Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 51-19 (2018) 98–101

Evaluation of Derivative-Free Optimizers for Parameter Estimation in Systems Biology

Yannik Schälte * **Paul Stapor** * **Jan Hasenauer** *

* *Institute of Computational Biology, Helmholtz Center Munich & Chair of Mathematical Modeling of Biological Systems, Technical University Munich*

Abstract: Derivative-free optimization can be used to estimate parameters without computing derivatives. As there exist many methods, it is unclear which to use in practice. Here, we present two comparative studies of 18 state-of-the-art methods: Firstly, we evaluate them on a set of 466 classic optimization test problems of dimension 2 to 300. Secondly, we study their performance in parameter estimation on 8 ODE models of biological processes, comparing them to state-of-the-art derivative-based optimization. We observe that different problem features necessitate the use of different methods, for which we can give suggestions based on our findings. Our analysis suggests that classic test problems are not representative for problems in systems biology. For ODE models, we find that purely derivative-free methods are for most problems not reliable or at least inferior to derivative-based methods.

Keywords: Derivative-Free Optimization, Parameter Estimation, ODE models.

1. INTRODUCTION

In this study, we are concerned with optimizing an objective function $J : \mathbb{R}^n \to \mathbb{R}$ subject to box constraints: We seek $\min_x J(x)$ subject to $l \leq x \leq u$, where $x, l, u \in \mathbb{R}^n$, $l \leq u$ component-wise. A common approach is to iteratively improve a current parameter guess by local hill-climbing in the objective function landscape. Derivative-based algorithms use derivative information of J to find a good descent direction (Martínez and Raydan, 2017). Under certain smoothness and convexity assumptions, such algorithms are guaranteed to find a locally optimal solution in a number of steps independent of n (Nesterov, 2013). However, derivatives cannot always be computed reliably, e.g. J may be non-smooth, time-consuming to evaluate, or noisy, such that algorithms relying on derivatives or computing finite differences may fail. Another problem is multi-modality, where hill-climbing algorithms only find local minima. In *derivative-free optimization* (DFO), i.e. optimization using only the function values of J, diverse strategies exist to deal with these challenges, most of them based on heuristics. Only for some algorithms, convergence proofs under certain assumptions can be given. For further information see e.g. Koziel and Yang (2011).

DFO algorithms can be classified by various aspects: A basic distinction is between local and global searches. *Local* searches in principle do hill-climbing. While *direct* local methods try to minimize J directly, *model-based* methods build an adaptive surrogate model for J to guide the search. Local searches are designed to deliver locally optimal solutions. Therefore, to find globally optimal solutions, in multi-modal landscapes one has to run multiple searches from different starting points, i.e. do *multi-start local optimization* (Raue et al., 2013). *Global* searches attempt to efficiently cover the entire search space to find global optima. Another distinction is between *deterministic* searches and *stochastic* searches, which exploit randomness to overcome certain shortcomings like too early stopping. The classifications are non-exclusive, and *hybrid* methods exist which combine different ideas.

We compared the performance of state-of-the-art implementations of DFO algorithms on test problems. The setup was based on a recent comprehensive study by Rios and Sahinidis (2012), but with a different focus: Apart from considering other optimizers, we used a multi-start setting to adequately compare local and global methods, and we examined also the influence of noise. Our study on classic test problems being of independent interest, thereafter we performed tests on biological ODE models to see whether our findings can be transferred to this special problem type.

In contact with the respective developers, we improved several of the implementations during our study. The full MATLAB code is available at `https://doi.org/10.5281/zenodo.1254078`.

2. METHODS AND IMPLEMENTATION

2.1 Test problems

We used 46 types of classic optimization test problems, based on a collection by Jamil and Yang (2013). Of these, 18 were of fixed dimension 2 to 4, and 28 defined for arbitrary dimension. The latter we considered in dimensions 2, 3, 4, 5, 10, 15, 20, 25, 30, 35, 40, 50, 75, 100, 200, 300, covering the range of small- to medium-scale systems biology ODE models. So, we had 466 test problems in total, thereof 195 convex, 271 non-convex, 287 smooth, and 179 non-

2405-8963
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2018.09.025

Table 1. Optimization software.

