Optimization: Advanced Topics

Alberto TONDA

UMR 518 MIA-PS, INRAE, AgroParisTech, Université Paris-Saclay
UAR 3611, Institut des Systèmes Complexes de Paris Île-de-France

Outline

- Finding multiple optima
- Quality-diversity optimization
- Deceptive objective functions
- Flat objective functions
- Expensive objective functions
- Optimizing under incertitude
- Dynamic objective functions

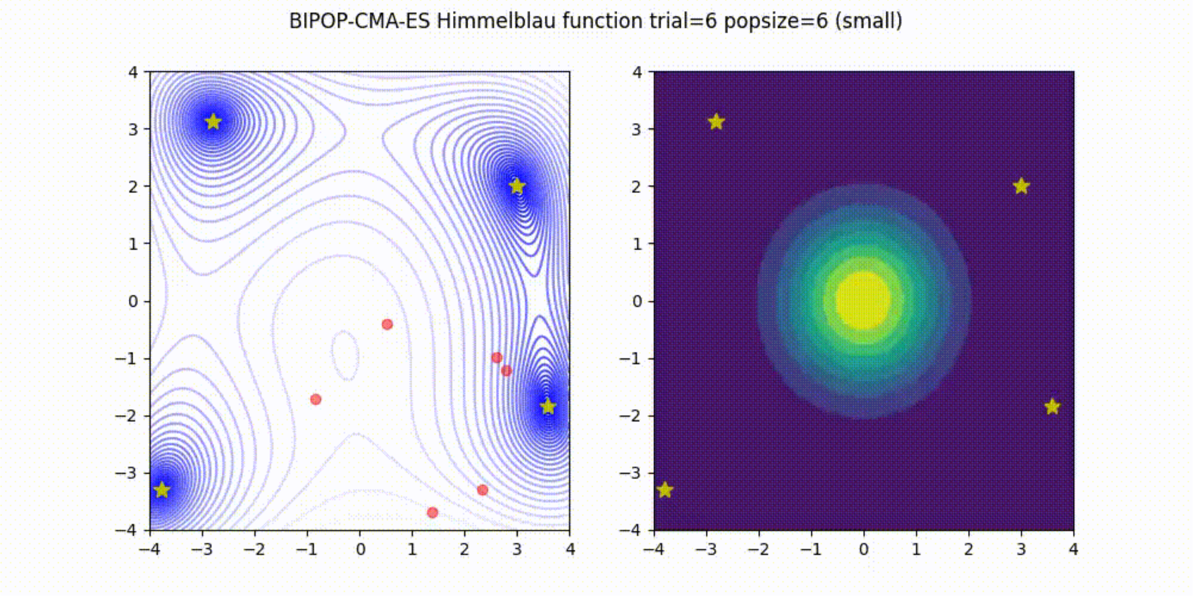
Finding multiple optima

- Multimodal functions
  - Several global optima
  - Or a global optimum and several strong local optima

- Optimization algorithms tend to end up in the same places
  - How to force the algorithm to explore other areas?
  - Any ideas?

Finding multiple optima

- BIPOP-CMA-ES
  - Set a budget with a total number of evaluations
  - After a run stops for heuristic conditions
  - Restart the run with larger populations, until budget exhausted



Finding multiple optima

- Flattening explored areas of the objective function
    - Run algorithm once, store best point found
    - Modify value of objective function around best point
    - Set the value around best point to undesired (e.g. 0.0 if maximize)
    - Run the algorithm a second time; iterate several times

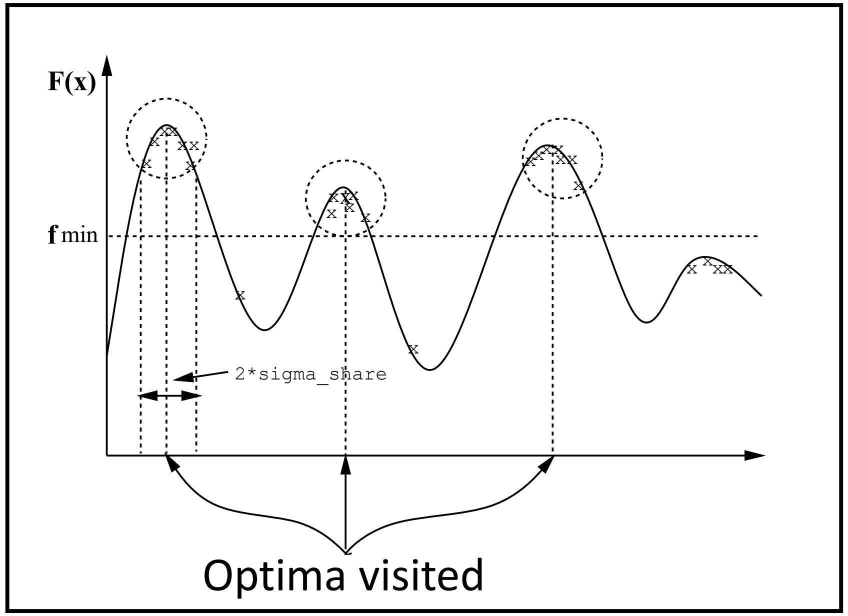
- "Removing" areas of the search space already explored
- Changing the objective function lead algorithms elsewhere

