GlobalSearch-rs: A multistart framework for global optimization written in Rust

Germán Martín Heim [1,2]

1 Universidad Nacional del Sur, Department of Chemical Engineering, Bahía Blanca, Buenos Aires, Argentina **ROR** **2** The Dow Chemical Company, Bahía Blanca, Buenos Aires, Argentina

Summary

GlobalSearch-rs is an open-source global optimization library written in Rust that implements a multistart optimization strategy. It is based on the OptQuest/NLP (OQNLP) algorithm, combining scatter search metaheuristics with local nonlinear programming (NLP) solvers. The software is designed to solve nonlinear global optimization problems by leveraging efficient parallel computing and Rust's memory safety guarantees. Additionally, GlobalSearch-rs provides Python bindings, allowing seamless integration with Python-based workflows, making it a versatile tool for working in scientific computing and data-driven optimization tasks. This library provides an alternative to proprietary solutions like MATLAB's GlobalSearch, offering a high-performance, open-source alternative accessible to the scientific and engineering communities.

Statement of need

Global optimization is a fundamental problem in engineering, physics, and applied mathematics, where objective functions often exhibit multiple local minima. The existing OQNLP algorithm (Ugray et al., 2007) effectively balances exploration (broad exploration of the search space) and exploitation (local refinement of promising regions) by employing a multi-start strategy with scatter search and local optimization.

Building on these algorithmic advances, practical software implementations have become essential for researchers and engineers. MATLAB's GlobalSearch, for example, has been widely adopted, with applications ranging from airfoil design optimization (Liu et al., 2022) to kinetic mechanism analysis of polyurethane decomposition (Chen et al., 2021). However, despite its popularity, to the best of our knowledge, no actively maintained open-source alternative currently exists. The only open-source implementation, MEIGO (Egea et al., 2014), is still written in MATLAB, a proprietary language, limiting accessibility.

GlobalSearch-rs addresses this gap and extends its functionality with additional features designed to improve practical usability. It offers a modern, high-performance implementation for bound-constrained nonlinear programming (NLP) problems:

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & f(x) \\ \text{s. t.} \quad & l_i \le x_i \le u_i, \quad i = 1, \dots, n \end{aligned}$$

Developed in Rust—a free and open-source programming language—it combines accessibility with performance, ensuring that researchers are not limited by proprietary ecosystems. Rust's ownership model and concurrency features enable safe and efficient execution, making the

library suitable for high-performance computing applications, a suitability further supported by recent work demonstrating Rust's advantages in high-performance computing ([Sudwoj, 2020](#)).

Figure 1: `GlobalSearch-rs` logo.

Library features

`GlobalSearch-rs` implements scatter search metaheuristics, which balances diversification and intensification strategies to navigate complex objective landscapes. By leveraging these techniques, the algorithm improves the exploration of global minima while refining promising candidate solutions through local search. However, as with other stochastic global optimization algorithms, `GlobalSearch-rs` cannot guarantee that the best solution found is the true global minimum. Instead, it aims to identify high-quality solutions with high probability. Users can increase confidence by performing multiple independent runs or using different random seeds.

The library integrates with both the `cobyla` and `argmin` crates. These Rust-based optimization libraries provide a range of local NLP solvers. Supported methods include COBYLA, LBFGS, Nelder-Mead, Steepest Descent, Newton-Conjugate-Gradient and Trust Region solvers, ensuring flexibility in handling different types of objective functions ([Lavezzi et al., 2023](#)).

Parallel execution is a core feature of `GlobalSearch-rs`, utilizing the Rayon library to distribute computation across multiple threads. This has the potential to enhance computational efficiency, enabling users to tackle large-scale optimization problems with improved performance compared to single-threaded alternatives. To further support long-running tasks, the library provides checkpointing capabilities, allowing users to save and resume optimization states without restarting from scratch.