Optimizer	Version	Literature	Type
BOBYQA	2009	Powell (2009)	local, model-based
CMAES	3.61beta	Hansen and Ostermeier (1996)	global, evolutionary
DHC	1.0	De La Maza and Yuret (1994)	local, direct
DIRECT	4.0	Finkel (2003)	global, partitioning
FMINCON	R2017A	Byrd et al. (2000)	local, gradient-based
FMINSEARCHBND	R2017A	Nelder and Mead (1965)	local, direct
GA	R2017B	Mitchell (1998)	global, evolutionary
IMFIL	1.0	Kelley (2011)	local, direct
MCS	2.0	Huyer and Neumaier (1999)	global, partitioning
MEIGO-ESS	03-07.2014	Egea et al. (2014)	global, hybrid
PARTICLESWARM	R2017B	Eberhart and Kennedy (1995)	global, swarm-based
PATTERNSEARCH	R2017B	Torczon (1997)	local, direct
PSWARM	2.1	Vaz and Vicente (2009)	global, swarm-based
RCS	1.0	-	local, direct
SIMULANNEALBND	R2017B	Kirkpatrick et al. (1983)	local, direct

smooth. The problems cover issues optimizers typically have problems with, including multiple local minima, non-smoothness, global minima with a small attraction area, sharp ridges, crescent-shaped valleys, and flat plateaus.

In addition to the optimization test problems, we considered 8 ODE models of biological processes with 2 to 48 parameters: M1 ($n = 2$) is the model of a simple conversion reaction, M2 ($n = 4$) a model of an enzyme-catalyzed reaction, M3 ($n = 5$) a model for mRNA transfection (Leonhardt et al., 2014), M4 ($n = 7$) a discretized spatial model for Pom1 gradient formation (Hross et al., 2016), M5 ($n = 11$) a model with Hopf bifurcation (Ballnus et al., 2017), M6 ($n = 17$) a model of Epo-induced JAK-STAT signaling (Swameye et al., 2003), M7 ($n = 28$) a model of RAF-MEK-ERK signaling (Fiedler et al., 2016), and M8 ($n = 48$) a model of histone methylation (Zheng et al., 2012). For all these models, the negative log-likelihood of measurement data given simulations assuming a Gaussian noise model was used as objective function.

2.2 Optimizers

We focused on open-source implementations freely available for research purposes, and algorithms from MATLAB toolboxes. For details on the diverse algorithms we refer to the references in Table 1 and to the provided code. Note that a classification of the type is generally not clear.

As local methods, we examined the direct methods DHC (an own implementation of dynamic hill-climbing), MEIGO-DHC (a DHC implementation from the MEIGO toolbox), FMINSEARCHBND (simplex-based), IMFIL (implicit filtering), PATTERNSEARCH with GPS (generalized pattern search) and MADS (mesh adaptive direct search) as polling methods, RCS (randomized coordinate search, own implementation), SIMULANNEALBND (simulated annealing), as well as the model-based method BOBYQA (quadratic approximation).

As global methods, we examined the evolutionary methods GA (genetic) and CMAES (covariance matrix adaptation), the particle swarm methods PARTICLESWARM and PSWARM, and the partitioning methods DIRECT (dividing rectangles) and MCS (multilevel coordinate search). Furthermore, we included the explicitly hybrid MEIGO-ESS method (enhanced scatter search), with local searches via BOBYQA, MEIGO-DHC, and DHC.

For comparison, we included the gradient-based MATLAB FMINCON interior point algorithm, which approximates gradients via finite differences when given no derivatives.

For the ODE models, we considered optimizers which had performed reasonably on the classic test problems: BOBYQA, DHC, DIRECT, MCS, CMAES, PSWARM, MEIGO-ESS-BOBYQA/DHC, and FMINCON. For comparison, we also considered optimization with FMINCON using analytic gradients (denoted as +g), which can be computed for ODE systems efficiently via forward or adjoint sensitivity analysis. For this, we used the AMICI toolbox (Frhlich et al., 2017).

