APPLICATION NOTE

Beyond Traditional Optimizers for Device Modeling

KEYSIGHT

Understanding the Limitations of Traditional Optimizers

Compact model extraction can be formulated as an optimization problem in which a vector of model parameters is adjusted to minimize the discrepancy between measured and simulated device characteristics. The optimization task is to find parameter values that minimize the cost function, subject to applicable parameter bounds and modeling constraints.

In practice, several challenges arise:

- **Local minima:** Different parameter combinations can produce different local minima in the cost function. When a traditional optimizers finds a local minimum, it exits and misses the true global minimum.
- **Parameter coupling:** Multiple parameters may influence the same electrical characteristic, making their individual contributions difficult to distinguish.
- **Sensitivity differences:** Parameters may have substantially different scales and sensitivities, complicating numerical optimization.
- **Poor initial conditions:** An initial parameter set may lie in a region where local optimization makes limited progress or converges to an unacceptable solution.
- **Computational expense:** Repeated circuit simulations and cost-function evaluations can dominate extraction time when solution doesn't converge.
- **PDK-specific requirements:** Different compact models, device geometries, and characterization datasets may require different optimization strategies.

The key is not to replace every traditional optimizer. However, to complement traditional optimization with methods that explore the parameter space more effectively or allow modeling engineers to tailor the search process to meet their needs.

Keysight Integrated Circuit Characterization and Analysis Program (IC-CAP) 2027 offers a broader set of advanced optimization search methods for addressing these limitations. This includes the updated Keysight Machine Learning (ML) Optimizer, Hybrid ML Optimizer, Particle Swarm Optimization (PSO), and Bring Your Own (BYO) Python Optimizer along with several new features such as dynamic error plotting, custom Python cost functions etc.

This application note examines how and when these new complementary capabilities can be used to build flexible and efficient device-model extraction workflows. This helps engineers with selecting an optimization strategy that matches the characteristics of their modeling problem.

New Optimizers in IC-CAP 2027

IC-CAP 2027 provides several optimizer-based search methods, each with different characteristics and intended uses as shown in Table 1.

Optimization method	Key differentiator	Practical consideration
Keysight ML Optimizer	Adaptive optimization search	Useful for complex model fits that require more extensive search, may take longer times but yields accurate results e.g. shipping PSP flow example
Hybrid ML Optimizer	Global search followed by local refinement	Combines exploration and refinement in a single workflow. Offers runtime improvements without sacrificing much of fitting accuracy e.g. BSIM-BULK flow example can finish at least 3X faster compared to ML optimizer under identical conditions.
Particle Swarm Optimization	Population-based search	Useful for difficult initial conditions and multimodal error surfaces e.g. PSP flow with default textbook values and ranges.
BYO Python Optimizer	User-defined optimization strategy	Enables customized algorithms and optimization logic e.g. Differential Evolution optimizer implemented through Scikit-learn

Table 1. Overview of new optimizers

The choice of optimizer should reflect the fitting objective, number of parameters, computational budget, parameter constraints, and desired balance between exploration and refinement.

Keysight ML Optimizer

The updated Keysight ML Optimizer offers a gradient-free, machine-learning based search along the normalized parameter space. You can use it for high-dimensional problems and for wider parameter ranges while each parameter is able to use linear or logarithmic sampling. To improve poor fitting, you can try increasing Max Iterations, increasing population size, narrowing the ranges and lower the learning rate. Table 2 below shows default and recommendations for each option:

Option	Default	Practical recommendations
Max Iterations	500	Depends on the problem. Simpler problems may converge in 100-300 iterations while complex, high-dimensional problems can benefit from 1,000 to 5,000 iterations
Population Size	10	For fewer than 5 parameters, a population of 6 to 10 is usually enough. For 20 or more parameters, or for a highly non-separable problem, a population of 20 to 40 usually gives a better result.
Initial Step Size	0.5	A value near 0.5 explores about half of the parameter range on the first iteration. A smaller value, such as 0.001, keeps the search nearer the nominal values
Learning Rate	1.0	A lower value, such as 0.3, makes the search smoother. The default is a good start.

