solar: A solar thermal power plant simulator for blackbox optimization benchmarking

Sébastien Le Digabel

GERAD
GROUP FOR RESEARCH IN DECISION ANALYSIS

POLYTECHNIQUE MONTRÉAL
TECHNOLOGICAL UNIVERSITY

SIAM CSE23, 2023-03-02

Presentation outline

Contributors

- This work is based on the MSc thesis of Mathieu Lemyre Garneau [Lemyre Garneau, 2015]
- The other contributors are
  - Charles Audet
  - Miguel Diago
  - Aïmen Gheribi
  - Mona Jeunehomme
  - Xavier Lebeuf
  - Viviane Rochon Montplaisir
  - Bastien Talgorn
  - Nicolau Andres Thio
  - Christophe Tribes
- MLG, MD, and AG, combine several expertises in concentrated solar power (CSP)

Context: Blackbox Optimization (BBO)

$$\min_{\mathbf{x} \in \mathcal{X}} \quad F(\mathbf{x}) \text{ s.t. } \mathbf{x} \in \Omega = \{\mathbf{x} \in \mathcal{X} : c_j(\mathbf{x}) \leq 0, j = 1, 2, \ldots, m\}$$

$\mathcal{X}$ is a n-dimensional space, F can have $p = 1$ or $p = 2$ components, and the evaluations of F and the c_j's are provided by a blackbox:

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

Issues with BBO benchmarking

- Benchmarking must consider many problems, which is problematic in BBO
- Testing on true applications is difficult because
  - Evaluations are time-consuming
  - Codes are confidential
  - Codes depend on in-house or expensive libraries
  - Codes are difficult to install
  - The original designers are no longer available
- This results in the use of collections of artificial problems that are based on inexpensive analytical functions
- These collections are necessary, given the lack of true applications, but they are not sufficient: This leads to biased hierarchies of solvers that are useless for practitioners

Objectives of this work

Provide a realistic application for "true" BBO benchmarking, that

- includes numerical simulations
- is easy to install (stand-alone, standard code)
- is multiplatform
- allows to reproduce results
- includes many options allowing to
  - test different aspects of BBO such as
    - time-consuming evaluations
    - discrete/categorical variables
    - constraints handling
    - noise in the blackbox outputs
    - static surrogates
    - multiobjective optimization
  - propose sets of instances to draw performance/data profiles

Introduction

The solar **simulator**

The solar **instances**

The solar **features**

Examples of results

References

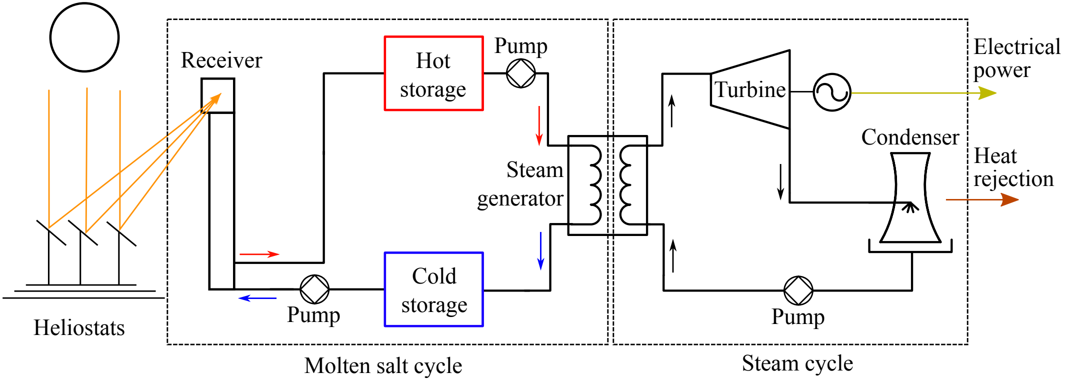
CSP tower plant with molten salt thermal energy storage