Finding multiple optima

- All the techniques require restarting!
- Niching: push for exploration during a single runtime
  - Technique developed for Evolutionary Algorithms
  - Lower value of candidate solution based on crowding
  - Isolated solutions are favored

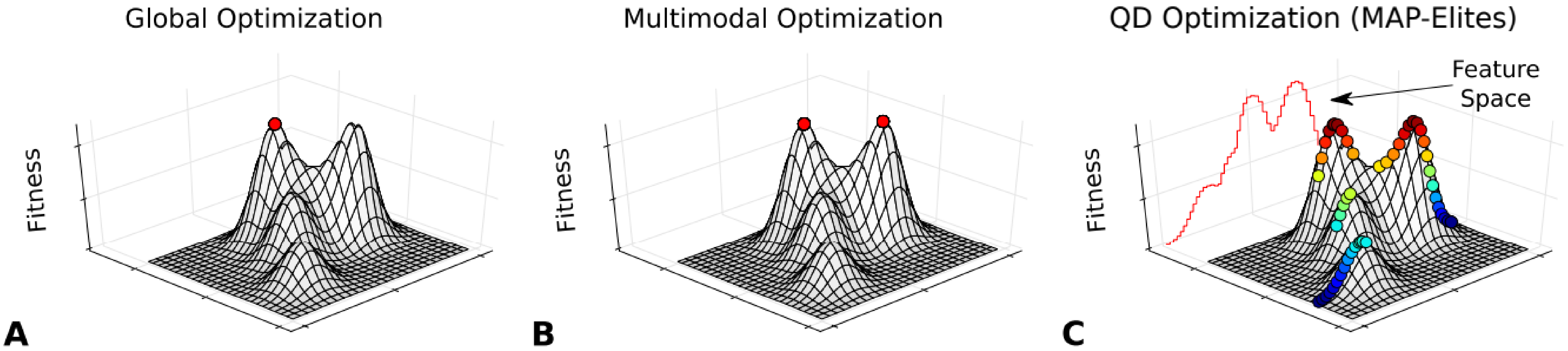
$$\text{Fitness'}(x) = \frac{\text{Fitness}(x)}{\sum_{x' \in Vois(x)} Sh(d(x, x'))}$$

$$Sh(d) = \begin{cases} 1 - \left(\frac{d}{\sigma_{share}}\right)^{\alpha} & \text{if } d < \sigma_{share} \\ 0 & \text{otherwise} \end{cases}$$



Quality-diversity optimization

- Recent trend (>2015)
  - Objective is finding set of **high-performing** and **diverse** solutions
  - Diversity is problem-specific, called *behavior* or *feature space*