Table 2. Default and recommended values of ML optimizer options

IC-CAP 2027 offers tools such as Error plot and Optimizer logging to help you diagnose the performance and convergence behavior of any optimizer. You can visualize the optimizer error evolution with iterations plot by selecting appropriate drop down menu. You can also save out the optimization information per evaluation to a JSON file by setting the following environment variable:

ICCAP_OPTIMIZER_LOG_DIR: Directory for all optimizer log files. If unset it defaults to:

(On windows): C:\Documents\Keysight\IC-CAP\optimizer_logs

(On Linux): ~/Documents/Keysight/IC-CAP/optimizer_logs

ICCAP_OPTIMIZER_LOG_SIM_VALUES: Also write per-evaluation simulated values during optimization to the log directory in a HDF5 file. It is off by default.

ICCAP_OPTIMIZER_LOG_MAX_RUNS: Maximum number of completed runs to keep in the log directory. If unset it defaults to 10.

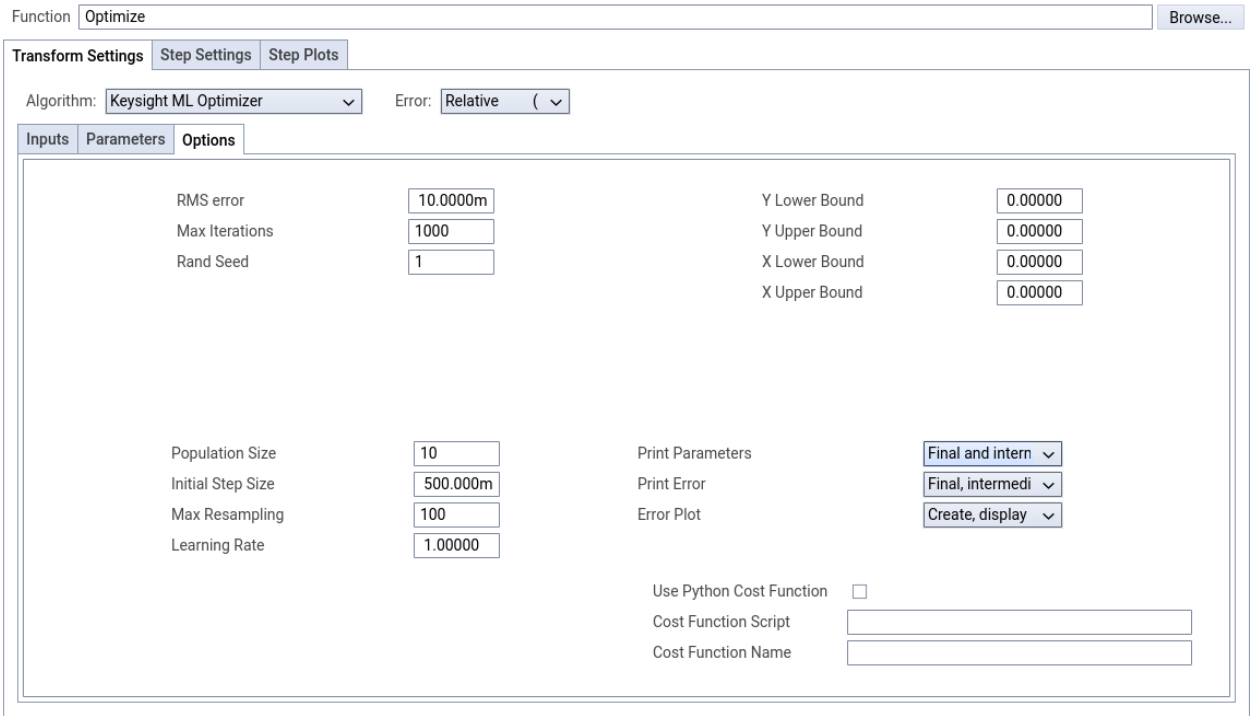


Figure 1. Options for Keysight ML optimizer

Hybrid ML Optimizer

The Hybrid ML Optimizer provides an integrated strategy for problems that benefit from both global exploration and local exploitation while offering a significant runtime improvement at a negligible loss of accuracy. As an example, for BSIM-BULK flow, this optimizer shows 3X speed up with less than ~5% loss of accuracy. Searching globally first allows the optimizer to explore the parameter space before concentrating on improving the fit locally. Conceptually, exploration must be concluded before initiating the local refinement. Hence, Global-to-local ratio option plays an important role which by default (0) allows automatic allocation of 20-60% of the iteration budget in the exploration phase. This strategy is particularly useful when modeling engineers want a better fit through a single integrated search process without manually coordinating separate global and local optimization runs. However, when fit quality remains the primary objective and the problem is particularly difficult, a longer Keysight ML Optimizer search may be appropriate.