- 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

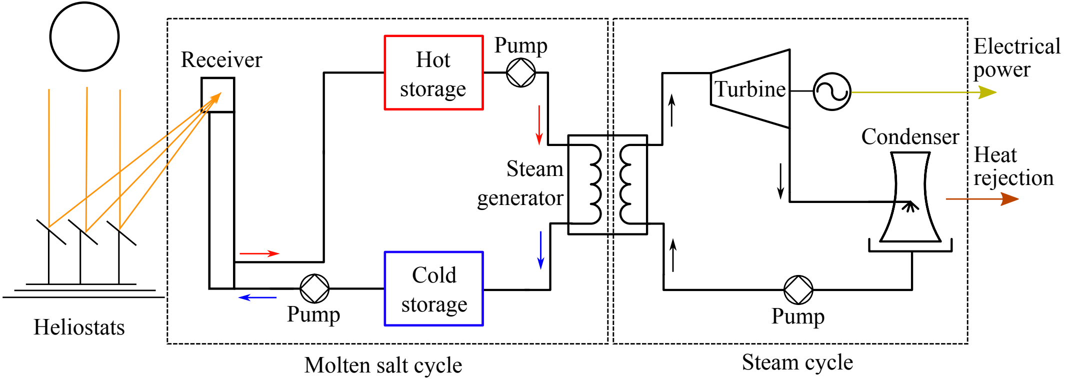
System dynamics

- Thermal power is extracted by raising the temperature of molten salt pumped through the receiver



