OPTIMUS: A Multidimensional Global Optimization Package

Ioannis G. Tsoulos[1][*][¶], Vasileios Charilogis[1][*], Glykeria Kyrou[1][*], Vasileios N. Stavrou[2][*], and Alexandros Tzallas[1][*]

1 Department of Informatics and Telecommunications, University of Ioannina, Greece **2** Division of Physical Sciences, Hellenic Naval Academy, Greece **¶** Corresponding author ***** These authors contributed equally.

Summary

The location of global minima is applied in a variety of problems derived from physics (Duan et al., 1992), chemistry (Liwo et al., 1999), medicine (Lee, 2007) etc. Also, global optimization can be used in complex problems, such as the training of neural networks (Chiroma et al., 2017). Methods used in the global optimization process can be divided into stochastic and deterministic and a comparison between methods of these categories can be found in the work of Sergeyev et al.(Sergeyev et al., 2018). This article proposes a software that incorporates a variety of stochastic methods to handle the optimization of multidimensional functions.

Statement of need

This paper presents the GlobalOptimus software, written in ANSI C++, which allows researchers wanting to solve optimization problems in their domain to formulate their optimization problems and apply different global optimization methods to identify the global minimum of the problem. The main features of the proposed software are: a) Coding of the objective problem in a high level language such as ANSI C++ b) Incorporation of many global optimization techniques to solve the objective problem c) Parameterization of global optimization methods using user-defined parameters d) Usage of a GUI application to control the optimization strategy. Some of the frequently used optimization tools are MATLAB Optimization Toolbox, SciPy optimization module and Gurobi. MATLAB and SciPy are usually based on high-level languages, which can limit performance when computational problems are required. GlobalOptimus is an algorithmic optimization tool written in C++, which combines high performance with ease of parameterization for solving optimization problems in various domains. Unlike other tools, it stands out for its open architecture and adaptability, allowing full customization and integration of software from other platforms without restrictions. In addition, GlobalOptimus supports multiple optimization algorithms, such as genetic algorithms and differential evolution, giving flexibility in formulating the optimal method for each problem. Although both MATLAB and SciPy offer support for such algorithms, GlobalOptimus provides one extensible approach. Finally, Gurobi is an extremely powerful tool for mathematical optimization, but it requires a commercial license. Instead, GlobalOptimus is open source, which makes it more accessible to researchers and developers who wish to adapt it to their needs.

Usage of the Software

The software is compiled using the QT library, available from Qt main url The researcher should code their objective function and a number of other mandatory functions in the C++

---

programming language, specifying the dimension of the objective function and the bounds of the function domain. Subsequently, they can select a global optimization method to apply to the problem from a wide range of available methods.

In addition, the user can create his own objective function by implementing the methods of category UserProblem, the methods of which are presented in the following diagram.
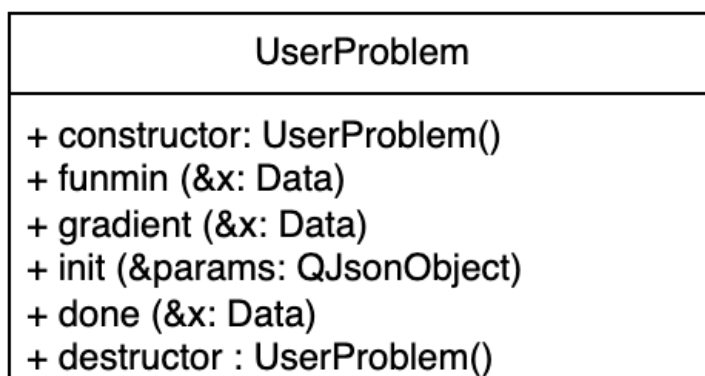
```
┌─────────────────────────────────────┐
│             UserProblem             │
├─────────────────────────────────────┤
│ + constructor: UserProblem()        │
│ + funmin (&x: Data)                 │
│ + gradient (&x: Data)               │
│ + init (&params: QJsonObject)       │
│ + done (&x: Data)                   │
│ + destructor : UserProblem()        │
└─────────────────────────────────────┘
```

