

Natural Computing Cluster (NACO)

Presenter: Dr. Yingjie Fan

Universiteit Leiden

NACO



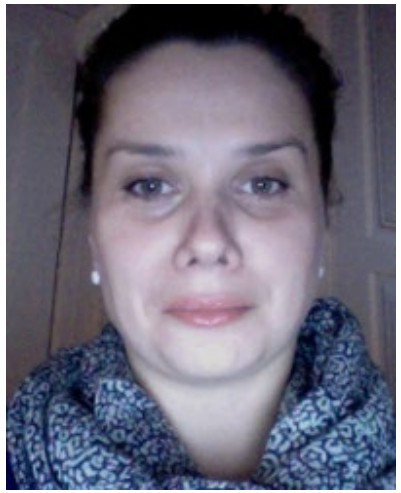
Prof. Dr. Thomas Bäck



Dr. Niki van Stein



Dr. Hao Wang



Dr. Anna V. Kononova



Dr. Elena Raponi



Dr. Yingjie Fan

Research Topics

- Multi-objective optimization
- Learning-based optimization
- Bayesian optimization
- Automated algorithm design using LLMs
- Hyperparameter optimization
- Optimization solution methods: evolutionary computation,metaheuristics and classical methods
- Explainable AI
- Decision support and applications in engineering, medicine, supply chain, logistics, and hydrogen transport and production

Multi-Objective Optimization

Case study: green delivery routing

Flexible time windows create room for better operational trade-offs.



Problem setup

- $ — Minimize route cost
- CO_2 — Minimize diesel use / emissions
- ☺ — Minimize customer dissatisfaction

Customers provide three ranked delivery windows; the optimizer selects route sequence and time window assignments.

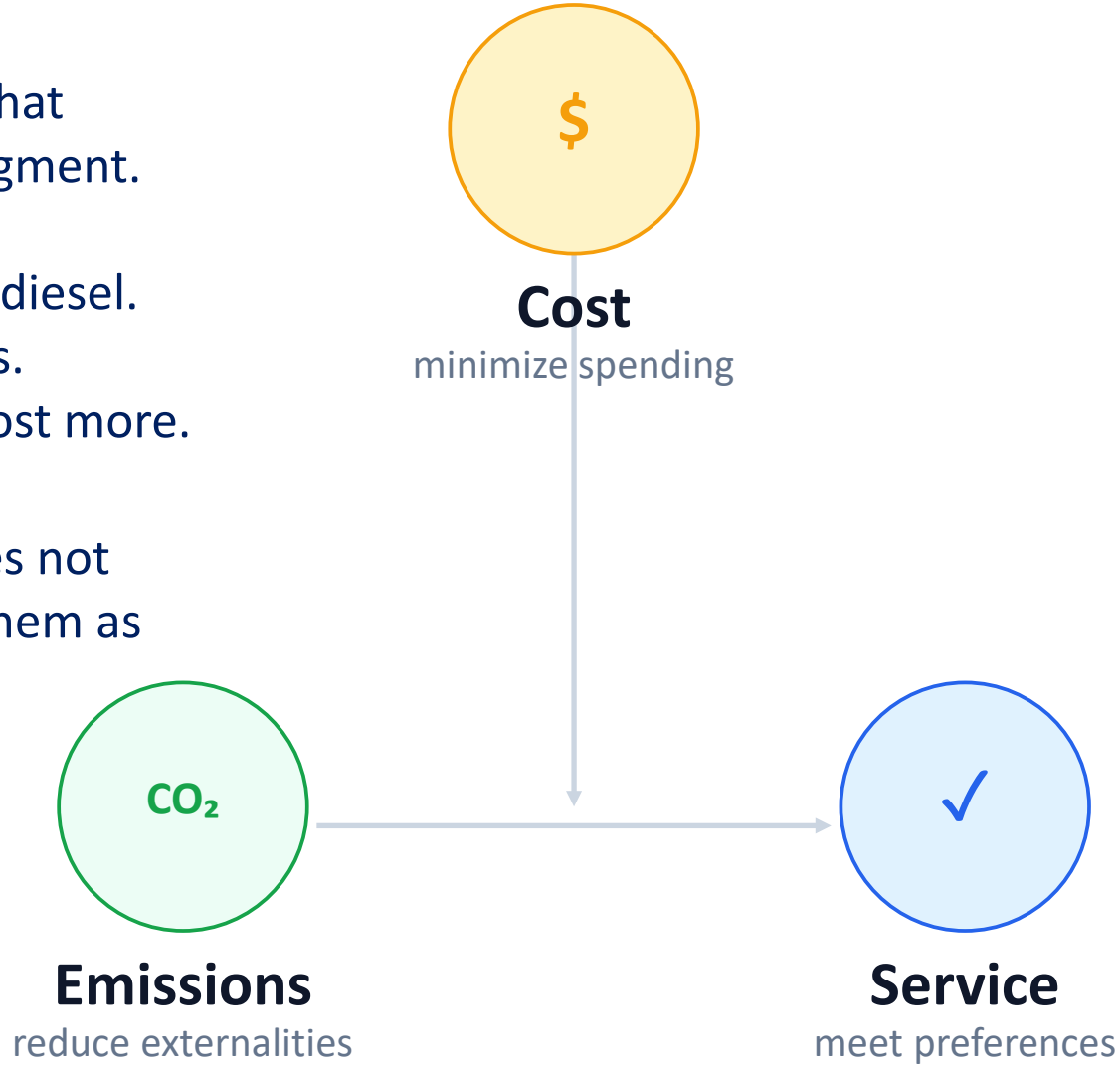
Gülmez, Burak, Michael Emmerich, and Yingjie Fan. "Multi-objective optimization for green delivery routing problems with flexible time windows." *Applied Artificial Intelligence* 38, no. 1 (2024): 2325302.

Why multi-objective optimization?

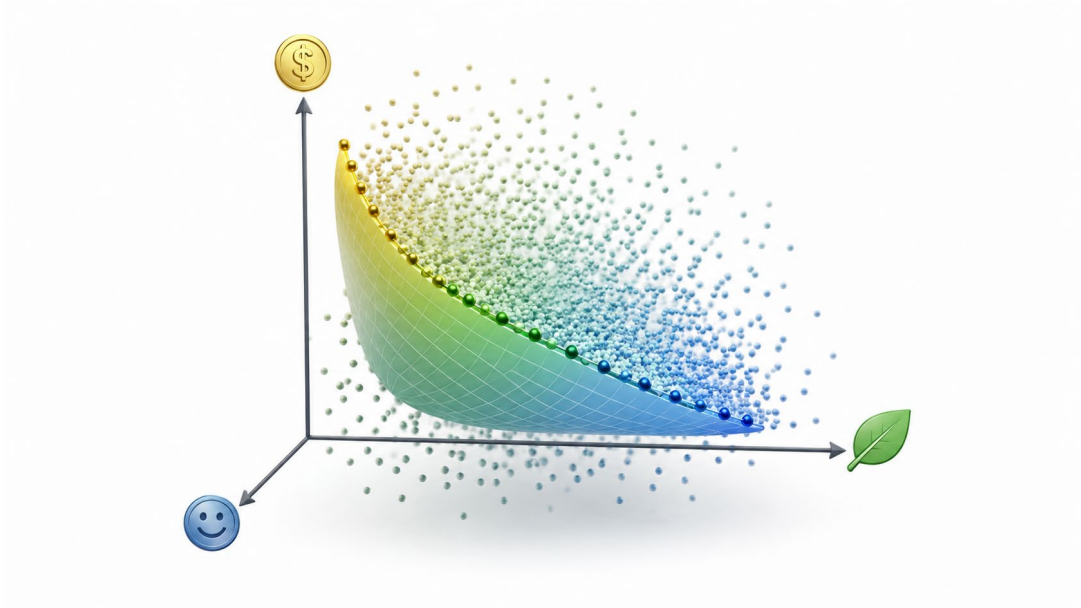
Real decisions contain conflicts that cannot be collapsed without judgment.

A cheaper route may burn more diesel.
A greener route may add detours.
A highly convenient route may cost more.

Multi-objective optimization does not hide these tensions. It exposes them as alternatives.



Pareto Fronts



For minimization, solution A dominates B if:

$f_i(A) \leq f_i(B)$ for all objectives i
and
$f_j(A) < f_j(B)$ for at least one objective j

The Pareto front is the set of objective-space points corresponding to non-dominated solutions.

Example

Route	Cost	CO_2	Late
A	90	50	2
B	110	65	2
C	80	85	1

Objectives

$$\min F_1 = T_e \cdot c_s + \sum_{i=0}^{n}\sum_{j=0}^{n}(d_{ij} \cdot c_d \cdot X_{ij}^{d}) + \sum_{i=0}^{n}\sum_{j=0}^{n}(d_{ij} \cdot c_e \cdot X_{ij}^{e}) \quad (1)$$

$$\min F_2 = \sum_{i=0}^{n}\sum_{j=0}^{n}\left(d_{ij} \cdot X_{ij}^{d}\right) \quad (2)$$

$$\min F_3 = \sum_{i=0}^{n}\left[\left(c_1^p \cdot P_{i1}\right) + \left(c_2^p \cdot P_{i2}\right) + \left(c_3^p \cdot P_{i3}\right) + \left(c_4^p \cdot P_{i4}\right)\right] \quad (3)$$

Constraints

We use the following two constraints (4) and (5) to ensure that customers are only visited once.

$$\sum_{i=0}^{n} X_{ij} = 1, i \neq j, \forall i \in N \quad (4)$$

$$\sum_{j=0}^{n} X_{ij} = 1, i \neq j, \forall j \in N \quad (5)$$

...

Solution Methods

Scalarization methods

Convert multiple objectives into one score.

Weighted sum
ε-constraint
Goal programming

Useful when preferences are known before search.

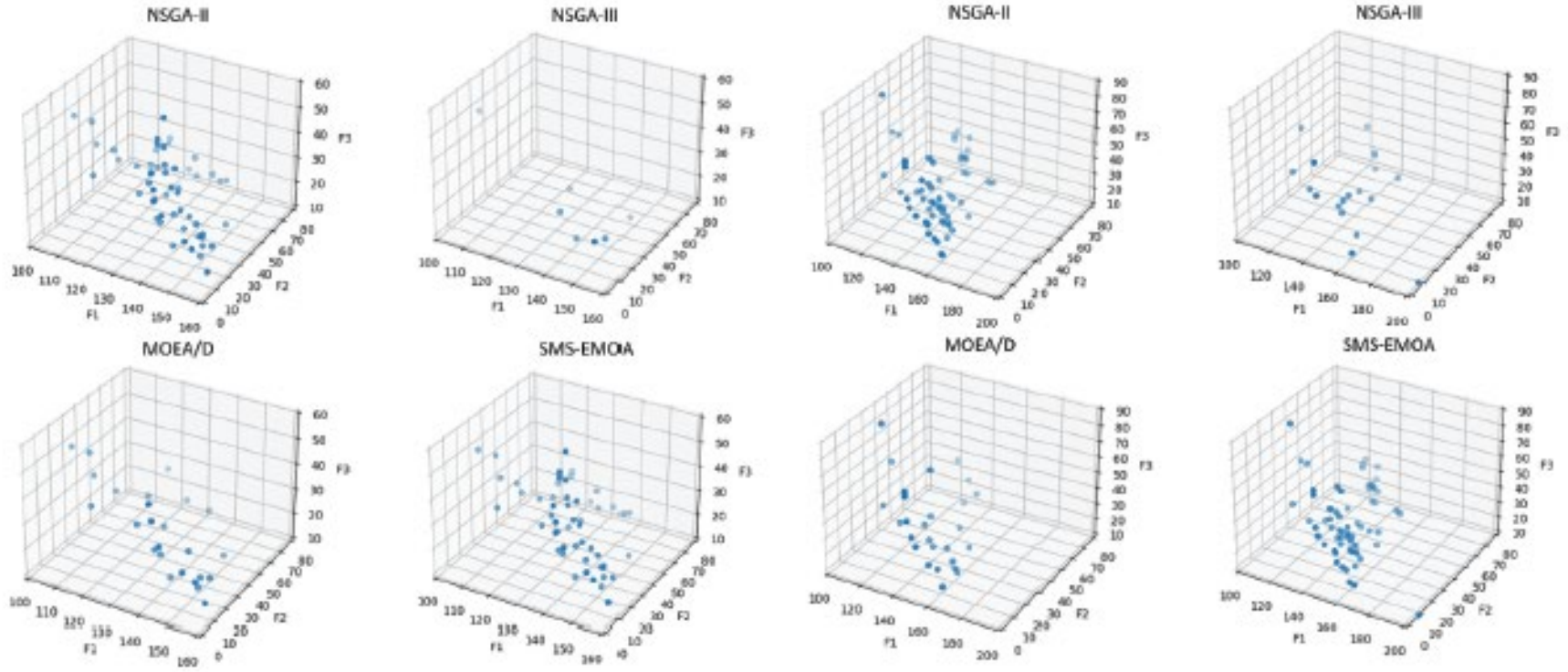
Pareto-based methods

Search for a set of non-dominated solutions.

NSGA-II / NSGA-III
MOEA/D
SMS-EMOA

Useful when preferences are uncertain or contested.

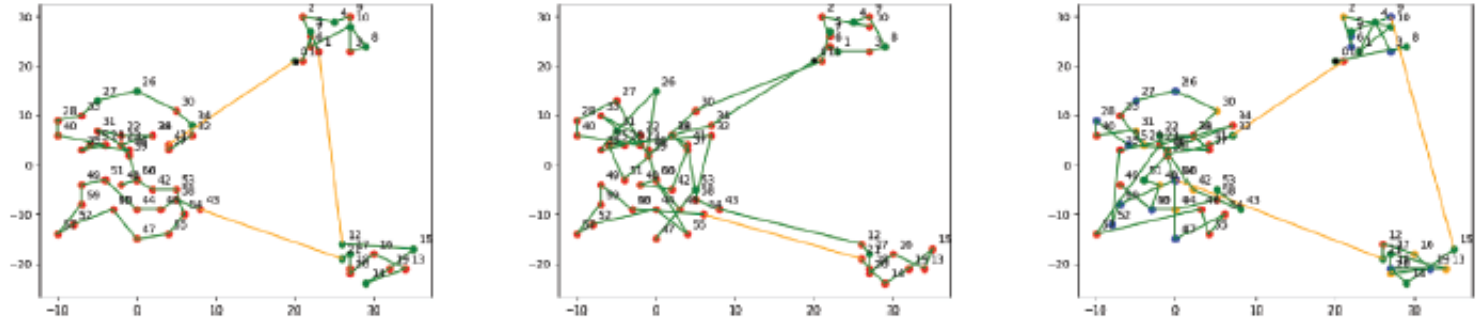
Results



(a) 10-node instance[a] (b) 11-node instance[a]

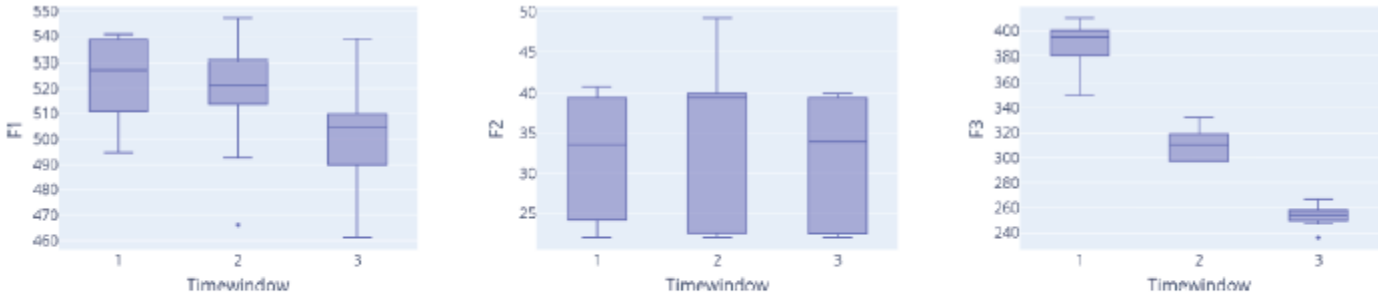
Figure 3. Pareto fronts of small instances.

Results



(a) minimum cost[a] (b) lowest emissions[a] (c) highest satisfaction[a]

Figure 6. Routes for different objectives in 60-customer instance.



(a) Cost (b) Emissions (c) Customer Dissatisfaction

Figure 9. Time windows and objectives in 60-customer instance (Pareto optimal solutions generated with NSGA-II).

Solution Quality – Hypervolume

Table 4. Hypervolume results.

	Hypervolume		Time	
Algorithm	Mean	Stdev	Mean	Stdev
(a) 60-customer instance				
NSGA-II	0.305	0.036	1132.646	33.643
NSGA-III	0.291	0.034	1092.858	9.148
MOEA/D	0.294	0.036	415.255	8.962
SMS-EMOA	0.299	0.035	2865.694	428.330
(b) 40-customer instance				
NSGA-II	0.427	0.032	879.861	18.293
NSGA-III	0.412	0.034	732.394	15.890
MOEA/D	0.420	0.024	272.534	6.638
SMS-EMOA	0.430	0.030	2173.747	510.884
(c) 30-customer instance				
NSGA-II	0.540	0.026	688.565	8.023
NSGA-III	0.529	0.026	589.528	6.260
MOEA/D	0.519	0.020	242.609	4.137
SMS-EMOA	0.534	0.023	3129.497	557.273

Hypervolume measures the objective-space region dominated by a Pareto solution set relative to a reference point, with larger values generally indicating better convergence and diversity.

What the Pareto front teaches

The value is not just better routes—it is interpretable trade-off information.

Trade-off examples from the green-delivery study

–74%

possible emission reduction

+ up to 50% cost; − up to 9% satisfaction

+82%

possible satisfaction gain

+ up to 53% cost; + up to 29% emissions

3 Time Windows

more routing flexibility

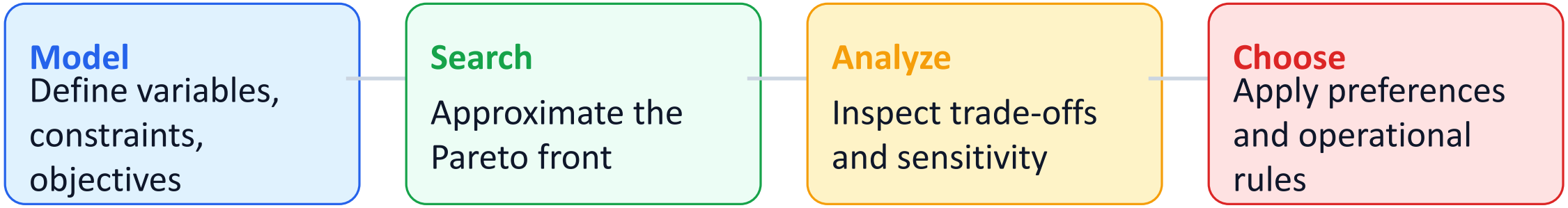
lower cost and higher satisfaction than one window

Algorithm choice also matters

NSGA-II and SMS-EMOA achieved stronger hypervolume, while MOEA/D was fastest but generally lower quality.

Trade-off numbers and algorithm comparison summarized from Gülmez, Emmerich & Fan (2024), results and managerial-insights sections.

Multi-objective optimization for decision making – A practical workflow

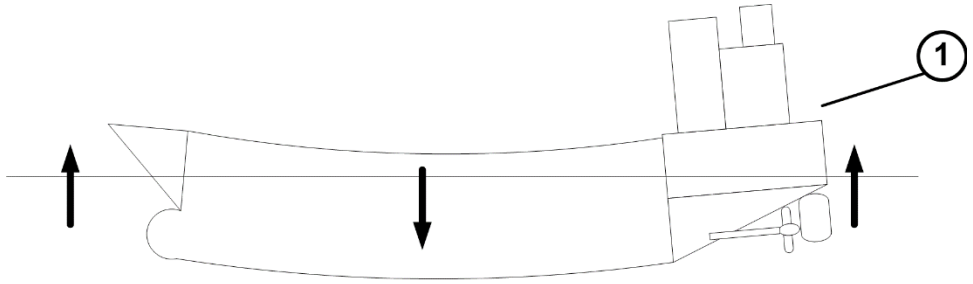


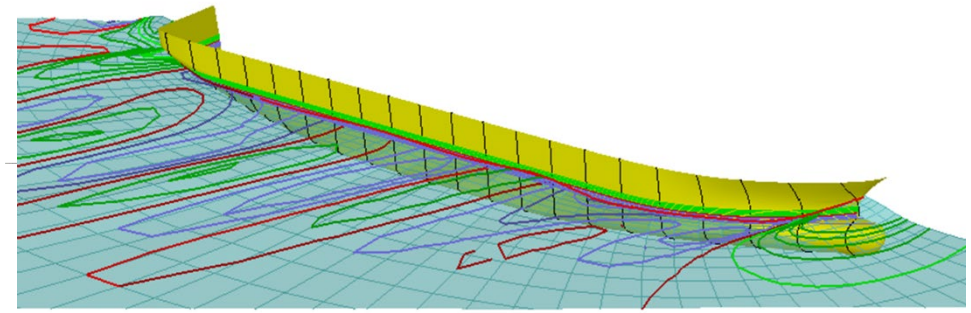
Important: the final solution is not “found” by mathematics alone. It is selected using stakeholder values, budget, policy constraints, and risk tolerance.