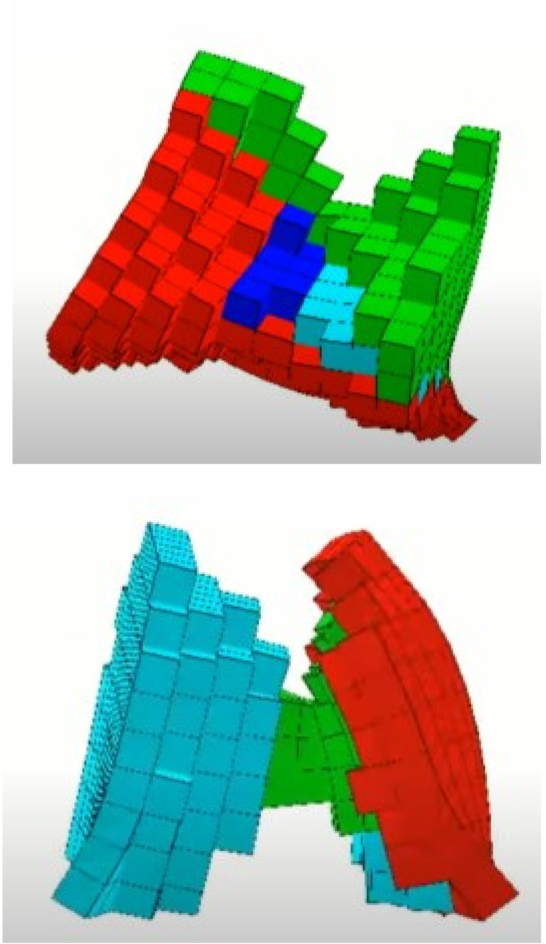


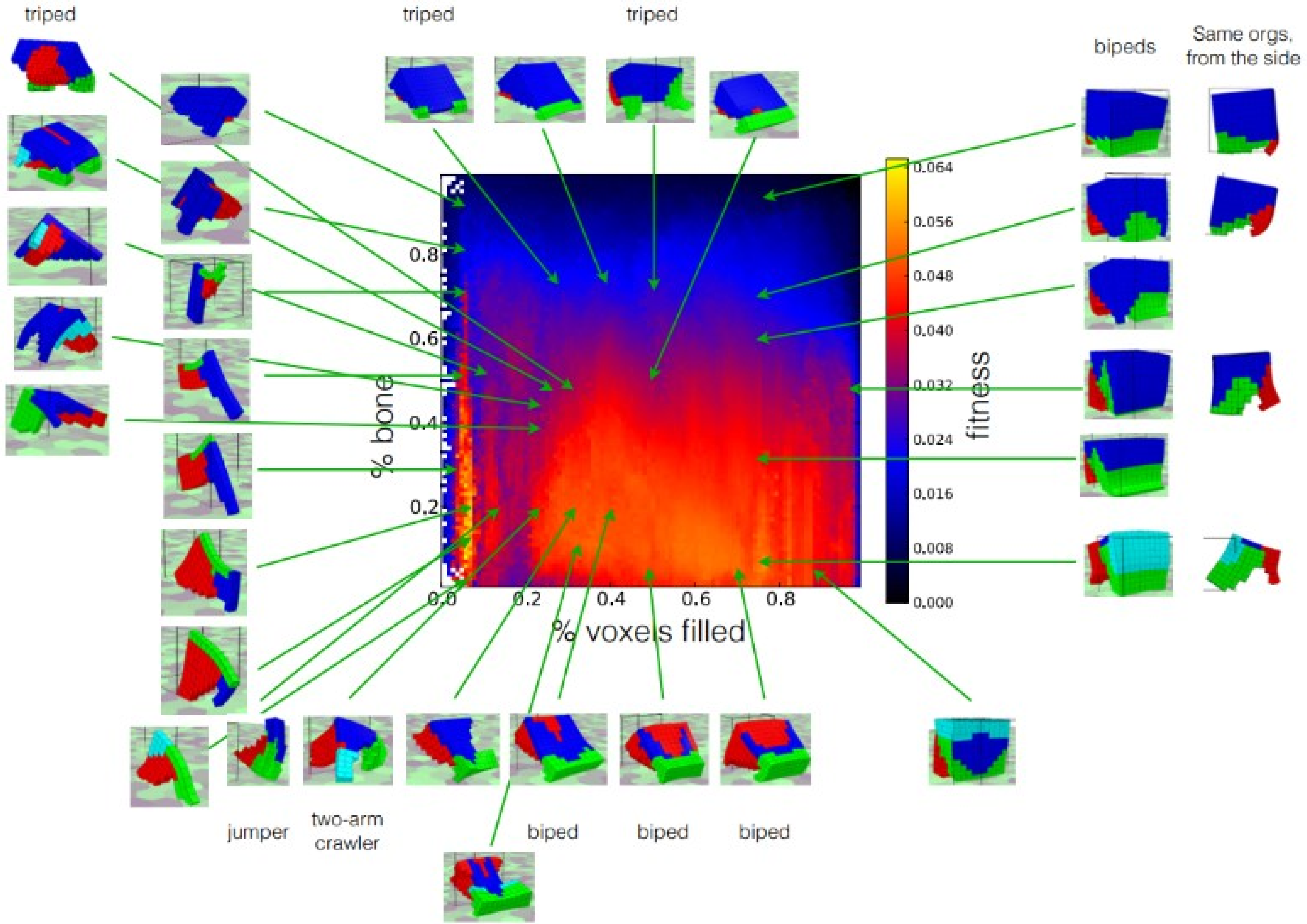
Chatzilygeroudis et al., *Quality-Diversity Optimization: a novel branch of stochastic optimization*, 2020

