Advanced Optimization

Lecture/Exercise 5: Critically Looking at Data

January 8, 2019

Master AIC

Université Paris-Saclay, Orsay, France

Anne Auger
INRIA Saclay – Ile-de-France

Dimo Brockhoff
INRIA Saclay – Ile-de-France

Course Overview

	Date		Topic
1	Tue, 20.11.2018	Dimo	Randomized Algorithms for Discrete Problems
2	Tue, 27.11.2018	Dimo	Exercise: The Travelling Salesperson Problem
3	Tue, 4.12.2018	Dimo	Evolutionary Multiobjective Optimization I
4	Tue, 11.12.2018	Dimo	Evolutionary Multiobjective Optimization II
	vacation		
5	Tue, 8.1.2019	Dimo	Looking at Data
6	Tue, 15.1.2019 23h59 CET	Anne	Continuous Optimization I deadline abstract submission
7	Tue, 29.1.2019 23h59 CET	Anne	Continuous Optimization II deadline slides submission*
	Tue, 12.2.2019		oral presentations (individual time slots)

all lectures from 14h00 till 17h15

here in E107 in Nov/Dec and in E105 in January

* best by email to me

Organization Oral Exams

	Tuesday, Feb 12, 2019	
9:30am	Martin	
10am	Robin	
10:30am	Hao	
11am	Malik	
11:30am	Jiaxin	
12am	Samuel	
12:30pm	Nouredine	
1:30pm	Mirwaisse	
2:00pm	Luca	
2:30pm	Alexandre	
3pm	Luc	
3:30pm	Cedric	
4pm	Antoine	

Critically Looking at Data

why?

(Some) Main Research Goals

- novelty
- repeatability
- applicability

A Possible Way to Learn Science...

- ...is to look at how others do it ☺
- ...is to critically ask whether what others are doing is the right thing
- ...is to get your hands dirty and tackle a difficult open question yourself (most time consuming part probably)
- ...is to actively review papers

Paper Review:
"Dynamic Search in Fireworks Algorithm"

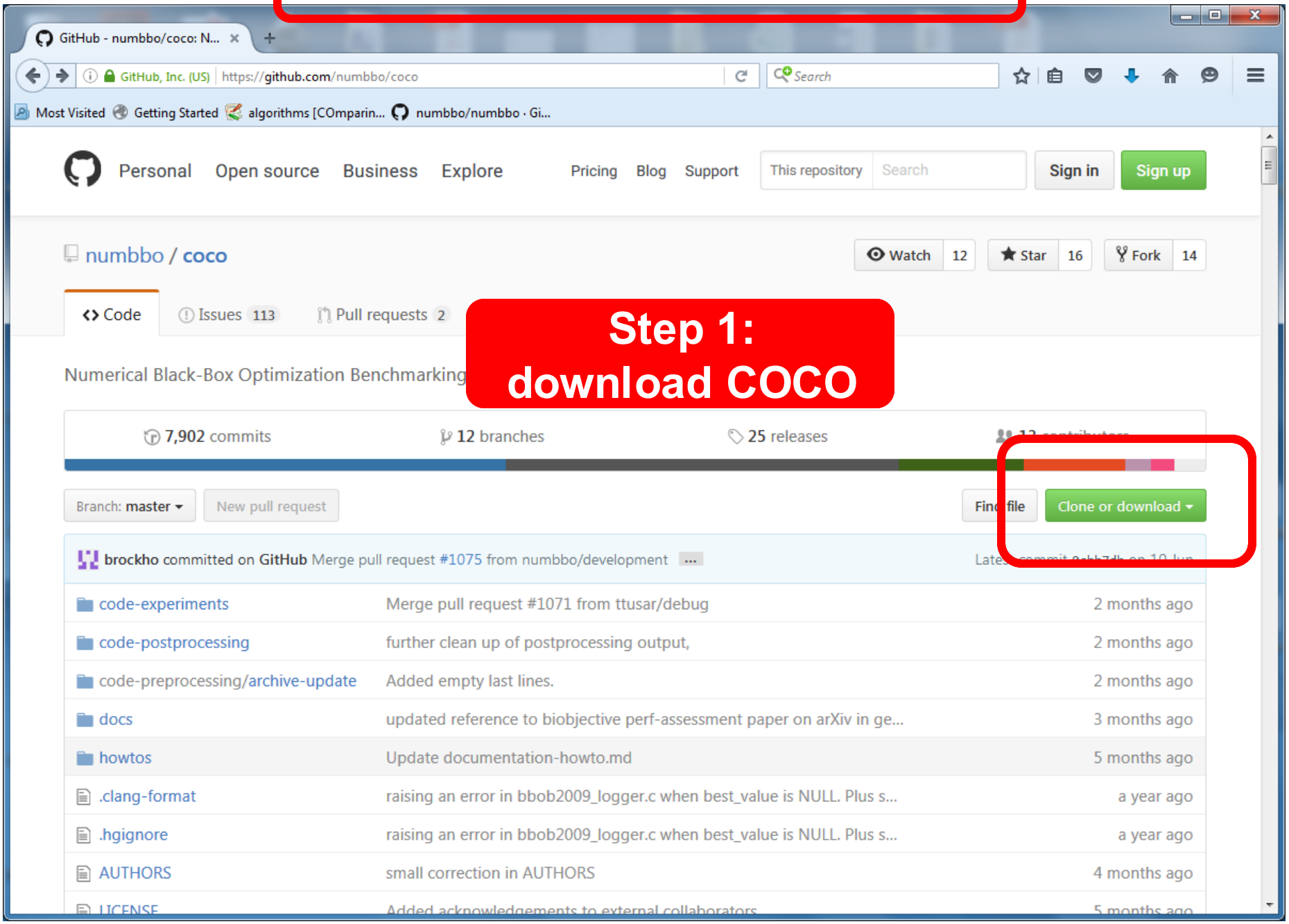
Dynamic Search in Fireworks Algorithm

- Read Sec. V
  - Sec. V.B less important
  - read rather only until V.A and look at the results
- Do not care about what the algorithms are actually doing
- Questions:
  - What is well done in the experimental comparison?
  - What can be improved?
  - What shall be done and is not done?
  - Concretely: Mark in Tables I and II what you find remarkable

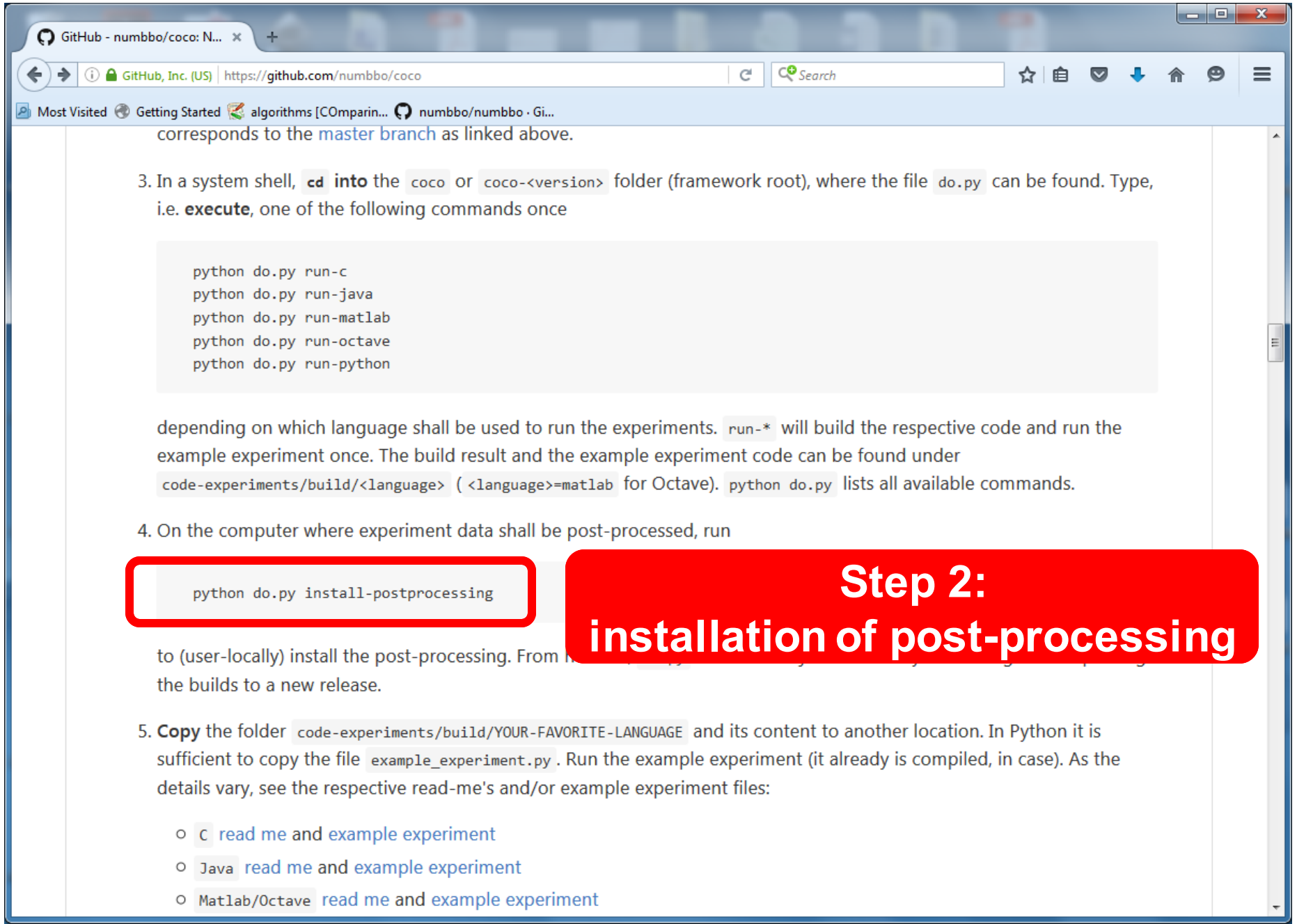
wrt. repeatability, interpretability, clarity, ...

