



Metaheuristics for Smart Manufacturing

7. Comparing Optimization Algorithms

Thomas Weise · 汤卫思

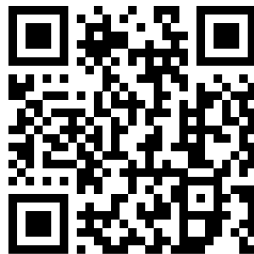
twaise@hfu.edu.cn · <http://iao.hfu.edu.cn>

Hefei University, South Campus 2
Faculty of Computer Science and Technology
Institute of Applied Optimization
230601 Shushan District, Hefei, Anhui, China
Econ. & Tech. Devel. Zone, Jinxiu Dadao 99

合肥学院 南艳湖校区/南2区
计算机科学与技术系
应用优化研究所
中国 安徽省 合肥市 蜀山区 230601
经济技术开发区 锦绣大道99号

- 1 Introduction
- 2 Performace Indicators
- 3 Statistical Measures
- 4 Statistical Comparisons
- 5 Testing is Not Enough
- 6 Summary

The slides are available at <http://iao.hfuu.edu.cn/155>, the book at <http://thomasweise.github.io/aitoa>, and the source code at <http://www.github.com/thomasWeise/aitoa-code>

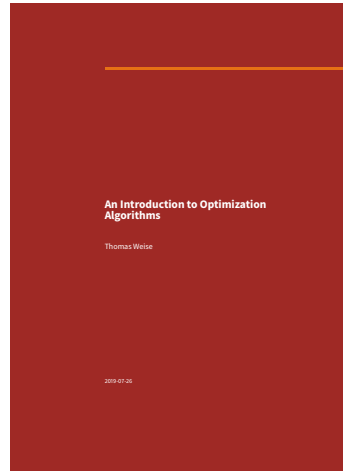


course book



course material

The contents of this course are available as free electronic book “*An Introduction to Optimization Algorithms*”^[1] at <http://thomasweise.github.io/aitoa> in [pdf](#), [html](#), [azw3](#), and [epub](#) format, created with our [bookbildeR](#) tool chain.



- 1 Introduction
- 2 Performace Indicators
- 3 Statistical Measures
- 4 Statistical Comparisons
- 5 Testing is Not Enough
- 6 Summary

- There are many optimization algorithms

- There are many optimization algorithms
- For solving an optimization problem, we want to use the algorithm most suitable for it.

- There are many optimization algorithms
- For solving an optimization problem, we want to use the algorithm most suitable for it.
- What does this mean?

- 1 Introduction
- 2 Performace Indicators**
- 3 Statistical Measures
- 4 Statistical Comparisons
- 5 Testing is Not Enough
- 6 Summary

- Key parameters^[2–5]

- Key parameters^[2–5]:
 - ① Solution quality reached after a certain runtime

- Two key parameter ^[2-5]:
 - ① Solution quality reached after a certain runtime
 - ② Runtime to reach a certain solution quality

- Two key parameter ^[2-5]:
 - ① Solution quality reached after a certain runtime
 - ② Runtime to reach a certain solution quality
- Measure data samples A containing the results from **multiple** runs and **estimate** key parameters.

- What actually is *runtime*?

Measure the (absolute) consumed runtime of the algorithm in ms

Measure the (absolute) consumed runtime of the algorithm in ms

- Advantages

Measure the (absolute) consumed runtime of the algorithm in ms

- **Advantages:**
 - Results in many works reported in this format

Measure the (absolute) consumed runtime of the algorithm in ms

- **Advantages:**
 - Results in many works reported in this format
 - A quantity that makes physical sense

Measure the (absolute) consumed runtime of the algorithm in ms

- **Advantages:**
 - Results in many works reported in this format
 - A quantity that makes physical sense
 - Includes all “hidden complexities” of algorithm

Measure the (absolute) consumed runtime of the algorithm in ms

- **Advantages:**
 - Results in many works reported in this format
 - A quantity that makes physical sense
 - Includes all “hidden complexities” of algorithm
- **Disadvantages**

Measure the (absolute) consumed runtime of the algorithm in ms

- **Advantages:**
 - Results in many works reported in this format
 - A quantity that makes physical sense
 - Includes all “hidden complexities” of algorithm
- **Disadvantages:**
 - Strongly machine dependent

Measure the (absolute) consumed runtime of the algorithm in ms

- **Advantages:**
 - Results in many works reported in this format
 - A quantity that makes physical sense
 - Includes all “hidden complexities” of algorithm
- **Disadvantages:**
 - Strongly machine dependent
 - Granularity of about 10ms: many things seem to happen at the same time

Measure the (absolute) consumed runtime of the algorithm in ms

- **Advantages:**

- Results in many works reported in this format
- A quantity that makes physical sense
- Includes all “hidden complexities” of algorithm

- **Disadvantages:**

- Strongly machine dependent
- Granularity of about 10ms: many things seem to happen at the same time
- Can be biased by “outside effects”, e.g., OS, scheduling, other processes, I/O, swapping, ...

Measure the (absolute) consumed runtime of the algorithm in ms

- **Advantages:**

- Results in many works reported in this format
- A quantity that makes physical sense
- Includes all “hidden complexities” of algorithm

- **Disadvantages:**

- Strongly machine dependent
- Granularity of about 10ms: many things seem to happen at the same time
- Can be biased by “outside effects”, e.g., OS, scheduling, other processes, I/O, swapping, ...
- Inherently incomparable

Measure the (absolute) consumed runtime of the algorithm in ms

- **Advantages:**
 - Results in many works reported in this format
 - A quantity that makes physical sense
 - Includes all “hidden complexities” of algorithm
- **Disadvantages:**
 - Strongly machine dependent
 - Granularity of about 10ms: many things seem to happen at the same time
 - Can be biased by “outside effects”, e.g., OS, scheduling, other processes, I/O, swapping, ...
 - Inherently incomparable
- Hardware, software, OS, etc. all have nothing to do with the *optimization algorithm* itself and are relevant only in a specific application...

Measure the number of fully constructed and tested candidate solutions

Measure the number of fully constructed and tested candidate solutions

- Advantages

Measure the number of fully constructed and tested candidate solutions

- **Advantages:**
 - Results in many works reported in this format (or FEs can be deduced)

Measure the number of fully constructed and tested candidate solutions

- **Advantages:**
 - Results in many works reported in this format (or FEs can be deduced)
 - Machine-independent measure

Measure the number of fully constructed and tested candidate solutions

- **Advantages:**
 - Results in many works reported in this format (or FEs can be deduced)
 - Machine-independent measure
 - Cannot be influenced by “outside effects”

Measure the number of fully constructed and tested candidate solutions

- **Advantages:**
 - Results in many works reported in this format (or FEs can be deduced)
 - Machine-independent measure
 - Cannot be influenced by “outside effects”
 - In many optimization problems, computing the objective value is the most time consuming task

Measure the number of fully constructed and tested candidate solutions

- **Advantages:**
 - Results in many works reported in this format (or FEs can be deduced)
 - Machine-independent measure
 - Cannot be influenced by “outside effects”
 - In many optimization problems, computing the objective value is the most time consuming task
- **Disadvantages**

Measure the number of fully constructed and tested candidate solutions

- **Advantages:**
 - Results in many works reported in this format (or FEs can be deduced)
 - Machine-independent measure
 - Cannot be influenced by “outside effects”
 - In many optimization problems, computing the objective value is the most time consuming task
- **Disadvantages:**
 - No clear relationship to real runtime

Measure the number of fully constructed and tested candidate solutions

- **Advantages:**
 - Results in many works reported in this format (or FEs can be deduced)
 - Machine-independent measure
 - Cannot be influenced by “outside effects”
 - In many optimization problems, computing the objective value is the most time consuming task
- **Disadvantages:**
 - No clear relationship to real runtime
 - Does not contain “hidden complexities” of algorithm

Measure the number of fully constructed and tested candidate solutions

- **Advantages:**
 - Results in many works reported in this format (or FEss can be deduced)
 - Machine-independent measure
 - Cannot be influenced by “outside effects”
 - In many optimization problems, computing the objective value is the most time consuming task
- **Disadvantages:**
 - No clear relationship to real runtime
 - Does not contain “hidden complexities” of algorithm
 - 1 FE: very different costs in different situations!

Measure the number of fully constructed and tested candidate solutions

- **Advantages:**

- Results in many works reported in this format (or FEs can be deduced)
- Machine-independent measure
- Cannot be influenced by “outside effects”
- In many optimization problems, computing the objective value is the most time consuming task

- **Disadvantages:**

- No clear relationship to real runtime
 - Does not contain “hidden complexities” of algorithm
 - 1 FE: very different costs in different situations!
- Relevant for comparing algorithms, but not so much for the practical application

- Rewrite the two key parameters by choosing a time measure ^[2, 4]

- Rewrite the two key parameters by choosing a time measure ^[2, 4]:
 - ① Solution quality reached after a certain number of FEs

- Rewrite the two key parameters by choosing a time measure ^[2, 4]:
 - ① Solution quality reached after a certain **number of FEs**
 - ② **Number FEs** needed to reach a certain solution quality

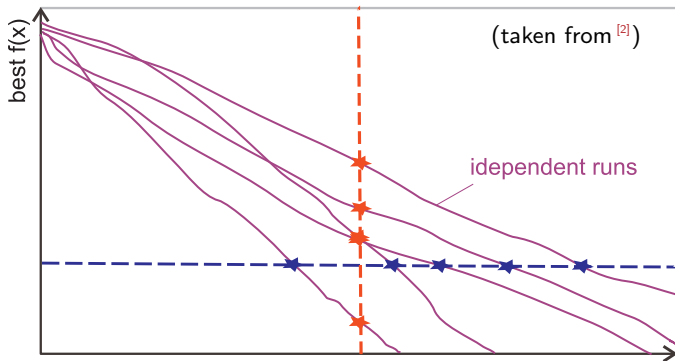
- Common measure of solution quality: Objective function value of best solution discovered.

- Common measure of solution quality: Objective function value of best solution discovered.
- Rewrite the two key parameters ^[2, 4]

- Common measure of solution quality: Objective function value of best solution discovered.
- Rewrite the two key parameters ^[2, 4]:
 - ① Best objective function value reached after a certain number of FEs

- Common measure of solution quality: Objective function value of best solution discovered.
- Rewrite the two key parameters ^[2, 4]:
 - ① Best objective function value reached after a certain number of FEs
 - ② Number FEs needed to reach a certain objective function value

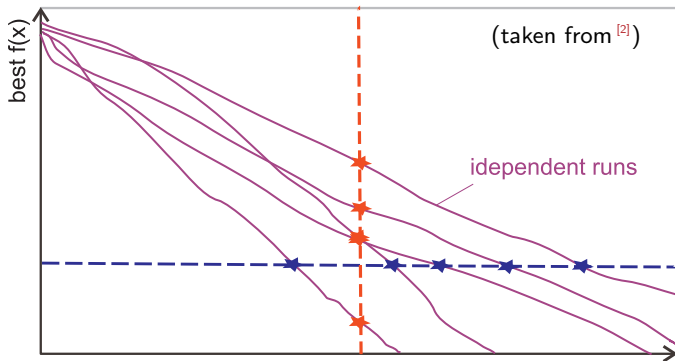
- Which one is the better performance indicator?
 - 1 Best objective function value reached after a certain number of FEs



horizontal cut: "number of FEs to reach certain best $f(x)$ " FEs

vertical cut: "best $f(x)$ after certain number of Fes"

- Which one is the better performance indicator?
 - 1 Best objective function value reached after a certain number of FEs
 - 2 Number FEs needed to reach a certain objective function value



horizontal cut: "number of FEs to reach certain $\text{best } f(x)$ " FEs

vertical cut: "best $f(x)$ after certain number of FEs"

Which Indicator is better?



- Number FEs needed to reach a certain objective function value
- Preferred by Hansen et al. ^[2]

- Number FEs needed to reach a certain objective function value
- Preferred by Hansen et al. [2]:
 - Measures a time needed to reach a target function value \Rightarrow “Algorithm A is two/ten/hundred times faster than Algorithm B in solving this problem”

- Number FEs needed to reach a certain objective function value
- Preferred by Hansen et al. [2]:
 - Measures a time needed to reach a target function value \Rightarrow “Algorithm A is two/ten/hundred times faster than Algorithm B in solving this problem”
 - Benchmark Perspective: No interpretable meaning to the fact that Algorithm A reaches a function value that is two/ten/hundred times smaller than the one reached by Algorithm B

- Best objective function value reached after a certain number of FEs

Which Indicator is better?



- Best objective function value reached after a certain number of FEs
- Preferred by many benchmark suites such as ^[6].

- Best objective function value reached after a certain number of FEs
- Preferred by many benchmark suites such as ^[6].
- Practice Perspective: Best results achievable with given time budget wins.

- Best objective function value reached after a certain number of FEs
- Preferred by many benchmark suites such as ^[6].
- Practice Perspective: Best results achievable with given time budget wins.
- This perspective maybe less suitable for benchmarking, but surely true in practice.

- Best objective function value reached after a certain number of FEs
- Preferred by many benchmark suites such as ^[6].
- Practice Perspective: Best results achievable with given time budget wins.
- This perspective maybe less suitable for benchmarking, but surely true in practice.
- This is the scenario in our JSSP example, too.

- No official consensus on which view is “better”.

- No official consensus on which view is “better”.
- This also strongly depends on the situation.

