Blackbox optimization: Algorithms and applications

Sébastien Le Digabel

GERAD
GROUPE D'ÉTUDES ET DE RECHERCHE EN ANALYSE DES DÉCISIONS

POLYTECHNIQUE MONTRÉAL
TECHNOLOGICAL UNIVERSITY

CIRRELT, 2025-03-28

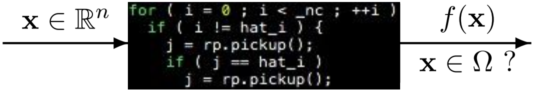
Presentation outline

Blackbox / Derivative-Free Optimization

We consider

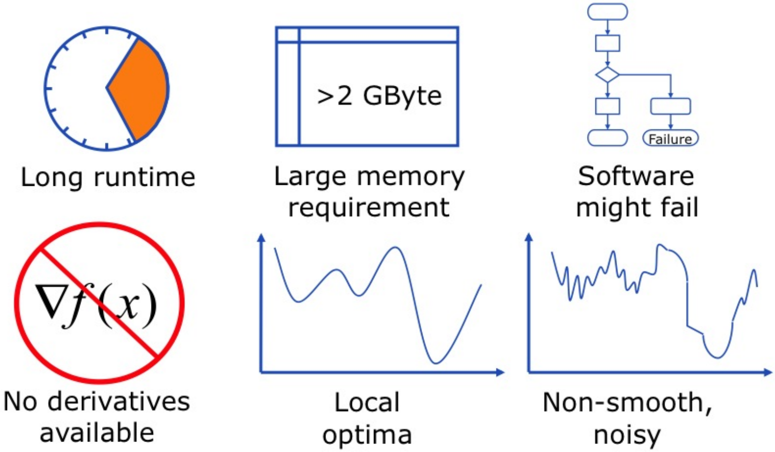
$$\min_{\mathbf{x}\in\Omega} \quad f(\mathbf{x})$$

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



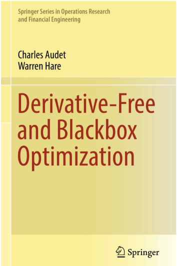
- Each call to the simulation may be expensive
- The simulation can fail
- Sometimes $f(\mathbf{x}) \neq f(\mathbf{x})$
- Derivatives are not available and cannot be approximated

Blackboxes as illustrated by a Boeing engineer



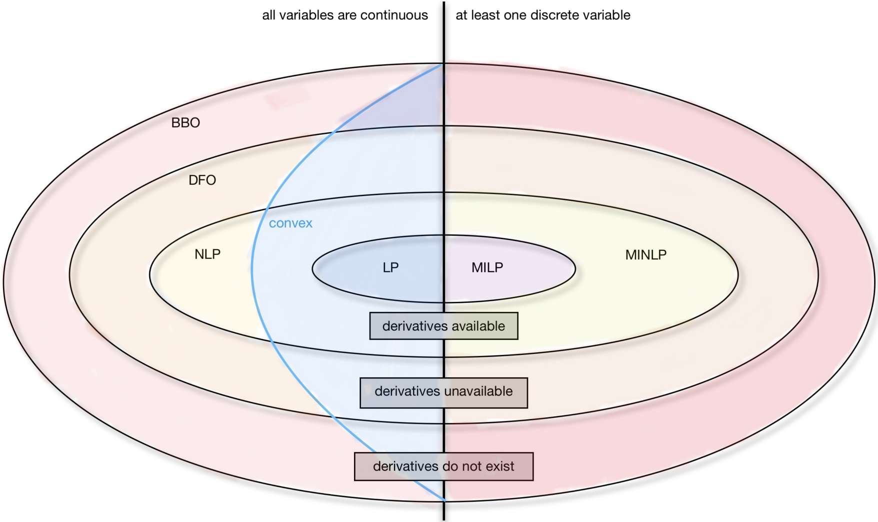
Terms

- *"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
  - Obj./constraints may be analytical functions of the outputs
  - Derivatives may be available (ex.: PDEs)
  - Sometimes referred as *Simulation-Based Optimization (SBO)*

Optimization: Global view



Aircraft takeoff trajectories

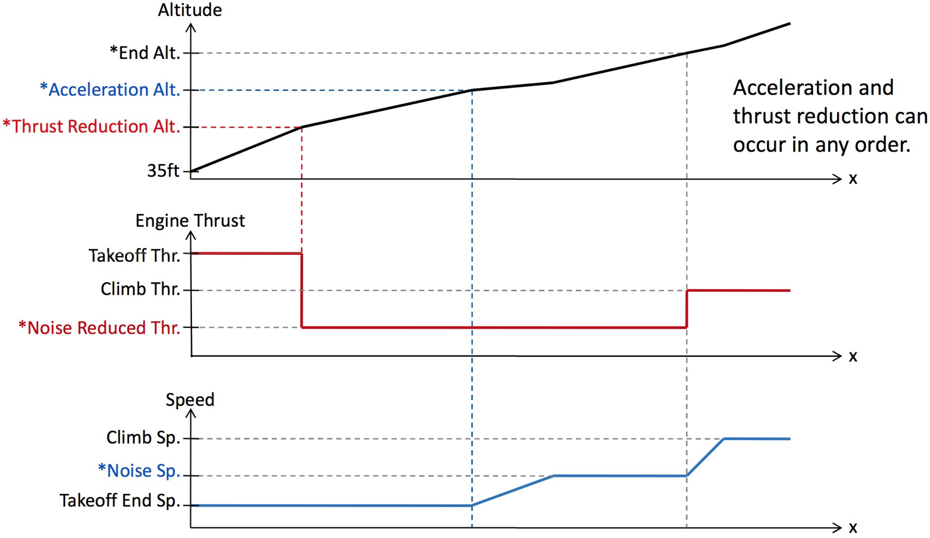
- [Torres et al., 2011]

- AIRBUS problem involving (among others): O. Babando, C. Bes, J. Chaptal, J.-B. Hiriart-Urruty, B. Talgorn, B. Tessier, and R. Torres
- Biobjective optimization problem

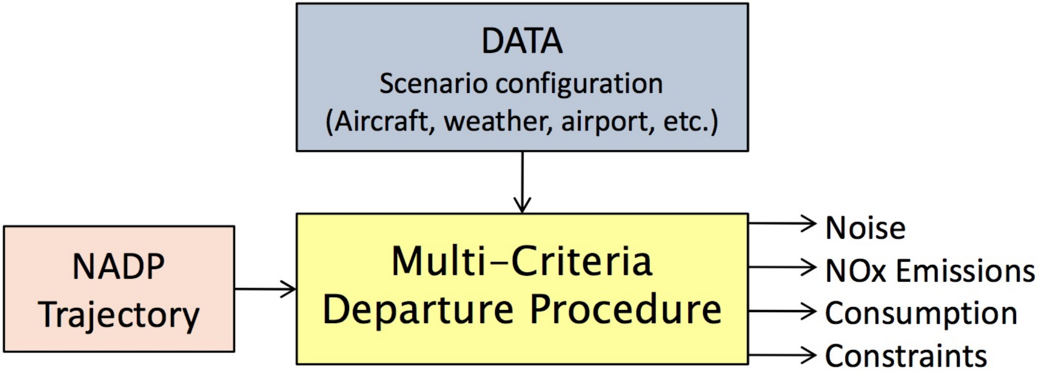
Definition of the optimization problem

- 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
- Variables define the NADP (Noise Abatement Departure Procedure): 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

Parametric Trajectory: 5 optimization variables (*): $x \in \mathbb{R}^5$



The blackbox: Multi-Criteria Departure Procedure

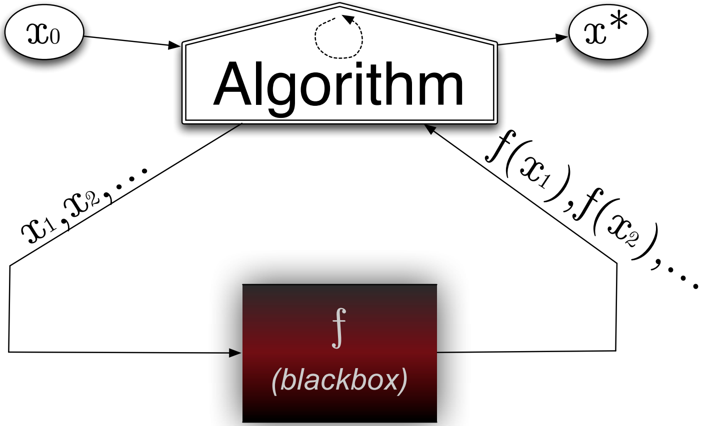


One evaluation $\simeq$ 2 seconds

Special features

- Must execute on different platforms including some old Solaris distributions
- The best trajectory parameters are returned to the pilot who enters them in the aircraft system manually → the less decimals the better
- Finite precision on optimization parameters: Discretization of optimization variables → granular variables [Audet et al., 2019]

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
- Deal with multiobjective optimization
- Deal with integer and categorical variables
- Easily exploit parallelism
- Have a publicly available implementation
- Have convergence properties ensuring first-order local optimality in the smooth case – otherwise why using it on more complicated problems?