Quality-diversity optimization

- Multidimensional Archive of Phenotypic Elites (MAP-Elites)
  - Stochastic optimization (EA)
  - Solutions are not selected just on objective function value
  - Keep best-performing solutions in cells of *behavior/feature space*
  - Cells are part of a grid that is user-defined

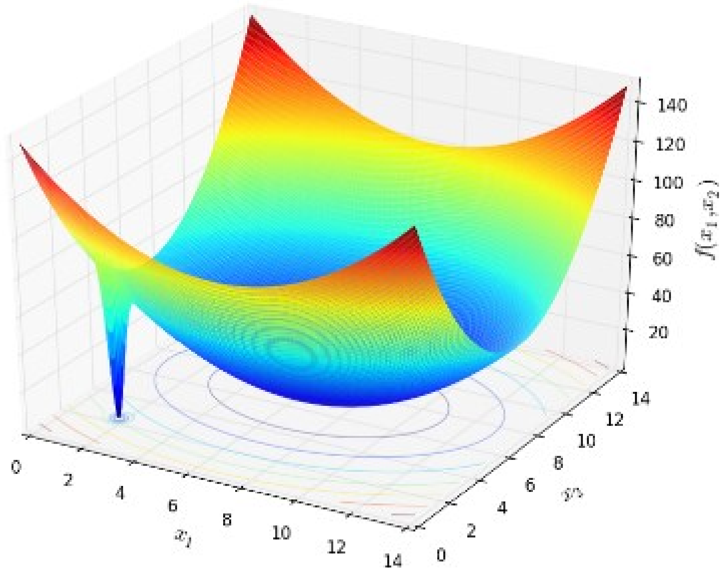
Example: evolving soft robots

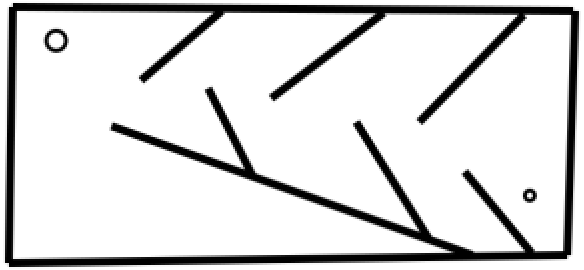




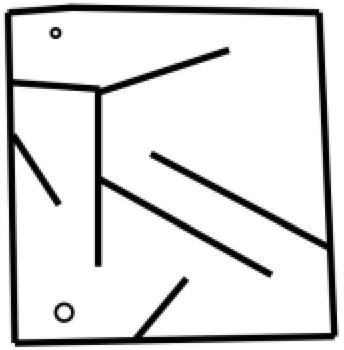
Deceptive objective functions

- Feedback from the objective function is *deceptive*
  - Following feedback leads away from global optimum
  - Strong local optima surrounding the global one