- No official consensus on which view is “better”.
- This also strongly depends on the situation.
- Best approach: Evaluate algorithm according to both methods. [4, 5, 7]

- How to determine the right maximum FEs or target function values?

- How to determine the right maximum FEs or target function values?
 - 1 From the constraints of a practical application

- How to determine the right maximum FEs or target function values?
 - ① From the constraints of a practical application
 - ② From studies in literature regarding similar or the same problem.

- How to determine the right maximum FEs or target function values?
 - ① From the constraints of a practical application
 - ② From studies in literature regarding similar or the same problem.
 - ③ From experience.

- How to determine the right maximum FEs or target function values?
 - ① From the constraints of a practical application
 - ② From studies in literature regarding similar or the same problem.
 - ③ From experience.
 - ④ From prior, small-scale experiments.

- How to determine the right maximum FEs or target function values?
 - ① From the constraints of a practical application
 - ② From studies in literature regarding similar or the same problem.
 - ③ From experience.
 - ④ From prior, small-scale experiments.
 - ⑤ Based on known lower bounds

- 1 Introduction
- 2 Performace Indicators
- 3 Statistical Measures**
- 4 Statistical Comparisons
- 5 Testing is Not Enough
- 6 Summary

- Special situation: Randomized Algorithms

- Special situation: Randomized Algorithms
- Performance values cannot be given absolute!

- Special situation: Randomized Algorithms
- Performance values cannot be given absolute!
- 1 run = 1 application of an optimization algorithm to a problem, runs are independent from all prior runs

- Special situation: Randomized Algorithms
- Performance values cannot be given absolute!
- 1 run = 1 application of an optimization algorithm to a problem, runs are independent from all prior runs
- Results can be different for each run!

- Special situation: Randomized Algorithms
- Performance values cannot be given absolute!
- 1 run = 1 application of an optimization algorithm to a problem, runs are independent from all prior runs
- Results can be different for each run!
- Executing algorithm one time does not give reliable information

- Special situation: Randomized Algorithms
- Performance values cannot be given absolute!
- 1 run = 1 application of an optimization algorithm to a problem, runs are independent from all prior runs
- Results can be different for each run!
- Executing algorithm one time does not give reliable information
- Statistical evaluation over a set of runs necessary

- Crucial Difference: **distribution** and **sample**

- Crucial Difference: distribution and sample
- A **sample** is what we *measure*.

- Crucial Difference: distribution and sample
- A sample is what we *measure*.
- A **distribution** is the asymptotic result of the ideal process

- Crucial Difference: distribution and sample
- A sample is what we *measure*.
- A distribution is the asymptotic result of the ideal process
- Statistical parameters of the distribution can be **estimated** from a sample

- Crucial Difference: distribution and sample
- A sample is what we *measure*.
- A distribution is the asymptotic result of the ideal process
- Statistical parameters of the distribution can be estimated from a sample
- Example: Dice Throw



- Crucial Difference: distribution and sample
- A sample is what we *measure*.
- A distribution is the asymptotic result of the ideal process
- Statistical parameters of the distribution can be estimated from a sample
- Example: Dice Throw
- How likely is it to roll a ①, ②, ③, ④, ⑤, or ⑥?



# throws	number	$f(\textcircled{1})$	$f(\textcircled{2})$	$f(\textcircled{3})$	$f(\textcircled{4})$	$f(\textcircled{5})$	$f(\textcircled{6})$
1	5	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000



# throws	number	$f(1)$	$f(2)$	$f(3)$	$f(4)$	$f(5)$	$f(6)$
1	5	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
2	4	0.0000	0.0000	0.0000	0.5000	0.5000	0.0000



# throws	number	$f(1)$	$f(2)$	$f(3)$	$f(4)$	$f(5)$	$f(6)$
1	5	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
2	4	0.0000	0.0000	0.0000	0.5000	0.5000	0.0000
3	1	0.3333	0.0000	0.0000	0.3333	0.3333	0.0000



# throws	number	$f(1)$	$f(2)$	$f(3)$	$f(4)$	$f(5)$	$f(6)$
1	5	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
2	4	0.0000	0.0000	0.0000	0.5000	0.5000	0.0000
3	1	0.3333	0.0000	0.0000	0.3333	0.3333	0.0000
4	4	0.2500	0.0000	0.0000	0.5000	0.2500	0.0000



# throws	number	$f(1)$	$f(2)$	$f(3)$	$f(4)$	$f(5)$	$f(6)$
1	5	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
2	4	0.0000	0.0000	0.0000	0.5000	0.5000	0.0000
3	1	0.3333	0.0000	0.0000	0.3333	0.3333	0.0000
4	4	0.2500	0.0000	0.0000	0.5000	0.2500	0.0000
5	3	0.2000	0.0000	0.2000	0.4000	0.2000	0.0000









# throws	number	$f(\text{①})$	$f(\text{②})$	$f(\text{③})$	$f(\text{④})$	$f(\text{⑤})$	$f(\text{⑥})$
1	5	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
2	4	0.0000	0.0000	0.0000	0.5000	0.5000	0.0000
3	1	0.3333	0.0000	0.0000	0.3333	0.3333	0.0000
4	4	0.2500	0.0000	0.0000	0.5000	0.2500	0.0000
5	3	0.2000	0.0000	0.2000	0.4000	0.2000	0.0000
6	3	0.1667	0.0000	0.3333	0.3333	0.1667	0.0000



# throws	number	$f(1)$	$f(2)$	$f(3)$	$f(4)$	$f(5)$	$f(6)$
1	5	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
2	4	0.0000	0.0000	0.0000	0.5000	0.5000	0.0000
3	1	0.3333	0.0000	0.0000	0.3333	0.3333	0.0000
4	4	0.2500	0.0000	0.0000	0.5000	0.2500	0.0000
5	3	0.2000	0.0000	0.2000	0.4000	0.2000	0.0000
6	3	0.1667	0.0000	0.3333	0.3333	0.1667	0.0000
7	2	0.1429	0.1429	0.2857	0.2857	0.1429	0.0000
8	1	0.2500	0.1250	0.2500	0.2500	0.1250	0.0000
9	4	0.2222	0.1111	0.2222	0.3333	0.1111	0.0000
10	2	0.2000	0.2000	0.2000	0.3000	0.1000	0.0000



# throws	number	f()	f()	f()	f()	f()	f()
1	5	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
2	4	0.0000	0.0000	0.0000	0.5000	0.5000	0.0000
3	1	0.3333	0.0000	0.0000	0.3333	0.3333	0.0000
4	4	0.2500	0.0000	0.0000	0.5000	0.2500	0.0000
5	3	0.2000	0.0000	0.2000	0.4000	0.2000	0.0000
6	3	0.1667	0.0000	0.3333	0.3333	0.1667	0.0000
7	2	0.1429	0.1429	0.2857	0.2857	0.1429	0.0000
8	1	0.2500	0.1250	0.2500	0.2500	0.1250	0.0000
9	4	0.2222	0.1111	0.2222	0.3333	0.1111	0.0000
10	2	0.2000	0.2000	0.2000	0.3000	0.1000	0.0000
11	6	0.1818	0.1818	0.1818	0.2727	0.0909	0.0909
12	3	0.1667	0.1667	0.2500	0.2500	0.0833	0.0833



# throws	number	f(1)	f(2)	f(3)	f(4)	f(5)	f(6)
1	5	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
2	4	0.0000	0.0000	0.0000	0.5000	0.5000	0.0000
3	1	0.3333	0.0000	0.0000	0.3333	0.3333	0.0000
4	4	0.2500	0.0000	0.0000	0.5000	0.2500	0.0000
5	3	0.2000	0.0000	0.2000	0.4000	0.2000	0.0000
6	3	0.1667	0.0000	0.3333	0.3333	0.1667	0.0000
7	2	0.1429	0.1429	0.2857	0.2857	0.1429	0.0000
8	1	0.2500	0.1250	0.2500	0.2500	0.1250	0.0000
9	4	0.2222	0.1111	0.2222	0.3333	0.1111	0.0000
10	2	0.2000	0.2000	0.2000	0.3000	0.1000	0.0000
11	6	0.1818	0.1818	0.1818	0.2727	0.0909	0.0909
12	3	0.1667	0.1667	0.2500	0.2500	0.0833	0.0833
100	...	0.1900	0.2100	0.1500	0.1600	0.1200	0.1700
1'000	...	0.1700	0.1670	0.1620	0.1670	0.1570	0.1770
10'000	...	0.1682	0.1699	0.1680	0.1661	0.1655	0.1623
100'000	...	0.1671	0.1649	0.1664	0.1676	0.1668	0.1672
1'000'000	...	0.1673	0.1663	0.1662	0.1673	0.1666	0.1664



# throws	number	f(1)	f(2)	f(3)	f(4)	f(5)	f(6)
1	5	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
2	4	0.0000	0.0000	0.0000	0.5000	0.5000	0.0000
3	1	0.3333	0.0000	0.0000	0.3333	0.3333	0.0000
4	4	0.2500	0.0000	0.0000	0.5000	0.2500	0.0000
5	3	0.2000	0.0000	0.2000	0.4000	0.2000	0.0000
6	3	0.1667	0.0000	0.3333	0.3333	0.1667	0.0000
7	2	0.1429	0.1429	0.2857	0.2857	0.1429	0.0000
8	1	0.2500	0.1250	0.2500	0.2500	0.1250	0.0000
9	4	0.2222	0.1111	0.2222	0.3333	0.1111	0.0000
10	2	0.2000	0.2000	0.2000	0.3000	0.1000	0.0000
11	6	0.1818	0.1818	0.1818	0.2727	0.0909	0.0909
12	3	0.1667	0.1667	0.2500	0.2500	0.0833	0.0833
100	...	0.1900	0.2100	0.1500	0.1600	0.1200	0.1700
1'000	...	0.1700	0.1670	0.1620	0.1670	0.1570	0.1770
10'000	...	0.1682	0.1699	0.1680	0.1661	0.1655	0.1623
100'000	...	0.1671	0.1649	0.1664	0.1676	0.1668	0.1672
1'000'000	...	0.1673	0.1663	0.1662	0.1673	0.1666	0.1664
10'000'000	...	0.1667	0.1667	0.1666	0.1668	0.1667	0.1665
100'000'000	...	0.1667	0.1666	0.1666	0.1667	0.1667	0.1667
1'000'000'000	...	0.1667	0.1667	0.1667	0.1667	0.1667	0.1667

- Crucial Difference: distribution and sample
- A sample is what we *measure*.
- A distribution is the asymptotic result of the ideal process
- Statistical parameters of the distribution can be estimated from a sample
- Example: Dice Throw
- How likely is it to roll a ①, ②, ③, ④, ⑤, or ⑥?
- Never forget: All measured parameters are just estimates.



- Crucial Difference: distribution and sample
- A sample is what we *measure*.
- A distribution is the asymptotic result of the ideal process
- Statistical parameters of the distribution can be estimated from a sample
- Example: Dice Throw
- How likely is it to roll a ①, ②, ③, ④, ⑤, or ⑥?
- **Never forget: All measured parameters are just estimates.**
- The parameters of a random process cannot be measured directly, but only be approximated from multiple measures



- Assume that we have obtained a sample $A = (a_0, a_1, \dots, a_{n-1})$ of n observations from an experiment.

- Assume that we have obtained a sample $A = (a_0, a_1, \dots, a_{n-1})$ of n observations from an experiment, e.g., we have measured the quality of the best discovered solutions of 101 independent runs of an optimization algorithm.

- Assume that we have obtained a sample $A = (a_0, a_1, \dots, a_{n-1})$ of n observations from an experiment, e.g., we have measured the quality of the best discovered solutions of 101 independent runs of an optimization algorithm.
- We usually want to get reduce this set of numbers to a single value which can give us an impression of what the “average outcome” (or result quality is).

- Assume that we have obtained a sample $A = (a_0, a_1, \dots, a_{n-1})$ of n observations from an experiment, e.g., we have measured the quality of the best discovered solutions of 101 independent runs of an optimization algorithm.
- We usually want to get reduce this set of numbers to a single value which can give us an impression of what the “average outcome” (or result quality is).
- Two of the most common options for doing so, for estimating the “center” of a distribution, are to either compute the **arithmetic mean** or the **median**.

Definition (Arithmetic Mean)

The arithmetic mean $\text{mean}(A)$ is an **estimate** of the expected value of a data sample $A = (a_0, a_1, \dots, a_{n-1})$.

Definition (Arithmetic Mean)

The arithmetic mean $\text{mean}(A)$ is an **estimate** of the expected value of a data sample $A = (a_0, a_1, \dots, a_{n-1})$. It is computed as the sum of all n elements a_i in the sample data A divided by the total number n of values.

Definition (Arithmetic Mean)

The arithmetic mean $\text{mean}(A)$ is an **estimate** of the expected value of a data sample $A = (a_0, a_1, \dots, a_{n-1})$. It is computed as the sum of all n elements a_i in the sample data A divided by the total number n of values.

$$\text{mean}(A) = \frac{1}{n} \sum_{i=0}^{n-1} a_i$$

Definition (Median)

The median $\text{med}(A)$ is the value separating the bigger half from the lower half of a data sample or distribution.

Definition (Median)

The median $\text{med}(A)$ is the value separating the bigger half from the lower half of a data sample or distribution. It is the value right in the middle of a *sorted* data sample $A = (a_0, a_1, \dots, a_{n-1})$ where $a_{i-1} \leq a_i \forall i \in 1 \dots (n-1)$.

