Comparison of Multistart Global Optimization Algorithms on the BBOB Noiseless Testbed

László Pál
Sapientia - Hungarian University of Transylvania
530104 Miercurea-Ciuc, Piata Libertatii, Nr. 1, Romania
pallaszlo@sapientia.siculorum.ro

ABSTRACT

Multi Level Single Linkage is a multistart, stochastic global optimization method which relies on random sampling and local search. In this paper, we benchmarked three variants of the MLSL algorithm by using two gradient based and a derivative-free local search method on the noiseless function testbed. The three methods were also compared with a commercial multistart solver, called OQNLP (OptQuest/NLP).

Our experiment showed that, the results may be influenced essentially by the applied local search procedure. Depending on the type of the problem the gradient based local search methods are faster in the initial stage of the optimization, while the derivative-free method show a superior performance in the final phase for moderate dimensions. Considering the percentage of the solved problems, OQNLP is similar or even better (for multi-modal and weakly structured functions) in 5-D than the MLSL method equipped with the gradient type local search methods, while on 20-D the latter algorithms are usually more faster.

Categories and Subject Descriptors

G.1.6 [Numerical Analysis]: Optimization—global optimization, unconstrained optimization; F.2.1 [Analysis of Algorithms and Problem Complexity]: Numerical Algorithms and Problems

General Terms

Algorithms

Keywords

Benchmarking, Black-box optimization, Multi level methods, Multi start heuristic, Scatter search

1. INTRODUCTION

Multistart global optimization algorithms were introduced in the 1980s for bound constrained optimization problems.

Two important multistart type methods are the Clustering [1] and Multi Level Single Linkage (MLSL) [9] algorithms. The basic idea behind these methods is to form groups (clusters) of points around the local minimizers from a uniform sampled domain and start local searches no more than once in each of those groups.

The aim of the paper is to compare three variants of the MLSL method using the COCO framework [3] with OQNLP (OptQuest/NLP) [11], an other well known commercial multistart type algorithm.

The rest of this article is organized as follows. Section 2 reviews the MLSL and OQNLP algorithms. In Section 3, we describe the experiment procedure together with the algorithms parameter settings. The results are presented in Section 4 and discussed in Section 5. Section 6 concludes the paper and points out some directions for future work.

2. ALGORITHMS

Multi Level Single Linkage (MLSL) has two phases: a global and a local one. The global phase consists of sampling, while the local phase is based on local searches. The local minimizer points are found by means of a local search procedure (LS), starting from appropriately chosen points from the sample drawn uniformly within the set of feasibility. A local search procedure is applied to every sample point from the reduced sample, except if there is another sample point within some critical distance \( r_k \) (defined in [9]), which has a lower function value (see Algorithm 1). The reduced sample consists of the \( \gamma k N \) best points (0<\( \gamma \) ≤ 1) from the cumulated sample \( x_1, \ldots, x_{kN} \).

The local search method is an essential part of the MLSL. Depending on the applied local search procedure the quality of the found solution may vary significantly. Thus in this study we tested three MLSL variants by applying two gradient based and a derivative-free local search method (more details in Sec. 3).

OQNLP is a solver designed to find global optima of smooth constrained nonlinear problems. It is a multistart heuristic method which runs a local search from a variety of starting points in order to find a global minimum, or multiple local minima. The solver uses a scatter-search mechanism for generating start points. The solver steps are presented in the Algorithm 2. After an initial call to LS at the user-provided initial point, \( x_0 \), \( N_1 \) trial points are generated (Stage 1). The best point is chosen as the starting point for the next call to LS. In Stage 2, \( N_2 \) iterations are performed in which candidate starting points are generated and LS is
The smallest local minimum value found.

\begin{algorithm}
X' \leftarrow \varnothing; k \leftarrow 0
2 repeat
3 \quad k \leftarrow k + 1
4 \quad Generate N points \(x_{(k-1)N+1}, \ldots, x_{kN}\) with uniform distribution on \(X\).
5 \quad Determine the reduced sample \((X_r)\) consisting of the \(\gamma kN\) best points from the cumulated sample \(x_1, \ldots, x_{kN}\).
6 \quad for i \leftarrow 1 to length\((X_r)\) do
7 \quad \quad if NOT \((\text{there is such a } j \text{ that } f(x_j) < f(x_i) \text{ and } \|x_j - x_i\| < r_k)\) then
8 \quad \quad \quad \quad \quad \text{Start a local search method \(LS\) from } x_i.
9 \quad \quad \quad \quad \quad X'^* \leftarrow LS(x_i)
10 \quad \quad \quad \quad \quad X^* \leftarrow X^* \cup \{x^*\}
11 \quad until \text{Some global stopping rule is satisfied.}
12 \text{return The smallest local minimum value found.}
\end{algorithm}

MSL has four parameters to set: the number of sample points in an iteration, the size of the reduced sample, the maximum number of function evaluations for local search, and the used local search procedure. The sample was generated from a Sobol quasi-random sequence \([6]\) and its size was set to 50D. From the actual sample only the best 5D points are considered for further analysis.

