Benchmarking DFO algorithms

MTH8418

S. Le Digabel, Polytechnique Montréal

Winter 2020

(v2)

Plan

Introduction and problems

Tables and convergence graph

Performance and data profiles

References

Introduction

- How to measure the performance of an algorithm?
- How to compare algorithms amongst themselves?
- How to identify groups of problems?
- Are we interested in the final solution only, or in the progression of the algorithms?

How to benchmark DFO algorithms

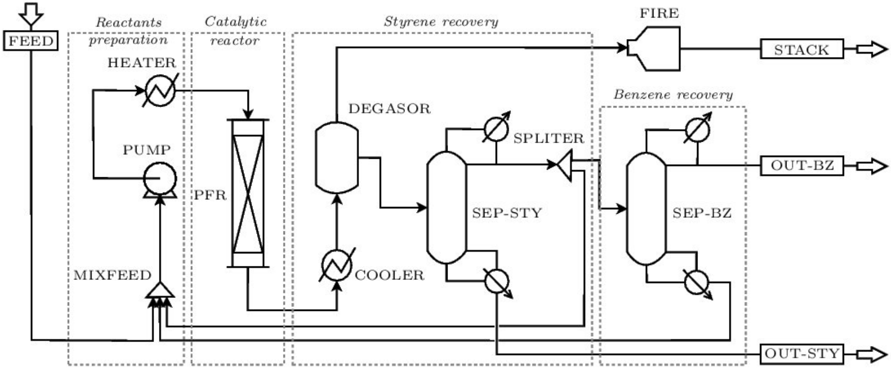
- Performance indicators:
  - Typically: Value of f and number of blackbox evaluations
  - Special measures for robustness, multiobjective optimization, etc.
- In the DFO context, CPU time is rarely relevant since the number of evaluations is a perfect machine-independent criterion
- However, CPU time is useful for:
  - Blackboxes with variable complexity
  - Parallelism
  - ...

How to find problems

- We want to achieve reproducibility. However most blackbox applications are proprietary and/or cannot be shared easily. There is not yet a universal and accepted collection of such problems.
- Typically, algorithms are tested on a mixture of analytic problems and real-life applications.
- When testing on real applications, it is necessary to use many instances, for example by changing the parameters of the applications, or by using several starting solutions.

A realistic blackbox for benchmarking

STYRENE problem [Audet et al., 2008]



8 variables, 11 constraints, one evaluation $\simeq$ 1s, $\simeq$20% of failures

Collections of analytic problems

The usual analytic collections are:

- The Hock and Schittkowski set [Hock and Schittkowski, 1981]
- The Lukšan and Vlček set [Lukšan and Vlček, 2000]
- Problems used in [Moré and Wild, 2009]. Core of 22 unconstrained CUTEst problems reformulated as 212 instances (smooth, nonsmooth, and noisy)
- CUTEst, the latest evolution of the CUTE and CUTEr collections [Gould et al., 2015]
- The COCO platform
- Isolated problems such as ROSENBROCK or GRIEWANK

Tables of numerical results

- Useful to summarize instance characteristics
- Give the exact values for each execution of Solver s on Problem p
- Problem: becomes difficult to read (*too much information*)

Table example

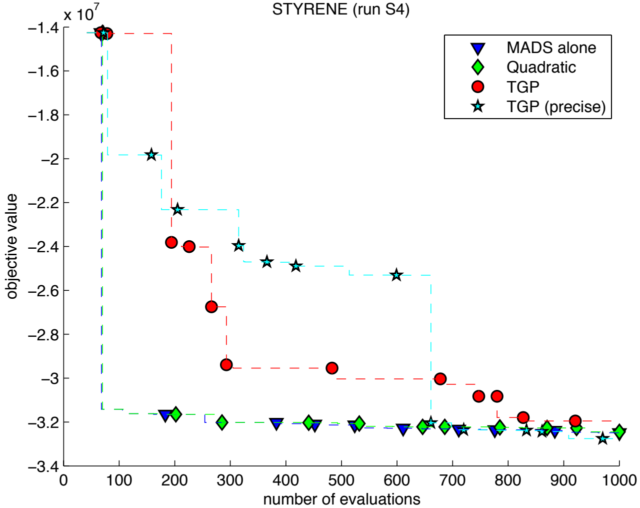
Table 6 Pooling Test Problems from the Literature

