applied sciences

MDPI

Article

Supplementary Materials: GENERAL PURPOSE OPTIMIZATION LIBRARY (GPOL): A FLEXIBLE AND EFFICIENT MULTI-PURPOSE OPTIMIZATION LIBRARY IN PYTHON.

Illya Bakurov [1]*, Marco Buzzelli [2], Mauro Castelli [1], Leonardo Vanneschi [1], and Raimondo Schettini [2]

1 Nova Information Management School (NOVA IMS), Universidade NOVA de Lisboa, Campus de Campolide, 1070-312, Lisboa, Portugal. {ibakurov,mcastelli,lvanneschi}@novaims.unl.pt
2 Università degli Studi di Milano-Bicocca, Dipartimento di Informatica Sistemistica e Comunicazione, Viale Sarca, 336, 20126 Milano, Italia. {marco.buzzelli,raimondo.schettini}@unimib.it
* Correspondence: ibakurov@novaims.unl.pt

check for updates

Citation: Bakurov, I.; Buzzelli, M.; Castelli, M.; Vanneschi, L.; Schettini, R. General Purpose Optimization Library (GPOL): A Flexible and Efficient Multi-Purpose Optimization Library in Python. *Appl. Sci.* **2021**, *11*, 4774. https://doi.org/10.3390/app11114774

Received: 30 March 2021
Accepted: 15 May 2021
Published: 23 May 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

1. Experimental verification

In this section, we study GPOL's algorithms accuracy on a varied set of conceptually distinct problems. Every subsequent subsection regards one of the three main problem types considered in GPOL: continuous, combinatorial and SML (Sections 1.1, 1.2 and 1.3, respectively).

1.1. Continuous optimization problems

To study the algorithms' performance on continuous problems, we considered four optimization functions commonly used in the scientific community for algorithms' assessment [1–4]: Rosenbrock, Rastgrigin, Ackley, and De Jong's spherical function. In our experiments, each problem was studied with $D = 2$ and $D = 30$, and the global minimum for all the 4 functions is 0 (at each dimension). The corresponding D-dimensional formulations are as follows:

$$f_{Rosenbrock}(x) = \sum_{i=1}^{D-1}\left(b(x_{i+1} - x_i^2)^2 + (a - x_i)^2\right) \tag{1}$$

with $a = 1$ and $b = 100$, and $-2.048 \leq x_i \leq 2.048$.

$$f_{Rastrigin}(x) = A \cdot D + \sum_{i=1}^{D}\left(x_i^2 - A\cos(2\pi x_i)\right) \tag{2}$$

with $A = 10$ and $-5.12 \leq x_i \leq 5.12$.

$$f_{Ackley}(x) = -a\exp\left(-b\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(cx_i)\right) + a + \exp(1) \tag{3}$$

with $a = 20$, $b = 0.2$, $c = 2\pi$, and $-32.768 \leq x_i \leq 32.768$.

$$f_{Sphere}(x) = \sum_{i=1}^{D}x_i^2 \tag{4}$$

with $-5.12 \leq x_i \leq 5.12$.

Several algorithms were considered in the study: RS (which serves as a baseline), HC, SA, GA, DE, S-PSO and A-PSO. The initial candidate solutions in each algorithm were generated under the continuous uniform distribution, with parameters corresponding to the constraints of the underlying problem. The neighborhood/mutation function was the ball mutation with a fixed radius of 0.3, and a probability of mutating a solution's representation a given index of 0.5 and 0.2 for $D = 2$ and $D = 30$, respectively. The GA was studied with tournament selection, whose pressure was set to 0.08, and geometric crossover; the probabilities of applying the crossover and the mutation operators were set to 0.7 and 0.3 respectively; elitism and reproduction were allowed. The PSO was studied in both synchronous and asynchronous variants (S-PSO and A-PSO, respectively). Both variants shared the same parameters. The learning factors that control the effect of the social and cognitive influence on a particle (C_1 and C_2, respectively), were set to 2. A time-decreasing inertia weight (w) was used in range $[0.4, 0.9]$. The velocity vector was clamped at ± 3.0. The DE was studied in the configuration $DE/best/2/bin$ which corresponds to $v_i = x_{best} + F_1(x_{r1} - x_{r2}) + F_2(x_{r3} - x_{r4})$, where x_{best} is the current best solution, F_1 and F_2 are the corresponding mutation factors with values $F_1 = F_2 = 0.7$, and the crossover is binomial.