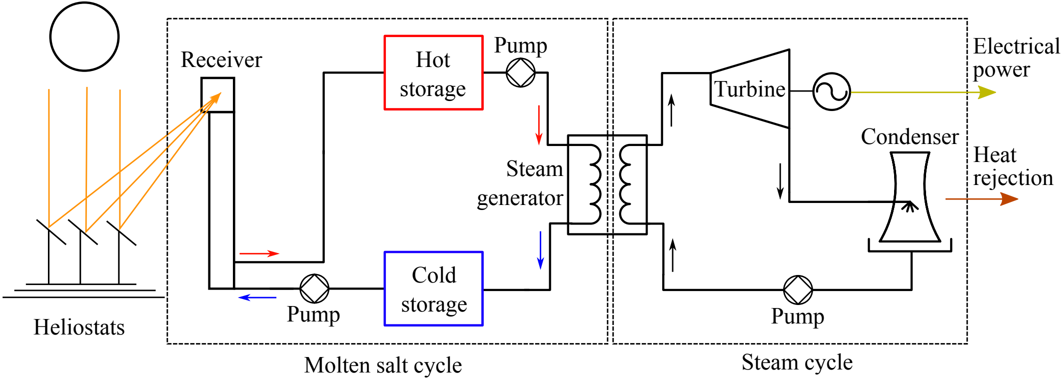
System dynamics

- The hot molten salt is directed to a hot storage tank



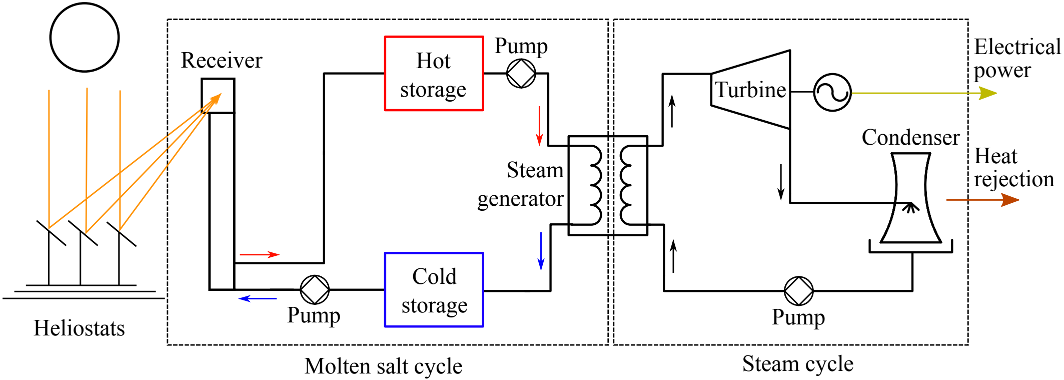
System dynamics

- Hot molten salt is pumped though the steam generator



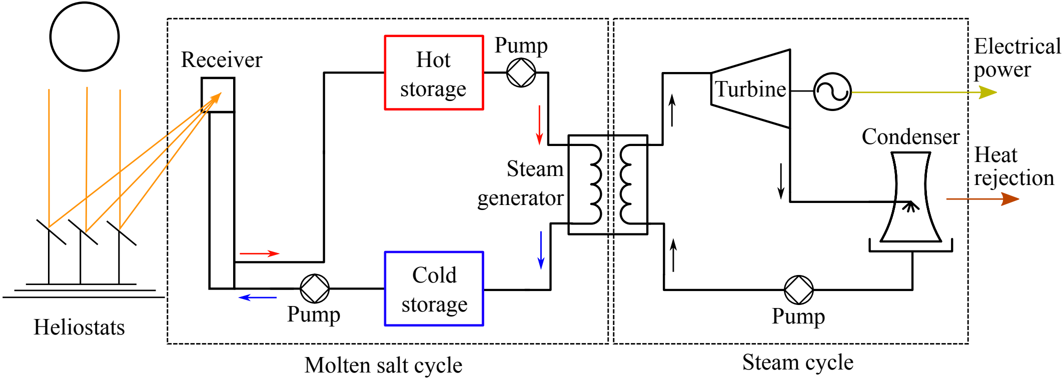
System dynamics

- Heat is transferred to a current of water on the other side of the steam generator which is transformed to superheated steam



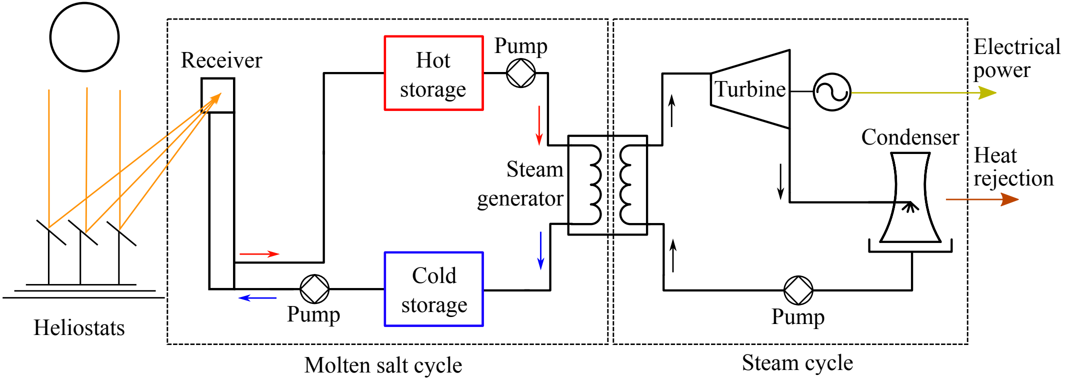
System dynamics

- Cold molten salt is recovered in the cold storage tank



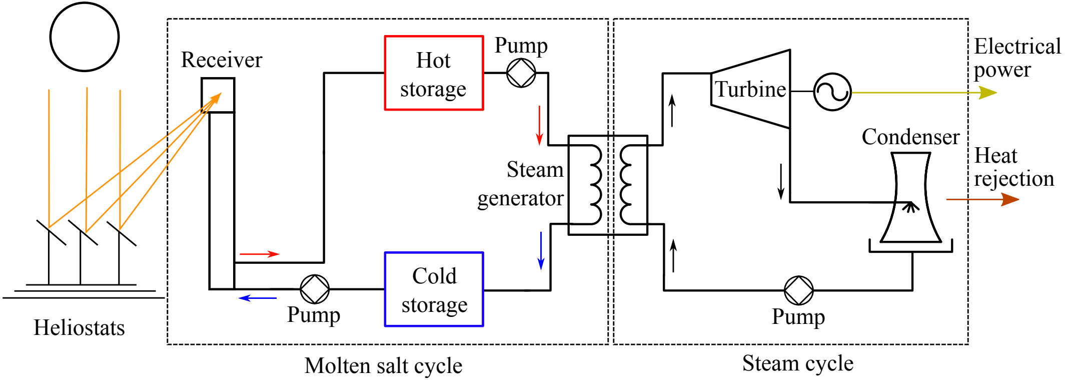
System dynamics

- Superheated high-pressure steam drives a turbine coupled to an electrical generator
- Low-pressure steam is condensed and pumped back as liquid water



System dynamics

- Losses due to non-idealities are accounted for in all components except the steam generator

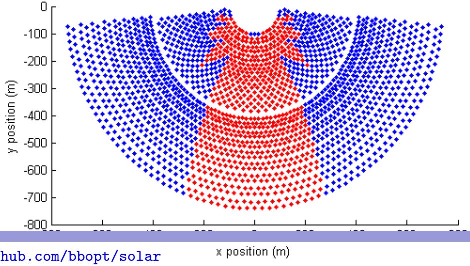


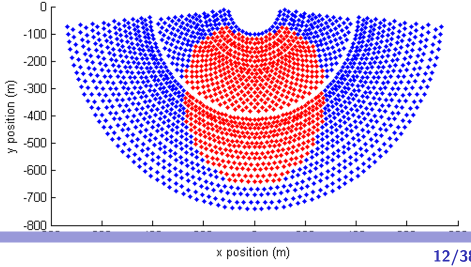
Heliostats field (1/2)