Exercise: Looking at COCO Data

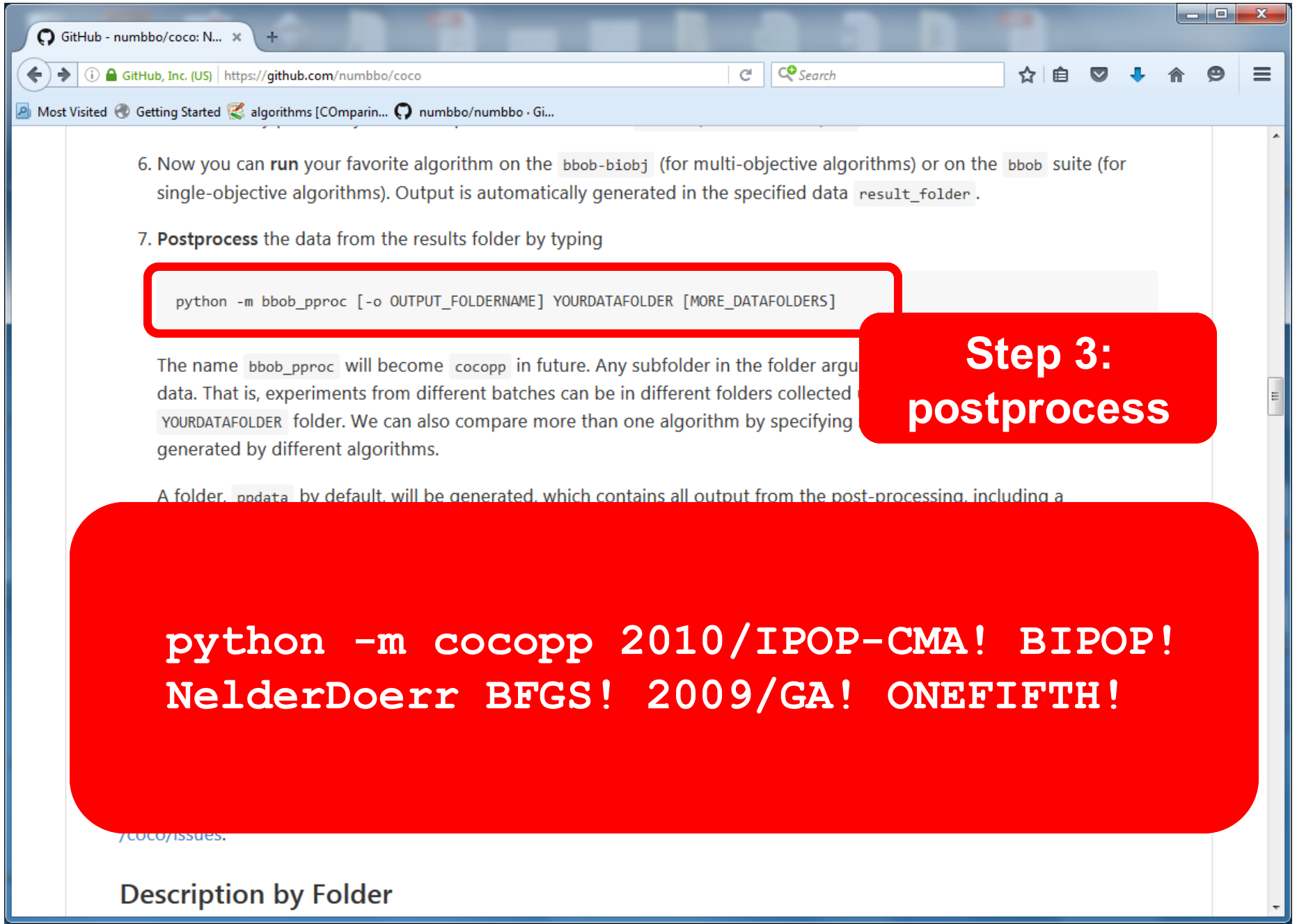
https://github.com/numbbo/coco



https://github.com/numbbo/coco

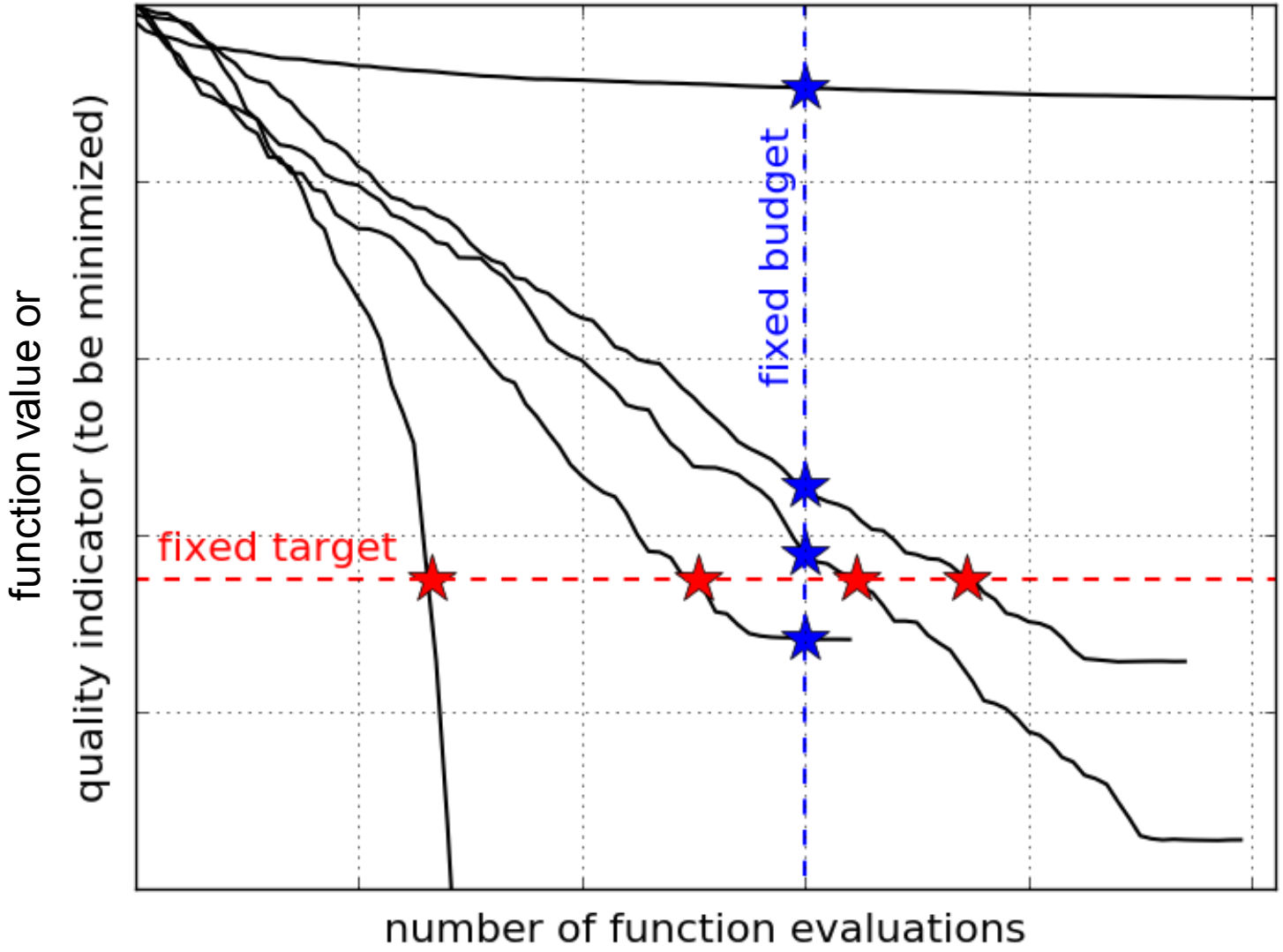


https://github.com/numbbo/coco



Measuring Performance Empirically

convergence graphs is all we have to start with...