All the algorithms were executed for 100 iterations with a neighborhood/population/swarm size of 50 solutions; the only exception was RS, which was run for an equivalent total number of fitness function evaluations (100x50). Performances were reported after 30 independent runs. Figure 1 shows the cross-run average fitness during various iterations, and Figure 2 illustrates the fitness distribution for the last generation of each run. The x axis of in the Figure 1 enumerates successive iterations of the experiment, whereas the y axis represents the respective averaged (across 30 runs) fitness values; the last row of the lattice-plot provides the title for the x label, that is shared by the sub-plots in above; at the top of each sub-figure, problems' names and dimensionalities are reported. The x axis of in the Figure 2 report different metaheuristics, whereas the y axis represents the respective fitness values; at the top of each sub-figure, problem's name and dimensionality are reported. All the problems are minimization problems, and the fitness value corresponds to the function evaluation itself. It is possible to observe that A-PSO performs consistently better than all other algorithms. Most of them outperform the results obtained with RS, with the exception of HC and SA for the 2D versions of Ackley and Rastrigin. Multiple explanations could describe this behavior, namely the "bad" initialization, propensity of the LS algorithms to get stuck on locally-optimal solutions, sub-optimal hyper-parameters, and mere unfitness of the specific algorithms for these functions. Nonetheless, the same algorithms effectively outperform RS on the 30D versions of the Ackley and Rastrigin, along with other problems and dimensions, thus suggesting a correct implementation.