As an open-source and cross-platform library, `GlobalSearch-rs` is designed to be accessible to researchers and engineers from various fields. The Python bindings, created using `PyO3` and `maturin`, enable users to leverage its capabilities within Python-based numerical and data science environments (e.g., `NumPy`, `JAX`), making it a practical choice for interdisciplinary research and engineering applications.

Figure 2: `PyGlobalSearch` logo.

The software is distributed under the permissive MIT license, encouraging use, modification, and contribution from the community. To ensure reliability and maintain quality, continuous integration testing is performed via GitHub Actions, providing automated validation of builds and functionality across supported platforms.

Example: Optimizing the Six-Hump Camel Function

To illustrate a concrete use case, let's consider a common benchmark problem in global optimization: the two-dimensional Six-Hump Camel function. The function is defined in Equation 1. It is continuous, differentiable and non-convex.

$$f(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{4} \right) x_1^2 + x_1 x_2 + (-4 + 4x_2^2)x_2^2 \tag{1}$$

The function is evaluated in the domain (bounds) $x_1 \in [-3, 3]$ and $x_2 \in [-2, 2]$. It has two global minima at approximately $\mathbf{x}^* = (0.0898, -0.7126)$ and $\mathbf{x}^* = (-0.0898, 0.7126)$ with a function value of $f(\mathbf{x}^*) \approx -1.0316$.

To solve this optimization problem using `GlobalSearch-rs`, the algorithm first generates an initial population of candidate solutions within the prescribed bounds, forming the reference set. Figure 3 presents the contour plot of the Six-Hump Camel function alongside the initial distribution of candidate solutions.
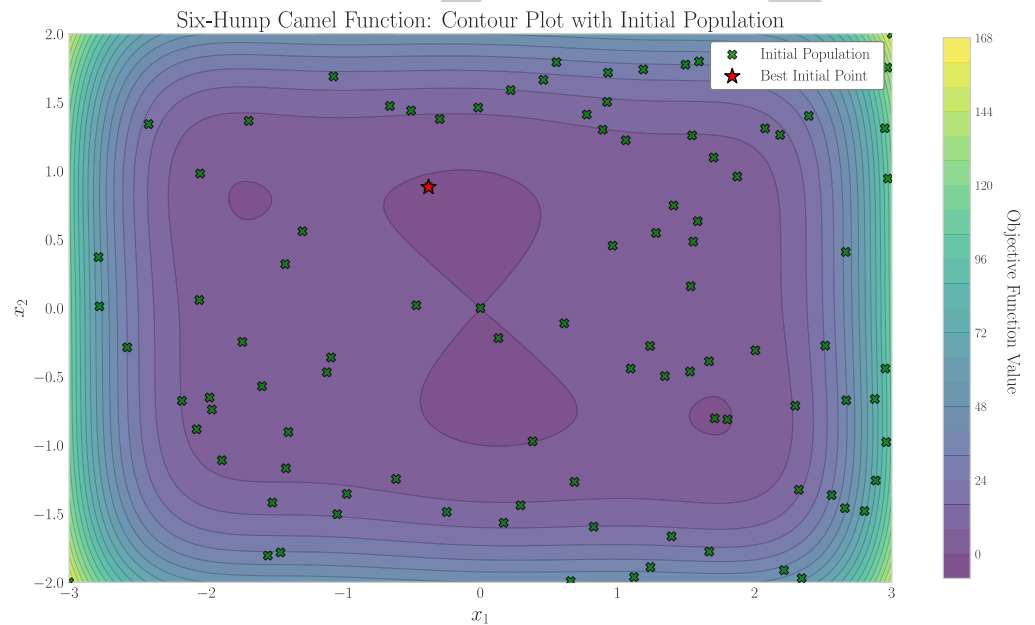

Figure 3: Contour plot of the Six-Hump Camel function with initial population.

The red star marks the best candidate in the initial population, used as the starting point for the first local optimization.