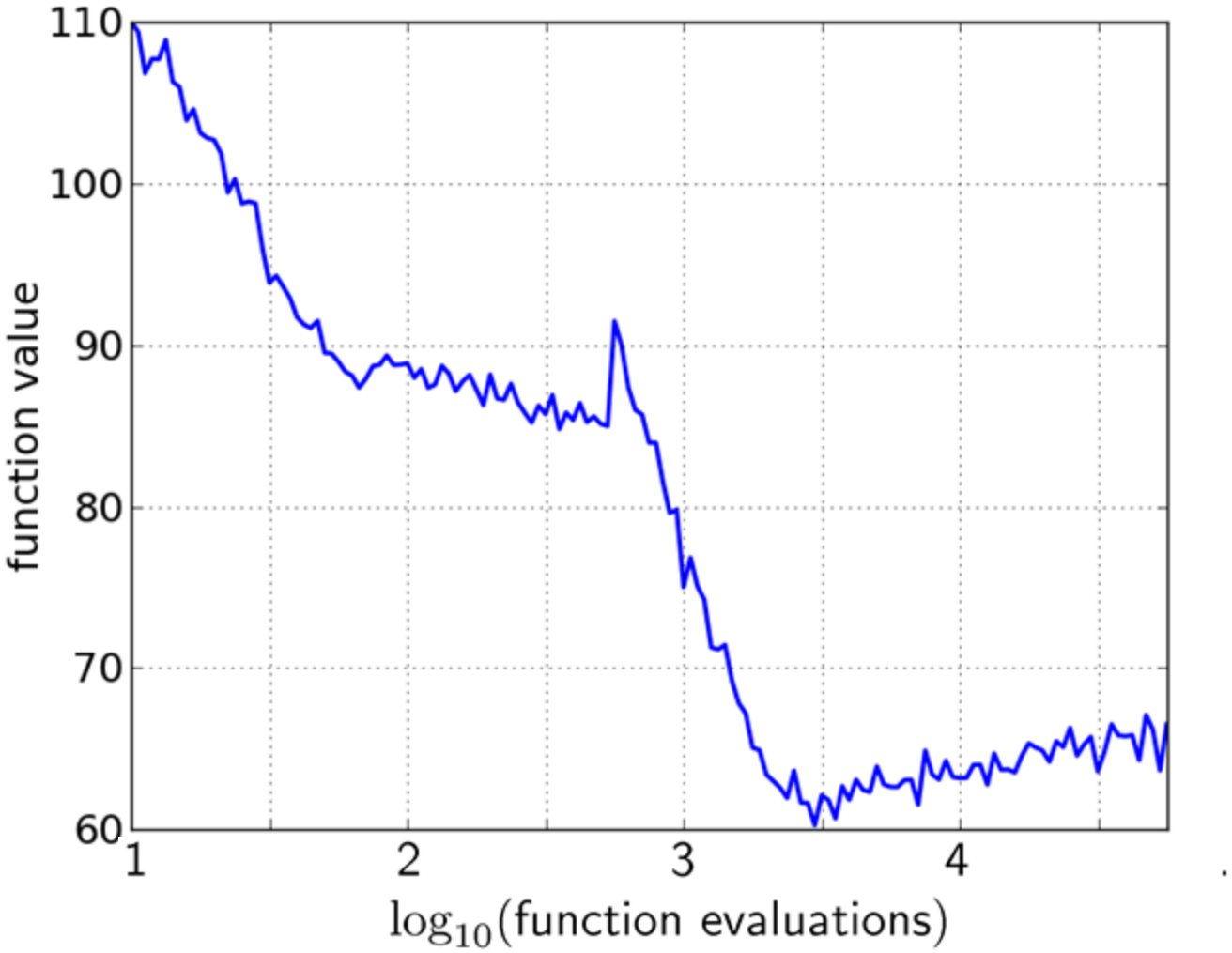


ECDF:

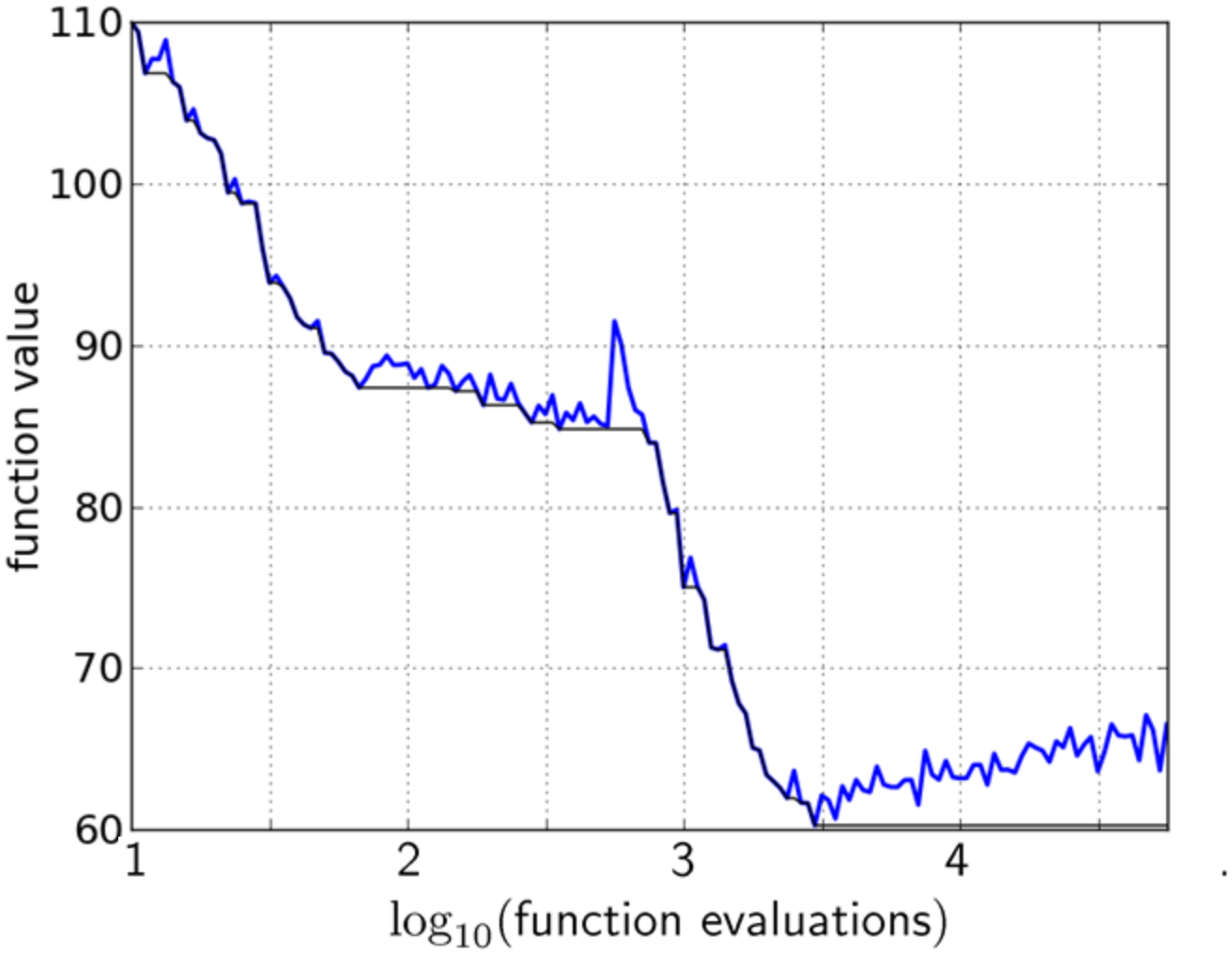
Empirical Cumulative Distribution Function of the Runtime

[aka data profile]