	Parameters			Solution				CPU time (sec)			Error (%)		
Example	*nsp*	k_{max}	*nt*	Exact	MSLP	MALT	VNS	MSLP	MALT	VNS	MSLP	MALT	VNS
Flow model													
AST1	1,000	10	1	549.803	276.661	532.901	545.27	2.20	2.45	2.81	49.68	3.07	0.82
AST2	1,500	10	1	549.803	284.186	535.617	543.909	9.18	5.21	5.68	48.31	2.58	1.07
AST3	1,000	10	1	561.048	255.846	397.441	412.145	18.71	4.96	5.34	54.35	29.09	26.47
AST4	230	0	0	877.649	—	876.206	876.206	0.82	0.77	1.01	—	0.16	0.16
BT4	5	0	0	45	39.6970	45	45	0.01	0.01	0.01	11.78	0	0
BT5	10	15	2	350	327.016	324.077	350	0.03	0.09	1.11	6.57	7.41	0
F2	120	10	1	110	100	107.869	110	0.07	0.44	0.57	9.09	1.94	0
H1	5	0	0	40	40	40	40	0.02	0.01	0.01	0	0	0
H2	5	0	0	60	60	60	60	0.02	0.01	0.01	0	0	0
H3	5	3	1	75	60.7332	70	75	0.02	0.01	0.03	19.02	6.67	0
RT1	5	0	0	4,136.22	126.913	4,136.22	4,136.22	1.34	0.04	0.04	96.93	0	0
RT2	5	5	1	4,391.83	—	4,330.78	4,391.83	0.04	0.47	0.60	—	1.39	0
GP1	50	5	1	60.5	28.732	35	46	0.01	0.04	0.08	52.51	42.15	23.97
Proportion model													
AST1	1,000	10	1	549.803	544.307	532.901	533.783	1.14	2.38	2.61	1	3.07	2.91
AST2	1,500	10	1	549.803	548.407	535.617	542.54	3.04	4.97	5.37	0.25	2.58	1.32
AST3	1,000	10	1	561.048	551.081	397.441	558.835	4.98	4.98	5.93	1.68	29.09	0.3
AST4	230	0	0	877.649	—	876.206	876.206	1.19	1.21	1.55	—	0.16	0.16
BT4	5	0	0	45	39.7019	45	45	0.01	0.02	0.02	11.77	0	0
BT5	10	15	2	350	292.532	323.12	350	0.12	0.16	1.53	16.42	7.68	0
F2	120	0	0	110	110	110	110	0.15	0.49	0.49	0	0	0
H1	5	0	0	40	40	40	40	0.02	0.01	0.01	0	0	0
H2	5	0	0	60	60	60	60	0.02	0.01	0.01	0	0	0
H3	5	3	1	75	69.9934	70	75	0.02	0.01	0.02	6.68	6.67	0
RT1	5	0	0	4,136.22	3,061.03	4,136.22	4,136.22	0.07	0.03	0.03	25.99	0	0
RT2	5	5	1	4,391.83	4,391.02	4,330.77	4,391.82	0.04	0.58	0.72	0.02	1.39	0

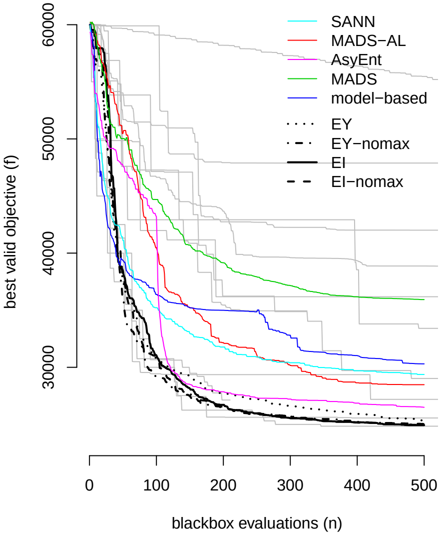
Convergence graph

- Provides a picture of the convergence of one or several methods for one given instance of a problem
- Represents only the successes or all evaluations
- Horizontal steps between two f values
- Problem: Only for one instance. Does not allow to draw any general conclusion

Convergence graph example (1)