Definition (Median)

The median $\text{med}(A)$ is the value separating the bigger half from the lower half of a data sample or distribution. It is the value right in the middle of a *sorted* data sample $A = (a_0, a_1, \dots, a_{n-1})$ where $a_{i-1} \leq a_i \forall i \in 1 \dots (n-1)$.

$$\text{median}(A) = \begin{cases} a_{\frac{n-1}{2}} & \text{if } n \text{ is odd} \\ \frac{1}{2} \left(a_{\frac{n}{2}-1} + a_{\frac{n}{2}} \right) & \text{otherwise} \end{cases}$$

- Sometimes the data contains outliers ^[8, 9].

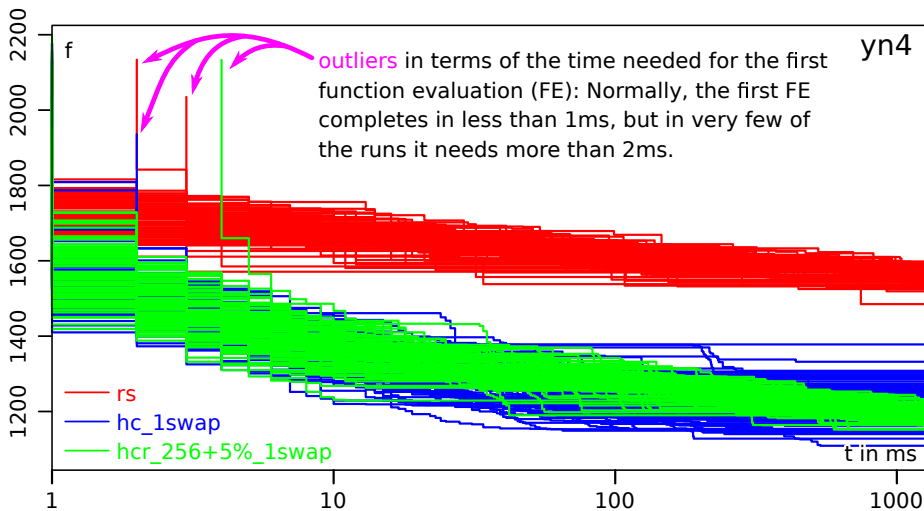
- Sometimes the data contains outliers ^[8, 9], i.e., observations which are much different from the other measurements.

- Sometimes the data contains outliers ^[8, 9], i.e., observations which are much different from the other measurements.
- They may be important, real data, e.g., represent some unusual side-effect in a clinical trial of a new medicine.

- Sometimes the data contains outliers ^[8, 9], i.e., observations which are much different from the other measurements.
- They may be important, real data, e.g., represent some unusual side-effect in a clinical trial of a new medicine.
- However, they also often represent measurement errors or observations which have been disturbed by unusual effects.

- Sometimes the data contains outliers ^[8, 9], i.e., observations which are much different from the other measurements.
- They may be important, real data, e.g., represent some unusual side-effect in a clinical trial of a new medicine.
- However, they also often represent measurement errors or observations which have been disturbed by unusual effects.
- For example, maybe the operating system was updating itself during a run of one of our JSSP algorithms and, thus, took away much of the 3 minute computation budget.

- Sometimes the data contains outliers ^[8, 9], i.e., observations which are much different from the other measurements.
- They may be important, real data, e.g., represent some unusual side-effect in a clinical trial of a new medicine.
- However, they also often represent measurement errors or observations which have been disturbed by unusual effects.
- For example, maybe the operating system was updating itself during a run of one of our JSSP algorithms and, thus, took away much of the 3 minute computation budget.
- We can see that such odd times are possible, as our experimental data shows that there are sometimes outliers in the time it takes to create and evaluate the first candidate solution.



- Two sets of data samples A and B with $n_a = n_b = 19$ values.

$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$

$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10\,008)$

- Two sets of data samples A and B with $n_a = n_b = 19$ values.

$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$

$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10\,008)$

- We find that

- Two sets of data samples A and B with $n_a = n_b = 19$ values.

$$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$$

$$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10\,008)$$

- We find that

- $\text{mean}(A) = \frac{1}{19} \sum_{i=0}^{18} a_i = \frac{133}{19} = 7$

- Two sets of data samples A and B with $n_a = n_b = 19$ values.

$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$

$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10008)$

- We find that

- $\text{mean}(A) = \frac{1}{19} \sum_{i=0}^{18} a_i = \frac{133}{19} = 7$ and
- $\text{mean}(B) = \frac{1}{19} \sum_{i=0}^{18} b_i = \frac{10127}{19} = 533$

- Two sets of data samples A and B with $n_a = n_b = 19$ values.

$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$

$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10\,008)$

- We find that
 - $\text{mean}(A) = \frac{1}{19} \sum_{i=0}^{18} a_i = \frac{133}{19} = 7$ and
 - $\text{mean}(B) = \frac{1}{19} \sum_{i=0}^{18} b_i = \frac{10127}{19} = 553$, while
 - $\text{med}(A) = a_9 = 6$

- Two sets of data samples A and B with $n_a = n_b = 19$ values.

$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$

$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10\,008)$

- We find that
 - $\text{mean}(A) = \frac{1}{19} \sum_{i=0}^{18} a_i = \frac{133}{19} = 7$ and
 - $\text{mean}(B) = \frac{1}{19} \sum_{i=0}^{18} b_i = \frac{10127}{19} = 553$, while
 - $\text{med}(A) = a_9 = 6$ and
 - $\text{med}(B) = b_9 = 6$.

- When describing a random process, we should **always use the median instead of the mean.** ^[10]

- When describing a random process, we should **always use the median instead of the mean.** ^[10], because
 - ① the median is more robust towards outliers,

- When describing a random process, we should **always use the median instead of the mean.** ^[10], because
 - ① the median is more robust towards outliers,
 - ② the mean is useful mainly for symmetric distributions and badly represents skewed distributions ^[11].

- When describing a random process, we should **always use the median instead of the mean.** ^[10], because
 - ① the median is more robust towards outliers,
 - ② the mean is useful mainly for symmetric distributions and badly represents skewed distributions ^[11].
- **The median is the first statistic we should take a look at!**

- The average gives us a good impression about the central value or location of a distribution.

- The average gives us a good impression about the central value or location of a distribution.
- It does not tell us much about the range of the data.

- The average gives us a good impression about the central value or location of a distribution.
- It does not tell us much about the range of the data.
- We do not know whether the data we have measured is very similar to the median or whether it may differ very much from the mean.

- The average gives us a good impression about the central value or location of a distribution.
- It does not tell us much about the range of the data.
- We do not know whether the data we have measured is very similar to the median or whether it may differ very much from the mean.
- For this, we can compute a measure of dispersion, i.e., a value that tells us whether the observations are stretched and spread far or squeezed tight around the center.

Definition (Variance)

The variance is the expectation of the squared deviation of a random variable from its mean.

Definition (Variance)

The variance is the expectation of the squared deviation of a random variable from its mean. The variance $\text{var}(A)$ of a data sample $A = (a_0, a_1, \dots, a_{n-1})$ with n observations can be estimated as:

$$\text{var}(A) = \frac{1}{n-1} \sum_{i=0}^{n-1} (a_i - \text{mean}(A))^2$$

Definition (Standard Deviation)

The statistical estimate $\text{sd}(A)$ of the standard deviation of a data sample $A = (a_0, a_1, \dots, a_{n-1})$ with n observations is the square root of the estimated variance $\text{var}(A)$.

$$\text{sd}(A) = \sqrt{\text{var}(A)}$$

- Small standard deviations indicate that the observations tend to be similar to the mean.

- Small standard deviations indicate that the observations tend to be similar to the mean.
- Large standard deviations indicate that they tend to be far from the mean.

- Small standard deviations indicate that the observations tend to be similar to the mean.
- Large standard deviations indicate that they tend to be far from the mean.
- Small standard deviations in optimization results and runtime indicate that the algorithm is reliable.

- Small standard deviations indicate that the observations tend to be similar to the mean.
- Large standard deviations indicate that they tend to be far from the mean.
- Small standard deviations in optimization results and runtime indicate that the algorithm is reliable.
- Large standard deviations indicate unreliable algorithms

- Small standard deviations indicate that the observations tend to be similar to the mean.
- Large standard deviations indicate that they tend to be far from the mean.
- Small standard deviations in optimization results and runtime indicate that the algorithm is reliable.
- Large standard deviations indicate unreliable algorithms, but may also offer a potential that could be exploited

- Small standard deviations indicate that the observations tend to be similar to the mean.
- Large standard deviations indicate that they tend to be far from the mean.
- Small standard deviations in optimization results and runtime indicate that the algorithm is reliable.
- Large standard deviations indicate unreliable algorithms, but may also offer a potential that could be exploited (see hill climber *with restarts*)

Definition (Quantile)

The q -quantiles are the cut points that divide a sorted data sample $A = (a_0, a_1, \dots, a_{n-1})$ where $a_{i-1} \leq a_i \forall i \in 1 \dots (n-1)$ into q -equally sized parts.

Definition (Quantile)

The q -quantiles are the cut points that divide a sorted data sample $A = (a_0, a_1, \dots, a_{n-1})$ where $a_{i-1} \leq a_i \forall i \in 1 \dots (n-1)$ into q -equally sized parts. quantile_q^k be the k^{th} q -quantile, with $k \in 1 \dots (q-1)$, i.e., there are $q-1$ of the q -quantiles.

$$\begin{aligned} h &= (n-1) \frac{k}{q} \\ \text{quantile}_q^k(A) &= \begin{cases} a_h & \text{if } h \text{ is integer} \\ a_{\lfloor h \rfloor} + (h - \lfloor h \rfloor) * (a_{\lfloor h \rfloor + 1} - a_{\lfloor h \rfloor}) & \text{otherwise} \end{cases} \end{aligned}$$

Definition (Quantile)

The q -quantiles are the cut points that divide a sorted data sample $A = (a_0, a_1, \dots, a_{n-1})$ where $a_{i-1} \leq a_i \forall i \in 1 \dots (n-1)$ into q -equally sized parts. quantile_q^k be the k^{th} q -quantile, with $k \in 1 \dots (q-1)$, i.e., there are $q-1$ of the q -quantiles.

$$\begin{aligned} h &= (n-1) \frac{k}{q} \\ \text{quantile}_q^k(A) &= \begin{cases} a_h & \text{if } h \text{ is integer} \\ a_{\lfloor h \rfloor} + (h - \lfloor h \rfloor) * (a_{\lfloor h \rfloor + 1} - a_{\lfloor h \rfloor}) & \text{otherwise} \end{cases} \end{aligned}$$

- The $\text{quantile}_1^2 A$ is the median of A

Definition (Quantile)

The q -quantiles are the cut points that divide a sorted data sample $A = (a_0, a_1, \dots, a_{n-1})$ where $a_{i-1} \leq a_i \forall i \in 1 \dots (n-1)$ into q -equally sized parts. quantile_q^k be the k^{th} q -quantile, with $k \in 1 \dots (q-1)$, i.e., there are $q-1$ of the q -quantiles.

$$\begin{aligned} h &= (n-1) \frac{k}{q} \\ \text{quantile}_q^k(A) &= \begin{cases} a_h & \text{if } h \text{ is integer} \\ a_{\lfloor h \rfloor} + (h - \lfloor h \rfloor) * (a_{\lfloor h \rfloor + 1} - a_{\lfloor h \rfloor}) & \text{otherwise} \end{cases} \end{aligned}$$

- The $\text{quantile}_1^2 A$ is the median of A
- 4-quantiles are called quartiles.

- Two data samples A and B with $n_a = n_b = 19$ values.

$$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$$

$$\text{mean}(A) = 7$$

$$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10008)$$

$$\text{mean}(B) = 533$$

- Two data samples A and B with $n_a = n_b = 19$ values.

$$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$$

$$\text{mean}(A) = 7$$

$$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10008)$$

$$\text{mean}(B) = 533$$

$$\text{var}A = \frac{1}{19-1} \sum_{i=1}^{19} (a_i - \text{mean}(a))^2 = \frac{198}{18} = 11$$

$$\text{var}B = \frac{1}{19-1} \sum_{i=1}^{19} (b_i - \text{mean}(b))^2 = \frac{94\,763\,306}{18} \approx 5\,264\,628.1$$

- Two data samples A and B with $n_a = n_b = 19$ values.

$$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$$

$$\text{mean}(A) = 7$$

$$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10008)$$

$$\text{mean}(B) = 533$$

$$\text{var}A = \frac{1}{19-1} \sum_{i=1}^{19} (a_i - \text{mean}(a))^2 = \frac{198}{18} = 11$$

$$\text{var}B = \frac{1}{19-1} \sum_{i=1}^{19} (b_i - \text{mean}(b))^2 = \frac{94\,763\,306}{18} \approx 5\,264\,628.1$$

$$\text{sd}A = \sqrt{\text{var}A} = \sqrt{11} \approx 3.316\,624\,79$$

$$\text{sd}B = \sqrt{\text{var}B} = \sqrt{\frac{94\,763\,306}{18}} \approx 2\,294.477\,743$$

- Two data samples A and B with $n_a = n_b = 19$ values.