- The heliostats are laid on a radially staggered grid that prevents blocking losses between them
- The grid is calculated as a function of individual heliostat dimensions and tower height
- Once the grid layout is determined, each position is rated according to the average optical efficiency
- Shadowing effects are considered when calculating the overall performance
- The actual heliostats field is generated by occupying the first grid positions with the highest average optical efficiency for the given receiver aperture and tower height

Heliostats field (2/2)

- The images below show how the arrangement of 700 heliostats on the same spatial grid of 1,960 points varies with the receiver aperture width (3 meters vs 15 meters)
- As the aperture narrows, the algorithm selects heliostats closer to the North-South axis to minimize spillage
- For wider apertures, the selection is dictated by cosine efficiency and atmospheric attenuation





Main components of the simulator

- Sun radiation model
- Thermal storage model
- Parasitic loads model
- Pumping models
- Shell-and-tubes models with stress models of the tubes in both the receiver and steam generator
- Energy losses model (reflective, emissive, convective, conductive)
- Powerblock model with only one parameter (=optimization categorical variable): the choice of the type of turbine
- Demand model
- Investment cost model

All models have been validated during MLG's masters thesis, using simulations, scenarios, and comparisons with literature results

Main numerical methods in the simulator

- Monte Carlo simulation to evaluate the field efficiency
- Newton's method for systems to find roots in thermal equations
- Kernel smoothing to interpolate various discrete data
- Iterative methods to solve Heat Transfer Fluid equations

Introduction

The solar simulator

The solar instances

The solar features

Examples of results

References

The solar code is

- a command-line application
- the "natural heir" of our STYRENE simulator [Audet et al., 2008]
- publicly available at `https://github.com/bbopt/solar` under the GNU Lesser General Public License
- a relatively simple code in standard C++ ($\simeq$15k lines of codes)
- stand-alone: no external library to install
- multi-platform: C++ compilator is the only requirement

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

Objectives

- solar1 Max. total solar energy concentrated on the receiver aperture through one day (stochastic)
- solar2 Min. total heliostats field surface to run a pre-determined powerplant (analytic): $x_3^2(x_9^2 - x_8^2)x_7\frac{\pi}{180}$
- solar3 Min. total investment cost
- solar4 Min. cost of powerplant to respect a given demand with a limited size of field
- solar5 Max. compliance to a demand profile
- solar6 Min. cost of storage
- solar7 Max. receiver efficiency (energy transferred to the molten salt) (stochastic)
- solar8 Max. heliostat field performance (absorbed energy) and min. cost of field, tower and receiver
- solar9 Max. power and min. losses (stochastic)
- solar10 Unconstrained version of solar6

Types of variables

$$\min_{\mathbf{x}\in\mathcal{X}} \quad F(\mathbf{x}) \text{ s.t. } \mathbf{x} \in \Omega = \{\mathbf{x} \in \mathcal{X} : c_j(\mathbf{x}) \leq 0, j = 1, 2, \ldots, m\}$$

- The n variables are described by the set $\mathcal{X}$. They can be continuous or discrete
- $\mathcal{X}$ includes bounds on most of the variables
- The solar6 and solar10 instances have no discrete variables. In these cases $\mathcal{X} \subset \mathbb{R}^5$
- One of the discrete variable (the type of turbine) is categorical. solar considers it as an integer in $\{1, 2, \ldots, 8\}$

The following slides list all 29 possible variables. Each instance considers a subset of these variables. solar4 and solar9 consider all $n = 29$ variables

All variables: Heliostats field

#	Symbol	Quantity	Unit	Type	Lower bound	Upper bound
1	L_{hs}	Heliostats length	m	cont.	1	40
2	W_{hs}	Heliostats width	m	cont.	1	40
3	H_{twr}	Tower height	m	cont	20	250
4	H_r	Receiver aperture height	m	cont.	1	30
5	W_r	Receiver aperture width	m	cont.	1	30
6	N_{hs}	Number of heliostats to fit		discr.	1	$+\infty$
7	θ_{hs}	Field angular width	deg	cont.	1	89
8	R_{hs}^{min}	Min. distance from tower	$\times H_{twr}$	cont.	0	20
9	R_{hs}^{max}	Max. distance from tower	$\times H_{twr}$	cont.	1	20

