Blackbox optimization with the MADS algorithm and the NOMAD software

Sébastien Le Digabel

MTL MLOpt, 2020-09-16
Contributors and partners
Research team

Professors

Research associates

Postdocs / Students

Graduates
Presentation outline

Introduction

The MADS algorithm

The NOMAD software package

Example 1: Aircraft takeoff trajectories

Example 2: Characterization of objects from radiographs

Example 3: Hyperparameters optimization

References
Introduction

The MADS algorithm

The NOMAD software package

Example 1: Aircraft takeoff trajectories

Example 2: Characterization of objects from radiographs

Example 3: Hyperparameters optimization

References
Blackbox / Derivative-Free Optimization

We consider

$$\min_{x \in \Omega} f(x)$$

where the evaluations of $f$ and the functions defining $\Omega$ are the result of a computer simulation (a blackbox)

- Each call to the simulation may be expensive
- The simulation can fail
- Sometimes $f(x) \neq f(x)$
- Derivatives are not available and cannot be approximated
Blackboxes as illustrated by a Boeing engineer

- **Long runtime**
- **Large memory requirement**
- **No derivatives available**
- **Local optima**
- **Non-smooth, noisy**
- **Software might fail**
“Derivative-Free Optimization (DFO) is the mathematical study of optimization algorithms that do not use derivatives” [Audet and Hare, 2017]

- Optimization without using derivatives
- Derivatives may exist but are not available
- Obj./constraints may be analytical or given by a blackbox

“Blackbox Optimization (BBO) is the study of design and analysis of algorithms that assume the objective and/or constraints functions are given by blackboxes” [Audet and Hare, 2017]

- A simulation, or a blackbox, is involved
- Sometimes referred as Simulation-Based Optimization (SBO)

“BBO” appears in the DFO literature at least since [Audet and Orban, 2006]
# Introduction

## The MADS algorithm

## The NOMAD software package

### Example 1: Aircraft takeoff trajectories

### Example 2: Characterization of objects from radiographs

### Example 3: Hyperparameters optimization

## References
Typical setting

Unconstrained case, with one initial starting solution
Algorithms for blackbox optimization

A method for blackbox optimization should ideally:

- Be efficient given a limited budget of evaluations
- Be robust to noise and blackbox failures
- Natively handle general constraints
- Have convergence properties ensuring first-order local optimality in the smooth case – otherwise why using it on more complicated problems?
- Easily exploit parallelism
- Deal with multiobjective optimization
- Deal with integer and categorical variables
- Have a publicly available implementation
Families of methods

▶ “Computer science” methods:
  ▶ Heuristics such as genetic algorithms
  ▶ No convergence properties
  ▶ Cost a lot of evaluations
  ▶ Should be used only in last resort for desperate cases

▶ “Statistical methods”:
  ▶ Bayesian optimization: EGO algorithm based on surrogates and expected improvement
  ▶ Still limited in terms of dimension
  ▶ Does not natively handle constraints
  ▶ Better to use these tools in conjunction with DFO methods

▶ Derivative-Free Optimization methods (DFO)
DFO methods

- **Model-based methods:**
  - Derivative-Free Trust-Region (DFTR) methods.
  - Based on quadratic models or radial-basis functions
  - Use of a trust-region
  - Better for \( \{ \text{DFO} \setminus \text{BBO} \} \)
  - Not resilient to noise and *hidden constraints*
  - Not easy to parallelize

- **Direct-search methods:**
  - Classical methods: Coordinate search, Nelder-Mead – the *other* simplex method
  - Modern methods: Generalized Pattern Search (GPS), Generating Set Search (GSS), Mesh Adaptive Direct Search (MADS)