$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$

$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10\,008)$

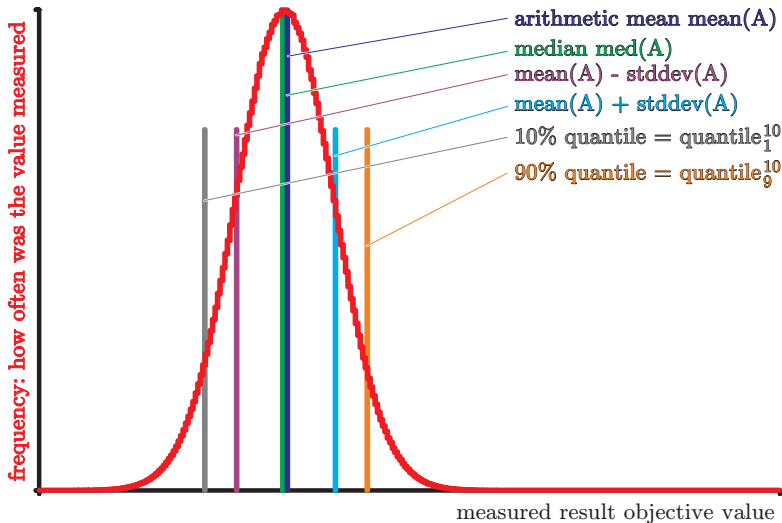
- Two data samples A and B with $n_a = n_b = 19$ values.

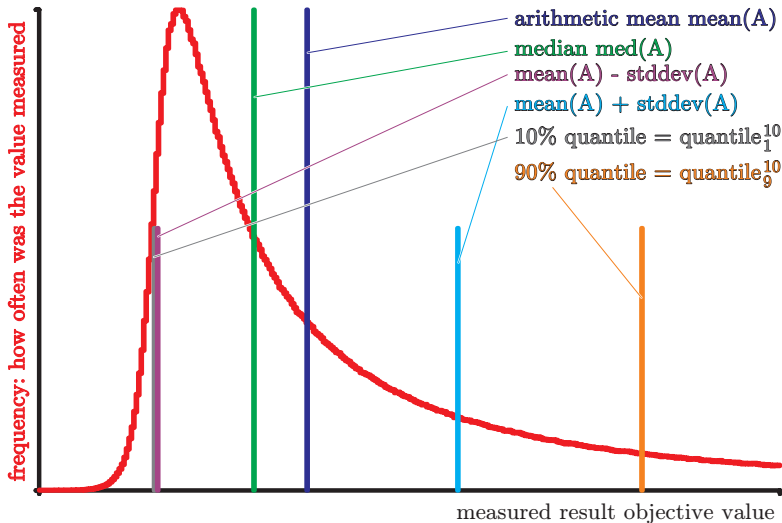
$$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$$

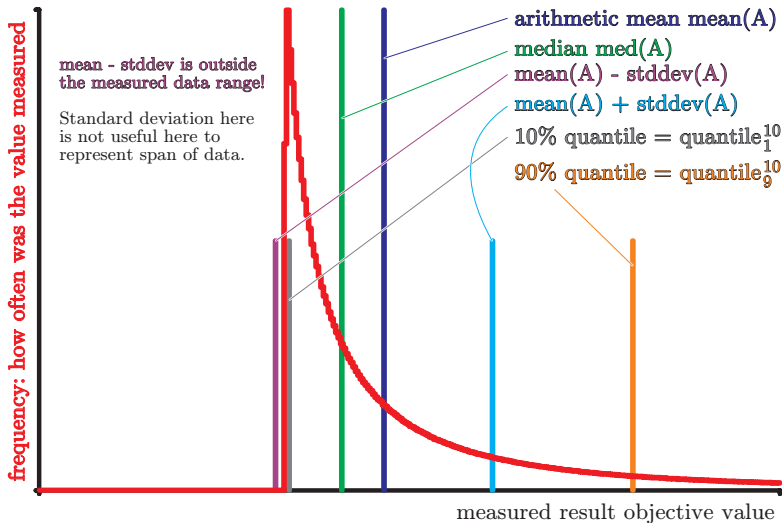
$$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10\,008)$$

$$\text{quantile}_4^1(A) = \text{quantile}_4^1(B) = 4.5$$

$$\text{quantile}_4^3(A) = \text{quantile}_4^3(B) = 9$$







- We should prefer robust statistical measures

- We should prefer robust statistical measures, which are:
 - ① Median
 - ② Quantiles

- We should prefer robust statistical measures, which are:
 - 1 Median
 - 2 Quantiles
- *Only if necessary*, compute the estimates of the
 - 1 Arithmetic Mean
 - 2 Standard Deviation

- 1 Introduction
- 2 Performace Indicators
- 3 Statistical Measures
- 4 Statistical Comparisons**
- 5 Testing is Not Enough
- 6 Summary

- We can now e.g., perform 20 runs each with two different optimization algorithms on one problem and compute the median of one of the two performance measures.

- We can now e.g., perform 20 runs each with two different optimization algorithms on one problem and compute the median of one of the two performance measures.
- Likely, they will be different.

- We can now e.g., perform 20 runs each with two different optimization algorithms on one problem and compute the median of one of the two performance measures.
- Likely, they will be different.
- For one of the two algorithms, the results will be better.

- We can now e.g., perform 20 runs each with two different optimization algorithms on one problem and compute the median of one of the two performance measures.
- Likely, they will be different.
- For one of the two algorithms, the results will be better.
- What does this mean?

- We can now e.g., perform 20 runs each with two different optimization algorithms on one problem and compute the median of one of the two performance measures.
- Likely, they will be different.
- For one of the two algorithms, the results will be better.
- What does this mean?
- It means that one of the two algorithms is better

- We can now e.g., perform 20 runs each with two different optimization algorithms on one problem and compute the median of one of the two performance measures.
- Likely, they will be different.
- For one of the two algorithms, the results will be better.
- What does this mean?
- It means that one of the two algorithms is better with a certain probability

- We can now e.g., perform 20 runs each with two different optimization algorithms on one problem and compute the median of one of the two performance measures.
- Likely, they will be different.
- For one of the two algorithms, the results will be better.
- **What does this mean?**
- It means that one of the two algorithms is better with a certain probability
- If we say “ A is better than B ”, we have a certain chance α to be wrong.

- We can now e.g., perform 20 runs each with two different optimization algorithms on one problem and compute the median of one of the two performance measures.
- Likely, they will be different.
- For one of the two algorithms, the results will be better.
- What does this mean?
- It means that one of the two algorithms is better with a certain probability
- If we say “ A is better than B ”, we have a certain chance α to be wrong.
- The statement “ A is better than B ” makes only sense if we can give an upper bound α for the error probability!

- Compare two data samples $A = (a_1, a_2, \dots)$ and $B = (b_1, b_2, \dots)$ and

- Compare two data samples $A = (a_1, a_2, \dots)$ and $B = (b_1, b_2, \dots)$ and
- Get a result (e.g., “The median of A is bigger than the median of B ”) **together with** an error probability p that the conclusion is wrong.

- Compare two data samples $A = (a_1, a_2, \dots)$ and $B = (b_1, b_2, \dots)$ and
- Get a result (e.g., “The median of A is bigger than the median of B ”) **together with** an error probability p that the conclusion is wrong.
- If p is less than a significance level (upper bound) α , we can accept the conclusion.

- Compare two data samples $A = (a_1, a_2, \dots)$ and $B = (b_1, b_2, \dots)$ and
- Get a result (e.g., “The median of A is bigger than the median of B ”) **together with** an error probability p that the conclusion is wrong.
- If p is less than a significance level (upper bound) α , we can accept the conclusion.
- Otherwise, the observation is not significant.

- We observe some ongoing process P and make some kind of observation O .

- We observe some ongoing process P and make some kind of observation O .
- Question: Can we say: “The observation O is a good approximation of what process P does”?

- We observe some ongoing process P and make some kind of observation O .
- Question: Can we say: “The observation O is a good approximation of what process P does”?
- Question: How likely is this observation O in the case that it is **NOT** an approximation of P .

- We observe some ongoing process P and make some kind of observation O .
- Question: Can we say: “The observation O is a good approximation of what process P does”?
- Question: How likely is this observation O in the case that it is **NOT** an approximation of P .
- In other words: What is the probability that O occurs if it does not represent the statistical distribution of the sampled process P ?

- Coin flip game: We flip a coin. If it is heads, I give you 1 RMB, if it is tails, you give me 1 RMB.

Example for Underlying Idea

- Coin flip game: We flip a coin. If it is heads, I give you 1 RMB, if it is tails, you give me 1 RMB.



heads

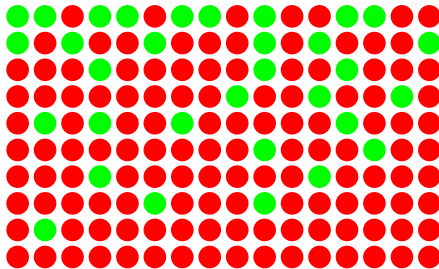


tails

- Coin flip game: We flip a coin. If it is heads, I give you 1 RMB, if it is tails, you give me 1 RMB.
- We play 160 times.
- I win 128 times. You win 32 times.

Example for Underlying Idea

- Coin flip game: We flip a coin. If it is heads, I give you 1 RMB, if it is tails, you give me 1 RMB.
- We play 160 times.
- I win 128 times. You win 32 times.



- Coin flip game: We flip a coin. If it is heads, I give you 1 RMB, if it is tails, you give me 1 RMB.
- We play 160 times.
- I win 128 times. You win 32 times.
- Did I cheat? Is my coin “fixed”? (i.e., is your chance to win \neq 50%)

- Coin flip game: We flip a coin. If it is heads, I give you 1 RMB, if it is tails, you give me 1 RMB.
- We play 160 times.
- I win 128 times. You win 32 times.
- Did I cheat? Is my coin “fixed”? (i.e., is your chance to win \neq 50%)
- **Assumption: I cheat.** (alternative hypothesis H_1)

- Coin flip game: We flip a coin. If it is heads, I give you 1 RMB, if it is tails, you give me 1 RMB.
- We play 160 times.
- I win 128 times. You win 32 times.
- Did I cheat? Is my coin “fixed”? (i.e., is your chance to win \neq 50%)
- Assumption: I cheat. (alternative hypothesis H_1)
- It is impossible to compute my winning probability if I cheated...

- Coin flip game: We flip a coin. If it is heads, I give you 1 RMB, if it is tails, you give me 1 RMB.
- We play 160 times.
- I win 128 times. You win 32 times.
- Did I cheat? Is my coin “fixed”? (i.e., is your chance to win \neq 50%)
- Assumption: I cheat. (alternative hypothesis H_1)
- It is impossible to compute my winning probability if I cheated. . .
- Counter-Assumption: I did not cheat. (null hypothesis H_0)

- Coin flip game: We flip a coin. If it is heads, I give you 1 RMB, if it is tails, you give me 1 RMB.
- We play 160 times.
- I win 128 times. You win 32 times.
- Did I cheat? Is my coin “fixed”? (i.e., is your chance to win \neq 50%)
- Assumption: I cheat. (alternative hypothesis H_1)
- It is impossible to compute my winning probability if I cheated. . .
- Counter-Assumption: I did not cheat. (null hypothesis H_0)
- How likely is it that I win **at least** 128 times if I did **not** cheat?

- Coin flip game: We flip a coin. If it is heads, I give you 1 RMB, if it is tails, you give me 1 RMB.
- We play 160 times.
- I win 128 times. You win 32 times.
- Did I cheat? Is my coin “fixed”? (i.e., is your chance to win \neq 50%)
- Assumption: I cheat. (alternative hypothesis H_1)
- It is impossible to compute my winning probability if I cheated. . .
- Counter-Assumption: I did not cheat. (null hypothesis H_0)
- How likely is it that I win **at least** 128 times if I did **not** cheat?
- (What we will do right now is called *binomial test*.)

- How likely is it that I win **at least** 128 times if I did not cheat?

- How likely is it that I win **at least** 128 times if I did not cheat?
- Then, the probabilities for heads and tails are
 $q = P(\text{head}) = P(\text{tail}) = 0.5$.