We benchmarked three variants of the MSL algorithm by using 2 gradient type and a derivative-free local search method. The gradient based methods are the following: a quasi-Newton type \(fminunc\) and an interior point \(fmincon\) procedure from MATLAB. The first is a well-known quasi-Newton method which approximates the Hessian by the BFGS formula, while the second is an interior-point algorithm for constrained nonlinear problems. The third local search algorithm is the Nelder-Mead \([7]\) simplex method which belongs to the class of direct search methods. All three versions of the algorithms were run on the whole testbed in all dimensions. The maximum number of function evaluations for local search was set to 10\% of the total budget while the termination tolerance parameter value was set to \(10^{-12}\).

In the case of the OQNLP method, we used the default parameters (see in [5]) except the iteration limit which was set to 300D. Using this limit we get approximately the same maximal budget as in the case of MSL. Furthermore it is important that OQNLP changes its search strategy depending on the iteration limit. The local search used by OQNLP is the LSGRG2, a generalized gradient projection method.

4. RESULTS

Results from experiments according to [3] on the benchmark functions given in [2, 4] are presented in Figures 1, 2 and 3 and in Tables 1 and 2. The expected running time (ERT), used in the figures and table, depends on a given target function value, \(f_t = f_{opt} + \Delta f\), and is computed over all relevant trials as the number of function evaluations executed during each trial while the best function value did not reach \(f_t\), summed over all trials and divided by the number of trials that actually reached \(f_t\) \([3, 8]\). Statistical significance is tested with the rank-sum test for a given target \(\Delta f_t\) \((10^{-8})\) as in Figure 1 using, for each trial, either the number of needed function evaluations to reach \(\Delta f_t\) (inverted and multiplied by \(-1\)), or, if the target was not reached, the best \(\Delta f\)-value achieved, measured only up to the smallest number of overall function evaluations for any unsuccessful trial under consideration.

4.1 CPU Timing Experiments

The timing experiments were carried out with \(f_5\) on a machine with Intel Dual-Core processor, 2.6 Ghz, with 2 GB RAM, on Windows 7 64bit in MATLAB R2011b 64bit. The average time per function evaluation in 2, 3, 5, 10, 20, 40 dimensions was about 13, 9.4, 7.1, 5.2, 3.9, 3.7 \(\times 10^{-4}\) s for \(fmincon\), about 6.1, 5.5, 3.9, 3.1, 2.9, 2.7 \(\times 10^{-4}\) s for \(fminunc\), about 4.5, 3.3, 2.9, 3.3, 4.6, 8.8 \(\times 10^{-4}\) s for \(simplex\), and about 8.1, 7.6, 5.7, 4.1, 3.9, 3.1 \(\times 10^{-4}\) s for OQNLP.

5. DISCUSSION

Although the MSL method cannot find the final solution in many cases, our aim was to reveal the differences between
the applied local search methods during the different stages of the optimization.

Considering the ERT numbers in different dimensions, we can state that the gradient type methods are usually more faster than the simplex method. Nevertheless there are situations when the latter method is significantly better in lower dimensions than the other methods. Such cases can be observed on the $f_7, f_{10}, f_{11}, f_{13}, f_{14}, f_{16}$, and $f_{23}$ functions (see Figure 1). The OQNLP solver is faster than the MLSL method with \texttt{fmincon} and \texttt{fminunc} on the $f_3, f_7, f_{20}$, and $f_{24}$ functions. On $f_{24}$ OQNLP is even faster than the best BBOB-2009 algorithm for 2, 3, and 5 dimensions. Regarding the proportion of solved instances, the general aspect is that the gradient type methods are faster on the initial phase of the optimization, while the derivative-free simplex method provides a better performance in the final stage for 2, 3, and 5 dimensions.