So far, the size of the instances (variables and constraints) is typically limited to \( \simeq 50 \), and we target local optimization.
MADS illustration with $n = 2$: Poll step

\[
\Delta^m_k = \Delta^p_k = 1
\]

poll trial points = \{t_1, t_2, t_3\}
MADS illustration with $n = 2$: Poll step

\[ \Delta^m_k = \Delta^p_k = 1 \]

\[ \Delta^m_{k+1} = 1/4 \]
\[ \Delta^p_{k+1} = 1/2 \]

poll trial points = \{t_1, t_2, t_3\} = \{t_4, t_5, t_6\}
**MADS illustration with \( n = 2 \): Poll step**

\[
\Delta^m_k = \Delta^p_k = 1 \\
\Delta^m_{k+1} = 1/4 \\
\Delta^p_{k+1} = 1/2 \\
\Delta^m_{k+2} = 1/16 \\
\Delta^p_{k+2} = 1/4
\]

Poll trial points:
- \( \{t_1, t_2, t_3\} \)
- \( \{t_4, t_5, t_6\} \)
- \( \{t_7, t_8, t_9\} \)
[0] Initializations \((x_0, \Delta_0)\)

[1] Iteration \(k\)

[1.1] Search
- select a finite number of mesh points
- evaluate candidates opportunistically

[1.2] Poll (if Search failed)
- construct poll set \(P_k = \{x_k + \Delta^m_k d : d \in D_k\}\)
- sort\(P_k\)
- evaluate candidates opportunistically

[2] Updates
- if success
  \[x_{k+1} \leftarrow \text{success point}\]
  increase \(\Delta^m_k\)
- else
  \[x_{k+1} \leftarrow x_k\]
  decrease \(\Delta^m_k\)
  \[k \leftarrow k + 1, \text{stop or go to [1]}\]

The MADS algorithm [Audet and Dennis, Jr., 2006]
Special features of MADS

- **Constraints** handling with the Progressive Barrier technique [Audet and Dennis, Jr., 2009]

- **Surrogates** [Talgorn et al., 2015]

- **Categorical variables** [Abramson, 2004]

- **Discrete variables** [Audet et al., 2019b]

- **Global optimization** [Audet et al., 2008a]

- **Parallelism** [Le Digabel et al., 2010, Audet et al., 2008b]

- **Multiobjective optimization** [Audet et al., 2008c]

- **Handling of stochastic blackboxes** [Alarie et al., 2019, Audet et al., 2019a]
Introduction

The MADS algorithm

The NOMAD software package

Example 1: Aircraft takeoff trajectories

Example 2: Characterization of objects from radiographs

Example 3: Hyperparameters optimization

References
NOMAD (Nonlinear Optimization with MADS)

- C++ implementation of the MADS algorithm
- Standard C++. Runs on Linux, Mac OS X and Windows
- Current version: 3.9 (June 2018); NOMAD v4 beta, 2020
- Parallel versions with MPI
- Multiple interfaces: Python, Julia, MATLAB, R, Excel
- Open and free – LGPL license
- Download at https://www.gerad.ca/nomad
- Support at nomad@gerad.ca

Blackbox conception (batch mode)

- Command-line program that takes in argument a file containing $x$, and displays the values of $f(x)$ and the $c_j(x)$’s

- Can be coded in any language

- Typically: `> bb.exe x.txt` displays `$f c1 c2$` (objective and two constraints)
Run NOMAD

> nomad parameters.txt

```bash
[loot ~/Desktop/2018_VQAC_NOMAD/demo_NOMAD/mac] > ../nomad.3.8.1/bin/nomad parameters.txt

NOMAD - version 3.8.1 has been created by {
  Charles Audet - Ecole Polytechnique de Montreal
  Sebastien Le Digabel - Ecole Polytechnique de Montreal
  Christophe Tribes - Ecole Polytechnique de Montreal
}

The copyright of NOMAD - version 3.8.1 is owned by {
  Sebastien Le Digabel - Ecole Polytechnique de Montreal
  Christophe Tribes - Ecole Polytechnique de Montreal
}

NOMAD v3 has been funded by APOER, Exxon Mobil, Hydro Quebec, Rio Tinto and IVADO.

NOMAD v3 is a new version of NOMAD v1 and v2. NOMAD v1 and v2 were created and developed by Mark Abramson, Charles Audet, Gilles Couture, and John E. Dennis Jr., and were funded by APOER and Exxon Mobil.

License : '$NOMAD_HOME/src/lgpl.txt'
User guide: '$NOMAD_HOME/doc/user_guide.pdf'
Examples : '$NOMAD_HOME/examples'
Tools : '$NOMAD_HOME/tools'

Please report bugs to nomad@gerad.ca

Seed: 0

MADS run {

  BHN    OBJ
  4      0.0000000000
  21     -1.0000000000
  23     -3.0000000000
  51     -4.0000000000
  563    -4.0000000000

  } end of run (mesh size reached NOMAD precision)

  blackbox evaluations : 563
  best infeasible solution (min. violation): ( 1.0000000013 1.0000000048 0.9999999797 0.99999992 -4 ) h=1.10134e-13 f=-4
  best feasible solution : ( 1 1 1 1 -4 ) h=0 f=-4
```
Introduction

