PDFO

A Cross-Platform MATLAB/Python Interface for Powell's Derivative-Free Optimization Solvers

Tom M. Ragonneau, joint work with Zaikun Zhang

July 21, 2021 (SIAM OP21, MS38)

Department of Applied Mathematics
The Hong Kong Polytechnic University
Funded by the Hong Kong Ph.D. Fellowship Scheme

Table of contents

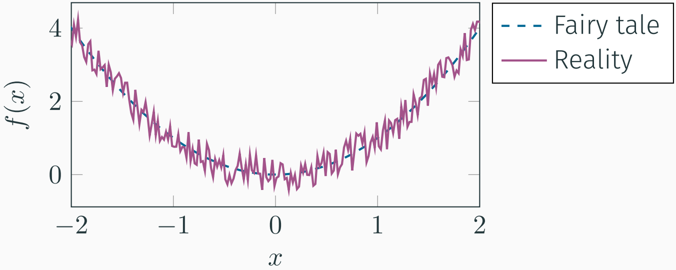
Introduction

Derivative-free optimization (DFO)

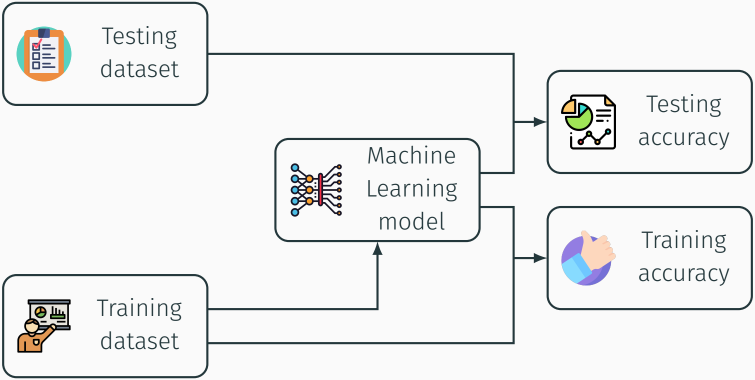
- Minimize a function f using function values but not derivatives.
- f can be a black-box resulting from experiments or simulations.

$$x \in \Omega \subseteq \mathbb{R}^n \longrightarrow \boxed{f : \mathbb{R}^n \to \mathbb{R}} \longrightarrow f(x)$$

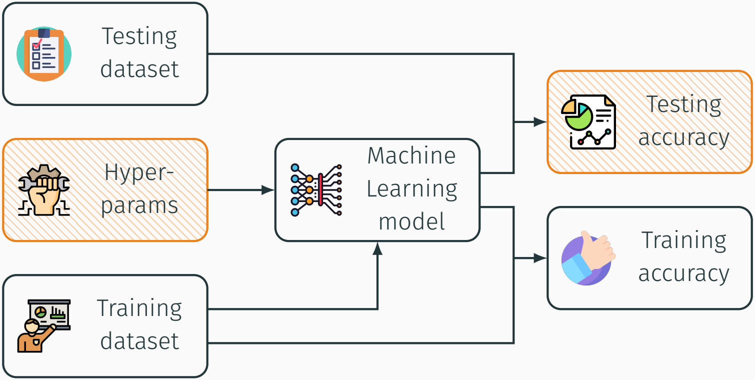
- f may be smooth, but ∇f cannot be numerically evaluated.
- Evaluations of f can be noisy and expensive.



An example of DFO problem



An example of DFO problem



How to optimize the accuracy of the model by tuning the hyperparameters? What is the gradient of the performance of the model (e.g., testing accuracy) with respect to the hyperparameters?

Powell’s derivative-free solvers

Two paradigms of methods

- Direct-search methods: sampling iteratively f at a finite number of points and choosing the iterates using simple comparisons.
  - *Examples*: Nelder-Mead, GPS, MADS, BFO, ...
- Model-based methods: modeling iteratively f using simple functions and choosing the iterates by minimizing the models.
  - *Globalization*: embedded in trust-region or line-search frameworks.
  - *Examples*: Powell's solvers, DFO, ORBIT, BOOSTERS, DFO-LS, ...