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:
  - Design of experiments
  - Bayesian optimization: EGO algorithm based on surrogates and expected improvement
  - Still limited in terms of dimension
  - Does not natively handle constraints
  - Good to use these tools in conjonction with DFO methods
- Derivative-Free Optimization methods (DFO)

DFO methods

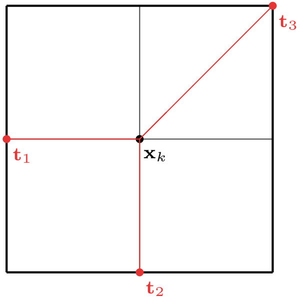
- **Model-based methods:**
  - Derivative-Free Trust-Region methods
  - Based on quadratic models or radial-basis functions
  - Use of a trust-region
  - Better for $\{$ DFO $\setminus$ BBO $\}$
  - Not resilient to noise and *hidden constraints*
  - Not easy to parallelize

- **Direct-search methods:**
  - Classical methods: Coordinate search, Nelder-Mead – the *other* simplex method
  - Modern methods: Generalized Pattern Search, Generating Set Search, 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

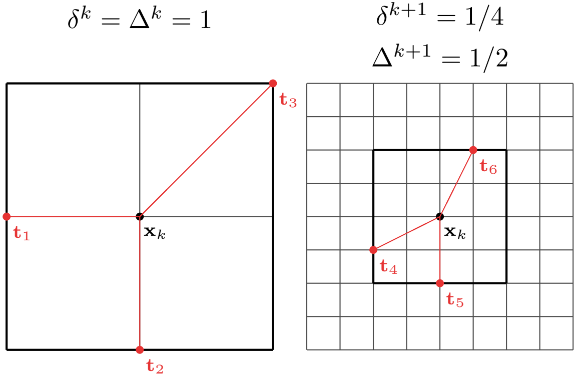
MADS illustration with $n = 2$: Poll step

$$\delta^k = \Delta^k = 1$$



poll trial points=$\{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3\}$

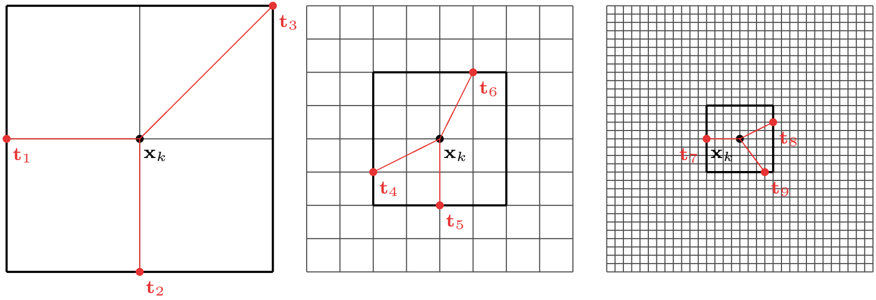
MADS illustration with $n = 2$: Poll step



poll trial points$=\{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3\}$ $= \{\mathbf{t}_4, \mathbf{t}_5, \mathbf{t}_6\}$

MADS illustration with $n = 2$: Poll step

$\delta^k = \Delta^k = 1$ $\qquad$ $\delta^{k+1} = 1/4$, $\Delta^{k+1} = 1/2$ $\qquad$ $\delta^{k+2} = 1/16$, $\Delta^{k+2} = 1/4$



poll trial points$=\{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3\}$ $\qquad = \{\mathbf{t}_4, \mathbf{t}_5, \mathbf{t}_6\}$ $\qquad = \{\mathbf{t}_7, \mathbf{t}_8, \mathbf{t}_9\}$

[0] Initializations $(\mathbf{x}_0, \delta^0)$

[1] Iteration k

  [1.1] Search (flexible part)

    select a finite number of mesh points

    evaluate candidates opportunistically

  [1.2] Poll (if Search failed) ("rigid" part)

    construct poll set $P_k = \{\mathbf{x}_k + \delta^k \mathbf{d} : \mathbf{d} \in D_k\}$

    sort(P_k)

    evaluate candidates opportunistically

[2] Updates

  if success

    $\mathbf{x}_{k+1} \leftarrow$ success point

    increase δ^k

  else

    $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k$

    decrease δ^k

  $k \leftarrow k + 1$, 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/Meta variables [Audet et al., 2023]
- Granular and discrete variables [Audet et al., 2019]
- Global optimization [Audet et al., 2008a]
- Parallelism [Le Digabel et al., 2010, Audet et al., 2008b]
- Multiobjective optimization [Audet et al., 2008c, Bigeon et al., 2021]
- Sensitivity analysis [Audet et al., 2012]
- Handling of stochastic blackboxes [Alarie et al., 2021, Audet et al., 2021]

Some MADS features

In the following slides, we focus on these MADS features:

- Constraints handling
- Use of surrogates

Constraints – with taxonomy of [Le Digabel and Wild, 2024]

Domain: $\Omega = \{\mathbf{x} \in \mathcal{X} : c_j(\mathbf{x}) \leq 0, j \in J\} \subset \mathbb{R}^n$

- $\mathcal{X}$ corresponds to unrelaxable constraints

Cannot be violated;

Example: $x > 0$ when $\log x$ is used inside the simulation

Constraints – with taxonomy of [Le Digabel and Wild, 2024]

Domain: $\Omega = \{\mathbf{x} \in \mathcal{X} : c_j(\mathbf{x}) \leq 0, j \in J\} \subset \mathbb{R}^n$

- $\mathcal{X}$ corresponds to unrelaxable constraints
- $c_j(\mathbf{x}) \leq 0$: Relaxable and quantifiable constraints

May be violated at intermediate designs

$c_j(\mathbf{x})$ measures the violation

Example: cost $\leq$ budget

Constraints – with taxonomy of [Le Digabel and Wild, 2024]

Domain: $\Omega = \{\mathbf{x} \in \mathcal{X} : c_j(\mathbf{x}) \leq 0, j \in J\} \subset \mathbb{R}^n$

- $\mathcal{X}$ corresponds to unrelaxable constraints
- $c_j(\mathbf{x}) \leq 0$: Relaxable and quantifiable constraints
- Hidden constraints

  when the simulation fails, even for points in Ω

  Example:

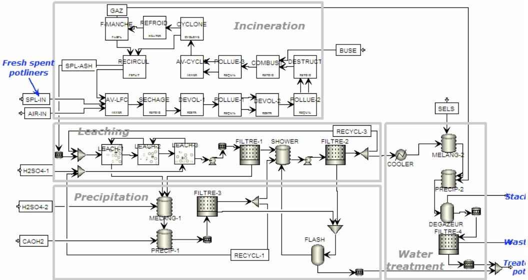
```
Segmentation fault
Bus error
ERROR 42
DIVISION BY ZERO
```

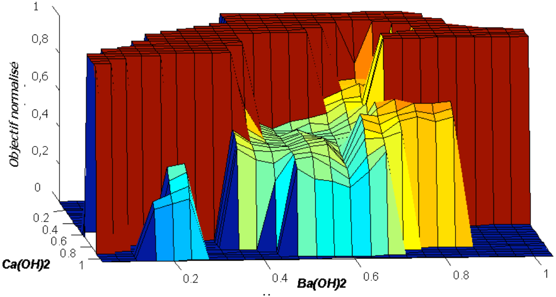
Constraints – with taxonomy of [Le Digabel and Wild, 2024]

Domain: $\Omega = \{\mathbf{x} \in \mathcal{X} : c_j(\mathbf{x}) \leq 0, j \in J\} \subset \mathbb{R}^n$