- How likely is it that I win **at least** 128 times if I did not cheat?
- Then, the probabilities for heads and tails are
 $q = P(\text{head}) = P(\text{tail}) = 0.5$.
- Flipping a coin n times is a Bernoulli Process

- How likely is it that I win **at least** 128 times if I did not cheat?
- Then, the probabilities for heads and tails are
 $q = P(\text{head}) = P(\text{tail}) = 0.5$.
- Flipping a coin n times is a Bernoulli Process
- The probability $P(k|n)$ to flip $k \in 0..n$ times heads (or tails) is thus:

$$P(k|n) = \binom{n}{k} 0.5^k * (1 - 0.5)^{n-k} = \binom{n}{k} 0.5^k * 0.5^{n-k} = \binom{n}{k} \frac{1}{2^n}$$

- How likely is it that I win **at least** 128 times if I did not cheat?
- Then, the probabilities for heads and tails are
 $q = P(\text{head}) = P(\text{tail}) = 0.5$.
- Flipping a coin n times is a Bernoulli Process
- The probability $P(k|n)$ to flip $k \in 0..n$ times heads (or tails) is thus:

$$P(k|n) = \binom{n}{k} 0.5^k * (1 - 0.5)^{n-k} = \binom{n}{k} 0.5^k * 0.5^{n-k} = \binom{n}{k} \frac{1}{2^n}$$

- For winning **at least** $z = 128$ times, we need to compute:

$$P(k \geq z|n) = \sum_{i=z}^n P(i|n)$$

- How likely is it that I win **at least** 128 times if I did not cheat?
- Then, the probabilities for heads and tails are
 $q = P(\text{head}) = P(\text{tail}) = 0.5$.
- Flipping a coin n times is a Bernoulli Process
- The probability $P(k|n)$ to flip $k \in 0..n$ times heads (or tails) is thus:

$$P(k|n) = \binom{n}{k} 0.5^k * (1 - 0.5)^{n-k} = \binom{n}{k} 0.5^k * 0.5^{n-k} = \binom{n}{k} \frac{1}{2^n}$$

- For winning **at least** $z = 128$ times, we need to compute:

$$P(k \geq z|n) = \sum_{i=z}^n P(i|n) = \sum_{i=128}^{160} P(i|160) = \sum_{i=128}^{160} \left[\binom{160}{i} \frac{1}{2^{160}} \right]$$

- How likely is it that I win **at least** 128 times if I did not cheat?
- Then, the probabilities for heads and tails are
 $q = P(\text{head}) = P(\text{tail}) = 0.5$.
- Flipping a coin n times is a Bernoulli Process
- The probability $P(k|n)$ to flip $k \in 0..n$ times heads (or tails) is thus:

$$P(k|n) = \binom{n}{k} 0.5^k * (1 - 0.5)^{n-k} = \binom{n}{k} 0.5^k * 0.5^{n-k} = \binom{n}{k} \frac{1}{2^n}$$

- For winning **at least** $z = 128$ times, we need to compute:

$$P(k \geq z|n) = \sum_{i=z}^n P(i|n) = \frac{1}{2^{160}} \sum_{i=128}^{160} \binom{160}{i}$$

- How likely is it that I win **at least** 128 times if I did not cheat?
- Then, the probabilities for heads and tails are
 $q = P(\text{head}) = P(\text{tail}) = 0.5$.
- Flipping a coin n times is a Bernoulli Process
- The probability $P(k|n)$ to flip $k \in 0..n$ times heads (or tails) is thus:

$$P(k|n) = \binom{n}{k} 0.5^k * (1 - 0.5)^{n-k} = \binom{n}{k} 0.5^k * 0.5^{n-k} = \binom{n}{k} \frac{1}{2^n}$$

- For winning **at least** $z = 128$ times, we need to compute:

$$\begin{aligned} P(k \geq z|n) &= \sum_{i=z}^n P(i|n) = \frac{1}{2^{160}} \sum_{i=128}^{160} \binom{160}{i} \\ &= \frac{1'538'590'628'148'134'280'316'221'828'039'113}{365'375'409'332'725'729'550'921'208'179'070'754'913'983'135'744} \end{aligned}$$

- How likely is it that I win **at least** 128 times if I did not cheat?
- Then, the probabilities for heads and tails are
 $q = P(\text{head}) = P(\text{tail}) = 0.5$.
- Flipping a coin n times is a Bernoulli Process
- The probability $P(k|n)$ to flip $k \in 0..n$ times heads (or tails) is thus:

$$P(k|n) = \binom{n}{k} 0.5^k * (1 - 0.5)^{n-k} = \binom{n}{k} 0.5^k * 0.5^{n-k} = \binom{n}{k} \frac{1}{2^n}$$

- For winning **at least** $z = 128$ times, we need to compute:

$$P(k \geq z|n) = \sum_{i=z}^n P(i|n) = \frac{1}{2^{160}} \sum_{i=128}^{160} \binom{160}{i} \approx \frac{1.539 * 10^{33}}{3.654 * 10^{47}}$$

- How likely is it that I win **at least** 128 times if I did not cheat?
- Then, the probabilities for heads and tails are
 $q = P(\text{head}) = P(\text{tail}) = 0.5$.
- Flipping a coin n times is a Bernoulli Process
- The probability $P(k|n)$ to flip $k \in 0..n$ times heads (or tails) is thus:

$$P(k|n) = \binom{n}{k} 0.5^k * (1 - 0.5)^{n-k} = \binom{n}{k} 0.5^k * 0.5^{n-k} = \binom{n}{k} \frac{1}{2^n}$$

- For winning **at least** $z = 128$ times, we need to compute:

$$\begin{aligned} P(k \geq z|n) &= \sum_{i=z}^n P(i|n) = \frac{1}{2^{160}} \sum_{i=128}^{160} \binom{160}{i} \approx \frac{1.539 * 10^{33}}{3.654 * 10^{47}} \\ &\approx 0.0000000000000000421098571 \end{aligned}$$

- How likely is it that I win **at least** 128 times if I did not cheat?
- Then, the probabilities for heads and tails are
 $q = P(\text{head}) = P(\text{tail}) = 0.5$.
- Flipping a coin n times is a Bernoulli Process
- The probability $P(k|n)$ to flip $k \in 0..n$ times heads (or tails) is thus:

$$P(k|n) = \binom{n}{k} 0.5^k * (1 - 0.5)^{n-k} = \binom{n}{k} 0.5^k * 0.5^{n-k} = \binom{n}{k} \frac{1}{2^n}$$

- For winning **at least** $z = 128$ times, we need to compute:

$$\begin{aligned} P(k \geq z|n) &= \sum_{i=z}^n P(i|n) = \frac{1}{2^{160}} \sum_{i=128}^{160} \binom{160}{i} \approx \frac{1.539 * 10^{33}}{3.654 * 10^{47}} \\ &\approx 4.211 \cdot 10^{-15} \end{aligned}$$

- Question: How likely is it that I win at least 128 times if I did not cheat?

- Question: How likely is it that I win at least 128 times if I did not cheat?
- If the coin was an ideal coin, the chance that I win at least 128 out of 160 times is about $4 \cdot 10^{-15}$.

- Question: How likely is it that I win at least 128 times if I did not cheat?
- If the coin was an ideal coin, the chance that I win at least 128 out of 160 times is about $4 \cdot 10^{-15}$.
- If you claim that I cheat, your chance to be wrong is about $4 \cdot 10^{-15}$.

- Question: How likely is it that I win at least 128 times if I did not cheat?
- If the coin was an ideal coin, the chance that I win at least 128 out of 160 times is about $4 \cdot 10^{-15}$.
- If you claim that I cheat, your chance to be wrong is about $4 \cdot 10^{-15}$.
- Thus, if we cannot accept a chance p to be wrong higher than a significance level $\alpha = 1\%$, we can still say:

The observation is significant, I did likely cheat.

- We want to compare two algorithms \mathcal{A} and \mathcal{B} on a given problem instance.

- We want to compare two algorithms \mathcal{A} and \mathcal{B} on a given problem instance.
- We have conducted a small experiment and measured objective values of their final runs in a few runs in form of the two data sets A and B , respectively:

$$\begin{aligned}A &= (2, 5, 6, 7, 9, 10) \\ B &= (1, 3, 4, 8)\end{aligned}$$

- We want to compare two algorithms \mathcal{A} and \mathcal{B} on a given problem instance.
- We have conducted a small experiment and measured objective values of their final runs in a few runs in form of the two data sets A and B , respectively:

$$A = (2, 5, 6, 7, 9, 10)$$

$$B = (1, 3, 4, 8)$$

- From this, we can estimate the arithmetic means:

- We want to compare two algorithms \mathcal{A} and \mathcal{B} on a given problem instance.
- We have conducted a small experiment and measured objective values of their final runs in a few runs in form of the two data sets A and B , respectively:

$$\begin{aligned}A &= (2, 5, 6, 7, 9, 10) \\ B &= (1, 3, 4, 8)\end{aligned}$$

- From this, we can estimate the arithmetic means:

$$\begin{aligned}\text{mean}(a) &= \frac{39}{6} = 6.5 \\ \text{mean}(b) &= \frac{16}{4} = 4\end{aligned}$$

$$\begin{aligned}\text{mean}(a) &= \frac{39}{6} = 6.5 \\ \text{mean}(b) &= \frac{16}{4} = 4\end{aligned}$$

- It looks like algorithm \mathcal{B} may produce the smaller objective values.

$$\begin{aligned}\text{mean}(a) &= \frac{39}{6} = 6.5 \\ \text{mean}(b) &= \frac{16}{4} = 4\end{aligned}$$

- It looks like algorithm \mathcal{B} may produce the smaller objective values.
- But is this assumption justified based on the data we have?

$$\begin{aligned}\text{mean}(a) &= \frac{39}{6} = 6.5 \\ \text{mean}(b) &= \frac{16}{4} = 4\end{aligned}$$

- It looks like algorithm \mathcal{B} may produce the smaller objective values.
- But is this assumption justified based on the data we have?
- Is the difference between $\text{mean}(A)$ and $\text{mean}(B)$ significant at a threshold of $\alpha = 2$?

- If \mathcal{B} is truly better than \mathcal{A} , which is our hypothesis H_1 , then we cannot calculate anything.

- If \mathcal{B} is truly better than \mathcal{A} , which is our hypothesis H_1 , then we cannot calculate anything.
- Let us therefore assume as null hypothesis H_0 the observed difference did just happen by chance and, well, $\mathcal{A} \equiv \mathcal{B}$.

- If \mathcal{B} is truly better than \mathcal{A} , which is our hypothesis H_1 , then we cannot calculate anything.
- Let us therefore assume as null hypothesis H_0 the observed difference did just happen by chance and, well, $\mathcal{A} \equiv \mathcal{B}$.
- Then, this would mean that the data samples A and B stem from the **same** algorithm (as $\mathcal{A} \equiv \mathcal{B}$).

- If \mathcal{B} is truly better than \mathcal{A} , which is our hypothesis H_1 , then we cannot calculate anything.
- Let us therefore assume as null hypothesis H_0 the observed difference did just happen by chance and, well, $\mathcal{A} \equiv \mathcal{B}$.
- Then, this would mean that the data samples A and B stem from the **same** algorithm (as $\mathcal{A} \equiv \mathcal{B}$).
- The division into the two sets would only be artificial, an artifact of our experimental design.

- If \mathcal{B} is truly better than \mathcal{A} , which is our hypothesis H_1 , then we cannot calculate anything.
- Let us therefore assume as null hypothesis H_0 the observed difference did just happen by chance and, well, $\mathcal{A} \equiv \mathcal{B}$.
- Then, this would mean that the data samples A and B stem from the **same** algorithm (as $\mathcal{A} \equiv \mathcal{B}$).
- The division into the two sets would only be artificial, an artifact of our experimental design.
- Instead of having two data samples, we only have one, namely the union set O with 10 elements:

- If \mathcal{B} is truly better than \mathcal{A} , which is our hypothesis H_1 , then we cannot calculate anything.
- Let us therefore assume as null hypothesis H_0 the observed difference did just happen by chance and, well, $\mathcal{A} \equiv \mathcal{B}$.
- Then, this would mean that the data samples A and B stem from the **same** algorithm (as $\mathcal{A} \equiv \mathcal{B}$).
- The division into the two sets would only be artificial, an artifact of our experimental design.
- Instead of having two data samples, we only have one, namely the union set O with 10 elements:

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

- Any division C into two sets with 4 and 6 elements has the same probability

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

- Any division C into two sets with 4 and 6 elements has the same probability
- $|O| = 10$

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

- Any division C into two sets with 4 and 6 elements has the same probability
- $|O| = 10$
- There are $\binom{10}{4} = 210$ different ways to draw 4 (or 6) elements from O

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

- Any division C into two sets with 4 and 6 elements has the same probability
- $|O| = 10$
- There are $\binom{10}{4} = 210$ different ways to draw 4 (or 6) elements from O
- If H_0 holds, all have the same probability

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

- Any division C into two sets with 4 and 6 elements has the same probability
- $|O| = 10$
- There are $\binom{10}{4} = 210$ different ways to draw 4 (or 6) elements from O
- If H_0 holds, all have the same probability
- Use a program to test the combinations

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

Listing: Small tester program...

```
public class EnumerateAtLeastAsExtremeScenarios {
    public static void main(String[] args) {
        int meanLowerOrEqualTo4 = 0; //how often did we find a mean <= 4
        int totalCombinations = 0; //total number of tested combinations

        for (int i = 10; i > 0; i--) { // as 0 = numbers from 1 to 10
            for (int j = (i - 1); j > 0; j--) { // we can conveniently iterate
                for (int k = (j - 1); k > 0; k--) { // over all 4-element combos
                    for (int l = (k - 1); l > 0; l--) { // with 4 such nested loops
                        if (((i + j + k + l) / 4.0) <= 4) { // check for the extreme cases
                            meanLowerOrEqualTo4++; // count the extreme case
                            totalCombinations++; // add up combos, to verify
                        }
                    }
                }
            }
        }

        System.out.println(meanLowerOrEqualTo4 + "␣" + totalCombinations);
    }
}
```

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

- Any division C into two sets with 4 and 6 elements has the same probability
- $|O| = 10$
- There are $\binom{10}{4} = 210$ different ways to draw 4 (or 6) elements from O
- If H_0 holds, all have the same probability
- There are 27 such combinations with a mean of **less or equal 4**.