Figure 1: The diagram of the user defined problem UserProblem

Implemented global optimization methods

In the proposed software, each implemented global optimization method has a set of parameters that can determine the overall optimization path and the effectiveness of the method. Some of the global optimization methods are as follows:

METHOD	DESCRIPTION
Differential Evolution	An evolutionary algorithm used in many areas (Pant et al., 2020)
Parallel Differential Evolution	A parallel version of the Differential Evolution method (Charilogis & Tsoulos, 2023).
Double precision genetic algorithm	An Improved version of the Genetic Algorithm (Tsoulos, 2008).
Improved Particle Swarm Optimization	Suggested as an improvement of the PSO method (Charilogis & Tsoulos, 2022).
Multistart	Initiates local searches from different start points (Martí et al., 2010).
NeuralMinimizer	Utilizes Radial Basis Functions (RBF) networks to increase the speed of the Multistart method (Tsoulos et al., 2023).
Parallel Particle Swarm optimizer	A novell parallel PSO variant, published recently (Charilogis et al., 2023).
Simulated annealing optimizer	included in the software under the name Simman (Kirkpatrick et al., 1983).
The optimal foraging algorithm (OFA)	motivated by animal behavioral ecology included the software named Ofa (Kyrou et al., 2024a).
Bio-inspired metaheuristic algorithm Giant Armadillo Optimization (GAO)	This process mimics the natural behavior of the giant armadillo (Kyrou et al., 2024b).
The Gray Wolf Optimizer (GWO)	This process mimics hunting mechanism of gray wolves in nature (Li et al., 2021).

Implemented local optimization methods

The parameter `--opt_localsearch` is used to select the used local optimization procedure. The implemented local optimization methods are the following:

METHOD	DESCRIPTION
The bfgs method	The Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm was implemented (Powell, 1989).
The lbfgs method	The limited memory BFGS method is implemented as an approximation of the BFGS method using a limited amount of computer memory (Liu & Nocedal, 1989).
The Gradient descent method	This method is denoted as gradient in the software and implements the Gradient Descent local optimization procedure (Amari, 1993).
The Nelder Mead method	The Nelder Mead simplex procedure for local optimization is also included in the software (Olsson & Nelson, 1975).
The adam method	The adam optimizer (Kingma, 2014).

Implementing a user-defined optimization method

The software can be extended by implementing custom optimization techniques by extending the optimization method class `UserMethod` and implementing the provided functions according to the requirements of the method.

Method	DESCRIPTION
init()	This function is called every time the optimization method starts
step()	This function implements the actual step of the optimization method.
terminated()	This function is used as the termination step of the optimization method.
done()	This function will be called when the optimization method terminates.
~UserMethod()	This is the destructor of the optimization method.

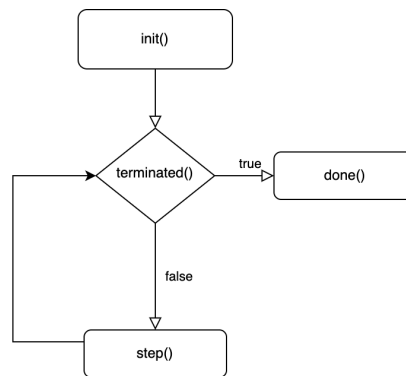A flowchart of any used optimization process provided below:

Figure 2: The diagram of the optimization process

An example run for the Rastrigin function and the Genetic algorithm is outlined in Figure 3.

Figure 3: An example run for the Rastrigin function

Conclusions

In this work, an environment for executing global optimization problems was presented, where the user can implement their optimization problem using a set of predefined functions and then has the possibility to choose among several global optimization methods to solve the defined problem. In addition, it is given the possibility to choose to use a local optimization method to enhance the reliability of the produced results. This programming environment is freely available and easy to extend to accommodate more global optimization techniques.