All variables: Heat transfer loop

#	Symbol	Quantity	Unit	Type	Lower bound	Upper bound
10	T_r^{out}	Receiver outlet temp.	K	cont.	793	995
11	H_{hot}	Hot storage height	m	cont.	1 or 2	30 or 50
12	d_{hot}	Hot storage diameter	m	cont.	1 or 2	30
13	t_{hot}	Hot storage insulation thickness	m	cont.	0.01	2 or 5
14	t_{cold}	Cold storage insulation thickness	m	cont.	0.01	2 or 5
15	T_{cold}^{min}	Min. cold storage temp.	K	cont.	495	650
16	$N_{r,tb}$	Receiver number of tubes		discr.	1	u[3]
17	t_r	Receiver insulation thickness	m	cont.	0.01 or 0.1	2 or 5
18	d_r	Receiver tubes inner diameter	m	cont.	0.005	0.1
19	D_r	Receiver tubes outer diameter	m	cont.	0.005 or 0.0055 or 0.006	0.1

[3] $u \in \{1{,}884,\ 7{,}853,\ 8{,}567,\ 9{,}424\}$ (obtained with a priori constraints)

All variables: Steam generator and powerblock

#	Symbol	Quantity	Unit	Type	Lower bound	Upper bound
20	S_t	Tubes spacing	m	cont.	0.006 or 0.007	0.2
21	L_{sg}	Tubes length	m	cont.	0.5	10
22	d_{sg}	Tubes inner diameter	m	cont.	0.005	0.1
23	D_{sg}	Tubes outer diameter	m	cont.	0.006	0.1
24	$H_{sg,baf}$	Baffles cut		cont.	0.15	0.4
25	$N_{sg,baf}$	Number of baffles		discr.	2	$+\infty$
26	$N_{sg,tb}$	Number of tubes		discr.	1	$+\infty$
27	$N_{sg,sh,p}$	Number of shell passes		discr.	1	10
28	$N_{sg,tb,p}$	Number of tube passes		discr.	1	9
29	ST	Type of turbine		cat.	1	8

Types of constraints

$$\min_{\mathbf{x}\in\mathcal{X}} \quad F(\mathbf{x}) \text{ s.t. } \mathbf{x} \in \Omega = \{\mathbf{x} \in \mathcal{X} : c_j(\mathbf{x}) \leq 0, j = 1, 2, \ldots, m\}$$

Following the taxonomy of constraints [Le Digabel and Wild, 2015]:

- $\mathcal{X}$ describes bounds on the variables and the discrete nature of some of the variables. These constraints are unrelaxable
- The m constraints in Ω may be a priori or simulation constraints
- A priori constraints are also unrelaxable. In case of violation, the solar executable returns a flag to indicate a potential solver not to count the evaluation
- Most of the a priori constraints are linear
- Simulation constraints are relaxable
- Presence of hidden constraints
- All constraints (except the hidden ones) are quantifiable

The following slide lists all 18 possible constraints. Each instance considers a subset of these constraints, for a maximum of $m = 17$ constraints in solar9

All possible constraints

- 7 a priori and unrelaxable constraints:
  1. Tower is at least twice as high as heliostats (linear)
  2. Min. distance from tower $\leq$ Max. distance from tower (linear)
  3. Receiver inside diameter $\leq$ outside diameter (linear)
  4. Steam generator outer tubes diameter $\leq$ tubes spacing (linear)
  5. Steam generator inside diameter $\leq$ steam generator outside diameter (linear)
  6. Field surface area
  7. Number of tubes in receiver fit inside receiver
- 11 simulation and relaxable constraints:
  1. Cost of plant $\leq$ budget
  2. Check that the heliostats can fit in the field
  3. Molten salt melting point $\leq$ hot storage lowest temperature
  4. Molten salt melting point $\leq$ steam generator outlet temperature
  5. Receiver outlet temperature $\geq$ steam turbine inlet temperature
  6. Compliance to demand (stochastic)
  7. Pressure in receiver tubes $\leq$ yield pressure (stochastic)
  8. Molten salt melting point $\leq$ cold storage lowest temperature (stoch.)
  9. Check if storage is back to initial conditions (stochastic)
  10. Parasitics do not exceed a % of energy production (stochastic)
  11. Minimal acceptable energy production (stochastic)

