On the Experimental Investigation of Pareto-Lipschitzian Optimization

Jonas Mockus, Justas Stasionis

Vilnius University, Institute of Mathematics and Informatics
Akademijos 4, LT-08663

e-mail: jmockus@gmail.com

Abstract A well-known example of global optimization that provides solutions within fixed error limits is optimization of functions with a known Lipschitz constant. In many real-life problems this constant is unknown.
To address that, we propose a novel method called Pareto Lipschitzian Optimization (PLO) that provides solutions within fixed error limits for functions with unknown Lipschitz constants.In the proposed approach, a set of all unknown Lipschitz constants is regarded as multiple criteria using the concept of Pareto Optimality (PO).

Keywords: Pareto, Lipschitz, Global Optimization

Introduction

The target of this paper is to present results of experimental calculations. The theoretical part of PLO is described in [6]. Preliminary computing results are discussed in Mockus11. We compare PLO to the existing algorithm DIRECT in [3]. We show that in contrast to PLO, the DIRECT algorithm considers only a small subset of PO decisions that are selected by a heuristic rule depending on an adjustable parameter. It means that some PO decisions are preferred to others. In contrast, PLO regards all PO decisions without preferences and is naturally suited to utilize highly parallel computing.

1 Pareto-Lipschitzian Optimization (PLO)

We start to explain the optimization of Lipschitz functions with unknown constants by considering this one-dimensional example.
Suppose that the interval $D = [a, b] \in R$ is partitioned into intervals $[a_i, b_i],\ i = 1, ..., I$ of lengths $l_i = b_i - a_i$ with midpoints $c_i = (b_i + a_i)/2$ and the values of the function $f_\omega(x)$ are known only at the midpoints c_i. The unknown Lipschitz constants are regarded as different components of multiple criteria. The variables x are represented by the intervals $a_i \leq\ x \leq\ b_i$ and the function $f_\omega(x)$ is approximated by the lower bounds: $f_\omega(x) \geq f(c_i) - \omega\ l_i/2,\ a_i \leq\ x \leq\ b_i$. (1)

Definition 1: The interval $i : a_i \leq x \leq b_i$ belonging to a compact set $D \in R$ dominates the interval $j : a_j \leq x \leq b_j$ if

$$f(c_i) - \omega\ l_i/2 \quad \leq \quad f(c_j) - \omega\ l_j/2 \quad for\ all\ \omega \in \Omega \tag{1}$$

$$f(c_i) - \omega\ l_i/2 \quad < \quad f(c_j) - \omega\ l_j/2,\ for\ at\ least\ one\ \omega \in \Omega\ (2). \tag{2}$$

Expressions (1)(2) show that the lower bound of the interval i is increasing with $f(c_i)$ and decreasing with l_i for all ω.
In [7] the set of Pareto Optimal (PO) intervals is defined as follows:
Definition 2: The interval $j : a_j \leq z \leq b_j$ belonging to a compact set $D \in R$ is called Pareto Optimal (PO)[1] *if there is no dominant interval i defined by (1)(2).*
The PLO algorithm is based on the theorems 1 and 2 [6].

2 Comparison of PLO with the DIRECT algorithm

The DIRECT algorithm [3, 2] for Lipschitzian optimization with unknown constants is defined as a heuristic without any references to the theory of vector optimization or Pareto optimality. However, it can be explained in terms of PLO as well. The basic idea of DIRECT is to select (and sample within) all Potentially Optimal (PTO) intervals during an iteration. A formal definition of PTO intervals follows.
Definition 3: The interval j is said to be PTO if there exists some rate-of-change constant $\omega > 0$ such that

$$f(c_j) - \omega\ l_j/2 \quad \leq \quad f(c_i) - \omega\ l_i/2\ for\ all\ i = 1, ..., I, \tag{3}$$

$$f(c_j) - \omega\ l_j/2 \quad \leq \quad f_{min} - \epsilon\ |f_{min}|. \tag{4}$$

Here $\epsilon > 0$ is a constant that defines the size of the set of PTO intervals, and f_{min} is the current best value.
One can see that inequality (3) includes all the intervals which are among the best for at least one $\omega > 0$. It means that these intervals are not dominated, thus, they belong to the PO set as well. However, condition (3) defines only a subset of not dominated intervals. The further restriction of this part of PO intervals is provided by condition (4) which depends on the parameter ϵ. Thus the actual performance of DIRECT algorithm is determined by this parameter.

3 Extension to Several Dimensions

We define the length l_i of the interval $a_i^k \leq z_k \leq b_i^k,\ k = 1, ..., K$ in a compact subset $D \in R^K$ as the longest length: $l_i = max_k\ l_i^k$. where $l_i^k = b_i^k - a_i^k,\ k = 1, ..., K$. The observation points c_i are in the middle $c_i^k = (b_i^k + a_i^k)/2,\ k = 1, ..., K$. Then the Definition of PO-intervals remains the same.