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

- Any division C into two sets with 4 and 6 elements has the same probability
- $|O| = 10$
- There are $\binom{10}{4} = 210$ different ways to draw 4 (or 6) elements from O
- If H_0 holds, all have the same probability
- There are 27 such combinations with a mean of less or equal 4.
- The probability p to observe a situation at least as extreme as A and B under H_0 is thus:

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

- Any division C into two sets with 4 and 6 elements has the same probability
- $|O| = 10$
- There are $\binom{10}{4} = 210$ different ways to draw 4 (or 6) elements from O
- If H_0 holds, all have the same probability
- There are 27 such combinations with a mean of less or equal 4.
- The probability p to observe a situation at least as extreme as A and B under H_0 is thus:

$$p = \frac{\text{\#cases } C : \text{mean}(c) \leq \text{mean}(b)}{\text{\#all cases}} = \frac{27}{210} = \frac{9}{70} \approx 0.1286$$

- Extreme cases into the other direction are the same:

$$O = A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$$

- Extreme cases into the other direction are the same:

$$\begin{aligned} O &= A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) \\ \sum_{\forall o \in O} o &= \sum_{o=1}^{10} o = \frac{10(10+1)}{2} = 55 \end{aligned}$$

- Extreme cases into the other direction are the same:

$$\begin{aligned} O &= A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) \\ \sum_{\forall o \in O} o &= \sum_{o=1}^{10} o = \frac{10(10+1)}{2} = 55 \\ \text{mean}(b) = \left(\frac{1}{4} \sum_{\forall b \in B} b \right) \leq 4 &\implies \left(\sum_{\forall b \in B} b \right) \leq 4 * 4 \leq 16 \end{aligned}$$

- Extreme cases into the other direction are the same:

$$\begin{aligned}O &= A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) \\ \sum_{\forall o \in O} o &= \sum_{o=1}^{10} o = \frac{10(10+1)}{2} = 55 \\ \text{mean}(b) = \left(\frac{1}{4} \sum_{\forall b \in B} b \right) \leq 4 &\implies \left(\sum_{\forall b \in B} b \right) \leq 4 * 4 \leq 16 \\ O = A \cup B &\implies \sum_{\forall a \in A} a = \left(\sum_{\forall o \in O} o \right) - \left(\sum_{\forall b \in B} b \right)\end{aligned}$$

- Extreme cases into the other direction are the same:

$$\begin{aligned}O &= A \cup B = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) \\ \sum_{\forall o \in O} o &= \sum_{o=1}^{10} o = \frac{10(10+1)}{2} = 55 \\ \text{mean}(b) = \left(\frac{1}{4} \sum_{\forall b \in B} b \right) \leq 4 &\implies \left(\sum_{\forall b \in B} b \right) \leq 4 * 4 \leq 16 \\ O = A \cup B &\implies \sum_{\forall a \in A} a = \left(\sum_{\forall o \in O} o \right) - \left(\sum_{\forall b \in B} b \right) \\ \sum_{\forall b \in B} b \leq 16 &\implies \left(\sum_{\forall a \in A} a \right) \geq 55 - 16 \geq 39\end{aligned}$$

- Extreme cases into the other direction are the same:

$$\begin{aligned}\sum_{\forall o \in O} o &= \sum_{o=1}^{10} o = \frac{10(10+1)}{2} = 55 \\ \text{mean}(b) = \left(\frac{1}{4} \sum_{\forall b \in B} b \right) \leq 4 &\implies \left(\sum_{\forall b \in B} b \right) \leq 4 * 4 \leq 16 \\ O = A \cup B &\implies \sum_{\forall a \in A} a = \left(\sum_{\forall o \in O} o \right) - \left(\sum_{\forall b \in B} b \right) \\ \sum_{\forall b \in B} b \leq 16 &\implies \left(\sum_{\forall a \in A} a \right) \geq 55 - 16 \geq 39 \\ \text{mean}(a) &= \frac{1}{6} \left(\sum_{\forall a \in A} a \right)\end{aligned}$$

- Extreme cases into the other direction are the same:

$$\text{mean}(b) = \left(\frac{1}{4} \sum_{\forall b \in B} b \right) \leq 4 \implies \left(\sum_{\forall b \in B} b \right) \leq 4 * 4 \leq 16$$

$$O = A \cup B \implies \sum_{\forall a \in A} a = \left(\sum_{\forall o \in O} o \right) - \left(\sum_{\forall b \in B} b \right)$$

$$\sum_{\forall b \in B} b \leq 16 \implies \left(\sum_{\forall a \in A} a \right) \geq 55 - 16 \geq 39$$

$$\text{mean}(a) = \frac{1}{6} \left(\sum_{\forall a \in A} a \right)$$

$$\text{mean}(b) \leq 4 \implies \text{mean}(a) \geq \frac{39}{6} \geq 6.5$$

- Extreme cases into the other direction are the same:

$$\text{mean}(b) = \left(\frac{1}{4} \sum_{\forall b \in B} b \right) \leq 4 \implies \left(\sum_{\forall b \in B} b \right) \leq 4 * 4 \leq 16$$

$$O = A \cup B \implies \sum_{\forall a \in A} a = \left(\sum_{\forall o \in O} o \right) - \left(\sum_{\forall b \in B} b \right)$$

$$\sum_{\forall b \in B} b \leq 16 \implies \left(\sum_{\forall a \in A} a \right) \geq 55 - 16 \geq 39$$

$$\text{mean}(a) = \frac{1}{6} \left(\sum_{\forall a \in A} a \right)$$

$$\text{mean}(b) \leq 4 \implies \text{mean}(a) \geq \frac{39}{6} \geq 6.5$$

- So – of course – we could have also done the test the other way around with the same result!

- The probability p to observe a constellation at least as extreme as A or B under H_0 is thus:

$$p = \frac{\text{\#cases } C : \text{mean}(c) \leq \text{mean}(b)}{\text{\#all cases}} = \frac{27}{210} = \frac{9}{70} \approx 0.1286$$

- The probability p to observe a constellation at least as extreme as A or B under H_0 is thus:

$$p = \frac{\#\text{cases } C : \text{mean}(c) \leq \text{mean}(b)}{\#\text{all cases}} = \frac{27}{210} = \frac{9}{70} \approx 0.1286$$

- If we claim that A and B are from distributions with different means. . .

- The probability p to observe a constellation at least as extreme as A or B under H_0 is thus:

$$p = \frac{\#\text{cases } C : \text{mean}(c) \leq \text{mean}(b)}{\#\text{all cases}} = \frac{27}{210} = \frac{9}{70} \approx 0.1286$$

- If we claim that A and B are from distributions with different means. . .
- . . . we are wrong with probability $p \approx 0.13$

- The probability p to observe a constellation at least as extreme as A or B under H_0 is thus:

$$p = \frac{\#\text{cases } C : \text{mean}(c) \leq \text{mean}(b)}{\#\text{all cases}} = \frac{27}{210} = \frac{9}{70} \approx 0.1286$$

- If we claim that A and B are from distributions with different means. . .
- . . . we are wrong with probability $p \approx 0.13$
- At a significance level of $\alpha = 2\%$, the **means** of A and B are not significantly different! ($2\% < 0.13$)

- The probability p to observe a constellation at least as extreme as A or B under H_0 is thus:

$$p = \frac{\#\text{cases } C : \text{mean}(c) \leq \text{mean}(b)}{\#\text{all cases}} = \frac{27}{210} = \frac{9}{70} \approx 0.1286$$

- If we claim that A and B are from distributions with different means. . .
- . . . we are wrong with probability $p \approx 0.13$
- At a significance level of $\alpha = 2\%$, the **means** of A and B are not significantly different! ($2\% < 0.13$)
- Actually: This here is an example for an *Randomization Test* ^[12, 13].

- The probability p to observe a constellation at least as extreme as A or B under H_0 is thus:

$$p = \frac{\text{\#cases } C : \text{mean}(c) \leq \text{mean}(b)}{\text{\#all cases}} = \frac{27}{210} = \frac{9}{70} \approx 0.1286$$

- If we claim that A and B are from distributions with different means. . .
- . . . we are wrong with probability $p \approx 0.13$
- At a significance level of $\alpha = 2\%$, the **means** of A and B are not significantly different! ($2\% < 0.13$)
- Actually: This here is an example for an *Randomization Test* ^[12, 13].
- The method here is only feasible for small sample sets, real tests are more sophisticated

- Two types of tests:

- Two types of tests:
 - ① Parametric Tests

- Two types of tests:
 - ① Parametric Tests
 - Assume that the data samples follow a certain distribution

- Two types of tests:
 - ① Parametric Tests
 - Assume that the data samples follow a certain distribution
 - Examples^[14]: t -test (assumes normal distribution)

- Two types of tests:
 - ① Parametric Tests
 - Assume that the data samples follow a certain distribution
 - Examples^[14]: t -test (assumes normal distribution)
 - The distribution of the data we measure is unknown. . .

- Two types of tests:
 - ① Parametric Tests
 - Assume that the data samples follow a certain distribution
 - Examples^[14]: t -test (assumes normal distribution)
 - The distribution of the data we measure is unknown...
 - ...and usually not normal, see further example on statistical measures.

- Two types of tests:
 - ① Parametric Tests
 - Assume that the data samples follow a certain distribution
 - Examples^[14]: t -test (assumes normal distribution)
 - The distribution of the data we measure is unknown...
 - ... and usually not normal, see further example on statistical measures.
 - The condition for using such tests cannot be met (known distribution)

- Two types of tests:
 - ① Parametric Tests
 - Assume that the data samples follow a certain distribution
 - Examples^[14]: t -test (assumes normal distribution)
 - The distribution of the data we measure is unknown...
 - ... and usually not normal, see further example on statistical measures.
 - The condition for using such tests cannot be met (known distribution)
 - Parametric Tests cannot be used here!

- Two types of tests:
 - 1 Parametric Tests
 - 2 Non-Parametric Tests

- Two types of tests:
 - ① Parametric Tests
 - ② Non-Parametric Tests
 - Make no assumption about the distribution from which the data was sampled.

- Two types of tests:
 - 1 Parametric Tests
 - 2 Non-Parametric Tests
 - Make no assumption about the distribution from which the data was sampled.
 - Examples^[10]: the Wilcoxon rank sum test with continuity correction (also called Mann-Whitney U test^[15–18], Fisher's Exact Test^[19], the Sign Test^[16, 20], the Randomization Test^[12, 13], and Wilcoxon's Signed Rank Test^[21].

- Two types of tests:
 - 1 Parametric Tests
 - 2 Non-Parametric Tests
 - Make no assumption about the distribution from which the data was sampled.
 - Examples^[10]: the Wilcoxon rank sum test with continuity correction (also called Mann-Whitney U test^[15–18], Fisher's Exact Test^[19], the Sign Test^[16, 20], the Randomization Test^[12, 13], and Wilcoxon's Signed Rank Test^[21].
 - These tests are more **robust** (less assumptions)

- Two types of tests:
 - 1 Parametric Tests
 - 2 Non-Parametric Tests
 - Make no assumption about the distribution from which the data was sampled.
 - Examples^[10]: the Wilcoxon rank sum test with continuity correction (also called Mann-Whitney U test^[15–18], Fisher's Exact Test^[19], the Sign Test^[16, 20], the Randomization Test^[12, 13], and Wilcoxon's Signed Rank Test^[21].
 - These tests are more **robust** (less assumptions)
 - **This is the kind of test we want to use!**

- Two types of tests:
 - 1 Parametric Tests
 - 2 Non-Parametric Tests
 - Make no assumption about the distribution from which the data was sampled.
 - Examples^[10]: the Wilcoxon rank sum test with continuity correction (also called Mann-Whitney U test^[15–18], Fisher's Exact Test^[19], the Sign Test^[16, 20], the Randomization Test^[12, 13], and Wilcoxon's Signed Rank Test^[21].
 - These tests are more **robust** (less assumptions)
 - **This is the kind of test we want to use!**
 - They work similar to the previous test example, but with larger sample sizes

- Two types of tests:
 - 1 Parametric Tests
 - 2 Non-Parametric Tests
 - Make no assumption about the distribution from which the data was sampled.
 - Examples^[10]: the Wilcoxon rank sum test with continuity correction (also called Mann-Whitney U test^[15–18], Fisher's Exact Test^[19], the Sign Test^[16, 20], the Randomization Test^[12, 13], and Wilcoxon's Signed Rank Test^[21].
 - These tests are more **robust** (less assumptions)
 - **This is the kind of test we want to use!**
 - They work similar to the previous test example, but with larger sample sizes
 - Often, the most suitable test is the Mann-Whitney U test.