Introduction

The solar simulator

The solar instances

The solar features

Examples of results

References

Getting started with solar

- Get the code at `https://github.com/bbopt/solar` and compile
- Command-line program that takes as arguments
  - a problem id (or instance number) in $\{1, 2, \ldots, 10\}$
  - the name of a file containing the coefficients of a point $\mathbf{x}$

  and displays the values of $F(\mathbf{x})$ and the $c_j(\mathbf{x})$'s
- Example: `> solar 7 x.txt` displays `f c1 c2 ... c6` (objective and six constraints)
- Simply executing `> solar` will guide the user and display the options, including a complete inline help with `> solar -help`

Check the solar installation

```
> solar -check
```

Mac:
Core i9: 659s
M1 Pro: 451s
M1 Max: 444s
M2 Max: 393s

Windows:
Core i7: 2,684s

Linux:
AMD EPYC: 1,284s

```
[[12:34:11] [~/Desktop] > ./solar -check

Validation tests (can take several minutes):

        RNG test  ( 1/ 2) ...... Ok     Time: CPU=8.8e-05       real=0
        RNG test  ( 2/ 2) ...... Ok     Time: CPU=9e-06         real=0
        Eval test ( 1/26) ...... Ok     Time: CPU=0.090865      real=0
        Eval test ( 2/26) ...... Ok     Time: CPU=0.164074      real=0
        Eval test ( 3/26) ...... Ok     Time: CPU=8.55466       real=9
        Eval test ( 4/26) ...... Ok     Time: CPU=14.3939       real=14
        Eval test ( 5/26) ...... Ok     Time: CPU=12.444        real=12
        Eval test ( 6/26) ...... Ok     Time: CPU=1.67694       real=2
        Eval test ( 7/26) ...... Ok     Time: CPU=1.714         real=2
        Eval test ( 8/26) ...... Ok     Time: CPU=0.000297      real=0
        Eval test ( 9/26) ...... Ok     Time: CPU=1.8335        real=2
        Eval test (10/26) ...... Ok     Time: CPU=16.9975       real=17
        Eval test (11/26) ...... Ok     Time: CPU=0.088462      real=0
        Eval test (12/26) ...... Ok     Time: CPU=1.76882       real=2
        Eval test (13/26) ...... Ok     Time: CPU=2.03457       real=2
        Eval test (14/26) ...... Ok     Time: CPU=57.289        real=57
        Eval test (15/26) ...... Ok     Time: CPU=76.4028       real=76
        Eval test (16/26) ...... Ok     Time: CPU=2.17247       real=2
        Eval test (17/26) ...... Ok     Time: CPU=50.1873       real=51
        Eval test (18/26) ...... Ok     Time: CPU=50.3843       real=50
        Eval test (19/26) ...... Ok     Time: CPU=50.3955       real=50
        Eval test (20/26) ...... Ok     Time: CPU=3.31858       real=4
        Eval test (21/26) ...... Ok     Time: CPU=3.21749       real=3
        Eval test (22/26) ...... Ok     Time: CPU=5.77947       real=6
        Eval test (23/26) ...... Ok     Time: CPU=0.003279      real=0
        Eval test (24/26) ...... Ok     Time: CPU=3.86108       real=4
        Eval test (25/26) ...... Ok     Time: CPU=2.24941       real=2
        Eval test (26/26) ...... Ok     Time: CPU=25.7252       real=26

This version of SOLAR is valid

CPU time : 392.748s
Real time: 393s
```

Execution times (for one replication)

	$\mathbf{x}_0$	$\mathbf{x}^\star$	worst observed	avg 10k (simu. completed)
solar1	0 sec	14 sec	81 sec	0.64 sec
solar2	15 sec	20 sec	2 min	
solar3	3 sec	3 sec		
solar4	3 sec	4 sec		
solar5	2 min	2 min		
solar6 & 10	4 sec	2 min	12 min	6 sec
solar7	5 sec	5 sec	8 sec	6 sec
solar8	9 sec		29 sec	
solar9	4 sec		1 min	