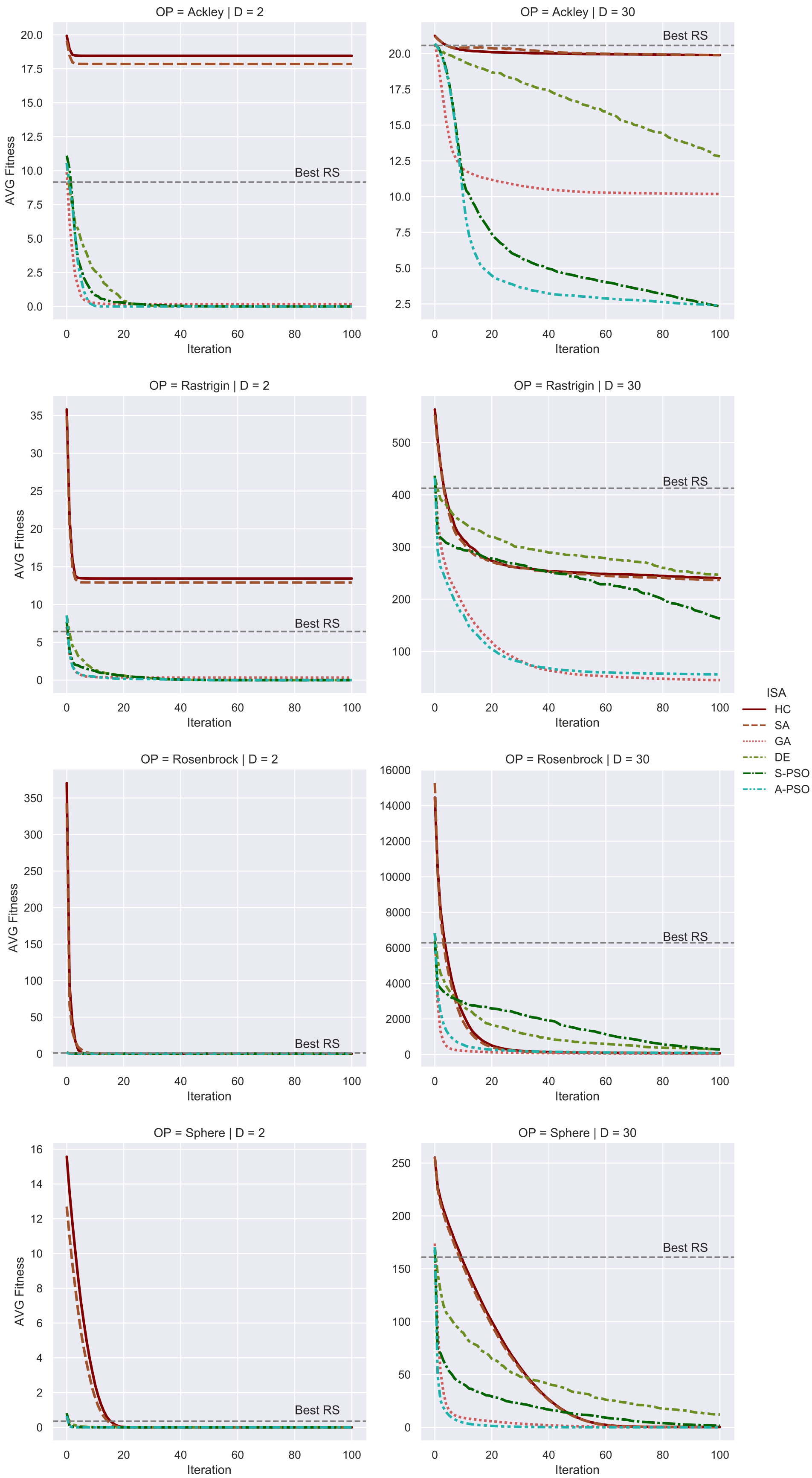

Figure 1. Comparative analysis of algorithms' performance on four common continuous optimization problems. Cross-run average fitness during various iterations.

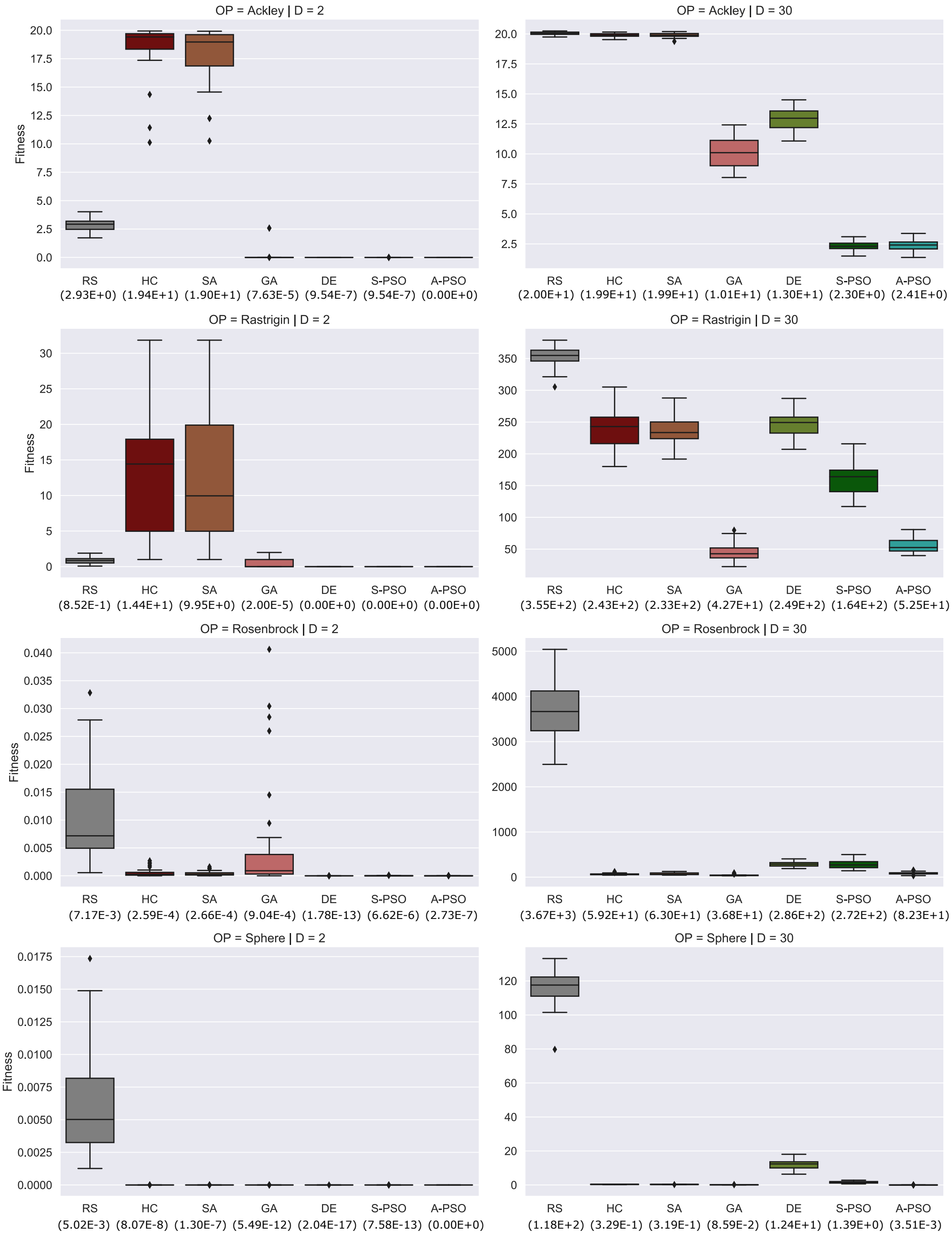

Figure 2. Comparative analysis of algorithms' performance on four common continuous optimization problems. Fitness distribution for the last generation of each run, median value in parentheses.