2.3 Experimental setup

A fair comparison of global and local optimizers is difficult: Global optimizers ideally converge in every run to the global optimum at the cost of a high computational budget, whereas local optimizers converge locally (and must therefore be used in a multi-start setting), but substantially faster. Further, multi-start local optimizers require no mutual communication and can thus be run in parallel on any distributed system, whereas parallelization for global optimizers is limited by the principles and implementation of the employed algorithm.

To account for all these aspects, the setup for the study on classic optimization test problems was as follows: In setting A1, all optimizers were run $k = 20$ times with budgets of $m = 2000$ function calls for each problem. The best function value J found by any of the k runs was compared to the known solution J^*, and the problem regarded as solved if $J \leq J^* + 0.01$. Optimizers accepting starting points were initialized via latin hypercube sampling. This we call the multi-start setting. Then, in A2, to assess reliability and average behavior, we considered each of the k runs on its own. This we call the single-start setting. In setting A3, we ran the global optimizers $k_g = 1$ time with a budget of $m_g = 40000$, and compared them to the local optimizers run as in settings A1, to account for global methods needing more function calls. In setting A4, to test behavior on noisy problems, we considered setting A1, and added Gaussian noise of variance 0.03^2 to every function call. A 10 minutes time limit was imposed on each run, which was however hit only 18 times altogether.

Apart, all optimizers were applied with their respective default settings, recommended for generic applications. This was because in practice users often lack capacity or knowledge to tune optimizers to their problems.

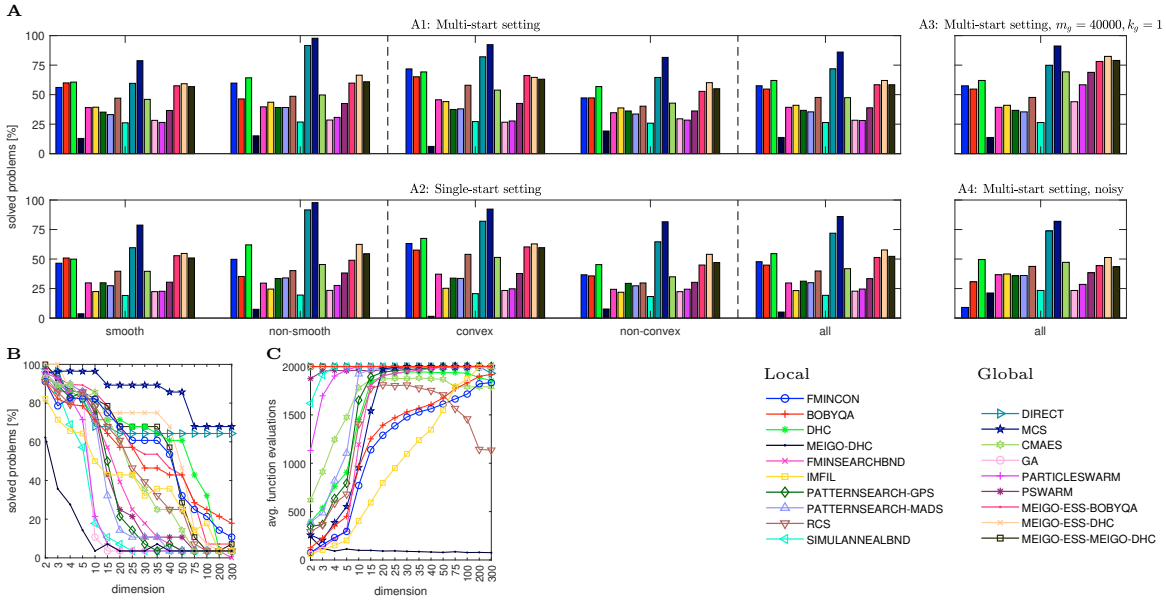

Fig. 1. Performance on classic problems. A: Fraction of solved problems by category. B: Fraction of solved problems in setting A1 by dimension. C: Average number of function calls in setting A2.