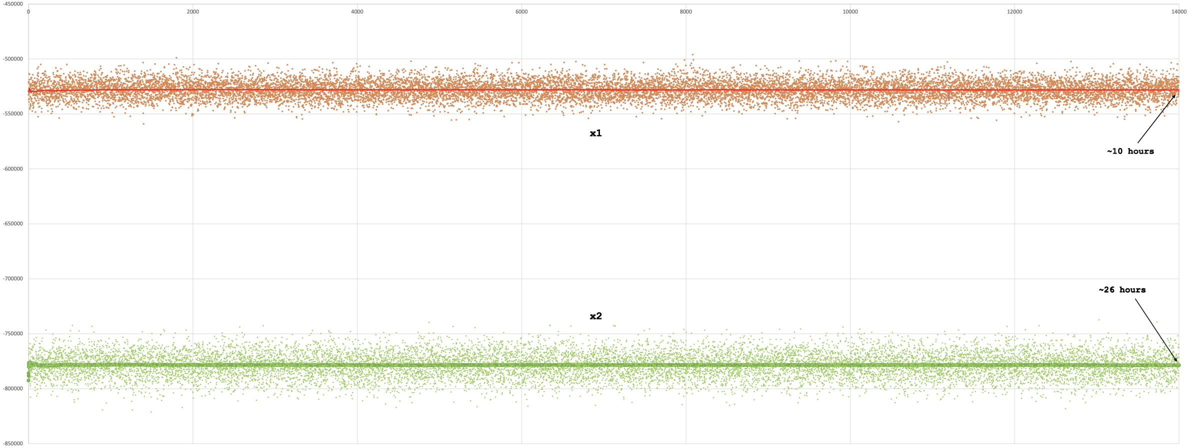
Impact on the execution time caused by violation of a priori constraints (instantaneous), violation of simulation constraints, number of heliostats, etc.

Stochasticity and replications

- Stochasticity is due to the Monte Carlo simulation for the heliostats field
- Random seed is set to the same value by default: This corresponds to a deterministic blackbox
- Use the option `-seed` to change the random seed
- The option `-seed=diff` makes the blackbox stochastic
- The option `-rep` executes several simulations and outputs average values
- A high number of replications will tend to decrease stochasticity but will lead to expensive evaluations (which is great in BBO benchmarking)

Illustration of replications for the objective of solar1

`> solar 1 -rep=14000 x1.txt` and `> solar 1 -rep=14000 x2.txt`



Multi-fidelity

- The option `-fid` with a value in $]0;1]$ changes the fidelity of the simulator
- It has been tuned by changing the stopping criteria and precisions in the different numerical methods in the simulator
- Each different value of this option generates a static surrogate
- `-fid=1` corresponds to the “true” blackbox (called the truth)
- This option allows to consider multi-fidelity metamodels or variable precision static surrogates
- Note that using the `-rep` option also allows to consider such surrogates when the truth is considered to be obtained with high number of replications

Illustration of the multi-fidelity in solar2 with its (infeasible) $\mathbf{x}_0$

fid.	time reduction	c_2	c_3	c_6	c_7	c_8	c_9	c_{10}	c_{13}
truth	0 (15 sec)	0	0	0	0	0	0	0	0
0.95	7 (14 sec.)	6	0	0	0.3	0	0	0	0
0.90	13	7	0	0	1	0	0	0	0
0.85	20	4	0	0	0.4	0	0	0	0
0.80	33	0.3	0	0	0.3	0	0	0	0
0.75	33	1	0	0	1	0	0	0	0
0.70	40	6	0	0	2	0	0.1	0	0
0.65	40	12	0	0	3	0	0.2	0	0
0.60	47	26	0	0	4	0	0.3	0	0
0.55	47	23	0	0	5	0	0.3	0	0
0.50	60	18	0	0	3	0	0.3	0	0
0.45	67	13	0	0	0.2	0	0.3	0	0
0.40	73	15	0	0	1	0	0.3	0	0
0.35	73	35	0	0	7	0	0.5	0	0
0.30	73	53	0	0	4	0	0.6	0	0
0.25	80	79	0	0	6	0	0.7	0	0
0.20	80	89	0	0	8	0	0.8	0	0
0.15	87	100	0	0	14	0	0.8	0	0
0.10	93	100	0	0	52	0	0.9	0	0
0.05	100 (0 sec.)	100	0	0	214	0.07	1	0	0