Considering all functions aggregated in 5-D (see Figure 2), the proportion of the solved problems by the algorithms varies between 62% and 78%. \texttt{fminunc} is the fastest for \#FEs $< 100D$, while between 100D and 500D the \texttt{fmincon} solves the largest proportion of problems. After 1000D evaluations the simplex method becomes the leader by solving 78% of the problems up to the final budget. This behavior is more pronounced on the ill-conditioned functions sub-group. For \#FEs $< 200D$, the \texttt{fmincon} is the best algorithm solving more than 60% of the problems, followed by \texttt{fminunc}, OQNLP and simplex solving 50%, 45% and 8% of the problems. For \#FEs $> 700D$, the simplex becomes the best competitor by solving 100% of the problems up to the final budget. This huge progress is due to the robustness of the method on the $f_{10}, f_{11}, f_{12}, f_{13}$ and $f_{14}$ functions. The OQNLP is slightly faster than the simplex algorithm on the multi-modal and weakly structured functions. This behavior is caused by the success of the OQNLP method on the $f_{19}$ and $f_{24}$ functions.

In the 20-D space (see Figure 3), the previously observed advantageous properties of the simplex method cannot be further observed. The largest proportion of solved problems by simplex is about 22% on the separable functions sub-group, while on the moderate group is the lowest (about 2%). Considering all functions aggregated, \texttt{fmincon} is the fastest by solving about 58% of the problems, followed by \texttt{fminunc}, OQNLP, and simplex solving 55%, 50% and 15% of the problems. The previous ranking of the algorithms can be observed for the other function groups too.

As a feature work we propose a strategy which tries to automatically select the best local search algorithm during the optimization.

### Acknowledgements

This work was supported by the Sapientia Foundation - Institute for Scientific Research with the grant No. 101/9/2013.

### 7. REFERENCES


Figure 1: Expected running time (ERT in number of \( f \)-evaluations) divided by dimension for target function value \( 10^{-8} \) as \( \log_{10} \) values versus dimension. Different symbols correspond to different algorithms given in the legend of \( f_1 \) and \( f_2 \). Light symbols give the maximum number of function evaluations from the longest trial divided by dimension. Horizontal lines give linear scaling, slanted dotted lines give quadratic scaling. Black stars indicate statistically better result compared to all other algorithms with \( p < 0.01 \) and Bonferroni correction number of dimensions (six). Legend: ○:fmincon, ▽:fminunc, ⊳:simplex, □:OQNLP
Figure 2: Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension (FEvals/D) for 50 targets in $10^{[-8..2]}$ for all functions and subgroups in 5-D. The “best 2009” line corresponds to the best ERT observed during BBOB 2009 for each single target.
Figure 3: Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension (FEvals/D) for 50 targets in $10^{1-20}$ for all functions and subgroups in 20-D. The “best 2009” line corresponds to the best ERT observed during BBOB 2009 for each single target.
<table>
<thead>
<tr>
<th></th>
<th>Δopt</th>
<th>5e-1</th>
<th>5e-2</th>
<th>5e-3</th>
<th>5e-4</th>
<th>5e-5</th>
<th>5e-7</th>
<th>5e-9</th>
<th>5e-11</th>
<th>5e-13</th>
<th>5e-15</th>
<th>5e-17</th>
<th>5e-19</th>
<th>5e-21</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>3.7c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
<td>3.1c</td>
</tr>
<tr>
<td>100</td>
<td>1.25c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
</tr>
<tr>
<td>1000</td>
<td>1.25c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
</tr>
<tr>
<td>10000</td>
<td>1.25c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
</tr>
<tr>
<td>100000</td>
<td>1.25c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
</tr>
<tr>
<td>1000000</td>
<td>1.25c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
<td>1.55c</td>
</tr>
</tbody>
</table>

Table 1: Expected running time (ERT in number of function evaluations) divided by the best exact ERT measured during BBOB-2009 (given in the respective first row) for different Δ of values in dimension 5. The central 80% range divided by two is given in braces. The median number of conducted function evaluations is additionally given in italics, if ERT(10^{-7}) = ∞. #succ is the number of trials that reached the final target \( f_{opt} + 10^{-8} \). Best results are printed in bold.
Table 2. Expected running time (EFT in number of function evaluations) divided by the respective best EFT measured during BBOB-2009 (given in the respective first row) for different $\Delta f$ values in dimension 20. The median number of conducted function evaluations is additionally given in brackets. If EFT(10^7) = $\infty$, #succ is the number of trials that reached the final target.