(a) Medium Map

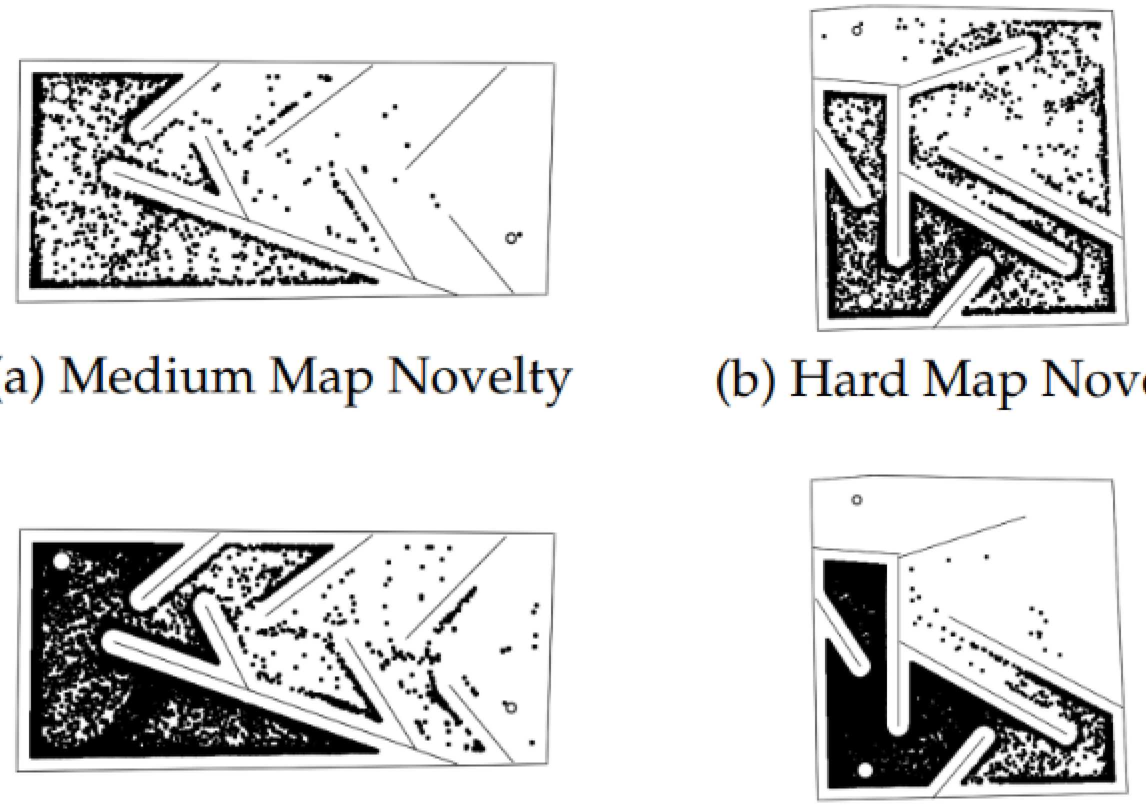


(b) Hard Map

Deceptive objective functions

- Solution (?): Novelty Search
  - Ignore feedback from the objective function
  - Evaluate candidate solutions based on diversity
  - Keep archive of solutions, search near solutions that are “novel”
  - Measure novelty: problem-dependent
  - Example: Average distance from k nearest neighbors
- Value of the objective function used to stop

