# Optimization Algorithms

## A. Comparing Optimization Algorithms

Thomas Weise · 汤卫思
tweise@hfuu.edu.cn · http://iao.hfuu.edu.cn/5

Institute of Applied Optimization (IAO) | 应用优化研究所
School of Artificial Intelligence and Big Data | 人工智能与大数据学院
Hefei University | 合肥学院
Hefei, Anhui, China | 中国安徽省合肥市

## Outline

1. Introduction
2. Performance Indicators and Time
3. Statistical Measures
4. Measures of Spread

# Introduction

# Introduction

- There are many optimization algorithms

# Introduction

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

# Introduction

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

# Performance Indicators and Time

# Performance Indicators

- Key parameters[3–6]

# Performance Indicators

- Key parameters[3–6]:
  1. Solution quality reached after a certain runtime

# Performance Indicators

- Two key parameter[3–6]:
  1. Solution quality reached after a certain runtime
  2. Runtime to reach a certain solution quality

# Performance Indicators

- Two key parameter[3–6]:
  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.

# Runtime)

- What actually is runtime?

## Absolute Runtime

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

# Absolute Runtime

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

- Advantages

## Absolute Runtime

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

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

## Absolute Runtime

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

# Absolute Runtime

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

# Absolute Runtime

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

## Absolute Runtime

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

## Absolute Runtime

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 10 ms: many things seem to happen at the same time

# Absolute Runtime

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 10 ms: many things seem to happen at the same time
  - Can be biased by "outside effects," e.g., OS, scheduling, other processes, I/O, swapping, ...

## Absolute Runtime

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 10 ms: 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

## Absolute Runtime

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 10 ms: 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...

## Absolute Runtime

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 10 ms: 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...
- ...so for research they may be less interesting, while for a specific application they do matter.

## Function Evaluations: FEs

Measure the number of fully constructed and tested candidate solutions

## Function Evaluations: FEs

Measure the number of fully constructed and tested candidate solutions

- Advantages

## Function Evaluations: FEs

Measure the number of fully constructed and tested candidate solutions

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

## Function Evaluations: FEs

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

## Function Evaluations: FEs

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”

## Function Evaluations: FEs

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

## Function Evaluations: FEs

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

## Function Evaluations: FEs

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

## Function Evaluations: FEs

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

## Function Evaluations: FEs

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!

## Function Evaluations: FEs

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!
- Relevant for comparing algorithms, but not so much for the practical application

# Runtime

- Rewrite the two key parameters by choosing a time measure[3 5]

# Runtime

- Rewrite the two key parameters by choosing a time measure[3 5]:
    1. Solution quality reached after a certain number of FEs

# Runtime

- Rewrite the two key parameters by choosing a time measure[3 5]:
    1. Solution quality reached after a certain number of FEs
    2. Number FEs needed to reach a certain solution quality

## Solution Quality

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

# Solution Quality

- Common measure of solution quality: Objective function value of best solution discovered.
- Rewrite the two key parameters[3] [5]

## Solution Quality

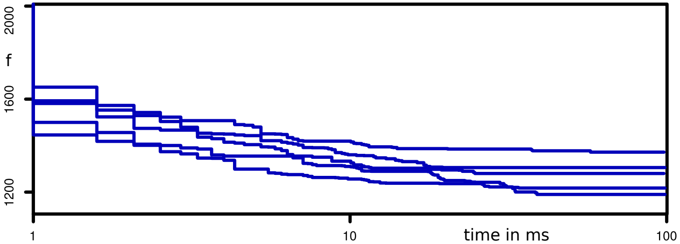
- Common measure of solution quality: Objective function value of best solution discovered.
- Rewrite the two key parameters[3 5]:
    1. Best objective function value reached after a certain number of FEs

# Solution Quality

- Common measure of solution quality: Objective function value of best solution discovered.
- Rewrite the two key parameters[3] [5]:
  1. Best objective function value reached after a certain number of FEs
  2. Number FEs needed to reach a certain objective function value

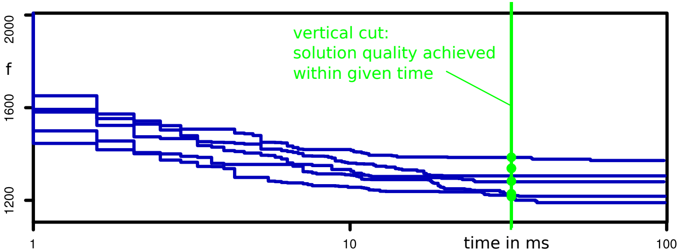
# Key Parameters

- Which one is the “better” performance indicator?