- For comparing $N \geq 2$ algorithms, we can compare any two algorithms with each other

- For comparing $N \geq 2$ algorithms, we can compare any two algorithms with each other
- N Algorithms $\Rightarrow k = N(N - 1)/2$ statistical tests (e.g., Mann-Whitney U)

- For comparing $N \geq 2$ algorithms, we can compare any two algorithms with each other
- N Algorithms $\Rightarrow k = N(N - 1)/2$ statistical tests (e.g., Mann-Whitney U)
- k tests and each with error probability α

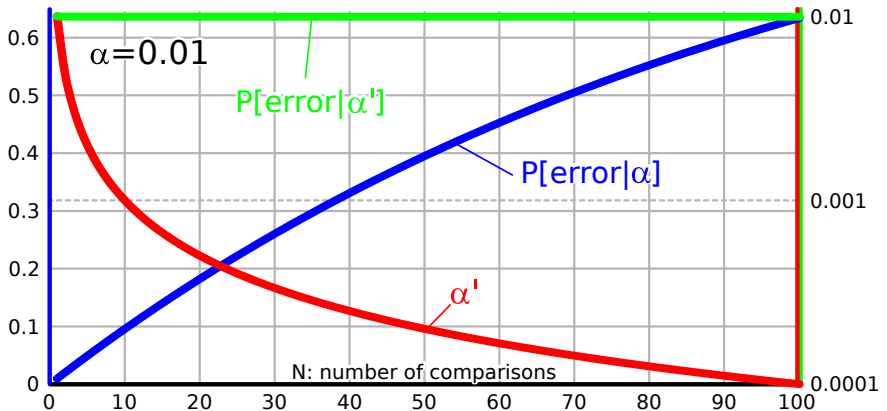
- For comparing $N \geq 2$ algorithms, we can compare any two algorithms with each other
- N Algorithms $\Rightarrow k = N(N - 1)/2$ statistical tests (e.g., Mann-Whitney U)
- k tests and each with error probability $\alpha \Rightarrow$ total probability E to make error $E = 1 - ((1 - \alpha)^k)$

- For comparing $N \geq 2$ algorithms, we can compare any two algorithms with each other
- N Algorithms $\Rightarrow k = N(N - 1)/2$ statistical tests
- k tests and each with error probability $\alpha \Rightarrow$ total probability E to make error $E = 1 - ((1 - \alpha)^k)$
- Correction needed: Bonferroni correction ^[22]

- For comparing $N \geq 2$ algorithms, we can compare any two algorithms with each other
- N Algorithms $\Rightarrow k = N(N - 1)/2$ statistical tests
- k tests and each with error probability $\alpha \Rightarrow$ total probability E to make error $E = 1 - ((1 - \alpha)^k)$
- Correction needed: Bonferroni correction^[22]: Use $\alpha' = \alpha/k$ as significance level instead of α , then the overall probability E to make an error will remain $E \leq \alpha$.

- For comparing $N \geq 2$ algorithms, we can compare any two algorithms with each other
- N Algorithms $\Rightarrow k = N(N - 1)/2$ statistical tests
- k tests and each with error probability $\alpha \Rightarrow$ total probability E to make error $E = 1 - ((1 - \alpha)^k)$
- Correction needed: Bonferroni correction^[22]: Use $\alpha' = \alpha/k$ as significance level instead of α , then the overall probability E to make an error will remain $E \leq \alpha$.

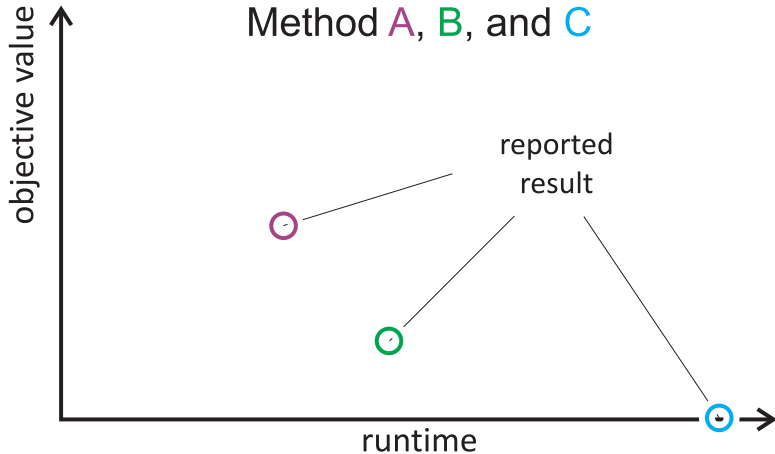
Compare $N \geq 2$ Algorithms



- 1 Introduction
- 2 Performace Indicators
- 3 Statistical Measures
- 4 Statistical Comparisons
- 5 Testing is Not Enough**
- 6 Summary

- Literature usually reports tuples “(instance, result, runtime)”

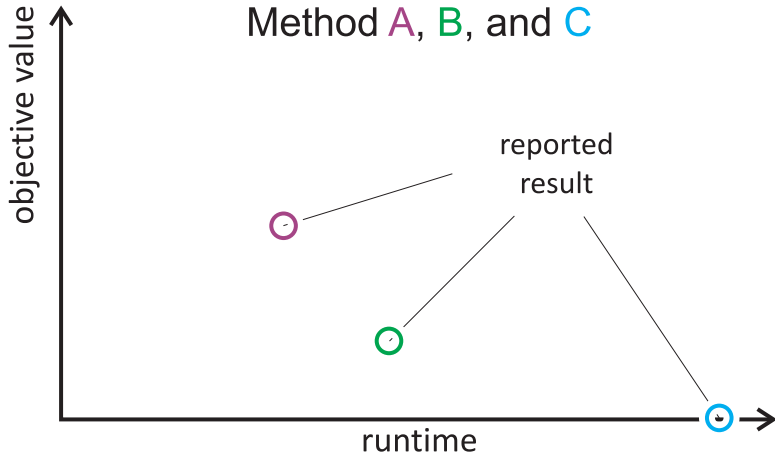
- Literature usually reports tuples “(instance, result, runtime)”
- Papers often use a different termination criterion

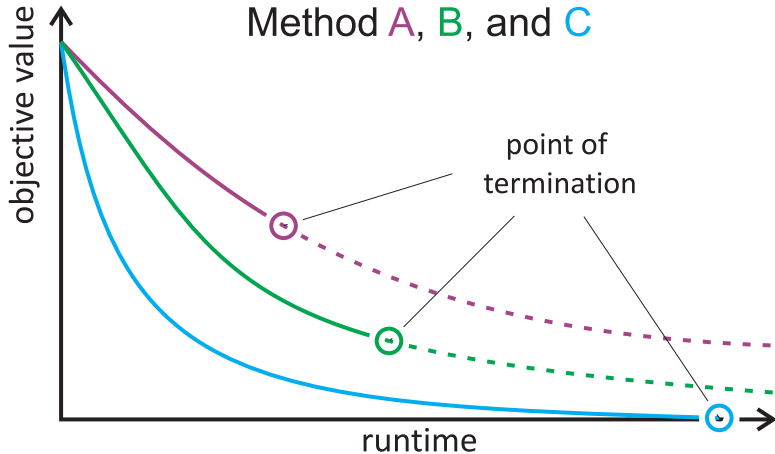


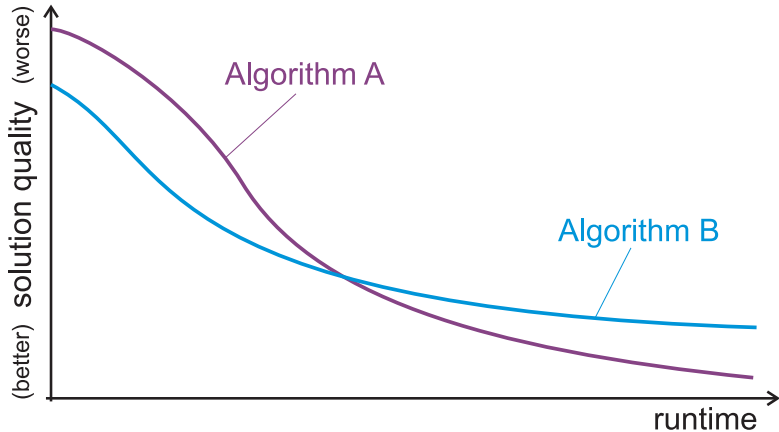
- Literature usually reports tuples “(instance, result, runtime)”
- **Problem:** Papers often use a different termination criterion

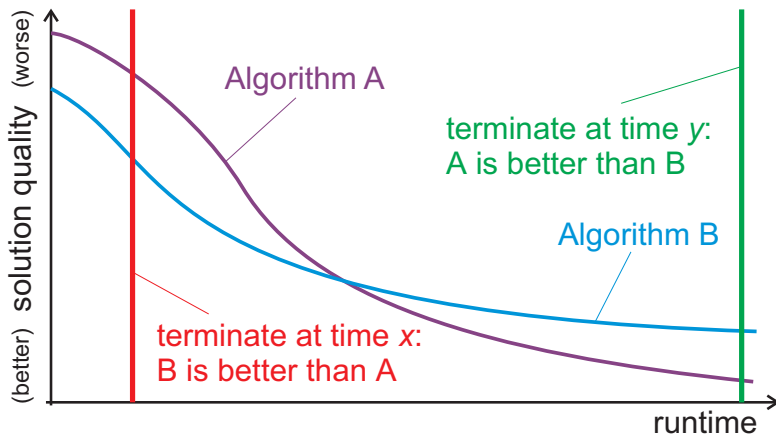
- Literature usually reports tuples “(instance, result, runtime)”
- Problem: Papers often use a different termination criterion
- Anytime Algorithms ^[23]

- Literature usually reports tuples “(instance, result, runtime)”
- Problem: Papers often use a different termination criterion
- Anytime Algorithms ^[23]: Always have approximate solution, refine it iteratively









- Literature usually reports tuples “(instance, result, runtime)”
- Problem: Papers often use a different termination criterion
- Anytime Algorithms ^[23]: Always have approximate solution, refine it iteratively
- One measure point per run or instance does not tell the whole story!

- Literature usually reports tuples “(instance, result, runtime)”
- Problem: Papers often use a different termination criterion
- Anytime Algorithms ^[23]: Always have approximate solution, refine it iteratively
- One measure point per run or instance does not tell the whole story!
- Using statistical tests cannot solve this issue (still: at one point in time).

- Literature usually reports tuples “(instance, result, runtime)”
- Problem: Papers often use a different termination criterion
- Anytime Algorithms ^[23]: Always have approximate solution, refine it iteratively
- One measure point per run or instance does not tell the whole story!
- Using statistical tests cannot solve this issue (still: at one point in time).
- We Should have the “whole curves”!

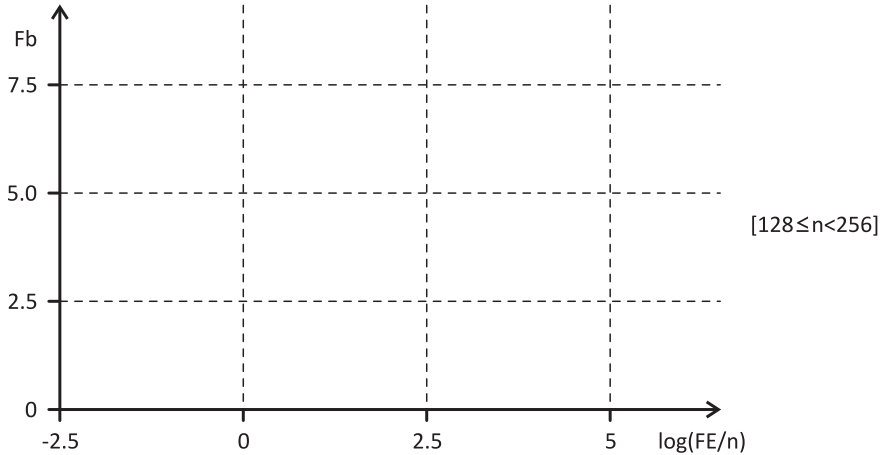
- Literature usually reports tuples “(instance, result, runtime)”
- Problem: Papers often use a different termination criterion
- Anytime Algorithms ^[23]: Always have approximate solution, refine it iteratively
- One measure point per run or instance does not tell the whole story!
- Using statistical tests cannot solve this issue (still: at one point in time).
- We Should have the “whole curves”! . . . ideally median curves over several runs!

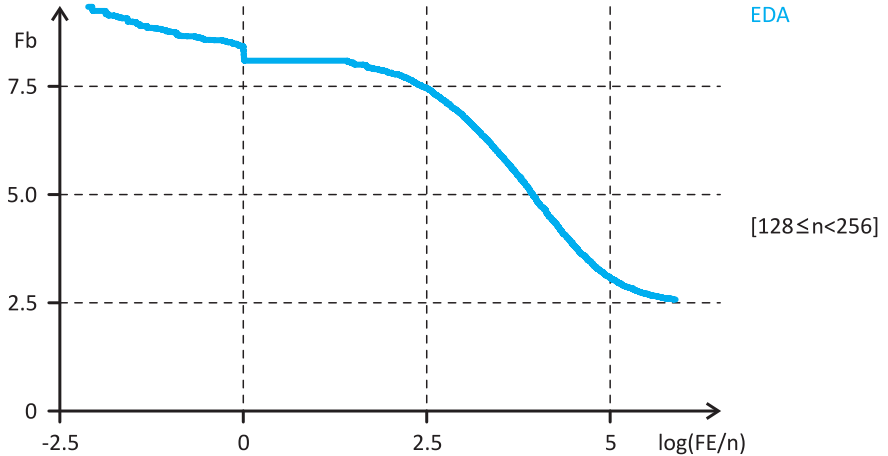
- Plot the best objective value reached over time

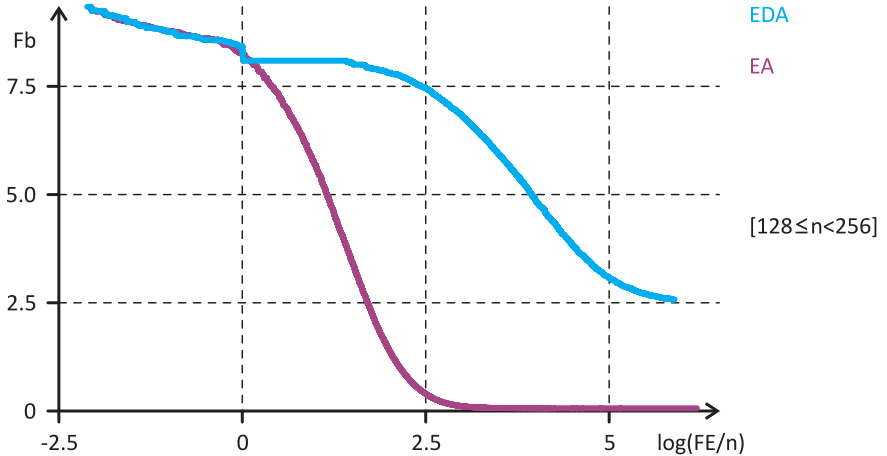
- Plot the **median** of the best objective value reached over time, over all runs