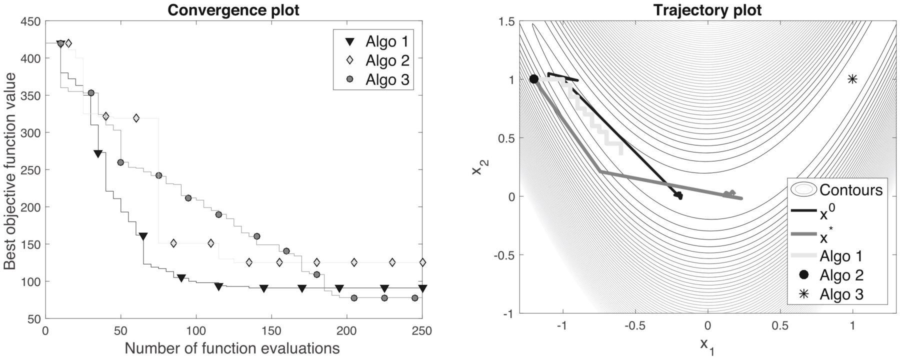


Convergence graph example (2)



From [Gramacy et al., 2015]

Convergence + trajectory plots for 2D examples



From [Audet and Hare, 2017]

Introduction and problems

Tables and convergence graph

Performance and data profiles

References

Performance and data profiles

- Introduced by [Dolan and Moré, 2002, Moré and Wild, 2009]
- Graphical and straightforward way of comparing methods on sets of problems
- **Relative** comparison of the methods
- Moré and Wild give the MATLAB tools to draw profiles at `http://www.mcs.anl.gov/~more/dfo/`
- We consider:
  - Unconstrained problems
  - Single-objective problems
  - Deterministic instances and algorithms
  - No parallelism

Profiles: Original version from the M&W paper

- $\mathcal{P}$: set of problems or instances
- $\mathcal{S}$: set of solvers, or algorithms, or methods
- Performance measure $t_{p,s} > 0$ available for each $p \in \mathcal{P}$ and $s \in \mathcal{S}$. Typically the number of evaluations required to satisfy a convergence test
- Small values of the performance measure are preferable
- Performance ratio for problem $p \in \mathcal{P}$ and solver $s \in \mathcal{S}$:

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,a} : a \in \mathcal{S}\}}$$

Convergence test (1/2)

- One possible convergence test is, for the candidate solution x:

$$f(x_0) - f(x) \geq (1 - \tau)(f(x_0) - f_L)$$

- Where:
  - $\tau > 0$: tolerance
  - x_0: **unique** and **feasible** starting point
  - f_L: smallest value of f obtained by any solver within a given budget of evaluations, for each $p \in \mathcal{P}$
- It requires that the reduction $f(x_0) - f(x)$ achieved by x be at least $1 - \tau$ times the best possible reduction $f(x_0) - f_L$
- τ represents the percentage decrease from $f(x_0)$. As it decreases, the accuracy of $f(x)$ as an approximation to f_L increases

Convergence test (2/2)

- **Example 1**: Values of $f(x_0)$ and of f after 100 evaluations, for 3 algorithms on 2 problems:

	Pb 1	Pb 2
$f(x_0)$	10	-703.57
f Algo. 1	0.01	-1,907.47
f Algo. 2	1.2	-3,964.20
f Algo. 3	0	-3,682.12

For $\tau = 0.1$, when does the convergence test pass?

- Other convergence tests include the relative error between $f(x)$ and f_L. For example $\frac{f(x)-f_L}{|f_L|} \leq \tau$ if $f_L \neq 0$

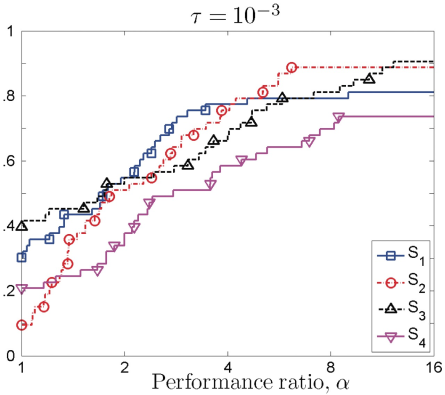
Performance profiles

- The best solver $s^* \in \mathcal{S}$ for a particular problem $p \in \mathcal{P}$ attains the lower bound $r_{p,s^*} = 1$
- $t_{p,s} = r_{p,s} = \infty$ when s fails to satisfy the convergence test on p
- The performance profile of s is the fraction of problems where the performance ratio is at most α:

$$\rho_s(\alpha) = \frac{1}{|\mathcal{P}|}\text{size}\{p \in \mathcal{P} : r_{p,s} \leq \alpha\}$$

- It is the probability distribution for the ratio $r_{p,s}$
- $\rho_s(1)$ is the fraction of problems for which s performs the best
- For α sufficiently large, $\rho_s(\alpha)$ is the fraction of problems solved by s
- Solvers with high values for ρ_s are preferable