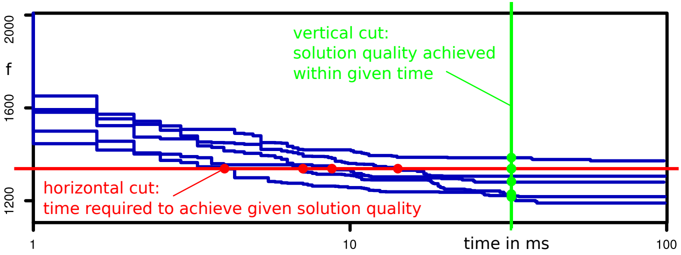
# Key Parameters

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



# Key Parameters

- 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



## Key Parameters

- 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
- This question actually does not really need an answer...

## Which Indicator is better?

- Number FEs needed to reach a certain objective function value
- Preferred by, e.g., the BBOB/COCO benchmark suite[3]

## Which Indicator is better?

- Number FEs needed to reach a certain objective function value
- Preferred by, e.g., the BBOB/COCO benchmark suite[3]:
  - Measures a time needed to reach a target function value ⇒ "Algorithm $A$ is two/ten/hundred times faster than Algorithm $B$ in solving this problem."

## Which Indicator is better?

- Number FEs needed to reach a certain objective function value
- Preferred by, e.g., the BBOB/COCO benchmark suite[3]:
  - 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 $B$.

## Which Indicator is better?

- 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[7].

# Which Indicator is better?

- Best objective function value reached after a certain number of FEs
- Preferred by many benchmark suites such as[7].
- Practice Perspective: Best results achievable with given time budget wins.

## Which Indicator is better?

- Best objective function value reached after a certain number of FEs
- Preferred by many benchmark suites such as[7].
- Practice Perspective: Best results achievable with given time budget wins.
- This perspective maybe less suitable for benchmarking, but surely true in practice.

## Which Indicator is better?

- Best objective function value reached after a certain number of FEs
- Preferred by many benchmark suites such as[7].
- 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.

## Key Parameters

- No official consensus on which view is "better."

## Key Parameters

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

## Key Parameters

- No official consensus on which view is "better."
- This also strongly depends on the situation.
- Best approach: Evaluate algorithm according to both methods.[5] [6] [8]

## Determining Target Values

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

# Determining Target Values

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

# Determining Target Values

- 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.

# Determining Target Values

- 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.

## Determining Target Values

- 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.

## Determining Target Values

- 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

# Statistical Measures

# Randomized Algorithms

- Special situation: Randomized Algorithms

## Randomized Algorithms

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

## Randomized Algorithms

- 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.

# Randomized Algorithms

- 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!

## Randomized Algorithms

- 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 a randomized algorithm one time does not give reliable information.

# Randomized Algorithms

- 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 a randomized algorithm one time does not give reliable information.
- Statistical evaluation over a set of runs necessary.

## Important Distinction

- Crucial Difference: distribution and sample

# Important Distinction

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

## Important Distinction

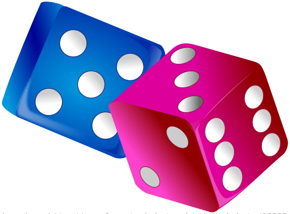
- Crucial Difference: distribution and sample
- A sample is what we *measure*.
- A distribution is the asymptotic result of the ideal process

# Important Distinction

- 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

# Important Distinction

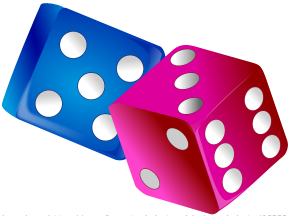
- Crucial Difference: distribution and sample
- A sample is what we *measure*.
- A distribution is the asymptotic result of the ideal process
- Statistical parameters of the distribution can be estimated from a sample
- Example: Dice Throw



based on http://www.freestockphotos.biz/stockphoto/16223

## Important Distinction

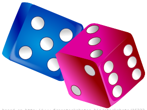
- Crucial Difference: distribution and sample
- A sample is what we *measure*.
- A distribution is the asymptotic result of the ideal process
- Statistical parameters of the distribution can be estimated from a sample
- Example: Dice Throw
- How likely is it to roll a 1, 2, 3, 4, 5, or 6?