For the biological models, we included all tested optimizers in the parameter estimation toolbox PESTO (Stapor et al., 2017), offering one common interface. Here, the function call budget was chosen so large that most optimizers stopped earlier. Each optimizer was run $k = 100$ times, using latin hypercube sampling for the local methods. A run was considered converged if it got below the best function value found by any of the optimizers plus a threshold of 3. We considered an absolute threshold since the objective functions are already on the log-scale. For the ODE models, a higher threshold was used due to numerics being involved.

3. RESULTS

3.1 Evaluation on classic optimization test problems

On the optimization test problems, the partitioning algorithms MCS and DIRECT performed best, solving 86% resp. 72% of all problems already in the multi-start setting A1 with a budget of $m = 2000$ (Fig. 1 A1). Of the local optimizers, DHC (62% of all problems), FMINCON (58%) and BOBYQA (55%) fared best. However, their performance differed with the problem category: Compared to smooth problems, on non-smooth problems DHC led more clearly. This becomes particularly clear in terms of reliability (A2): Of all DHC starts on non-smooth problems, 62% ended in the optimum, for FMINCON 49%, and for BOBYQA only 35%. This indicates that on non-smooth problems, direct methods are preferable to model-based methods. But also on other problems, BOBYQA and DHC can be alternatives to FMINCON relying on finite differences.

All global optimizers aside MCS and DIRECT profited considerably from a higher function call budget (Fig. 1 A3): For $m_g = 40000$ function calls, MEIGO-ESS (78-82%) got closer to MCS (91%). In addition, its enhanced scatter search made MEIGO-ESS superior to simple multi-start. Further, we observed an advantage of CMAES (69%) over GA (44%), and of PSWARM (69%) over PARTICLESWARM (58%), so also implementation matters. Surprisingly, 7 optimizers fared worse than RCS (48%), which is a rather simple method.

The presence of noise altered the results substantially (Fig. 1 A4). In particular, model-based BOBYQA (30% of all problems) and derivative-based FMINCON (9%) suffered considerably. Direct methods like DHC (50%) were less affected, and methods that perform no local search, or are inherently stochastic, like DIRECT (72%) and CMAES (47%, like in A1), were least affected.

A higher problem dimension led, as expected, to a worse performance (Fig. 1 B). But while MCS and DIRECT still solved 68% resp. 64% in dimension 300, all others ended below 20%. Performance decreased notably rapidly for the local optimizers SIMULANNEALBND, PATTERNSEARCH, and FMINSEARCHBND, suggesting to apply these only to low-dimensional problems.

Particularly when they are expensive, the average number of function calls (Fig. 1 C) is relevant. Here, IMFIL, FMINCON and BOBYQA continuously needed the fewest. Global optimizers required on average more calls. GA and MEIGO-ESS always used the entire budget. However, it should also be noted that there was considerable variability in the optimizer proper times, e.g. PSWARM took on average 4.3s, MCS 3.1s, DHC 0.4s.

3.2 Performance on ODE models of biological processes

The evaluation of the optimizer performance for the ODE models revealed that DFO methods were able to find the optimal values only on low-dimensional, simple problems (Fig. 2 A top). Especially the global methods DIRECT and MCS, which in each run perform the same search, often missed the optimal values, but also the other DFO methods were not reliable. This is a marked contrast to the performance on the classic test problems.