References

Amari, S. (1993). Backpropagation and stochastic gradient descent method. *Neurocomputing*, *5*(4-5), 185–196. https://doi.org/10.1016/0925-2312(93)90006-o

Charilogis, V., & Tsoulos, I. G. (2022). Toward an ideal particle swarm optimizer for multidimensional functions. *Information*, *13*(5), 217. https://doi.org/10.3390/info13050217

Charilogis, V., & Tsoulos, I. G. (2023). A parallel implementation of the differential evolution method. *Analytics*, *2*(1), 17–30. https://doi.org/10.3390/analytics2010002

Charilogis, V., Tsoulos, I. G., & Tzallas, A. (2023). An improved parallel particle swarm optimization. *SN Computer Science*, *4*(6), 766. https://doi.org/10.1007/s42979-023-02227-9

Chiroma, H., Noor, A. S. M., Abdulkareem, S., Abubakar, A. I., Hermawan, A., Qin, H., Hamza, M. F., & Herawan, T. (2017). Neural networks optimization through genetic algorithm searches: A review. *Appl. Math. Inf. Sci*, *11*(6), 1543–1564. https://doi.org/10.18576/amis/110602

Duan, Q., Sorooshian, S., & Gupta, V. (1992). Effective and efficient global optimization for conceptual rainfall-runoff models. *Water Resources Research*, *28*(4), 1015–1031. https://doi.org/10.1029/91wr02985

Kingma, D. P. (2014). Adam: A method for stochastic optimization. *arXiv Preprint arXiv:1412.6980*. https://doi.org/10.48550/arXiv.1412.6980

Kirkpatrick, S., Gelatt Jr, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680. https://doi.org/10.1126/science.220.4598.671

Kyrou, G., Charilogis, V., & Tsoulos, I. G. (2024a). EOFA: An extended version of the optimal foraging algorithm for global optimization problems. *Computation*, *12*(8), 158. https://doi.org/10.3390/computation12080158

Kyrou, G., Charilogis, V., & Tsoulos, I. G. (2024b). Improving the giant-armadillo optimization method. *Analytics*, *3*(2), 225–240. https://doi.org/10.20944/preprints202404.1784.v1

Lee, E. K. (2007). Large-scale optimization-based classification models in medicine and biology. *Annals of Biomedical Engineering*, *35*, 1095–1109. https://doi.org/10.1007/s10439-007-9317-7

Li, Y., Lin, X., & Liu, J. (2021). An improved gray wolf optimization algorithm to solve engineering problems. *Sustainability*, *13*(6), 3208. https://doi.org/10.3390/su13063208

Liu, D. C., & Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, *45*(1), 503–528. https://doi.org/10.1007/bf01589116

Liwo, A., Lee, J., Ripoll, D. R., Pillardy, J., & Scheraga, H. A. (1999). Protein structure prediction by global optimization of a potential energy function. *Proceedings of the National Academy of Sciences*, *96*(10), 5482–5485. https://doi.org/10.1073/pnas.96.10.5482

Martí, R., Moreno-Vega, J. M., & Duarte, A. (2010). Advanced multi-start methods. In *Handbook of metaheuristics* (pp. 265–281). Springer. https://doi.org/10.1007/978-1-4419-1665-5_9

Olsson, D. M., & Nelson, L. S. (1975). The nelder-mead simplex procedure for function minimization. *Technometrics*, *17*(1), 45–51. https://doi.org/10.2307/1267998

Pant, M., Zaheer, H., Garcia-Hernandez, L., Abraham, A., & others. (2020). Differential evolution: A review of more than two decades of research. *Engineering Applications of Artificial Intelligence*, *90*, 103479. https://doi.org/10.1016/j.engappai.2020.103479

Powell, M. (1989). A tolerant algorithm for linearly constrained optimization calculations. *Mathematical Programming*, *45*, 547–566. https://doi.org/10.1007/bf01589118

Sergeyev, Y. D., Kvasov, D., & Mukhametzhanov, M. (2018). On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. *Scientific Reports*, *8*(1), 453. https://doi.org/10.1038/s41598-017-18940-4

Tsoulos, I. G. (2008). Modifications of real code genetic algorithm for global optimization. *Applied Mathematics and Computation*, *203*(2), 598–607. https://doi.org/10.1016/j.amc.

2008.05.005

Tsoulos, I. G., Tzallas, A., Karvounis, E., & Tsalikakis, D. (2023). NeuralMinimizer: A novel method for global optimization. *Information*, *14*(2), 66. https://doi.org/10.3390/info14020066