The MADS algorithm

The NOMAD software package

Example 1: Aircraft takeoff trajectories

Example 2: Characterization of objects from radiographs

Example 3: Hyperparameters optimization

References
Aircraft takeoff trajectories

- [Torres et al., 2011]


- Biobjective optimization

- Must execute on different platforms including some old Solaris distributions
Definition of the optimization problem

- Concept: Optimization of vertical flight path based on procedures designed to reduce noise emission at departure to protect airport vicinity

- Minimization of environmental and economical impact: Noise and fuel consumption

- NADP (Noise Abatement Departure Procedure), variables: During departure phase, the aircraft will target its climb configuration:
  - Increase the speed up to climb speed (acceleration phase)
  - Reduce the engine rate to climb thrust (reduction phase)
  - Gain altitude
Introduction

The MADS algorithm

The NOMAD software package

Example 1: Aircraft takeoff trajectories

Example 2: Characterization of objects from radiographs

Example 3: Hyperparameters optimization

References
Characterization of objects from radiographs - LANL

We want to identify an unknown object inside a box, using a x-ray source that gives an image on a detector.

In this work, the unknown object is supposed to be spherical.
Radiograph

A radiograph is the observed image on the detector. For example:
Description of the problem

- The problem consist to **identify the unknown object** with sufficient precision so that the object can be classified as dangerous or not

- Must work **rapidly**

- Must work for radiographs **not created on a well-controlled experimental environment**

- Must **not crash** for unreasonable user inputs
Introduction

The MADS algorithm

The NOMAD software package

Example 1: Aircraft takeoff trajectories

Example 2: Characterization of objects from radiographs

Example 3: Hyperparameters optimization

References
HPO with HyperNOMAD

- PhD project of Dounia Lakhmiri
- Technical report [Lakhmiri et al., 2019] in revision for ACM TOMS
- We focus on the HPO of deep neural networks
- Our advantages:
  - Blackbox optimization problem:
    \[ \text{One blackbox call = Training + validation + test, for a fixed set of hyperparameters} \]
  - Presence of categorical variables \((\text{ex.: number of layers})\)
  - Existing methods are mostly heuristics \((\text{grid search, random search, GAs, etc.})\)
  - Based on the NOMAD implementation of MADS
Principle

Blackbox

Construct the network

Network training, validation, testing

New point

Test accuracy

HyperNOMAD optimizer

Hyperparameters
Block structure
Neighbors

NOMAD

Initial optimization parameters: dataset, initial point, budget of evaluations
HyperNOMAD

- HyperNOMAD is the interface between NOMAD and a deep learning platform
- Based on the PyTorch library
- Works with preexisting datasets such as MNIST or CIFAR-10, or on custom data
- Available at https://github.com/bbopt/HyperNOMAD
- We consider two types of hyperparameters:
  - Architecture of the neural network
  - Optimizer
  (plus one hyperparameter for the size of mini-batches)
### Hyperparameters for the architecture \((5n_1 + n_2 + 4)\ \text{parameters}\)