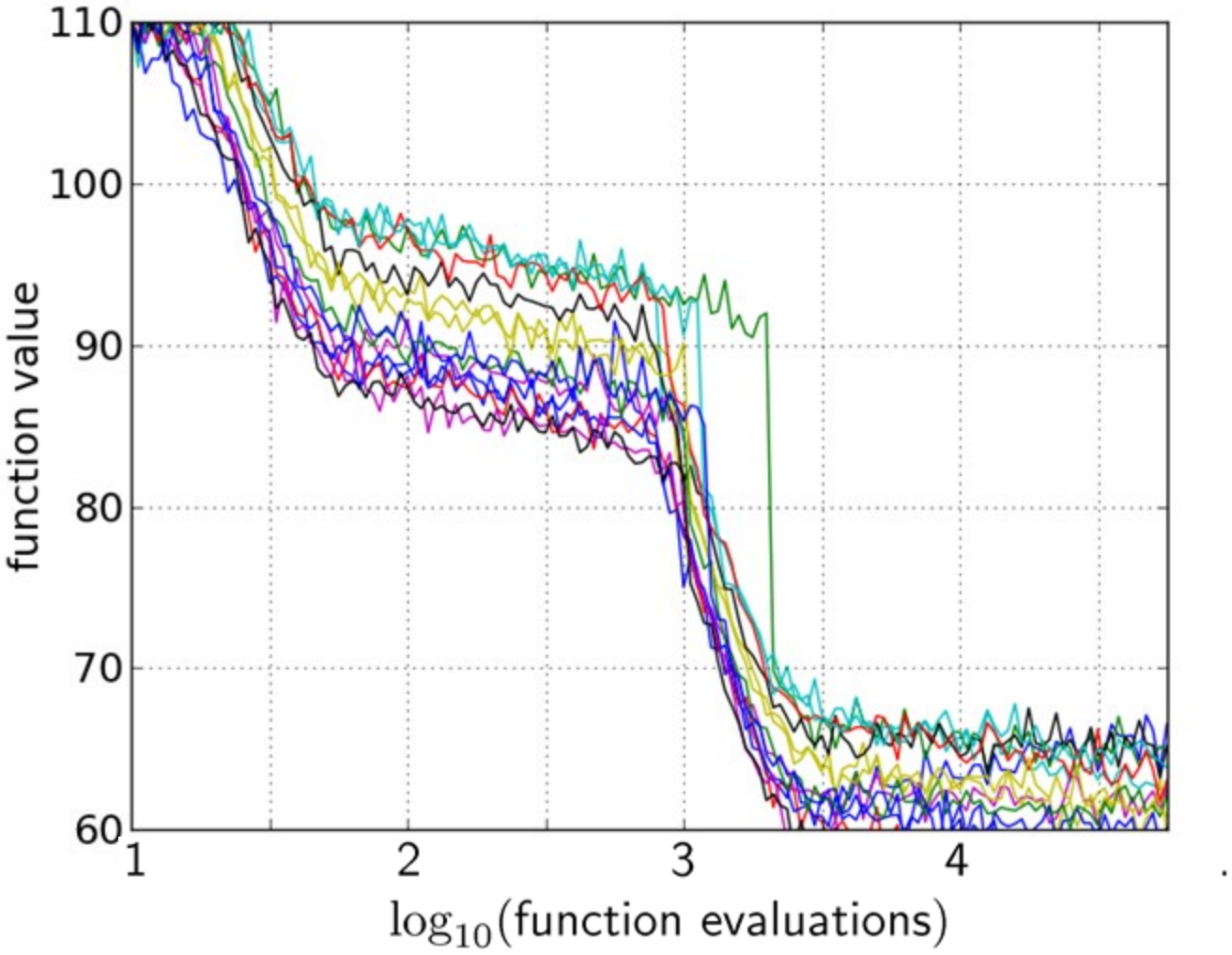
A Convergence Graph



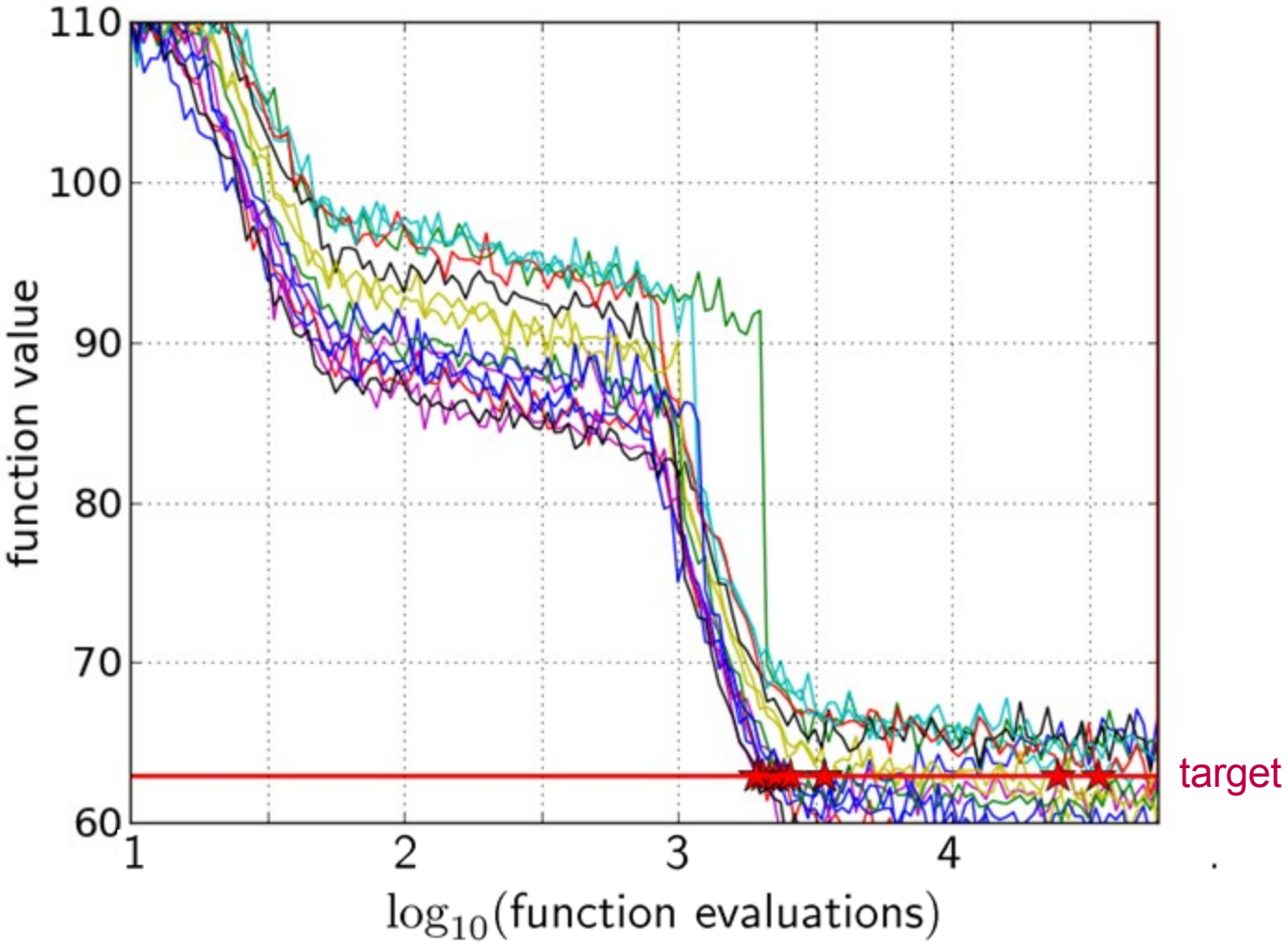
First Hitting Time is Monotonous



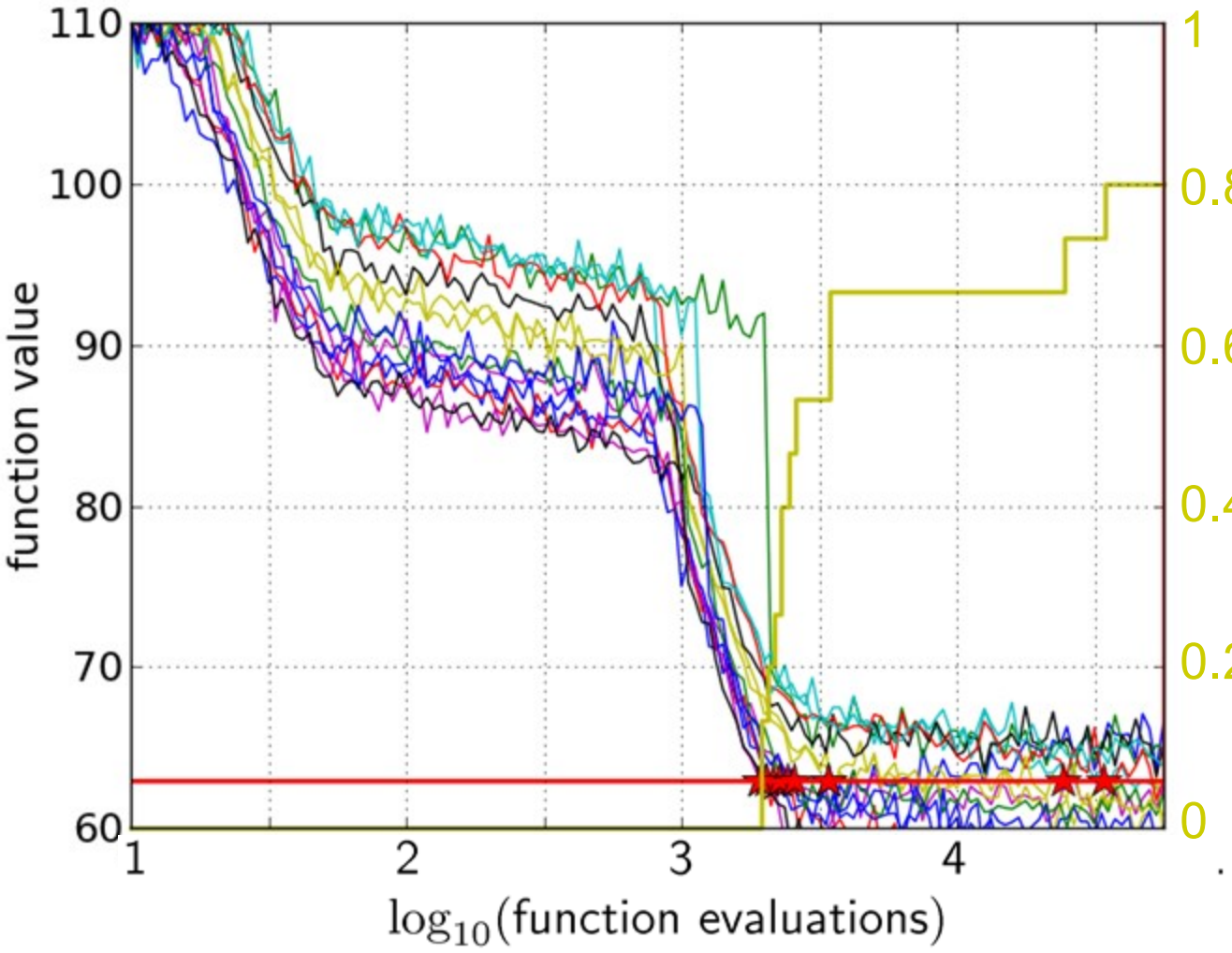
15 Runs


110
100
90
80
70
60
function value
1
2
3
4
$\log_{10}$(function evaluations)