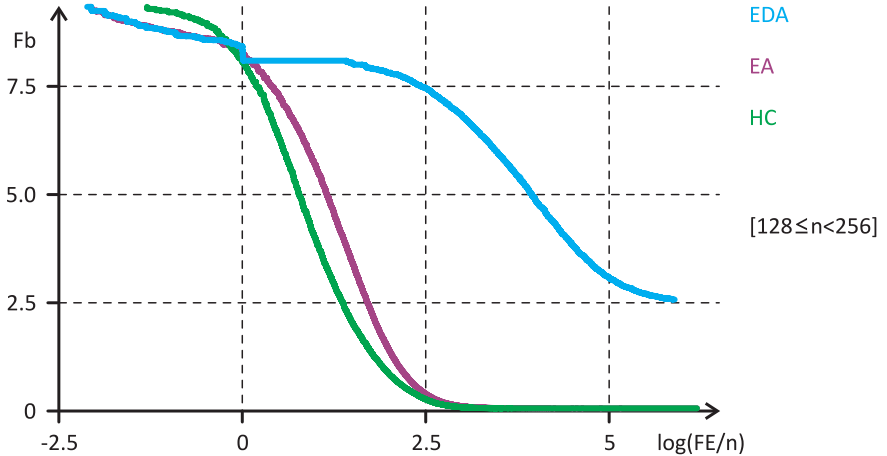
- Plot the **median** of the best objective value reached over time, over all runs, on a given benchmark instance

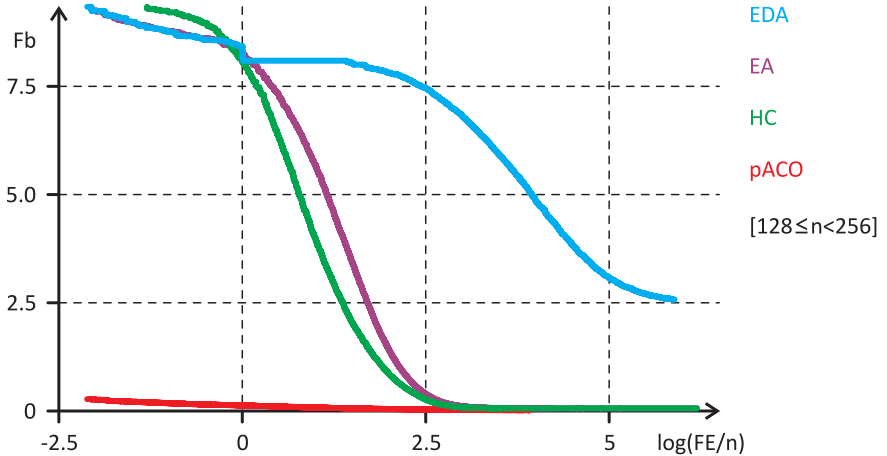
- Plot the **median** of the best objective value reached over time, over all runs, on a given benchmark instance or aggregated over several instances











- Plot the **median** of the best objective value reached over time, over all runs, on a given benchmark instance or aggregated over several instances
- The smaller the value, the better

- 1 Introduction
- 2 Performace Indicators
- 3 Statistical Measures
- 4 Statistical Comparisons
- 5 Testing is Not Enough
- 6 Summary**

- The optimization algorithms we consider in this lecture are **randomized**.

- The optimization algorithms we consider in this lecture are randomized.
- Comparing them must be done in a **statistical way** using data from **multiple runs**

- The optimization algorithms we consider in this lecture are randomized.
- Comparing them must be done in a statistical way using data from multiple runs
- Two key performance indicators

- The optimization algorithms we consider in this lecture are randomized.
- Comparing them must be done in a statistical way using data from multiple runs
- Two key performance indicators:
 - ① best result after fixed number of FEs/runtime

- The optimization algorithms we consider in this lecture are randomized.
- Comparing them must be done in a statistical way using data from multiple runs
- Two key performance indicators:
 - ① best result after fixed number of FEs/runtime
 - ② number of FEs/runtime needed to get certain result

- The optimization algorithms we consider in this lecture are randomized.
- Comparing them must be done in a statistical way using data from multiple runs
- Two key performance indicators:
 - ① best result after fixed number of FEs/runtime
 - ② number of FEs/runtime needed to get certain result
- For every single algorithm/configuration, compute

- The optimization algorithms we consider in this lecture are randomized.
- Comparing them must be done in a statistical way using data from multiple runs
- Two key performance indicators:
 - ① best result after fixed number of FEs/runtime
 - ② number of FEs/runtime needed to get certain result
- For every single algorithm/configuration, compute:
 - ① median of key performance indicators

- The optimization algorithms we consider in this lecture are randomized.
- Comparing them must be done in a statistical way using data from multiple runs
- Two key performance indicators:
 - ① best result after fixed number of FEs/runtime
 - ② number of FEs/runtime needed to get certain result
- For every single algorithm/configuration, compute:
 - ① median of key performance indicators
 - ② quartiles or top/bottom 1% quantile

- The optimization algorithms we consider in this lecture are randomized.
- Comparing them must be done in a statistical way using data from multiple runs
- Two key performance indicators:
 - ① best result after fixed number of FEs/runtime
 - ② number of FEs/runtime needed to get certain result
- For every single algorithm/configuration, compute:
 - ① median of key performance indicators
 - ② quartiles or top/bottom 1% quantile
 - ③ don't trust arithmetic mean or standard deviation

- The optimization algorithms we consider in this lecture are randomized.
- Comparing them must be done in a statistical way using data from multiple runs
- Two key performance indicators:
 - ① best result after fixed number of FEs/runtime
 - ② number of FEs/runtime needed to get certain result
- For every single algorithm/configuration, compute:
 - ① median of key performance indicators
 - ② quartiles or top/bottom 1% quantile
 - ③ don't trust arithmetic mean or standard deviation
- Do not only collect one data sample per run, try to plot progress curves

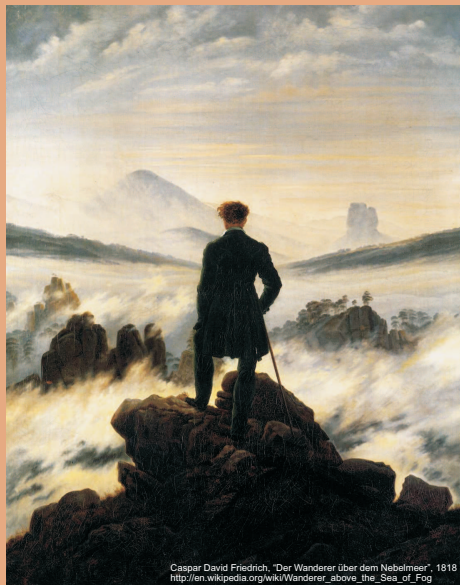
- The optimization algorithms we consider in this lecture are randomized.
- Comparing them must be done in a statistical way using data from multiple runs
- Two key performance indicators:
 - ① best result after fixed number of FEs/runtime
 - ② number of FEs/runtime needed to get certain result
- For every single algorithm/configuration, compute:
 - ① median of key performance indicators
 - ② quartiles or top/bottom 1% quantile
 - ③ don't trust arithmetic mean or standard deviation
- Do not only collect one data sample per run, try to plot progress curves
- For given problem class: Look for well-known benchmarks!

谢谢

Thank you

Thomas Weise [汤卫思]
tweise@hfu.edu.cn
<http://iao.hfu.edu.cn>

Hefei University, South Campus 2
Institute of Applied Optimization
Shushan District, Hefei, Anhui,
China





1. Thomas Weise. *An Introduction to Optimization Algorithms*. Institute of Applied Optimization (IAO), Faculty of Computer Science and Technology, Hefei University, Hefei, Anhui, China, 2019-06-25 edition, 2018–2019. URL <http://thomasweise.github.io/aitoa/>. see also ^[10].
2. Nikolaus Hansen, Anne Auger, Steffen Finck, and Raymond Ros. Real-parameter black-box optimization benchmarking 2010: Experimental setup. *Rapports de Recherche RR-7215*, Institut National de Recherche en Informatique et en Automatique (INRIA), March 9 2010. URL <https://hal.inria.fr/inria-00462481>. inria-00462481.
3. Steffen Finck, Nikolaus Hansen, Raymond Ros, and Anne Auger. Coco documentation, release 15.03, November 17 2015. URL <http://coco.lri.fr/COCOdoc/COCO.pdf>.
4. Thomas Weise, Li Niu, and Ke Tang. AOAB – automated optimization algorithm benchmarking. In *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO'10), July 7–11, 2010, Portland, OR, USA*, pages 1479–1486, New York, NY, USA, 2010. ACM Press. doi: 10.1145/1830761.1830763.
5. Thomas Weise, Xiaofeng Wang, Qi Qi, Bin Li, and Ke Tang. Automatically discovering clusters of algorithm and problem instance behaviors as well as their causes from experimental data, algorithm setups, and instance features. *Applied Soft Computing Journal (ASOC)*, 73:366–382, December 2018. doi: 10.1016/j.asoc.2018.08.030.
6. Ke Tang, Xiaodong Li, Ponnuthurai Nagarathnam Suganthan, Zhenyu Yang, and Thomas Weise. Benchmark functions for the cec'2010 special session and competition on large-scale global optimization. Technical report, University of Science and Technology of China (USTC), School of Computer Science and Technology, Nature Inspired Computation and Applications Laboratory (NICAL), Hefei, Anhui, China, January 8 2010.
7. Thomas Weise, Raymond Chiong, Ke Tang, Jörg Lässig, Shigeyoshi Tsutsui, Wenxiang Chen, Zbigniew Michalewicz, and Xin Yao. Benchmarking optimization algorithms: An open source framework for the traveling salesman problem. *IEEE Computational Intelligence Magazine (CIM)*, 9:40–52, August 2014. doi: 10.1109/MCI.2014.2326101.
8. Frank E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11:1–21, 1969. doi: 10.1080/00401706.1969.10490657.
9. Gangadharrao Soundalayarao Maddala. *Introduction to Econometrics*. MacMillan, New York, NY, USA, second edition, 1992. ISBN 978-0-02-374545-4.
10. Thomas Weise. *Global Optimization Algorithms – Theory and Application*. it-weise.de (self-published), Germany, 2009. URL <http://www.it-weise.de/projects/book.pdf>.
11. Uwe E. Reinhardt. What does 'economic growth' mean for americans? *The New York Times, Economix, Today's Economist*, September 2, 2011. URL <https://economix.blogs.nytimes.com/2011/09/02/what-does-economic-growth-mean-for-americans>.

12. Jürgen Bortz, Gustav Adolf Lienert, and Klaus Boehnke. *Verteilungsfreie Methoden in der Biostatistik*. Springer-Lehrbuch. Springer Medizin Verlag, Heidelberg, Germany, 3 edition, 2008. ISBN 3445110344. doi: 10.1007/978-3-540-74707-9.
13. Eugene S. Edgington. *Randomization Tests*. CRC Press, Inc., Boca Raton, FL, USA, 3 edition, 1995. ISBN 0824796691 and 9780824796693.
14. Shaun Burke. Missing values, outliers, robust statistics & non-parametric methods. *LC.GC Europe Online Supplement*, 1 (2):19–24, January 2001.
15. Daniel F. Bauer. Constructing confidence sets using rank statistics. *Journal of the American Statistical Association (J AM STAT ASSOC)*, 67:687–690, September 1972. doi: 10.1080/01621459.1972.10481279.
16. Sidney Siegel and N. John Castellan Jr. *Nonparametric Statistics for The Behavioral Sciences*. Humanities/Social Sciences/Languages. McGraw-Hill, New York, NY, USA, 1988. ISBN 0-07-057357-3.
17. Myles Hollander and Douglas Alan Wolfe. *Nonparametric Statistical Methods*. John Wiley & Sons, New York, USA, 1973. ISBN 047140635X.
18. Henry B. Mann and Donald R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics (AOMS)*, 18(1):50–60, March 1947. doi: 10.1214/aoms/1177730491. URL <http://projecteuclid.org/euclid.aoms/1177730491>.
19. Sir Ronald Aylmer Fisher. On the interpretation of χ^2 from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society*, 85:87–94, 1922. URL <http://hdl.handle.net/2440/15173>.
20. Lorenz Gygax. Statistik für Nutztierethologen – Einführung in die statistische Denkweise: Was ist, was macht ein statistischer Test?, June 2003. URL <http://www.proximate-biology.ch/documents/introEtho.pdf>.
21. Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, December 1945. URL <http://sci2s.ugr.es/keel/pdf/algorithm/articulo/wilcoxon1945.pdf>.
22. Olive Jean Dunn. Multiple comparisons among means. *Journal of the American Statistical Association (J AM STAT ASSOC)*, 56(293):52–64, March 1961. doi: 10.1080/01621459.1961.10482090.
23. Mark S. Boddy and Thomas L. Dean. Solving time-dependent planning problems. Technical Report CS-89-03, Brown University, Department of Computer Science, Providence, RI, USA, February 1989. URL <ftp://ftp.cs.brown.edu/pub/techreports/89/cs89-03.pdf>.