Deceptive objective functions

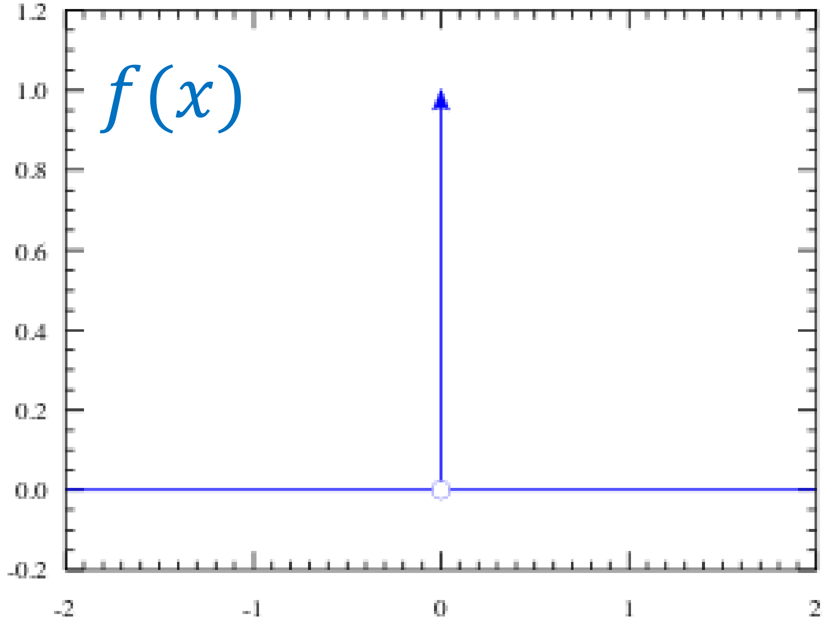


(a) Medium Map Novelty (b) Hard Map Novelty

(c) Medium Map Fitness (d) Hard Map Fitness

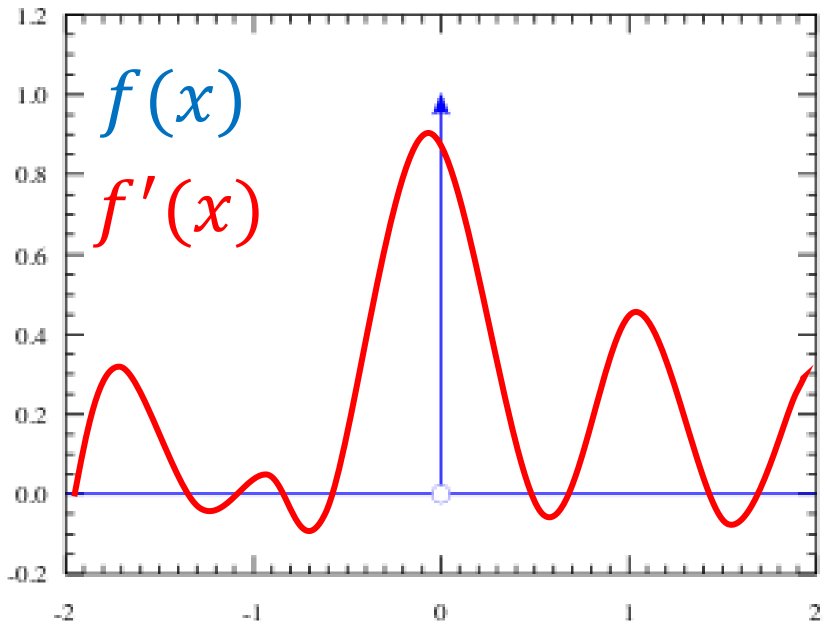
Flat objective functions

- Your objective function has the **same value** everywhere…
- …EXCEPT in **one (or few) specific points** you are interested in
- Example: bug in software/hardware
- Any ideas?



Flat objective functions

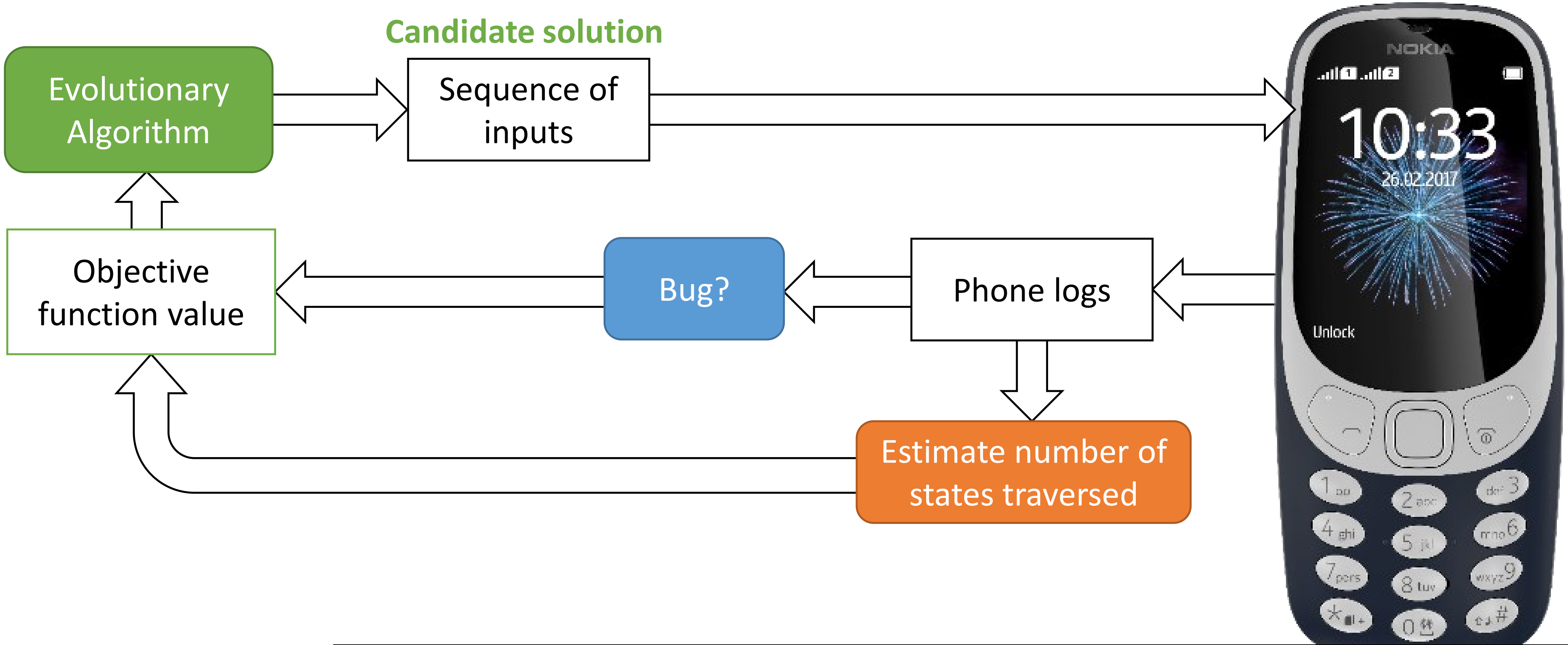
- There is NO SOLUTION
  - However, we can *smoothen* the objective function $f(x)$
  - Using domain knowledge, create another function $f'(x)$
  - New function is *at least correlated* to the "true" one
  - Global optimum of $f(x)$ is **on or near** an optimum (local or global) of $f'(x)$



