproducible capsule	https://doi.org/10.5281/zenodo.21605181
C4	Legal code license	MIT License
C5	Code versioning system used	Git
C6	Software code languages, tools and services used	MATLAB R2025b
C7	Compilation requirements, operating environments and dependencies	MATLAB-supported platforms (Windows, macOS, Linux), no additional toolboxes
C8	If available, link to developer documentation/manual	https://doi.org/10.5281/zenodo.19040390
C9	Support email for questions	pvmsim.matlab@gmail.com

* Corresponding author at: Department of Industrial Processes, Faculty of Engineering, Universidad Católica de Temuco, Temuco, Chile.

** Corresponding author at: Facultad de Informática, Universidad Complutense de Madrid, Madrid, Spain.

Email addresses: losorio@uct.cl (L. Osorio), mailyamor@ucm.es (M. Moreno-Espino).

1. Motivation and significance

Physics-based photovoltaic (PV) device models remain essential when the goal extends beyond prediction to include parameter interpretability and model-based diagnostics. Among compact models, the double-diode model (DDM) with series and shunt resistances is widely adopted because it accounts for recombination effects and improves current–voltage curve reproduction compared with simpler equivalents, at the cost of increased model complexity [1]. In practice, the main bottleneck is robust and reproducible parameter extraction from measured current–voltage (I–V) curves, since key model parameters are not directly measurable and are typically inferred by fitting the model to experimental I–V data [2]. The estimation problem is frequently challenging for optimization-based methods, which can be affected by multimodality, local minima, and sensitivity to initialization and user-defined algorithm settings [1–3]. As a result, reported parameter sets and fit metrics can be difficult to reproduce across studies when experimental variability and configuration choices are not fully specified [4].

PVMSim (PV Modeling and Simulation) addresses this gap through an end-to-end workflow that standardizes DDM parameter-extraction experiments. Users load a measured I–V curve and module configuration, execute staged optimization, and assess the fit using standardized metrics and diagnostics. Each run archives a machine-readable configuration containing input paths, metadata, and run options; an execution log; numerical outputs; and SHA-256 checksums. This methodological and infrastructural framework supports independent verification, controlled sensitivity analyses, and transparent comparisons across algorithms, datasets, and computational settings.

Existing PV software ecosystems provide validated forward models and system-level simulation capabilities, including open-source libraries such as pvlb and its extensions [4,5], as well as performance-oriented frameworks such as SAM and PVSyst [6,7]. These tools are indispensable for PV performance modeling, yet they do not typically provide standardized, experiment-oriented workflows for DDM parameter extraction from heterogeneous measured I–V datasets. According to the extant literature, publicly available MATLAB implementations of artifact-complete DDM parameter-extraction workflows remain limited, particularly those that combine configuration snapshots, execution logs, integrity checks, and diagnostic evidence with both interactive and scriptable execution modes. This positioning is consistent with a recent energy-device software publication that emphasizes source availability, documentation, validation, licensing, and reuse [8].

A complementary research line focuses on parameter extraction for diode-based models. Prior work commonly distinguishes between analytical, deterministic, and heuristic strategies for PV parameter identification, with metaheuristic approaches widely adopted for DDM fitting under nonlinear and multimodal objectives [9]. Metaheuristic methods such as particle swarm optimization, differential evolution, genetic algorithms, and simulated annealing have been adopted to improve robustness, at the cost of additional hyperparameters and computational budget [1]. In this context, PVMSim provides a reproducibility-focused platform for auditable and comparable parameter-extraction studies on measured I–V curves.

2. Software description

2.1. DDM and optimization problem

PVMSim implements the DDM with series and shunt resistances (Fig. 1). The terminal current I and voltage V satisfy the Kirchhoff's current law at the circuit terminals, as expressed by Eq. (1).

$$I = I_{ph} - I_{D1} - I_{D2} - I_{sh} \quad (1)$$

where I_{ph} is the photocurrent, I_{D1} and I_{D2} are the diode currents, and I_{sh} is the shunt current. Substituting the Shockley expressions for I_{D1}

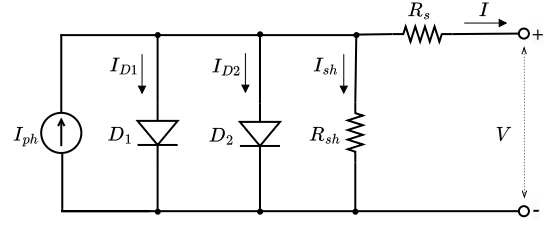


Fig. 1. Equivalent circuit of the DDM implemented in PVMSim.