15 Runs ≤ 15 Runtime Data Points

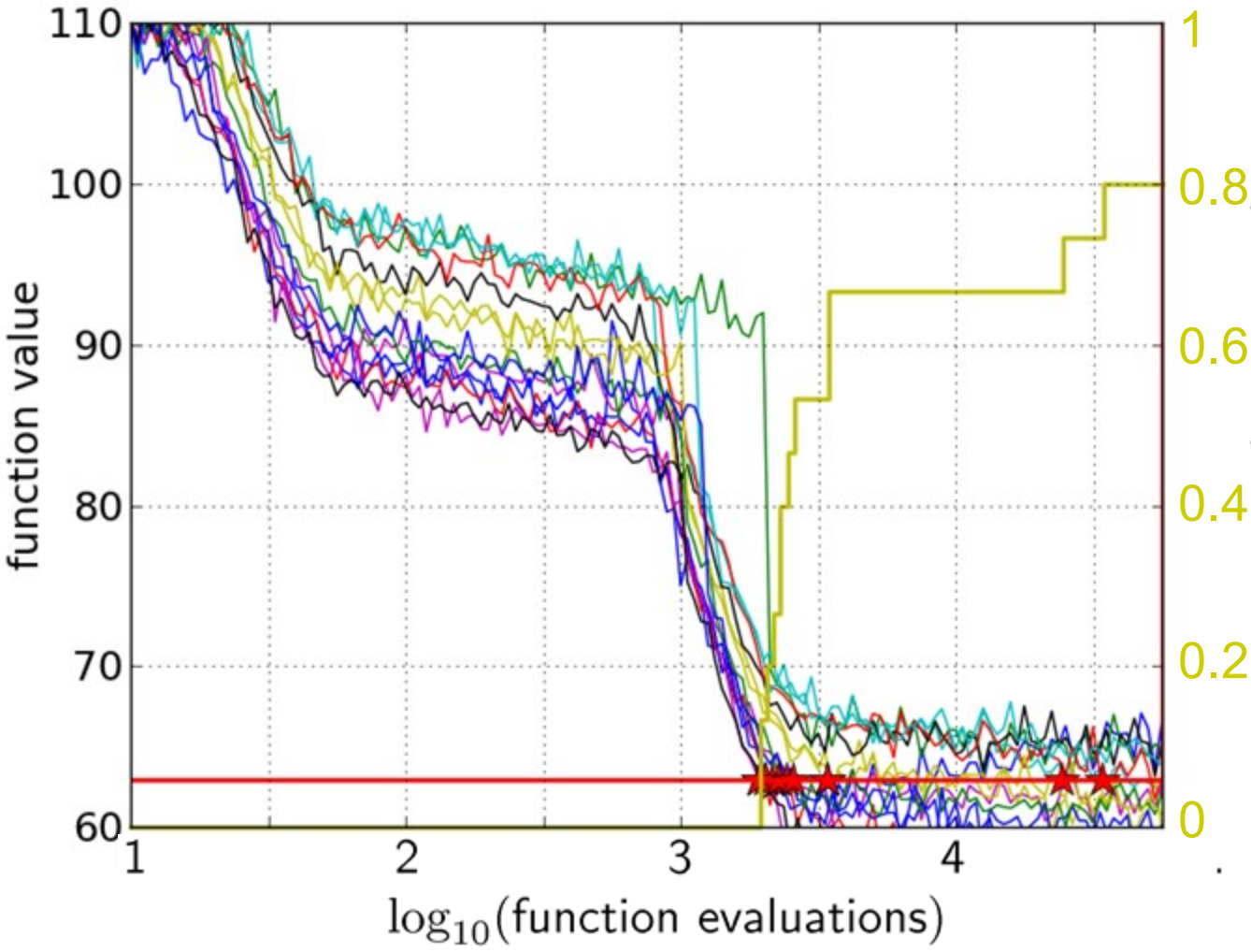


Empirical Cumulative Distribution



- the ECDF of run lengths to reach the target
- has for each data point a vertical step of constant size
- displays for each x-value (budget) the count of observations to the left (first hitting times)

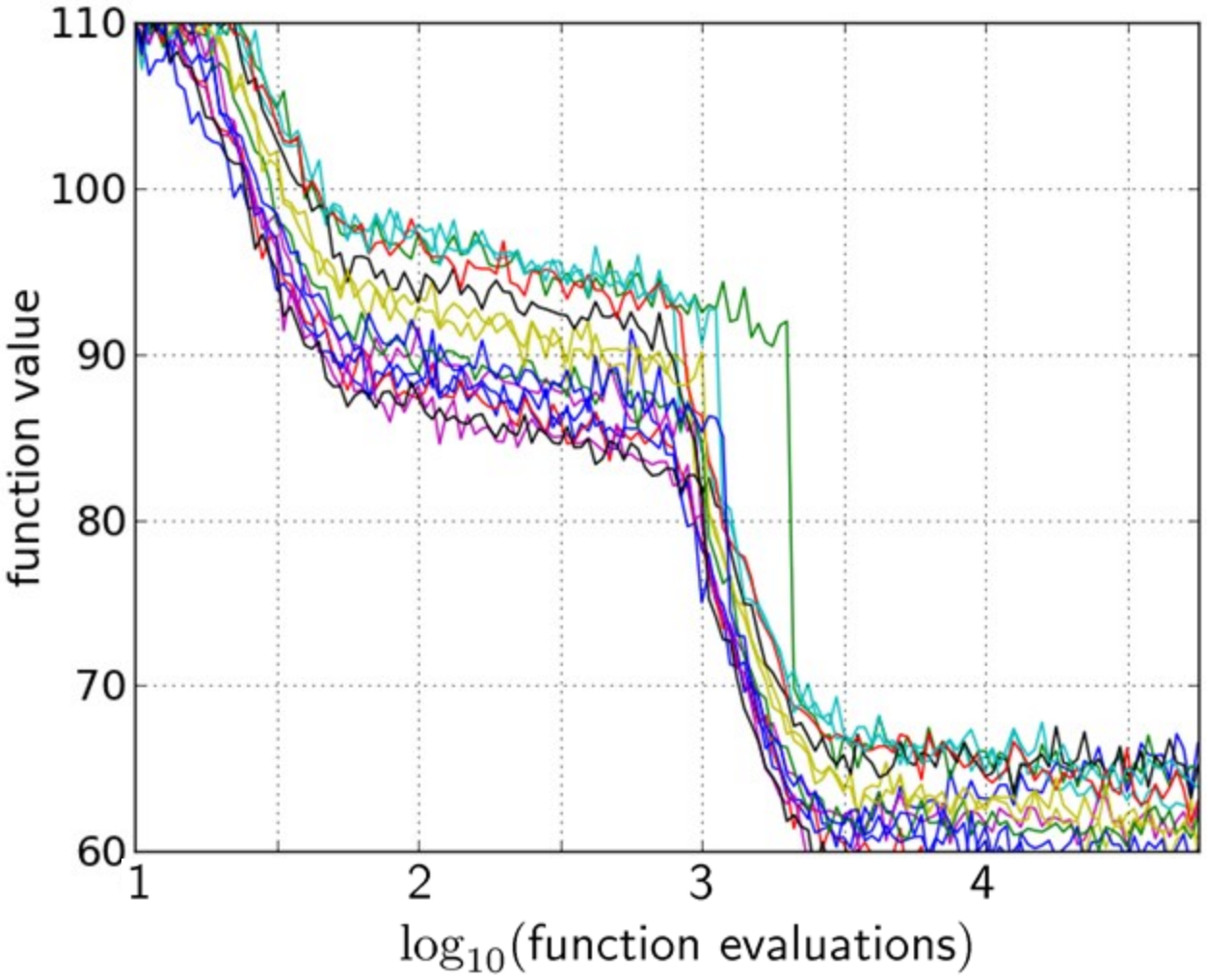
Empirical Cumulative Distribution



interpretations possible:

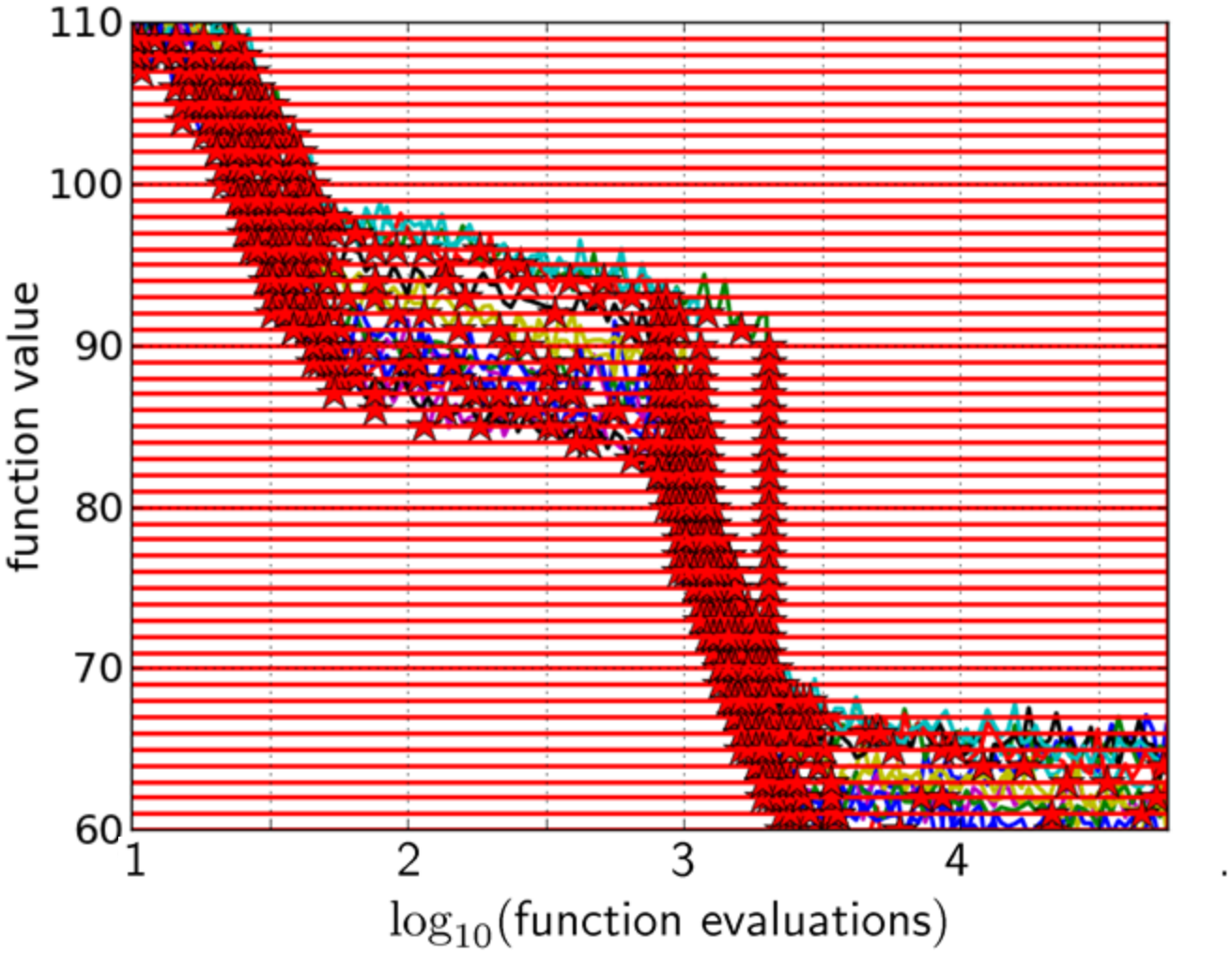
- 80% of the runs reached the target
- e.g. 60% of the runs need between 2000 and 4000 evaluations