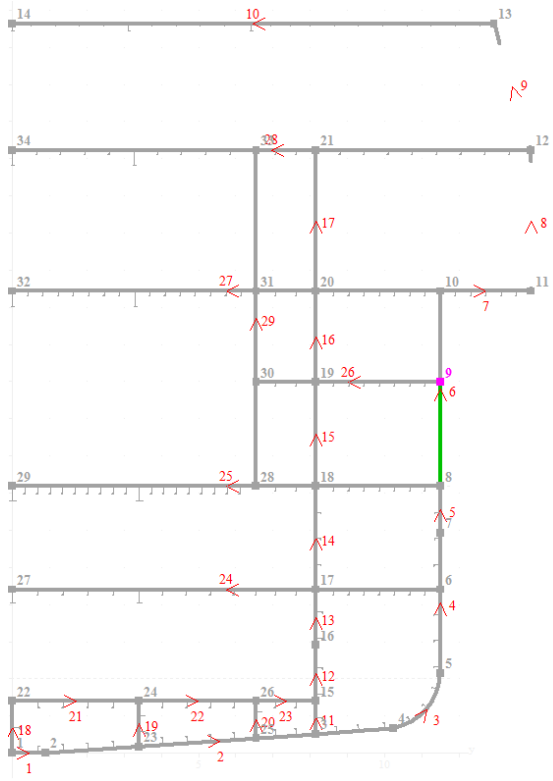
Other Real-World Applications – Dredger Ship Design

- Objectives: $F(\vec{X}) = [f_1(\vec{X}), f_2(\vec{X}), \ldots, f_k(\vec{X})]$

  Minimize: steel weight / building costs
  Minimize: resistance / operational expenses







de Winter, Roy, Jan Furustam, Thomas Bäck, and Thijs Muller. "Optimizing ships using the holistic accelerated concept design methodology." In *Practical design of ships and other floating structures*, pp. 38-50. Singapore: Springer Singapore, 2019.

Challenges

- Large solution space: The number of non-dominated solutions can grow rapidly with the number of objectives.
- Constraints and uncertainty: Real-world constraints, noisy data, and uncertain parameters make optimisation more difficult.
- Decision-making: Selecting one final solution from the Pareto front requires preferences or priorities from the decision-maker.

van der Laag, Robin, Hao Wang, Thomas Bäck, and Yingjie Fan. "Center-outward q-dominance: a sample-computable proxy for strong stochastic dominance in stochastic multi-objective optimisation." In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 31, pp. 26037-26045. 2026.

Learning-based Optimization

Dr. Yingjie Fan



Universiteit Leiden

Reinforcement Learning for Container Stowage Planning Problems (CSPP)

Problem Description: Containers are stored in stacks in the yard and must be loaded onto vessels in a predetermined order. To retrieve a container blocked by others stacked above it, shifting operations are often unavoidable.

Goal: to determine the loading sequence that minimizes

- the total number of shifting operations.
- the total loading time

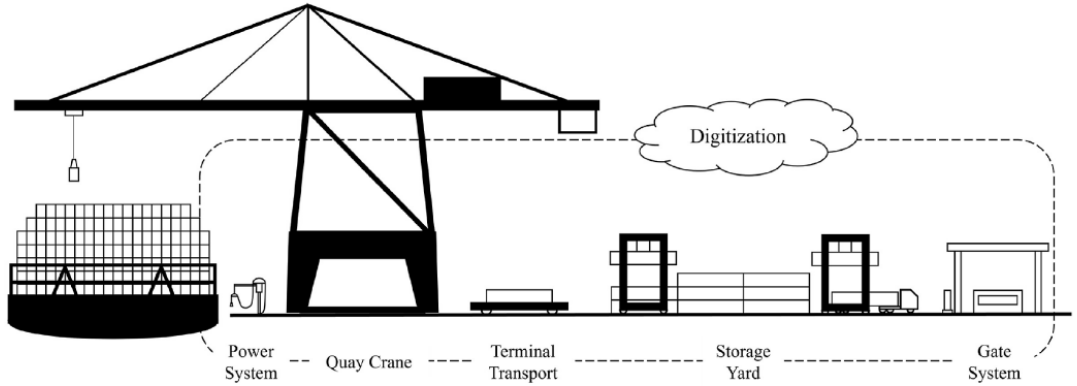
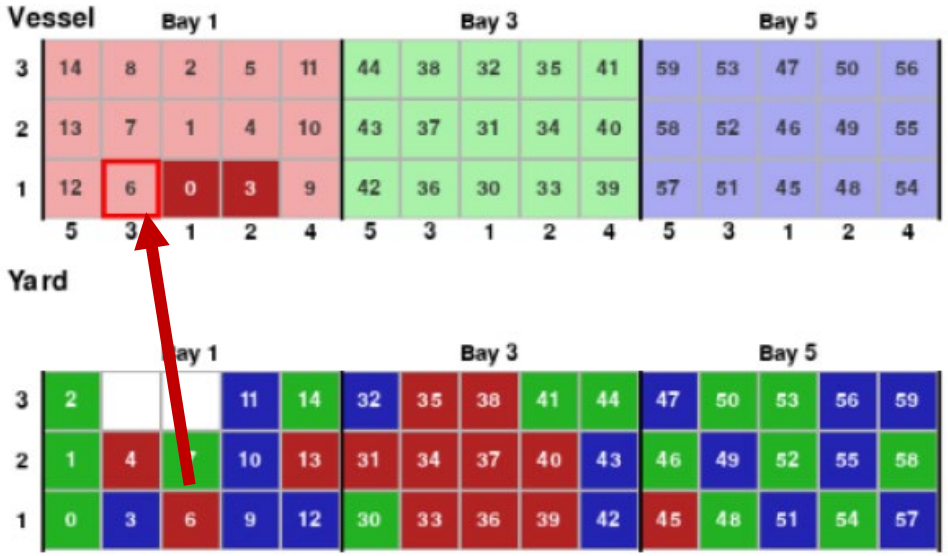


Figure 2: *Stowage operation process* (Zhou et al., 2022).



Reinforcement Learning for Container Stowage Planning Problems

Problem Description: Containers are stored in stacks in the yard and must be loaded onto vessels in a predetermined order. To retrieve a container blocked by others stacked above it, shifting operations are often unavoidable.

Goal: to determine the loading sequence that minimizes

- the total number of shifting operations.
- the total loading time

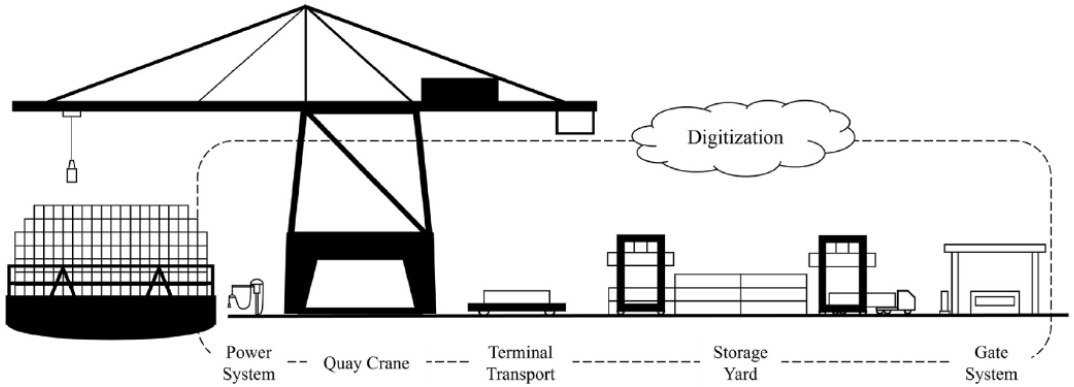
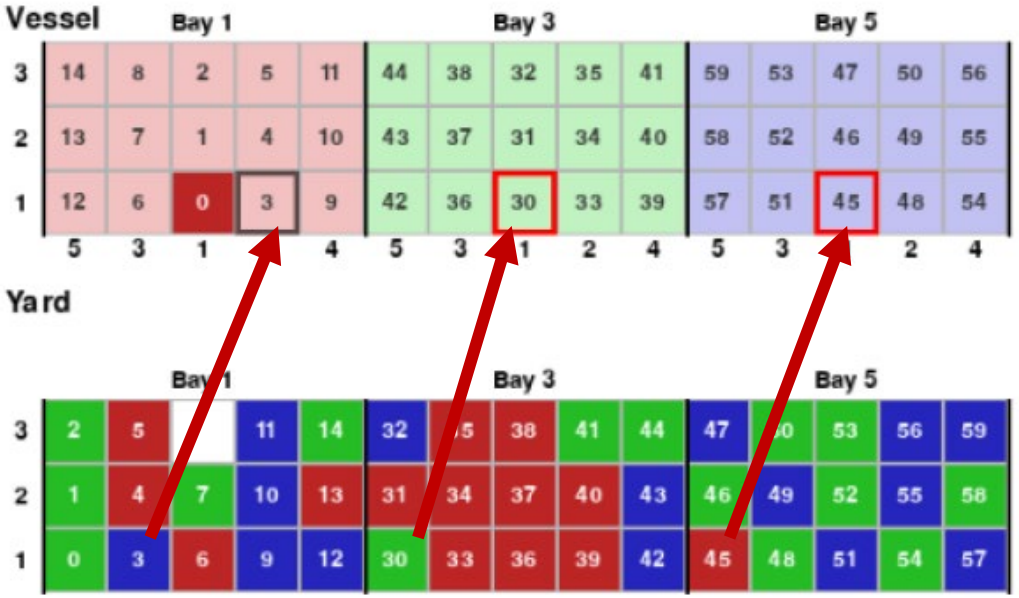


Figure 2: *Stowage operation process* (Zhou et al., 2022).



Problem Modelling

The Container Stowage Planning Problem (CSPP) assigns m containers from set C to n available ship slots in set S. The slots are considered sequentially, and at each step, one suitable unused container is selected. The objective is to find an ordered sequence

$$P =< p_1, p_2, \dots, p_m >$$

that maximises the objective, where $p_i \in C$ and $p_i \neq p_j \; \forall i \neq j$

Objective:

$$minimize \{T_{shift} + T_{move_to_load}\}$$

Huang, Yunqi, Nishith Chennakeshava, Alexis Carras, Vladislav Neverov, Wei Liu, Aske Plaat, and Yingjie Fan. "A Benchmark Study of Deep Reinforcement Learning Algorithms for the Container Stowage Planning Problem." In *International Conference on Dynamics in Logistics*, pp. 103-118. Cham: Springer Nature Switzerland, 2026.

Constraints

- **Slot and container availability:** Use only vacant loading slots and containers that are present in the yard and listed for loading.
- **Bay adjacency:** A slot is invalid if its adjacent bay is occupied, particularly where two 20-foot bays share one 40-foot position.
- **Hatch status:** When the hatch is open, load containers below deck; when closed, use only above-deck slots.
- **Loading sequence:** Load slots in increasing bay, row, and tier order, with seaside slots loaded before landside slots.
- **Weight limits:** Do not exceed the vessel or bay weight limits, and do not place heavier containers above lighter ones.

Vessel Structure

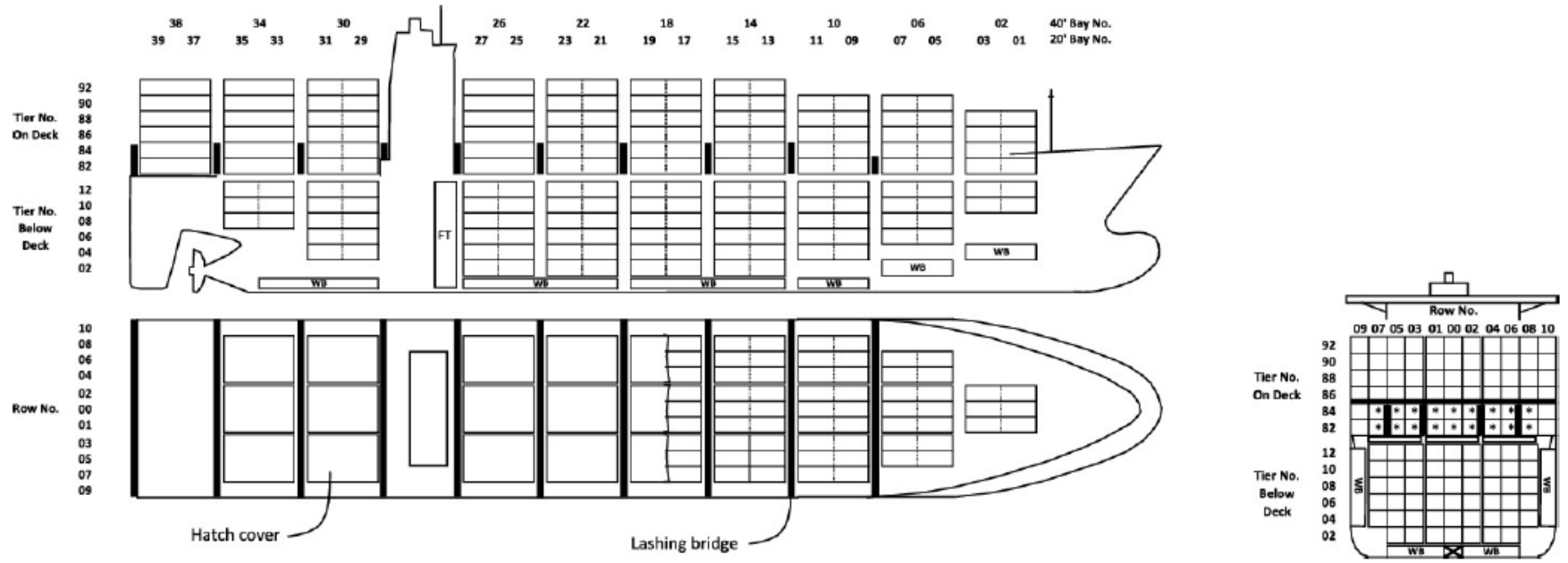


Figure 1: *Vessel structure* (van Twiller et al., 2024).

Modelling

The problem is modelled as a sequential decision process rather than generating the complete plan at once.

Reinforcement-Learning Framework – Markov Decision Process

$$\mathcal{M} = (S, A, P, R)$$

- **State** S: vessel, yard, target slot and crane information.
- **Action** A: select a container, or a container–crane pair.
- **Transition** P: update the vessel, yard and crane states after loading.
- **Reward** R: penalise reshuffling, invalid actions and operational costs.

The agent learns through repeated interaction with a simulated environment.

Simulation Environments

Environments developed and used

- **SPGE:** basic container stowage environment.
- **SPGE-MC:** joint container selection and crane scheduling.
- **SPAEC:** multi-agent formulation in which each crane acts as an agent.
- **LME:** industrial simulator using real-world vessel and yard information.

The vessel and yard are represented as three-dimensional slot structures. Slot attributes include:

- bay, row and tier coordinates;
- occupancy status;
- container or slot group.

State

Basic SPGE observation

- Current vessel-slot information.
- Current yard-slot information.
- The next vessel slot that must be filled.
- All information is flattened and concatenated into one observation vector.

Crane-scheduling extensions

- Crane availability.
- Containers currently handled by cranes.
- Crane sequencer states.
- Global timing information.

State

(a) Vessel State

Slot Number	Bay	Row	Tier	Occupancy	Group
0	1	1	1	1	0
3	1	2	1	1	0
6	1	3	1	0	0
30	3	1	1	0	1
...	...	...	...	...	...

(b) Yard State

Slot Number	Bay	Row	Tier	Occupancy	Group
0	1	1	1	1	1
1	1	1	2	1	1
4	1	2	2	1	0
5	1	2	3	0	-1[1]
...	...	...	...	...	...

[1] Since this yard slot has no container, the container attribute *group* does not apply.

(a) Crane Availability State

Crane 1	Crane 2	Crane 3
100[1]	0	0

(b) Crane Operating State

Crane 1	Crane 2	Crane 3
8	-1[2]	-1

(c) Crane Sequencer State

Crane 1	Crane 2	Crane 3
3	30	45

[1] Indicates that the crane will be available in 100 seconds.

[2] Indicates crane not currently loading a container.

Action Space

Basic SPGE

- $a_t = selected\ yard\ slot$
- The agent selects the next container to load.
- Action-space size equals the number of yard slots.

SPGE-MC

- $a_t = (container, crane)$
- A central agent jointly selects a container and an available crane.
- Action-space size:
- $| A | = n_C n_{CR}$

SPAEC

- The active crane selects a container.
- Crane assignment is handled through the multi-agent rotation mechanism.

Reward and Optimisation Objective

Reward for a valid action

- $r_t = -n_{shift,t}$

More reshuffling produces a larger negative reward. Maximising cumulative reward therefore minimises total shifters.

Invalid action

- The environment initially assigns a large penalty of -100.
- The environment state remains unchanged.

Action Masking

Many actions are invalid because:

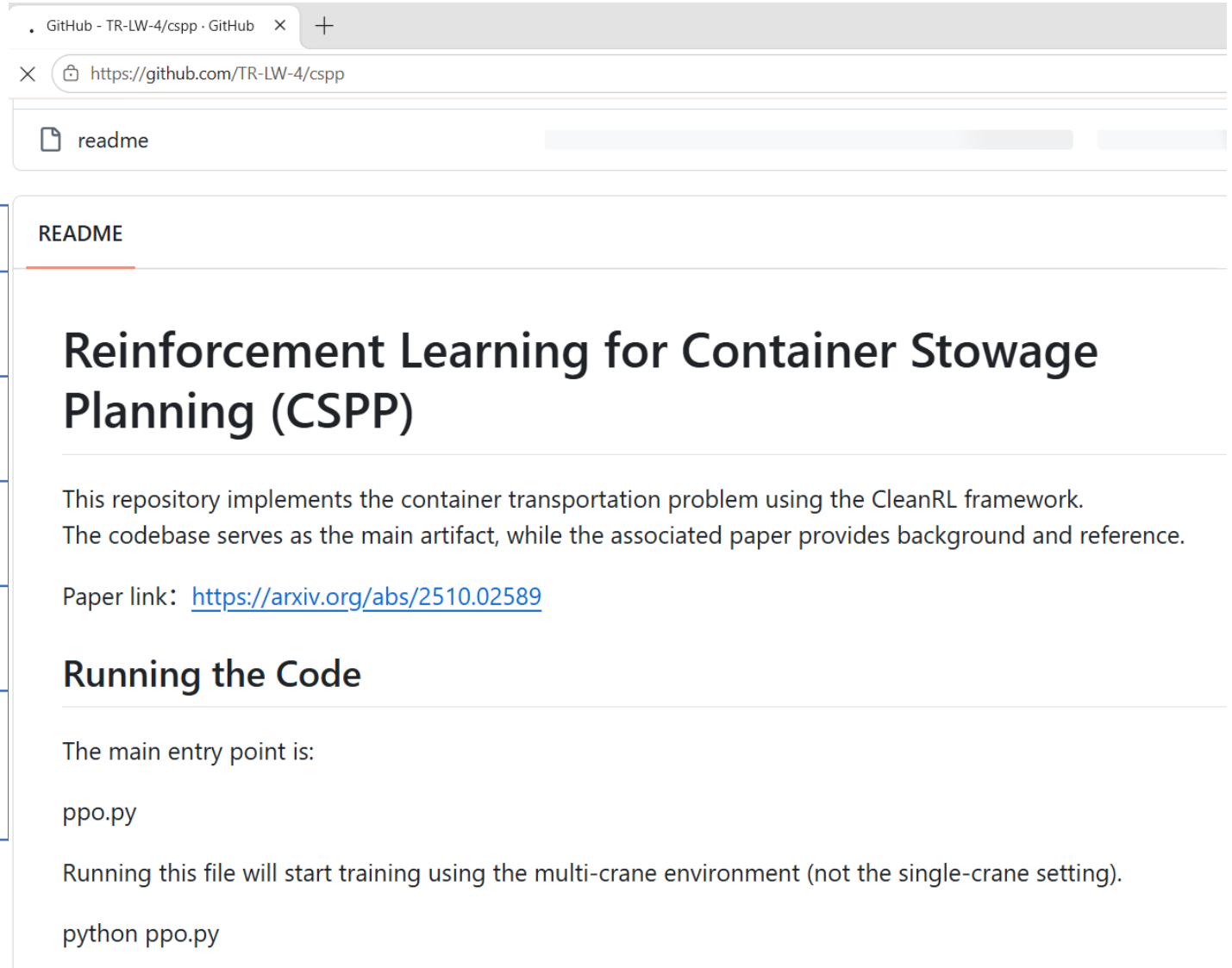
- a yard slot is empty;
- the container group does not match the vessel slot;
- a crane is unavailable;
- an operational constraint is violated.

Implementation

- Policy-based methods: invalid-action logits are set to $-\infty$.
- Value-based methods: invalid-action Q-values are set to $-\infty$.
- Invalid actions therefore have zero probability of being selected.
- Action masking substantially improves convergence, particularly for large action spaces.