- $\mathcal{X}$ corresponds to unrelaxable constraints
- $c_j(\mathbf{x}) \leq 0$: Relaxable and quantifiable constraints
- Hidden constraints

Example: Chemical process:





7 variables, 4 constraints. The ASPEN software fails on 43% of the calls

Three strategies to deal with constraints

- Extreme barrier (EB)

Treats the problem as being unconstrained,
by replacing the objective function $f(\mathbf{x})$ by

$$f_\Omega(\mathbf{x}) := \begin{cases} f(\mathbf{x}) & \text{if } \mathbf{x} \in \Omega \\ \infty & \text{otherwise} \end{cases}$$

The problem

$$\min_{\mathbf{x} \in \mathbb{R}^n} f_\Omega(\mathbf{x})$$

is then solved.
Remark: this strategy can also be applied to a priori constraints in order to avoid the costly evaluation of $f(\mathbf{x})$

Three strategies to deal with constraints

- Extreme barrier (EB)
- Progressive barrier (PB)

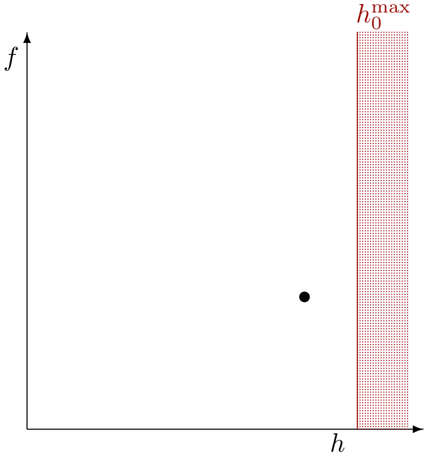
  Defined for relaxable and quantifiable constraints.
  As in the filter methods of Fletcher and Leyffer, it uses the non-negative constraint violation function $h : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$

$$h(\mathbf{x}) \ := \begin{cases} \displaystyle\sum_{j \in J} (\max(c_j(\mathbf{x}), 0))^2 & \text{if } \ \mathbf{x} \in \mathcal{X} \\ \infty & \text{otherwise} \end{cases}$$

  At iteration k, points with $h(\mathbf{x}) > h_k^{\max}$ are rejected by the algorithm, and $h_k^{\max}$ decreases toward 0 as $k \to \infty$

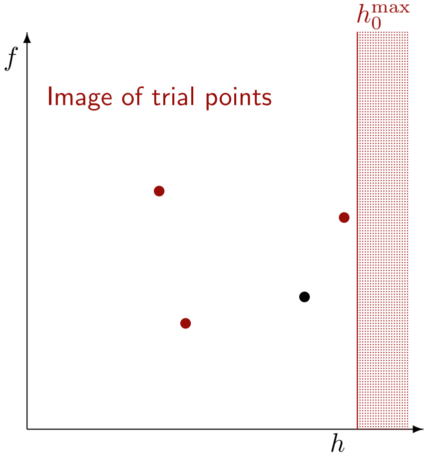
Three strategies to deal with constraints

- Extreme barrier (EB)
- Progressive barrier (PB)



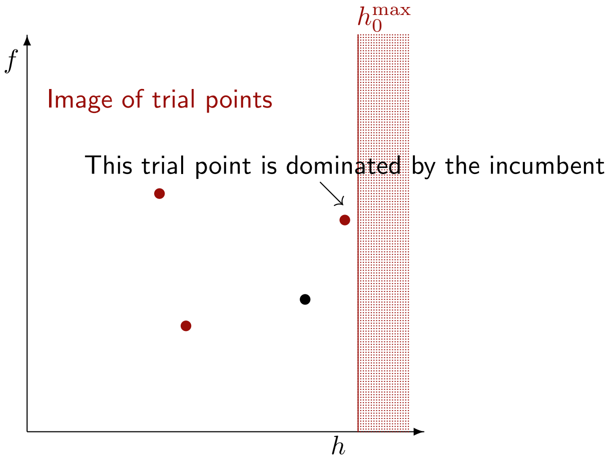
Three strategies to deal with constraints

- Extreme barrier (EB)
- Progressive barrier (PB)



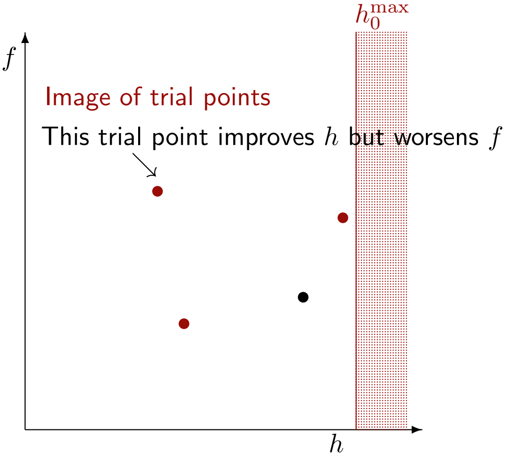
Three strategies to deal with constraints

- Extreme barrier (EB)
- Progressive barrier (PB)



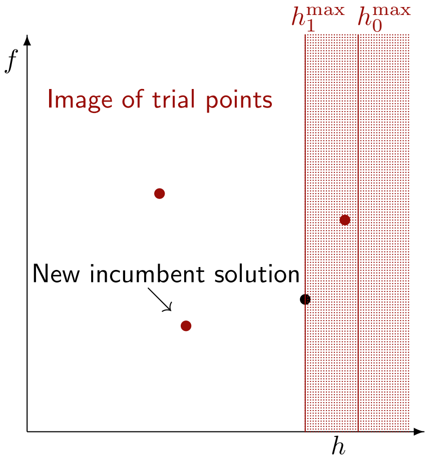
Three strategies to deal with constraints

- Extreme barrier (EB)
- Progressive barrier (PB)



Three strategies to deal with constraints

- Extreme barrier (EB)
- Progressive barrier (PB)



Three strategies to deal with constraints

- Extreme barrier (EB)
- Progressive barrier (PB)
- Progressive-to-Extreme Barrier (PEB)

  Initially treats a relaxable+quantifiable constraint by the progressive barrier. Then, if polling around the infeasible poll center generates a new infeasible incumbent that satisfies a constraint violated by the poll center, then that constraint moves from being treated by the progressive barrier to the extreme barrier

Static versus dynamic surrogates

- Static surrogate: A cheaper model defined a priori by the user. It is used as a blackbox. Typically a simplified physics model. Variable fidelity may be considered.
- Dynamic surrogate: Model managed by the algorithm, based on past evaluations. It can be periodically updated.

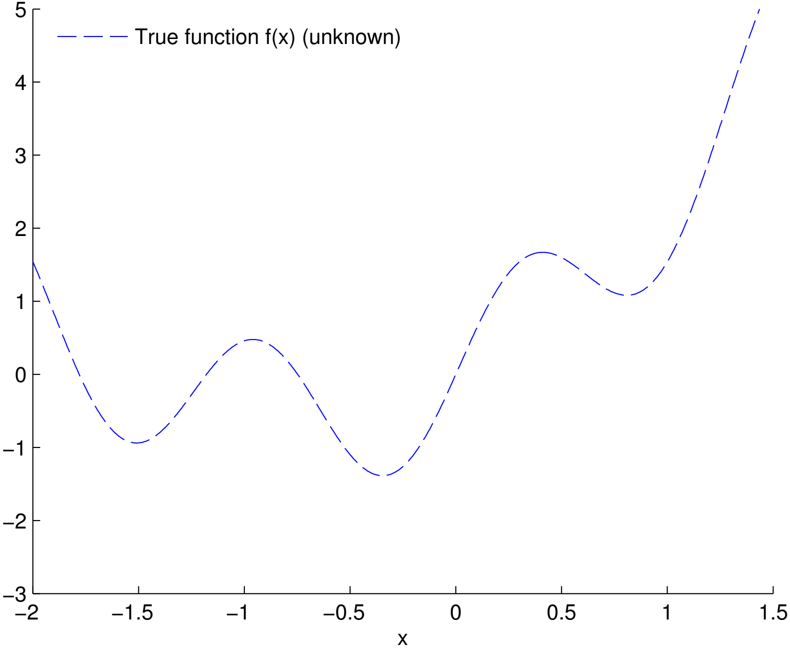
In the remaining, we focus on dynamic surrogates

Surrogate-assisted optimization [Booker et al., 1999]

