Benchmarking, Empirical Analysis, and Visualization

Dr. Hao Wang

November 25, 2019

Universiteit Leiden
The Netherlands

Overview

- ***Benchmarking*** optimization algorithms
  - Optimization
- ***Performance Analysis***
  - Performance indicator
- ***Statistical Comparison***
- ***Software and Visualization***
- ***Exercise***

Planning

- How to assess ***stochastic*** optimization algorithms empirically?
  - Performance indicator
  - Test functions
  - Statistics

- Black-box Optimization Benchmarking
  - Usage
  - Interpretation of the results

1

BLACK-BOX OPTIMIZATION

Intro...

- Black-box optimization problems
- Challenges in optimization problems
- Optimization algorithms

Optimization Problems

- Optimization problem - to search for the (global) ***optima*** (maxima, minima) of an ***objective function***:

$$f : \mathcal{X} \rightarrow \mathbb{R}^m$$

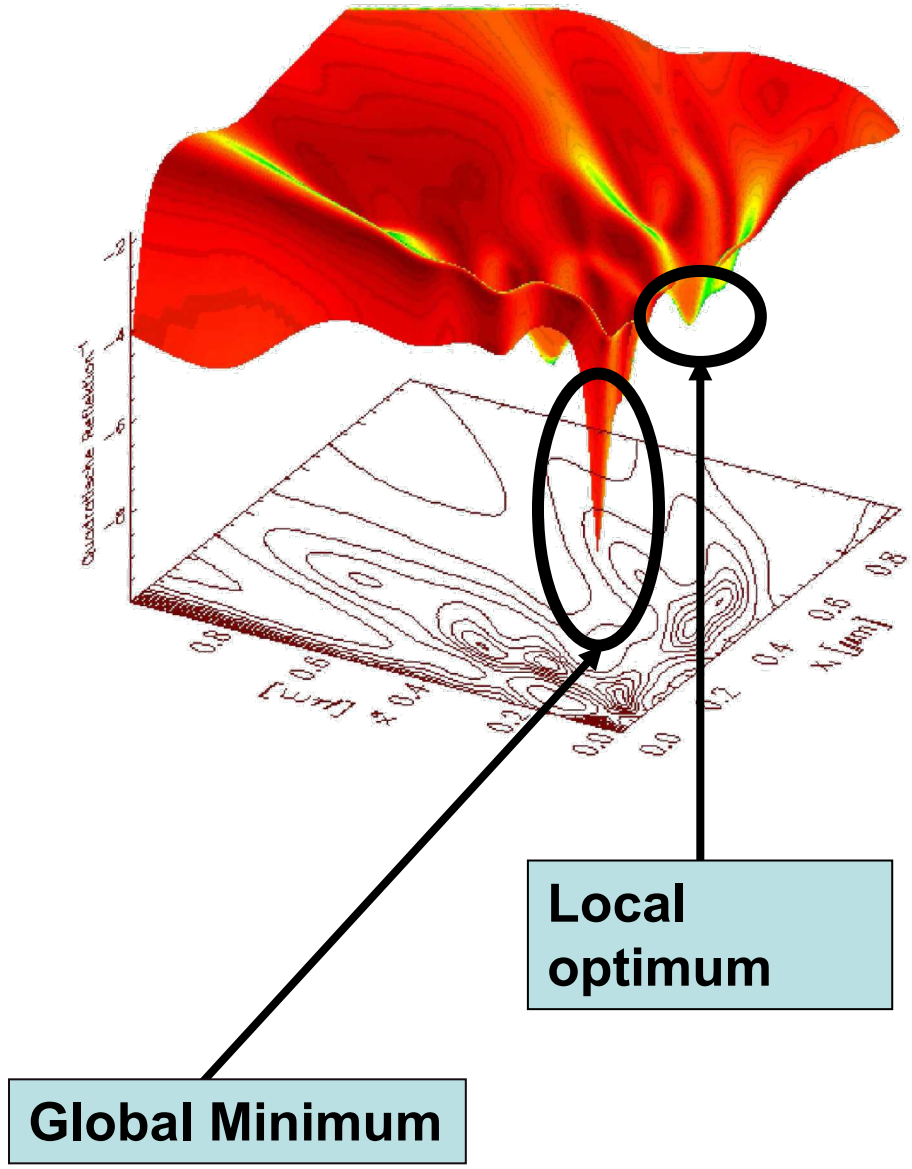
- $m = 1$: ***single-objective*** optimization problem - the topic of today
- $m > 1$: ***multi-objective*** optimization problem – looking for the Pareto front

- Search space: $\mathcal{X} := \mathbb{R}^d$ or $\mathbb{N}^d$ or $\{0, 1\}^d$ or a mix of those

- Examples
  - Optimization of a product design – improve physical properties
  - Hyper-parameter optimization in machine learning – e.g., maximize accuracy

Optimization Problems

- High-dimensional
- Non-linear, multimodal, discontinuous
- Noisy: $\forall \mathbf{x} \in \mathcal{X}, f(\mathbf{x})$ is a random variable
- Dynamic: f changes/evolves over time
- Heterogeneous
- Constraints
- Good (local) optimum is desired



Global and Local optimum

- **A metric space:** $(\mathcal{X}, D),$ where D is a distance metric
- **Global minimum**: $\mathbf{x}$ is called global mimimum iff

$$\forall \mathbf{x} \in \mathcal{X},\ f(\mathbf{x}^*) \leq f(\mathbf{x})$$

- **Local minimum**: $\tilde{\mathbf{x}}$ is called local minimum iff

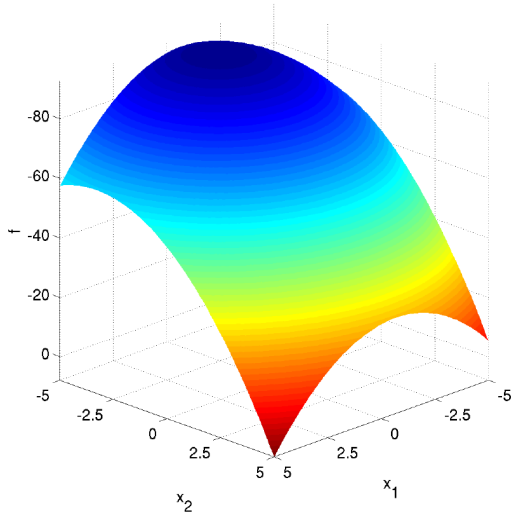
$$\exists N(\tilde{\mathbf{x}}) \in \mathcal{X}\ \forall \mathbf{x} \in N(\tilde{\mathbf{x}}),\ f(\tilde{\mathbf{x}}) \leq f(\mathbf{x})$$

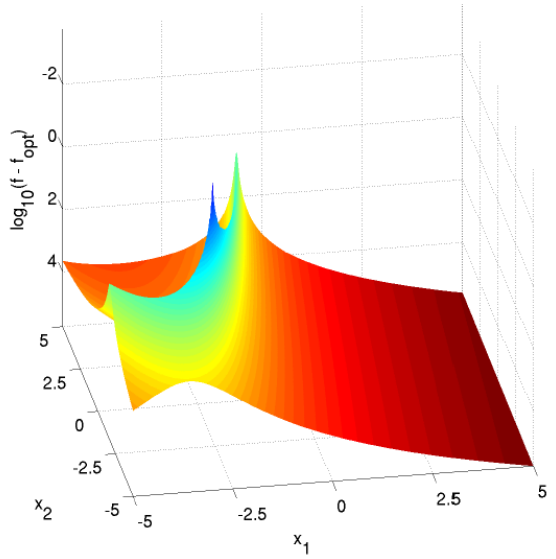
- ***Neighourhood***: e.g., an open ball centred at $\tilde{\mathbf{x}}$:

$$N_\varepsilon(\tilde{\mathbf{x}}) = \{\mathbf{x} \in \mathcal{X} \colon D(\mathbf{x}, \tilde{\mathbf{x}}) < \varepsilon\}$$

Examples

- **Continuous:**





- **Discrete:**
  - TSP, SAT, Knapsack, graph coloring…
  - **Pseudo-Boolean functions**

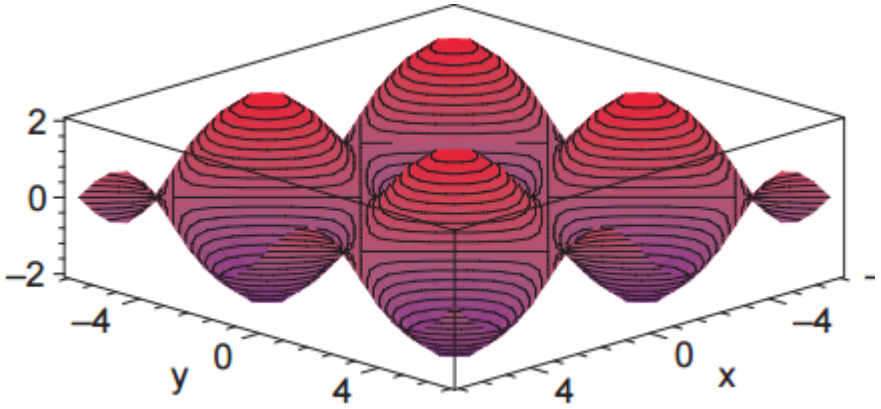
- **Mixed-integer/categorical:**
  - Tune the hyper-parameter of a Support Vector Machine: C – real , kernel (‘rbf’, ‘linear’…) – categorical, degree – integer
  - Test functions like:

$$f(\mathbf{r}, \mathbf{z}, \mathbf{c}) = \sum_{i=1}^{d_r} r_i^2 + \sum_{i=1}^{d_z} z_i^2 + \sum_{i=1}^{d_c} c_i^2$$

Continuous Optimization Problem: Challenges

What makes continuous problems hard to solve?

- ***(Non-)Separability***:
  - Additive: $f(\vec{x}) = \sum_i f_i(x_i)$
  - Multiplicative: $f(\vec{x}) = \prod_i f_i(x_i)$
  - Optimum can be reached by searching along each dimension



$\sin(x) + \sin(y)$

Continuous Optimization Problem: Challenges

What makes continuous problem hard to solve?

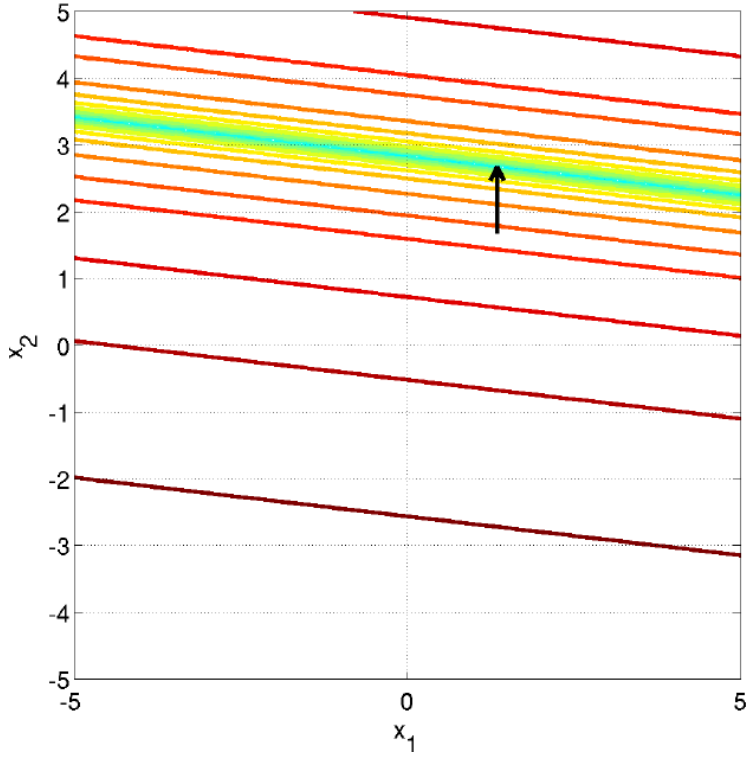
- ***Ill-Conditioning***:
  - measures how much the output value of the function can change for a small change in the input argument
  - let ***H*** be the Hessian matrix ($d \times d$) of f: $(\mathbf{H})_{ij} = \dfrac{\partial^2 f}{\partial x_i x_j}$
  - the ***condition number*** is:

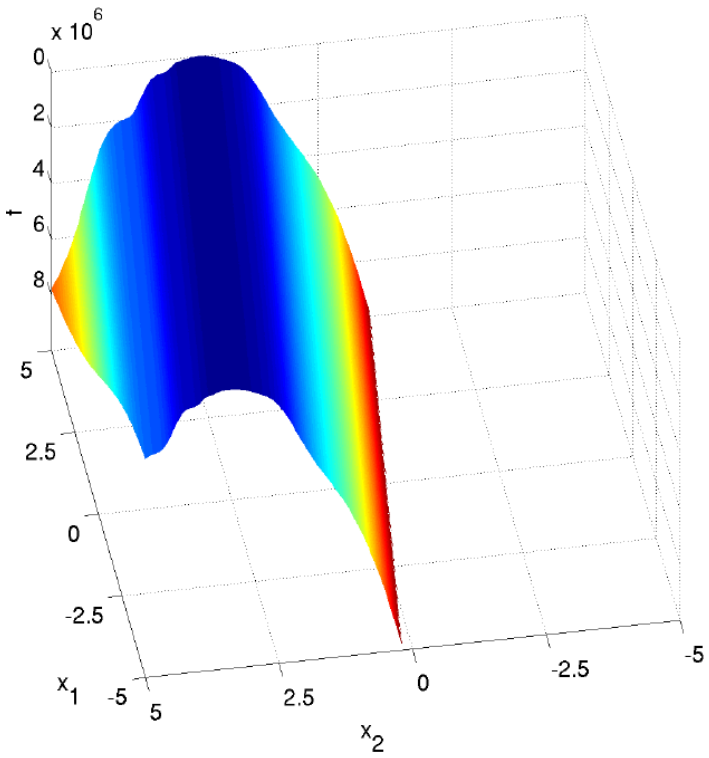
$$k(\mathbf{H}) = \frac{|\lambda_{\max}(\mathbf{H})|}{|\lambda_{\min}(\mathbf{H})|}$$

Continuous Optimization Problem: Challenges

What makes continuous problem hard to solve?

- ***Ill-Conditioning***:
  - The contourlines are extremely 'stretched'

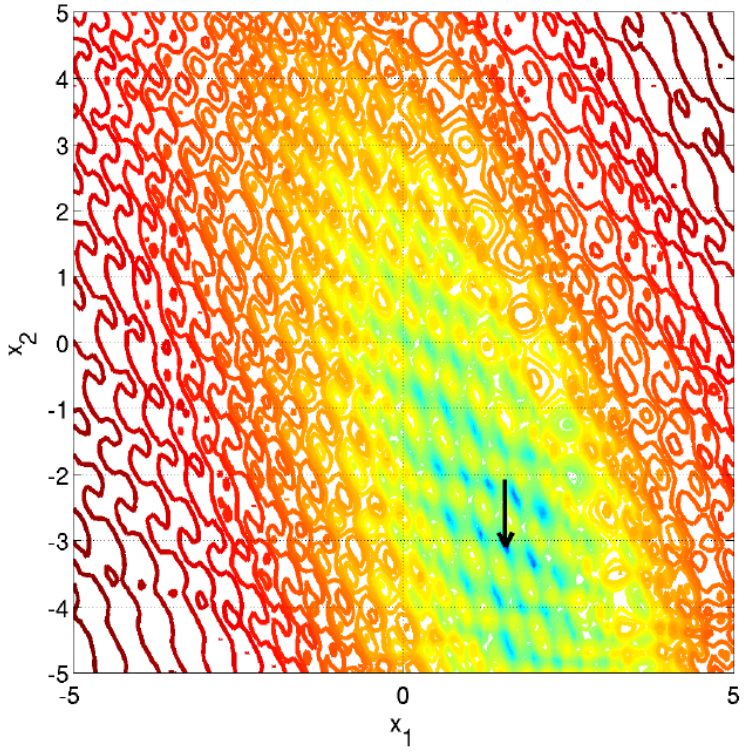


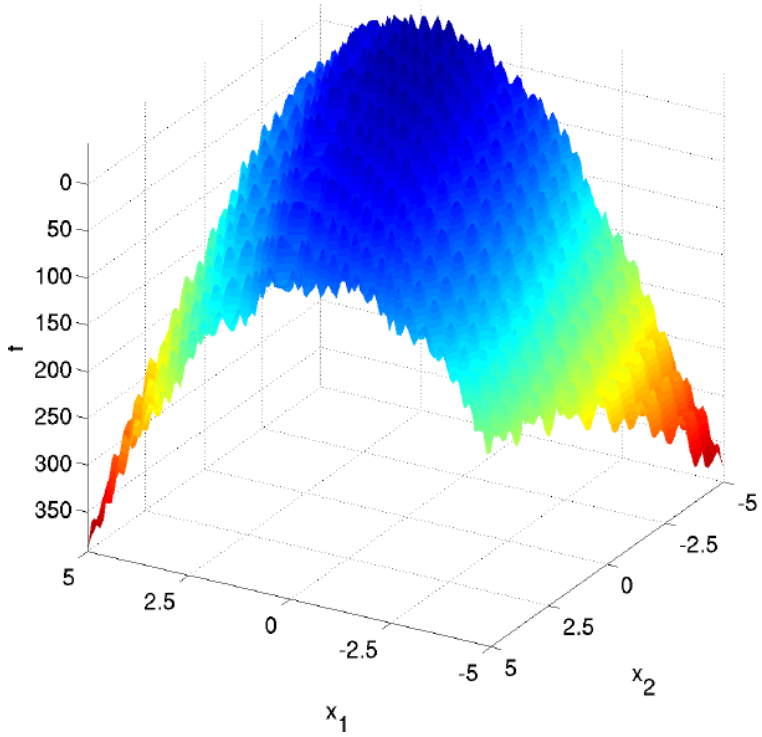


Continuous Optimization Problem: Challenges

What makes continuous problem hard to solve?