Algorithms

Algorithm	Category	Main characteristic
DQN	Value-based	Learns expected Q-values
QR-DQN	Distributional value-based	Learns a distribution of returns
A2C	Actor–critic	Uses actor, critic and N-step returns
PPO	Actor–critic	Limits policy updates through clipping
TRPO	Actor–critic	Constrains updates using a KL-divergence trust region



[GitHub - TR-LW-4/cspp · GitHub](https://github.com/TR-LW-4/cspp)

Training and Evaluation

Multiple scenarios progressively increase:

- vessel and yard size;
- number of containers;
- number of container types;
- number of cranes.

Evaluation is repeated several times to reduce the effect of randomness.

Performance indicators: episode reward; total shifters; total operation time.

Results

Algorithm	Scenario	KPI	SPGE-MC Value	SPAEC Value	Diff. (SMC-SPAEC)
TRPO	Scenario 6	Shifters	4.2 (2.25)	4.8 (2.82)	-0.5
		Op. Time	1025.9 (85.95)	983.8 (86.10)	42.1
	Scenario 7	Shifters	**148.2 (2.82)**	**146.3 (1.86)**	1.9
		Op. Time	**6401.2 (52.76)**	**6283.7 (61.26)**	117.6
	Scenario 8	Shifters	**87.7 (2.54)**	**89.3 (2.20)**	-1.6
		Op. Time	**3519.9 (47.36)**	**3431.0 (49.52)**	88.9
PPO	Scenario 6	Shifters	4.9 (1.84)	5.0 (1.85)	-0.1
		Op. Time	1003.8 (75.47)	**949.9 (85.14)**	54.0
	Scenario 7	Shifters	151.0 (3.07)	150.7 (2.80)	0.3
		Op. Time	6405.7 (88.58)	6380.6 (83.50)	25.2
	Scenario 8	Shifters	95.5 (3.73)	97.9 (3.52)	-2.4
		Op. Time	3616.5 (80.31)	3532.6 (70.03)	83.9
A2C	Scenario 6	Shifters	6.4 (3.85)	5.3 (2.50)	1.0
		Op. Time	1084.7 (143.37)	1049.0 (123.74)	35.7
	Scenario 7	Shifters	157.4 (3.09)	157.1 (3.22)	0.3
		Op. Time	6518.2 (84.91)	6451.8 (86.30)	66.4
	Scenario 8	Shifters	107.9 (6.03)	117.3 (6.32)	-9.3
		Op. Time	3699.7 (123.27)	3753.2 (162.01)	-53.5
DQN	Scenario 6	Shifters	4.4 (1.47)	4.3 (1.60)	0.1
		Op. Time	1000.6 (61.30)	982.6 (62.53)	18.0
	Scenario 7	Shifters	156.7 (5.06)	157.6 (8.08)	-0.9
		Op. Time	6577.9 (153.82)	6592.8 (260.21)	-14.9
	Scenario 8	Shifters	104.4 (6.99)	100.9 (6.97)	3.5
		Op. Time	3702.6 (131.09)	3603.4 (83.71)	99.3
QR-DQN	Scenario 6	Shifters	**3.9 (1.24)**	**3.5 (1.14)**	0.4
		Op. Time	**983.8 (60.57)**	1001.0 (74.36)	-17.2
	Scenario 7	Shifters	150.4 (3.72)	152.3 (2.79)	-1.8
		Op. Time	6412.4 (114.80)	6492.9 (155.58)	-80.5
	Scenario 8	Shifters	94.8 (4.11)	97.9 (4.11)	-3.1
		Op. Time	3607.2 (98.07)	3628.6 (126.93)	-21.3

Challenges and Next Steps

- Transfer learning
- Large-size problems
- Robustness in facing uncertainties
- Loading + Unloading
- Combine with land transportation of containers (job assignment + stowage planning)
- ...

LLMs for Evolutionary Code Optimization

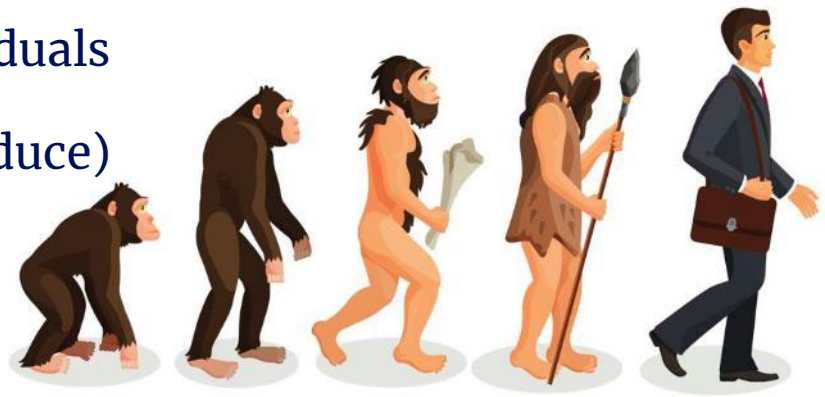
Niki van Stein

Evolutionary Computation

- **What is Evolutionary Computation?**
  - A nature-inspired approach to optimization
  - Inspired by the notion of *survival of the fittest* from Darwinian Evolution and modern genetics

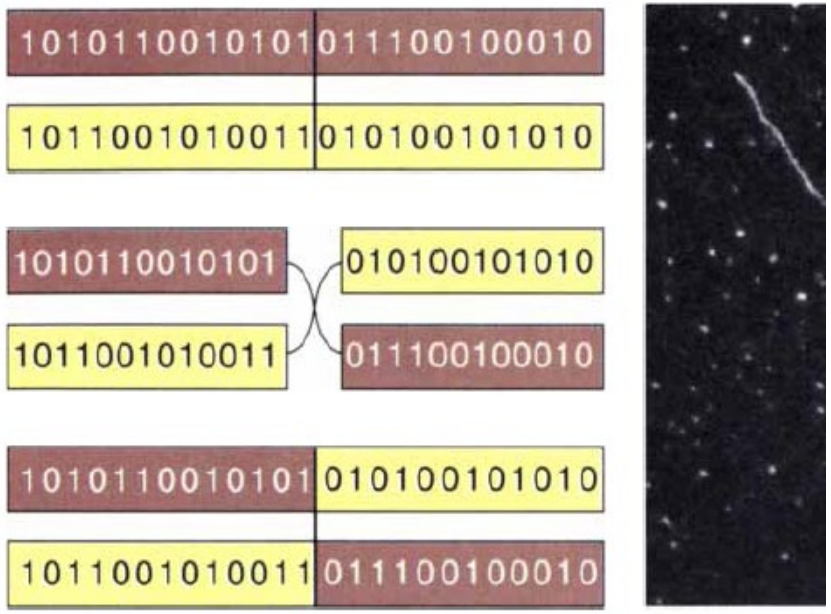
  Evolutionary computation refers to computer-based problem-solving systems that use computational models of evolutionary processes, such as natural selection, survival of the fittest and reproduction as the fundamental components of such computational systems.

- **What do we need?**
  - An **encoding** of solutions to the problem as a chromosome
  - A **function** to evaluate **fitness**, or survival strength of individuals
  - **Initialization** of the initial population
  - **Selection** operators (those that survive are those that reproduce)
  - **Reproduction** operators (crossover and mutation)



Genetic Algorithms (GA)

“Computer programs that ‘evolve’ in ways that resemble natural selection can solve complex problems even their creators do not fully understand.” – John H. Holland

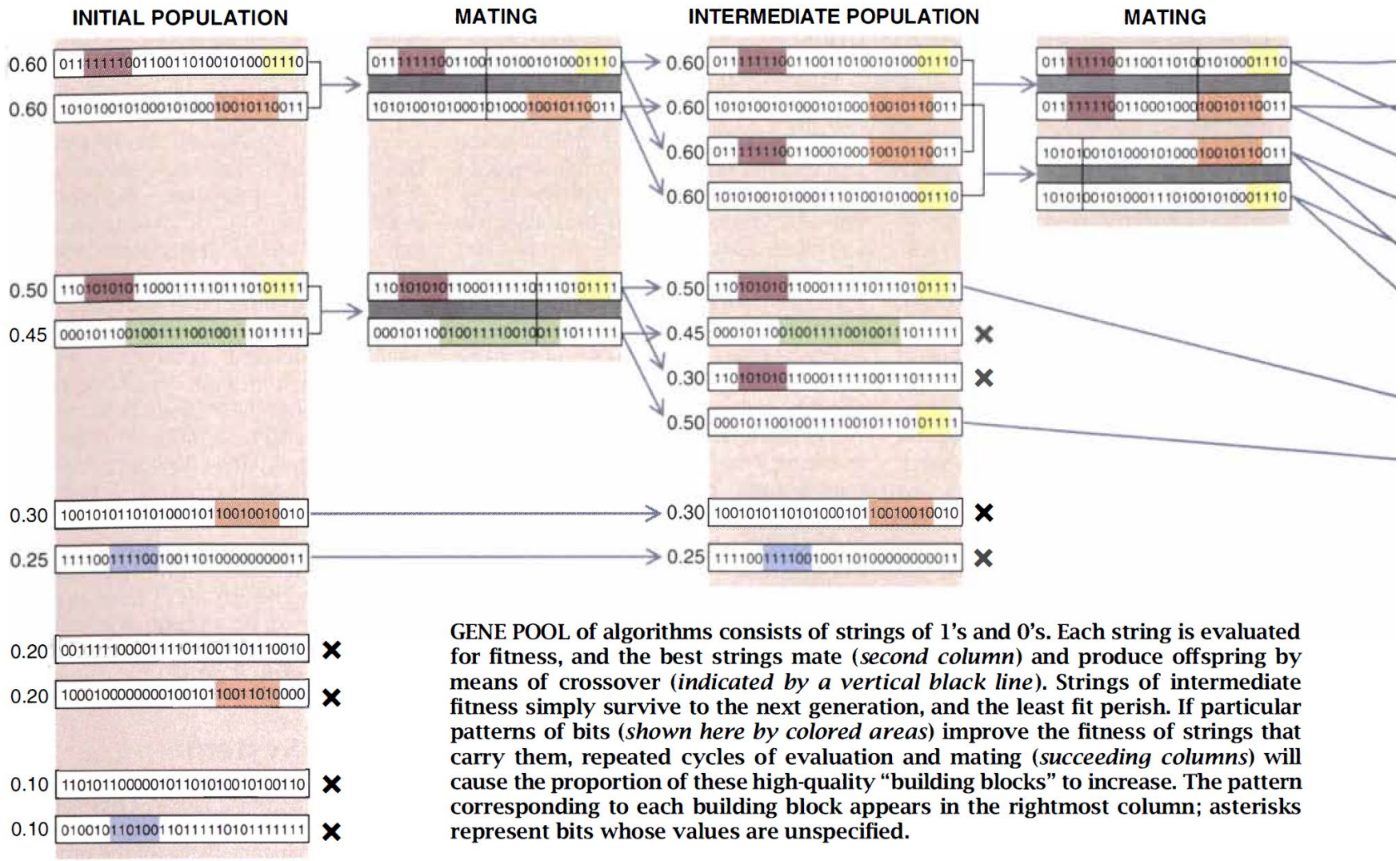


CROSSOVER is the fundamental mechanism of genetic rearrangement for both real organisms and genetic algorithms.

Holland, J. H. (1992). Genetic algorithms. *Scientific american*, *267*(1), 66-73.

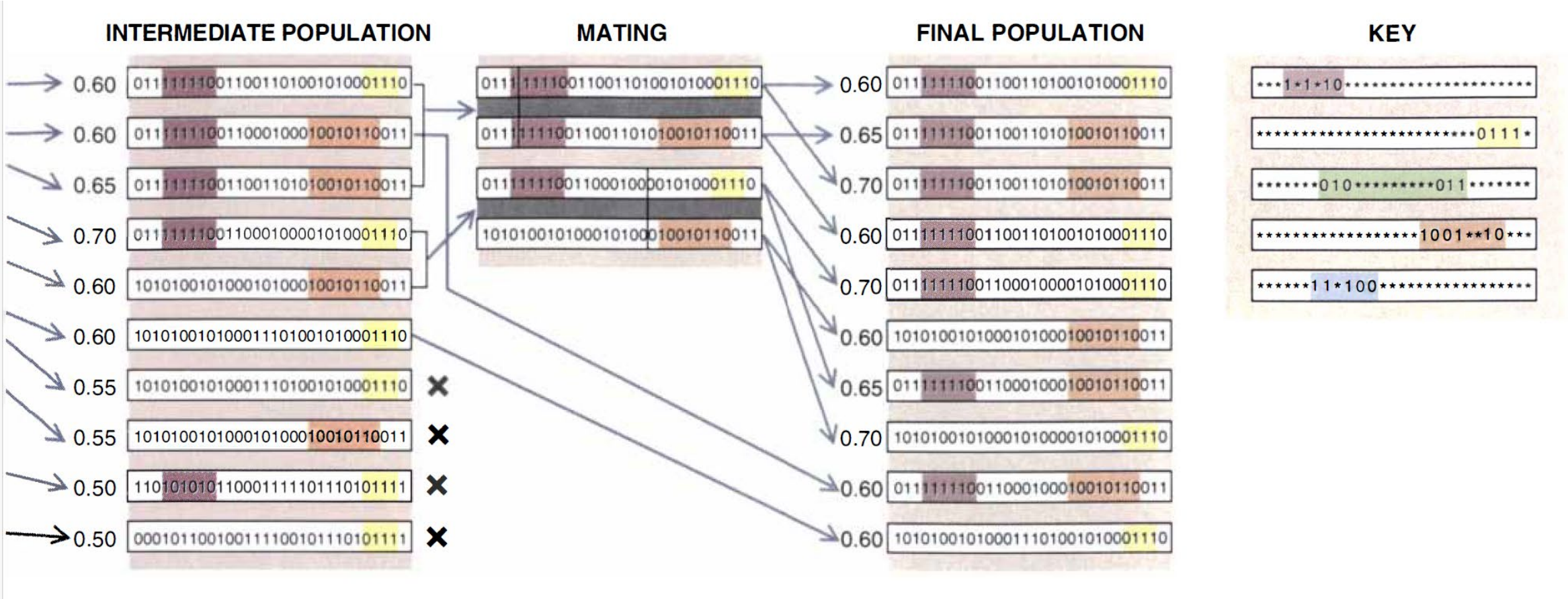
Genetic Algorithms (GA)

- Initialization
- Evaluation
- Selection
- Crossover
- Mutation



GENE POOL of algorithms consists of strings of 1's and 0's. Each string is evaluated for fitness, and the best strings mate (*second column*) and produce offspring by means of crossover (*indicated by a vertical black line*). Strings of intermediate fitness simply survive to the next generation, and the least fit perish. If particular patterns of bits (*shown here by colored areas*) improve the fitness of strings that carry them, repeated cycles of evaluation and mating (*succeeding columns*) will cause the proportion of these high-quality "building blocks" to increase. The pattern corresponding to each building block appears in the rightmost column; asterisks represent bits whose values are unspecified.

Genetic Algorithms (GA)



LLMs and Optimization



- Since 2023, many papers are addressing (simpler) tasks:

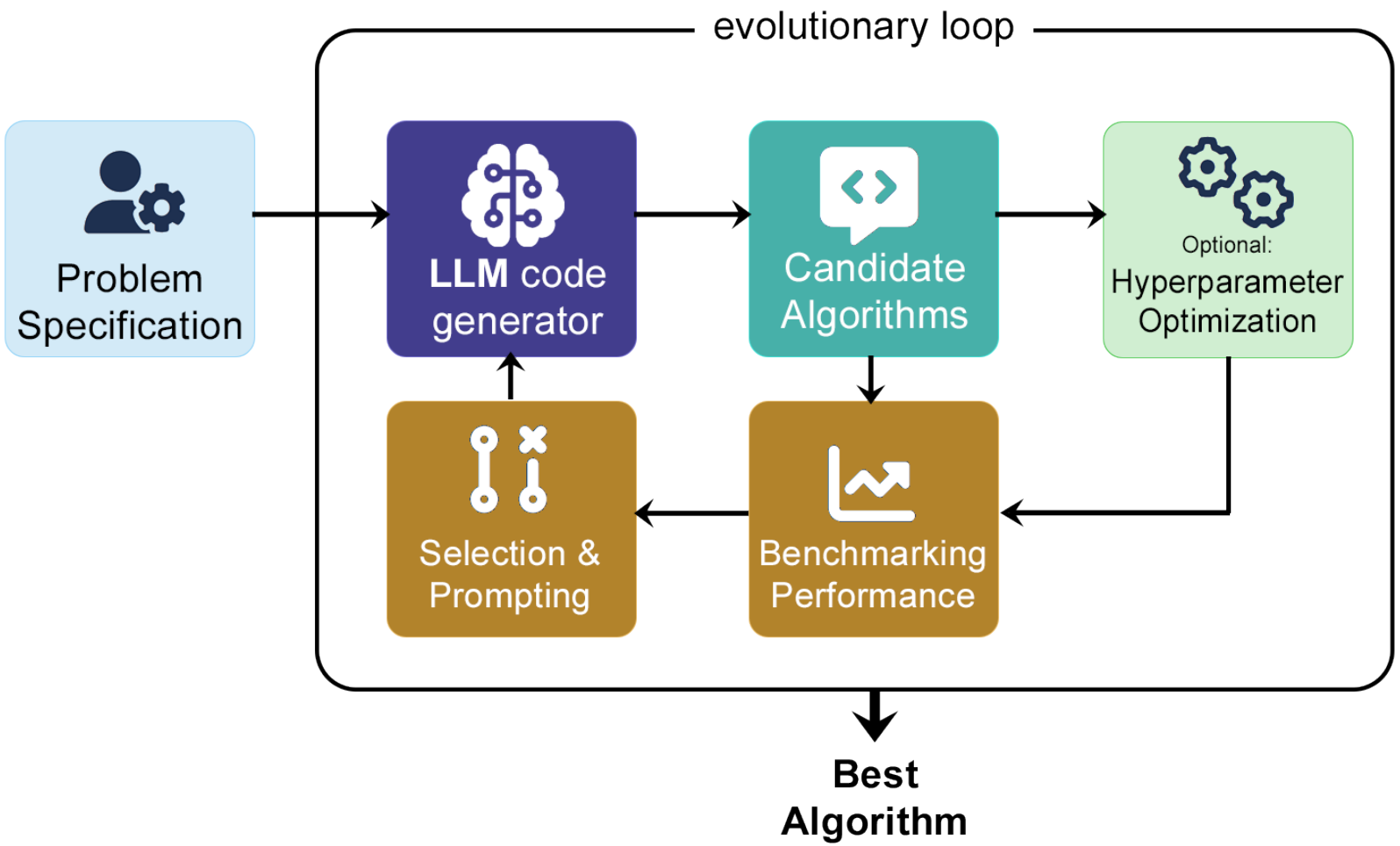
- LLMs and optimizing prompts/data selection
- LLMs as recommenders of hyperparameters
- LLMs acting as optimizers

- **Full algorithmic/models design by LLMs**



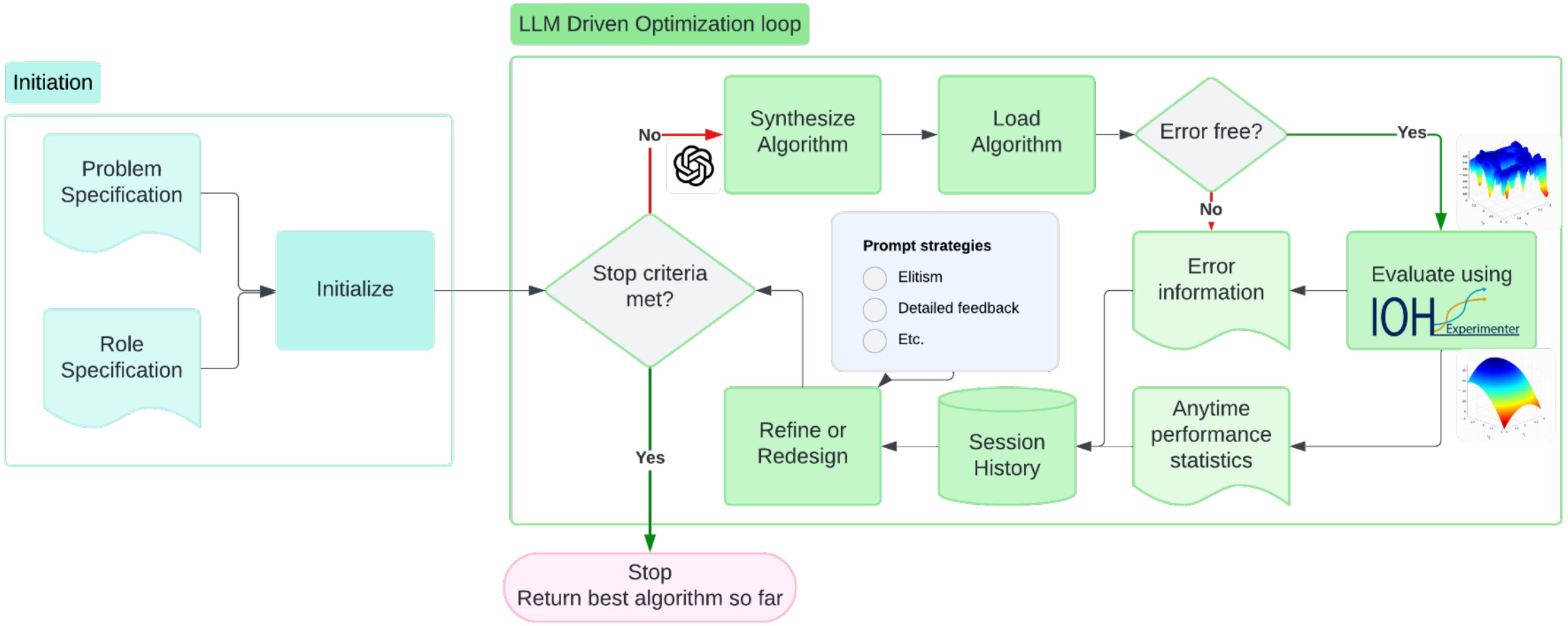
LLaMEA
An LLM Evolutionary algorithm

LLaMEA Framework



van Stein, N., & Bäck, T. (2024). LLaMEA: A large language model evolutionary algorithm for automatically generating metaheuristics. *IEEE Transactions on Evolutionary Computation*.