[1] Here PO decisions are defined as indexes of PO intervals. In this expression the PO decisions are expressed as continuous variables x.

3.1 Sampling

The sampling procedure of PLO is similar to that of the DIRECT algorithm [2]. However, the definition of Pareto Optimal intervals, used in PLO, is different from the definition of Potentially Optimal intervals in the DIRECT algorithm.

3.2 Experimental Computing

Preliminary results comparing the sequential version of PLO with other methods are in [6]. These and other calculations can be repeated independently using any of the following websites:
`http://soften.ktu.lt/~mockus`,
`http://optimum2.mii.lt/`,
`http://prof.if.ktu.lt/~jonas.mockus`,
section 'Software Systems', task 'GMJ4: Global Optimization of many Models with PloN' , start "Applet long: gmj4.html", select method "PloNj", task "Xcos" e.t.c., click "Operations" and "Run". Three special multi-modal test functions of 20 variables, called "Xcos" (5), "Xcos2" (6), and "Xcos3" (7), respectively, are used.

$$\sum_{i,j=1}^{20} (x_i\ x_j + a\ cos\ x_i\ cos\ x_j),\ -10 \le x_i \le 10. \tag{5}$$

$$\sum_{i,j=1}^{20} (x_i\ x_j + a\ cos\ x_i\ cosx_j + a\ cos\ x_i\ x_j), \quad -10 \le x_i \le 10. \tag{6}$$

$$\sum_{i,j=1}^{20} (x_i\ x_j + a\ cos\ x_i\ cos\ x_j,\ -50 \le x_i) \le 50. \tag{7}$$

The multiplier a defines the proportion of multi-modal components and the domain determines the number of local minima. The minimum of the first function "Xcos" is zero, for simplicity.
In addition, three models that simulates real optimization problems [5] are investigated. The first one, called "Eco Duel", in short, it is about a differential game that represents the competition of two servers, using the concept of Nash equilibrium. Here eight optimization variables are used. The second five-dimensional model optimizes the "mixture" of five heuristics for packing rectangular boxes of different size into the container and is called "Packer", in short. The third model "ModelTask" minimizes the deviation of the neuron gate model from the experimental results [8].
Table 1 illustrates the sample of PLO, BA, and MC comparison. In the all instances PLO was the best. BA was better MC with exception of the 'Xcos' function and the "ModelTask" model.

Figure 1 shows the results of optimal packing of hundred different rectangular packages into the rectangular container. In this example, both PLO and BA were used to optimize the mixture of five greedy heuristics, the function was the total volume of packages in the container.

Table 1: Comparison of three algorithms minimizing five test functions.

Function	*Iterations*	*PLO*	*BA*	*MC*
Xcos, a=100	1,000	0.000	13.39	1.835
Xcos2, a=100	10,000	-31,123	-8.421	-8.311
Xcos3, a=1000	10,000	0.157	2.901	4.073
EcoDuel	100	0.077	0.196	0.529
EcoDuel	1000	0.0285	0.0693	0.273
Packer	10	-2.2623	-2.2572	-2.2119
ModelTask	100	4.647	5.638	4.916

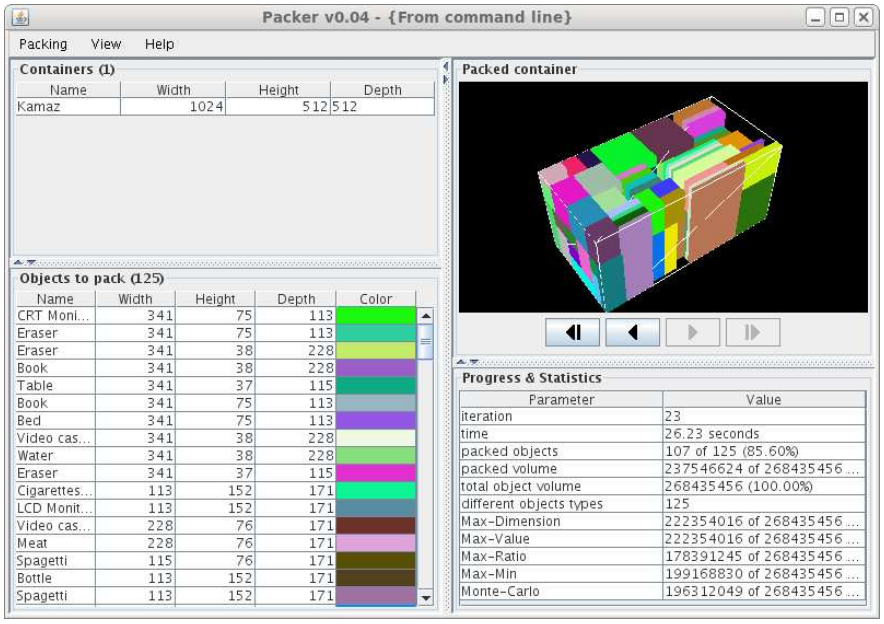

Figure 1: Results of optimal packing using PLO.