- ***Irregularities and high multimodality***:
  - Small, local, but visible variations of the landscape

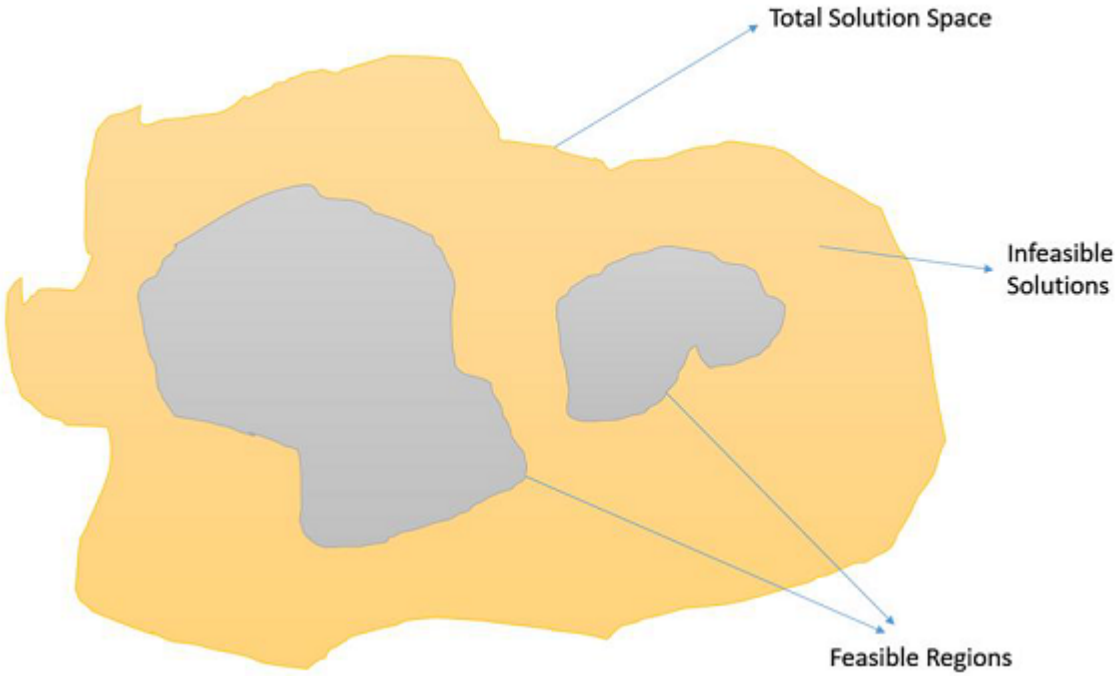




Continuous Optimization: Challenges

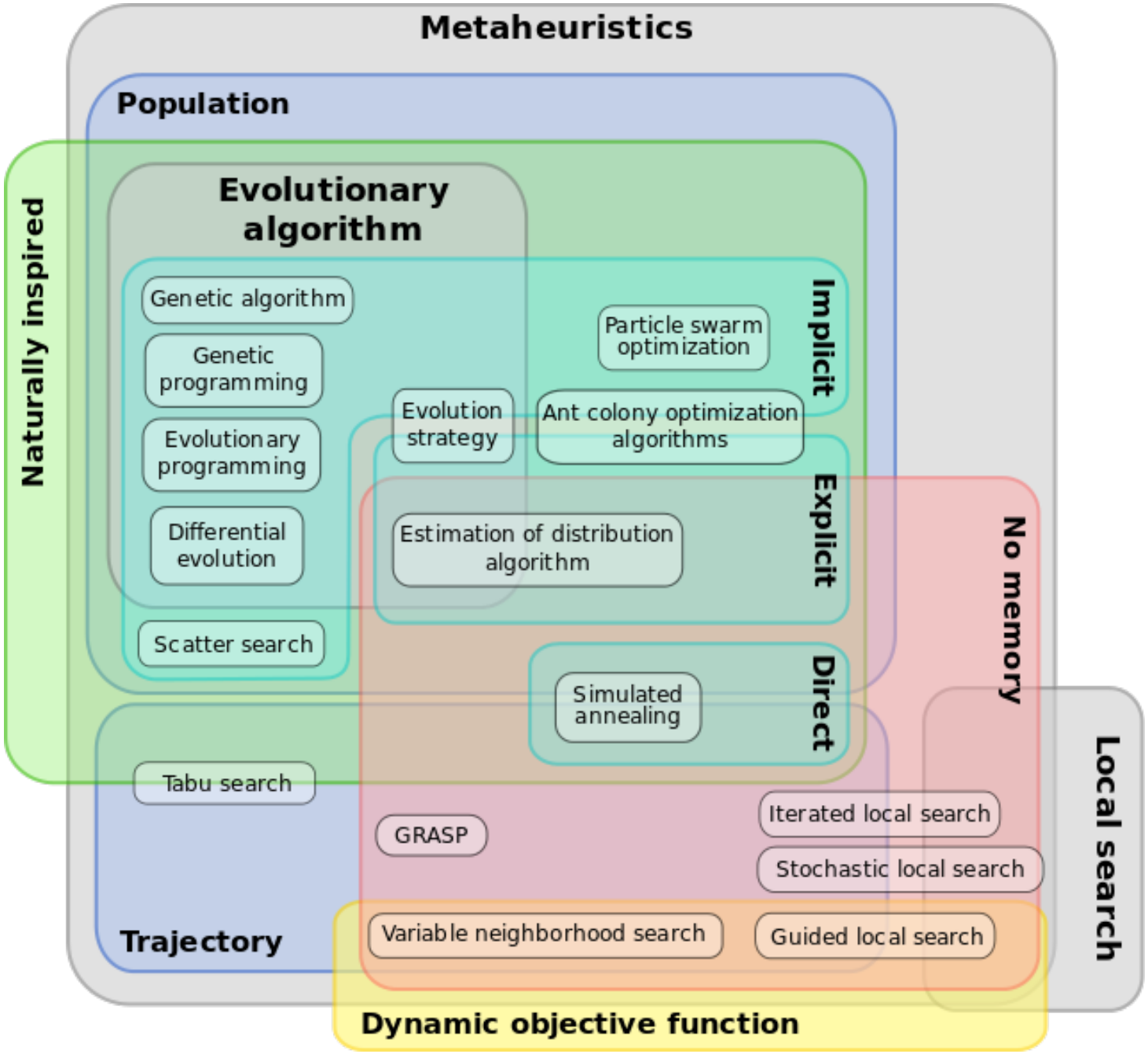
What makes continuous problem hard to solve?

- ***Constraints***: scattered feasible regions



Optimization Algorithms

- ***Direct*** search: algorithms only ***values*** of objective functions
  - As opposed to gradient/Hessian method
- ***Cost*** of search: ***function evaluations***
  - Represent time/money-costly physical experiments/simulation
- ***Global*** search: $t \to \infty, \quad \mathbf{x}_t \xrightarrow{P} \mathbf{x}^*$
- Most of them are iterative…



Iterative Optimization Heuristics (IOHs)

procedure IOH
  $t \leftarrow 0$ ▷ iteration counter
  $\mathcal{H} \leftarrow \emptyset$ ▷ search history
  Λ a probability distribution on $\mathbb{N}_{\geq 0}$ ▷ number of trial points in each iteration
  P a probability distribution on $\mathcal{X}$ ▷ strategy to generate search points
  while termination criterion not met **do**
    $t \leftarrow t + 1$
    update Λ based on $\mathcal{H}$
    update P based on $\mathcal{H}$
    $\lambda(t) \sim \Lambda$ ▷ number of points to be queried in t-th iteration
    for $i = 1 \rightarrow \lambda(t)$ **do**
      $\mathbf{x}_i^{(t)} \sim P$
      $y_i^{(t)} \leftarrow f(\mathbf{x}_i^{(t)})$ ▷ function evaluation
    end for
    $H \leftarrow \text{select}\left(\left\{(x_i^{(t)}, y_i^{(t)})\right\}_{i=1}^{\lambda(t)}\right)$ ▷ select a subset of the evaluation information
    $\mathcal{H} \leftarrow \mathcal{H} \bigcup H$ ▷ update the history
  end while
end procedure

Example IOHs

- Search space $\mathcal{X} = \{0,1\}^d$

Algorithm: Greedy hill climber (gHC)

Initialization: Sample $x \in \{0,1\}^n$ uniformly at random and evaluate $f(x)$;
Optimization: **for** $t = 1, 2, 3, \ldots$ **do**
    $x^* \leftarrow x$;
    Flip in x^* the entry in position $1 + (t \mod n)$ and evaluate $f(x^*)$;
    if $f(x^*) \geq f(x)$ **then** $x \leftarrow x^*$;

Example IOHs

Algorithm: The $(1 + \lambda)$ EA with static mutation rates

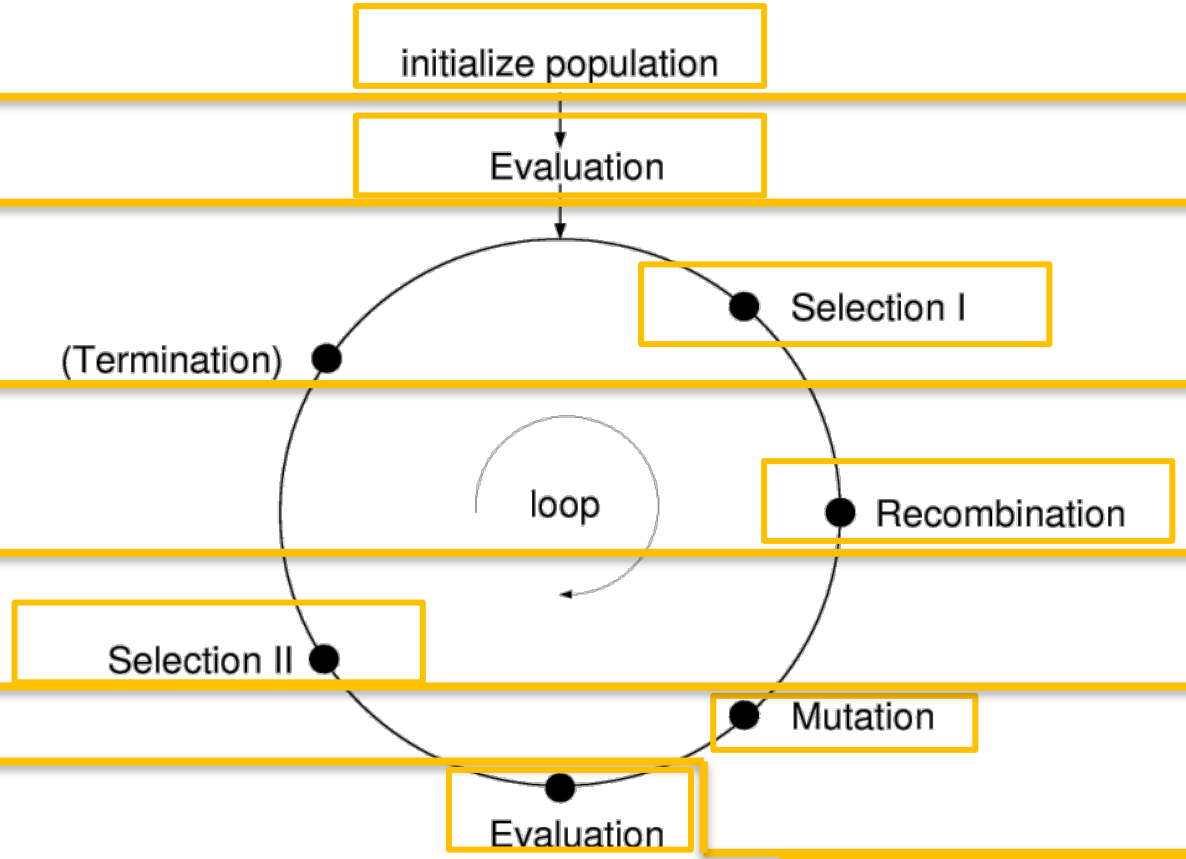
Initialization: Sample $x \in \{0,1\}^n$ uniformly at random and evaluate $f(x)$;
Optimization: **for** $t = 1, 2, 3, \ldots$ **do**
  for $i = 1, \ldots, \lambda$ **do**
    Sample $\ell^{(i)} \sim \text{Bin}_{>0}(n, 1/n)$;
    create $y^{(i)} \leftarrow \text{flip}_{\ell^{(i)}}(x)$, and evaluate $f(y^{(i)})$;
  $x^* \leftarrow \arg\max\{f(y^{(1)}), \ldots, f(y^{(\lambda)})\}$ (ties broken by selecting the first max $f(y^{(i)})$);
  if $f(x^*) \geq f(x)$ **then** $x \leftarrow x^*$;

Algorithm: flip_ℓ chooses ℓ different positions and flips the entries in these positions.

Input: $x \in \{0,1\}^n$, $\ell \in \mathbb{N}$;
Select ℓ pairwise different positions $i_1, \ldots, i_\ell \in [n]$ uniformly at random;
$y \leftarrow x$;
for $j = 1, ..., \ell$ **do** $y_{i_j} \leftarrow 1 - x_{i_j}$;

Example: Evolutionary Algorithms

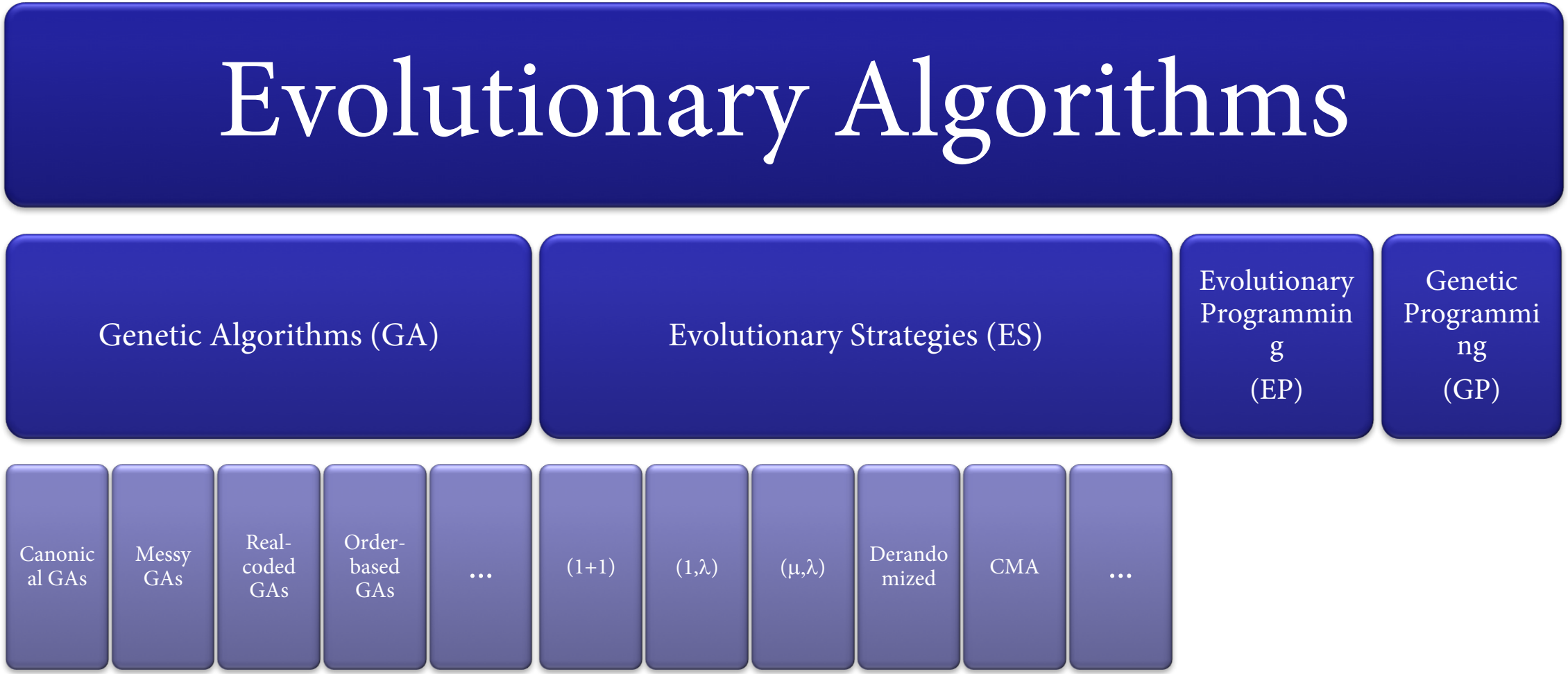
- ***Global, direct search*** algorithms



	Genetic Algorithm	Evolution Strategy
	$\vec{a} \in \{0,1\}^n$ Random	$\vec{a} \in \mathbb{R}^n$
	Objective function: $f(\vec{a}) \to max$	
	Fitness-proportional $p_i = \frac{f_i}{\sum_{j=1}^{\mu} f_j}$	Random $p_i = 1/\mu$
		
	None	Deterministic $(\mu, \lambda), (\mu + \lambda)$
	iid bit flips w.p. $p \in]0,1[$	iid 1-dim. / n-dim. Gaussian $X_i \sim N(0, \sigma^2)$ $\boldsymbol{X} \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$
	Objective function: $f(\vec{a}) \to max$	

Huge range of applications: Machine learning, search, engineering, logistics, science, etc.