Basic Idea: Use LLMs to generate and iteratively optimize Algorithms



The Prompts:

Detailed task prompt S

```
Your task is to design novel metaheuristic
 ↪  algorithms to solve black box
 ↪  optimization problems.
The optimization algorithm should handle a
 ↪  wide range of tasks,
which is evaluated on a large test suite of
 ↪  noiseless functions.
Your task is to write the optimization
 ↪  algorithm in Python code.
The code should contain one function `def
 ↪  __call__(self, f)`,
which should optimize the black box function
 ↪  `f` using
`budget` function evaluations.
The f() can only be called as many times as
 ↪  the budget allows.
An example of such code is as follows:
```
<initial example code>
```
Give a novel heuristic algorithm to solve
this task.
Give the response in the format:
# Name: <name of the algorithm>
# Code: <code>
```

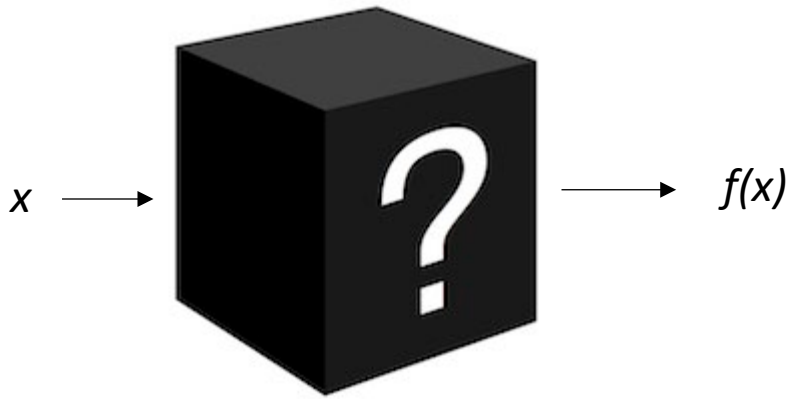
Feedback prompt template F

```
<Task prompt S>
<List of previously generated algorithm names
 ↪  with mean $AOCC$ score>
<selected algorithm to refine (full code) and
 ↪  mean and std $AOCC$ scores>
Either refine or redesign to improve the
 ↪  algorithm.
```

Black-Box Optimization

Black-box Optimization

- Also known as derivative-free optimization
- Structure of objective function and constraints is unknown, unexploitable, or non-existent
- It works like a “black-box”:
  - Possible to query it with an $\boldsymbol{x}$
  - It will provide $f(\boldsymbol{x})$ as an answer
- Examples:
  - Legacy code: no access to what is inside the library
  - Simulation models
  - Real-life experiments: crash tests, chemical reactions, etc.



Black-box Optimization

How much does a function evaluation “cost”?

With *cost* we mean any measure of the resources needed to evaluate the function (convertible somehow to money)



Cheap function: thousands of evaluations



Costly function: a few evaluations (typically < 200)

Black-box Optimization

Some examples

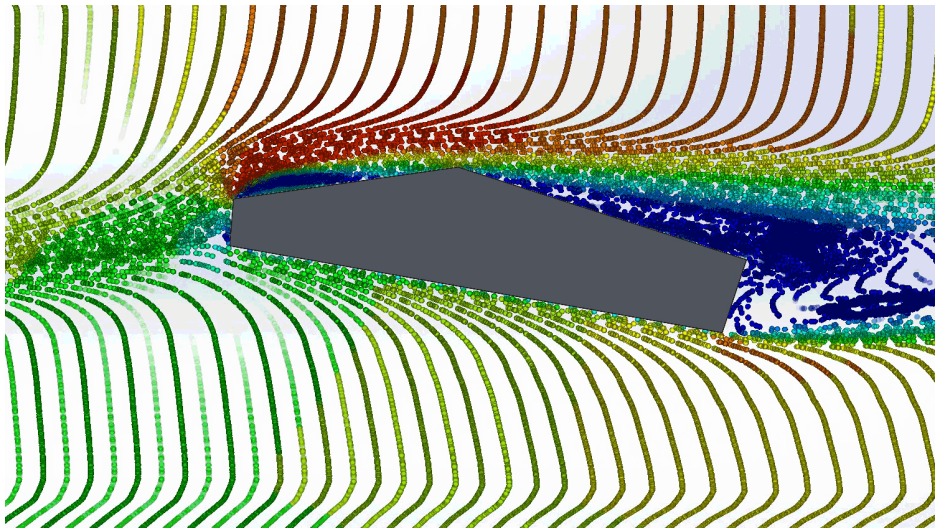
$$\min_x f(x)$$

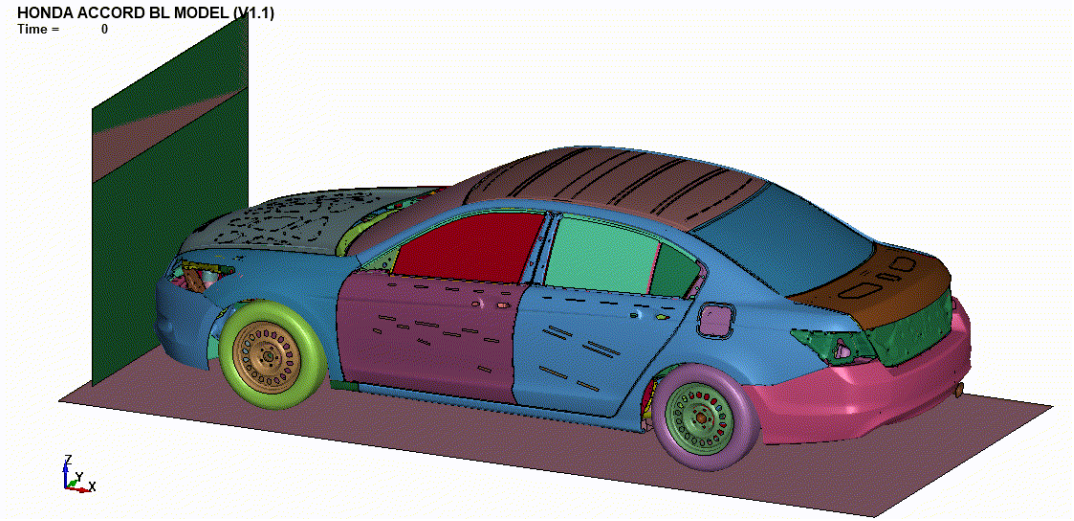
$$g_j(x) \sim ?$$

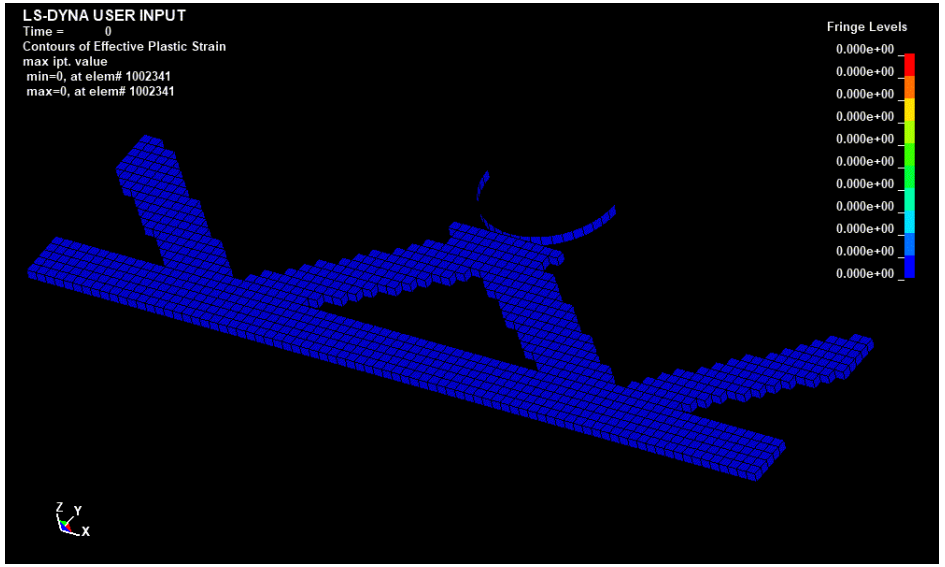
$$x_i \in [l_i, u_i] \subseteq [-\infty, +\infty]$$

$$j = 1 \ldots k$$

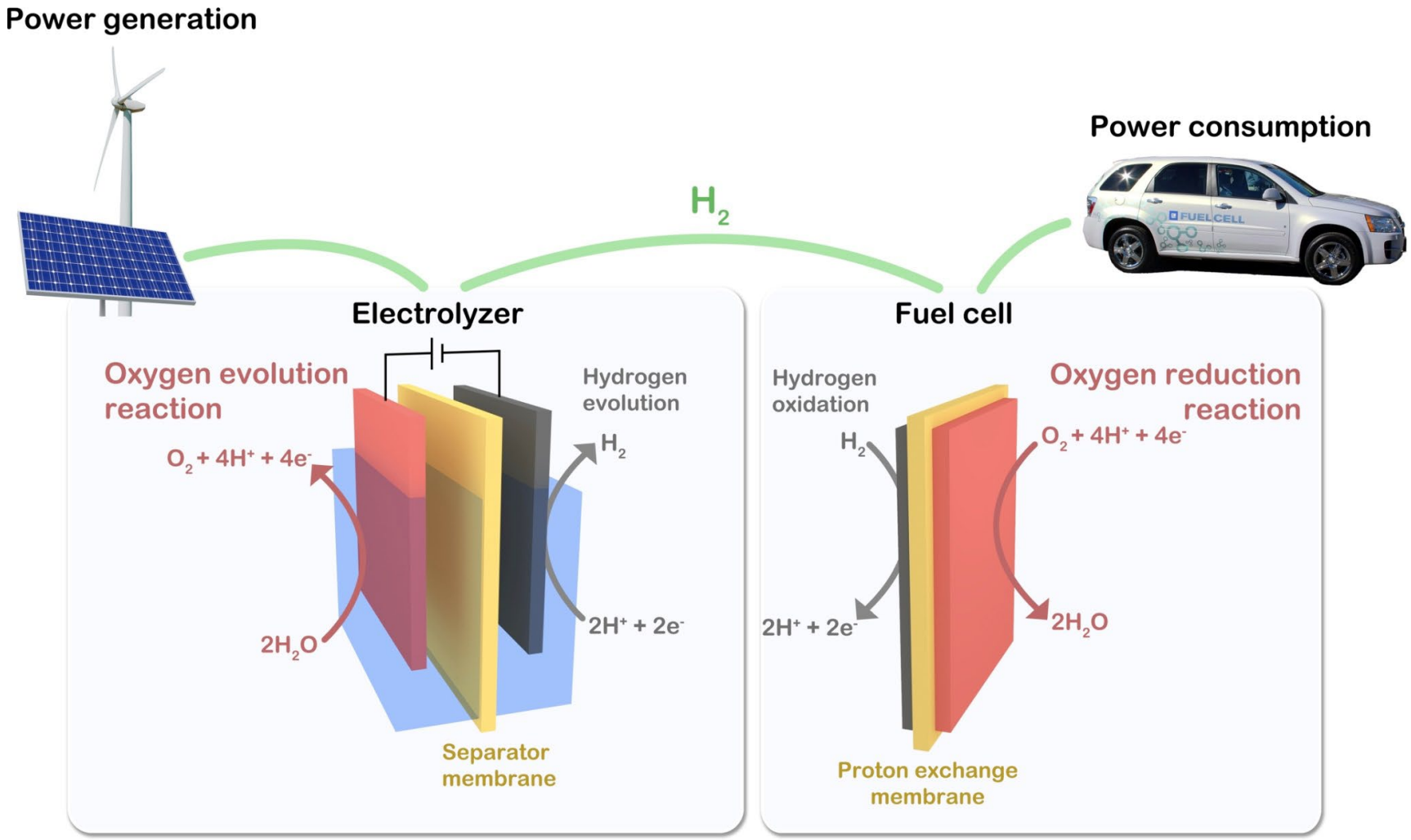
$$i = 1 \ldots n$$







A Real-World Application – The electrochemistry of green hydrogen



Black-box Optimization

Many algorithms work based only on function evaluations
→ **Metaheuristics**

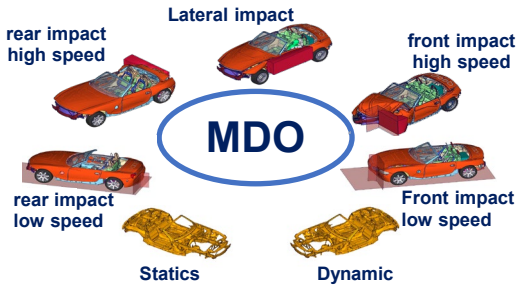
Metaheuristic: computational procedure that determines the most convenient way to find an optimal solution by iteratively trying to improve a candidate solution with respect to a given measure of quality.

→ **Which algorithm is best for black-box optimization?**

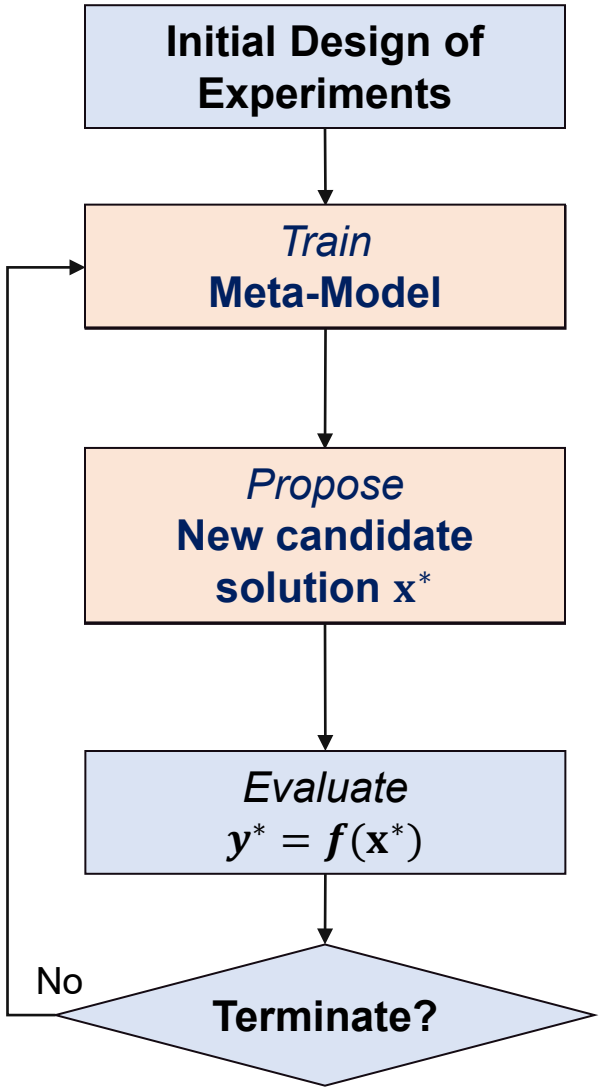
Model-Based Optimization

Global, black-box, expensive optimization:

$$\min_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x}), \qquad f:\mathcal{D} \rightarrow \mathbb{R}$$



E.g., crash in automotive



Bayesian optimization:
Gaussian Process Regression

How to pick $\mathbf{x}^*$?
$\mathbf{x}^* = argmax_{\mathbf{x}}(\mathbf{Acquisition\ fct})$

"Bayesian optimization does not refer to one particular algorithm but rather to a **philosophical approach to optimization grounded in Bayesian inference** from which an extensive family of algorithms have been derived"

(Roman Garnett, Bayesian Optimization Book, 2023)

Donald Jones et al.: Efficient global optimization of expensive black-box functions. *Journal of Global Optimization* **13**:455-492, 1998.

Jonas Močkus: On Bayesian methods for seeking the extremum, *Optimization Techniques IFIP Technical Conference*, 400-404, 1975.

Acquisition Functions

- The acquisition function is used to determine the next point for evaluation

  Balancing model prediction (mean) and uncertainty (variance)
  Evaluate where variance is large → ***exploration***
  Evaluate where mean is low (for minimization) → ***exploitation***

Predicted fct. value ↔ uncertainty of prediction

- Most widely used:

  Expected ***I***mprovement (EI) w.r.t. objective function value
  Probability of ***I***mprovement (PI): probability of making an improvement

- Requires mean AND variance of prediction

  Gaussian Process regression is ideally suited for this
  Provides variance analytically and unbiased

- New acquisition function: Moment-generating function (MGF)

  Unifies many existing ones for Bayesian optimization
  From exploration to exploitation → "cooling" parameter t

H. WANG, B. VAN STEIN, M. EMMERICH, TH. BÄCK: Moment-Generating Function Based Acquisition Function for Bayesian Optimization. *IEEE Conference on Systems, Man, and Cybernetics (IEEE SMC), Banff, AB, Oct. 5 – 8, 2017*.