based on http://www.freestockphotos.biz/stockphoto/16223

# Important Distinction

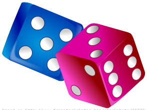
| # throws | number | f(1) | f(2) | f(3) | f(4) | f(5) | f(6) |
|---|---|---|---|---|---|---|---|
| 1 | 5 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 1.0000 | 0.0000 |



based on http://www.freestockphotos.biz/stockphoto/16223

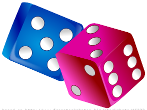
# Important Distinction

| # 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 |



# Important Distinction

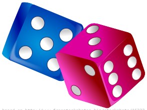
| # throws | number | f(1) | f(2) | f(3) | f(4) | f(5) | f(6) |
|---|---|---|---|---|---|---|---|
| 1 | 5 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 1.0000 | 0.0000 |
| 2 | 4 | 0.0000 | 0.0000 | 0.0000 | 0.5000 | 0.5000 | 0.0000 |
| 3 | 1 | 0.3333 | 0.0000 | 0.0000 | 0.3333 | 0.3333 | 0.0000 |



based on http://www.freestockphotos.biz/stockphoto/16223

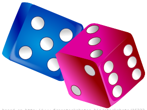
# Important Distinction

| # 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 |



# Important Distinction

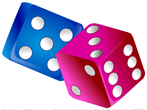
| # throws | number | f(1) | f(2) | f(3) | f(4) | f(5) | f(6) |
|---|---|---|---|---|---|---|---|
| 1 | 5 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 1.0000 | 0.0000 |
| 2 | 4 | 0.0000 | 0.0000 | 0.0000 | 0.5000 | 0.5000 | 0.0000 |
| 3 | 1 | 0.3333 | 0.0000 | 0.0000 | 0.3333 | 0.3333 | 0.0000 |
| 4 | 4 | 0.2500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



based on http://www.freestockphotos.biz/stockphoto/16223

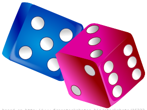
# Important Distinction

| # 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 |



# Important Distinction

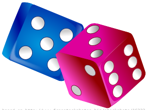
| # throws | number | f(1) | f(2) | f(3) | f(4) | f(5) | f(6) |
|---|---|---|---|---|---|---|---|
| 1 | 5 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 1.0000 | 0.0000 |
| 2 | 4 | 0.0000 | 0.0000 | 0.0000 | 0.5000 | 0.5000 | 0.0000 |
| 3 | 1 | 0.3333 | 0.0000 | 0.0000 | 0.3333 | 0.3333 | 0.0000 |
| 4 | 4 | 0.2500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



based on http://www.freestockphotos.biz/stockphoto/16223

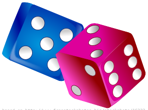
# Important Distinction

| # 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 |



# Important Distinction

| # 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 |



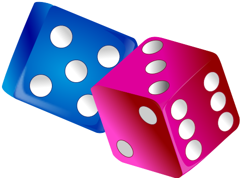
based on http://www.freestockphotos.biz/stockphoto/16223

# Important Distinction

| # 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 |

# Important Distinction

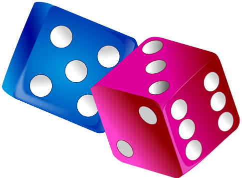
- Crucial Difference: distribution and sample
- A sample is what we *measure*.
- A distribution is the asymptotic result of the ideal process
- Statistical parameters of the distribution can be estimated from a sample
- Example: Dice Throw
- How likely is it to roll a 1, 2, 3, 4, 5, or 6?
- Never forget: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



based on http://www.freestockphotos.biz/stockphoto/16223

## Important Distinction

- 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



based on http://www.freestockphotos.biz/stockphoto/16223

# Measures of the Average

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

# Measures of the Average

- 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.

# Measures of the Average

- 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 reduce this set of numbers to a single value which can give us an impression of what the "average outcome" (or result quality is).

## Measures of the Average

- 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 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.

# Arithmetic Mean

## Definition (Arithmetic Mean)

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

# Arithmetic Mean

## Definition (Arithmetic Mean)

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

# Arithmetic Mean

## Definition (Arithmetic Mean)

The arithmetic mean $\mathrm{mean}(A)$ 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.

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

# Median

## Definition (Median)

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

# Median

## Definition (Median)

The median $\mathrm{med}(A)$ 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 $A = (a_0, a_1, \ldots, a_{n-1})$ where $a_{i-1} \leq a_i \; \forall i \in 1 \ldots (n-1)$.