Evolutionary Algorithms Taxonomy



2

BENCHMARKING

Benchmark Design

- A problem class of interest
  - *Domain*: Continuous, discrete, …
  - Single-/multi-objective?
  - Black/White/Grey-Box?
  - *Dimensionality*: low, medium, high?
  - Expensive, global, graph problems, game, Machine Learning problem, quantum computing…
- A set of test functions: $\mathcal{F} = \{f_1, f_2, \ldots\}$
  - Universality: the larger, the better…
  - Test functions should not be similar – Explorative Landscape Analysis
- A set of optimization algorithms: $\mathcal{A} = \{A_1, A_2, \ldots\}$

Benchmark Design

- Performance measure/indicators
  - ~~Speed – CPU/Wall clock time → machine-dependent, not recommended~~
  - Speed – Running time/the number of function evaluations, integer-valued r.v.

$$T(A, f, d, v) \in [1..B] \cup \{\infty\}$$

  - (single-objective) Quality – Function value, real-valued r.v.

$$V(A, f, d, v) \in \mathbb{R}$$

  - (multi-objective) Performance indicators for multi-objective problems, e.g., the hypervolume indicator, generational distance (GD)
  - We are interested in the distribution of T and V…

Zitzler, Eckart, Lothar Thiele, Marco Laumanns, Carlos M. Fonseca, and Viviane Grunert Da Fonseca. "Performance assessment of multiobjective optimizers: An analysis and review." *IEEE Transactions on evolutionary computation* 7, no. 2 (2003): 117-132.

Benchmark Design

- ***Execution:*** run each pair of (A, f, d) for several times independently
  - Estimate the empirical distribution $\{v_1, v_2, \ldots, v_r\} \rightarrow \hat{F}_V$

    $\{t_1, t_2, \ldots, t_r\} \rightarrow \hat{F}_T$
  - ***Stopping Criteria?*** e.g., a budget on the function evaluation, or a target function value to hit
  - The behavior of the optimization algorithm is usually stochastic
  - How to determine the number of runs/repetitions r? – to obtain enough data/evidence

Benchmark Design

- Data collection/format
  - What to store? Running time and the corresponding function value, some internal (dynamic) parameters?
  - When to store?
- Problem invariances? A limited number of test functions → the risk of ***overfitting***

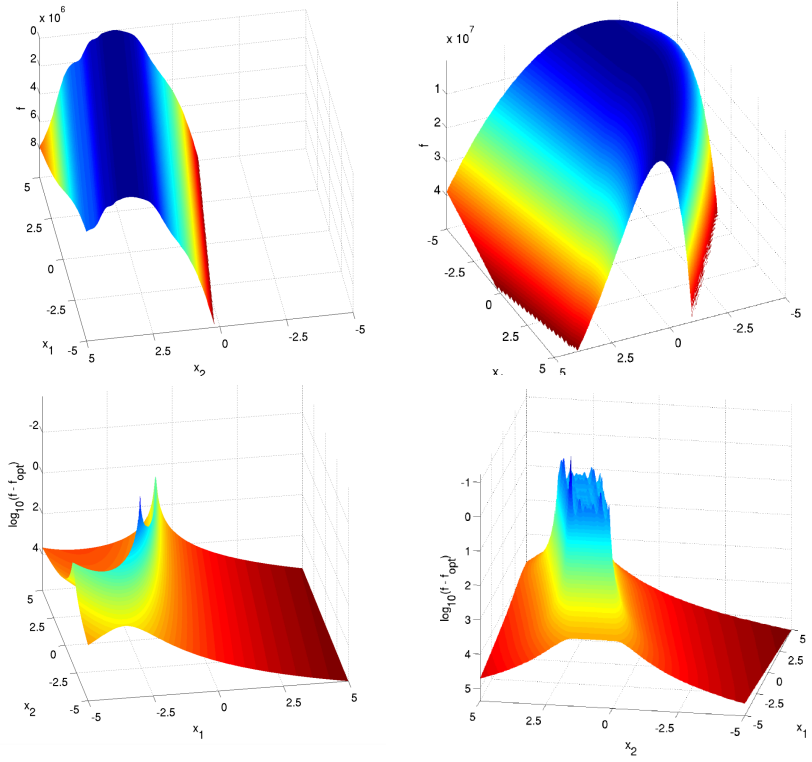
Black-Box Optimization Benchmarking (BBOB)

- Single-objective, continuous, black-box, (noisy) optimization tasks
- Widely used by many scientists and conference, e.g. GECCO Workshop for Real-Parameter Optimization

- 24 test functions
  - 3 unimodal, separable (e.g., sphere)
  - 7 unimodal, non-separable (e.g., bent cigar)
  - 2 multimodal, separable (e.g., Rastrigin)
  - 12 multimodal, non-separable (e.g., Schwefel)

- https://coco.gforge.inria.fr/

- https://github.com/numbbo/coco



Black-Box Optimization Benchmarking (BBOB)

- ***Function instance*** – each test function is modified 'slightly' by:
  - Translation

$$\tilde{f} := f(\mathbf{x} + \mathbf{x}_{\text{offset}}) + y_{\text{offset}}$$

  - Rotation of the search space

$$\tilde{f} := f(\mathbf{Rx}),\ \mathbf{R} \text{ is a random orthonormal matrix}$$

  - Small irregularities on the function landscape

Hansen, Nikolaus, Steffen Finck, Raymond Ros, and Anne Auger. "Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions." (2009).

Pseudo-Boolean Optimization (PBO)

- Binary alphabet, black-box, deterministic optimization tasks
- 23 test problems $f : \{0,1\}^d \to \mathbb{R}$
- Online performance analysis tool, **IOHanalyzer,** https://iohprofiler.liacs.nl/
- https://iohprofiler.github.io/
- https://github.com/IOHprofiler/IOHexperimenter

Pseudo-Boolean Optimization (PBO)

- F1: OneMax
- F2: LeadingOnes
- F3: Harmonic Linear Function
- F4-F10: **W-Model** on top of OneMax
  - W-model allows to scale effective dimension, epistasis, neutrality, ruggedness, fitness plateaus, etc.
- F11-17: **W-Model** on top of LeadingOnes
- F18: LABS: Low Autocorrelation Binary Sequences
- F19: Ising Model: Ring
- F20: Ising Model: Torus
- F21: Ising Model: Triangular (Isometric 2D Grid)
- F22: MIVS: Maximum Independent Vertex Set
- F23: N-Queens Problem

Weise, Thomas, and Zijun Wu. "Difficult features of combinatorial optimization problems and the tunable w-model benchmark problem for simulating them." In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 1769-1776. ACM, 2018.

Pseudo-Boolean Optimization (PBO)

- F1: OneMax

$$\mathrm{OM} : \{0,1\} \to [0..d], x \mapsto \sum_{i=1}^{d} x_i.$$

- F2: LeadingOnes

$$\mathrm{LO} : \{0,1\}^d \to [0..d], x \mapsto \max\{i \in [0..d] \mid \forall j \leq i : x_j = 1\} = \sum_{i=1}^{d} \prod_{j=1}^{i} x_j,$$

- F3: Harmonic Linear Function

$$f : \{0,1\}^d \to \mathbb{R}, x \mapsto \sum_{i} i x_i$$

Pseudo-Boolean Optimization (PBO)

- F4 – F17: OneMax transformed by the so-called W-model
  - **Dummy variables:** $(x_1, \ldots, x_d) \to (x_{i_1}, \ldots x_{i_m}),\ m < d$
    $i_1, \ldots i_m$ are random chosen from $[1..d]$
  - **Neutrality:** $(\underbrace{x_1, \ldots, x_\mu}_{x'_1}, \underbrace{x_{\mu+1}, \ldots, x_{2\mu}}_{x'_2}, \ldots, \underbrace{x_{d-\mu+1}, \ldots, x_d}_{x'_{d/\mu}})$

    x'_i's are the majority vote in each block
  - **Epistasis:** local permutations
  - **Fitness perturbation**

Pseudo-Boolean Optimization (PBO)

- F11-17: LeadingOnes transformed by the so-called W-model
- F18: LABS: Low Autocorrelation Binary Sequences

$$F_{\text{LABS}}(\vec{x}) = \frac{d^2}{2\sum_{k=1}^{d-1}\left(\sum_{i=1}^{d-k} x'_i \cdot x'_{i+k}\right)^2} \quad \text{where } x'_i = 2x_i - 1.$$

Autocorrelation with gap *k*

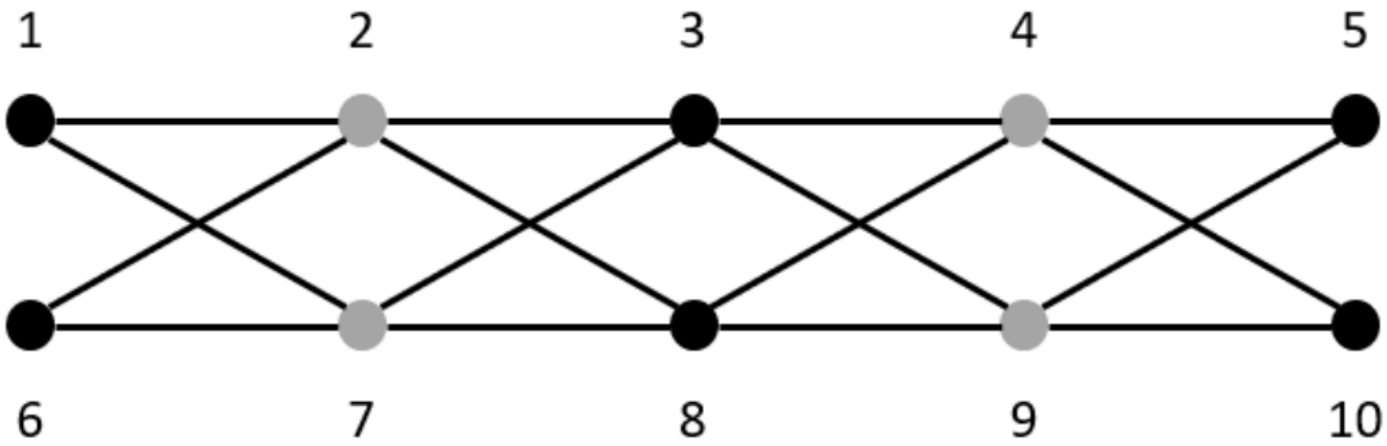
Mertens, Stephan. "Exhaustive search for low-autocorrelation binary sequences." *Journal of Physics A: Mathematical and General* 29, no. 18 (1996): L473.

Pseudo-Boolean Optimization (PBO)

- F22: Maximum Independent Vertex Set (MIVS)
  - Given a graph $G = ([d], E)$

$$F_{\text{MIVS}}(x) = \sum_i x_i - d \cdot \sum_{i,j} x_i x_j e_{i,j},$$

  - A scalable graph structure:



3

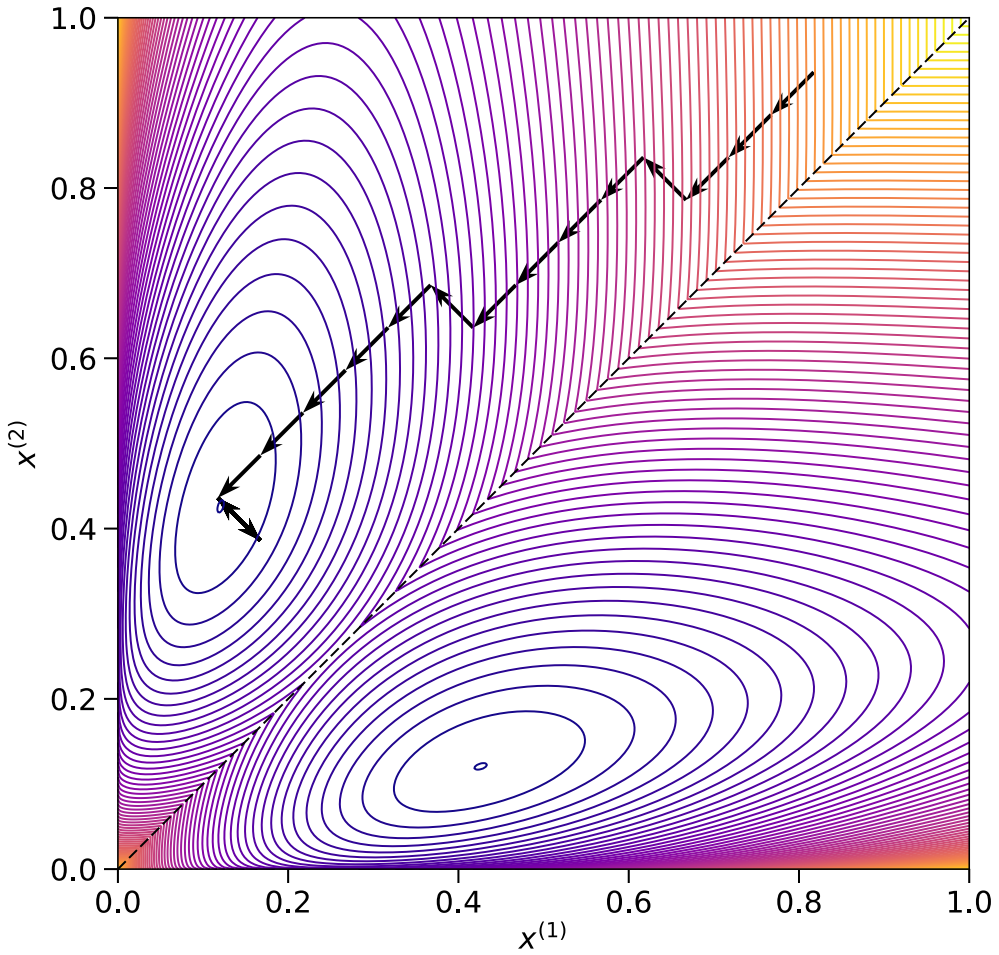
PERFORMANCE ANALYSIS

Empirical Investigation

- How to evaluate an iterative optimization heuristic?
  - Running time/evaluation cost for finding the optimum
  - Fixed cost error to the optimization target
  - Probability of achieving the target value

Empirical Behavior

- What to record/look at?
  - ***Trajectory in the search space***
  - Evolution of objective values