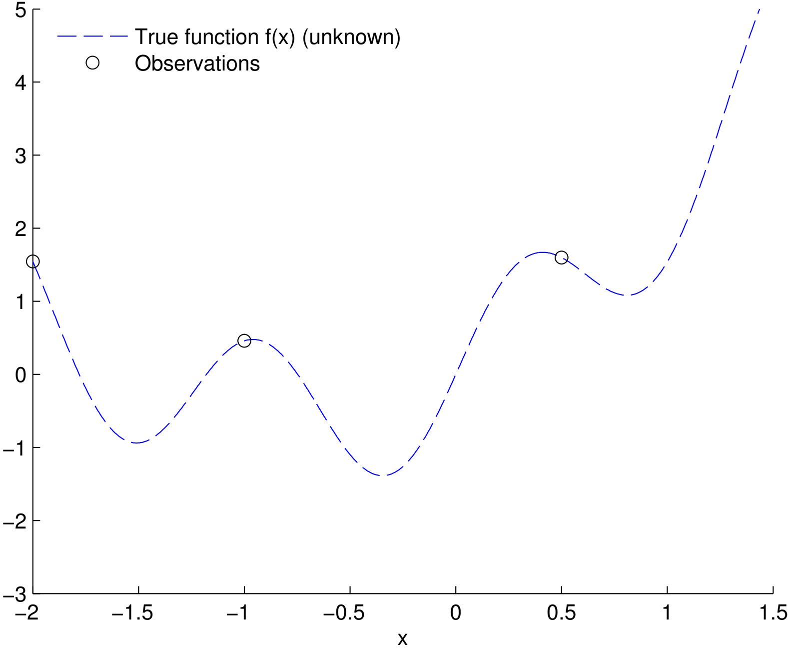
1. Use $[\mathbf{X}, f(\mathbf{X})]$ to build a surrogate $\hat{f}$ of the function f
2. Find $\mathbf{x}_S \in \underset{\mathbf{x}}{\operatorname{argmin}} \ \hat{f}(\mathbf{x})$ (or minimize another criteria such as the EI)
3. Evaluate $f(\mathbf{x}_S)$
4. $\mathbf{X} \leftarrow \mathbf{X} \cup \{\mathbf{x}_S\}$
5. Go back to Step **1.**

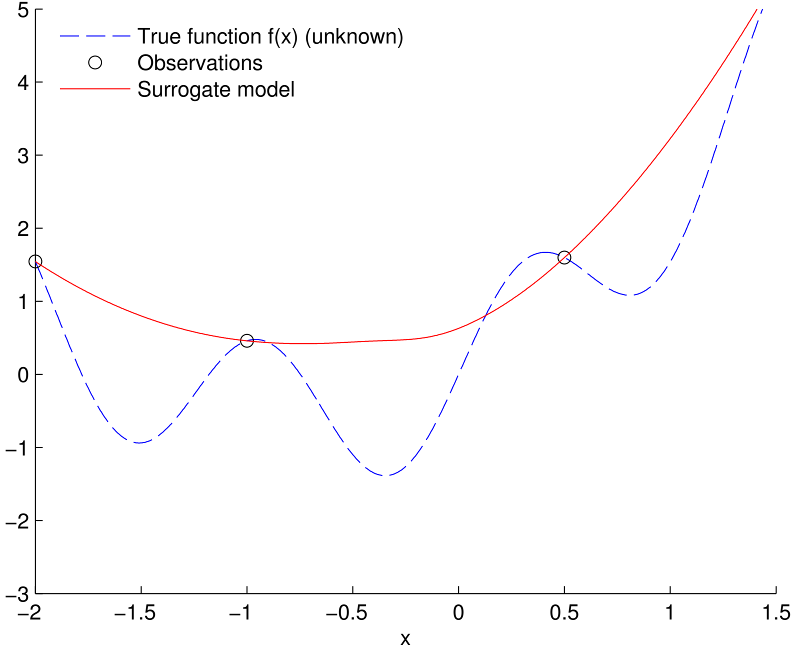
For constrained problems the same method can be used for constrained problems:

- Build the models of the constraints
- $\mathbf{x}_S \leftarrow$ minimizer of $\hat{f}$ subject to the constraints $\hat{c}_j \leq 0$, $j = 1, 2, \ldots, m$

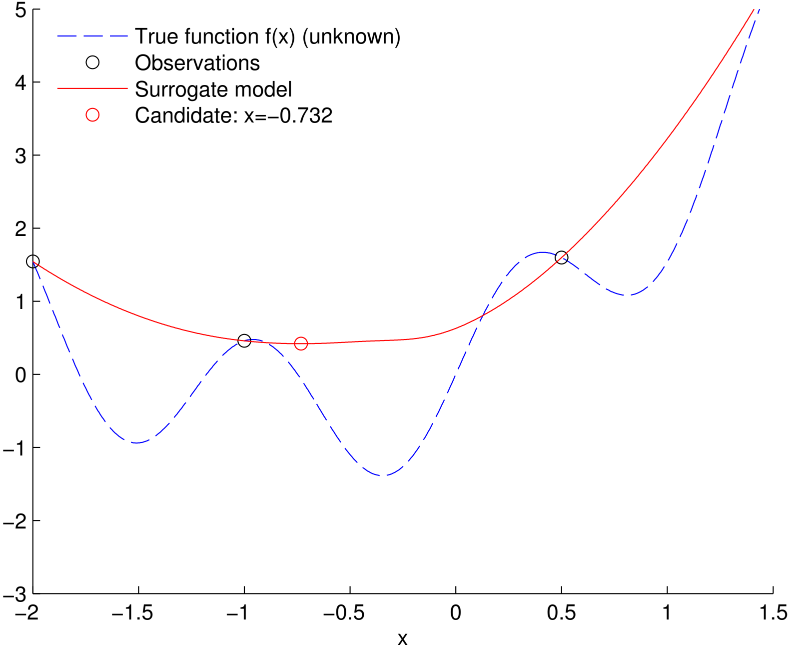


True function f(x) (unknown)
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x

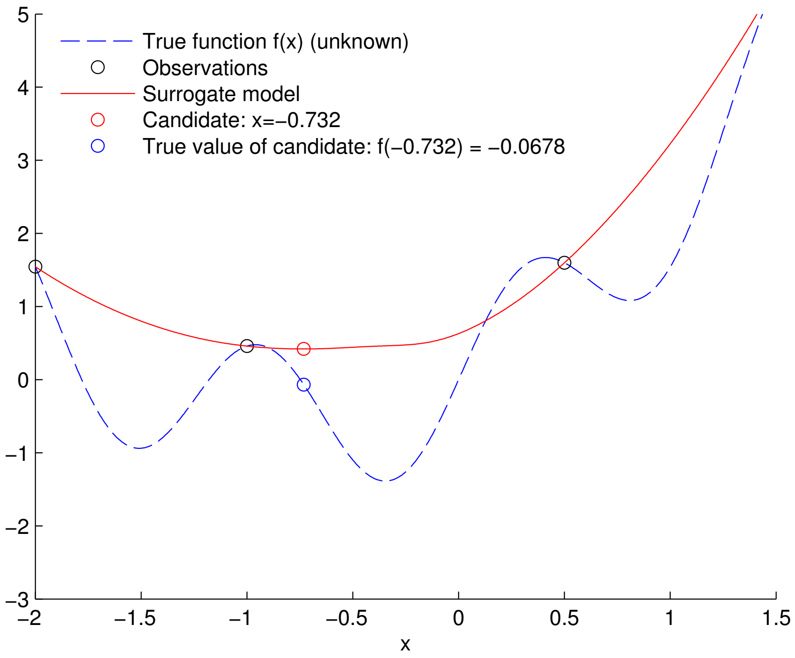

True function f(x) (unknown)
Observations
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x