1.2. Combinatorial optimization problems

We continued our study with the combinatorial problems, namely the Travelling Salesman Problem (TSP), Binary Knapsack (also referred to as 0-1 Knapsack), and Bounded Knapsack. TSP was formulated in a 13-city instance of the problem, adopting the same distance matrix as used in [5]. Both 0-1 Knapsack and Bounded Knapsack refer to random problem instances with 100 items and a total capacity of 1000. The Bounded Knapsack had a maximum item repetition set to 4 items.

The study comprised RS, HC, SA, and GA. Other algorithms, due to their nature and the underlying problem types, could not be used in this study. The initial candidate solutions in each algorithm were generated at random, according to the problems' search space. For the 0-1 and Bounded Knapsack problems, the initial solutions were generated under discrete uniform with parameters corresponding to the constraints of the underlying problem. For the TSP problem, the initial solutions are random permutations of cities. We used the swap, bit-flip and discrete-range neighborhood/mutation functions for the TSP, 0-1 and Bounded Knapsack problem instances, respectively; the probability of applying the operator at a given index of the solution's representation was set to 0.2. The GA was studied with tournament selection, whose pressure was set to 0.08. When solving the 0-1 and Bounded Knapsack problem instances, the one-point crossover was used; whereas, when solving the TSP instance, partially-mapped crossover was applied. The probabilities of applying the crossover and the mutation operators were set to 0.7 and 0.3 respectively; elitism and reproduction were allowed.

All the algorithms were executed for 100 iterations with a neighborhood/population size of 500 individuals; the only exception was RS, which was run for an equivalent total number of fitness function evaluations (100x500). Performances were reported after 30 independent runs. Figure 3 shows the cross-run average fitness on various iterations, obtained from the current-best solution, and Figure 4 illustrates the fitness distribution for the best-found solution on the generation of each run. Note that with the adopted standard formulation, TSP is a minimization problem, while the two Knapsack variants are maximization problems. Although no individual algorithm appears to have a definitive dominance across problems, all of them outperform, as expected, the results obtained with RS.

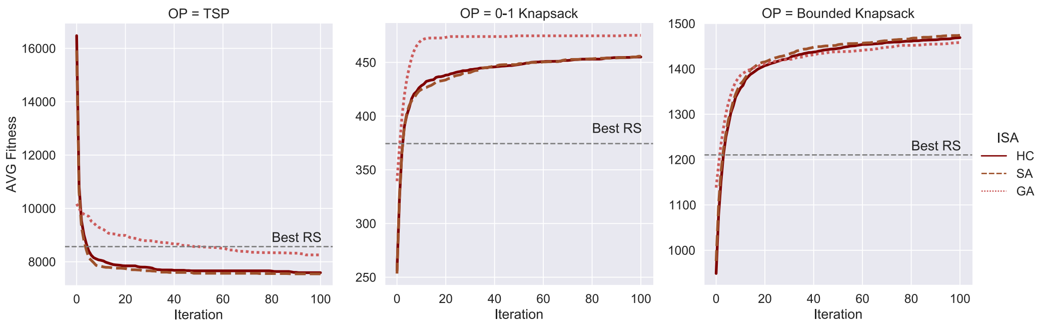

Figure 3. Comparative analysis of algorithms' performance on three common combinatorial optimization problems. Cross-run average fitness during various iterations.