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Type</th>
<th>Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of convolutional layers ((n_1))</td>
<td>Categorical</td>
<td>{0, 1, \ldots, 20}</td>
</tr>
<tr>
<td>Number of output channels</td>
<td>Integer</td>
<td>{0, 1, \ldots, 50}</td>
</tr>
<tr>
<td>Kernel size</td>
<td>Integer</td>
<td>{0, 1, \ldots, 10}</td>
</tr>
<tr>
<td>Stride</td>
<td>Integer</td>
<td>{1, 2, 3}</td>
</tr>
<tr>
<td>Padding</td>
<td>Integer</td>
<td>{0, 1, 2}</td>
</tr>
<tr>
<td>Pooling size</td>
<td>Integer</td>
<td>{1, 2, \ldots, 5}</td>
</tr>
<tr>
<td>Number of full layers ((n_2))</td>
<td>Categorical</td>
<td>{0, 1, \ldots, 30}</td>
</tr>
<tr>
<td>Size of the full layer</td>
<td>Integer</td>
<td>{0, 1, \ldots, 500}</td>
</tr>
<tr>
<td>Dropout rate</td>
<td>Real</td>
<td>[0;1]</td>
</tr>
<tr>
<td>Activation function</td>
<td>Categorical</td>
<td>{ReLU, Sigmoid, Tanh}</td>
</tr>
</tbody>
</table>
# Hyperparameters for the optimizer (5 parameters)

<table>
<thead>
<tr>
<th>Optimizer</th>
<th>Update rule</th>
<th>Hyperparameter</th>
<th>Type</th>
<th>Scope</th>
</tr>
</thead>
</table>
| SGD | $d_{k+1} = \mu d_k + (1 - \delta) \nabla J(\Theta_{k+1})$  
$\Theta_{k+1} = \Theta_k - \eta d_{k+1}$ | $\eta$: Initial learning rate | Real | $[0;1]$ |
| |  | $\mu$: Momentum | Real | $[0;1]$ |
| |  | $\delta$: Dampening | Real | $[0;1]$ |
| |  | $\lambda$: Weight decay | Real | $[0;1]$ |
| Adagrad | $G_{k+1} = G_k + \nabla J(\Theta_{k+1})(\nabla J(\Theta_{k+1}))^\top$  
$\Theta_{k+1} = \Theta_k - \tau \frac{\eta}{\sqrt{\epsilon I + \text{diag}(G_{k+1})}} \nabla J(\Theta_k)$ | $\eta$: Initial learning rate | Real | $[0;1]$ |
| |  | $\tau$: Learning rate decay | Real | $[0;1]$ |
| |  | $\epsilon$: Smoothing constant | Real | $[0;1]$ |
| |  | $\lambda$: Weight decay | Real | $[0;1]$ |
| RMSProp | $d_{k+1} = \mu d_k + (1 - \mu) \nabla J(\Theta_{k+1})$  
$G_{k+1} = \gamma G_k + (1 - \gamma) \nabla J(\Theta_{k+1})(\nabla J(\Theta_{k+1}))^\top$  
$\Theta_{k+1} = \Theta_k - \frac{\eta}{\sqrt{\epsilon I + \text{diag}(G_{k+1})}} d_{k+1}$ | $\eta$: Initial learning rate | Real | $[0;1]$ |
| |  | $\mu$: Momentum | Real | $[0;1]$ |
| |  | $\gamma$: Forgetting factor | Real | $[0;1]$ |
| |  | $\lambda$: Weight decay | Real | $[0;1]$ |
| Adam | $d_{k+1} = \beta_1 d_k + (1 - \beta_1) \nabla J(\Theta_k)$  
$G_{k+1} = \beta_2 G_k + (1 - \beta_2) \nabla J(\Theta_{k+1})(\nabla J(\Theta_{k+1}))^\top$  
$\hat{d}_{k+1} = \frac{d_{k+1}}{1 - \beta_1^k}$  
$\hat{G}_{k+1} = \frac{G_{k+1}}{1 - \beta_2^k}$  
$\Theta_{k+1} = \Theta_k - \frac{\eta}{\sqrt{\epsilon I + \text{diag}(G_{k+1})}} \hat{d}_{k+1}$ | $\eta$: Initial learning rate | Real | $[0;1]$ |
| |  | $\beta_1$ | Real | $[0;1]$ |
| |  | $\beta_2$ | Real | $[0;1]$ |
| |  | $\lambda$: Weight decay | Real | $[0;1]$ |
Blocks of hyperparameters

