Wind farm optimization with NOMAD

Merlin Keller Sébastien Le Digabel
J. Gobert A. Lesage-Landry J. Pelamatti

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

POLYTECHNIQUE MONTRÉAL
TECHNOLOGICAL UNIVERSITY

SAMOURAI Workshop, 2024-12-10

Presentation outline

Industrial Context

In the context of wind farm development, one of the central quantities of interest is the:

Levelized Cost of Energy

Definition: Average revenue per unit of electricity generated required to recover the costs of building and operating a generating plant during an assumed financial life and duty cycle.

$$\mathrm{LCOE} := \frac{\text{Sum of costs over lifetime}}{\text{Sum of electrical energy produced over lifetime}}$$

Source: `https://en.wikipedia.org/wiki/Levelized_cost_of_electricity`

Goals : Focusing on energy yield assessment, through the *expected annual production (AEP)*, We wish to address

- Uncertainty Quantification
- Design optimization (*cf* Babacar's PhD)

Wind farm energy yield assessment

Based on a (simplistic) computer model of the wind-farm behaviour:

$$\mathbf{P} = \mathcal{G}(\mathbf{X}, d) \tag{1}$$

with:

- $\mathbf{P} = (P_t)_{1 \le t \le T}$ power production time-series
- $\mathcal{G}$ Wind farm simulator, based on an air-flow model (PyWake) and a turbine model (power-curve)
- $\mathbf{X} = (X_t)_{1 \le t \le T}$ Uncertain inputs: onsite wind speed time-series
- d Design parameters (Turbine layout)

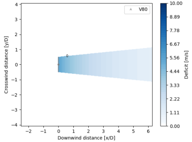
Air-flow modeling using PyWake

https://topfarm.pages.windenergy.dtu.dk/PyWake/

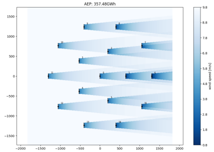
- PyWake is an open-sourced wind farm simulation tool developed by DTU for computing flow fields and power production of a wind farm
- Main features:
  - Simplified model, superposing user-chosen *deficit functions* accounting for wake and blockage effects,
  - Static model, each time-step is treated separately:

$$\mathcal{G}(\mathbf{X}, d) = (\mathcal{G}(X_t, d))_{1 \leq t \leq T}$$

  - Quicker but less accurate than more refined CFD solvers (Meteodyn, code_saturne)



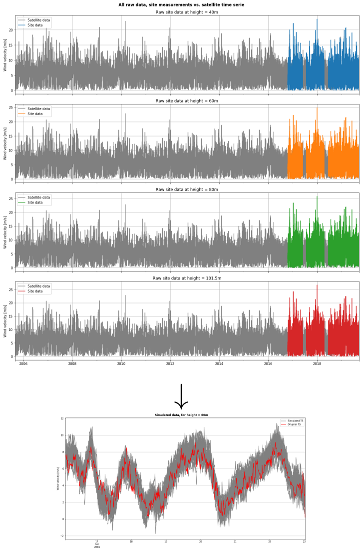
$\downarrow$



Wind speed modeling

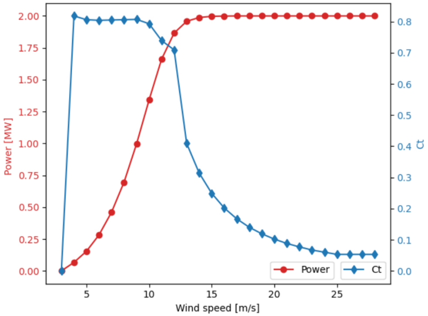
- Two different data sources for wind speed:
  - onsite mast measures (colored TS), accurate but on limited to a few years)
  - *re-analys*, or *satellite* data (grey TS), less accurate but available on decades

- Using both, it is possible to extrapolate onsite measures to long-time period and hub-height [Keller et al., 2023]