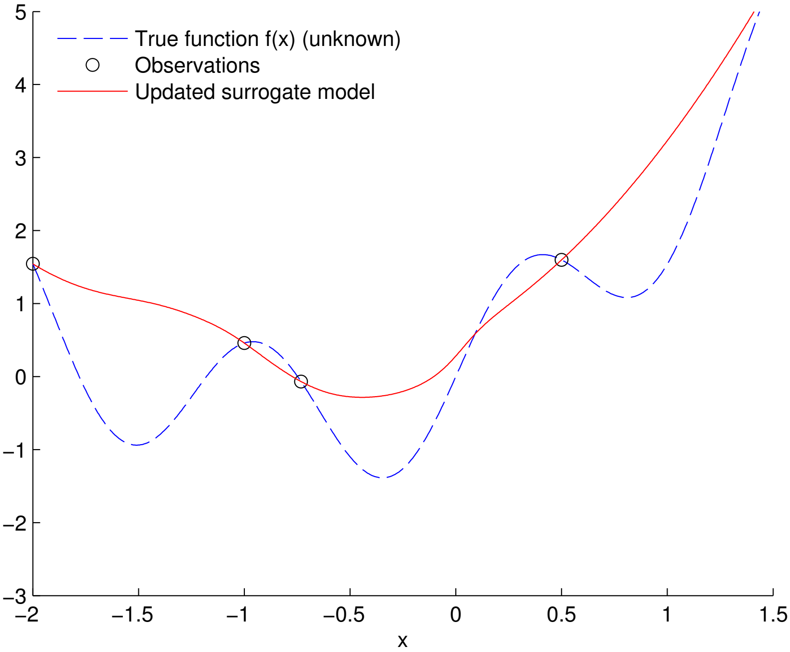


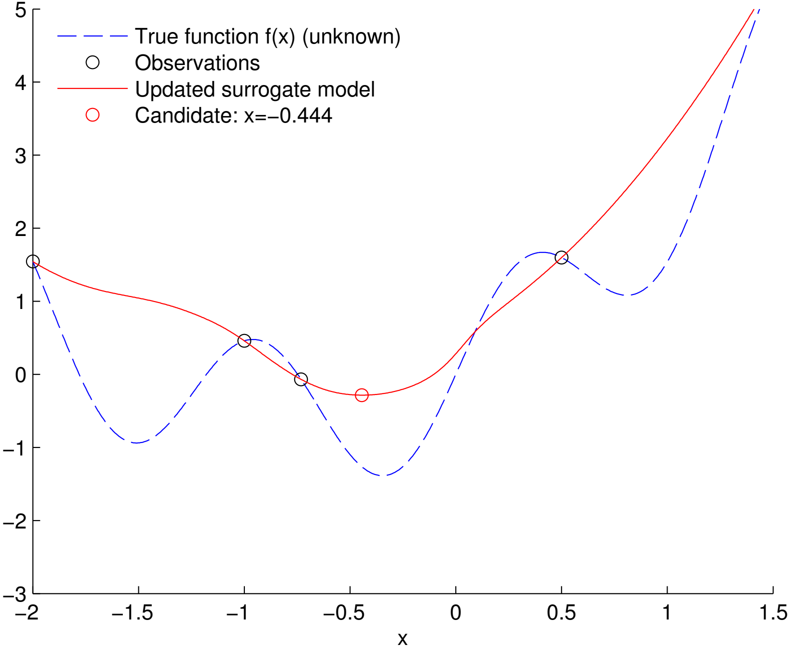
True function f(x) (unknown)
Observations
Surrogate model
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x

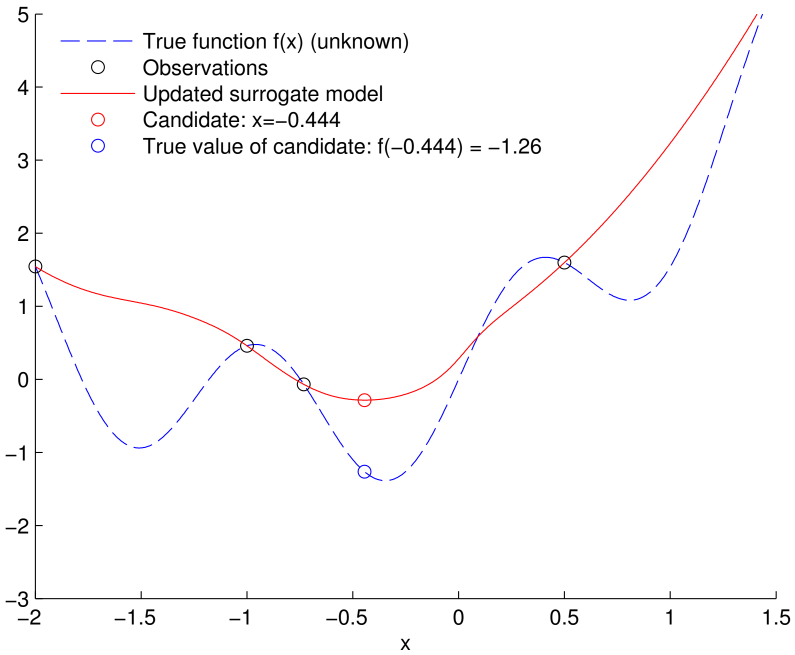


True function f(x) (unknown)
Observations
Surrogate model
Candidate: x=−0.732
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x

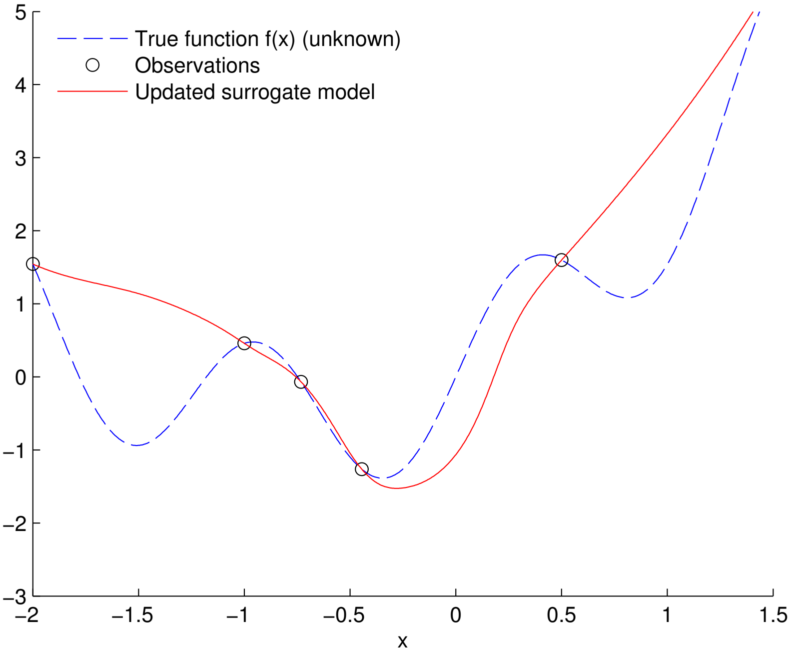

True function f(x) (unknown)
Observations
Surrogate model
Candidate: x=−0.732
True value of candidate: f(−0.732) = −0.0678
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x


True function f(x) (unknown)
Observations
Updated surrogate model
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x

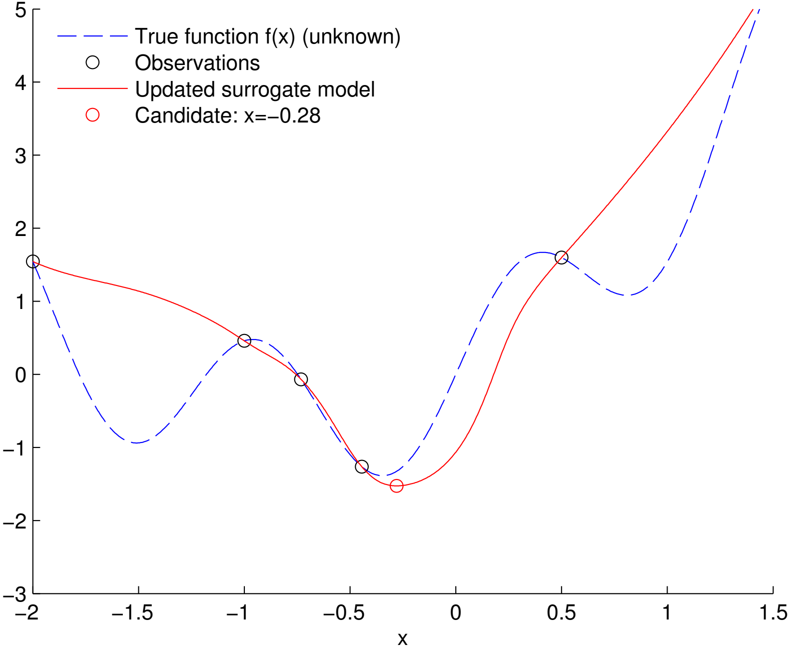

True function f(x) (unknown)
Observations
Updated surrogate model
Candidate: x=−0.444
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x