Idea of trust-region frameworks

Given a model f_k of f around x_k, the trial step d_k approximates

$$\arg\min \big\{ f_k(x_k + d) : x_k + d \in \Omega_k, \|d\| \leq \Delta_k \big\},$$

where Δ_k is the trust-region radius and $\Omega_k \approx \Omega$ around x_k. Accept the trial point $x_k + d_k$ if it satisfies some reduction condition, and update the trust-region radius Δ_k accordingly.

General overview of the Powell's derivative-free solvers

Powell developed five derivative-free trust-region solvers.

Solvers	References	Constraint types	Model types
COBYLA	Powell (1994)	nonlinear	linear (FD[1])
UOBYQA	Powell (2002)	unconstrained	quadratic (FD)
NEWUOA	Powell (2006)	unconstrained	quadratic (UD[2])
BOBYQA	Powell (2009)	bounds	quadratic (UD)
LINCOA	Powell (2015)	linear	quadratic (UD)

[1] FD: obtained by fully-determined interpolations.
[2] UD: obtained by underdetermined interpolations.

Original implementation of the solvers

Powell implemented the five solvers in Fortran 77 ...

Models for NEWUOA, BOBYQA, and LINCOA

Given a nondegenerate interpolation set $\mathcal{X}_k \subseteq \mathbb{R}^n$, the kth quadratic model f_k of the objective function f solves

$$\begin{aligned} \min \quad & \left\|\nabla^2 Q - \nabla^2 f_{k-1}\right\|_{\mathsf{F}} \\ \text{s.t.} \quad & Q(x) = f(x),\ x \in \mathcal{X}_k, \\ & Q \in \mathcal{Q}_n, \end{aligned}$$

where $\mathcal{Q}_n$ is the set of quadratic functions in $\mathbb{R}^n$.

- Typically, $\mathcal{X}_k$ has $\mathcal{O}(n)$ elements, instead of $\mathcal{O}(n^2)$.
- At each iteration, at most one point of $\mathcal{X}_k$ is modified, causing an at-most rank-2 update of the KKT matrix of the system.
- Geometry of $\mathcal{X}_k$ is maintained using Lagrange polynomials.

The PDFO package

Current version of the PDFO package

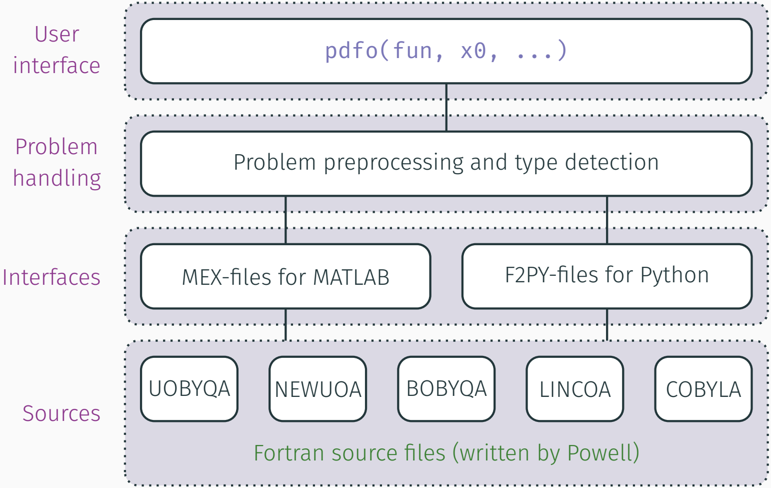
Interfaces for Powell's solvers

PDFO provides MATLAB/Python interfaces for using Powell's derivative-free solvers.

- More languages will be added in the future.
- It supports Linux, MacOS, and even Windows.
- It is NOT a reimplementation, but rather interfaces (reimplementation will come in the future)!

Visit PDFO homepage
`https://www.pdfo.net/`

Current version of the PDFO package



Core features of PDFO

PDFO preprocesses a problem as follows

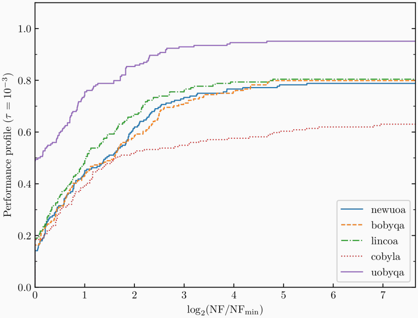
- Detect obvious infeasibility.
- Attempt to project the initial guess onto the feasible set for linearly-constrained problems.
- Eliminate linear equality constraints (QR factorization).
- Reformulate the constraints to call the Powell's solvers.
- Handle possible overflows and faults of the inputs.