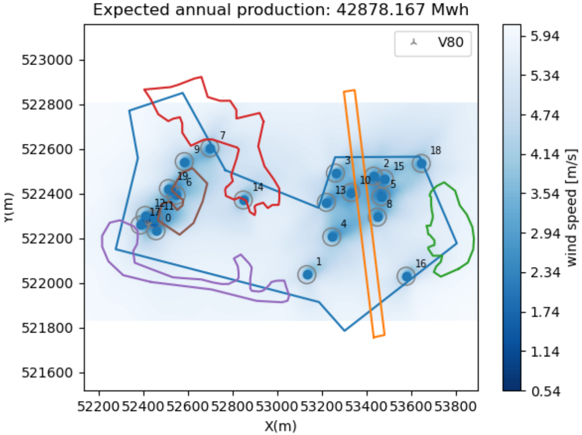


Power curve : The Vestas V80-2 wind-turbine

- Power as a function of wind-speed
- Coefficient of thrust : ratio of axial force to incoming flow momentum, characterizing wake



Example flow map for test site, with random layout



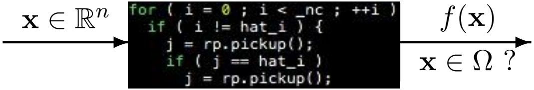
What is the layout maximizing the EAP?

Blackbox / Derivative-Free Optimization

We consider

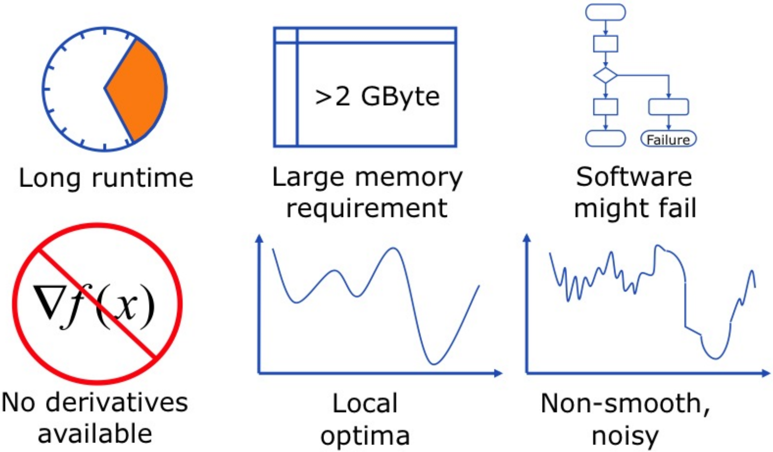
$$\min_{\mathbf{x}\in\Omega=\{c_j(\mathbf{x})\leq 0 \;\; j=1,2,\ldots,m\}} \qquad f(\mathbf{x})$$

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



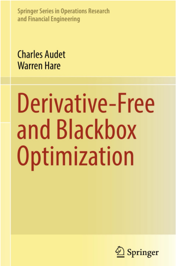
- Each call to the simulation may be expensive
- The simulation can fail
- Sometimes $f(\mathbf{x}) \neq f(\mathbf{x})$
- Derivatives are not available and cannot be approximated