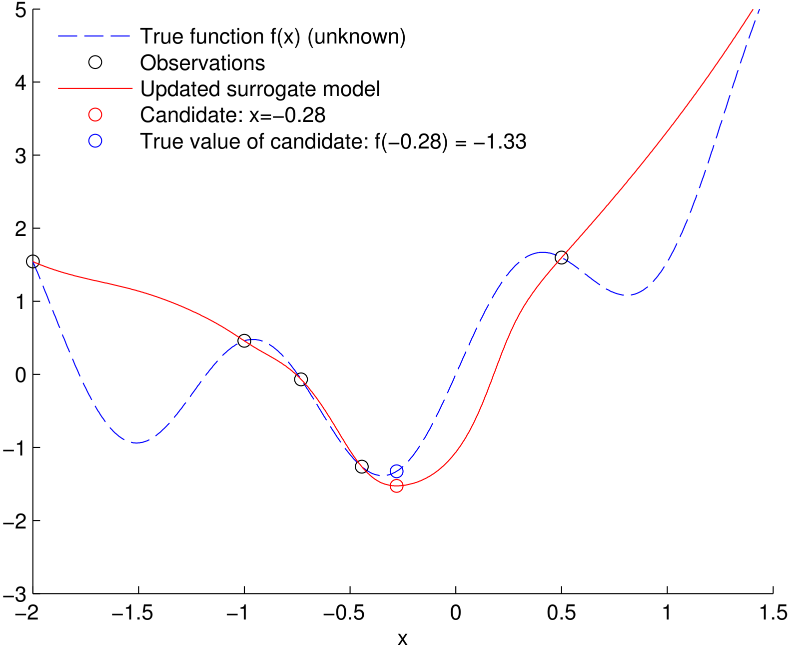


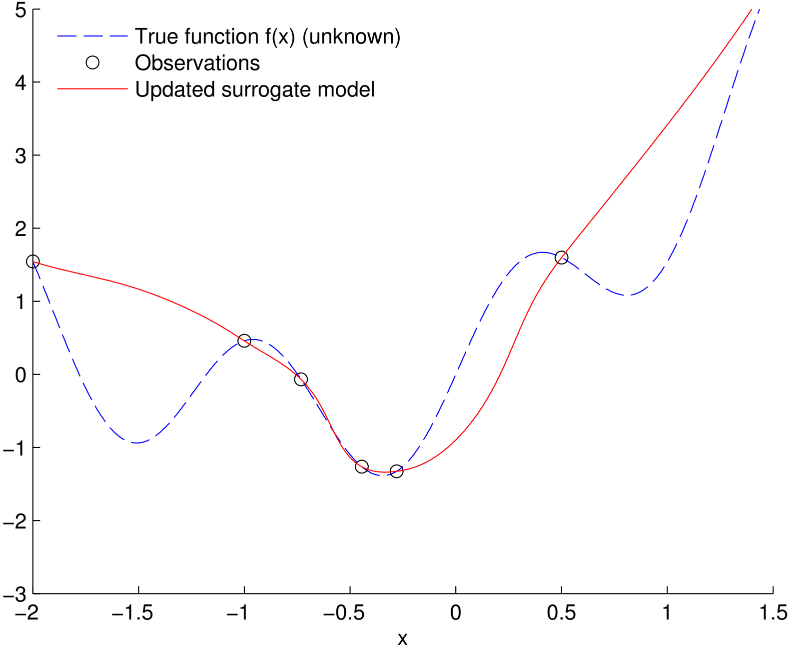
True function f(x) (unknown)
Observations
Updated surrogate model
Candidate: x=−0.444
True value of candidate: f(−0.444) = −1.26
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x

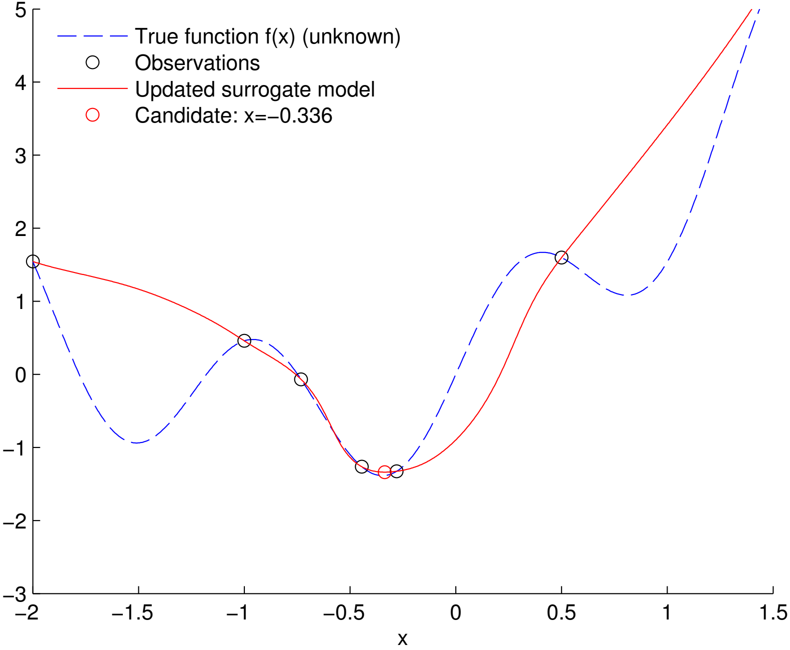


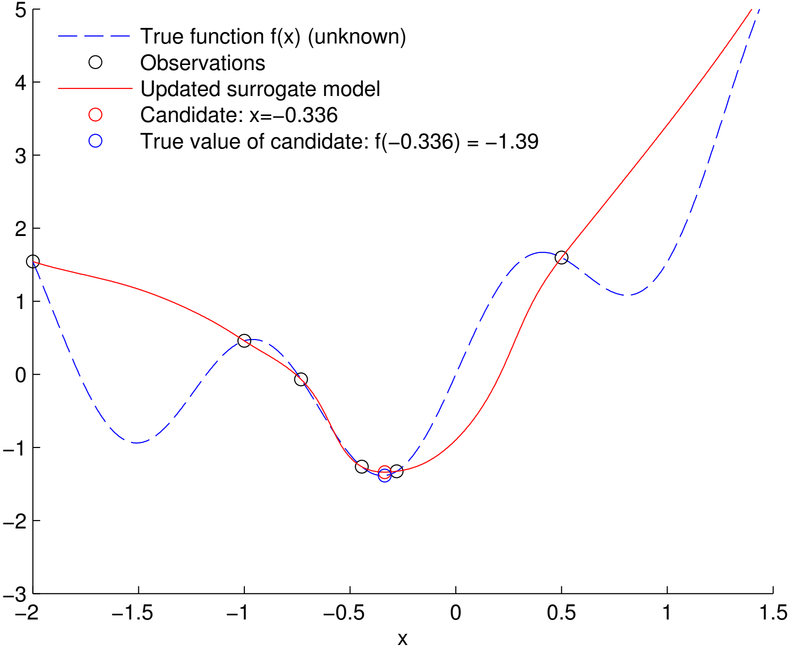
True function f(x) (unknown)
Observations
Updated surrogate model
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x

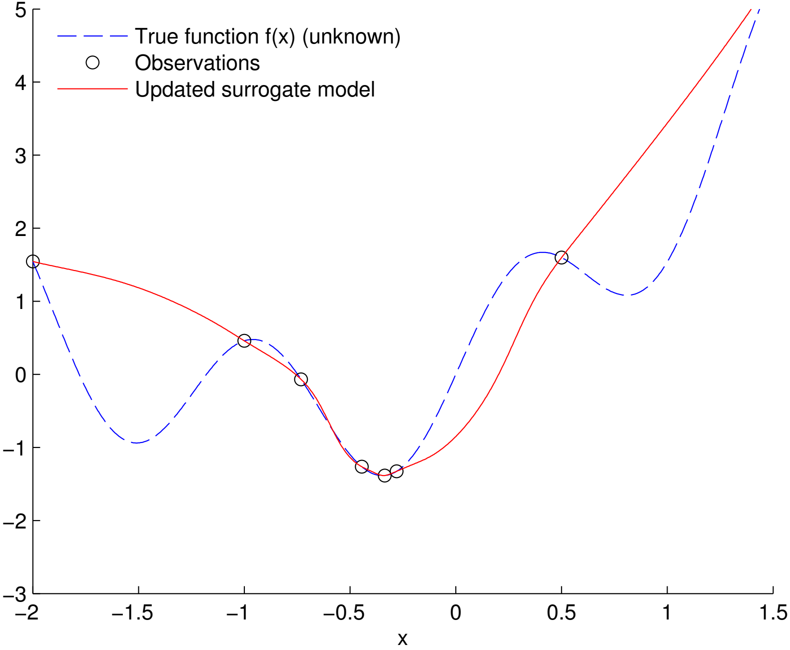

True function f(x) (unknown)
Observations
Updated surrogate model
Candidate: x=−0.28
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x