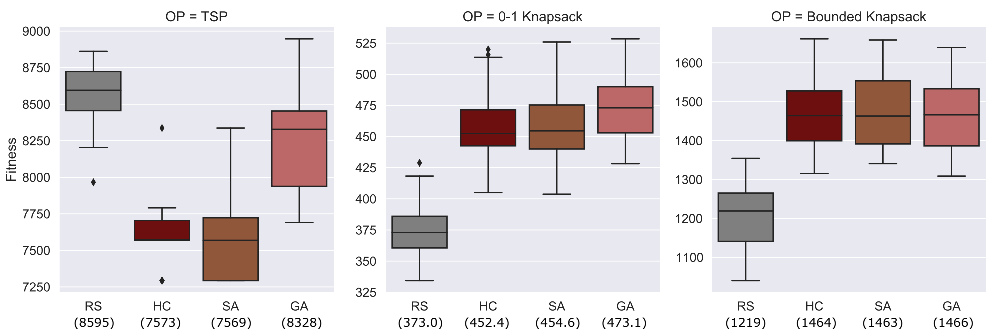

Figure 4. Comparative analysis of algorithms' performance on three common combinatorial optimization problems. Fitness distribution for the last generation of each run, median value in parentheses.

1.3. Supervised Machine Learning problems

We conclude our study with SML problems, addressed from the perspective of inductive programming [6,7]. Specifically, we focused on the Boston regression problem and on the Breast Cancer classification problem: two popular benchmark problems in the scientific community [8]. The Boston regression problem stems from the Harrison and Rubinfeld housing dataset [9], which contains 506 instances of Boston houses described by 13 numeric or categorical attributes. The target is predicting the value (price) of the dataset houses.

The Breast Cancer classification problem is related to a dataset first addressed by Street et al. [10], and the target is to assign a "benign" or "malignant" label to 569 samples of breast exams. These are described by 30-dimensional precomputed features, extracted from from a digitized image of a fine needle aspirate (FNA) of the breast mass.

In addition to the baseline RS, our study involved the standard GP, GSGP, as well as HC and SA applied to the tree-based representations (HC-GP and SA-GP, respectively)

The initial candidate solutions in each algorithm were generated at random, according to the problems' search space. For the LS algorithms, grow initialization with a maximum initial depth of five was used; the PB algorithms were initialized with RHH algorithms (also with a maximum depth of five levels). The neighborhood/mutation function was the sub-tree mutation; in the case of GSGP, the so-called geometric semantic mutation was used, with a maximum mutation step of 5. The both GP and GSGP were studied with tournament selection, whose pressure was set to 0.08. For GP, swap crossover was used. The probabilities of applying the crossover and the mutation operators were set to 0.7 and 0.3 for GP; for GSGP, only the mutation was used.

All the algorithms were executed for 30 iterations with a neighborhood/population size of 500 individuals; the only exception was RS, which was run for an equivalent total number of fitness function evaluations (30x500). Performances were reported after 30 independent runs. The fitness function used for both problems is root-mean-square error (RMSE), computed as the difference between the predicted "probabilities" and the binary target in the case of binary classification.

Coherently with the analysis conducted for continuous and combinatorial optimization problems, Figures 5 and 6 clearly highlight the improvement over the RS.

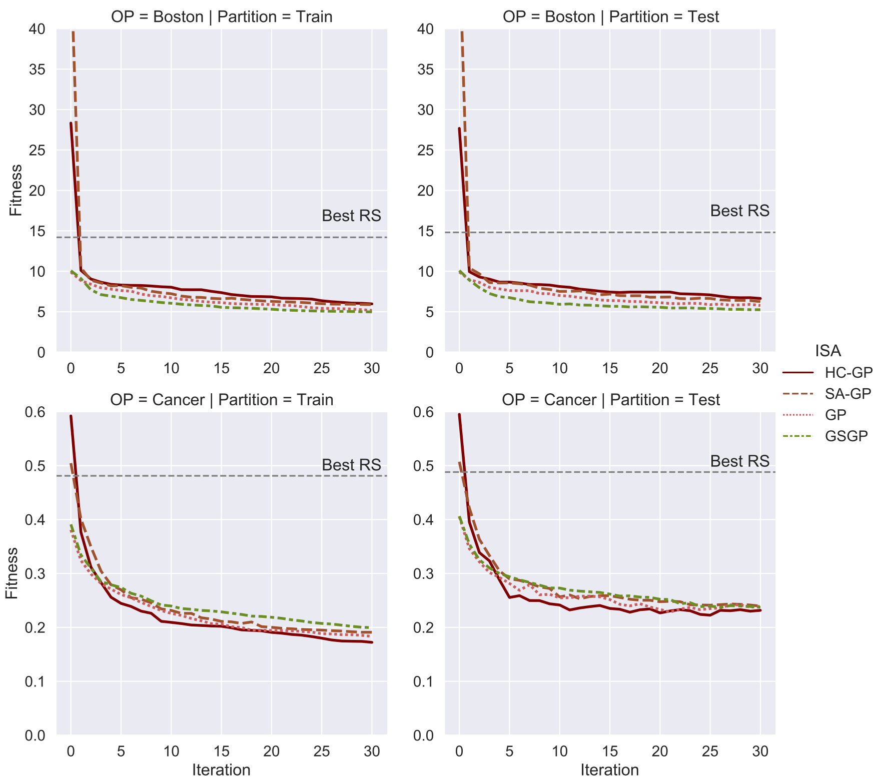

Figure 5. Comparative analysis of algorithms' performance on two supervised machine learning problems. Cross-run average fitness during various iterations.