and I_{D2} and the Ohmic term for I_{sh} yields the implicit DDM relation as Eq. (2).

$$I = I_{ph} - I_{01} \left[\exp \left(\frac{V + IR_s}{n_1 V_t} \right) - 1 \right] - I_{02} \left[\exp \left(\frac{V + IR_s}{n_2 V_t} \right) - 1 \right] - \frac{V + IR_s}{R_{sh}} \quad (2)$$

where I_{01} and I_{02} are the reverse saturation currents, n_1 and n_2 are diode ideality factors, and R_s and R_{sh} are series and shunt resistances. The thermal voltage is determined by Eq. (3).

$$V_t = \frac{N_s k T}{q} \quad (3)$$

with N_s the number of series-connected cells, k the Boltzmann constant (1.3806×10^{-23} J/K), q the elementary charge (1.6021×10^{-19} C), and T the absolute temperature associated with the fitted I–V curve, expressed in kelvin as $T_K = T_{°C} + 273.15$. The measured-curve temperature is used when available; otherwise, the module reference temperature is adopted.

Given measured samples $\{(V_i, I_i^{\text{meas}})\}_{i=1}^N$, PVMSim estimates the parameter vector $\theta = [I_{ph}, I_{01}, I_{02}, R_s, R_{sh}, n_1, n_2]$ by minimizing the root mean square error (RMSE) of an implicit residual evaluated at the measured current by Eqs. (4)–(5).

$$r_i(\theta) \triangleq I_{ph} - I_{01} \left[\exp \left(\frac{V_i + I_i^{\text{meas}} R_s}{n_1 V_t} \right) - 1 \right] - I_{02} \left[\exp \left(\frac{V_i + I_i^{\text{meas}} R_s}{n_2 V_t} \right) - 1 \right] - \frac{V_i + I_i^{\text{meas}} R_s}{R_{sh}} - I_i^{\text{meas}} \quad (4)$$

$$\text{RMSE}_{\text{obj}}(\theta) = \sqrt{\frac{1}{N} \sum_{i=1}^N r_i(\theta)^2}, \quad \theta^* = \arg \min_{\theta} \text{RMSE}_{\text{obj}}(\theta) \quad (5)$$

$$\text{RMSE}_I(\theta^*) = \sqrt{\frac{1}{N} \sum_{i=1}^N (I_i^{\text{calc}}(\theta^*) - I_i^{\text{meas}})^2} \quad (6)$$

where θ^* denotes the estimated optimal parameter vector. RMSE_{obj} is the implicit-residual metric minimized during parameter extraction and avoids solving Eq. (2) at each objective evaluation. In contrast, RMSE_I is a post-estimation current-fit metric computed from I_i^{calc} , obtained by solving Eq. (2) using the Newton–Raphson method. Therefore, the two RMSE definitions are not interchangeable.

A common bounded search-space policy, informed by ranges previously adopted in PV parameter-extraction studies [2,3], was used throughout the optimization. Specifically, $I_{01}, I_{02} \in [10^{-12}, 5 \times 10^{-5}]$ A, $R_s \in [0, 0.8]$ Ω , $R_{sh} \in [5, 2 \times 10^3]$ Ω , and $n_1, n_2 \in [1, 2]$. The ideality-factor interval follows the usual physical range of 1–2 per junction [7], while the photocurrent was calculated from the complete measured I–V curve and constrained to $I_{ph} \in [0.9 I_{sc}, 1.1 I_{sc}]$ because the short-circuit current provides a close approximation to the photocurrent [7].

More generally, bounded diode-model estimation requires the definition of physically admissible parameter domains [10]. The strictly positive lower bounds for I_{01} , I_{02} , and R_{sh} , the common upper bound for R_s , and the $\pm 10\%$ photocurrent interval are implementation choices intended to avoid degenerate solutions and maintain a single reproducible search policy across modules. This common policy was adopted to facilitate reproducible cross-case comparisons and to avoid case-specific bound tuning.