Aggregation



15 runs

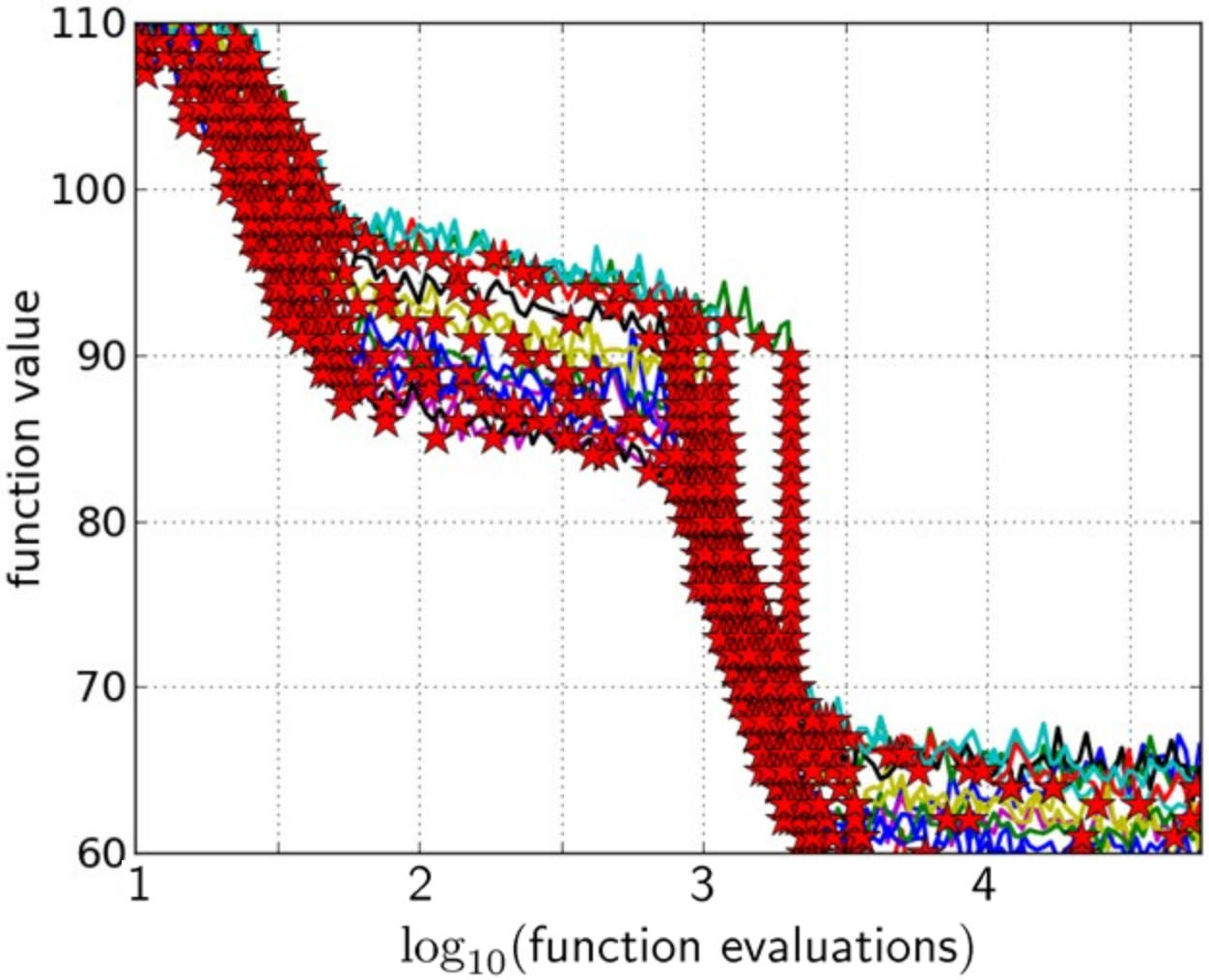
Aggregation



15 runs

50 targets

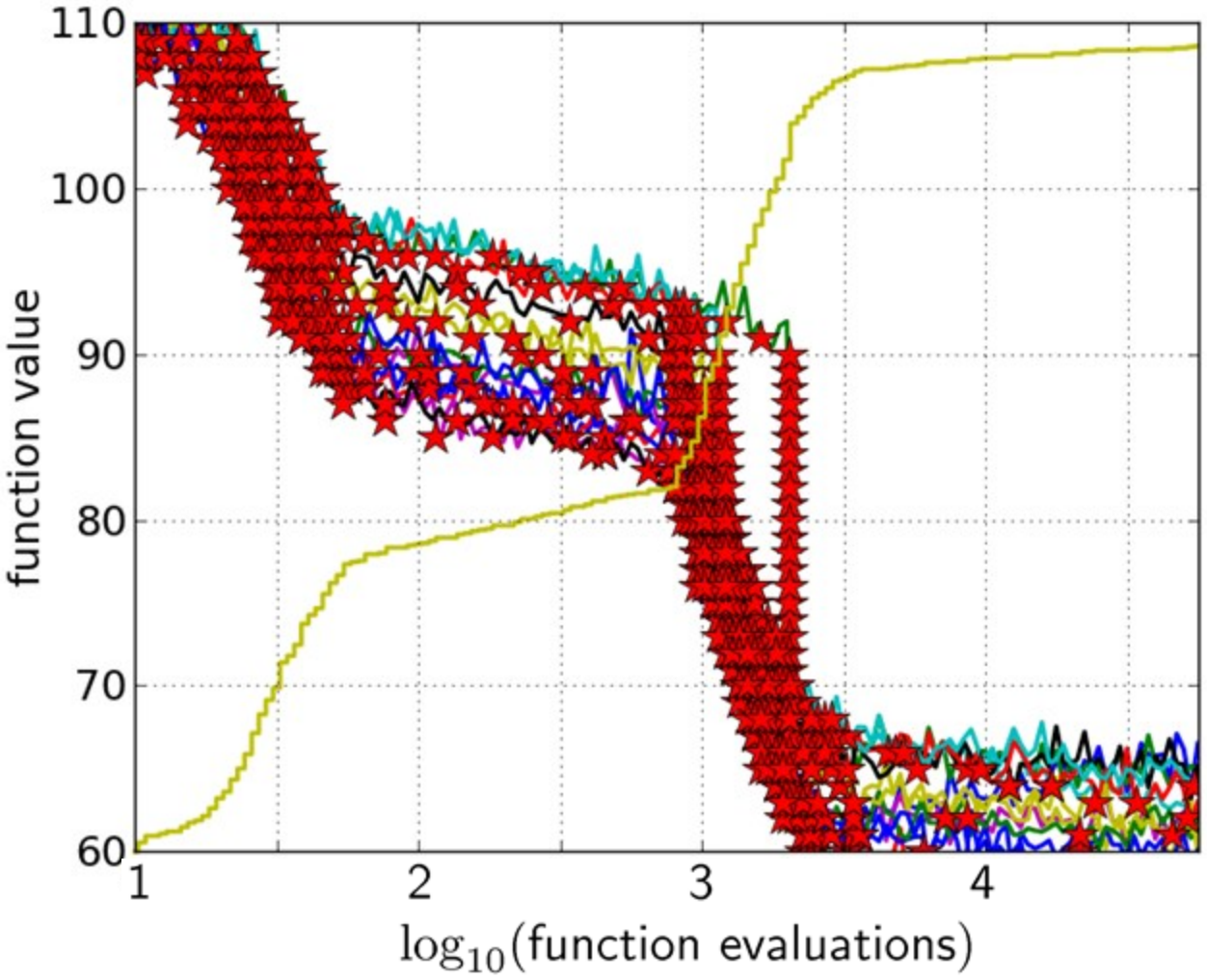
Aggregation



15 runs

50 targets

Aggregation

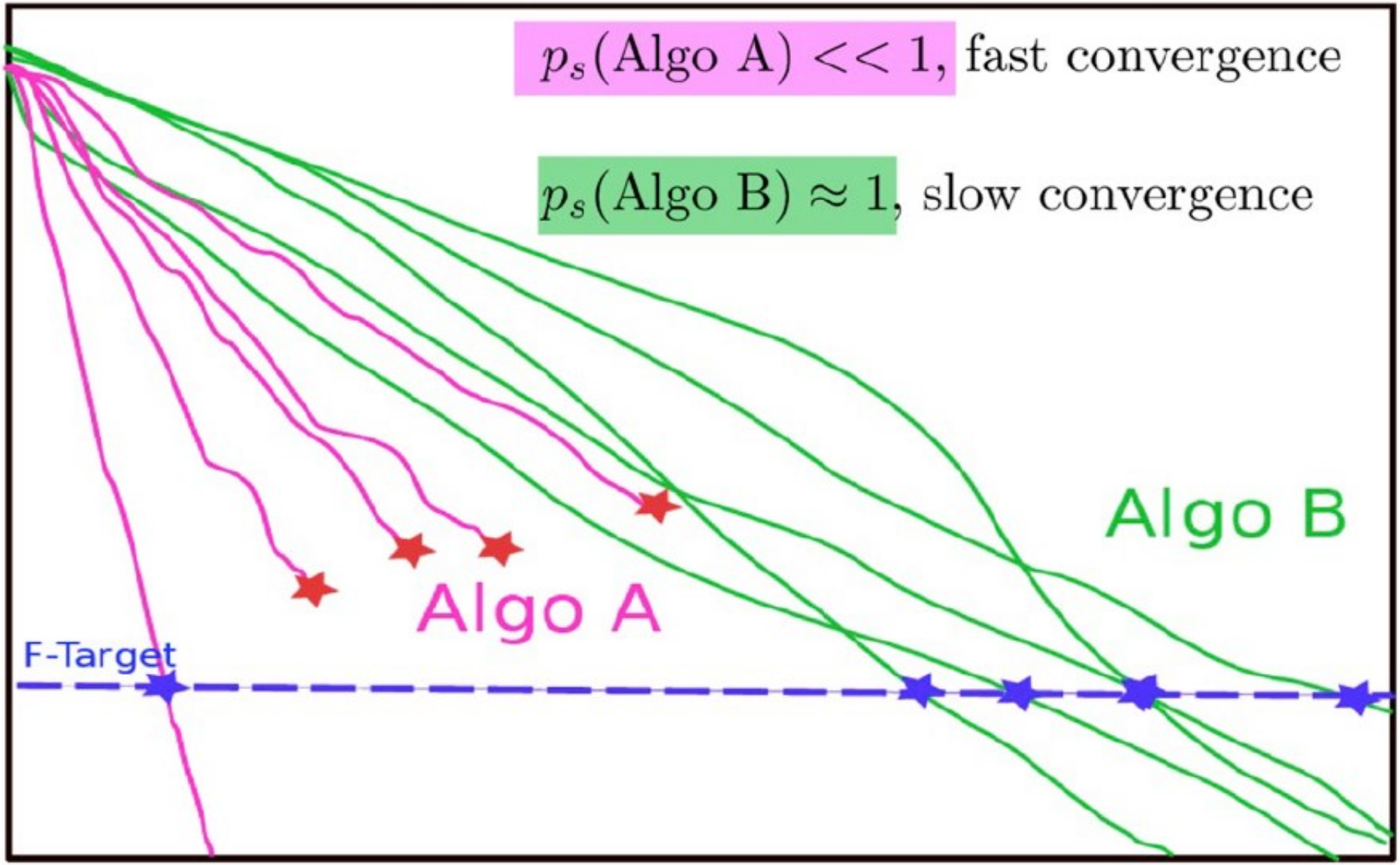


15 runs

50 targets

ECDF with 750 steps

Fixed-target: Measuring Runtime



Fixed-target: Measuring Runtime

- Algo Restart A:

p_s(Algo Restart A) = 1

- Algo Restart B:

p_s(Algo Restart B) = 1

Fixed-target: Measuring Runtime

- Expected running time of the restarted algorithm:

$$E[RT^r] = \frac{1-p_s}{p_s} E\left[RT_{unsuccessful}\right] + E[RT_{successful}]$$

- Estimator average running time (aRT):

$$\hat{p_s} = \frac{\#\text{successes}}{\#\text{runs}}$$

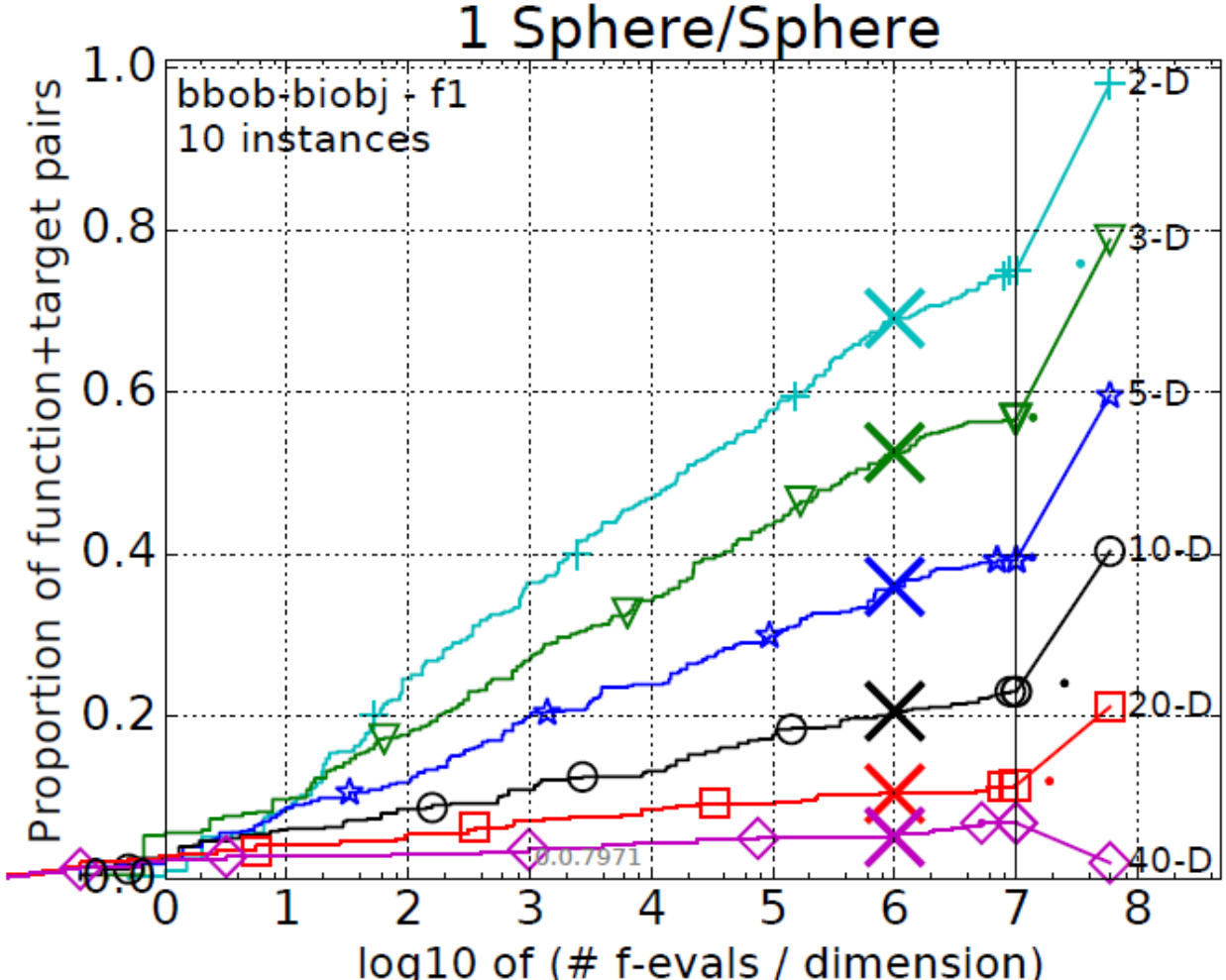
$$\widehat{RT_{unsucc}} = \text{Average evals of unsuccessful runs}$$