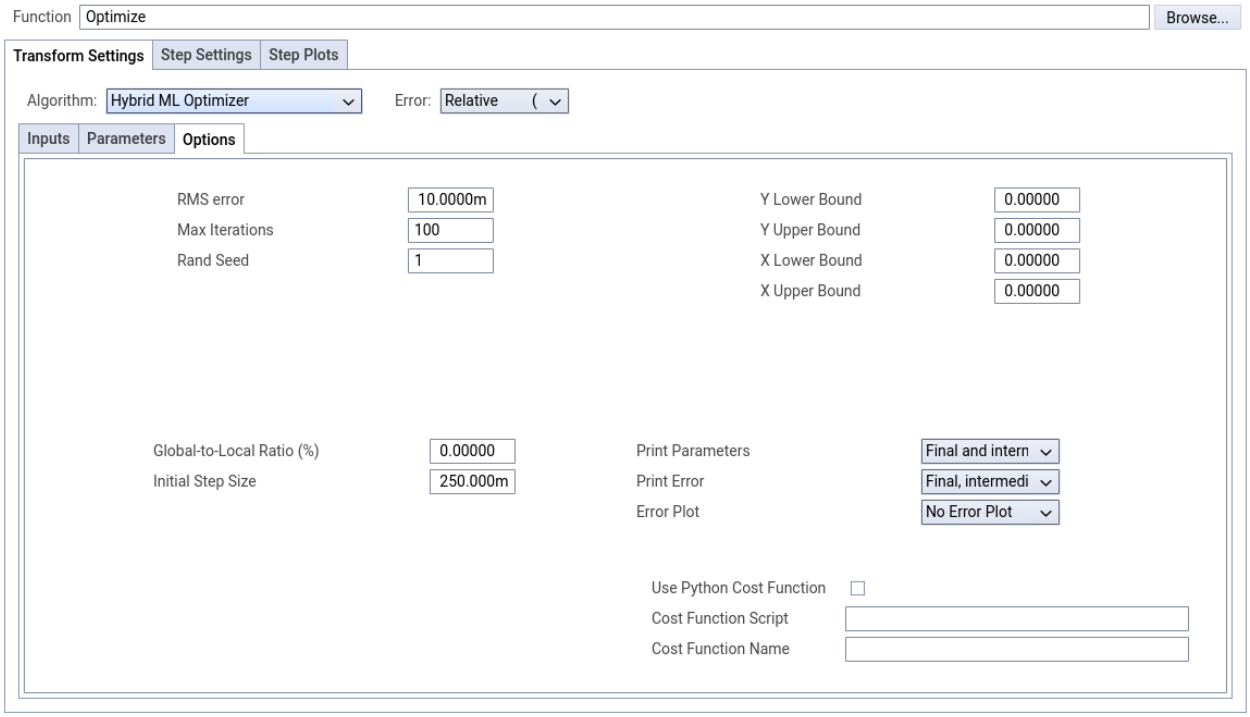


Figure 2. Options for Hybrid ML optimizer

Consider Hybrid ML when:

- The initial parameter estimates are not sufficiently reliable for local refinement alone.
- The error surface may contain multiple local minima and you want the global search and the local refinement in one run.
- The extraction process needs to balance solution quality and engineering time.

The resulting fit can be assessed against the relevant measured characteristics, parameter ranges, and physical modeling considerations.

Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a population-based optimization method inspired by the collective movement of a swarm. Instead of updating one candidate solution at a time through a local derivative-based search, PSO evaluates multiple candidate parameter vectors and uses information from their search history to guide subsequent exploration.

Each particle represents a candidate set of compact model parameters in IC-CAP. As the swarm evolves, candidate solutions move through the parameter space according to the PSO update rules (position, velocity). PSO is useful for device modeling problems when initial parameter estimates are far from desired solution and hence, it is usually followed by Levenberg-Marquardt (L-M) or Keysight ML optimizer to further refine the model fit.