In addition to the residual-based objective metric $RMSE_{obj}$, PVMSim reports complementary post-estimation metrics based on the calculated and measured responses. These include the current-fit root mean square error $RMSE_I$ defined by Eq. (6), the mean absolute error (MAE) defined by Eq. (7), the integral absolute error (IAE), defined in Eq. (8) and numerically evaluated using the trapezoidal approximation, and the relative error at the maximum power point defined by Eq. (9).

$$MAE(\theta^*) = \frac{1}{N} \sum_{i=1}^N |X_i^{calc} - X_i^{meas}| \quad (7)$$

$$IAE_X(\theta^*) = \int_{V_1}^{V_N} |X^{calc}(V) - X^{meas}(V)| dV \\ \approx \sum_{i=1}^{N-1} \frac{V_{i+1} - V_i}{2} (|e_{X,i+1}| + |e_{X,i}|) \quad (8)$$

$$\epsilon_{P_{mpp}}(\%) = 100 \frac{|P_{mpp}^{calc} - P_{mpp}^{meas}|}{P_{mpp}^{meas}} \quad (9)$$

where $e_{X,i} = X_i^{calc} - X_i^{meas}$, and X denotes either current or power. Accordingly, the current-based IAE has units of A V, whereas the power-based IAE has units of W V.

2.2. Software architecture

PVMSim follows a five-layer architecture (Fig. 2) that decouples interaction, inputs, computation, evaluation, and export, while keeping the execution path identical for the GUI and a scriptable CLI entry point.

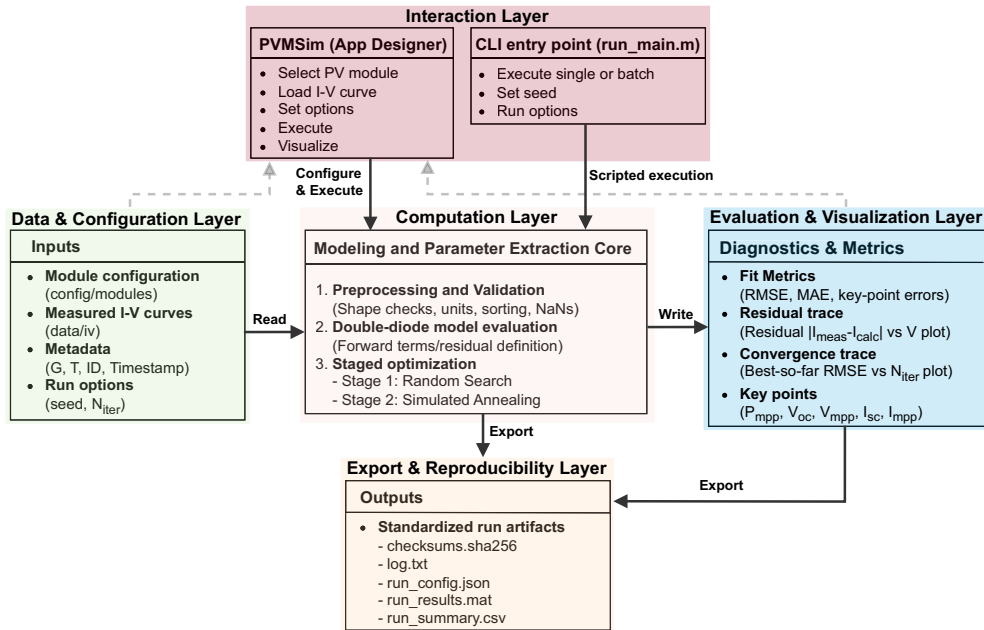


Fig. 2. Architecture of PVMSim. The GUI and CLI dispatch runs to a shared pipeline that produces standardized run artifacts for traceable reruns and comparisons. ID denotes the curve identifier, G is the irradiance, T is the temperature, and N_{iter} is the maximum number of iterations. Key points are the maximum power P_{mpp} , the open-circuit voltage V_{oc} , the maximum-power voltage V_{mpp} , the short-circuit current I_{sc} , and the maximum-power current I_{mpp} .

- **Interaction layer:** GUI and CLI entry points that dispatch runs to a shared pipeline and surface run outputs for inspection.
- **Data & configuration layer:** PV module data, measured I-V datasets, optional metadata, and run controls.
- **Computation layer:** input validation and preprocessing, DDM residual evaluation, and staged parameter extraction. The extraction problem is posed as a bounded optimization that minimizes the objective in Eq. (5). The estimation core follows a two-stage hybrid optimization scheme implemented as the RS7MSA routine, where RS7MSA denotes a Random Search and Simulated Annealing workflow for seven-parameter DDM extraction. In Stage 1, Random Search performs global exploration and rough parameter screening. In Stage 2, Simulated Annealing refines the best candidates through temperature-controlled local search.
- **Evaluation & visualization layer:** fit metrics and diagnostic evidence derived from the run results. When simulated currents are required for operating-point comparisons, Eq. (2) is solved by the Newton–Raphson method.
- **Export & reproducibility layer:** standardized run artifacts (configuration snapshots, logs, tabular summaries, MATLAB result files, and integrity hashes) written to structured output folders to enable traceable reruns and comparisons.