Flat objective functions

- Example: finding bugs in software / hardware
  - Candidate solution: input to the device under test
  - Fitness function: we found a bug (crash) / we did not find a bug
  - Smoothening: number of different functionalities activated
  - “The more functions activated, the more likely to trigger a bug”
- As there are no gradients, stochastic/approximate (EAs)
  - Meta (Facebook) uses **Sapienz** to test/debug user interfaces
  - Motorola used it to test phones (2008)
  - “Search-based software engineering”

Example



Gandini et al., *A framework for automated detection of power-related software errors in industrial verification processes*, 2010

Expensive objective functions

- Evaluating one candidate solution takes a lot of time
- Surrogate models
  - With domain knowledge, create function that is faster to compute
  - Same inputs, output is approximate but useful
  - Surrogate function discriminates solutions before long evaluation

Objective function

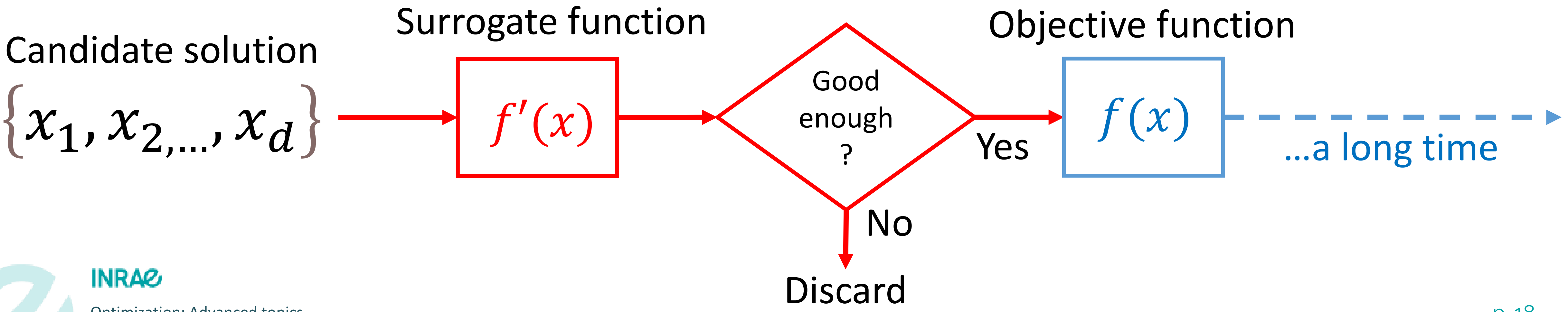
Candidate solution

$\{x_1, x_2, \ldots, x_d\} \longrightarrow \boxed{f(x)} \dashrightarrow$ Value

...a long time

Expensive objective functions

- Evaluating one candidate solution takes a lot of time
- Surrogate models
  - With domain knowledge, create function that is faster to compute
  - Same inputs, output is approximate but useful
  - Surrogate function discriminates solutions before long evaluation



Expensive objective functions

- Examples of surrogate models
  - Classifier (yes/no)
  - Regressor trained on samples of your original function
  - A physics-based model with less precision
  - (ML) Same function, but use only a subset of the samples