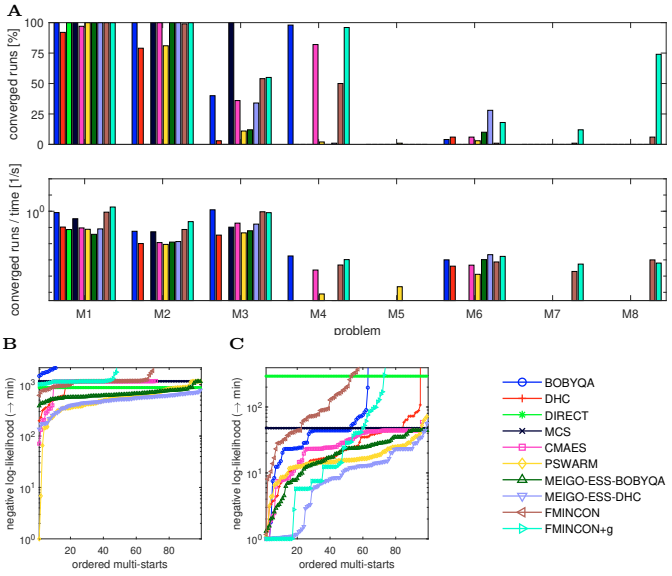

Fig. 2. Performance on ODE models. A: Number of converged runs as a measure of reliability (top), and converged runs per time as a measure of efficience (bottom). B, C: Waterfall plots (i.e. sorted multi-start results) for M5, M6.

In more detail, out of the 8 problems, DIRECT found an acceptable value only for 1, MCS for 3, DHC for 4, BOBYQA, CMAES for 5, PSWARM for 6, and FMINCON(+g) for 7 problems. In particular, for the larger models only FMINCON+g proved reliable. Indeed, the comparison of FMINCON and FMINCON+g revealed that analytic gradients improved convergence rate considerably.

The typical behavior of the optimizers can be seen exemplarily in the waterfall plot for M6 (Fig. 2 C), where FMINCON+g is the only optimizer showing plateaus indicating local minima. An exception was M5 (Fig. 2 B), which possesses many local optima and for which gradient evaluation is unstable due to stiffness of the ODE system. Here, the DFO methods PSWARM and CMAES performed best, albeit not reliably. This indicates that when derivative-based methods fail, DFO may be an alternative.

4. DISCUSSION

We performed a comprehensive evaluation of DFO methods for systems biological ODE models. The results suggest that classic optimization test problems are not representative for systems biology. This necessitates more suited collections of test problems to facilitate meaningful evaluations.

In general, DFO methods did not perform well on the ODE models and were outperformed by derivative-based methods. We are not aware of any study on typical properties of biological ODE models, but a possible explanation consists in the rather sharp global optima (although the attraction area may be big), for which gradients are of considerable value in optimization. Further investigation of the typical properties would be of use.

Recently, it has been suggested that hybrid schemes combining an intelligent derivative-free global search and derivative-based local searches perform very well, in particular when gradients can be computed efficiently (Villaverde et al., 2018). This clearly encourages a further pursuit of this path of research.

REFERENCES

Ballnus, B., Hug, S., Hatz, K., Grlitz, L., Hasenauer, J., and Theis, F.J. (2017). Comprehensive benchmarking of Markov chain Monte Carlo methods for dynamical systems. *BMC Syst Biol*, 11(1).

Byrd, R.H., Gilbert, J.C., and Nocedal, J. (2000). A trust region method based on interior point techniques for nonlinear programming. *Math. Program.*, 89(1), 149–185.

De La Maza, M. and Yuret, D. (1994). Dynamic hill climbing. *AI expert*, 9.

Eberhart, R. and Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Micro Machine and Human Science, Proc. 6th Int. Symp.*, 39–43.

Egea, J.A., Henriques, D., Cokelaer, T., Villaverde, A.F., MacNamara, A., Danciu, D.P., Banga, J.R., and 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.

Fiedler, A., Raeth, S., Theis, F.J., Hausser, A., and Hasenauer, J. (2016). Tailored parameter optimization methods for ordinary differential equation models with steady-state constraints. *BMC Syst Biol*, 10(1).

Finkel, D.E. (2003). Direct optimization algorithm user guide. *Center for Research in Scientific Computation, North Carolina State University*, 2.