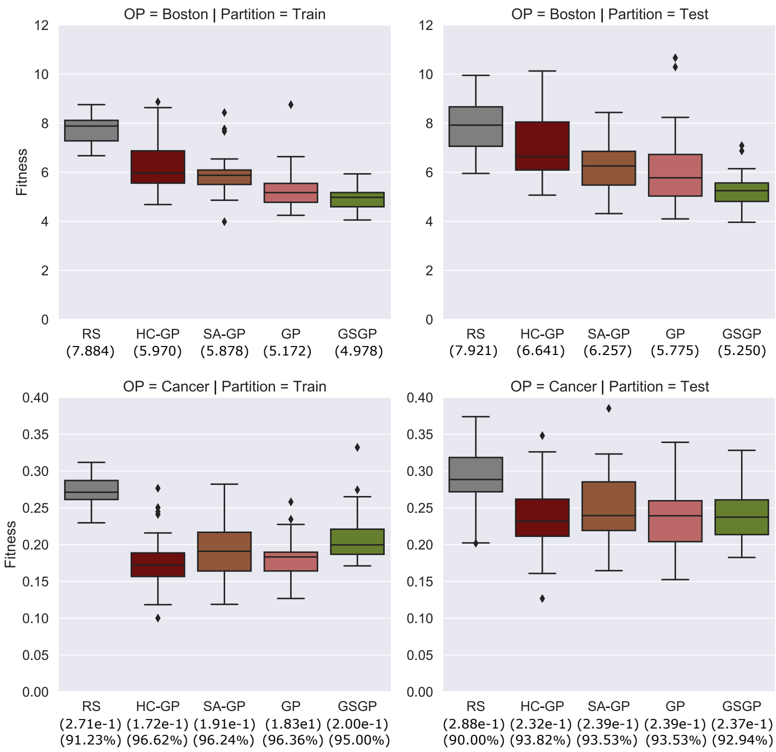

Figure 6. Comparative analysis of algorithms' performance on two supervised machine learning problems. Fitness distribution for the last generation of each run, median fitness value and median classification accuracy in parentheses.

References

1. Shi, Y.; Eberhart, R. A modified particle swarm optimizer. *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)* **1998**, pp. 69–73.
2. Das, S.; Suganthan, P. Differential Evolution: A Survey of the State-of-the-Art. *IEEE Trans. Evolutionary Computation* **2011**, *15*, 4–31.
3. Liang, J.J.; Qu, B.Y.; Suganthan, P.N. Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore* **2014**, *635*, 490.
4. Aziz, N.; Mubin, M.; Mohamad, M.; Ab. Aziz, K. A Synchronous-Asynchronous Particle Swarm Optimisation Algorithm. *TheScientificWorldJournal* **2014**, *2014*, 123019. doi:10.1155/2014/123019.
5. Traveling Salesman Problem | OR-Tools | Google Developers, Accessed on 16 February 2021. https://developers.google.com/optimization/routing/tsp.
6. Kitzelmann, E.; Schmid, U. Inductive Synthesis of Functional Programs: An Explanation Based Generalization Approach. *Journal of Machine Learning Research* **2006**, *7*, 429–454.
7. Schmid, U. *Inductive Synthesis of Functional Programs, Universal Planning, Folding of Finite Programs, and Schema Abstraction by Analogical Reasoning*; Vol. 2654, Springer Science & Business Media, 2003. doi:10.1007/b12055.
8. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **2011**, *12*, 2825–2830.
9. Harrison, D.; Rubinfeld, D.L. Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management* **1978**, *5*, 81–102.
10. Street, W.N.; Wolberg, W.H.; Mangasarian, O.L. Nuclear feature extraction for breast tumor diagnosis. Biomedical image processing and biomedical visualization. International Society for Optics and Photonics, 1993, Vol. 1905, pp. 861–870.