## Median

### Definition (Median)

The median $\mathrm{med}(A)$ 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 $A = (a_0, a_1, \ldots, a_{n-1})$ where $a_{i-1} \leq a_i \; \forall i \in 1 \ldots (n-1)$.

$$\mathrm{med}(A) = \begin{cases} a_{\frac{n-1}{2}} & \text{if } n \text{ is odd} \\ \frac{1}{2}\left(a_{\frac{n}{2}-1} + a_{\frac{n}{2}}\right) & \text{otherwise} \end{cases} \quad \text{if } a_{i-1} \leq a_i \; \forall i \in 1 \ldots (n-1) \tag{2}$$

# Outliers

- Sometimes the data contains outliers[9] [10].

## Outliers

- Sometimes the data contains outliers[9 10], i.e., observations which are much different from the other measurements.

# Outliers

- Sometimes the data contains outliers[9] [10], 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.

# Outliers

- Sometimes the data contains outliers[9 10], 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.
- They may also represent measurement errors or observations which have been been disturbed by unusual effects.

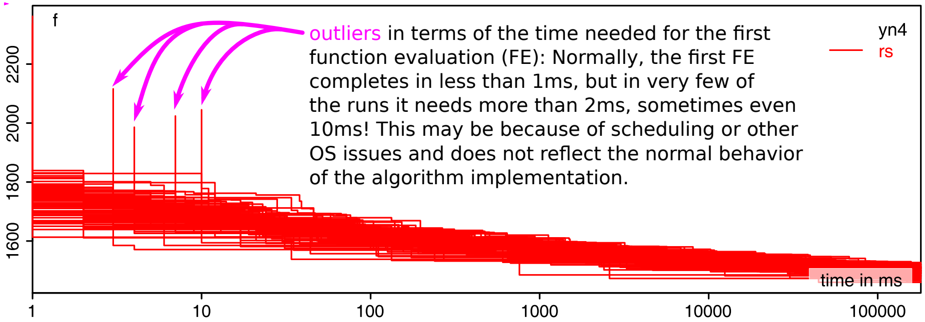
## Outliers

- Sometimes the data contains outliers[9 10], 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.
- They may also represent measurement errors or observations which have been 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.

## Outliers

- Sometimes the data contains outliers[9] [10], 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.
- They may also represent measurement errors or observations which have been 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.

# Outliers


f
outliers in terms of the time needed for the first function evaluation (FE): Normally, the first FE completes in less than 1ms, but in very few of the runs it needs more than 2ms, sometimes even 10ms! This may be because of scheduling or other OS issues and does not reflect the normal behavior of the algorithm implementation.
yn4
rs
2200
2000
1800
1600
time in ms
1
10
100
1000
10000
100000

# Outliers

# Example for Data Samples w/o Outlier

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

$$
\begin{aligned}
A &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14) \\
B &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, {\color{red}10'008})
\end{aligned}
$$

## Example for Data Samples w/o Outlier

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

$$\begin{aligned} A &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14) \\ B &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10'008) \end{aligned}$$

- We find that

## Example for Data Samples w/o Outlier

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

$$\begin{aligned} A &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14) \\ B &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10'008) \end{aligned}$$

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

## Example for Data Samples w/o Outlier

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

$$
\begin{aligned}
A &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14) \\
B &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10'008)
\end{aligned}
$$

- We find that
  - $\mathrm{mean}(A) = \frac{1}{19} \sum_{i=0}^{18} a_i = \frac{133}{19} = 7$ and
  - $\mathrm{mean}(B) = \frac{1}{19} \sum_{i=0}^{18} b_i = \frac{10'127}{19} = 553$

# Example for Data Samples w/o Outlier

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

$$\begin{aligned} A &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14) \\ B &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, \textcolor{red}{10'008}) \end{aligned}$$

- We find that
  - $\mathrm{mean}(A) = \frac{1}{19}\sum_{i=0}^{18} a_i = \frac{133}{19} = 7$ and
  - $\mathrm{mean}(B) = \frac{1}{19}\sum_{i=0}^{18} b_i = \frac{10'127}{19} = 553$, while
  - $\mathrm{med}(A) = a_9 = 6$

# Example for Data Samples w/o Outlier

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

$$\begin{aligned} A &= (1,3,4,4,4,5,6,6,6,6,7,7,9,9,9,10,11,12,14) \\ B &= (1,3,4,4,4,5,6,6,6,6,7,7,9,9,9,10,11,12,10'008) \end{aligned}$$

- We find that
  - $\mathrm{mean}(A) = \frac{1}{19}\sum_{i=0}^{18} a_i = \frac{133}{19} = 7$ and
  - $\mathrm{mean}(B) = \frac{1}{19}\sum_{i=0}^{18} b_i = \frac{10'127}{19} = 553$, while
  - $\mathrm{med}(A) = a_9 = 6$ and
  - $\mathrm{med}(B) = b_9 = 6$.