$$\widehat{RT_{succ}} = \text{Average evals of successful runs}$$

$$aRT = \frac{\text{total \#evals}}{\#\text{successes}}$$

ECDFs with Simulated Restarts

What we typically plot are ECDFs of the simulated restarted algorithms:



The single-objective BBOB functions

The bbob Testbed

- 24 functions in 5 groups:

1 Separable Functions	
f1	Sphere Function
f2	Ellipsoidal Function
f3	Rastrigin Function
f4	Büche-Rastrigin Function
f5	Linear Slope
2 Functions with low or moderate conditioning	
f6	Attractive Sector Function
f7	Step Ellipsoidal Function
f8	Rosenbrock Function, original
f9	Rosenbrock Function, rotated
3 Functions with high conditioning and unimodal	
f10	Ellipsoidal Function
f11	Discus Function
f12	Bent Cigar Function
f13	Sharp Ridge Function
f14	Different Powers Function

4 Multi-modal functions with adequate global structure	
f15	Rastrigin Function
f16	Weierstrass Function
f17	Schaffers F7 Function
f18	Schaffers F7 Functions, moderately ill-conditioned
f19	Composite Griewank-Rosenbrock Function F8F2
5 Multi-modal functions with weak global structure	
f20	Schwefel Function
f21	Gallagher's Gaussian 101-me Peaks Function
f22	Gallagher's Gaussian 21-hi Peaks Function
f23	Katsuura Function
f24	Lunacek bi-Rastrigin Function

- 6 dimensions: 2, 3, 5, 10, 20, (40 optional)

Notion of Instances

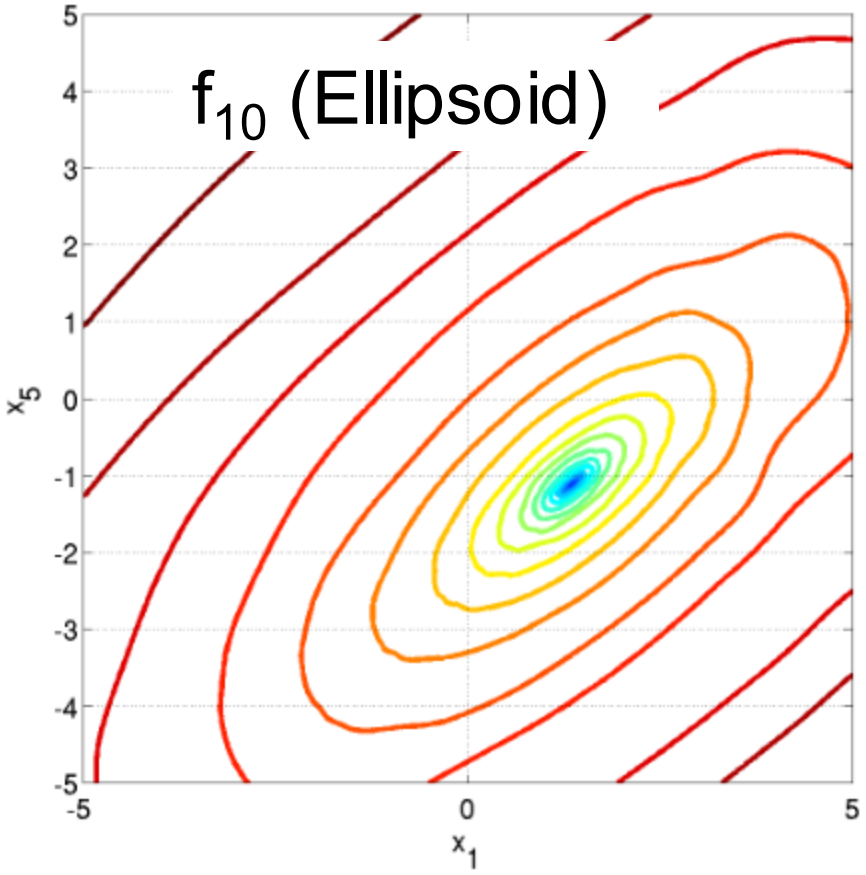
- All COCO problems come in form of instances
  - e.g. as translated/rotated versions of the same function
- Prescribed instances typically change from year to year
  - avoid overfitting
  - 5 instances are always kept the same

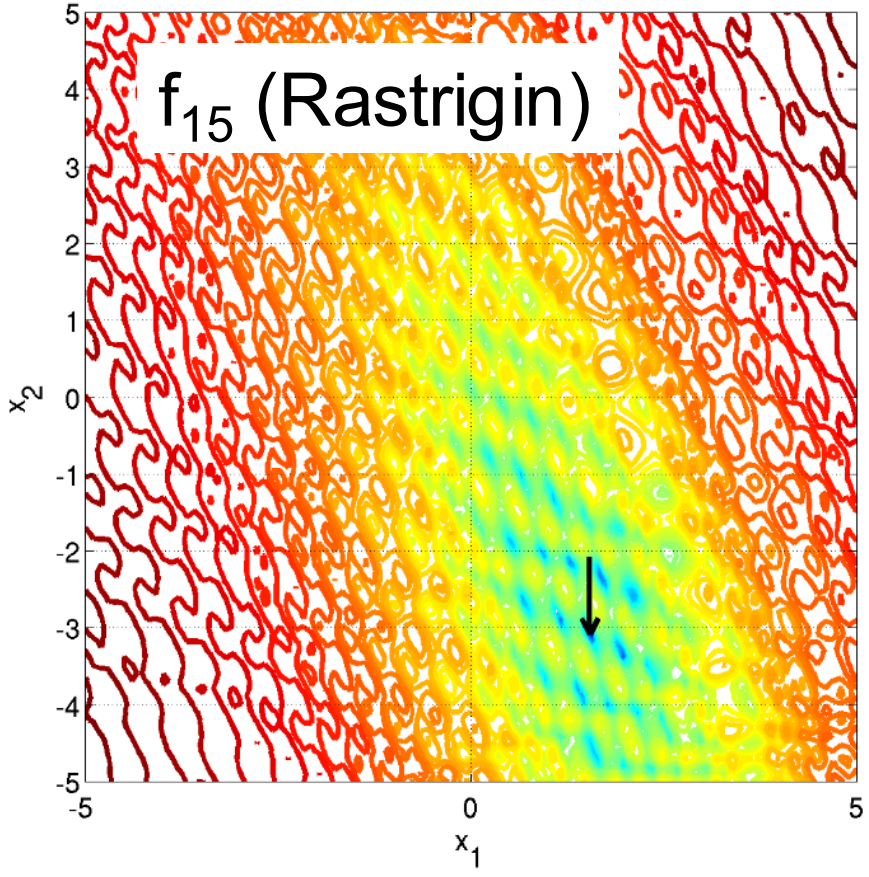
Plus:

- the bbob functions are locally perturbed by non-linear transformations

Notion of Instances

- All COCO problems come in form of instances





linear transformations

Exercise (Part 2)

Objectives:

- investigate the performance of these 6 algorithms:
  - CMA-ES ("IPOP-CMA-ES" version)
  - CMA-ES ("BIPOP-CMA-ES" version)
  - Nelder-Mead simplex (use "NelderDoerr" version here)
  - BFGS quasi-Newton
  - Genetic Algorithm: discretization of cont. variables ("GA")
  - ONEFIFTH: (1+1)-ES with 1/5 rule
- postprocessed already (earlier today) so now:
  investigate the data!

Exercise (Part 3)

Objective:

investigate the data:

a) which algorithms are the best ones?

b) does this depend on the dimension?

c) look at single graphs: can we say something about the algorithms' invariances, e.g. wrt. rotations of the search space?

d) what's the impact of covariance-matrix-adaptation?

e) what do you think: are the displayed algorithms well-suited for problems with larger dimension?