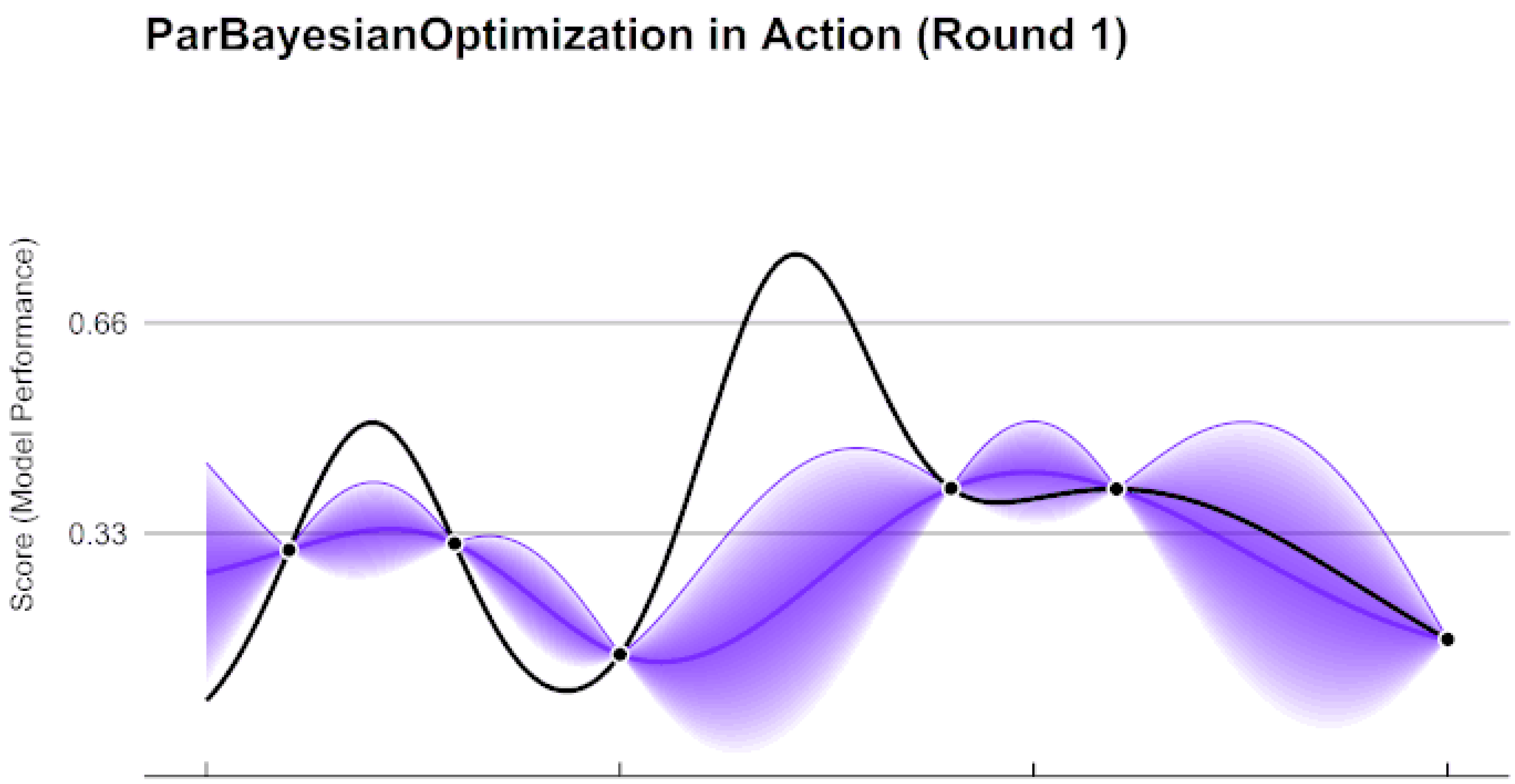
- However, surrogate models are **problem-specific**
- Require domain knowledge and expertise from user

Expensive objective functions

- Bayesian optimization
  - Use a generic surrogate model, a random function with a prior
  - Most common methodology uses kriging with Gaussian processes
  - The surrogate model is updated at each evaluation
  - Next point explored where surrogate model predicts improvement

- Gaussian processes also estimate incertitude around a point
- Sampling the point reduces uncertainty around it
- De-facto build approximate model of the search space

Expensive objective functions



Optimizing under incertitude

- Objective function is *noisy* or *stochastic*
  - Evaluate each candidate solution several times
  - Obtain a **mean** and a **standard deviation**

- Compare candidate solutions using **statistical tests**
- **Multi-objective**: add minimization of standard deviation

Dynamic objective functions

- Objective function: $\mathrm{y} = f(\boldsymbol{x}, t)$
- Assumption: the function does not change *too* abruptly
  - Re-evaluate current best solution(s)
  - If a change is detected, re-run optimization
  - Start search from an area around current best point
  - Store past solutions, eventually re-inject them (periodic?)

Questions?

Bibliography

- Lehman & Stanley, *Abandoning Objectives: Evolution through the Search for Novelty Alone*, 2011
- Alshawhan et al., *Deploying Search Based Software Engineering with Sapienz at Facebook*, 2018
- Frazier, *A Tutorial on Bayesian Optimization*, 2018
- Branke & Schmeck, *Designing Evolutionary Algorithms for Dynamic Optimization Problems*, 2003
- Mouret & Clune. *Illuminating search spaces by mapping elites*, 2015.

Images: unless otherwise stated, I stole them from the Internet. I hope they are not copyrighted, or that their use falls under the Fair Use clause, and if not, I am sorry. Please don't sue me.