Minor modifications to the Fortran source code have been made.

- The original COBYLA code may NOT return the best evaluated point.
- The original UOBYQA and LINCOA code might encounter infinite cyclings on ill-conditioned problems.
- Other programming-related bugs have also been patched.

Comparison of the Powell's solvers using PDFO

We tested PDFO using performance profiles[3] on problems from the CUTEst[4] dataset of Gould, Orban, and Toint (2015).

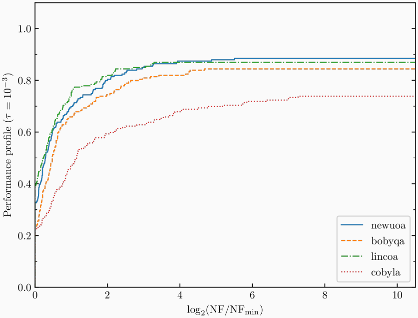


Unconstrained problems of dimensions at most 10

[3]See Dolan and Moré (2002) and Moré and Wild (2009).
[4]We used the PyCUTEst package by J. Fowkes and L. Roberts.

Comparison of the Powell's solvers using PDFO

We tested PDFO using performance profiles[3] on problems from the CUTEst[4] dataset of Gould, Orban, and Toint (2015).

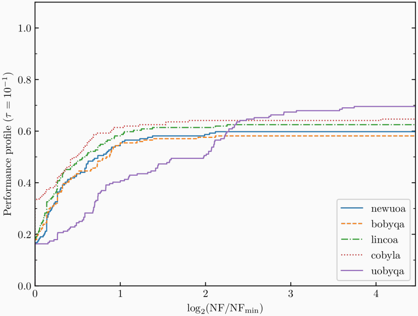


Unconstrained problems of dimensions at most 50

[3] See Dolan and Moré (2002) and Moré and Wild (2009).
[4] We used the PyCUTEst package by J. Fowkes and L. Roberts.

Comparison of the Powell's solvers using PDFO

We tested PDFO using performance profiles[3] on problems from the CUTEst[4] dataset of Gould, Orban, and Toint (2015).



Unconstrained noisy problems of dimensions at most 10

[3] See Dolan and Moré (2002) and Moré and Wild (2009).
[4] We used the PyCUTEst package by J. Fowkes and L. Roberts.

A synthetic noisy nonsmooth problem i

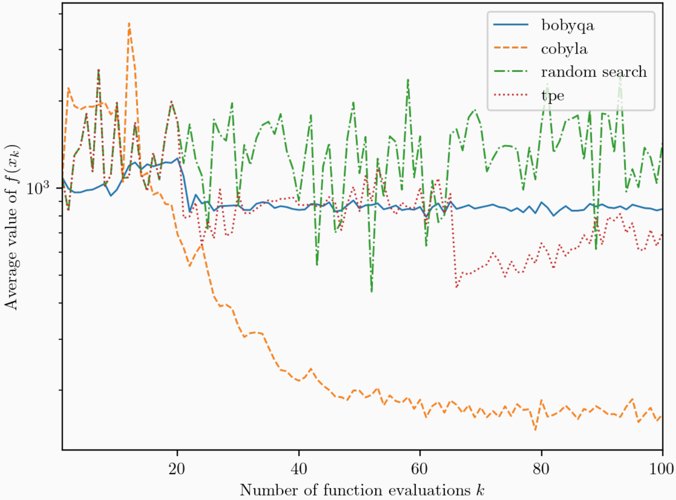
Consider the noisy Rosenbrock-like nonsmooth function

$$f(x) = \big(1 + \sigma e(x)\big) r(x) \quad \text{with} \quad r(x) = \sum_{i=1}^{n-1} 4\big|x_{i+1} - x_i^2\big| + |1 - x_i|,$$

where $e(x) \sim \mathcal{N}(0, 1)$ and $\sigma \geq 0$. In our experiment:

- dimension is $n = 10$,
- constraints are $-10 \leq x_i \leq 1/i$ for all $i = 1, 2, \ldots, n$,
- noise level is $\sigma = 0.1$,
- budget is 100 function evaluations, and
- number of random experiments is 20.