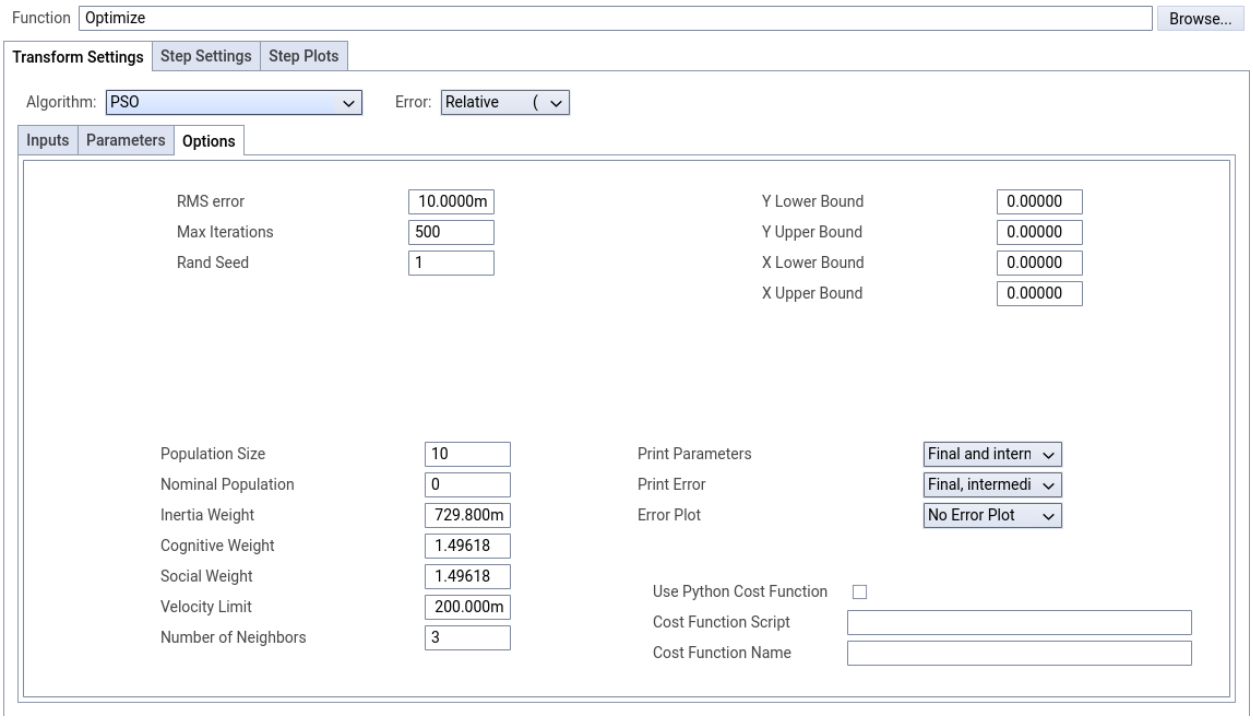


Figure 3. Options for PSO optimizer

BYO Python Optimizer

Different modeling problems may require optimization capabilities that are not adequately addressed by a single predefined optimizer. The Bring Your Own Python (BYO Python) Optimizer provides a route for incorporating a user-defined Python optimization strategy into the IC-CAP modeling workflow. Rather than restricting the engineer to a fixed search algorithm, this approach allows the optimization logic to be tailored to the specific extraction problem and allows flexibility to let you define your own optimization logic.

Potential applications include:

- Integrating a preferred numerical optimization algorithm or experimenting with alternative search strategies.
- Implementing custom parameter-update rules or candidate-selection logic.
- Incorporating domain-specific constraints into the search process.
- Introduce customized stopping criteria or your own Python custom cost function.
- Connecting existing Python optimization code to a device-modeling workflow.