Performance profiles: Example



Accurate view of the performance for $\tau = 10^{-3}$. Taken from [Moré and Wild, 2009]

Example 2

Draw the performance profiles for the following table:

$t_{p,s}$	Pb 1	Pb 2
Algo. 1	35	∞
Algo. 2	∞	1,200
Algo. 3	112	500

$t_{p,s}$ represents the number of evaluations for which the convergence test succeeded, for a given value of τ

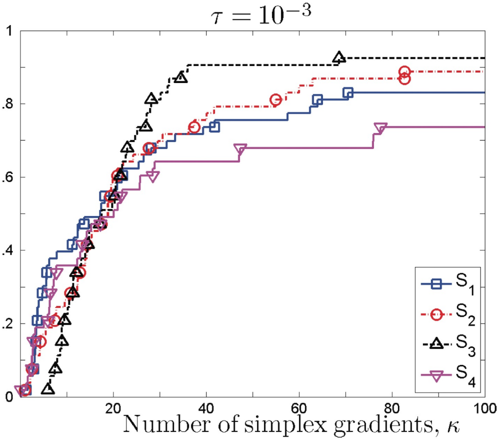
Data profiles

- We are interested in the percentage of problems that can be solved, for a given tolerance τ with a variable budget of evaluations
- The data profile of Solver s is

$$d_s(\kappa) = \frac{1}{|\mathcal{P}|}\text{size}\left\{p \in \mathcal{P} : \frac{t_{p,s}}{n_p + 1} \leq \kappa\right\},$$

  where n_p is the number of variables in Problem p
- It represents the percentage of problems that can be solved with κ groups of $n_p + 1$ function evaluations, or simplex gradient estimates
- $n_p + 1$ is the number of evaluations needed to compute a one-sided finite-difference estimate of the gradient

Data profiles: Example



Taken from [Moré and Wild, 2009]

Example 3

Draw the data profiles for the following table.

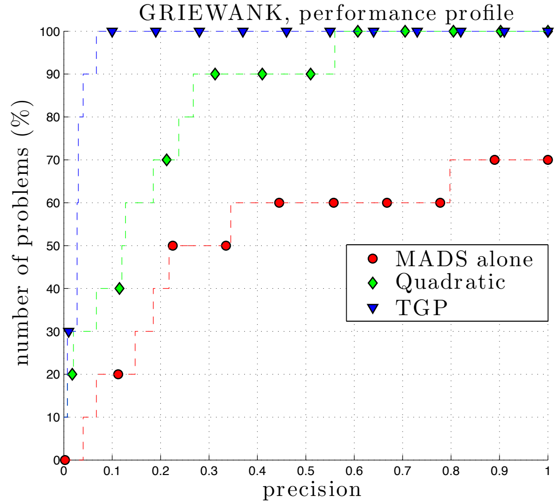
$t_{p,s}$	Pb 1	Pb 2
Algo. 1	35	∞
Algo. 2	∞	1,200
Algo. 3	112	500

$t_{p,s}$ represents the number of evaluations for which the convergence test succeeded, for a given value of τ. Problem 1 has 2 variables and Problem 2 has 9 variables

Simplified performance profiles

- Consider only the final solutions. The stopping criteria must be the same for all the algorithms (i.e. same budget of evaluations)
- x-axis: tolerance τ
- y-axis: percentage of problems *solved* given the τ tolerance
- "solved" must be defined
- At $\tau = 0$:
  - We see the methods that gave the best solutions
  - The performance values may sum up to a value $< 100\%$

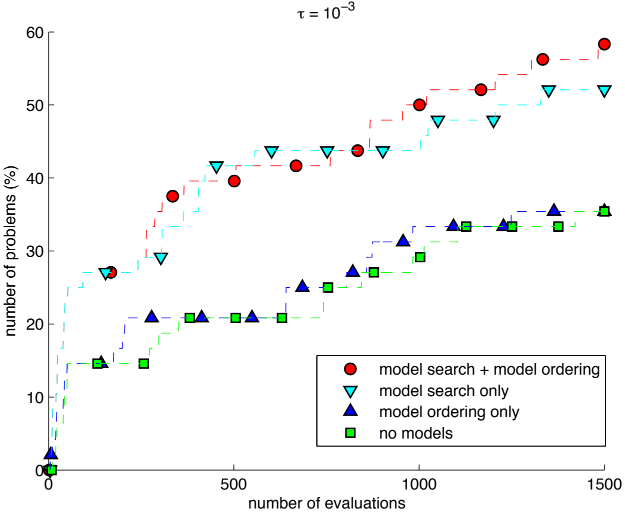
Simplified performance profiles: Example