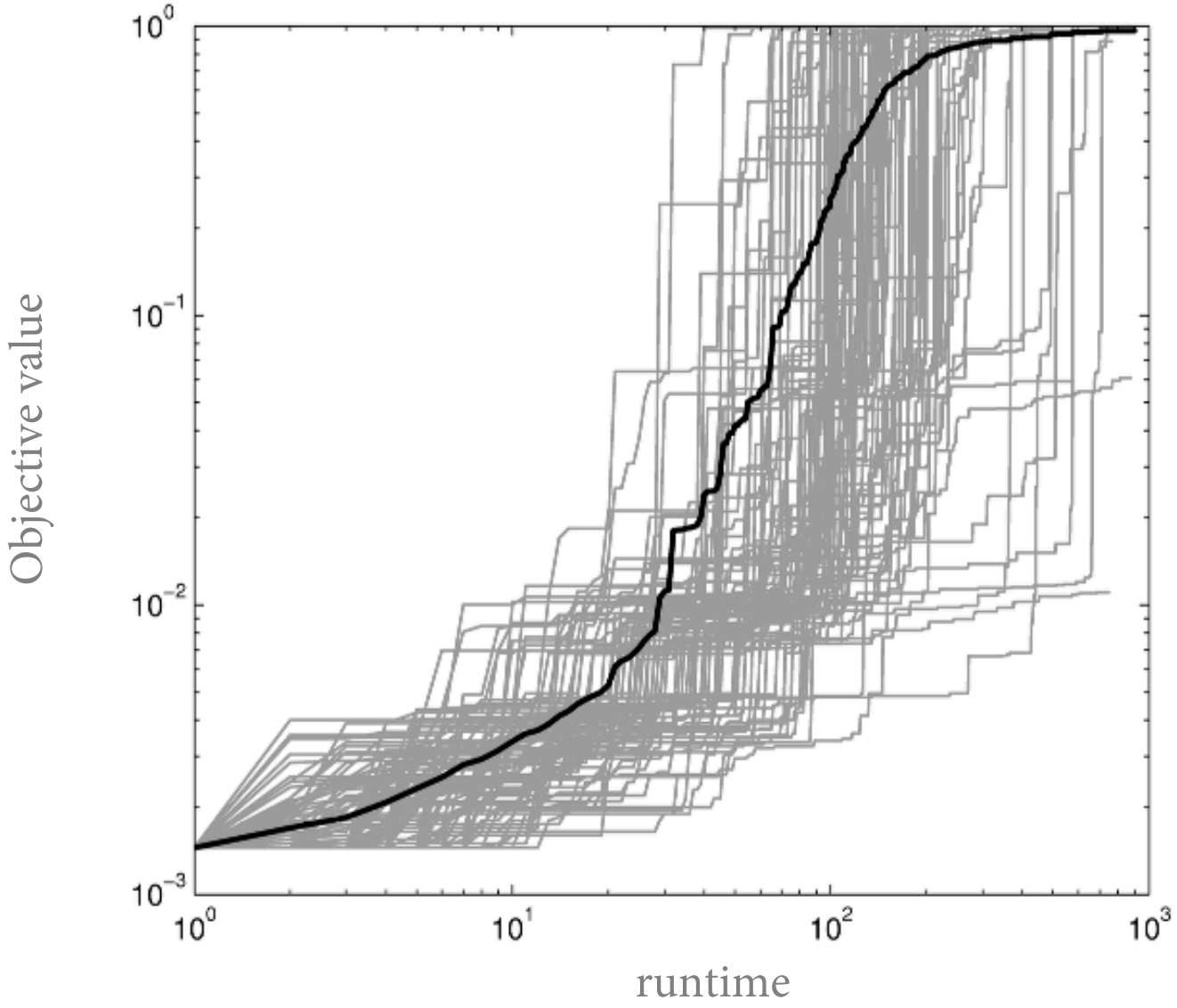


Empirical Behavior

- What to record/look at?
  - Trajectory in the search space
  - ***Evolution of objective values***

```
runtime objective value
1       +4.684923580e+01
2       +3.845582229e+01
5       +1.524053192e+01
6       +3.712980563e+00
9       +1.112282445e-01
33      +1.561853208e-02
66      +5.275020213e-03
81      +2.446356491e-04
89      +3.933668859e-05
114     +1.742187840e-05
115     +1.535847593e-05
118     +8.255657917e-06
124     +6.025853736e-06
136     +1.445371026e-06
139     +1.420293074e-07
152     +9.457812666e-08
178     +4.383980468e-08
181     +2.249237241e-08
203     +5.198899089e-09
runtime objective value
1       +4.964301589e+00
2       +2.352169383e+00
3       +1.196940489e+00
14      +9.820808830e-01
25      +1.208380990e-02
55      +7.985427427e-03
61      +7.019147663e-04
68      +4.893246434e-04
77      +2.294279057e-05
94      +1.084655583e-05
95      +5.658112698e-07
128     +3.576668917e-07
135     +6.931486496e-08
142     +8.638266991e-09
```

Evolution of Objective Values



Performance Measure

- Stochastic behavior → statistical characteristics

- ***Convergence rate***

$$r = \lim_{n \to \infty} \frac{\Delta f_n}{T_n}$$

  - Error in each iteration: $\Delta f_n = |f_n^{\text{best}} - f^*|$
  - No. of function evaluations until iteration n: $T_n = \sum_{i=1}^{n} \#\,\text{EVAL}_i$
  - ***Asymptotic behavior***

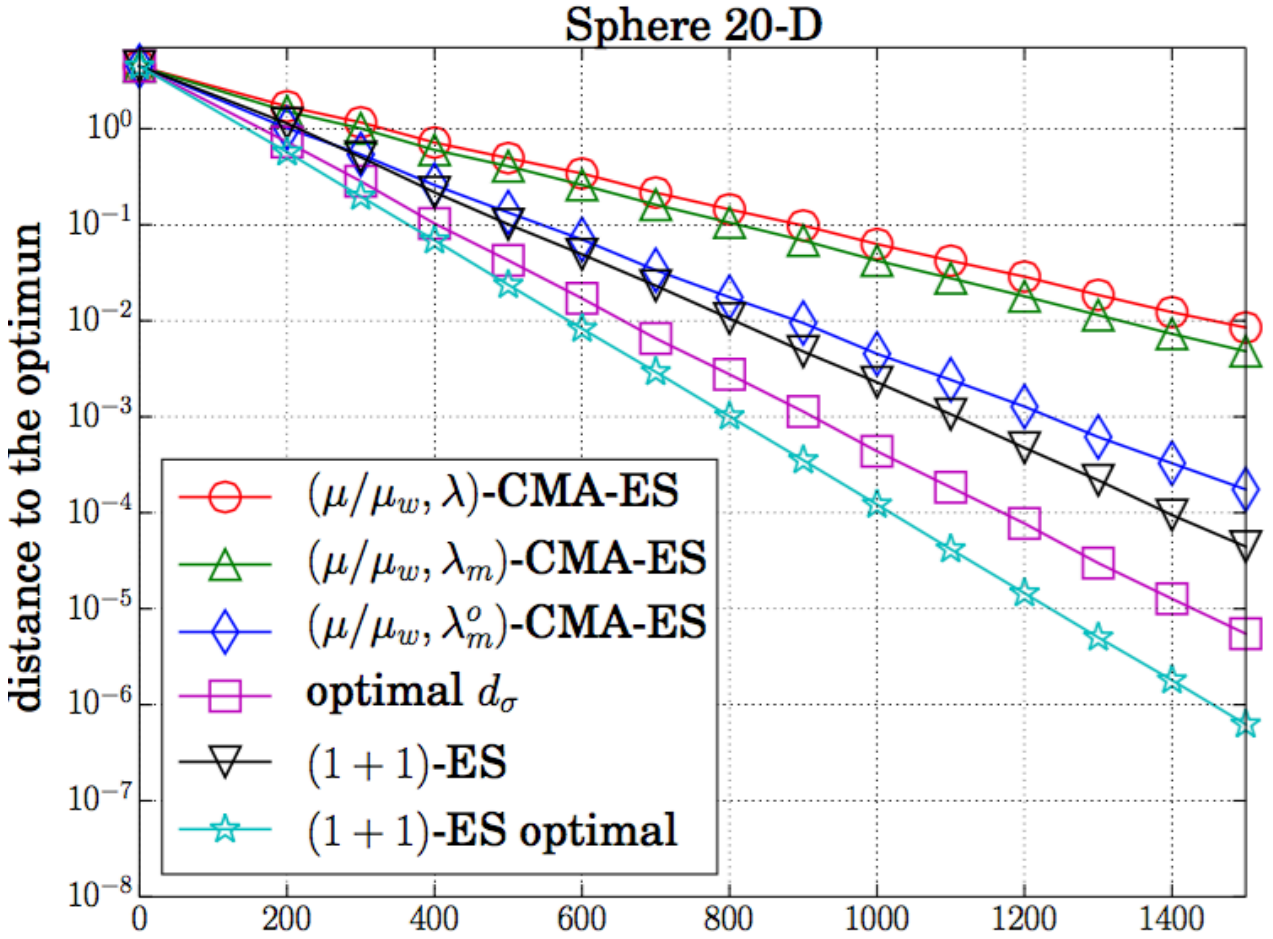
Convergence Rate

- ***Convergence rate in practice***
  - Instead of running the algorithm forever…
  - Estimated by averaging over many runs

$$\hat{r} = \frac{1}{M} \sum_{i=1}^{M} \frac{\Delta f_n}{T_n}$$

Convergence Rate

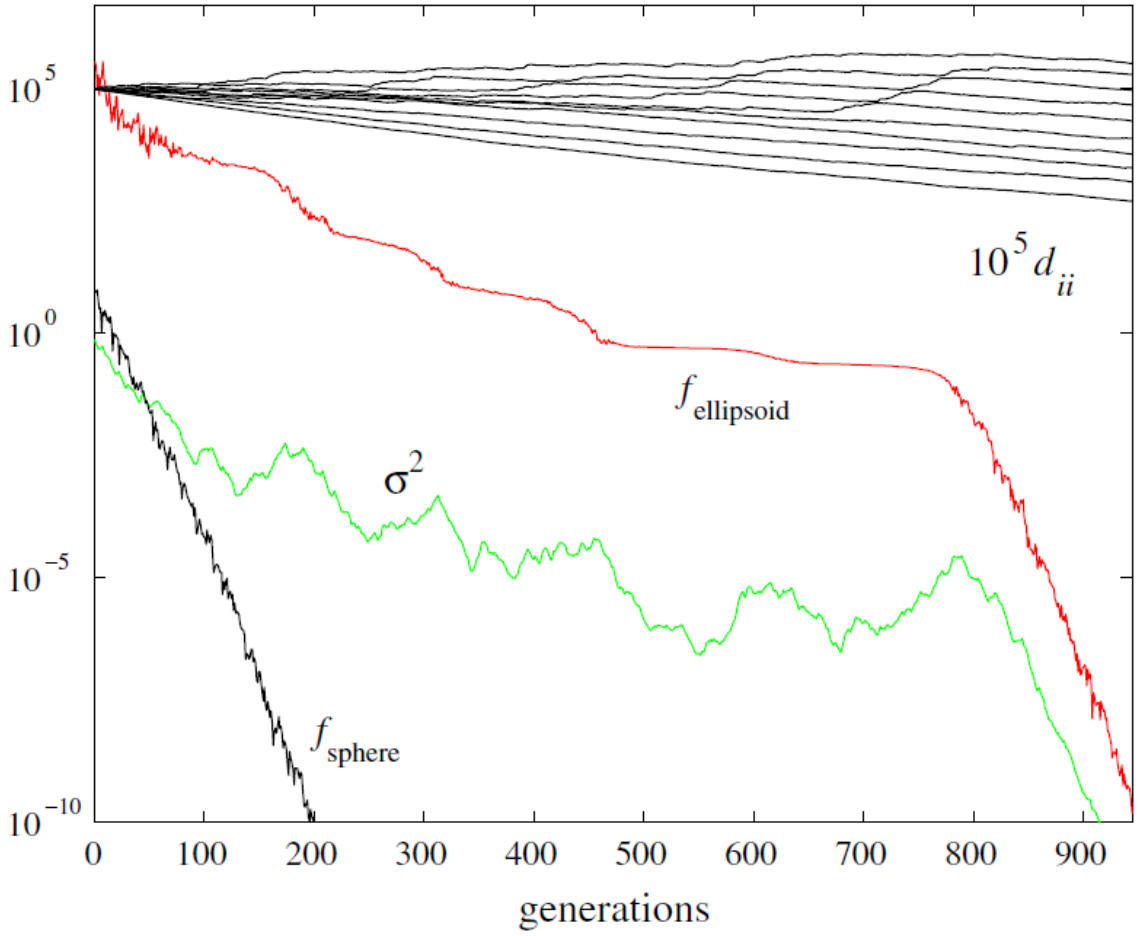
Convergence rate: slope of the curve



Convergence Rate

Not very practical

- Requires lots of runs
- Not always linear
- Not always constant



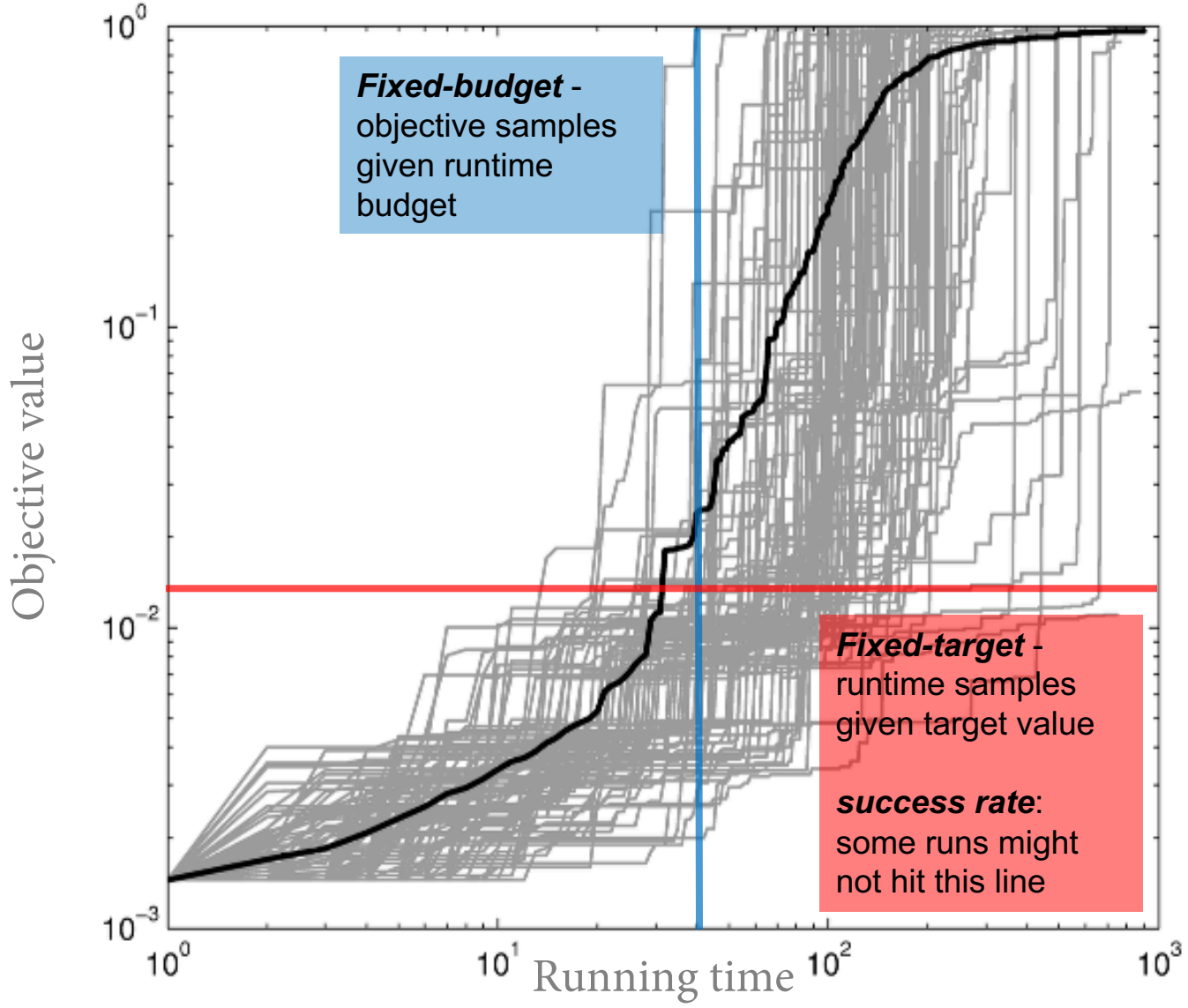
Expected Improvement

- ***Expected Improvement / Progress rate***

$$c = \mathbb{E}||f_n^{\text{best}} - f_{n+1}^{\text{best}}||$$

  - Progress is stochastic → take the mean
  - In practice, runtime is considered more important than objective values
  - Mainly used in the theory…

How to assess Empirical Performance?



How to Assess Empirical Performance?