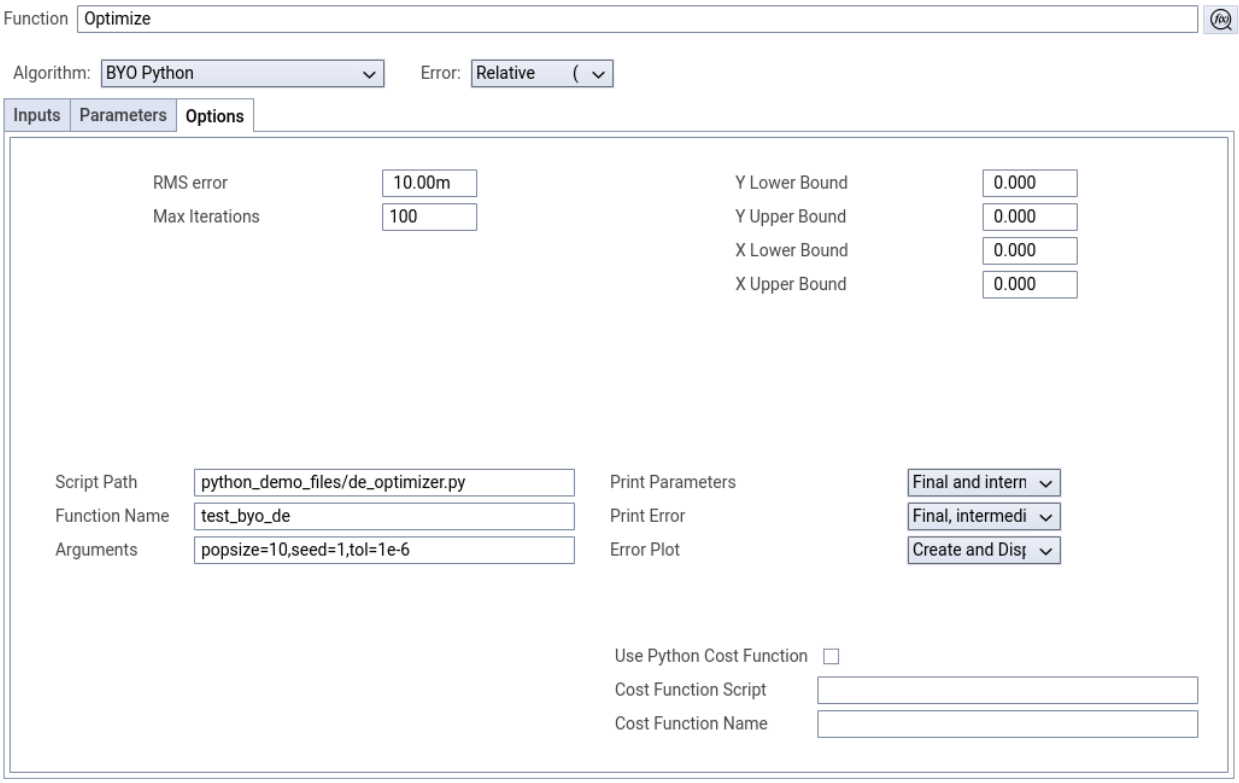


Figure 4. Options for BYO Python optimizer

The broader Python optimization ecosystem offers numerous methods that can be useful when designing a custom search strategy. The appropriate algorithm depends on whether derivatives are available, whether the search is local or global, and what constraints must be supported. While working within the IC-CAP framework, measurement data, circuit simulation, parameter denormalization, setting ranges and weights, cost evaluation and stopping condition enforcement are all handled by IC-CAP for your new BYO Python optimizer candidate. Please refer to differential evolution optimizer example shipping in IC-CAP 2027.

Choosing the Right Optimization Method

The following decision framework provides a practical starting point for selecting an optimization strategy.