Table 2 illustrates the comparison the results of Monte Carlo (MC), PLO, DIRECT obtained using standard test functions of global optimization [3, 1, 4]. MC and PLO represent methods intended for parallel computing with no adjustable parameters. This version of DIRECT is for sequential realization with one adjustable parameter ϵ selected by the method authors. This means that we compare algorithms representing different families of optimization methods.

Table 2: Comparison of three algorithms using five test functions.

Funct	*Iter*	*MC*	*DIRECT*	*PLO*	$f(x*)$
Shekel, m=10	97	-1.201	-10.435	-8.54	-10.5364
Hartman-3,	83	-3.564	-3.862	-3.862	-3.86278
Hartman-6,	213	-2.505	-3.321	-3.322	- 3.32237
Brcos	63	1.066	0.42	0.0	0.397887
GolPri	101	8.86	3.03	3.00	3.00
SixH	113	-0.82	-1.022	-1.027	-1.031628453
Shub2D	2883	-162.3	-184.73	-172.44	-186.7309
Ackley2	561	4.394	5.64609E-05	0.003	0
Ackley10	6,780,239	-	9.4929E-02	0.143*	0
Easom	7019	-0.58	0.999989985	-0.997	-1
Griewank2	41005	0.06	7.88274E-07	0.0000	0
Mich2	67	-0.97	-1.801272488	-1.801	-1.8013
Mich5	14005	-3.313	-4.68721324	-4.19	-4.687658

DIRECT was better than PLO for 4 test functions Shekel, m=10, Ackley2, Easom, and Mich5. For two functions Hartman-3 and Mich2 the results were equal. PLO was better than DIRECT for 5 functions Hartman-6, Brcos, GolPri, SixH.and Griewank2 PLO. Using the Ackley10 function the PLO result 0.143* was obtained by 1,500,000 iterations (the DIRECT result was produced by 6,780,239 observations), since the particular implementation of PLO cannot exceed this number of observations[2] . No results of MC were produced using this function due to a similar reason.
Table 2 shows some unexpected results: minimizing Brcos function PLO obtained zero while the published minimum is 0.397887. The possible explanation is that comparison was performed using different domains.

4 Summary

The theoretical contribution is the definition of the problem of Lipschitzian optimization with unknown Lipschitz constants in terms of Pareto optimality (PO). The computational contribution is implementation the Pareto-Lipschitzian Optimization (PLO) algorithm as the Java applet with experimental calculations illustrating the PLO efficiency for functions up to 10 variables. We expect that in the future, the extensive computer simulation with various test and real functions will reveal additional aspects of the proposed algorithm and that would be an interesting new investigation.

[2]To reach 1,500,000 observations the PloN version was applied.

References

[1] Mattias Bjorkman and Kenneth Holmstrom. Global optimization using the DIRECT algorithm in Matlab 1. *AMO - Advanced Modeling and Optimization*, 1:17–37, 1999.

[2] D.E. Finkel. DIRECT optimization algorithm user guide. Technical report, Center for Research in Scientific Computation. North Carolina State University, Raleigh, NC 27695-8205, 2003.

[3] D.R. Jones, C.D. Perttunen, and B.E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Application*, 79:157–181, 1993.

[4] G. Liuzzi, S. Lucidi, and V. Piccialli. A DIRECT-based approach exploiting local minimizations for the solution of large-scale global optimization problems. *Comput Optim Appl*, 45:353–375, 2010.

[5] J. Mockus. *A Set of Examples of Global and Discrete Optimization: Application of Bayesian Heuristic Approach.* Kluwer Academic Publishers, Dordrecht-Boston-London., 2000.

[6] J. Mockus. On the Pareto-Lipschitzian optimization. *Informatica*, 2011. submitted.

[7] J. Mockus, W. Eddy, A. Mockus, L. Mockus, and G. Reklaitis. *Bayesian Heuristic Approach to Discrete and Global Optimization.* Kluwer Academic Publishers, ISBN 0-7923-4327-1, Dordrecht-London-Boston, 1997.

[8] N. Paulauskas, M. Pranevicius, H. Pranevicius, and F. Bukauskas. A stochastic four-state model of contingent gating of gap junction channels containing two fast gates sensitive to transjunctional voltage. *Biophysical Journal*, 96:3936–3948, 2009.