- ***Measure Cost – fixed-target Running Time***
  - ***Horizontal*** perspective
  - Recommended in the benchmarking
  - Advantage due to ***quantitative*** comparison

- ***Measure quality – Fixed-budget objective value***
  - ***Vertical*** perspective
  - Only for ***qualitative*** comparison
  - Directly applicable for runs with very small number of function evaluations

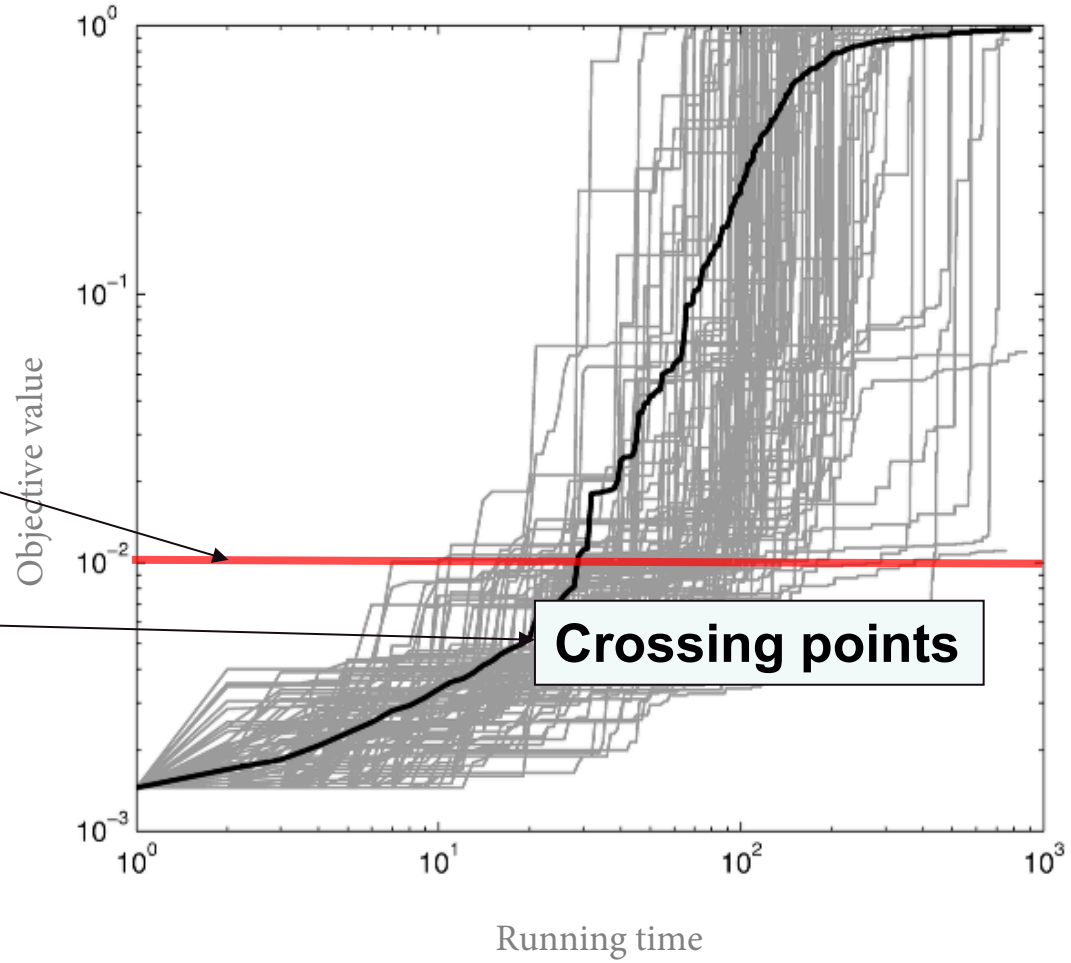
Formal Setup

- Running time – random variable
  - Parameterized by ***a target value***

$$T(f_{\text{target}}) \in \mathbb{N}_{>0}$$

- The number of runs – r

$$\{t_1(f_{\text{target}}), \ldots, t_r(f_{\text{target}})\}$$



- Successful? $t_i(f_{\text{target}}) < \infty$
- Unsuccessful? $t_i(f_{\text{target}}) = \infty$

- Number of successes $N_{\text{succ}} = \sum_{i=1}^{r} \mathbf{1}(t_i(f_{\text{target}}) < \infty)$

Formal Setup

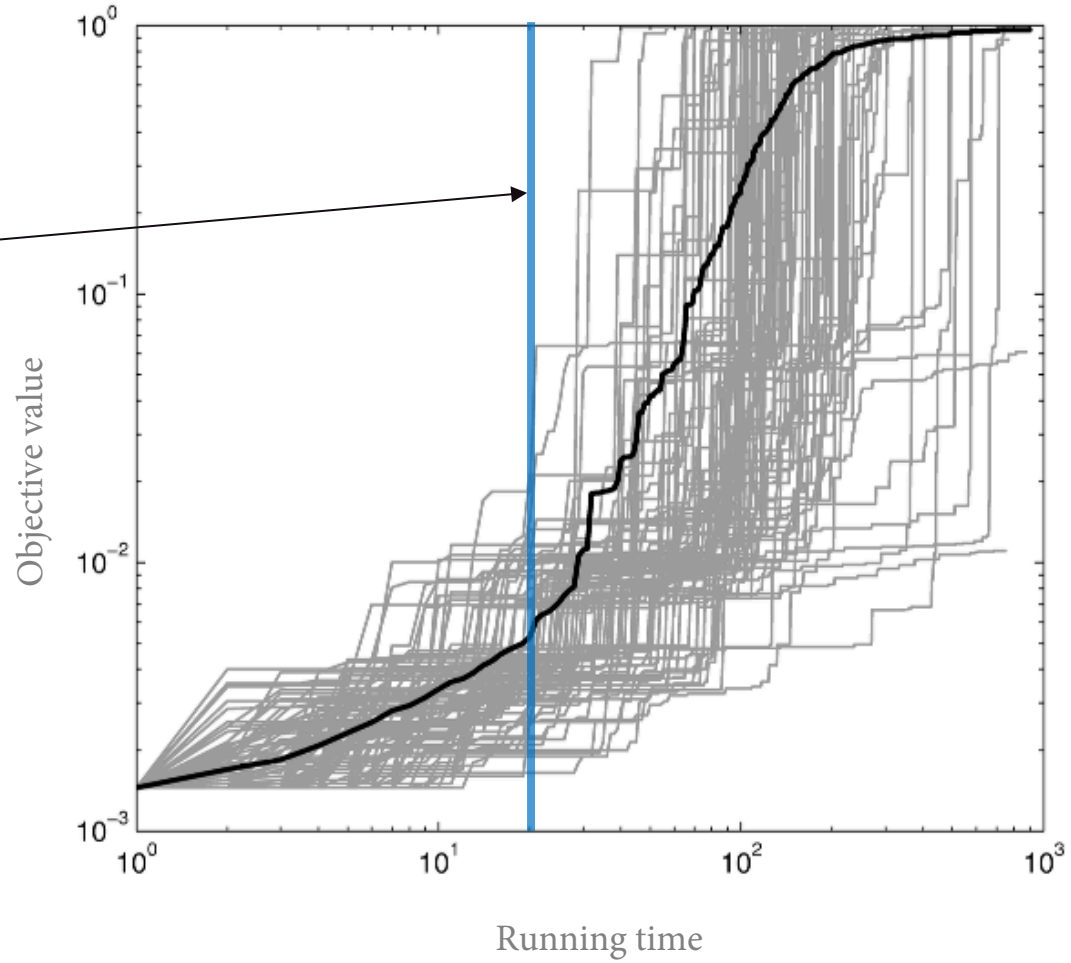
- Function value – random variable
  - Parameterized by ***a budget value***

$$V(b) \in \mathbb{R}, b \leq B$$

- The number of runs – r

$$\{v_1(b), \ldots, t_r(b)\}$$

Crossing points



Descriptive Statistics

- Sample mean, median, standard deviation…

Underestimates the mean of running times when infinite budget is allowed

$$\bar{T}(v) = \frac{1}{r}\sum_{i=1}^{r} \min\left\{t_i(A, f, d, v), B\right\}, \quad \bar{V}(t) = \frac{1}{r}\sum_{i=1}^{r} v_i(A, f, d, t).$$

- Sample quantiles, e.g., 2%, 5%, …, 98%

$$|\{t_i \leq Q_{m\%}\}|/r = m\%$$

- Empirical success rate

$$\widehat{p}_s = \sum_{i=1}^{r} \mathbb{1}(t_i(A, f, d, v) < \infty)/r \xrightarrow{P} \mathrm{E}[\mathbb{1}(T(A, f, d, v) < \infty)]$$

Expected Running Time

- ***Unbiased*** estimator the mean running time
- Algorithm A: ***40*** runs, ***2000*** evaluations on average ***75%*** success rate
- Algorithm B: ***40*** runs, ***3000*** evaluations on average ***90%*** success rate

Expected Running time

- ***Expected Running Time*** (ERT)

$$\mathrm{ERT}(A, f, d, v) = \frac{\sum_{i=1}^{r} \min\{t_i(A, f, d, v), B\}}{\sum_{i=1}^{r} \mathbb{1}(t_i(A, f, d, v) < \infty)}.$$

- ERT algorithm A: $\frac{2000}{0.75} = 2666.7$
- ERT algorithm B: $\frac{3000}{0.9} = 3333.3$

Empirical Distribution Functions

- What if ERTs of two algorithms are the same?
  - *Algorithm A:* ***15*** *runs,* ***2000*** *evaluations on average 10 runs: 2200 evals, 5 runs: 1600 evals, success rate* ***1.0***
  - *Algorithm B:* ***15*** *runs,* ***2000*** *evaluations on average 2 runs: 3000 evals, 8 runs: 1900 evals, 5 runs: 1760 evals, success rate* ***1.0***
  - Which is better?

- ***Empirical Cumulative Distribution Functions of runtime***

  - Instead of just looking at the mean of runtime samples
  - Estimate the ***distribution*** of runtime

Empirical Cumulative Distribution Functions (ECDFs)

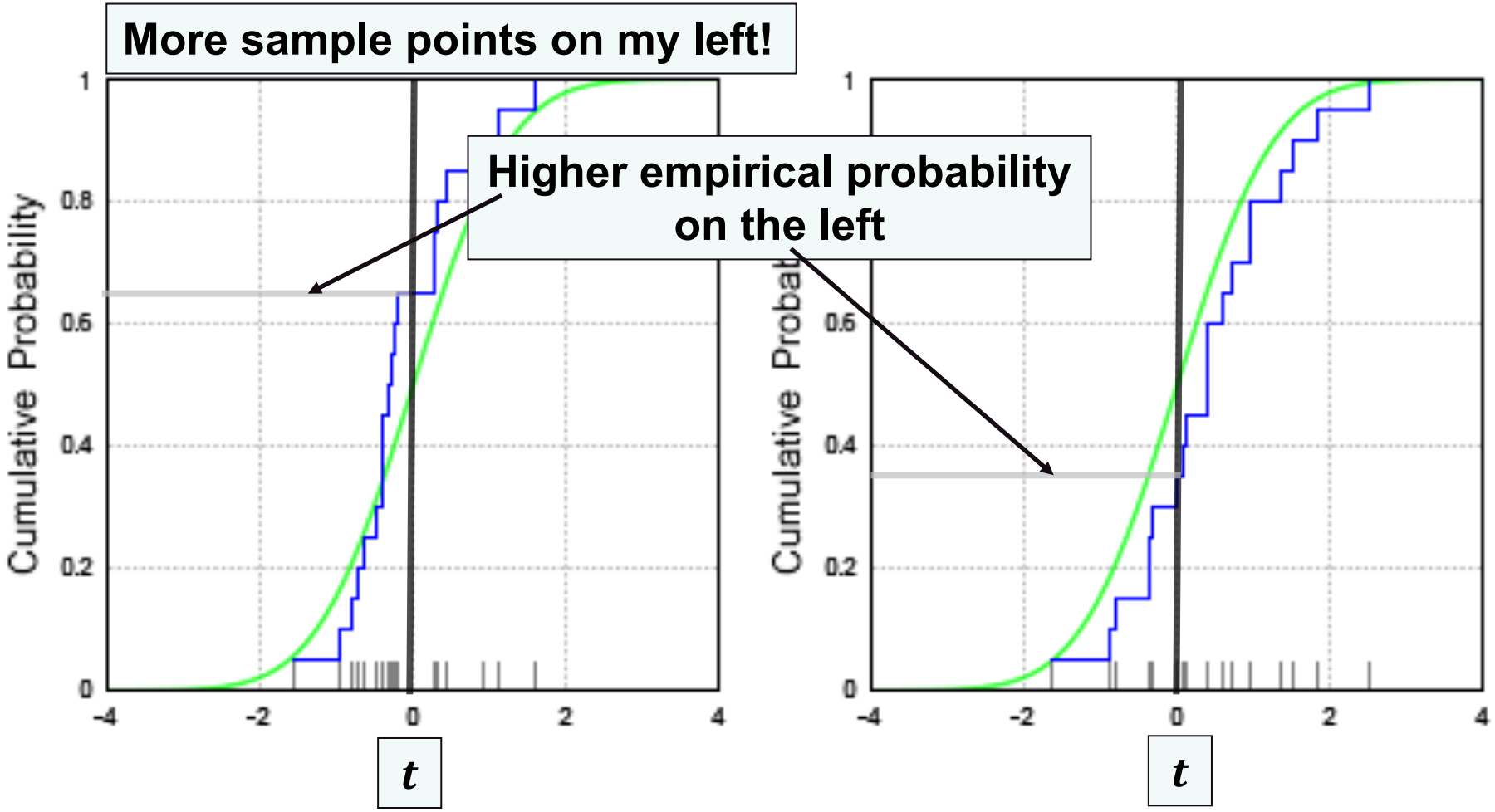
- Taking the running time for example

$$\widehat{F}_T(t\ ;\ A, f, d, v) = \frac{\text{the number of running time values} \leq t}{r} = \frac{1}{r}\sum_{i=1}^{r} \mathbb{1}(t_i(A, f, d, v) \leq t).$$

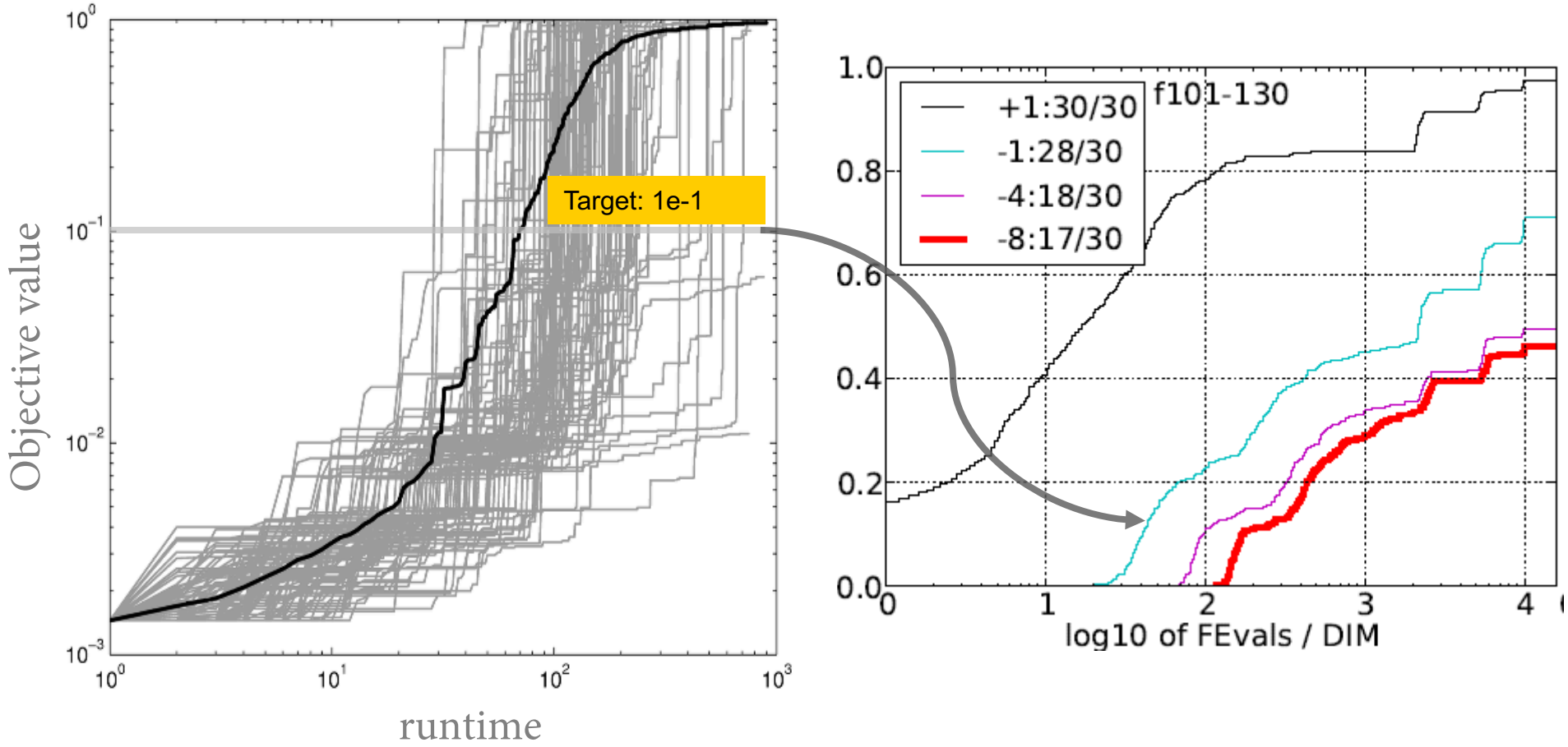
- ECDF converges to the 'true' distribution function

$$\sqrt{r}\left(\widehat{F}_T(t) - F_T(t)\right) \xrightarrow{d} \mathcal{N}(0, F_T(t)(1 - F_T(t)))$$