Frhlich, F., Kaltenbacher, B., Theis, F.J., and Hasenauer, J. (2017). Scalable parameter estimation for genome-scale biochemical reaction networks. *PLoS Comput Biol*, 13(1), e1005331.

Hansen, N. and Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Evolutionary Computation*, 312–317.

Hross, S., Fiedler, A., Theis, F.J., and Hasenauer, J. (2016). Quantitative comparison of competing PDE models for Pom1p dynamics in fission yeast. In R. Findeisen, E. Bullinger, E. Balsa-Canto, and K. Bernaerts (eds.), *Proc. 6th IFAC Conf. Found. Syst. Biol. Eng.*, volume 49, 264–269. IFAC-PapersOnLine.

Huyer, W. and Neumaier, A. (1999). Global optimization by multilevel coordinate search. *Journal of Global Optimization*, 14(4), 331–355.

Jamil, M. and Yang, X.S. (2013). A literature survey of benchmark functions for global optimization problems. *arXiv preprint arXiv:1308.4008v1*.

Kelley, C.T. (2011). *Implicit Filtering*. Society for Ind. and App. Mathematics.

Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671–680.

Koziel, S. and Yang, X.S. (2011). *Computational optimization, methods and algorithms*, volume 356. Springer.

Leonhardt, C., Schwake, G., Stgbauer, T.R., Rappl, S., Kuhr, J.T., Ligon, T.S., and Rdler, J.O. (2014). Single-cell mRNA transfection studies: Delivery, kinetics and statistics by numbers. *Nanomedicine: Nanotechnology, Biology and Medicine*, 10(4), 679–688.

Martínez, J.M. and Raydan, M. (2017). Cubic-regularization counterpart of a variable-norm trust-region method for unconstrained minimization. *Journal of Global Optimization*, 68(2), 367–385.

Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press.

Nelder, J.A. and Mead, R. (1965). A simplex method for function minimization. *The Computer Journal*, 7(4), 308–313.

Nesterov, Y. (2013). *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media.

Powell, M. (2009). The bobyqa algorithm for bound constrained optimization without derivatives.

Raue, A., Schilling, M., Bachmann, J., Matteson, A., Schelke, M., Kaschek, D., Hug, S., Kreutz, C., Harms, B.D., Theis, F.J., Klingmller, U., and Timmer, J. (2013). Lessons learned from quantitative dynamical modeling in systems biology. *PLoS ONE*, 8(9), e74335.

Rios, L.M. and Sahinidis, N.V. (2012). Derivative-free optimization: A review of algorithms and comparison of software implementations. *Journal of Global Optimization*, 56(3), 1247–1293.

Stapor, P., Weindl, D., Ballnus, B., Hug, S., Loos, C., Fiedler, A., Krause, S., Hro, S., Frhlich, F., and Hasenauer, J. (2017). PESTO: Parameter EStimation TOolbox. *Bioinformatics*.

Swameye, I., Muller, T.G., Timmer, J., Sandra, O., and Klingmuller, U. (2003). Identification of nucleocytoplasmic cycling as a remote sensor in cellular signaling by databased modeling. *Proceedings of the National Academy of Sciences*, 100(3), 1028–1033.

Torczon, V. (1997). On the convergence of pattern search algorithms. *SIAM Journal of Optimization*.

Vaz, A.I. and Vicente, L.N. (2009). PSwarm: A hybrid solver for linearly constrained global derivative-free optimization. *Optimization Methods and Software*, 24(4-5), 669–685.

Villaverde, A.F., Froehlich, F., Weindl, D., Hasenauer, J., and Banga, J.R. (2018). Benchmarking optimization methods for parameter estimation in large kinetic models. *bioRxiv*.

Zheng, Y., Sweet, S.M.M., Popovic, R., Martinez-Garcia, E., Tipton, J.D., Thomas, P.M., Licht, J.D., and Kelleher, N.L. (2012). Total kinetic analysis reveals how combinatorial methylation patterns are established on lysines 27 and 36 of histone h3. *Proc. National Academy of Sciences*, 109(34), 13549–13554.