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)*

Problem definition

Blackbox / Derivative-Free Optimization

The MADS algorithm and the NOMAD software package

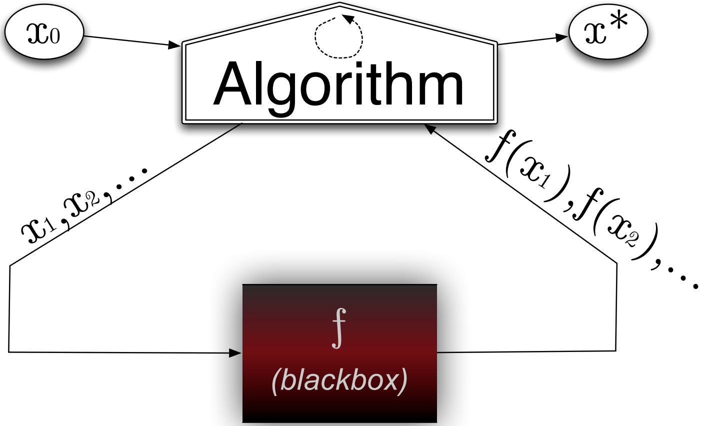
The blackbox model

Preliminary computational results

The amon package

Conclusion

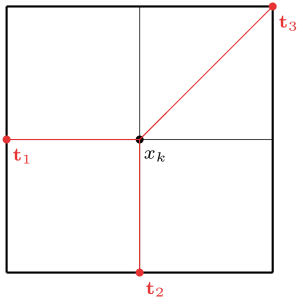
Typical setting of a BBO method



Unconstrained case, with one initial starting solution

MADS illustration with $n = 2$: Poll step

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

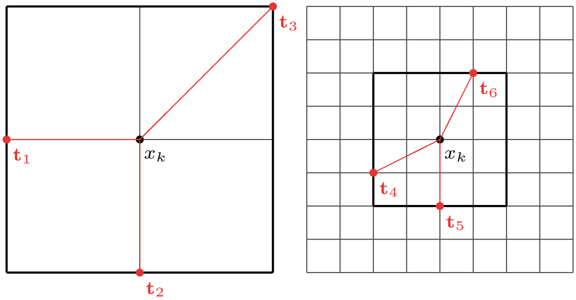


poll trial points$=\{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3\}$

MADS illustration with $n = 2$: Poll step

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

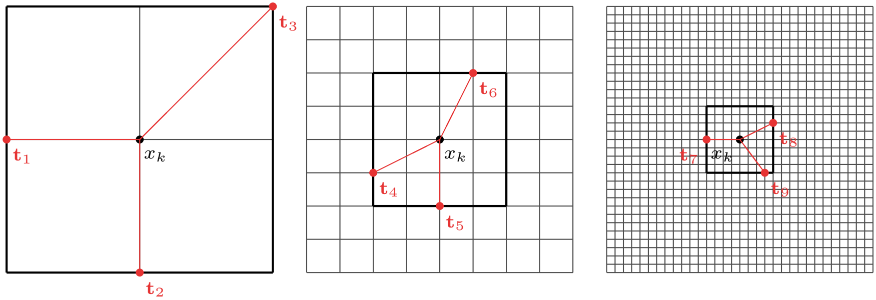
$$\Delta^{k+1} = 1/2$$



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

MADS illustration with $n = 2$: Poll step

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



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

[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] (unconstrained version)

Two 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 $f(\mathbf{x})$

Two strategies to deal with constraints

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

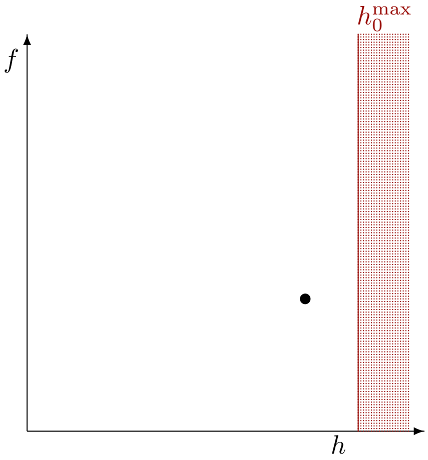
  Defined for relaxable and quantifiable constraints.
  Other constraints define the set $\mathcal{X}$.
  As in the filter method of [Fletcher and Leyffer, 2002], it uses the non-negative constraint violation function $h : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$