Both front ends call the same pipeline and produce the same standardized run artifacts without modifying the computational core. The staged parameter extraction workflow and its run artifacts are summarized in Algorithm 1.

2.3. Software functionalities

PVMSim provides the following core functionalities:

- **PV module configuration management:** PV module data are loaded from a configurable library (`config/modules`) using plain-text .txt files with MATLAB-style assignments (e.g., `Ns = 36;`). The parsed values are mapped into a panel structure so that electrical ratings and temperature-related parameters can be reused consistently across runs.

Algorithm 1 PVMSim staged parameter-extraction workflow.

Require: Module data file, measured I–V file, optional metadata, and run options (seed and iteration budgets)

Ensure: Estimated parameters θ^* , fit metrics, diagnostics, and exported run artifacts

- 1: Load the PV module data from `config/modules` and parse the parameters into a panel structure
- 2: Load the measured I–V samples from `data/iv` and validate their format and ranges
- 3: Initialize the random number generator using the configured seed and record the effective seed
- 4: **Stage 1 (Random Search):** sample candidate parameter vectors; estimate I_{ph} from the arithmetic mean over all measured samples and constrain it to $[0.9I_{sc}, 1.1I_{sc}]$; evaluate $RMSE_{obj}$; and retain the top- K candidates
- 5: **Stage 2 (Simulated Annealing):** refine the top- K candidates using the same objective function and parameter bounds, and retain the best solution
- 6: Compute the post-estimation metrics and diagnostics
- 7: Export `run_config.json`, `log.txt`, CSV summaries, MATLAB results, and `checksums.sha256`
- 8: **return** θ^* and the exported run artifacts

- **Operating-point descriptors for case setup and assessment:** the key electrical points available from the datasheet or derived from the measured curve are organized, including P_{mpp} , V_{oc} , V_{mpp} , I_{mpp} , I_{sc} , fill factor (FF), and efficiency (η) when available. These descriptors support consistency checks between the selected module and the measured dataset, operating-point assessment, and interpretation of the extracted model at characteristic PV conditions.

- **Measured I–V dataset ingestion:** measured I–V curves are imported from plain-text `.txt` files (`data/iv`) containing voltage–current pairs (V, I) and optional header metadata lines (e.g., `% ID=`, `% G=`, `% T=`, `% Date=`) used to label and organize experiments.

- **Staged parameter extraction for the DDM:** the estimation core executes a staged workflow based on the RS7MSA routine under user-defined iteration budgets and a configurable random number generator seed, which initializes the pseudo-random sequence used by the stochastic search stages. The default seed is 42.

- **Fit assessment outputs:** each run reports objective and fit metrics (RMSE-based indicators) together with run-level summaries intended to support critical evaluation of the extracted model over the measured curve and key operating regions (short-circuit, open-circuit, and maximum-power regions).

- **Reproducible run exports:** each execution writes a timestamped run folder with a standard run package, enabling traceable reruns and independent verification. An automated smoke test, accessible through the GUI and CLI, runs the STM6-40/36 benchmark with seed 42 and verifies the expected $RMSE_{obj}$, the required run artifacts, and the checksum entries.

3. Illustrative examples

Detailed operational guidance for the present version is provided in the archived user manual available through Zenodo [11]. The workflow starts by selecting a PV module definition file and a measured I–V dataset, both provided as plain-text `.txt` files. Fig. 3 summarizes the GUI workflow views, which share a common interface scaffold across tabs, including the Menubar (No. 1) for *File*, *Edit*, *Tools*, and *Help* commands.

The tab row organizes the main workflow into *Project and Data*, *Measured I–V Data*, *Algorithm*, *Plots & Metrics*, and *Batch Runs*. The persistent left-side workspace contains project and folder selection controls

(No. 2), the datasheet-parameter table with status flags for missing or validated entries (No. 3), and the application-level run log for setup and interaction events (No. 4). This persistent area supports input consistency checks before model execution and separates shared project configuration from the tab-specific analysis views.

Fig. 3(a) shows the *Measured I–V Data* tab. The graphical area (No. 5) displays the measured I–V response together with the associated P–V representation for initial inspection of the experimental input. Curve-level metadata are then summarized in the selected-curve table (No. 6). The main operating-point descriptors are reported in the key electrical points table (No. 7). This layout enables verification that the selected dataset, operating conditions, and derived operating-point descriptors are consistent before starting parameter extraction.