Illustration: 1-d example with EI

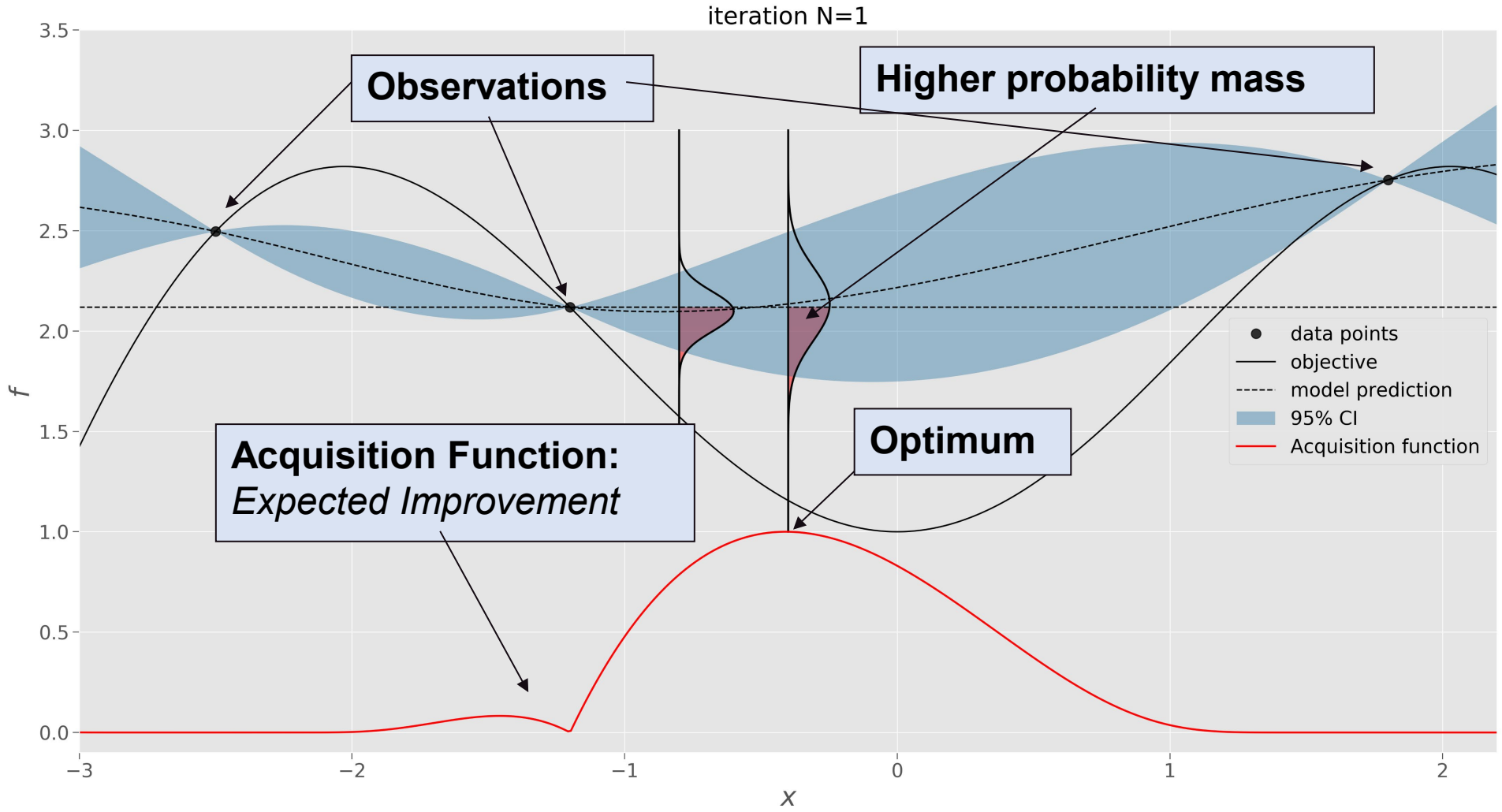


Illustration: 1-d example

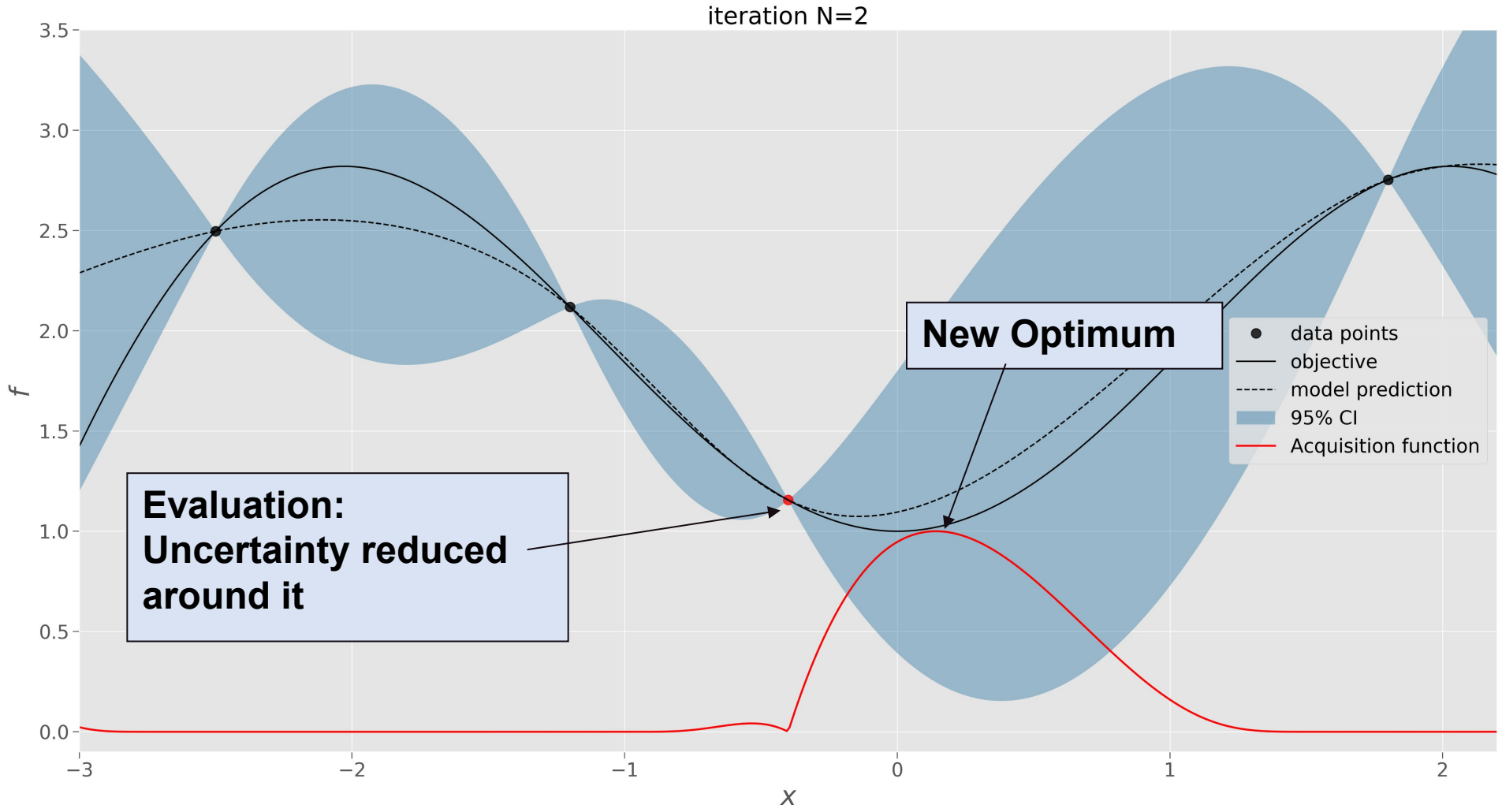


Illustration: 1-d example

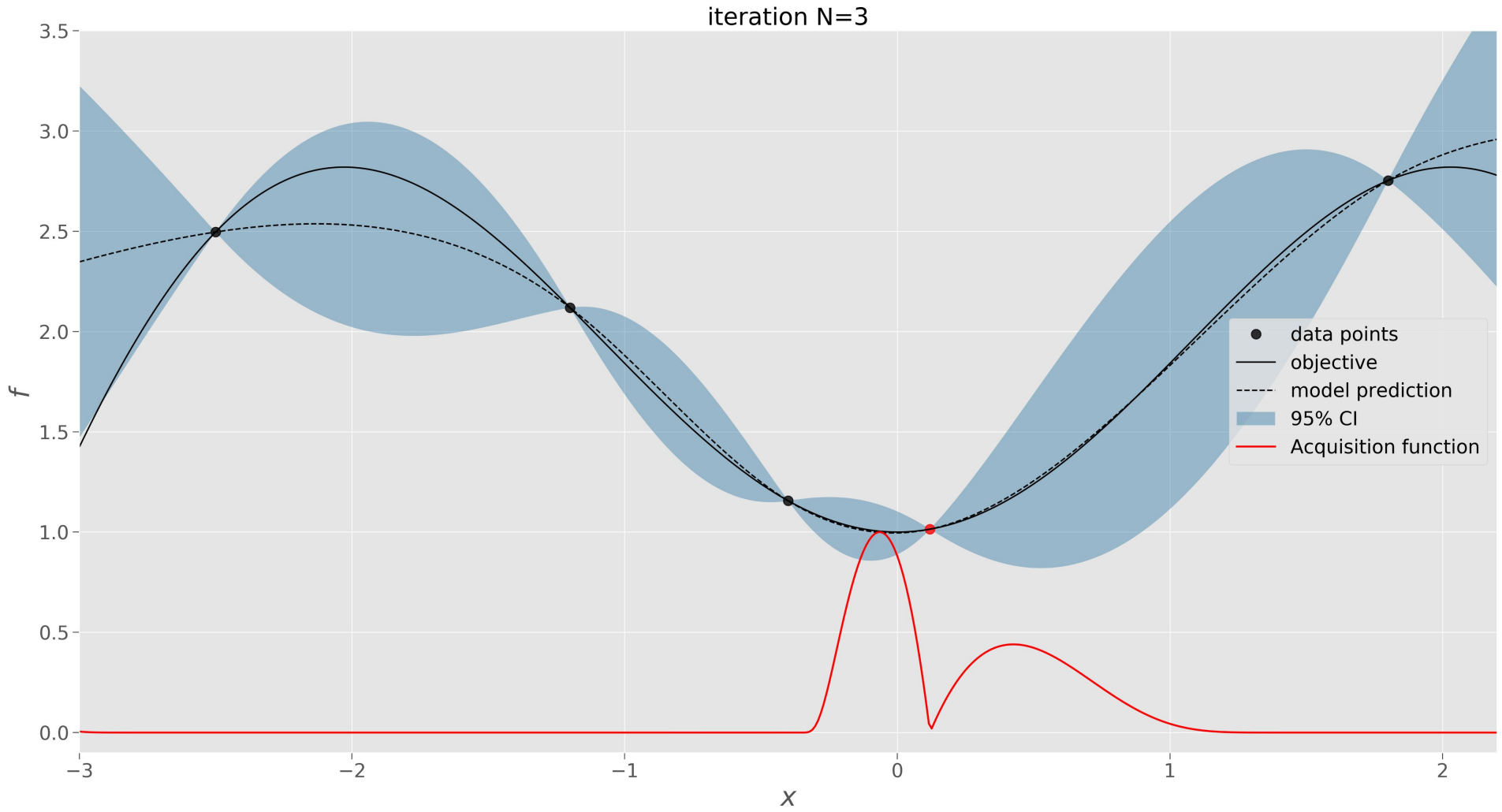


Illustration: 1-d example

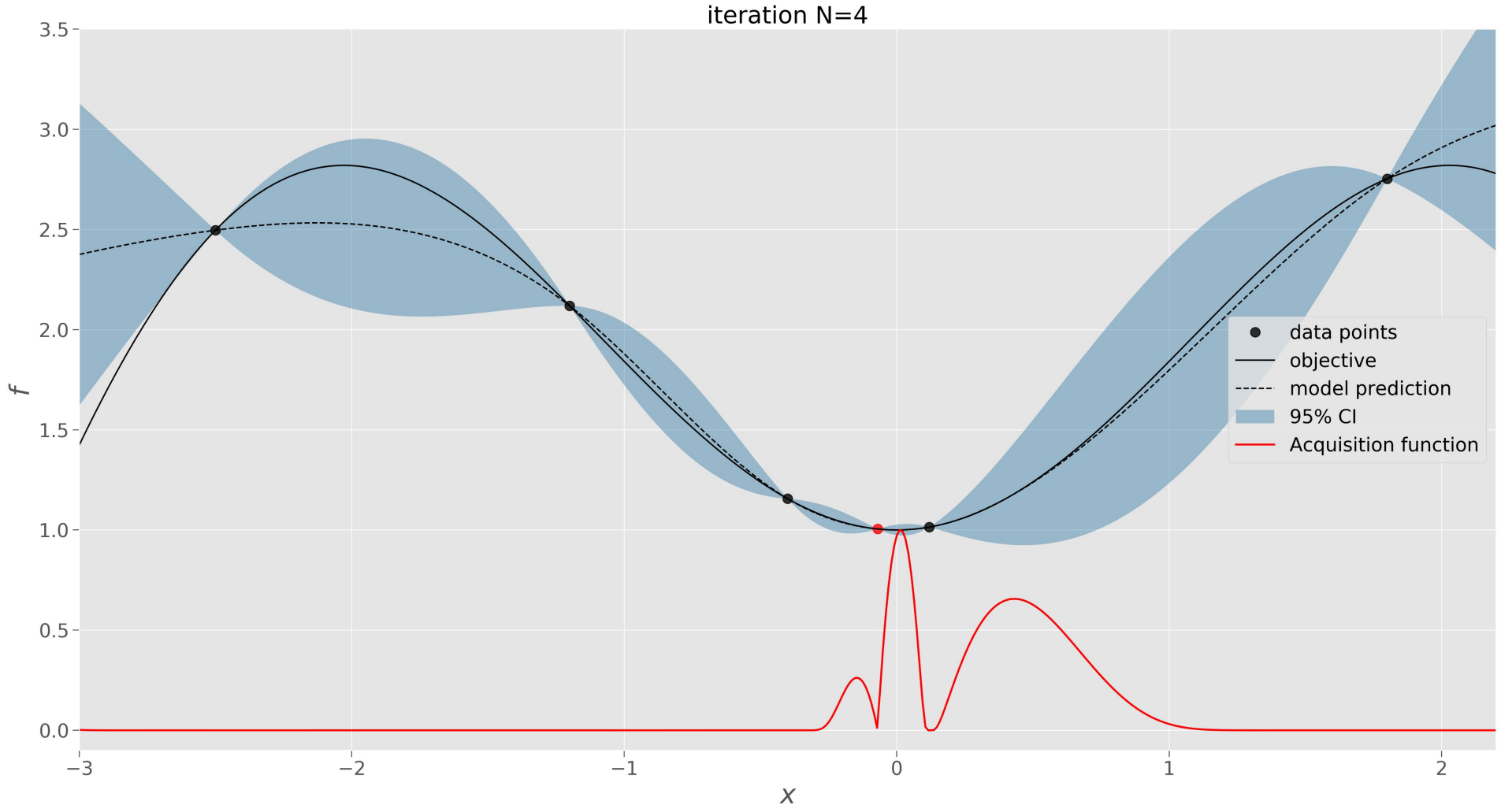
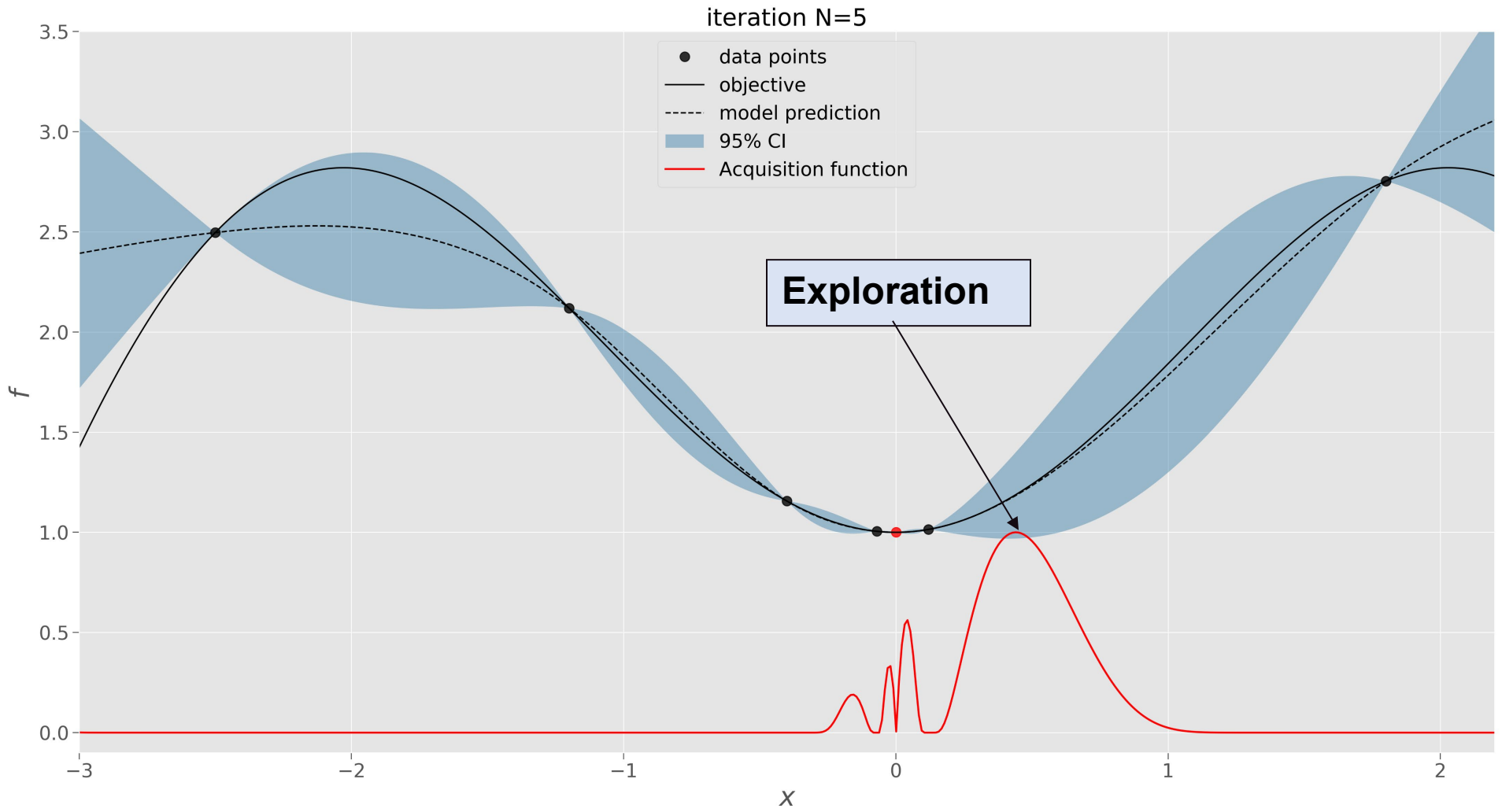


Illustration: 1-d example



Moment Generating Function (MGF)

- Moments of the Improvement

  Fully characterize its distribution
  Probability of Improvement $\mathbb{E}[I^0(\mathbf{x}) \mid I(\mathbf{x}) > 0]$ → the 'zero' order moment
  Expected Improvement: $\mathbb{E}[I(\mathbf{x})]$ → the first order moment
  Generalized Expected Improvement $\mathbb{E}[I^2(\mathbf{x})]$, $\mathbb{E}[I^3(\mathbf{x})]$, … to infinity

- All the infinite moments are combined in the ***moment-generating function***:

Add PI

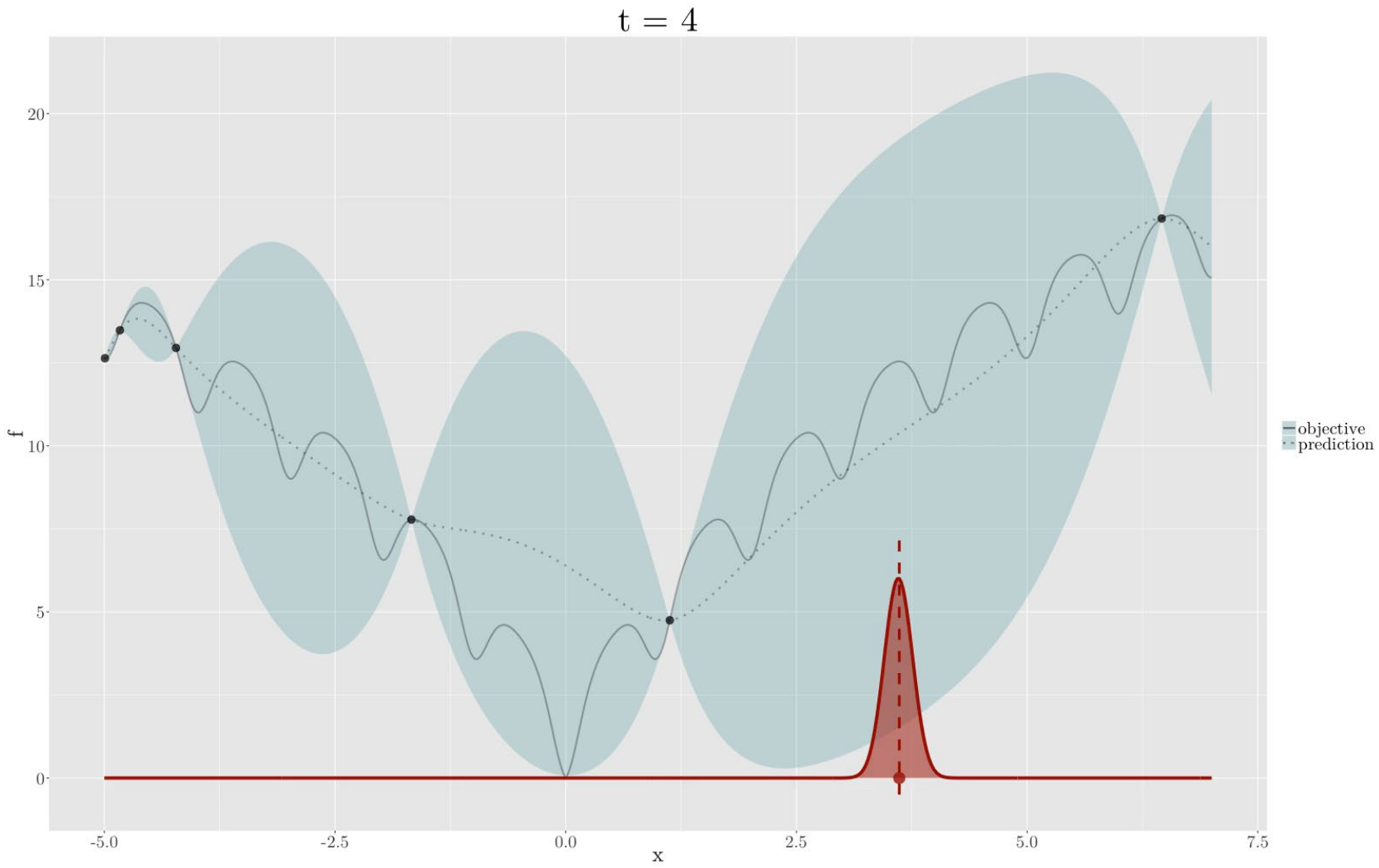
Normalize

$$\begin{aligned}\mathcal{M}(\mathbf{x};t) &= \mathrm{PI}(\mathbf{x}) + \frac{t}{e^t}\mathbb{E}[I(\mathbf{x})] + \frac{t^2}{2!e^t}\mathbb{E}[I^2(\mathbf{x})] + \frac{t^3}{3!e^t}\mathbb{E}[I^3(\mathbf{x})] + \cdots \\ &= \frac{M(\mathbf{x},t) - 1 + \mathrm{PI}(\mathbf{x})}{e^t} \\ &= \Phi\left(\frac{f_{\min} - m'(\mathbf{x})}{s(\mathbf{x})}\right)\exp\left((f_{\min} - m(\mathbf{x}) - 1)\,t + \frac{s^2(\mathbf{x})t^2}{2}\right)\end{aligned}$$