The algorithm then enters the stage two, where it performs local optimization on the selected reference set points that have been identified as promising (i.e. those within the merit and distance thresholds). Figure 4 shows the contour plot and surface of the function colored by the stage two iteration number.
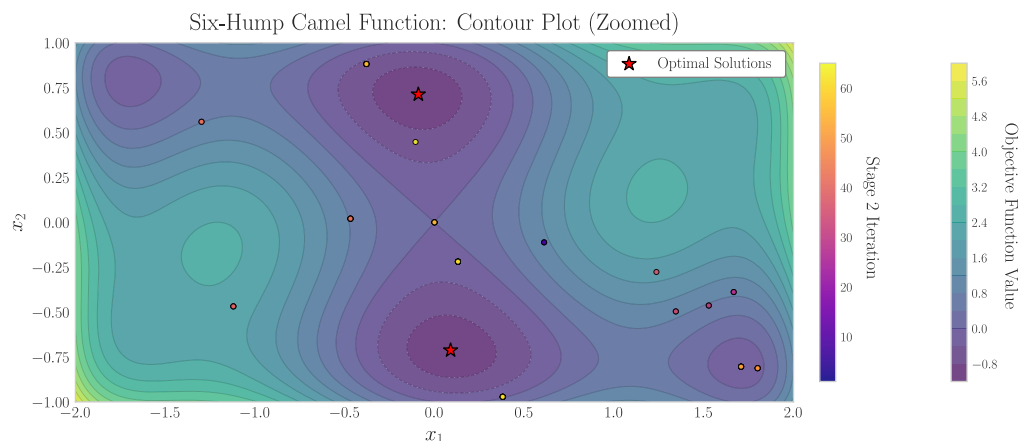
Figure 4: Contour plot of the Six-Hump Camel function during stage two.

The algorithm successfully finds both global minima of the Six-Hump Camel function.

Conclusion

GlobalSearch-rs provides a robust and accessible open-source framework for nonlinear global optimization. By combining parallelization, flexible local solvers, and Python integration, it offers a compelling alternative to proprietary tools. The Rust implementation ensures efficiency and reliability, supporting transparent and reproducible optimization workflows across scientific and engineering domains. This tool advances the state of open-source global optimization by delivering a modern and extensible solution for complex problems.

References

Chen, Y., He, H., & Liu, Z. (2021). Application of the global search algorithm to analyze the kinetic mechanism of the thermal decomposition of flexible polyurethane foam. *Progress in Reaction Kinetics and Mechanism*, *46*. https://doi.org/10.1177/1468678320982185

Egea, J. A., Henriques, D., Cokelaer, T., Villaverde, A. F., MacNamara, A., Danciu, D.-P., Banga, J. R., & Saez-Rodriguez, J. (2014). MEIGO: An open-source software suite based on metaheuristics for global optimization in systems biology and bioinformatics. *BMC Bioinformatics*, *15*(1), 136. https://doi.org/10.1186/1471-2105-15-136

Lavezzi, G., Guye, K., Cichella, V., & Ciarcià, M. (2023). Comparative analysis of nonlinear programming solvers: Performance evaluation, benchmarking, and multi-UAV optimal path planning. *Drones*, *7*(8). https://doi.org/10.3390/drones7080487

Liu, X., Wei, F., & Zhang, G. (2022). Uncertainty optimization design of airfoil based on adaptive point adding strategy. *Aerospace Science and Technology*, *130*. https://doi.org/10.1016/j.ast.2022.107875

Sudwoj, M. (2020). *Rust programming language in the high-performance computing environment* [PhD thesis]. https://doi.org/10.3929/ethz-b-000474922

Ugray, Z., Lasdon, L., Plummer, J., Glover, F., Kelly, J., & Martí, R. (2007). Scatter search and local NLP solvers: A multistart framework for global optimization. *INFORMS Journal on Computing*, *19*(3), 328–340. https://doi.org/10.1287/ijoc.1060.0175