A synthetic noisy nonsmooth problem ii



A hyperparameter tuning problem

Similarly to Bradley (1997) and Ghanbari and Scheinberg (2017), consider the following hyperparameter tuning problem: optimize the AUC score[5] of an SVM (2 hyperparameters) in binary classification on given LIBSVM[6] datasets.

Dataset "splice" ($1,000$ data, 60 features)

Solvers	No. evaluations	AUC Scores	Testing accuracies
PDFO	65	0.96	0.89
Random search	100	0.64	0.53
Random search	200	0.79	0.53
TPE (Bayesian)	100	0.50	0.50

[5] See Hanley and McNeil (1982).
[6] Available at `https://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

A hyperparameter tuning problem

Similarly to Bradley (1997) and Ghanbari and Scheinberg (2017), consider the following hyperparameter tuning problem: optimize the AUC score[5] of an SVM (2 hyperparameters) in binary classification on given LIBSVM[6] datasets.

Dataset "ijcnn1" ($49,990$ data, 22 features)

Solvers	No. evaluations	AUC Scores	Testing accuracies
PDFO	59	0.99	0.98
Random search	100	0.98	0.97
Random search	200	0.98	0.97
TPE (Bayesian)	100	0.98	0.98

[5] See Hanley and McNeil (1982).

[6] Available at `https://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

Summary and conclusion

Summary and conclusion

- We have developed an initial version of PDFO.
    - It provides MATLAB/Python interfaces for Powell's DFO solvers.
    - Encouraging feedbacks are received from both academia and industry (IRT Saint-Exupéry, Toulouse, France).
    - We made brief comparisons with Bayesian optimization.
- We are now working on a new derivative-free trust-region method for nonlinear constrained problems, to be included in PDFO.

Thank you!

Contact: `tom.ragonneau@connect.polyu.hk`.

References i

- Bradley, A. P. (1997). "The use of the area under the ROC curve in the evaluation of machine learning algorithms". In: *Pattern Recognit.* 30, pp. 1145–1159.
- Dolan, E. D. and Moré, J. J. (2002). "Benchmarking optimization software with performance profiles". In: *Math. Program.* 91, pp. 201–213.
- Ghanbari, H. and Scheinberg, K. (2017). *Black-box optimization in machine learning with trust region based derivative free algorithm*. Tech. rep. 17T-005. Bethlehem, PA, US: COR@L.
- Gould, N. I. M., Orban, D., and Toint, Ph. L. (2015). "CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization". In: *Comput. Optim. Appl.* 60, pp. 545–557.

References ii

- Hanley, J. A. and McNeil, B. J. (1982). “The meaning and use of the area under a receiver operating characteristic (ROC) curve”. In: *Radiology* 143, pp. 29–36.
- Moré, J. J. and Wild, S. M. (2009). “Benchmarking derivative-free optimization algorithms”. In: *SIAM J. Optim.* 20, pp. 172–191.
- Powell, M. J. D. (1994). “A direct search optimization method that models the objective and constraint functions by linear interpolation”. In: *Advances in Optimization and Numerical Analysis*. Ed. by S. Gomez and J. P. Hennart. Dordrecht, NL: Springer, pp. 51–67.
- ——— (2002). “UOBYQA: unconstrained optimization by quadratic approximation”. In: *Math. Program.* 92, pp. 555–582.

References iii

- Powell, M. J. D. (2006). "The NEWUOA software for unconstrained optimization without derivatives". In: *Large-Scale Nonlinear Optimization*. Ed. by G. Di Pillo and M. Roma. New York, NY, US: Springer, pp. 255–297.
- ——— (2009). *The BOBYQA algorithm for bound constrained optimization without derivatives*. Tech. rep. DAMTP 2009/NA06. Cambridge, UK: Department of Applied Mathematics and Theoretical Physics, University of Cambridge.
- ——— (2015). "On fast trust region methods for quadratic models with linear constraints". In: *Math. Program. Comput.* 7, pp. 237–267.