GPR prediction

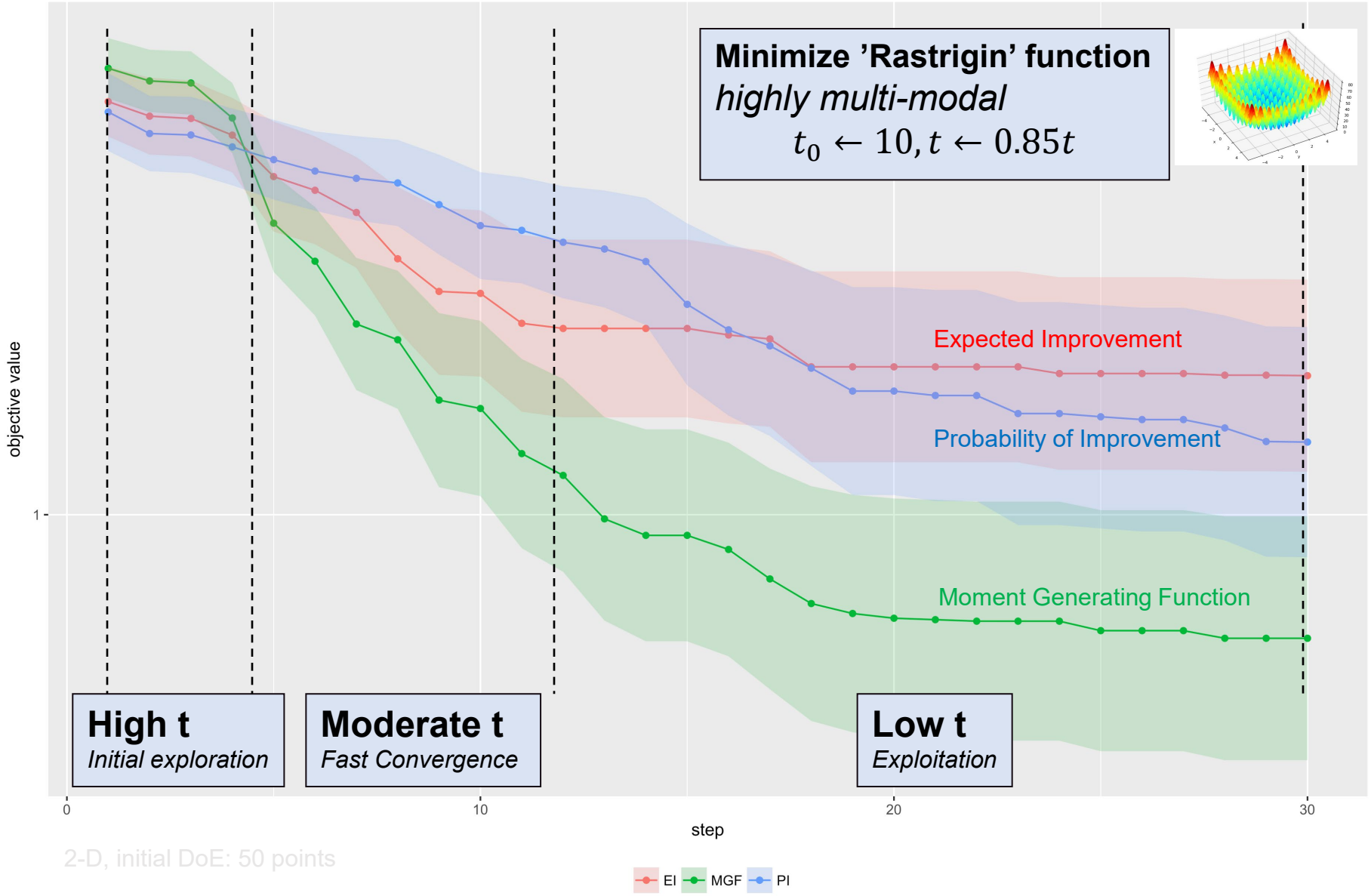
H. WANG, B. VAN STEIN, M. EMMERICH, TH. BÄCK: Moment-Generating Function Based Acquisition Function for Bayesian Optimization. *IEEE Conference on Systems, Man, and Cybernetics (IEEE SMC), Banff, AB, Oct. 5 – 8, 2017.*

The Effect of “Cooling”



Decrease t, the maximum of the acquisition function moves towards the known best point: exploration → exploitation

Cooling Schedule: Convergence



Self-Adaptation of Temperature

- ***Comparison-based method***

  generate q candidate points with different temperatures and choose the best

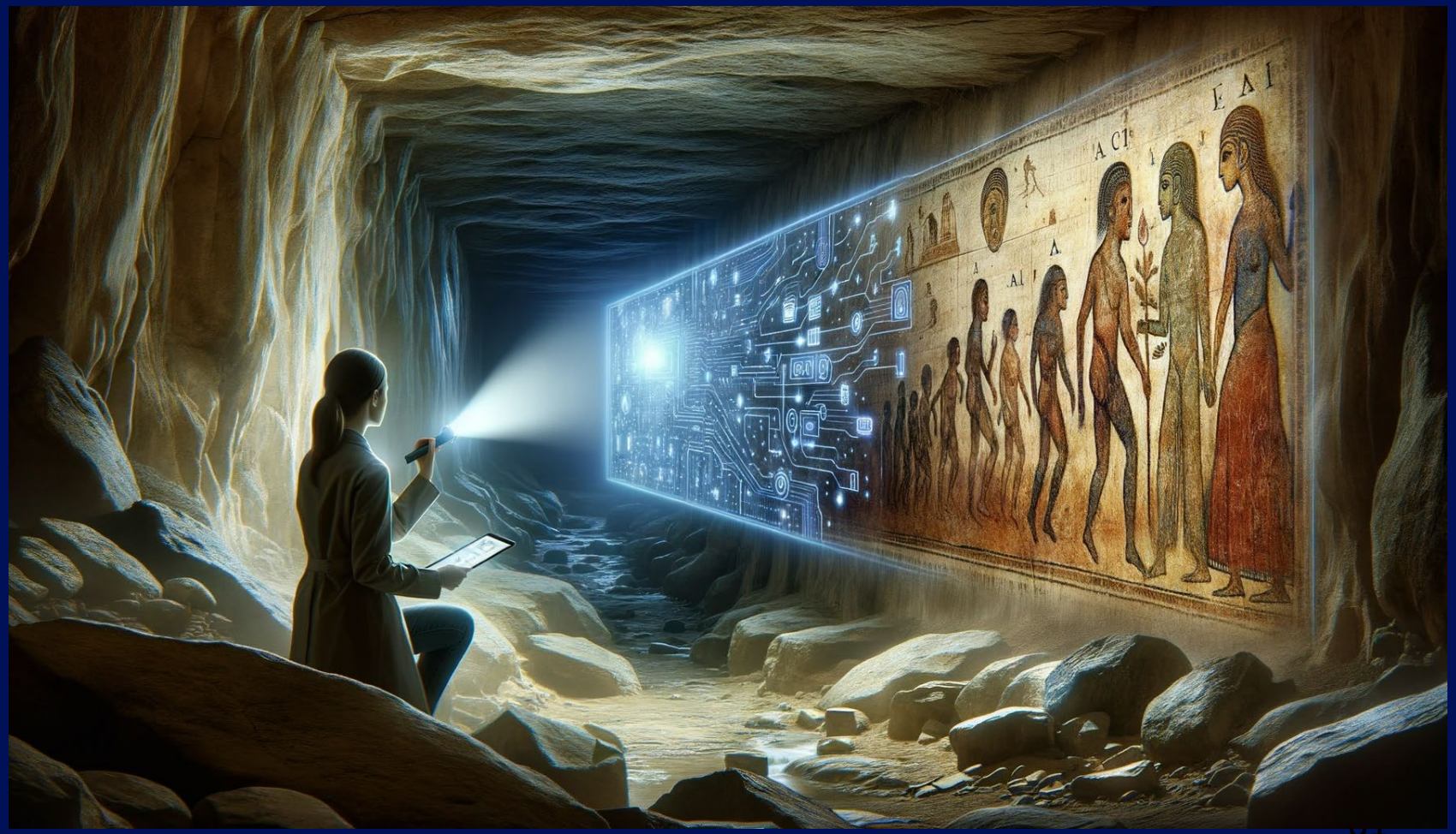
  Choose the temperature associated with the best objective value

Algorithm 1 Mixed Integer Parallel EGO for Hyper-parameters Optimization

procedure MIP-EGO($\mathcal{C}, F, q, t_0$)
  sample the initial data set (X, Y)
  evaluate $Y \leftarrow \{F(\mathbf{x}_1), F(\mathbf{x}_2), \ldots, F(\mathbf{x}_n)\}$
  train random forest: $\hat{F}, \hat{s}^2 \leftarrow (X, Y)$
  while stopping criteria are not fulfilled **do**
    for $i = 1 \rightarrow q$ **do** ▷ parallel for-loop
      $t \leftarrow t_0 \exp(\mathcal{N}(0, 1))$
      $\mathbf{x}' \leftarrow \arg\max_{\mathbf{x} \in \mathcal{C}} \mathcal{M}(\mathbf{x}; \hat{F}, \hat{s}^2, t)$
      compute $y' \leftarrow F(\mathbf{x}')$ ▷ Eq. (1)
    end for
    $X \leftarrow X \cup \{\mathbf{x}'\}$
    $Y \leftarrow Y \cup \{y'\}$
    re-training: $\hat{F}, \hat{s}^2 \leftarrow (X, Y)$
  end while
  return the best solution in X
end procedure

XAI - eXplainable Artificial Intelligence

Niki van Stein




Imagine you are the captain of a ship


EARLY WARNING ALERT
But WHY?

eXplainable AI

- Key role in enabling AI in industry.
- Required by legislation (EU AI act).
- Many open challenges in the field:

How to benchmark XAI methods?
How to apply XAI in a meaningful way to timeseries?
How to deal with complex non-linear interactions?

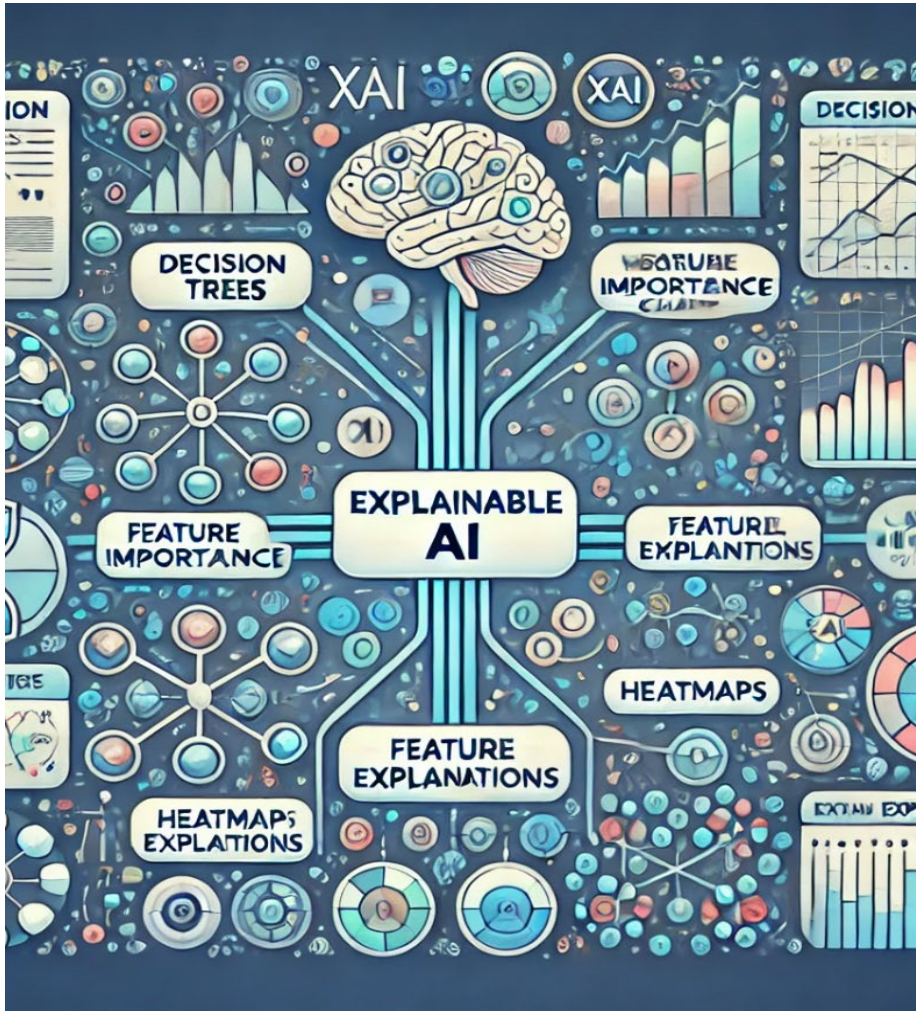


Typical use-cases of XAI

- Provide **explanations** to an end-user (domain expert).
- Debug AI models (see why things go wrong).
- Identify and mitigate **unwanted** bias in training data or model preferences.
- Understand **feature importance and interactions** for a given problem.

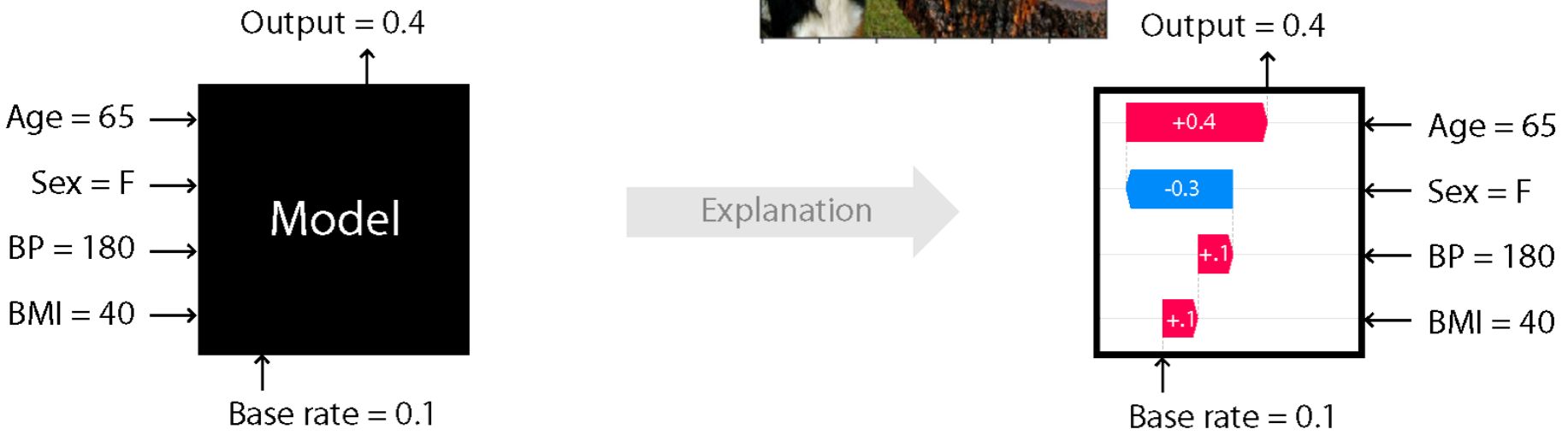
What are explanations?

- Visual – (mostly input feature attributions)
- As a decision tree (or path in the tree)
- As a ranking of features (in order of importance)
- Textual explanations (e.g. generated by an LLM)
- Almost always require domain knowledge **and** AI knowledge to understand
- Different users require different explanations
- Level of detail versus correctness



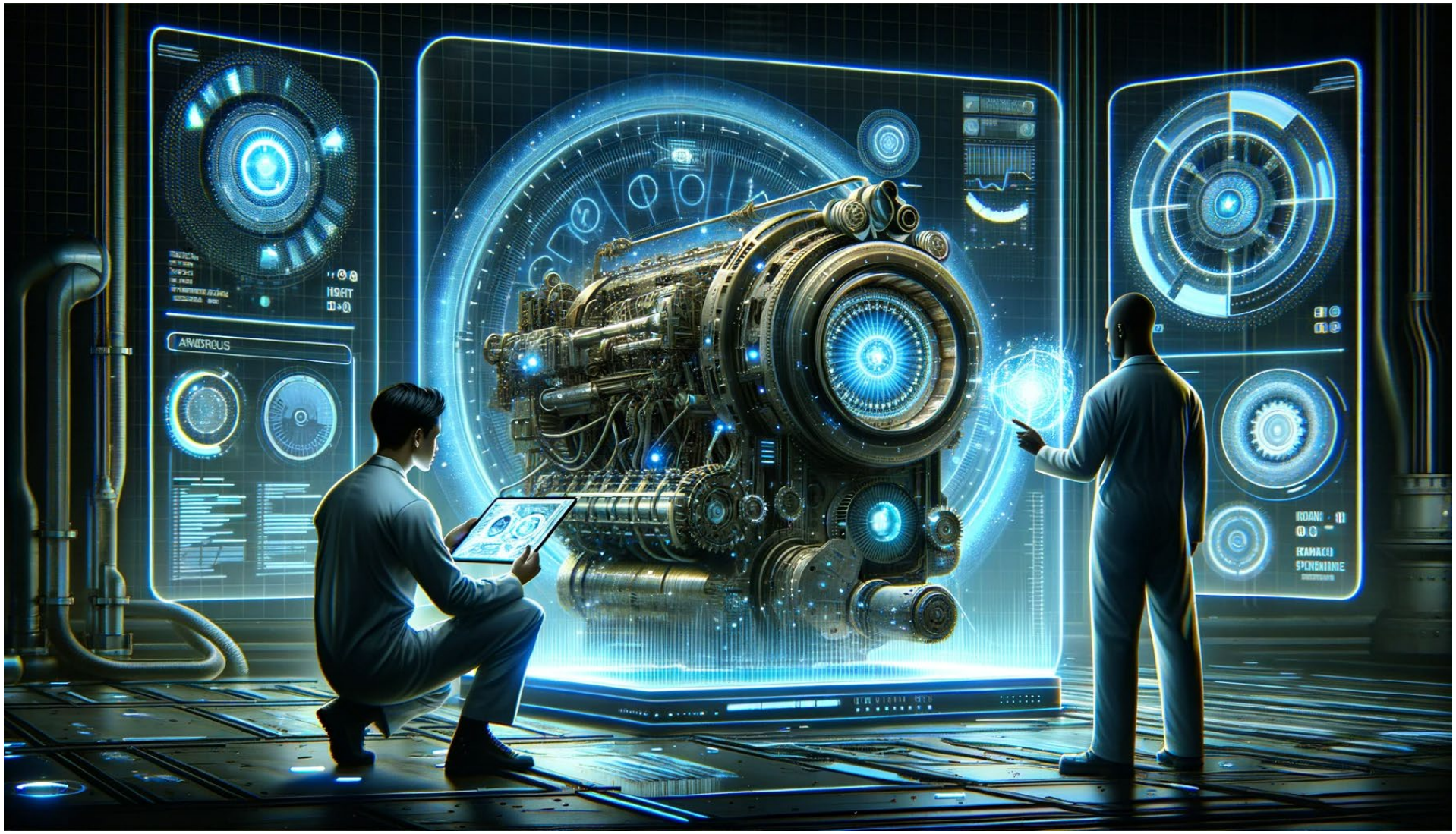
Local model-agnostic XAI methods



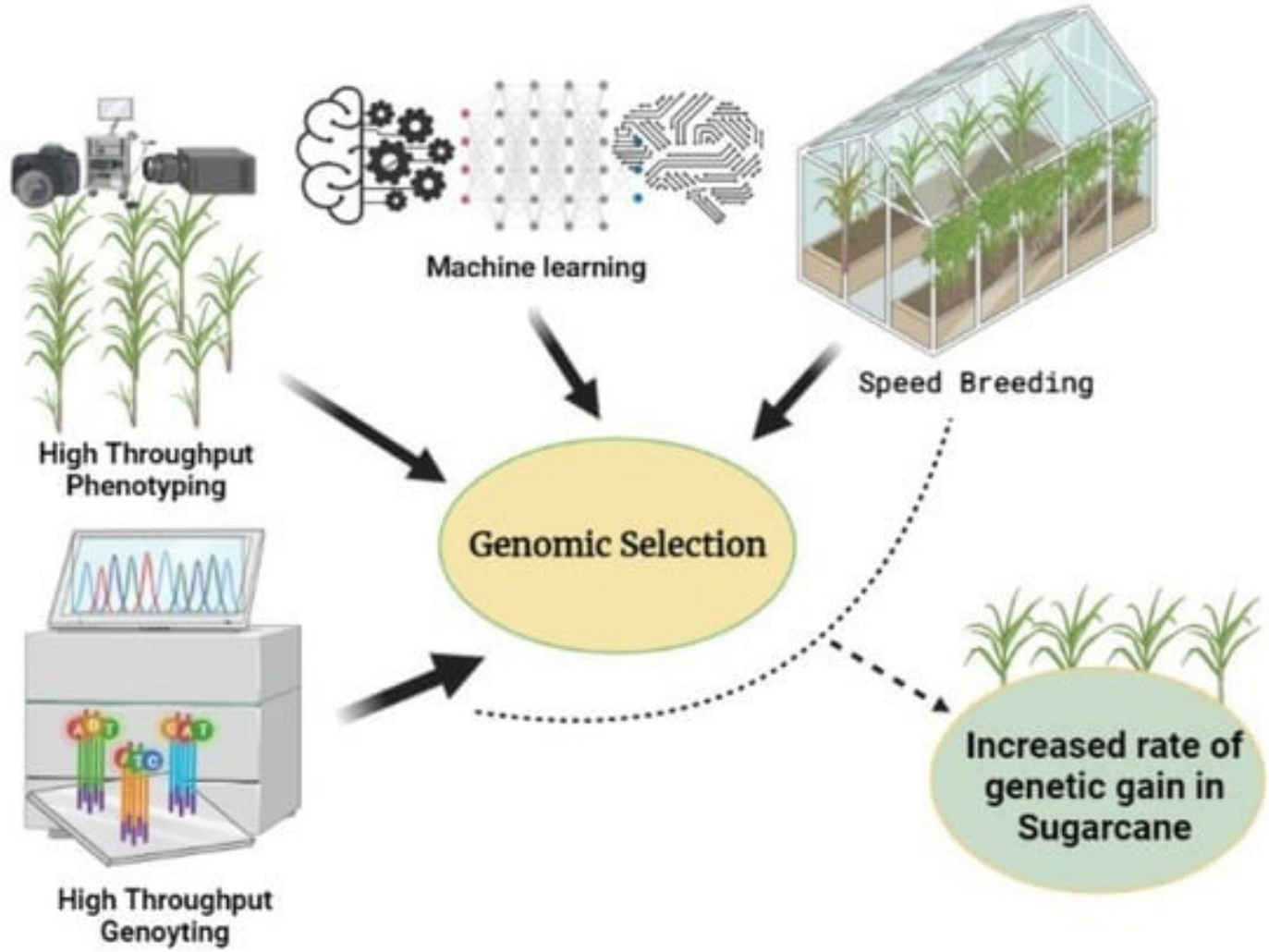


Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. "" Why should i trust you?" Explaining the predictions of any classifier." *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016.