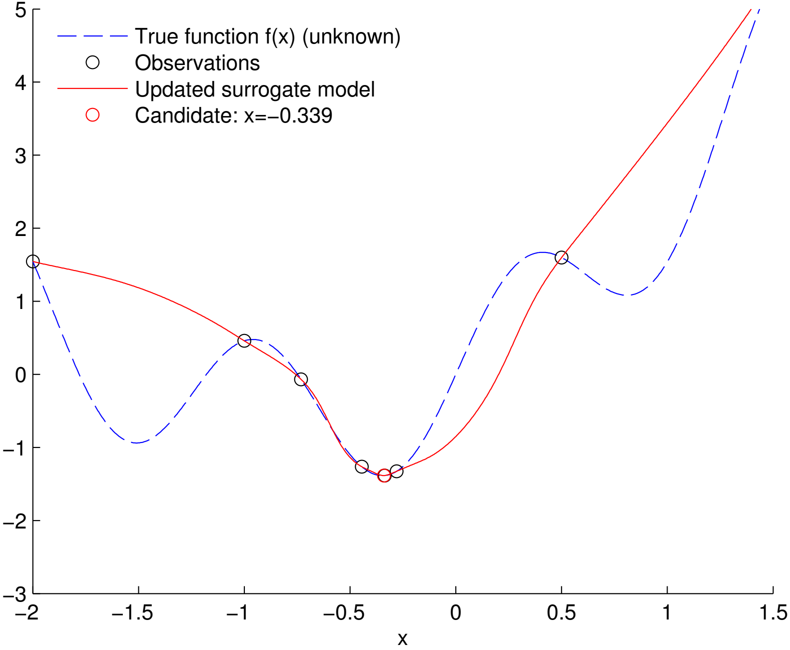

True function f(x) (unknown)
Observations
Updated surrogate model
Candidate: x=−0.28
True value of candidate: f(−0.28) = −1.33
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x


True function f(x) (unknown)
Observations
Updated surrogate model
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x


True function f(x) (unknown)
Observations
Updated surrogate model
Candidate: x=−0.336
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x


True function f(x) (unknown)
Observations
Updated surrogate model
Candidate: x=−0.336
True value of candidate: f(−0.336) = −1.39
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x


True function f(x) (unknown)
Observations
Updated surrogate model
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x


True function f(x) (unknown)
Observations
Updated surrogate model
Candidate: x=−0.339
5
4
3
2
1
0
−1
−2
−3
−2
−1.5
−1
−0.5
0
0.5
1
1.5
x

Surrogate-assisted optimization in MADS

1. Initialization:
   - Initial design $(\mathbf{x}_0)$
   - Initial mesh and poll sizes $(\delta^0,\ \Delta^0)$

2. Search
   - Build the **surrogates** $\hat{f}$ and $\{\hat{c}_j\}_{j=1,2,\ldots,m}$
   - $\mathbf{x}_S \leftarrow$ solution of the surrogate problem, projected on the current mesh
   - If $\mathbf{x}_S$ is a success, repeat the search

3. Poll
   - Construct the poll candidates
   - Use the **surrogates** to order the poll candidates
   - Evaluate the poll candidates *opportunistically*

4. If no stopping criteria is met, go back to Step **2.**

What is a good model for surrogate-assisted optimization

- Good model of the objective f: respects the **order** between two candidates:

$$f(\mathbf{x}) \leq f(\mathbf{x}') \Leftrightarrow \hat{f}(\mathbf{x}) \leq \hat{f}(\mathbf{x}') \text{ for all } \mathbf{x}, \mathbf{x}' \in \mathcal{X}$$

- Good model of a constraint c_j: respects the **sign** of the function:

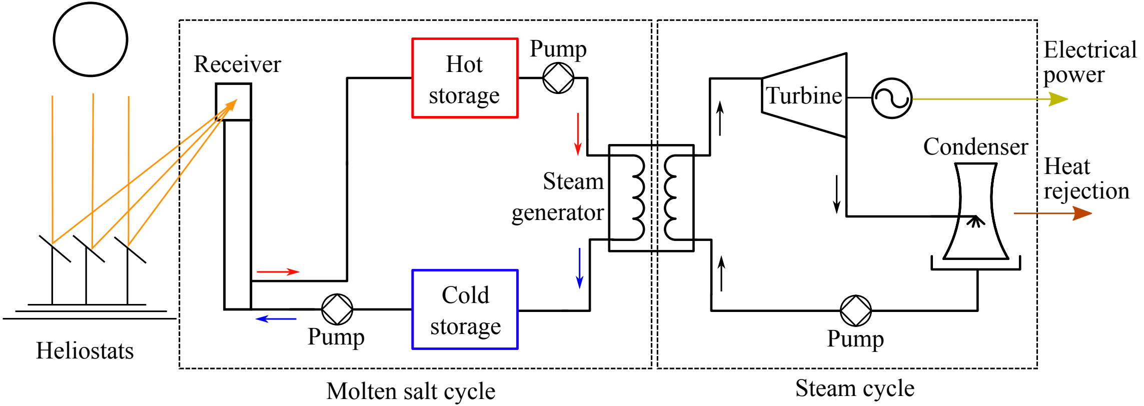
$$c_j(\mathbf{x}) \leq 0 \Leftrightarrow \hat{c}_j(\mathbf{x}) \leq 0 \text{ for all } \mathbf{x} \in \mathcal{X}$$

CSP power plant with molten salt thermal energy storage

- Work published in [Andrés-Thió et al., 2025]
- A large number of mirrors (heliostats) reflects solar radiation on a receiver at the top of a tower
- The heat collected from the concentrated solar flux is removed from the receiver by a stream of molten salt
- Hot molten salt is then used to feed thermal power to a conventional power block
- The photo shows the Thémis CSP power plant, the first built with this design

Source: https://commons.wikimedia.org/wiki/File:Themis_2.jpg

System dynamics



Ten instances

Instance	# of variables			# of obj.	# of constraints			# of stoch. outputs	Static
	cont.	discr. (cat.)	n	p	simu.	a priori (lin.)	m	(obj. or constr.)	surrogate
solar1	8	1 (0)	9	1	2	3 (2)	5	1	no
solar2[1]	12	2 (0)	14	1	9	4 (2)	13	3	yes
solar3	17	3 (1)	20	1	8	5 (3)	13	5	yes
solar4	22	7 (1)	29	1	9	7 (5)	16	6	yes
solar5	14	6 (1)	20	1	8	4 (3)	12	0	no
solar6	5	0 (0)	5	1	6	0 (0)	6	0	no
solar7	6	1 (0)	7	1	4	2 (1)	6	3	yes
solar8	11	2 (0)	13	2	4	5 (3)	9	3	yes
solar9	22	7 (1)	29	2	10	7 (5)	17	6	yes
solar10[2]	5	0 (0)	5	1	0	0 (0)	0	0	yes