Empirical Cumulative Distribution Functions

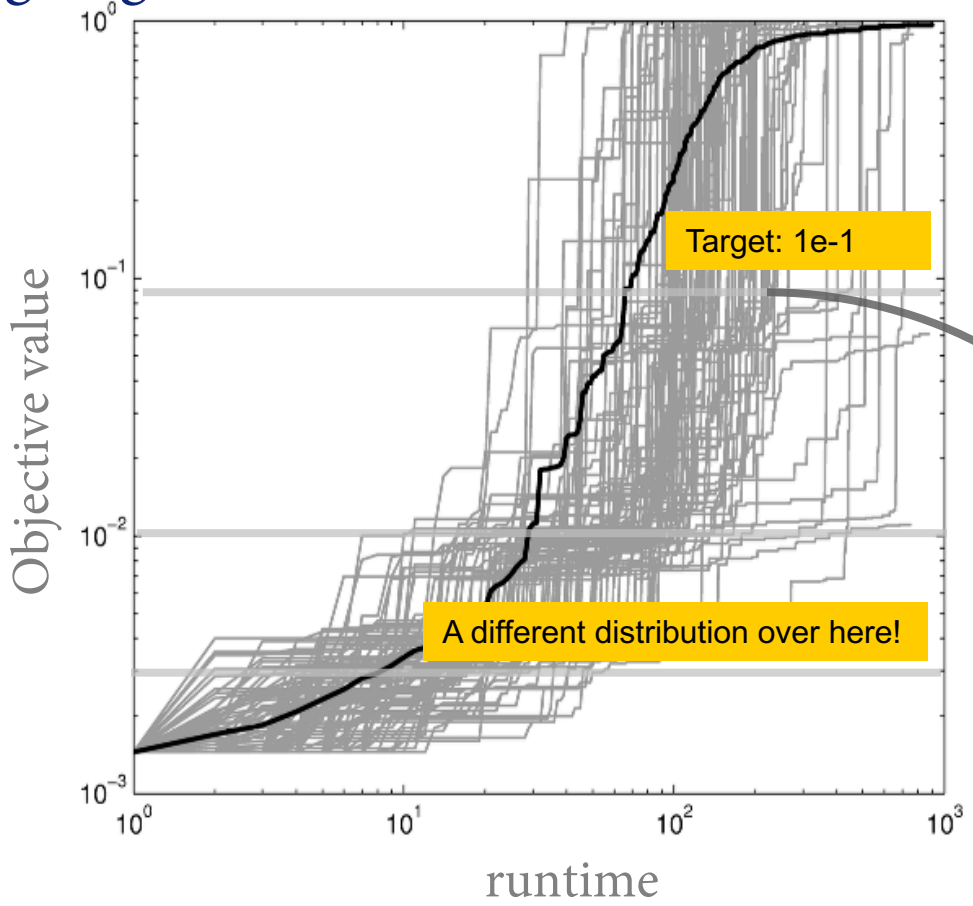


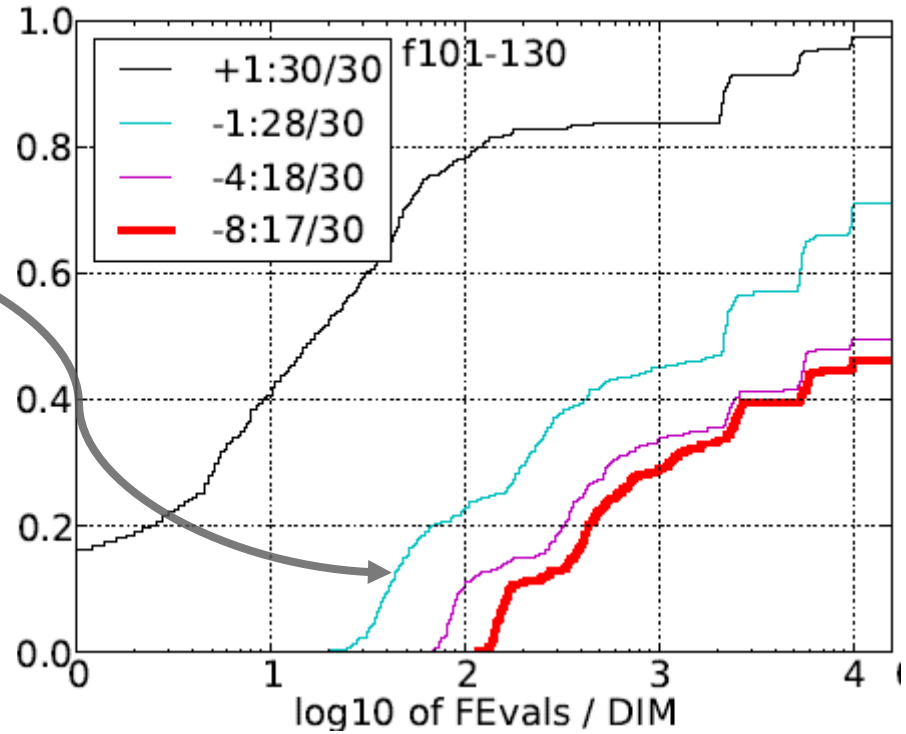
Empirical Cumulative Distribution Functions



Empirical Cumulative Distribution Functions

Varying targets…





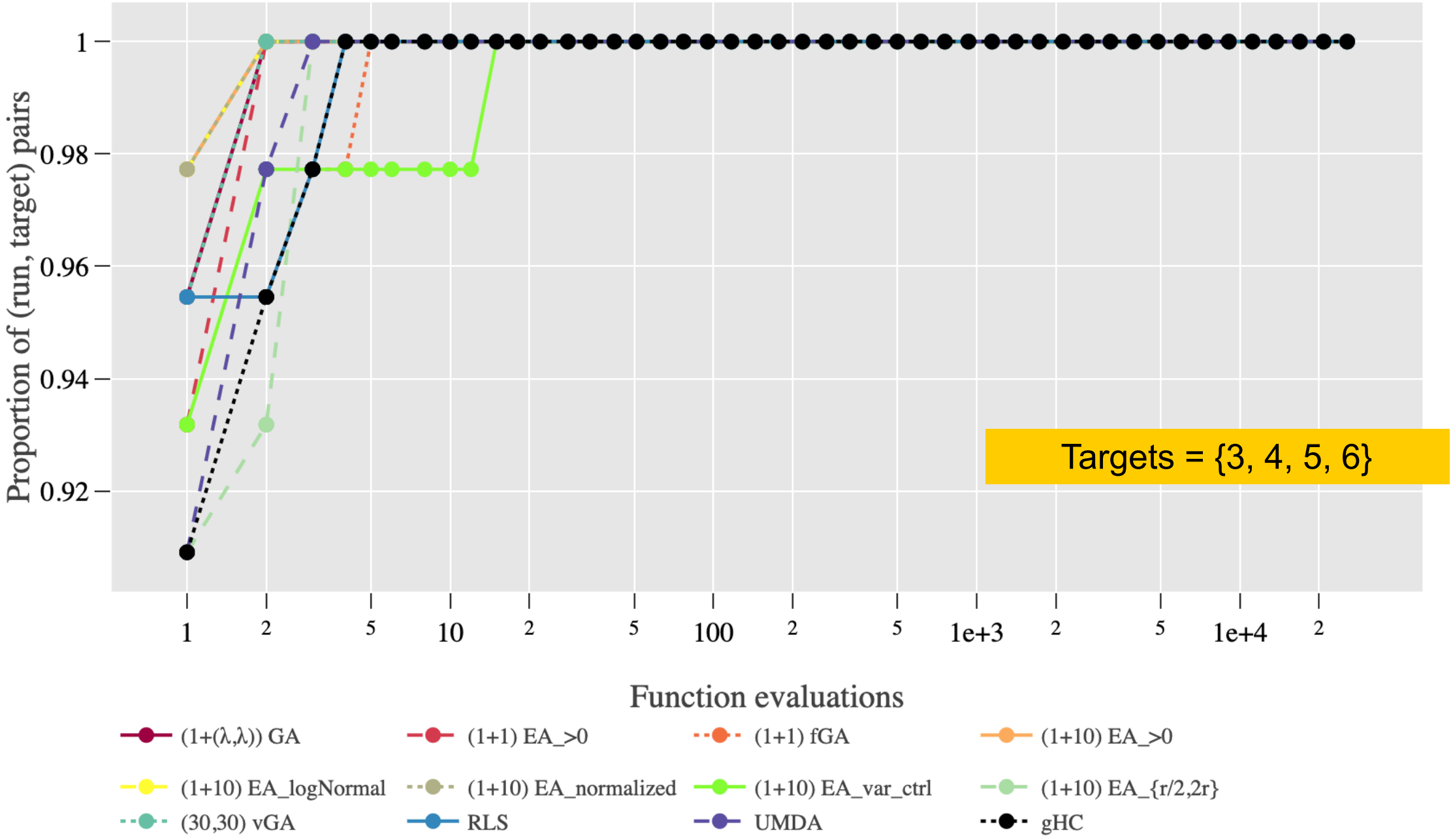
Empirical Cumulative Distribution Functions (ECDFs)

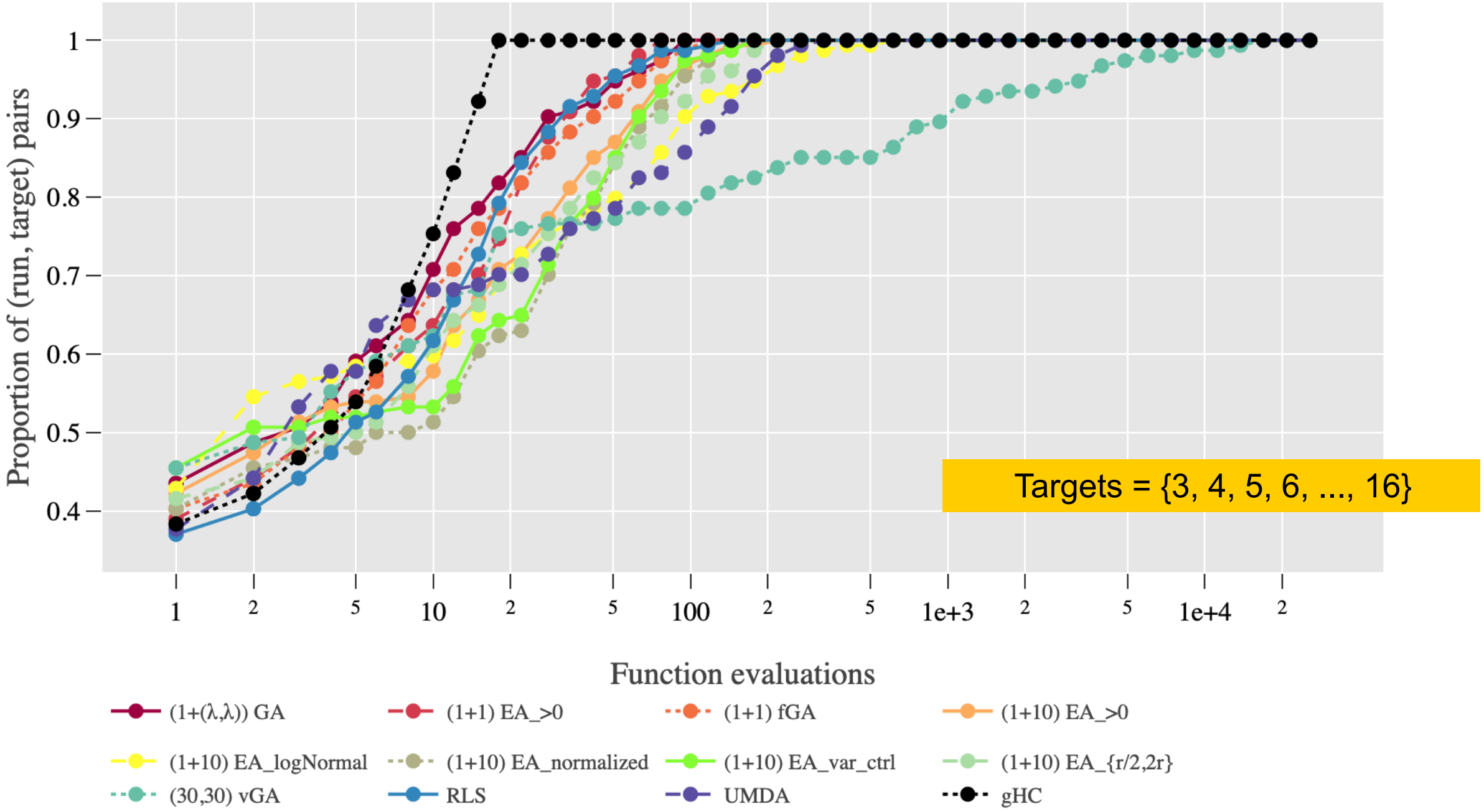
- Aggregate ECDFs
  - Over multiple **targets**: $\mathcal{V} = \{v_1, v_2, \dots\}$

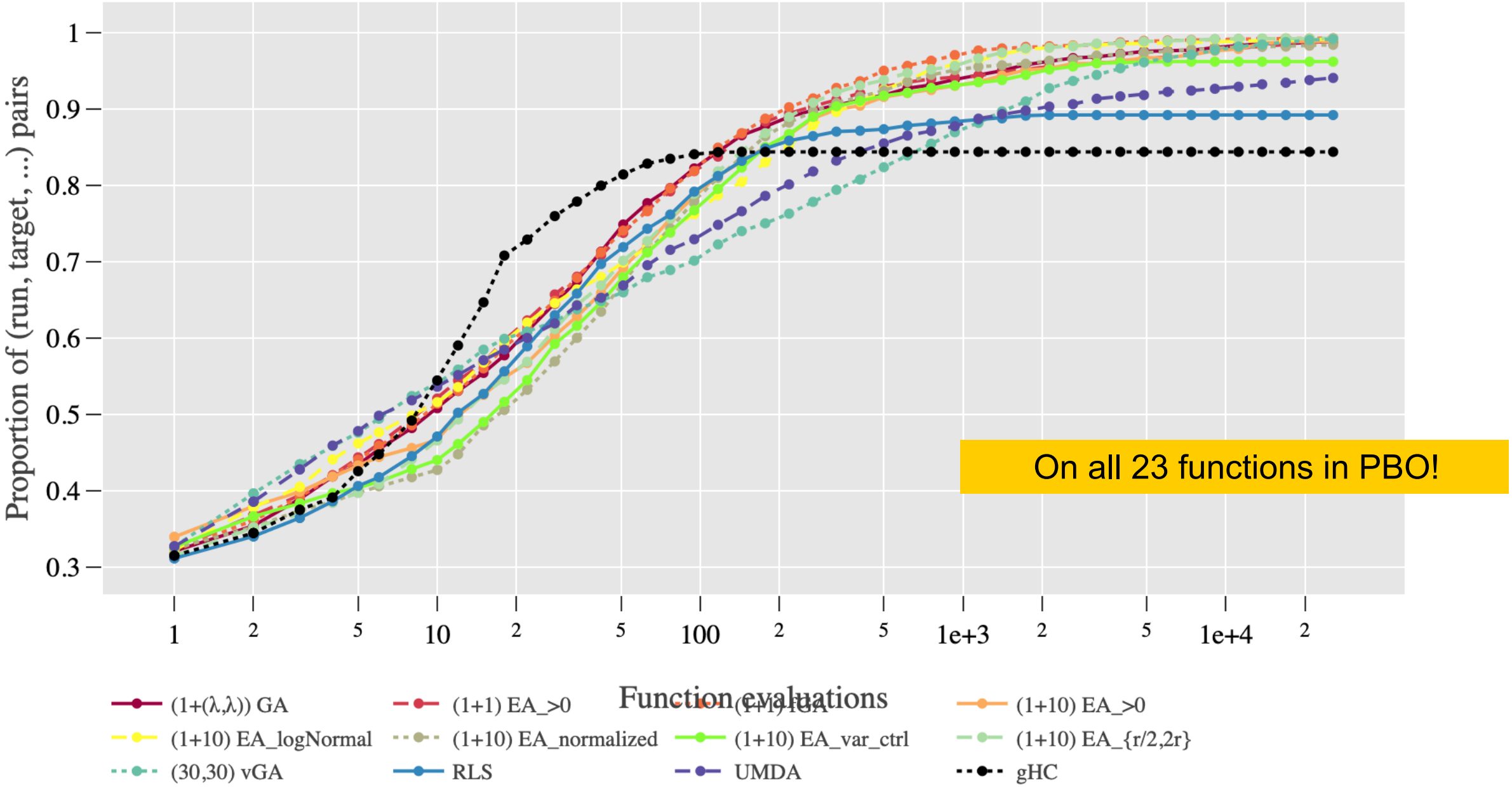
$$\widehat{F}_T(t\ ;\ A, f, d, \mathcal{V}) = \frac{1}{r|\mathcal{V}|} \sum_{v \in \mathcal{V}} \sum_{i=1}^{r} \mathbb{1}(t_i(A, f, d, v) \leq t).$$

  - Over multiple **functions**: $\mathcal{F} = \{f_1, f_2, \dots\}$

$$\widehat{F}_T(t\ ;\ A, \mathcal{F}, d, \mathcal{V}) = \frac{1}{r|\mathcal{V}||\mathcal{F}|} \sum_{f \in \mathcal{F}} \sum_{v \in \mathcal{V}} \sum_{i=1}^{r} \mathbb{1}(t_i(A, f, d, v) \leq t).$$







On all 23 functions in PBO!