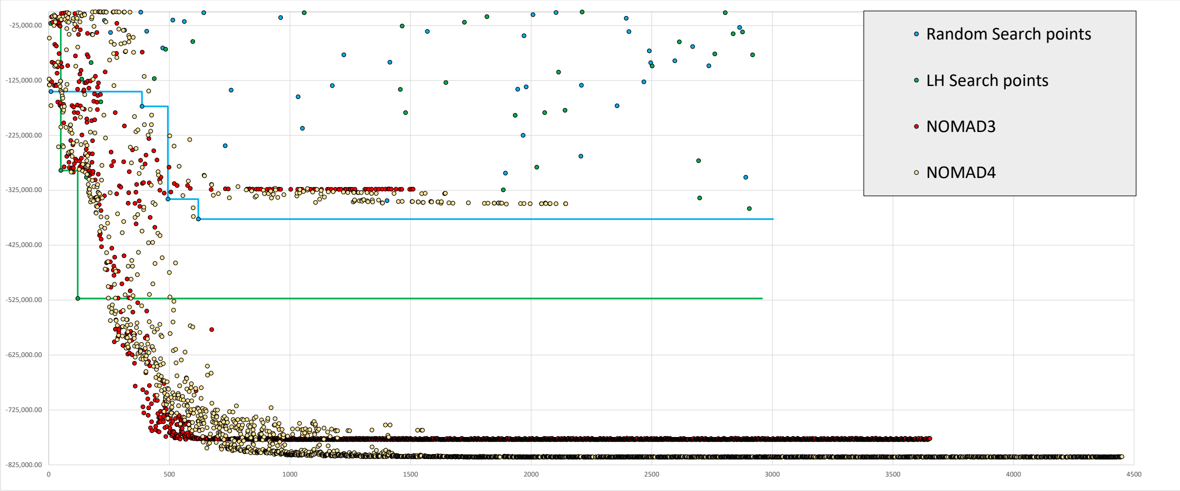
- Values correspond to relative errors with the truth (in %)
- Obj. and *a priori* constraints are not shown
- Some constraints can be evaluated at no cost
- Others (c_2 and c_7) need the default fidelity of 1

Feasibility with sampling and NOMAD

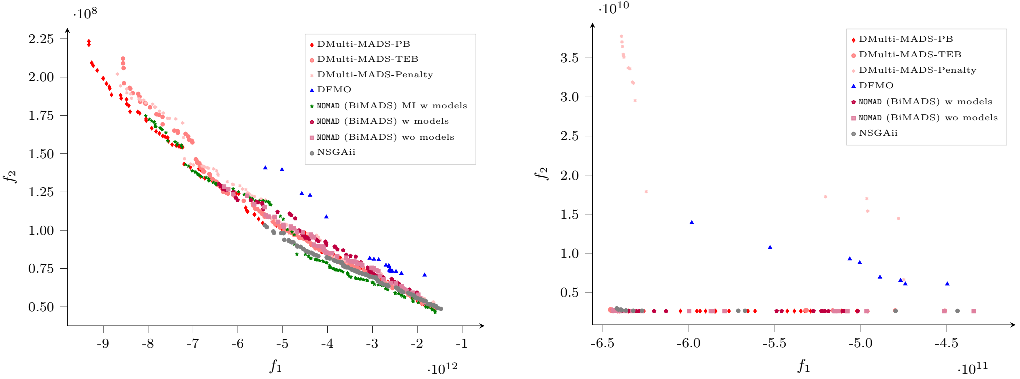
Instance	LH search (10k points)		NOMAD3		
	satisf. ap constr.	feas. pts	satisf. ap constr.	feas. pts	number of eval.
solar1	30%	0.35%	96%	74%	3,792
solar2	0%	0%	97%	0%	1,635
solar3	0.49%	0%	99%	9%	30,525
solar4	0%	0%	83%	0%	44,303
solar5	0%	0%	83%	59%	3,405
solar6	90%	5%	99%	0%	3,539
solar7	2%	1%	74%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



Biobjective optimization (by L. Salomon)



Pareto front approximations for solar8 (left) and solar9 (right) with different solvers with a budget of 5K evaluations. Taken from [Bigeon et al., 2022]

Introduction

The solar simulator

The solar instances

The solar features

Examples of results

References

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.

Bigeon, J., Le Digabel, S., and Salomon, L. (2022).
Handling of constraints in multiobjective blackbox optimization.
Technical Report G-2022-10, Les cahiers du GERAD.

Le Digabel, S. and Wild, S. (2015).
A Taxonomy of Constraints in Simulation-Based Optimization.
Technical Report G-2015-57, Les cahiers du GERAD.

Lemyre Garneau, M. (2015).
Modelling of a solar thermal power plant for benchmarking blackbox optimization solvers.
Master's thesis, Polytechnique Montréal.