Simplified data profiles

- We consider the entire history of all executions: We focus on the convergence
- The tolerance τ is fixed
- x-axis: Convergence measure: Number of evaluations or number of simplex gradients when the problems have different dimensions
- y-axis: Percentage of problems *solved* given the τ tolerance
- For $x = 0$, we should have $y = 0$
- For $x = x_{max}$, we should observe the same values on the performance profiles, at τ
- Equivalent to the original version. Only the presentation differs

Simplified data profiles: Example



Extensions

- How to extend performance and data profiles
  - To the constrained case?
  - With parallelism?
  - With stochastic algorithms?
- Several choices have to be taken. This is the subject of Homework #1 for the constrained case
- Accuracy profiles [Audet and Hare, 2017] for the robustness and quality of the final solution

Introduction and problems

Tables and convergence graph

Performance and data profiles

References

References

- Performance and data profiles [Dolan and Moré, 2002, Moré and Wild, 2009]
- Test problems [Hock and Schittkowski, 1981, Lukšan and Vlček, 2000, Moré and Wild, 2009, Gould et al., 2015]
- Some benchmark papers [Whitley et al., 2006, Fowler et al., 2008, Moré and Wild, 2009, Rios and Sahinidis, 2013, Martelli and Amaldi, 2014]

References I

Audet, C., Béchard, V., and Le Digabel, S. (2008).
Nonsmooth optimization through Mesh Adaptive Direct Search and Variable Neighborhood Search.
Journal of Global Optimization, 41(2):299–318.

Audet, C. and Hare, W. (2017).
Derivative-Free and Blackbox Optimization.
Springer Series in Operations Research and Financial Engineering. Springer International Publishing, Berlin.

Dolan, E. and Moré, J. (2002).
Benchmarking optimization software with performance profiles.
Mathematical Programming, 91(2):201–213.

Fowler, K., Reese, J., Kees, C., Dennis Jr., J., Kelley, C., Miller, C., Audet, C., Booker, A., Couture, G., Darwin, R., Farthing, M., Finkel, D., Gablonsky, J., Gray, G., and Kolda, T. (2008).
Comparison of derivative-free optimization methods for groundwater supply and hydraulic capture community problems.
Advances in Water Resources, 31(5):743–757.

Gould, N., Orban, D., and Toint, P. (2015).
CUTEst: a Constrained and Unconstrained Testing Environment with safe threads for mathematical optimization.
Computational Optimization and Applications, 60(3):545–557.
Code available at `http://ccpforge.cse.rl.ac.uk/gf/project/cutest/wiki`.

References II

Gramacy, R., Gray, G., Le Digabel, S., Lee, H., Ranjan, P., Wells, G., and Wild, S. (2015).
Modeling an Augmented Lagrangian for Improved Blackbox Constrained Optimization.
To appear in *Technometrics* (with discussion).

Hock, W. and Schittkowski, K. (1981).
Test Examples for Nonlinear Programming Codes, volume 187 of *Lecture Notes in Economics and Mathematical Systems*.
Springer Verlag, Berlin, Germany.

Lukšan, L. and Vlček, J. (2000).
Test problems for nonsmooth unconstrained and linearly constrained optimization.
Technical Report V-798, ICS AS CR.

Martelli, E. and Amaldi, E. (2014).
PGS-COM: A hybrid method for constrained non-smooth black-box optimization problems: Brief review, novel algorithm and comparative evaluation.
Computers and Chemical Engineering, 63:108–139.

Moré, J. and Wild, S. (2009).
Benchmarking derivative-free optimization algorithms.
SIAM Journal on Optimization, 20(1):172–191.
Test problems and results available at `http://www.mcs.anl.gov/~more/dfo/`.

References III

Rios, L. and Sahinidis, N. (2013).
Derivative-free optimization: a review of algorithms and comparison of software implementations.
Journal of Global Optimization, 56(3):1247–1293.

Whitley, D., Lunacek, M., and Sokolov, A. (2006).
Comparing the Niches of CMA-ES, CHC and Pattern Search Using Diverse Benchmarks.
In *Parallel Problem Solving from Nature - PPSN IX*, volume 4193/2006 of *Lecture Notes in Computer Science*, pages 988–997. Springer Berlin / Heidelberg.