- **Convolution block**: 2 convolutional layers with
  (number of output channels, kernel size, stride, padding, pooling) = (16, 5, 1, 1, 0) and (7, 3, 1, 1, 1):

  ![Convolution block diagram]

- **Fully connected block**: 3 fully connected layers with sizes of output = 1200, 512, 20:

  ![Fully connected block diagram]

- **Optimizer block**: SGD with learning rate = 0.1, momentum = 0.9, dampening = 1e-4, and weight decay = 0:

  ![Optimizer block diagram]
Blocks and neighbors

Illustration of how HyperNOMAD plays with neighbors of blocks

Current convolutional block

1st neighbor: Add a convolutional group
2nd neighbor: Remove a convolutional group

Current fully connected block

1st neighbor: Add a fully connected group
2nd neighbor: Remove a fully connected group

Current optimizer block

Neighbor optimizer block

<table>
<thead>
<tr>
<th>SGD</th>
<th>Adam</th>
<th>Adagrad</th>
<th>RMSProp</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0.2, 0.95, 1e-4, 0.03)</td>
<td>(0.1, 0.9, 0.99, 0)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example 1: Aircraft trajectories

Example 2: Radiographs

Example 3: HPO

References
Results on MNIST (left) and Fashion-MNIST (right)

Comparison between HyperNOMAD, TPE and RS (Hyperopt) when launched from the default starting point of HyperNOMAD
Results on CIFAR-10 (vs Hyperopt)

- Training with 40,000 images, validation/test on 10,000 images
- One evaluation (training+test) \(\sim 2\) hours
  (i7-6700@3.4 GHz, GeForce GTX 1070)

(a) Default starting point

(b) From a VGG architecture
Summary

- **Blackbox optimization** motivated by industrial applications
- Algorithmic features backed by mathematical convergence analyses and published in optimization journals
- **NOMAD**: Software package implementing MADS
- Open source; LGPL license
- **Features**: Constraints, biobjective, global optimization, surrogates, several types of variables, parallelism
- **HyperNOMAD**: Library for the HPO problem
- Fast support at nomad@gerad.ca
- NOMAD has become the baseline for benchmarking DFO/BBO algorithms
Introduction

The MADS algorithm

The NOMAD software package

Example 1: Aircraft takeoff trajectories

Example 2: Characterization of objects from radiographs

Example 3: Hyperparameters optimization

References
References I

Mixed variable optimization of a Load-Bearing thermal insulation system using a filter pattern search algorithm.

Optimization of noisy blackboxes with adaptive precision.

Nonsmooth optimization through Mesh Adaptive Direct Search and Variable Neighborhood Search.

Audet, C. and Dennis, Jr., J. (2006).

Audet, C. and Dennis, Jr., J. (2009).
A Progressive Barrier for Derivative-Free Nonlinear Programming.

Parallel Space Decomposition of the Mesh Adaptive Direct Search Algorithm.

Trade-off studies in blackbox optimization.
References II

StoMADS: Stochastic blackbox optimization using probabilistic estimates.

Derivative-Free and Blackbox Optimization.

The Mesh Adaptive Direct Search Algorithm for Granular and Discrete Variables.

Finding optimal algorithmic parameters using derivative-free optimization.

Multiobjective Optimization Through a Series of Single-Objective Formulations.

References III


Algorithm 909: NOMAD: Nonlinear Optimization with the MADS algorithm.

Le Digabel, S., Abramson, M., Audet, C., and Dennis, Jr., J. (2010).
Parallel Versions of the MADS Algorithm for Black-Box Optimization.
In Optimization days, Montreal. GERAD.

Statistical Surrogate Formulations for Simulation-Based Design Optimization.

Optimal, Environmentally-Friendly Departure Procedures for Civil Aircraft.