Fig. 3(b) presents the *Algorithm* tab. The convergence and execution area (No. 8) displays the best-so-far RMSE versus iteration and provides the main run controls for the staged estimation routine. The configuration controls (No. 9) define the random seed, iteration budget, axis-scaling option, and export or reset actions used to launch and inspect the run. The extracted-parameter table (No. 10) reports the estimated vector θ^* with units, while the algorithm log (No. 11) records the effective settings, execution events, and final fitting status. This layout supports traceable interpretation of each run under fixed computational conditions.

Fig. 3(c) illustrates the *Plots & Metrics* tab. The main comparison plots (No. 12) show measured and calculated I–V and P–V responses across the full operating range, while the residual plot (No. 13) supports identification of localized mismatch patterns and systematic bias. The metrics table (No. 14) reports current- and power-based fit indicators, including RMSE, MAE, IAE, and the relative error at P_{mpp} (Eqs. 5–9), enabling complementary graphical and numerical assessment of model adequacy.

Fig. 3(d) corresponds to the *Batch Runs* tab. The run-status table (No. 15) lists run-level outputs for repeated executions, including seed, objective RMSE, runtime, and completion status. The batch-summary panel (No. 16) aggregates dispersion statistics such as best, worst, mean, standard deviation, and average execution time per run. The execution controls (No. 17) define the number of runs, consecutive-seed mode, cancellation, and summary export. This tab extends single-run analysis toward systematic quantification of run-to-run variability and robustness under repeated optimization trials.

In the tabs that include plotting axes (Fig. 3a, No. 5; Fig. 3b, No. 8; Fig. 3c, No. 12 and 13), the graphics inherit MATLAB's built-in axes interactions. Through the native menu, plots can be copied to the clipboard and exported to common formats (e.g., PNG, PDF, and SVG), along with standard navigation options such as zoom, pan, and data inspection, supporting rapid figure reuse in reports and documentation.

3.1. Comparison with published parameter extraction results

Table 1 reports $RMSE_{obj}$ statistics, defined in Eq. (5), for 21 independent runs reported by Bakir [2] using a five-parameter PV model and 30 independent PVMSim runs using seeds 42–71. The table therefore compares residual-based fitting statistics on common benchmark datasets rather than identical model parameterizations. Each PVMSim run used 880,010 objective-function evaluations: 80,000 in Stage 1 and 800,010 in Stage 2. Because the computational budgets differ, efficiency is not compared. Run artifacts and supplementary diagnostics support reproducibility, fitting assessment, convergence analysis, and run-to-run variability. For direct current-fit assessment, PVMSim also reports $RMSE_c$; the corresponding 30-run statistics for each module are provided in the Supplementary Materials.

For Photowatt-PWP201, the negative current values near the high-voltage end in Supplementary Figs. S7–S8 were retained from the published benchmark dataset. The corresponding negative power values follow from $P_i = V_i I_i$ and were preserved for comparability [3].

PVMSim reached comparable minimum RMSE values for STP6-120/36 and STM6-40/36 relative to the strongest literature baselines,