[1] analytic objective
[2] unconstrained

Features for BBO benchmarking

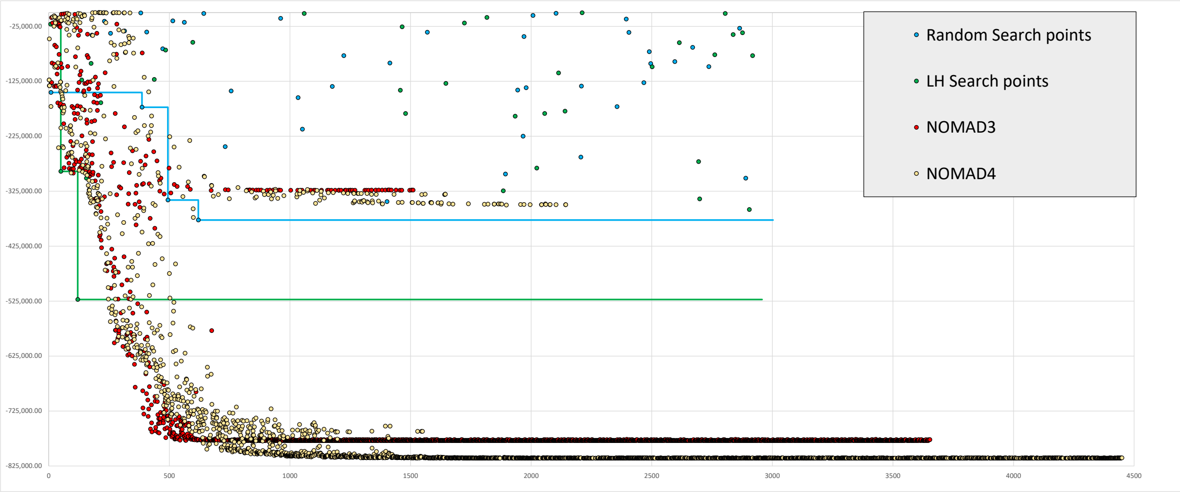
- Several numerical methods: real-world blackbox
- Reproducibility accros all platforms
- Continuous and discrete variables
- Different types of constraints (quantifiable, relaxable, a priori, hidden)
- Stochastic and deterministic outputs
- Static surrogates with variable fidelity
- Number of replications is controlable

Feasibility with sampling and NOMAD

Instance	LH search (10k points)		NOMAD3		
	satisf. ap constr.	feas. pts	satisf. ap constr.	feas. pts	number of eval.
solar1	30%	0.35%	96%	74%	3,792
solar2	0%	0%	97%	0%	1,635
solar3	0.49%	0%	99%	9%	30,525
solar4	0%	0%	83%	0%	44,303
solar5	0%	0%	83%	59%	3,405
solar6	90%	5%	99%	0%	3,539
solar7	2%	1%	74%	72%	2,224
solar8	1%	0.03%			
solar9	1%	0%			

there has been no violation of hidden constraints during the construction of this table

Optimization on solar1



NOMAD (Nonlinear Optimization with MADS)

- C++ implementation of the MADS algorithm [Audet and Dennis, Jr., 2006]
- Standard C++. Runs on Linux, Mac OS X and Windows
- Parallel versions
- MATLAB versions; Multiple interfaces (Python, Julia, etc.)
- Open and free – LGPL license
- Download at https://www.gerad.ca/nomad
- Support at nomad@gerad.ca
- Related articles in TOMS [Le Digabel, 2011] and [Audet et al., 2022]

Main functionalities (1/2)

- Single or biobjective optimization
- Variables:
  - Continuous, integer, binary, categorical, granular
  - Periodic
  - Fixed
  - Groups of variables
- Searches:
  - Latin-Hypercube
  - Variable Neighborhood Search
  - Nelder-Mead Search
  - Quadratic models
  - Statistical surrogates
  - User search

Main functionalities (2/2)

- Constraints treated with 4 different methods:
  - Progressive Barrier (default)
  - Extreme Barrier
  - Progressive-to-Extreme Barrier
  - Filter method
- Several direction types:
  - Coordinate directions
  - LT-MADS
  - OrthoMADS
  - Hybrid combinations
- Sensitivity analysis

→ default values for all parameters

→ all items correspond to published or submitted papers

Blackbox conception (batch mode)

- Command-line program that takes in argument a file containing $\mathbf{x}$, and displays the values of $f(\mathbf{x})$ and the $c_j(\mathbf{x})$'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
```

```
[iota ~/Desktop/2018_UQAC_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 AFOSR, Exxon Mobil, Hydro Québec, 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 AFOSR 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 {

        BBE     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): ( 1.000000013 1.000000048 0.9999999797 0.999999992 -4 ) h=1.10134e-13 f=-4
best feasible solution                   : ( 1 1 1 1 -4 ) h=0 f=-4
```

Introduction

Example 1: Aircraft takeoff trajectories

The MADS algorithm

Example 2: Solar thermal power plant

The NOMAD software package

Summary and references

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
- Fast support at nomad@gerad.ca
- NOMAD has become a baseline for benchmarking DFO algorithms

References I

Alarie, S., Audet, C., Bouchet, P.-Y., and Le Digabel, S. (2021).
Optimisation of stochastic blackboxes with adaptive precision.
SIAM Journal on Optimization, 31(4):3127–3156.

Andrés-Thió, N., Audet, C., Diago, M., Gheribi, A., Le Digabel, S., Lebeuf, X., Lemyre Garneau, M., and Tribes, C. (2025).
`solar`: A solar thermal power plant simulator for blackbox optimization benchmarking.
Technical Report G-2024-37, Les cahiers du GERAD.
To appear in *Optimization and Engineering*.

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

Audet, C. and Dennis, Jr., J. (2006).
Mesh Adaptive Direct Search Algorithms for Constrained Optimization.
SIAM Journal on Optimization, 17(1):188–217.

Audet, C. and Dennis, Jr., J. (2009).
A Progressive Barrier for Derivative-Free Nonlinear Programming.
SIAM Journal on Optimization, 20(1):445–472.

Audet, C., Dennis, Jr., J., and Le Digabel, S. (2008b).
Parallel Space Decomposition of the Mesh Adaptive Direct Search Algorithm.
SIAM Journal on Optimization, 19(3):1150–1170.

References II

Audet, C., Dennis, Jr., J., and Le Digabel, S. (2012).
Trade-off studies in blackbox optimization.
Optimization Methods and Software, 27(4–5):613–624.

Audet, C., Dzahini, K., Kokkolaras, M., and Le Digabel, S. (2021).
Stochastic mesh adaptive direct search for blackbox optimization using probabilistic estimates.
Computational Optimization and Applications, 79(1):1–34.

Audet, C., Hallé-Hannan, E., and Le Digabel, S. (2023).
A General Mathematical Framework for Constrained Mixed-variable Blackbox Optimization Problems with Meta and Categorical Variables.
Operations Research Forum, 4(12).

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

Audet, C., Le Digabel, S., Rochon Montplaisir, V., and Tribes, C. (2022).
Algorithm 1027: NOMAD version 4: Nonlinear optimization with the MADS algorithm.
ACM Transactions on Mathematical Software, 48(3):35:1–35:22.

Audet, C., Le Digabel, S., and Tribes, C. (2019).
The Mesh Adaptive Direct Search Algorithm for Granular and Discrete Variables.
SIAM Journal on Optimization, 29(2):1164–1189.

References III

Audet, C., Savard, G., and Zghal, W. (2008c).
Multiobjective Optimization Through a Series of Single-Objective Formulations.
SIAM Journal on Optimization, 19(1):188–210.

Bigeon, J., Le Digabel, S., and Salomon, L. (2021).
DMulti-MADS: Mesh adaptive direct multisearch for bound-constrained blackbox multiobjective optimization.
Computational Optimization and Applications, 79(2):301–338.

Booker, A., Dennis, Jr., J., Frank, P., Serafini, D., Torczon, V., and Trosset, M. (1999).
A Rigorous Framework for Optimization of Expensive Functions by Surrogates.
Structural and Multidisciplinary Optimization, 17(1):1–13.

Le Digabel, S. (2011).
Algorithm 909: NOMAD: Nonlinear Optimization with the MADS algorithm.
ACM Transactions on Mathematical Software, 37(4):44:1–44:15.

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*, Montréal.
Slides available at `https://www.gerad.ca/Sebastien.Le.Digabel/talks/2010_JOPT_25mins.pdf`.

Le Digabel, S. and Wild, S. (2024).
A taxonomy of constraints in black-box simulation-based optimization.
Optimization and Engineering, 25(2):1125–1143.

References IV

Talgorn, B., Le Digabel, S., and Kokkolaras, M. (2015).
Statistical Surrogate Formulations for Simulation-Based Design Optimization.
Journal of Mechanical Design, 137(2):021405–1–021405–18.

Torres, R., Bès, C., Chaptal, J., and Hiriart-Urruty, J.-B. (2011).
Optimal, Environmentally-Friendly Departure Procedures for Civil Aircraft.
Journal of Aircraft, 48(1):11–22.