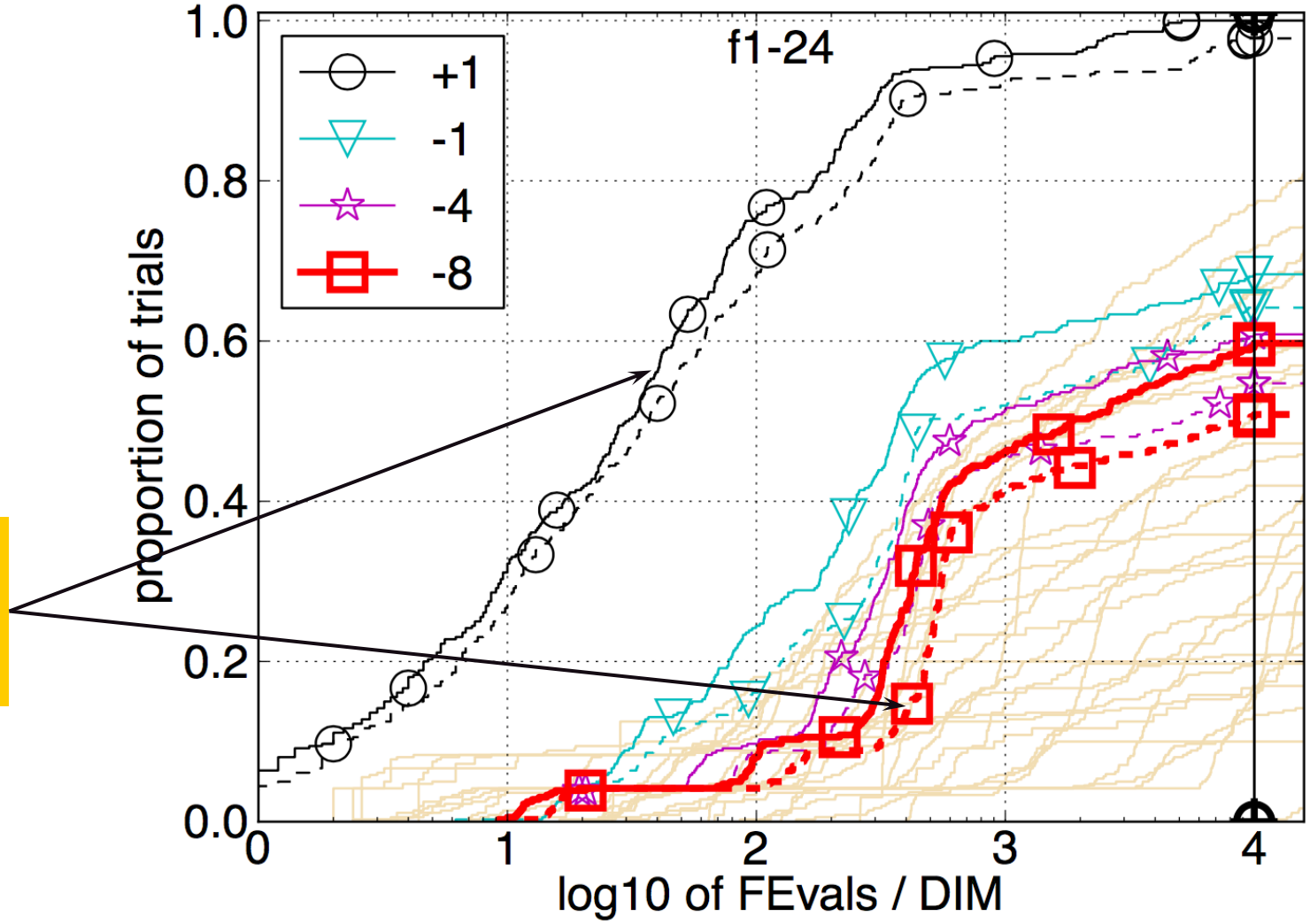
4

STATISTICAL COMPARISON

Hypothesis Testing

- Why?

The difference observed here might caused by random fluctuations!!



Hypothesis Testing

- How? (taking ECDFs for example)

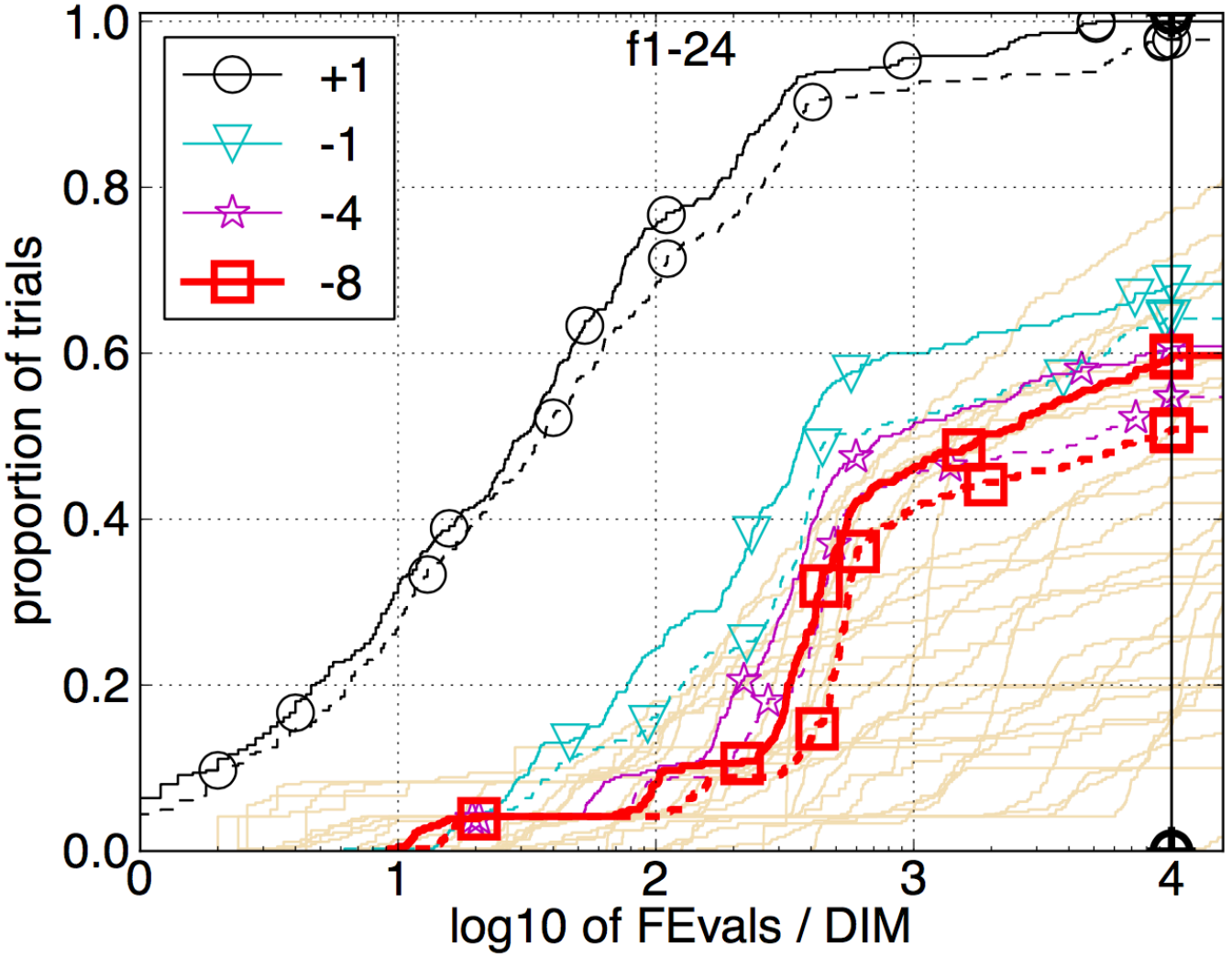
Hypothesis: there is no difference between those two

↓

The observed difference is much larger than the random fluctuation?

↓

Reject (or fail to reject)



Hypothesis Testing: Z-test

- Two samples – assume means unknown, variances known

$$X_1 = \{t_1(f_{\text{target}}), \ldots, t_r(f_{\text{target}})\} \text{ and } X_2 = \{t'_1(f_{\text{target}}), \ldots, t'_r(f_{\text{target}})\}$$

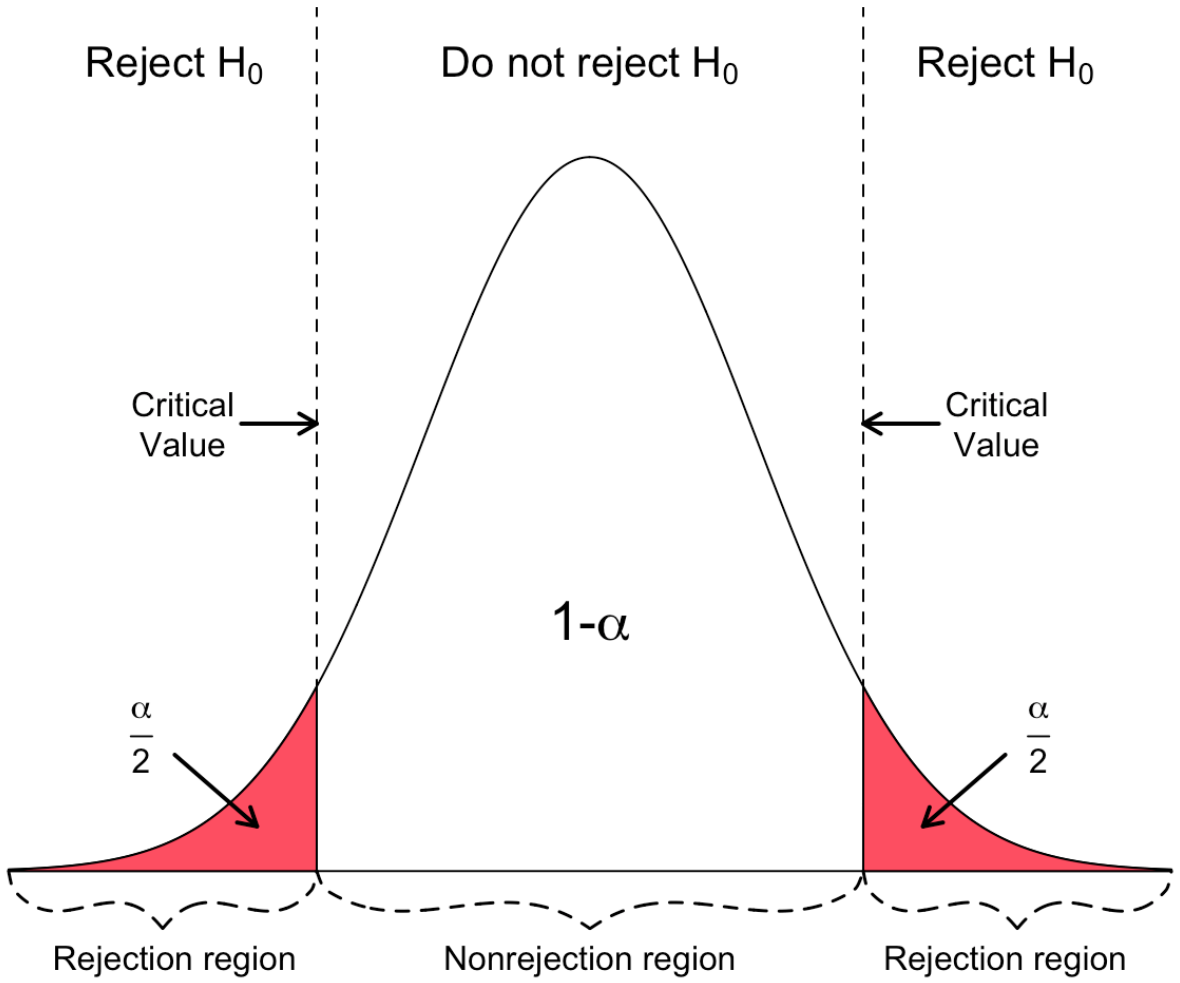
- Formulate hypotheses:
  - H_0: $\bar{X}_1 - \bar{X}_2 = 0$
  - H_a: $\bar{X}_1 - \bar{X}_2 \neq 0$
- Design a test statistic $$Z = \frac{\bar{X}_1 - \bar{X}_2}{\sigma\sqrt{r}}$$
- Set a level of significance $\alpha \in (0, 1)$

Hypothesis Testing: formal setup

- The probability distribution of the test statistic is known under the null hypothesis
  - Probability space $(\Omega, \mathcal{B}, (P_v)),\ v \in \{H_0, H_a\}$
  - Real-valued test statistic $Z : \Omega \to \mathbb{R}$
  - The probability distribution of Z is P_{H_0}, when the null hypothesis is true

Hypothesis Testing: formal setup

- The probability distribution of Z is P_{H_0}, when the null hypothesis is true
- Parametric test – parametric form
- Non-parametric test…



https://userpage.fu-berlin.de/soga/200/2070_hypothesis_tests/20713_The_Critical_Value_and_the_p-Value_Approach_to_Hypothesis_Testing.html