Explainable AI – Use Cases



Explaining Deep Learning models for Genomics

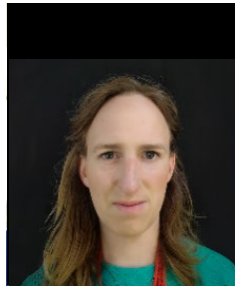


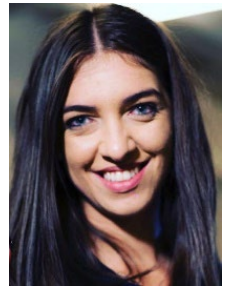
Global Sensitivity Analysis Reporting Tool



https://github.com/nikivanstein/GSAreport

- Application containing **many GSA methods**
- **Dimension and sample-size dependent**, it selects the appropriate methods to perform global sensitivity analysis.
- Works with **pre-generated datasets** or **generates the sample** itself based on problem specifications
- Developed within the **Natural Computing** Cluster - LIACS


Niki van Stein


Elena Raponi


Thomas Bäck

Naco Natural Computing Computer Science Leiden University

Inspiration

Original Problem

Single Nucleotide Polymorphism (SNP) Genotyping



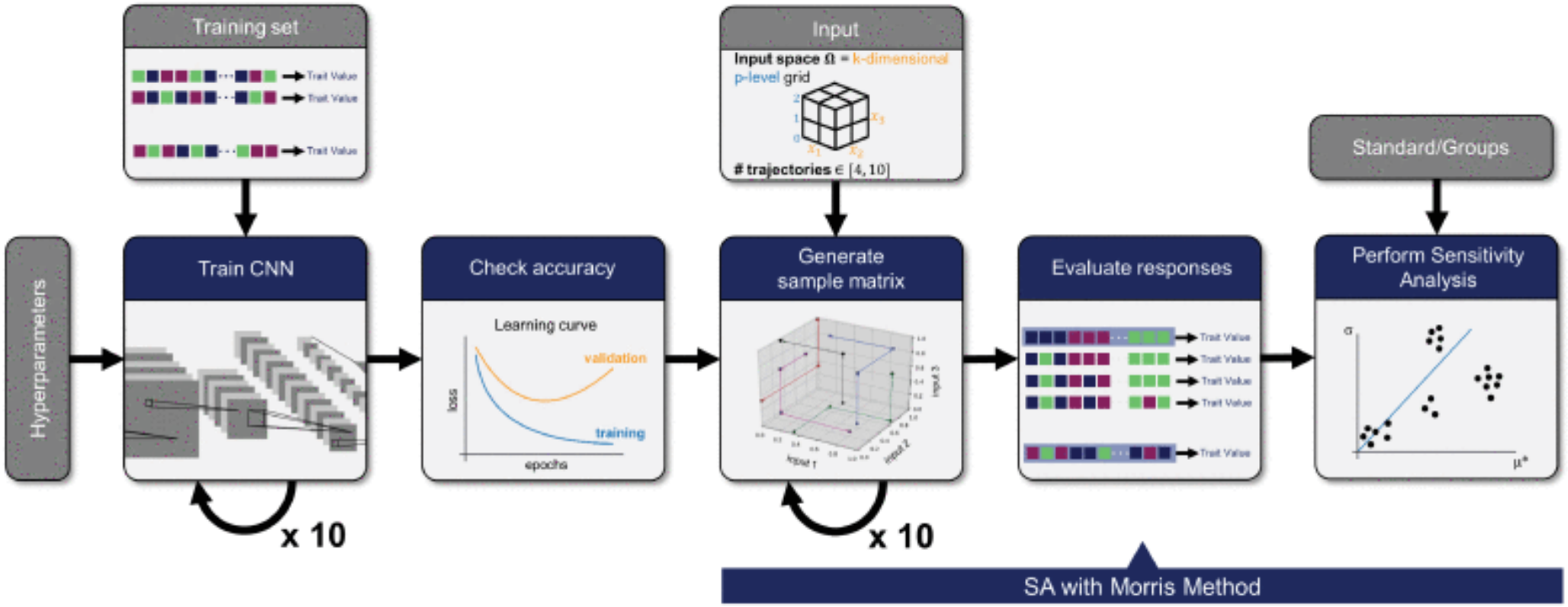
- Plant breeding → Process to develop new plant varieties trough mutation and recombination to improve the genetic potential of plants.
- SNP contributions can be strong/moderate, independent of each other, or influence and modulate the effect of each other.
- We want to evaluate how SNP variations impact a target

Sample 1	Sample 2	Sample 3	Sample 4
CCCTAAACCCTAAAA	CCCTAAACCCTAAAC	CCCTAAACCCTAAAA	CCCTAAACCCTAAAA
CCTAAACCCTAAACC	CCTAAACCCTAAACC	CCTAAACCCTAAACC	CCTAAACCCTGAACC
TCTGAATCCTTAATC	TGTGAATCCTTAATC	TGTGAATCCTTAATC	TCTGAATCCTTAATC
CCTAAATCCCTAAAT	CCTAAATCCCTAAAT	CCTAAATCCCTAAAT	CCTAAATCCCTAAAT
CTTTAACTCTTACATC	CTTTAACTCCTACATC	CTTTAAATCCTACAT	CTTTAAATCCTACAT
CATGAATCCCTAAAT	CATGAATCCCTAAAT	CCATGAATCCCTAAA	CCATGAATCCCTAAA
ACGC	ACGC	TACGC	TACGC

→ Trait value

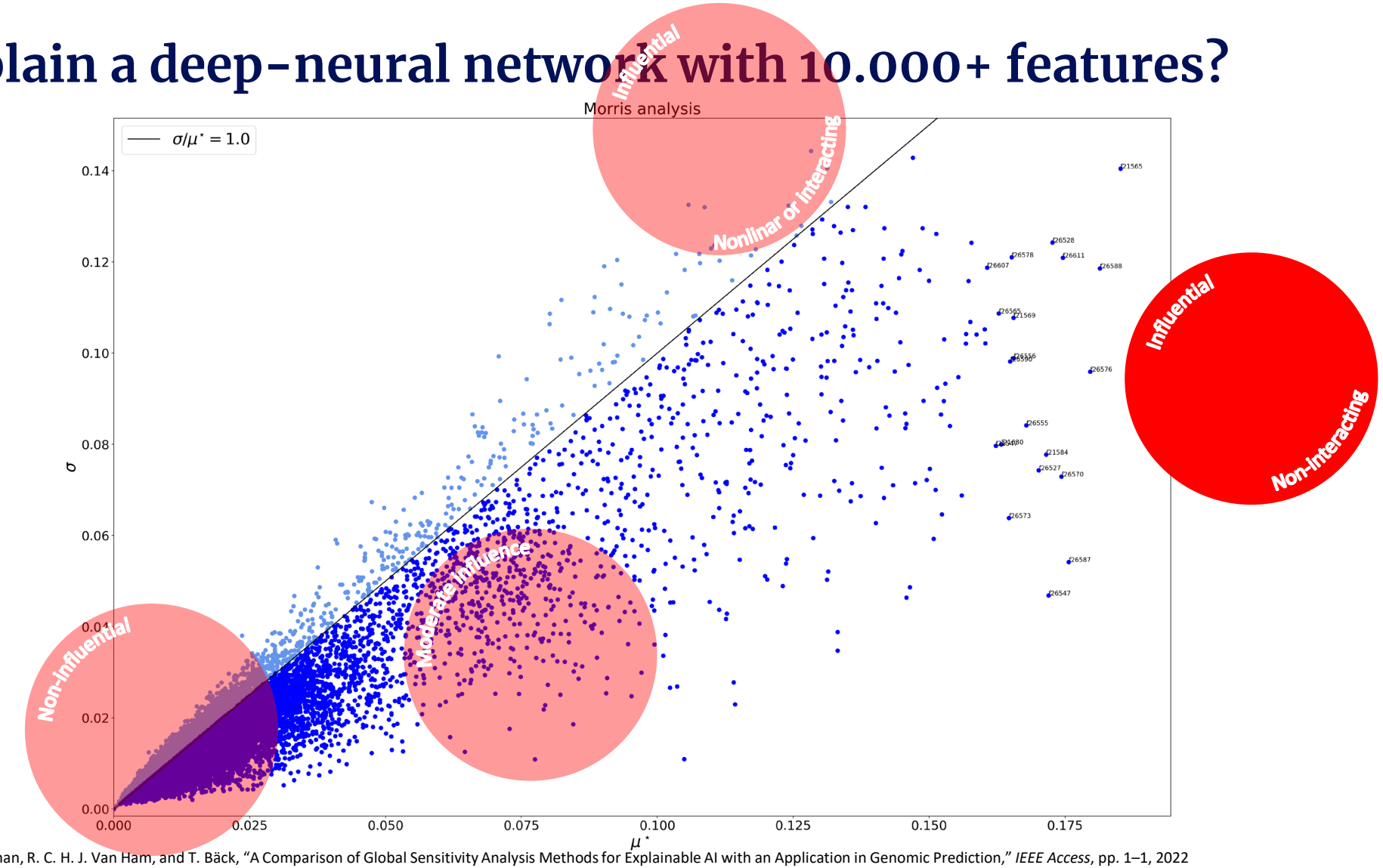
KeyGene

How to explain a deep-neural network with 10.000+ features?



A comparison of global sensitivity analysis methods for explainable AI
N van Stein, E Raponi, Z Sadeghi, N Bouman, RCHJ Van Ham, T Bäck 2022 Journal IEEE Access

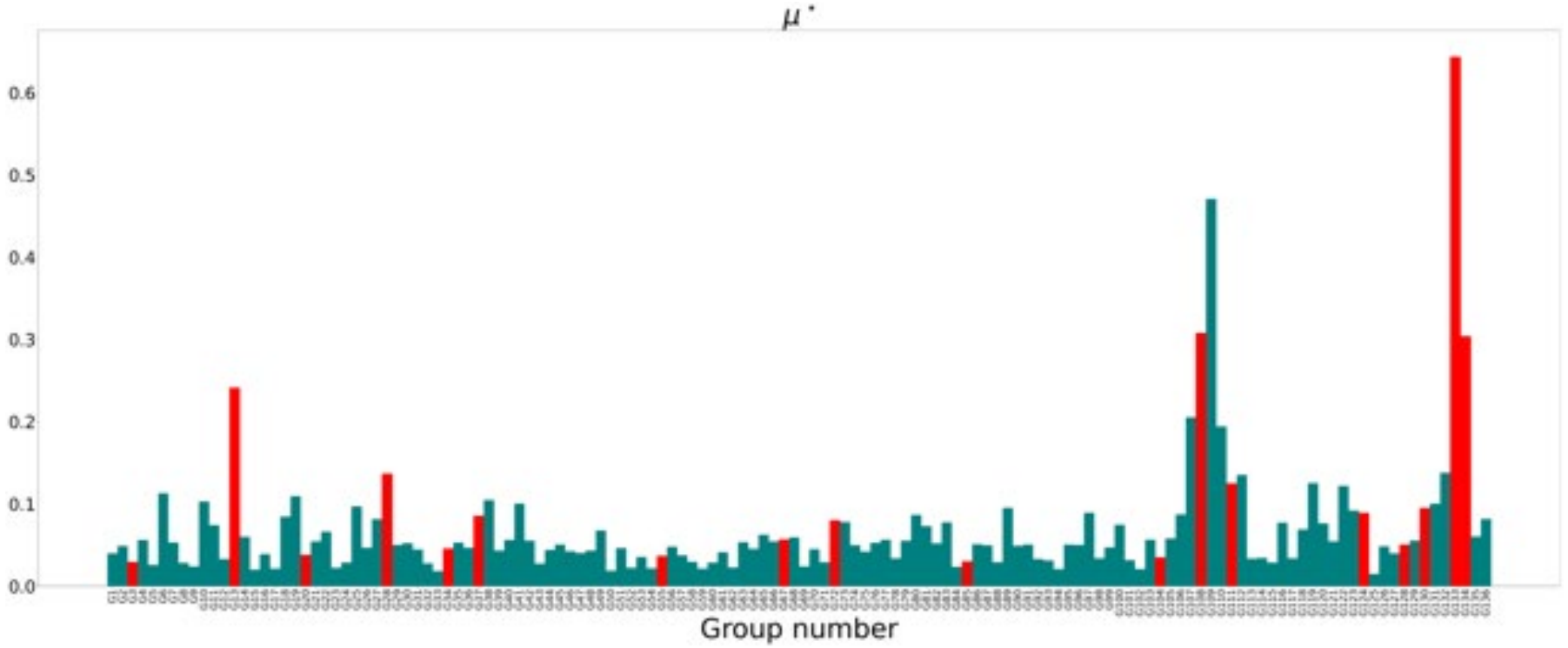
How to explain a deep-neural network with 10.000+ features?



N. Van Stein, E. Raponi, Z. Sadeghi, N. Bouman, R. C. H. J. Van Ham, and T. Bäck, "A Comparison of Global Sensitivity Analysis Methods for Explainable AI with an Application in Genomic Prediction," *IEEE Access*, pp. 1–1, 2022

Grouped Sensitivity Analysis

136 groups of SNPS. each containing 200 features.
Red bars correspond to groups including ground truth important features.



Application Case from Computational Mechanics

Drawability Assessment of Deep-Drawn Components

- In the early-stage development of sheet metal parts, key design properties of new structures must be specified.
- This affects drawability.
- GSA can provide insight into the impact of various changes in drawing configurations on drawability.

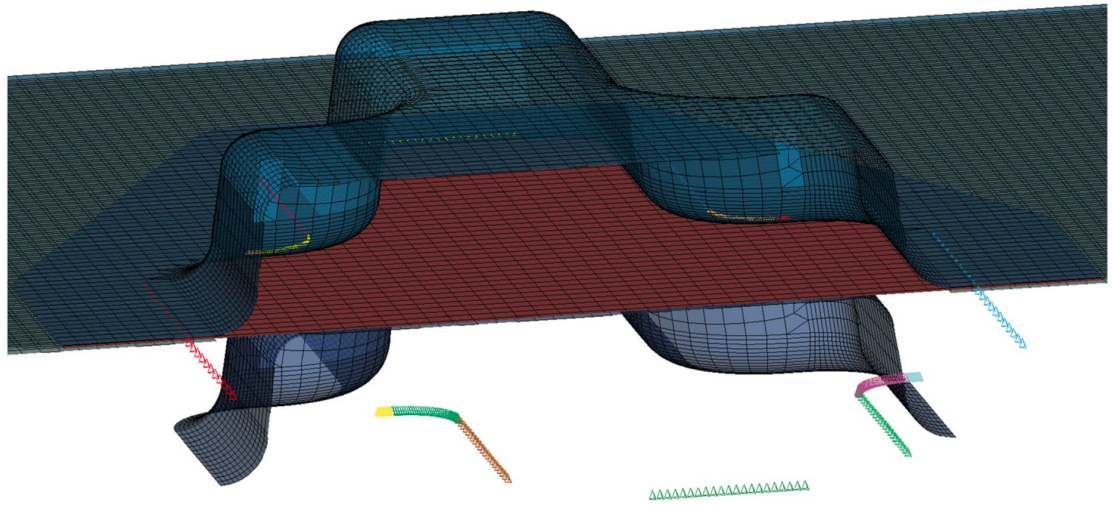
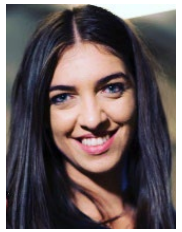


Figure: Arrangement of geometries in a sheet metal forming procedure.



Elena Raponi

T. Lehrer, A. Kaps, I. Lepenies, E. Raponi, M. Wagner, and F. Duddeck, "Complementing Drawability Assessment of Deep-Drawn Components With Surrogate-Based Global Sensitivity Analysis," *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering*, vol. 10, no. 3, p. 031204, Sep. 2024, doi: 10.1115/1.4065143.

Application Case from Computational Mechanics

Problem formulation

Symbol	Parameter range	Parameter
t	(0.8–2.0) mm	Sheet thickness
r	(0.8–2.5)	Lankford coefficient
σ_y	(140.0–180.0) MPa	Yield strength of the blank
f_{blkh}	(130.0–190.0) kN	Blankholder force
μ_{d}	(0.05–0.12)	Dynamic coefficient of friction

$$f_{\text{distances}} = \begin{cases} f_{\text{n-d}}, & \text{if } f_{\text{n-d}} > f_{\text{dt}} \\ f_{\text{d}}, & \text{otherwise} \end{cases}$$

f_{n-d} = non-drawable configurations
f_d = drawable configurations
f_{dt} = drawability threshold

Results

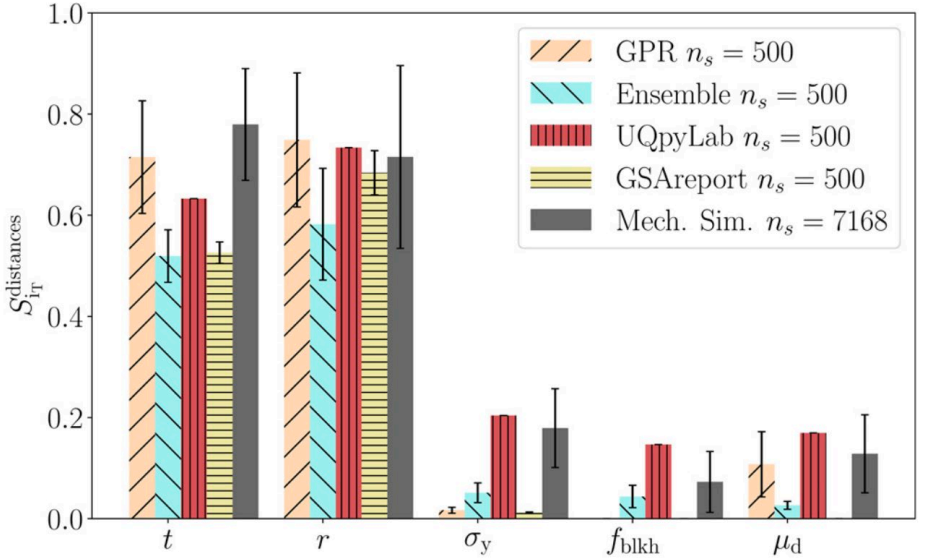
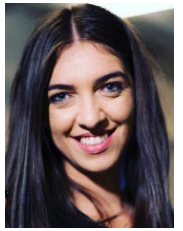


Figure: SI for the ensemble model, GPR surrogate, UQpyLab, and GSAreport. The SI based on the mechanical simulation is given as reference. Results are shown on the **distances objective function** for every parameter.



Elena Raponi

T. Lehrer, A. Kaps, I. Lepenies, E. Raponi, M. Wagner, and F. Duddeck, "Complementing Drawability Assessment of Deep-Drawn Components With Surrogate-Based Global Sensitivity Analysis," *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering*, vol. 10, no. 3, p. 031204, Sep. 2024, doi: 10.1115/1.4065143.

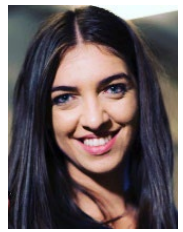
A comparison of global sensitivity analysis methods for explainable AI

N van Stein, E Raponi, Z Sadeghi, N Bouman, RCHJ Van Ham, T Bäck
2022 Journal IEEE Access

GitHub

- Pypi package **gsareport**
- Github: https://github.com/nikivanstein/GSAreport


Niki van Stein


Elena Raponi


Thomas Bäck

Naco Natural Computing Computer Science Leiden University

Open challenges

https://nikivanstein.nl



How to evaluate different explainers fairly?



How to explain the results of XAI of time-series or other complex data?



When problems are inherently complex to understand, so are the explanations of their solutions, can we do better?