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

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

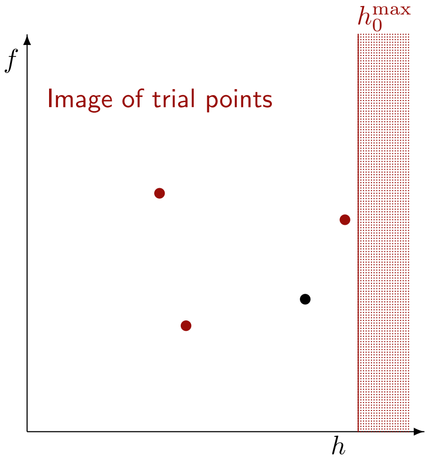
Two strategies to deal with constraints

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



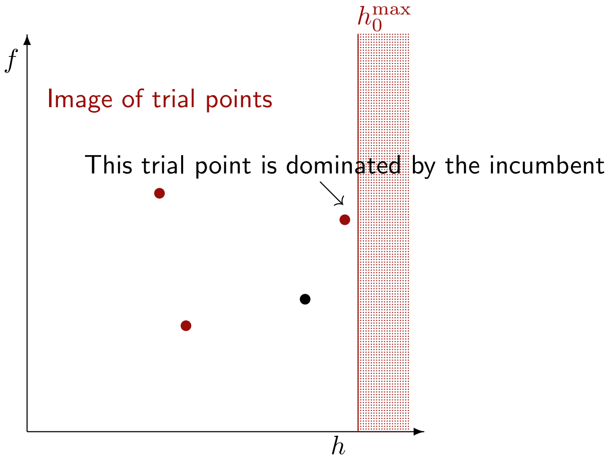
Two strategies to deal with constraints

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



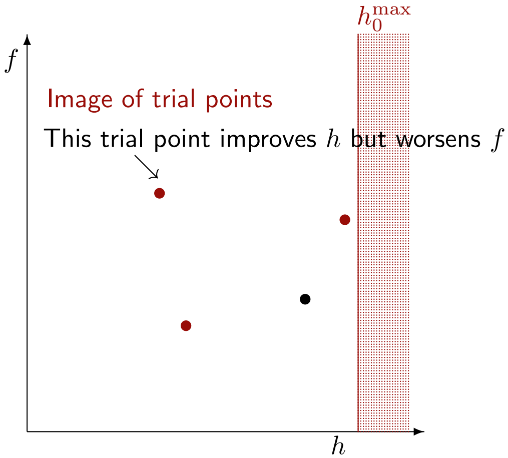
Two strategies to deal with constraints

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



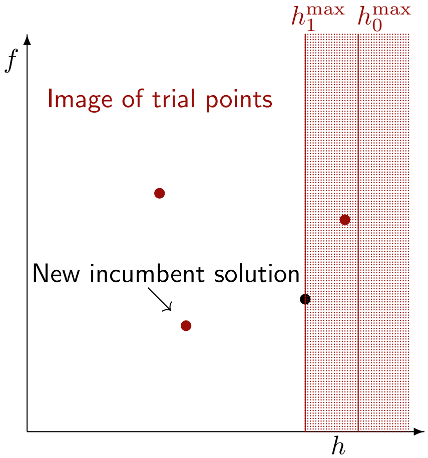
Two strategies to deal with constraints

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



Two strategies to deal with constraints

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



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]