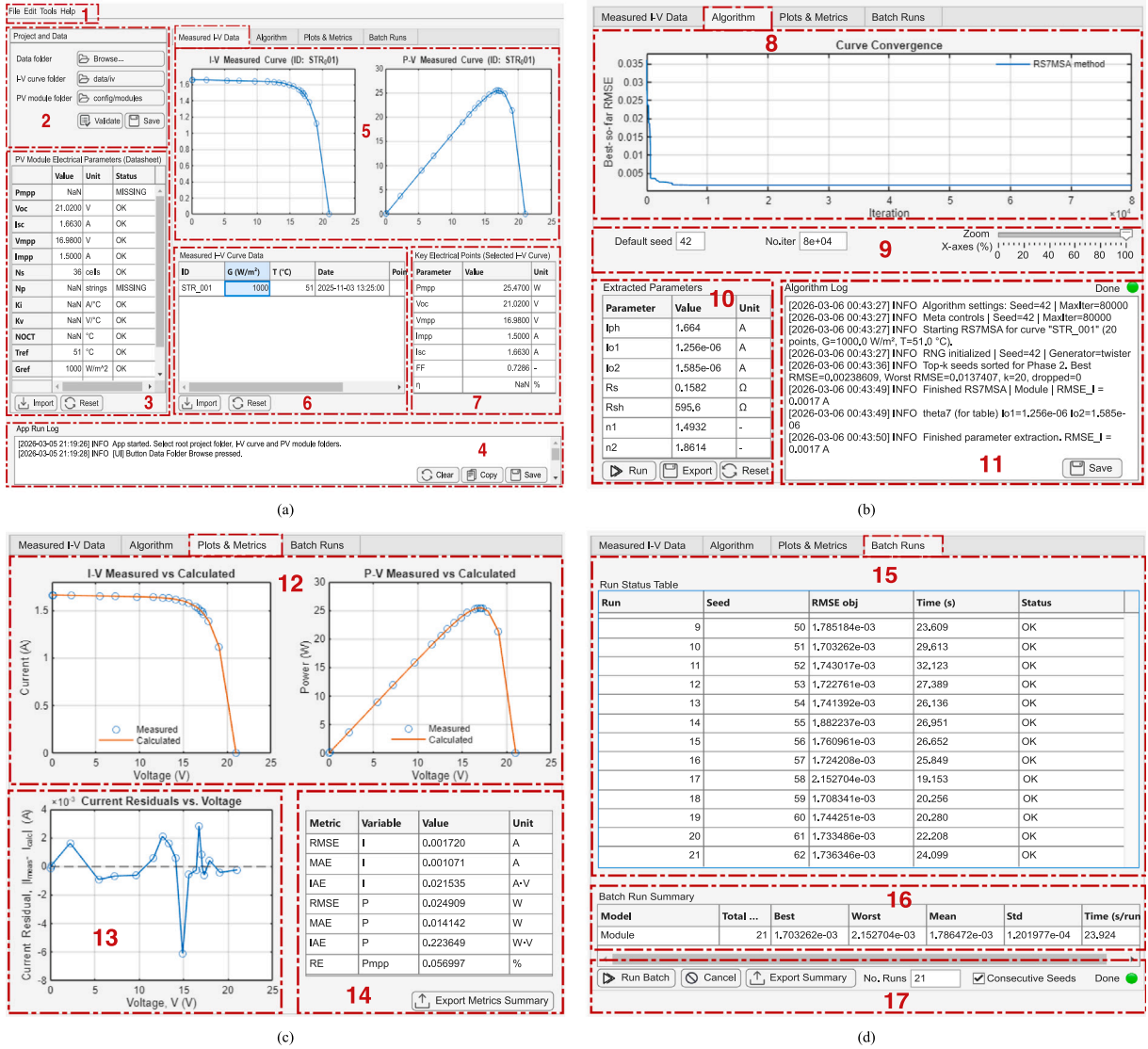


Fig. 3. Graphical interface of PVMSim across the main workflow views: (a) project and data setup, measured I–V data visualization, and key electrical points; (b) staged parameter-extraction settings; (c) model validation and fit assessment; and (d) multi-run execution and summary statistics.

while maintaining low run-to-run dispersion under the 30-run protocol. For Photowatt-PWP201, PVMSim produced stable repeated-run behavior but higher RMSE than the best-performing published methods. PVMSim applied the same search-space policy to all three benchmark PV modules, whereas the comparison studies used module-specific parameter ranges [2,3]. This common policy was adopted to facilitate reproducible cross-case comparisons and avoid case-specific bound tuning. Because the effect of alternative bounds was not isolated experimentally, the higher RMSE obtained for Photowatt-PWP201 cannot be attributed conclusively to the search-space policy.

3.2. Limitations

The current version of PVMSim requires a MATLAB execution environment. The present validation is based on benchmark I–V datasets; broader field measurements are required to assess PVMSim under non-ideal operating and measurement conditions [12,13]. These findings also motivate careful specification of physically admissible bounds in bounded parameter estimation [10]. In addition, low fitting error does not necessarily imply parameter uniqueness or physical validity, while RS7MSA performance may depend on the random seed, evaluation

budget, and parameter bounds. PVMSim records these settings and the corresponding run artifacts to support traceable reruns and controlled comparisons.

4. Impact

PVMSim supports reproducible DDM parameter extraction from measured I–V curves by standardizing run configuration, execution, evaluation, and archival. This methodological framework facilitates benchmarking of optimization methods under common inputs and reporting conventions, including analyses of run-to-run variability, sensitivity of extracted parameter sets to algorithm settings, and comparative assessment of alternative optimization strategies.

Existing DDM parameter-extraction studies frequently rely on custom scripts and study-specific reporting practices, which hinder direct comparison even when similar metrics and graphical diagnostics are used [2,3]. PVMSim addresses this limitation through a shared execution pipeline that integrates module configuration, dataset ingestion, staged optimization, metric reporting, batch processing, and standardized artifact export. The MATLAB application and CLI toolkit are distributed through a version-controlled repository containing source code, example

Table 1Summary statistics of $RMSE_{obj}$ (A) reported by Bakır [2] and computed with PVMSim.