Bayesian Optimizationn

Elena Raponi

BOapp Group

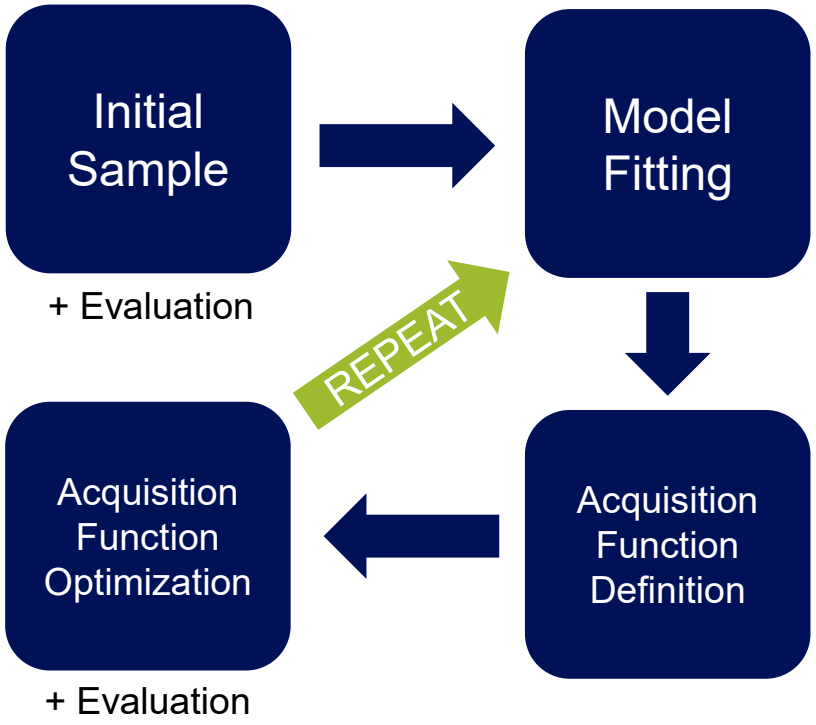


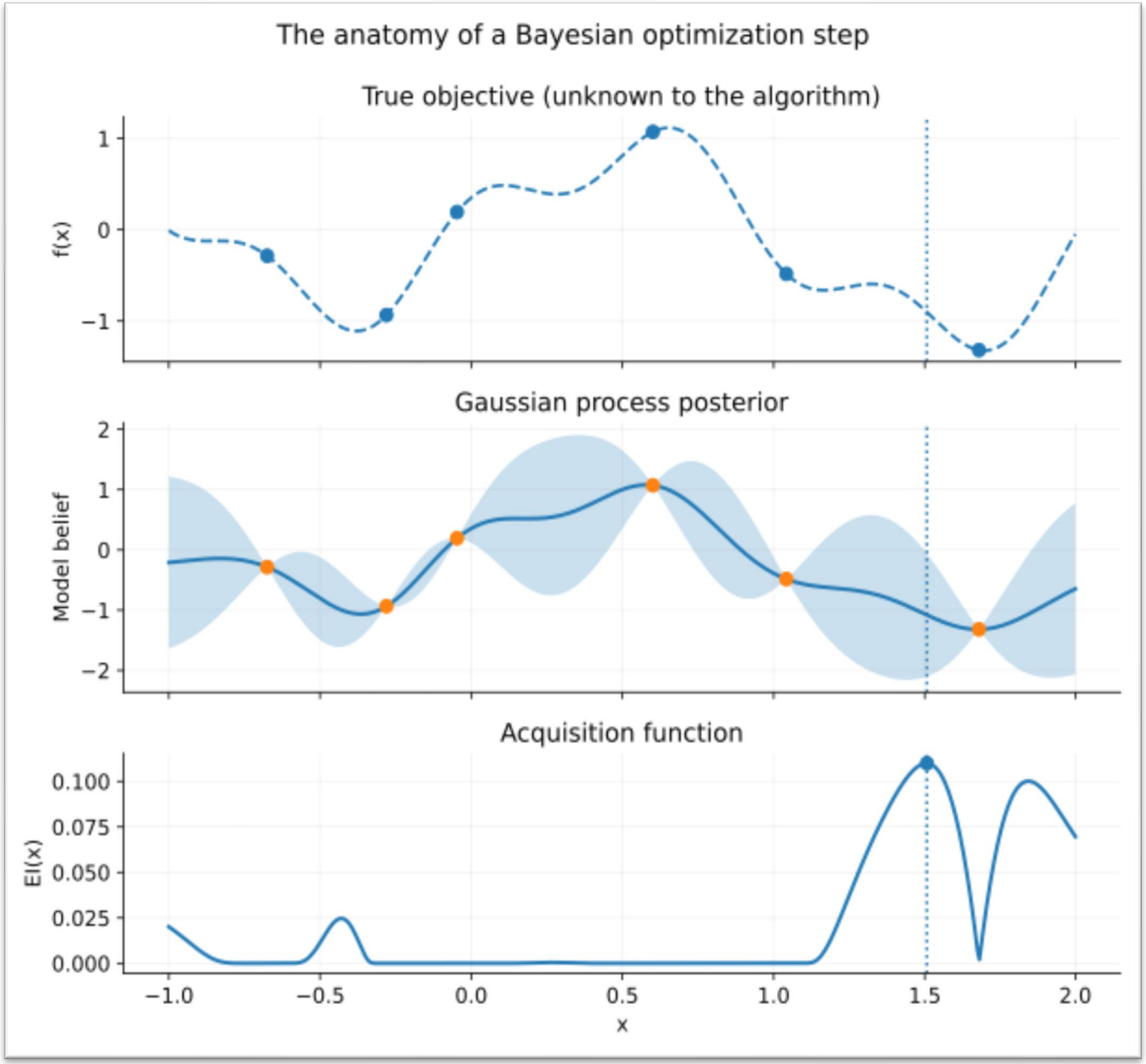
- Bayesian Optimization for **high-dimensional**, **constrained**, and **mixed-variable** problems
- **Automatic Bayesian Optimization** algorithm design
- **Benchmarking design** and evaluation of optimization algorithms
- **Application-driven optimization** inspired by real-world challenges
- Black-box optimization for **engineering design**
- **Problem-similarity** detection with small data

Bayesian Optimization

“Bayesian optimization does not refer to one particular algorithm but rather to a **philosophical approach to optimization grounded in Bayesian inference** from which an extensive family of algorithms have been derived”

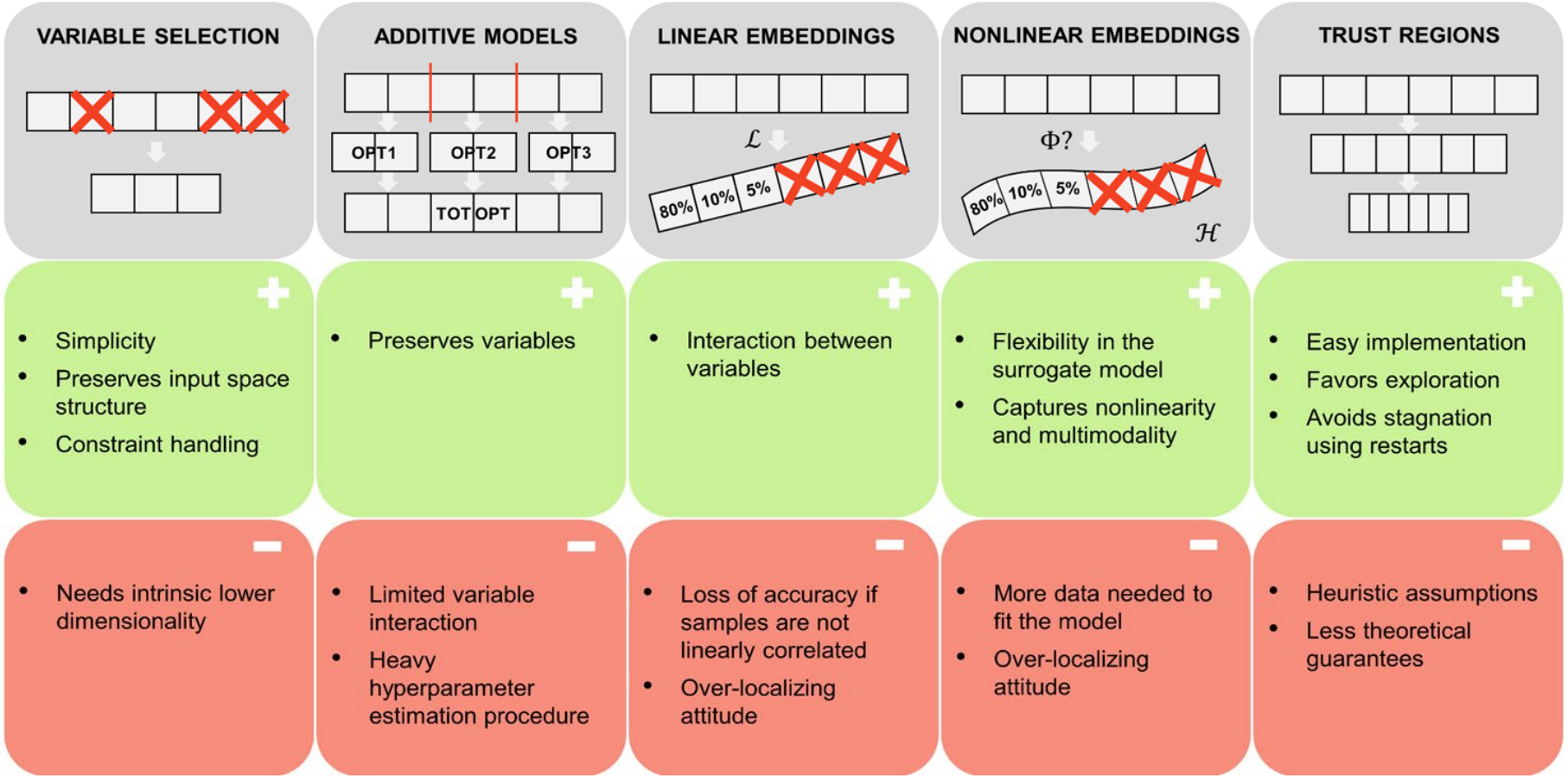
(Roman Garnett, Bayesian Optimization Book, 2023)





High-Dimensional Bayesian Optimization

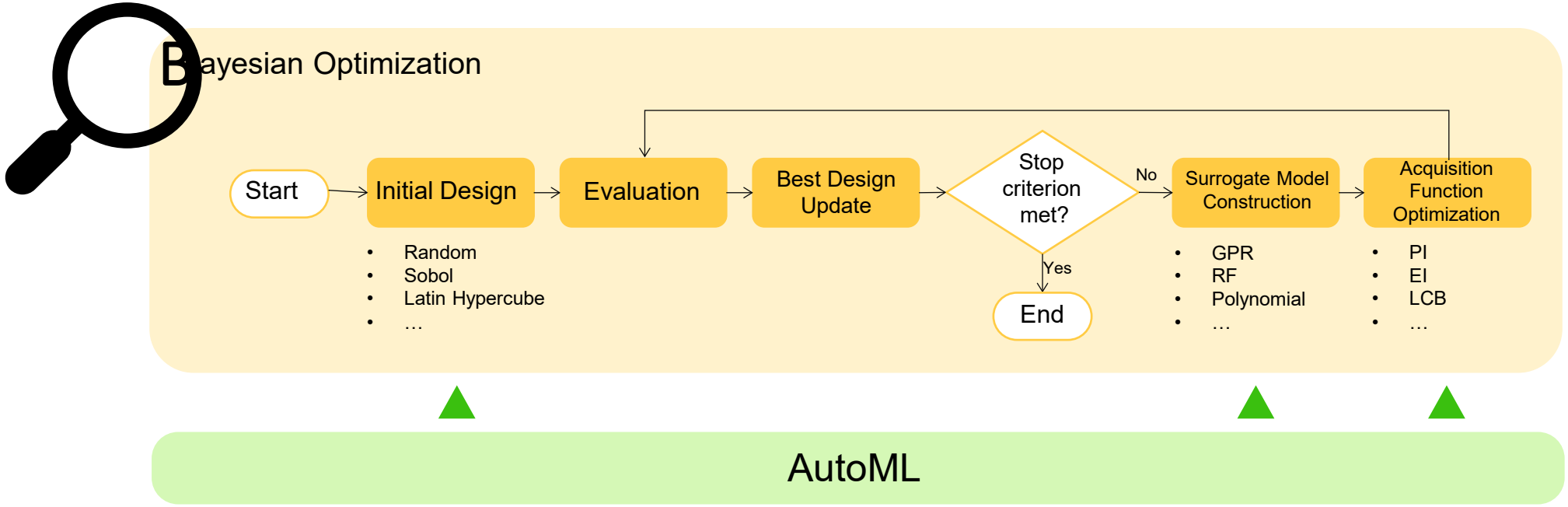
Bayesian optimization excels when evaluations are expensive, but **in high dimensions it faces the curse of dimensionality**: the surrogate model requires many more observations to accurately represent the objective, and optimizing the acquisition function also becomes much more challenging.



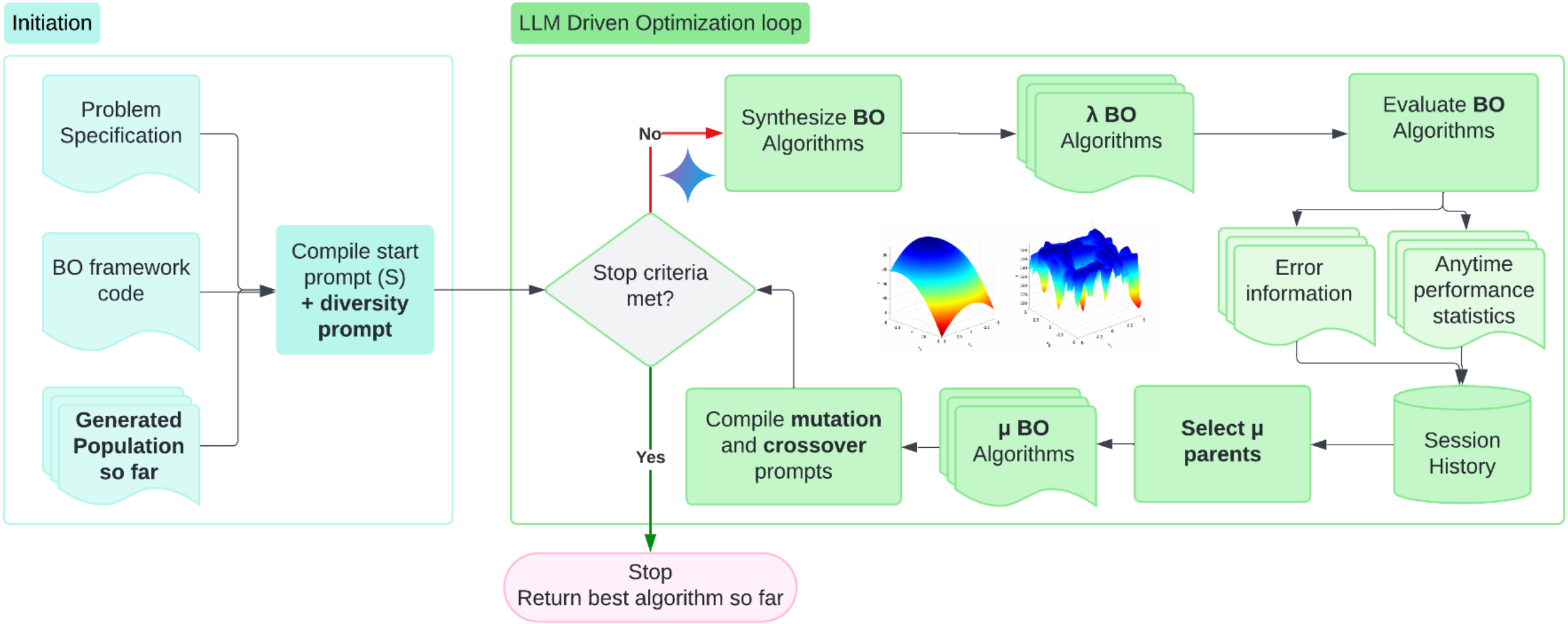
M. L. Santoni, E. Raponi, R. D. Leone, and C. Doerr, "Comparison of High-Dimensional Bayesian Optimization Algorithms on BBOB," *ACM Trans. Evol. Learn. Optim.*, vol. 4, no. 3, p. 17:1-17:33, Jul. 2024, doi: 10.1145/3670683.

Modular Bayesian Optimization

- BO pipeline itself → highly configurable, many different design choices
- Our understanding of how to select suitable components for a problem at hand is very limited
- We worked on the selection and definition of the acquisition function



LLaMEA-BO: LLM-Driven Automated Generation of BO Algorithms



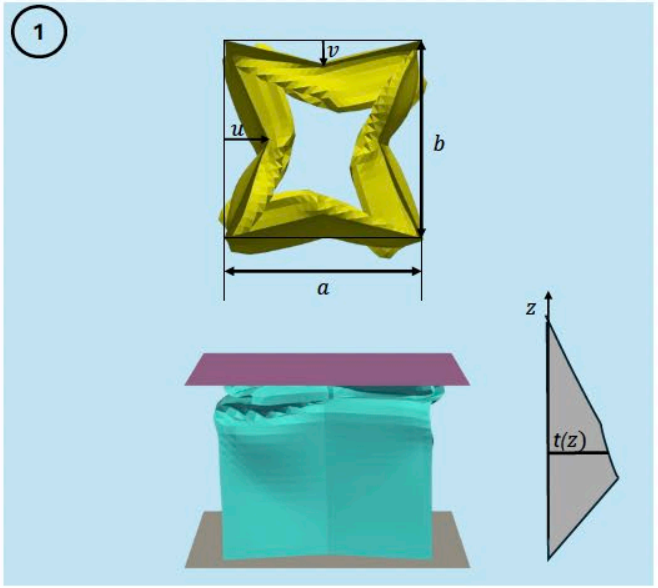
MechBench

Design of benchmark problems derived from structural mechanics applications.

Links

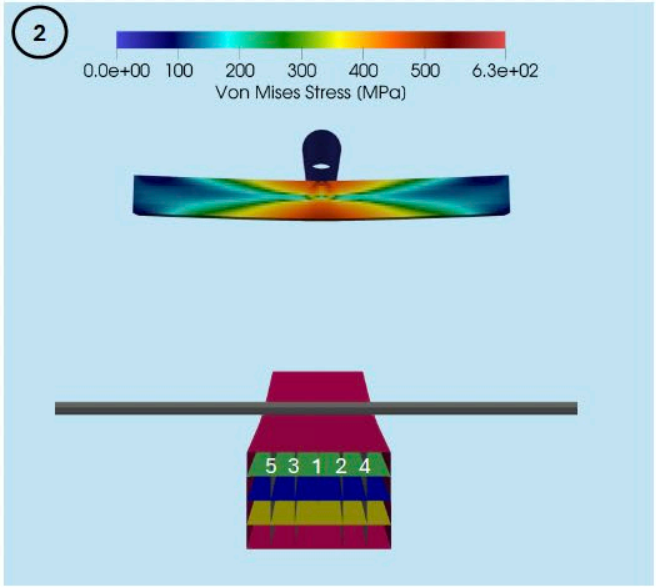
GitHub arXiv

Physical Models/Cases



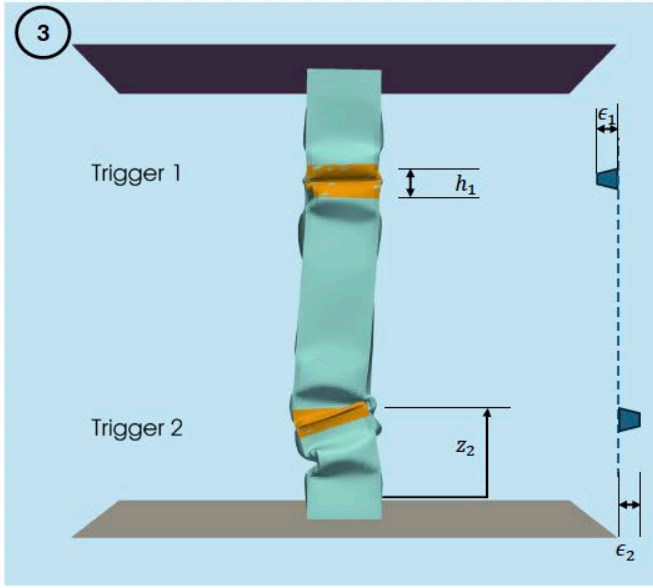
Star-box

Variables: the **star-shape** of the crash-box and the **thickness** profile of the metal sheet along the height.
Dimension Range: $D \in [1,34]$



3-point-bending of 5-layer beam

Variables: the **thickness** profile along each of the **5 vertical ribs** of the beam.
Dimension Range: $D \in [1,40]$



Long Crash-box

Variables: **position, protrusion** and **height** of the **triggers** (10 in total)
Dimension Range: $D \in [1,30]$

Summary

- Multi-objective optimization
- Learning-based optimization
- Bayesian optimization
- Automated algorithm design using LLMs
- Explainable AI
- Decision support and applications in engineering, medicine, supply chain, logistics, and hydrogen transport and production
- Optimization solution methods: evolutionary computation,metaheuristics and classical methods
- Hyperparameter optimization

Thank you!



Universiteit
Leiden