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
```

Blackbox optimization model

- Model described in [Thomas et al., 2023]
- $\mathbf{x} = (x_1, y_1, x_2, y_2, \ldots, x_{N_T}, y_{N_T})$ = locations of the N_T turbines
- N_D and N_S: Number of discretized values for wind direction and speed
- Maximization of the Annual Energy Production (AEP):

$$f(\mathbf{x}) = -AEP(\mathbf{x}) = -\left[\frac{\text{hours}}{\text{day}}\right]\left[\frac{\text{days}}{\text{year}}\right]\sum_{i=1}^{N_D}\sum_{j=1}^{N_S} f_{ij}\sum_{k=1}^{N_T} P_{ij}(x_k, y_k)$$

- $P_{ij}(x_k, y_k)$: power for the turbine k with a wind direction i and speed j (computed with `PyWake` [Pedersen et al., 2023])
- f_{ij} the probability of a given wind speed and wind direction combination

Constraint 1: Placement of the turbines in the boundary zone and in the constructible zone

- Boundary zone: This corresponds to the bounds of $\mathbf{x}$ (set $\mathcal{X}$)
- Constructible zone:
  - **Binary version:** (x_i, y_i) must be admissible for all $i = 1, 2, \ldots, N_T$: this is easily checked by a routine before launching `PyWake` (A priori binary constraint). Infeasible designs are not sent to `PyWake` and a blackbox evaluation is not counted.
  - **Continuous version:**

$$\sum_{k=1}^{N_T} d_k \leq 0$$

    where d_k is the distance between turbine k and the closest constructible zone. This is an A priori, relaxable, and quantifiable constraint. The infeasible points are sent to `PyWake` and a blackbox evaluation is counted.

  - **Variant based on a projection:** Turbines are projected to the neared constructible zone

Constraint 2: Minimal space between two wind turbines

$$S_{ij} \geq 2d \text{ for all } i, j = 1, 2, \ldots, N_T \text{ and } i \neq j$$

- S_{ij}: Spacing between turbines i and j
- d: Diameter of a wind turbine
- This is an A priori and quantifiable constraint
- Two variants: relaxable or unrelaxable
- Current version: $N_T(N_T - 1)/2$ constraints
- Alternate version: One aggregated constraint

Constraints with EB or PB

Placement	Spacing	AEP (GWh)	Time (s)
PB	PB	55.556	17,236
EB	PB	55.321	17,714
PB	EB	54.994	17,375
EB	EB	54.980	16,913

NOMAD **vs Monte Carlo**

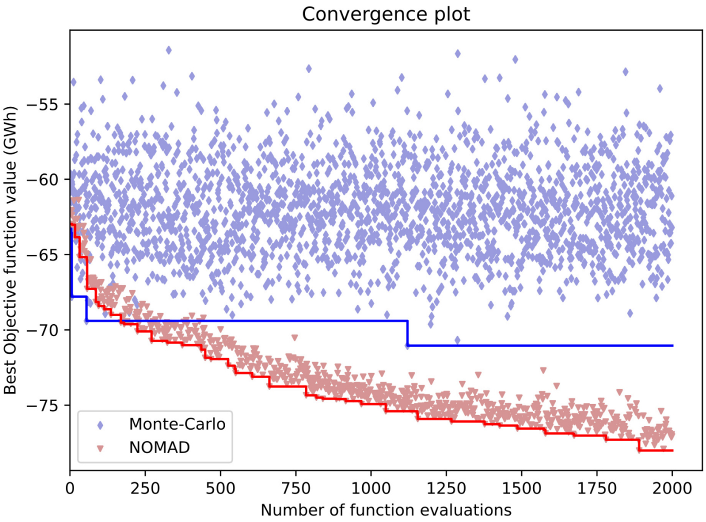
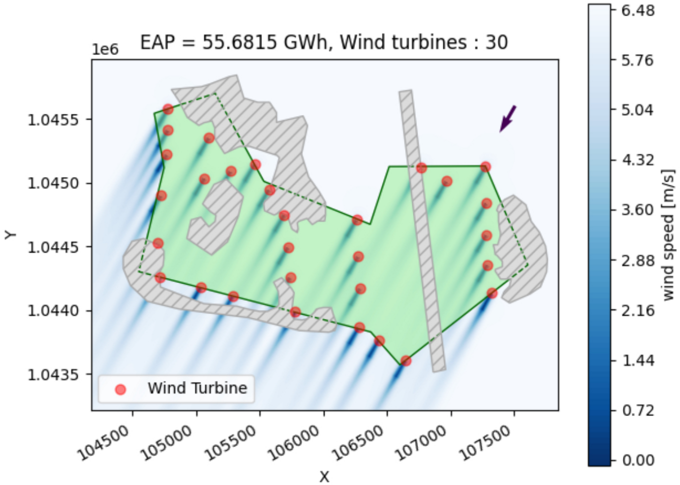
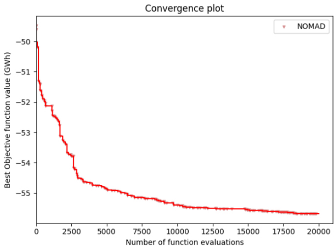


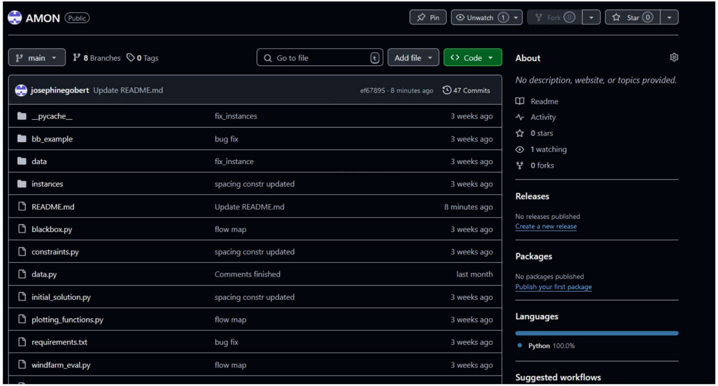