Hypothesis Testing: formal setup

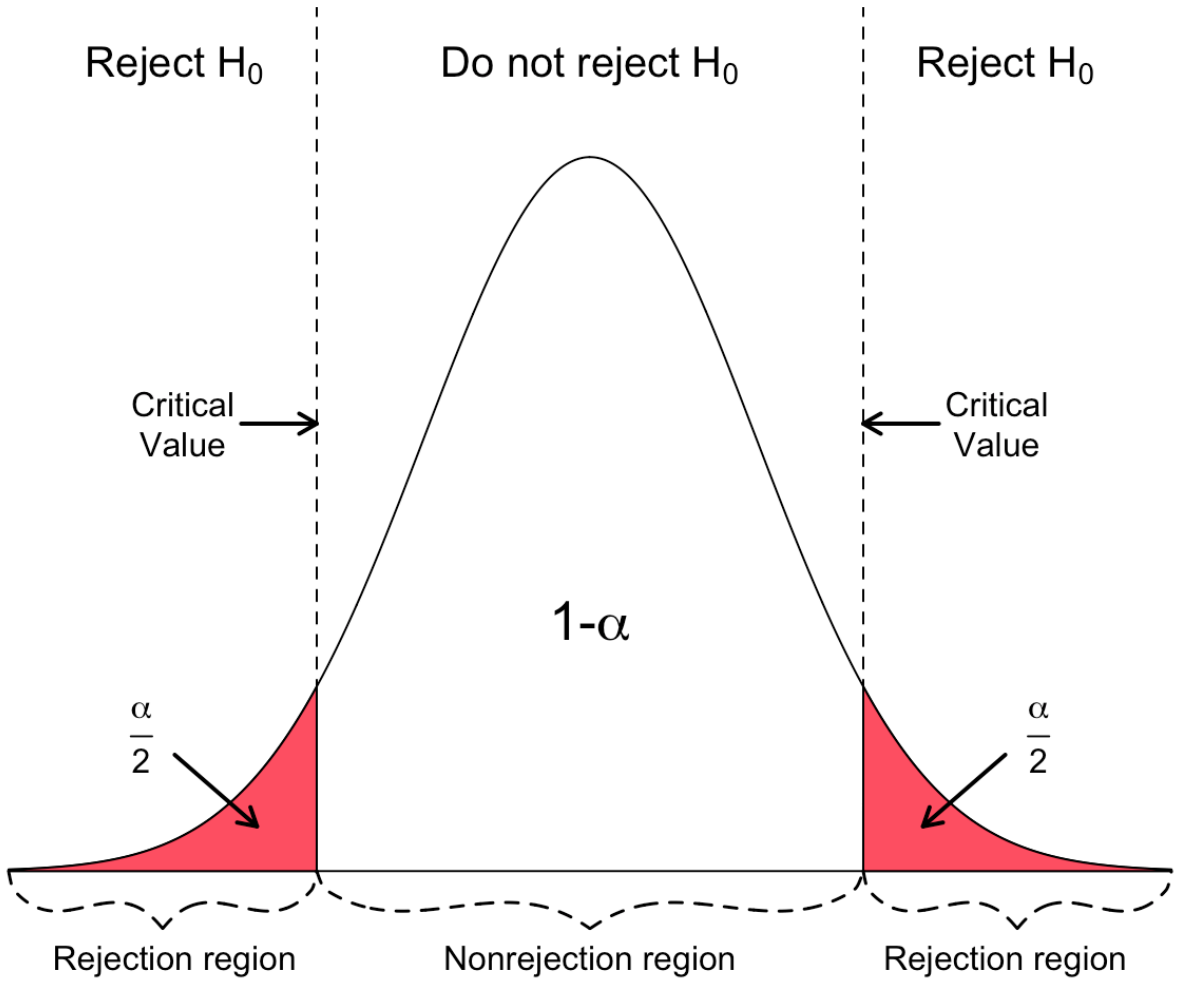
Define the **rejection region**

- Centrality of the distribution
- The region on which a sample is not likely to be drawn $\Gamma_\alpha \subset \mathbb{R}$

- For instance,

$$\Gamma_\alpha = (-\infty, c_{\text{left}}] \cup [c_{\text{right}}, \infty)$$

$$P_{H_0}(z \in \Gamma_\alpha) = \alpha$$



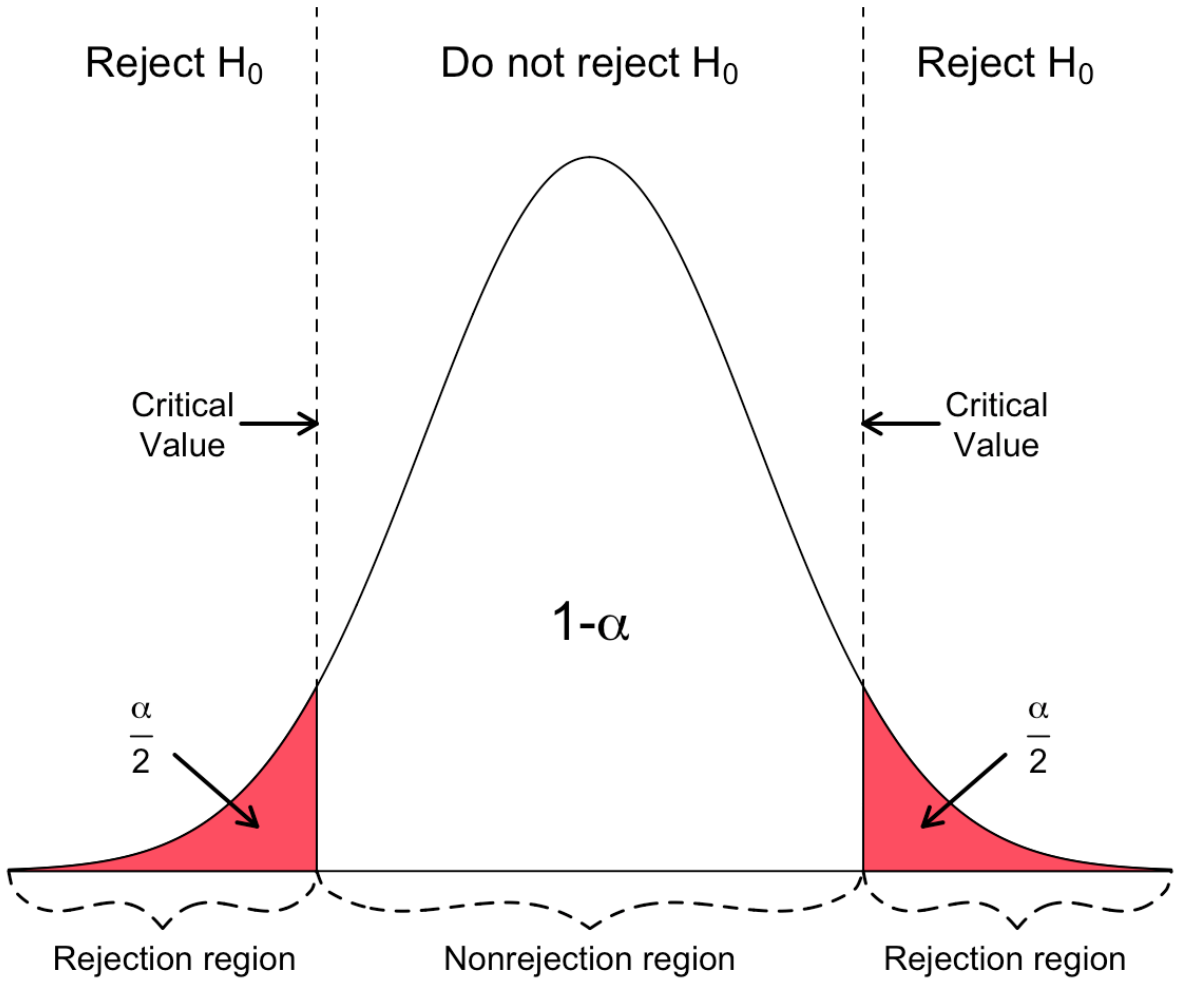
https://userpage.fu-berlin.de/soga/200/2070_hypothesis_tests/20713_The_Critical_Value_and_the_p-Value_Approach_to_Hypothesis_Testing.html

Hypothesis Testing: formal setup

- Reject the null hypothesis if the observed test statistic falls in the rejection region:

$$Z(\bar{X}_1 - \bar{X}_2) \in \Gamma_\alpha$$

- Type-I error/False Positive



https://userpage.fu-berlin.de/soga/200/2070_hypothesis_tests/20713_The_Critical_Value_and_the_p-Value_Approach_to_Hypothesis_Testing.html

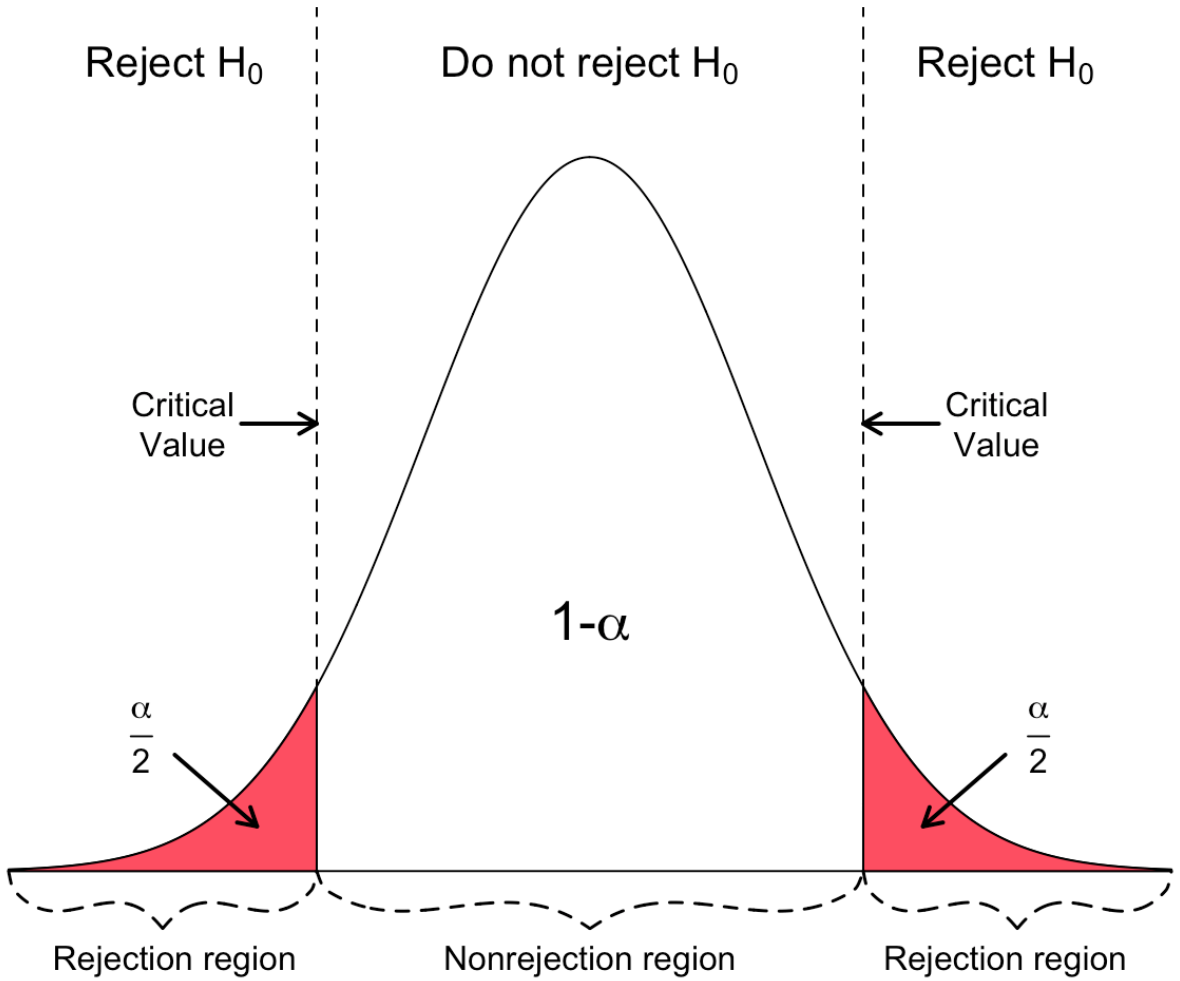
Hypothesis Testing: formal setup

- The probability of the test statistic being more extreme than the observed value, D = $|X_1 - X_2|$

$$p = P_{H_0}(z \leq -D \text{ or } z \geq D)$$

- ***p-value*** is the alternative decision tool to the rejection region

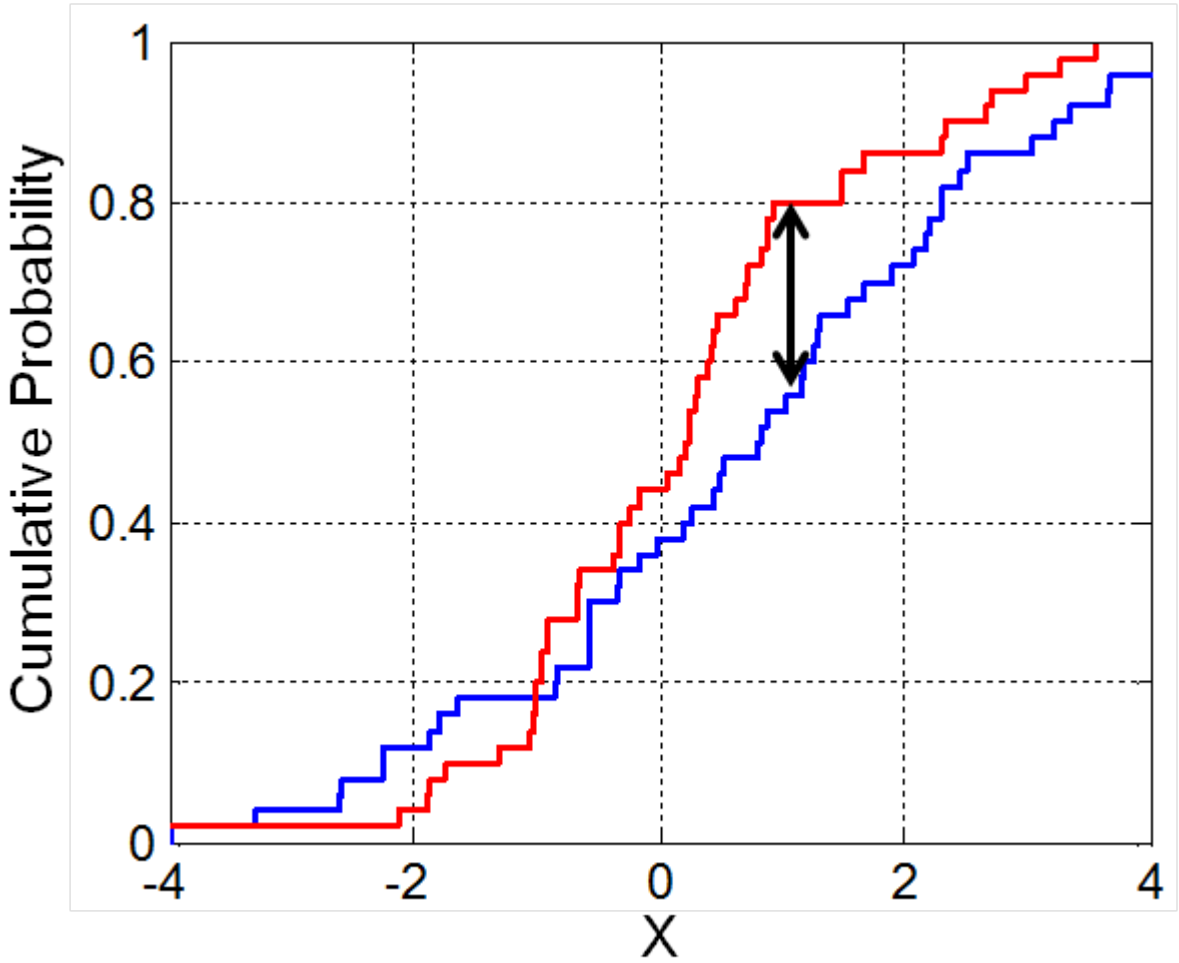
$$p < \alpha \iff Z \in \Gamma_\alpha$$



Komogorov-Smirnov (KS) Test

- Two-sample (two algorithms)
- H_0: two samples are drawn from the same distribution
- Test statistic (on running times)

$$\sup_{t \in [1..B]} |F_T(t) - F_T(t)'|$$



Multiple Algorithms – Multiple testing

- Many-sample $\mathcal{A} = \{A_1, A_2, \ldots\}$
- KS test is not designed for this case…
- Perform KS test pairwisely $\binom{N}{2}$
- Problem?

Multiple testing – Bonferroni correction

- Familywise Error Rate (FWER) should be controlled
  - Probability of making at least one type-I error
  - Let $M = \binom{N}{2}$

$$\mathrm{FWER} = P_{H_0}\left(\bigcup_{i=1}^{M}(p_i \leq \alpha)\right) \leq \sum_{i=1}^{M} P_{H_0}(p_i \leq \alpha) = M\alpha$$

  - Solution- Bonferroni correction: to use α/M

Statistical Procedures for Many Samples

Number of Random Samples	Independent Samples	Dependent Samples	Hypothesis Test Involving
Two	+ Wilcoxon rank sum test (a, b, d, e) - Squared ranks test (c, d) + Klotz test (c, d) + Kolmogorov-Smirnov test (b, d, e) + Cramér-von Mises test (d) - Randomization test or Fisher's permutation test (a) + Bootstrap (a, c, d) + Jacknife (a, c, d) + Siegel-Tukey test (c) - Moses test (c)	- Wilcoxon matched-pairs signed-ranks test (a, b, d, e) + Binomial sign test (b, d, e) - Randomization test for matched pairs or Fisher's matched samples test (a)	a. Means (medians) b. Confidence interval c. Variances
Several	+ Median test (a) + Kruskal-Wallis test (a, d, e) + van der Waerden test (a, d, e) - Squared ranks test (c, d) + Birnbaum-Hall test (d) + k-sample Smirnov test (d, e)	+ Friedman two-way analysis of variance by ranks (a)	d. Identical populations or distributions e. Ordering