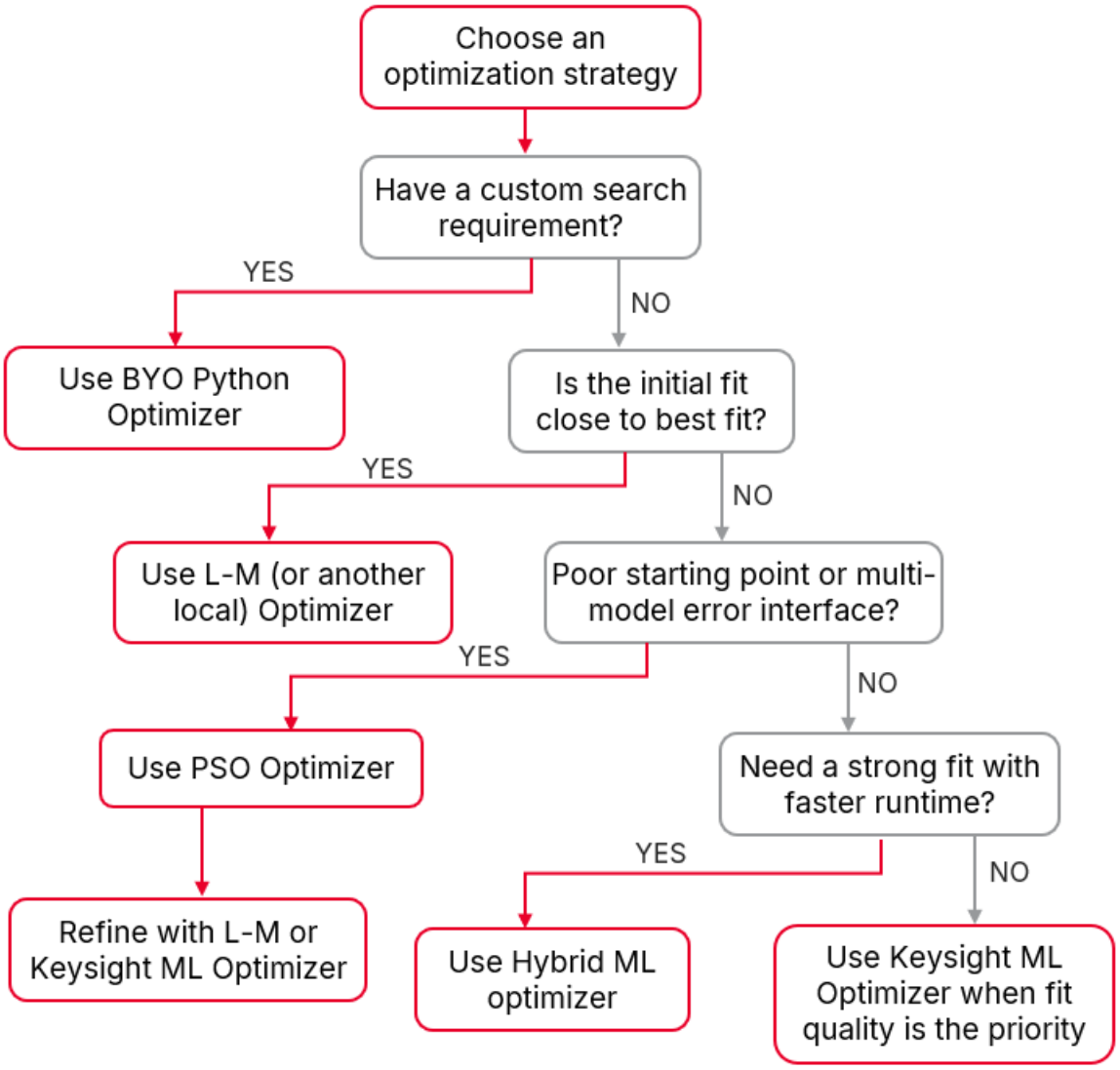


Figure 5. Systematic flow to decide which optimizer to choose from

These recommendations are starting points rather than universal rules. The appropriate method should be selected based on the actual device model, measurement data, available parameter ranges, and available computational resources.

Conclusion

Efficient compact model extraction requires more than selecting an optimization algorithm. It requires a search strategy that matches the complexity of the model, the quality of the initial parameter estimates, and the resources available.

Keysight IC-CAP offers complementary approaches for this task. Hybrid ML combines global exploration with local refinement, PSO provides population-based exploration for challenging search landscapes, and BYO Python enables engineers to develop customized optimization strategies within the supported modeling workflow.

Traditional methods such as L-M remain valuable for local refinement, particularly when the model evaluation is near the desired solution. By combining complementary search methods and assessing their performance, engineers can develop more flexible workflows for challenging compact model extraction tasks.

Keysight enables innovators to push the boundaries of engineering by quickly solving design, emulation, and test challenges to create the best product experiences. Start your innovation journey at www.keysight.com.

KEYSIGHT

This information is subject to change without notice. © Keysight Technologies, 2026, Published in USA, October 27, 2026, 3126-1853.EN