Illustration of a solution



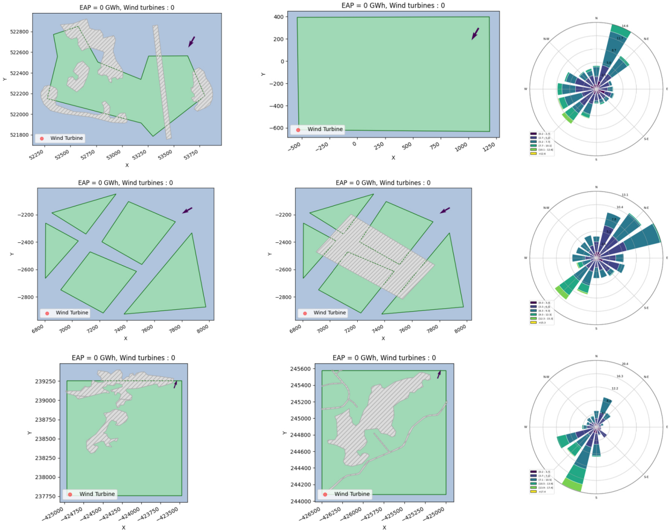


The amon package

- Our interface to PyWake is available at https://github.com/josephinegobert/AMON
- Several instances have been defined as realistic benchmarking instances for the BBO community



The amon package: Instances



Summary

- Blackbox optimization motivated by industrial applications, like the optimization of the layout of wind turbines in a wind farm
- NOMAD, the software package implementing MADS, is used for this BBO problem
- The amon package is available for the BBO community for benchmarking with a realistic blackbox
- Future work includes
  - more testing of the different ways of modeling the problem
  - comparing NOMAD with other methods like [Sow et al., 2024]
  - specializing NOMAD for locating 2D objects on a map
  - improving amon and its documentation
  - integrate more accurate / costly CFD wind flow models, in a multi-fidelity approach

References I

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 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.

Fletcher, R. and Leyffer, S. (2002).
Nonlinear programming without a penalty function.
Mathematical Programming, Series A, 91:239–269.

Keller, M., Zannane, S., Paul, N., Dall'Ozzo, C., and Ding, L. (2023).
Bootstrap Uncertainty Quantification for long-term Wind Farm Production.
In *23rd Conference of the Eurpean Network for Business and Industrial Statitics (ENBIS)*.

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

References II

Pedersen, M., Forsting, A., van der Laan, P., Riva, R., Romàn, L. A., Criado Risco, J., Friis-Møller, M., Quick, J., Schøler Christiansen, J., Valotta Rodrigues, R., Tobias Olsen, B., and Réthoré, P.-E. (2023).
PyWake 2.5.0: An open-source wind farm simulation tool.
Software available at `https://gitlab.windenergy.dtu.dk/TOPFARM/PyWake`.

Sow, B., Le Riche, R., Pelamatti, J., Keller, M., and Zannane, S. (2024).
Wasserstein-Based Evolutionary Operators for Optimizing Sets of Points: Application to Wind-Farm Layout Design.
Applied Sciences, 14(17).

Thomas, J., Baker, N., Malisani, P., Quaeghebeur, E., Sanchez Perez-Moreno, S., Jasa, J., Bay, C., Tilli, F., Bieniek, D., Robinson, N., Stanley, A., Holt, W., and Ning, A. (2023).
A comparison of eight optimization methods applied to a wind farm layout optimization problem.
Wind Energy Science, 8(5):865–891.