PV Module	Method	Minimum	Mean	Maximum	Std. Dev.
STP6-120/36	dFDB-LSHADE	1.621182E-02	1.621182E-02	1.621182E-02	2.409392E-16
	NMPA	1.690965E-02	3.992353E-02	6.487523E-02	1.663914E-02
	HO	5.924062E-02	4.665467E-01	1.413072E + 00	4.872211E-01
	MPA	1.652519E-02	3.765048E-02	5.215434E-02	1.239230E-02
	WO	2.275096E-02	4.658804E-02	5.347595E-02	8.848680E-03
	EDO	1.621219E-02	1.831568E-02	2.272765E-02	1.488258E-03
	MRFO	1.636772E-02	1.644778E-02	1.660994E-02	6.216908E-05
	PVMSim (RS7MSA)	1.622450E-02	1.6705981E-02	1.788860E-02	6.453528E-04
STM6-40/36	dFDB-LSHADE	1.729813E-03	1.806862E-03	3.347834E-03	3.530810E-04
	NMPA	2.774349E-03	6.547779E-03	1.393116E-02	3.441122E-03
	HO	9.460250E-03	1.223769E-01	3.114239E-01	1.381055E-01
	MPA	5.235363E-03	7.014880E-03	9.575224E-03	1.578341E-03
	WO	2.754829E-03	5.359935E-03	7.994318E-03	1.485238E-03
	EDO	1.820817E-03	2.277044E-03	3.089282E-03	2.878172E-04
	MRFO	1.740955E-03	1.861272E-03	2.172230E-03	9.705280E-05
	PVMSim (RS7MSA)	1.703262E-03	1.786472E-03	2.152704E-03	1.201977E-04
Photowatt-PWP201	dFDB-LSHADE	2.425074E-03	2.425074E-03	2.425074E-03	4.395094E-17
	NMPA	2.916215E-03	6.951525E-03	1.454081E-02	3.441122E-03
	HO	1.193079E-02	8.842807E-02	2.780153E-01	7.315842E-02
	MPA	2.938104E-03	8.177273E-03	1.567910E-02	3.920409E-03
	WO	2.894690E-03	7.500612E-03	9.443548E-03	1.905744E-03
	EDO	2.526134E-03	3.134788E-03	4.150668E-03	3.729543E-04
	MRFO	2.455927E-03	2.665693E-03	2.991614E-03	1.438556E-04
	PVMSim (RS7MSA)	1.219914E-02	1.231293E-02	1.240668E-02	8.104750E-05

dFDB-LSHADE: dynamic fitness-distance balance LSHADE; NMPA: nonlinear marine predator algorithm; HO: hippopotamus optimization; MPA: marine predator algorithm; WO: walrus optimizer; EDO: exponential distribution optimizer; MRFO: manta-ray foraging optimization.

datasets, and documentation, reducing the need for ad hoc implementations and supporting later inspection and replication.

Although developed primarily for academic research, PVMSim may also support field-oriented workflows in which measured I–V curves are used to investigate PV performance losses. Such studies commonly require standardized processing, including IEC 60891 correction to standard test conditions, to compare measurements acquired under different operating conditions [14]. Parameters inferred from I–V curves can contribute to the interpretation of long-term performance changes and module-to-module mismatch [15]. In this context, PVMSim provides traceable parameter extraction, diagnostic visualization, and standardized run artifacts for validation-oriented analyses.

5. Conclusions

PVMSim is a MATLAB application for traceable parameter extraction of the double-diode photovoltaic model from measured I–V curves. The software integrates data ingestion, staged optimization, fit assessment, and standardized export of run artifacts into a shared workflow accessible through both GUI and CLI execution modes. This design supports controlled reruns, transparent inspection of run settings and outputs, and structured comparisons across datasets and computational configurations.

By packaging configuration snapshots, logs, numerical summaries, result files, and integrity hashes within each run, PVMSim provides a practical framework for reproducible DDM studies based on measured curves. The current version is suited for benchmarking optimization workflows, assessing fitting consistency, and supporting traceable parameter-identification studies in photovoltaic research.

CRedit authorship contribution statement

Lionnis Osorio: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Mailyn Moreno-Espino:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis,

Conceptualization. **Laurent Duchêne:** Writing – review & editing, Supervision, Methodology, Formal analysis. **Víctor Tuninetti:** Writing – review & editing, Validation, Methodology, Formal analysis. **Carlos Zalazar:** Writing – original draft, Validation, Software, Data curation. **Rodrigo Irrázaval:** Writing – original draft, Software, Resources, Data curation. **Yoalys Retirado-Mediaceja:** Writing – original draft, Visualization, Methodology, Investigation. **Marco Rivera:** Writing – review & editing, Validation, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work has been supported by the **Comunidad de Madrid**, Programa de Atracción de Talento Investigador «César Nombela» 2024, grant **T1/COM-31503**.

Appendix A. Supplementary data

Supplementary data to this article can be found online at doi:10.1016/j.softx.2026.102975.

References

- [1] Venkateswari R, Rajasekar N. Review on parameter estimation techniques of solar photovoltaic systems. *Int Trans Electr Energy Syst* 2021;31:e13113. <https://doi.org/10.1002/2050-7038.13113>
- [2] Bakır H. Elevating PV model performance: accurate and reliable parameter extraction of solar cell models with state-of-art metaheuristic algorithms. *Next Energy* 2026;11:100513. <https://doi.org/10.1016/j.nxener.2026.100513>
- [3] Li S, Gong W, Yan X, Hu C, Bai D, Wang L, Gao L. Parameter extraction of photovoltaic models using an improved teaching-learning-based optimization. *Energy Convers Manag* 2019;186:293–305. <https://doi.org/10.1016/j.enconman.2019.02.048>
- [4] Núñez R, Moreno M, Herrero R, Askins S, Antón I, Domínguez C. cpvlib: a comprehensive open-source tool for modeling CPV systems. *Sol Energy Mater Sol Cells* 2025;279:113245. <https://doi.org/10.1016/j.solmat.2024.113245>

- [5] Holmgren WF, Hansen CW, Mikofski MA. pvlib Python: a Python package for modeling solar energy systems. *J Open Source Softw* 2018;3:884. <https://doi.org/10.21105/joss.00884>
- [6] Nishat MM, Faisal F. Investigation and modeling of solar PV: a simulation method. In: 2024 21st international conference on electrical Engineering/Electronics, computer, telecommunications and information technology (ECTI-CON); 2024. p. 1–6. <https://doi.org/10.1109/ECTI-CON60892.2024.10594817>
- [7] Mermoud A, Lejeune T. Performance assessment of a simulation model for PV modules of any available technology. In: 25th European photovoltaic solar energy conference, vol. 1; 2010. p. 5. [http://refhub.elsevier.com/S0927-0248\(24\)00557-9/sref6](http://refhub.elsevier.com/S0927-0248(24)00557-9/sref6).
- [8] Cugnet M, Gallois F, Kirchev A, Dutykh D. NEOLAB: a Scilab tool to simulate the negative electrode of lead-acid batteries. *SoftwareX* 2023;22:101394. <https://doi.org/10.1016/j.softx.2023.101394>
- [9] Karayel M, Maden D, Abdel-Salam M, Öztürk N, Kaplan O, Tejani GG, Sharma SK, Baljon M. Reconfigured single- and double-diode models for improved photovoltaic parameter estimation. *Sci Rep* 2025;15:19. <https://doi.org/10.1038/s41598-025-86063-2>
- [10] Cárdenas-Bravo C, Dutykh D, Lespinats S. On the parameters domain of the single-diode model. *Sol Energy* 2024;277:112718. <https://doi.org/10.1016/j.solener.2024.112718>
- [11] Osorio L, Moreno-Espino M, Duchêne L, Tuninetti V, Zalazar C, Irrarrázaval R, Retirado-Mediacaja Y, Rivera M. PVMSim user manual. Zenodo; 2026. <https://doi.org/10.5281/zenodo.19040390>
- [12] Cárdenas-Bravo C, Dutykh D, Ha DL. Computation of faulty IV curves based on a distributed solar cell algorithm. In: 8th world conference on photovoltaic energy conversion (WCPEC-8); 2022. p. 701–3. <https://doi.org/10.4229/WCPEC-82022-3BV.3.32>
- [13] Cárdenas-Bravo C, Lespinats S, Dutykh D. A practical example of the impact of uncertainty on the one-dimensional single-diode model. In: 2024 IEEE international symposium on systems engineering (ISSE); 2024. p. 1–5. <https://doi.org/10.1109/ISSE63315.2024.10741111>
- [14] Osorio L, Duchêne L, Tuninetti V, Narayan S, Moreno-Espino M, Rivera M. Degradation and projected lifetime of polycrystalline silicon photovoltaic modules after 10 years of field exposure in the Atacama Desert. *Renew Energy* 2026;260:125118. <https://doi.org/10.1016/j.renene.2025.125118>
- [15] Huang C, Wang L. Simulation study on the degradation process of photovoltaic modules. *Energy Convers Manag* 2018;165:236–43. <https://doi.org/10.1016/j.enconman.2018.03.056>