## Why are outliers important?

- If you think about, where could outliers in our experiments come from?

## Why are outliers important?

- If you think about, where could outliers in our experiments come from?
    1. The operating systems scheduling or other strange effects could mess with our timing.

## Why are outliers important?

- If you think about, where could outliers in our experiments come from?
  1. The operating systems scheduling or other strange effects could mess with our timing.
  2. This could cause worse results.

## Why are outliers important?

- If you think about, where could outliers in our experiments come from?
    1. The operating systems scheduling or other strange effects could mess with our timing.
    2. This could cause worse results.
    3. But this is already it.

## Why are outliers important?

- If you think about, where could outliers in our experiments come from?
    1. The operating systems scheduling or other strange effects could mess with our timing.
    2. This could cause worse results.
    3. But this is already it. There are hardly any other "outside" effects that could mess up our results!

## Why are outliers important?

- If you think about, where could outliers in our experiments come from?
  1. The operating systems scheduling or other strange effects could mess with our timing.
  2. This could cause worse results.
  3. But this is already it. There are hardly any other "outside" effects that could mess up our results!
  4. Instead, there could be: bugs in our code!

## Why are outliers important?

- If you think about, where could outliers in our experiments come from?
  1. The operating systems scheduling or other strange effects could mess with our timing.
  2. This could cause worse results.
  3. But this is already it. There are hardly any other “outside” effects that could mess up our results!
  4. Instead, there could be: bugs in our code!
  5. Or: bad worst-case behaviors of our algorithm!

## Why are outliers important?

- If you think about, where could outliers in our experiments come from?
  1. The operating systems scheduling or other strange effects could mess with our timing.
  2. This could cause worse results.
  3. But this is already it. There are hardly any other "outside" effects that could mess up our results!
  4. Instead, there could be: bugs in our code!
  5. Or: bad worst-case behaviors of our algorithm!
- Thus, we often want that outliers influence our statistics.

## Mean vs. Median

- In our application scenarios, there are very few acceptable reasons for outliers.

## Mean vs. Median

- In our application scenarios, there are very few acceptable reasons for outliers.
- We therefore want to know the arithmetic mean.

## Mean vs. Median

- In our application scenarios, there are very few acceptable reasons for outliers.
- We therefore want to know the arithmetic mean.
- We also want to know the median, because it shows us what we can normally expect as results.

## Mean vs. Median

- In our application scenarios, there are very few acceptable reasons for outliers.
- We therefore want to know the arithmetic mean.
- We also want to know the median, because it shows us what we can normally expect as results.
- If the arithmetic mean and median are very different, then

# Mean vs. Median

- In our application scenarios, there are very few acceptable reasons for outliers.
- We therefore want to know the arithmetic mean.
- We also want to know the median, because it shows us what we can normally expect as results.
- If the arithmetic mean and median are very different, then
  - maybe we have a bug in our code that only sometimes has an impact.

## Mean vs. Median

- In our application scenarios, there are very few acceptable reasons for outliers.
- We therefore want to know the arithmetic mean.
- We also want to know the median, because it shows us what we can normally expect as results.
- If the arithmetic mean and median are very different, then
  - maybe we have a bug in our code that only sometimes has an impact or
  - our algorithm has bad worst-case behavior (which is also good to know).

# Mean vs. Median

- In our application scenarios, there are very few acceptable reasons for outliers.
- We therefore want to know the arithmetic mean.
- We also want to know the median, because it shows us what we can normally expect as results.
- If the arithmetic mean and median are very different, then
  - maybe we have a bug in our code that only sometimes has an impact or
  - our algorithm has bad worst-case behavior (which is also good to know).
- So we can conclude: It is best to have both the mean and median statistic of a given performance indicator.

# Measures of Spread

# Measures of Spread

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

# Measures of Spread

- 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.

## Measures of Spread

- 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.

## Measures of Spread

- 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.

## Variance

### Definition (Variance)

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

# Variance

## Definition (Variance)

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

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

## Standard Deviation

### Definition (Standard Deviation)

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

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

## Standard Deviation

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

## Standard Deviation

- 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.

## Standard Deviation

- 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.

# Standard Deviation

- 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

## Standard Deviation

- 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

## Standard Deviation

- 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*)

# Quantiles

## Definition (Quantile)

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

## Quantiles

### Definition (Quantile)

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

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

## Quantiles

### Definition (Quantile)

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

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

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

## Quantiles

### Definition (Quantile)

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

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

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

# Quantiles

## Definition (Quantile)

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

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

- The $\text{quantile}_1^2 A$ is the median of $A$
- 4-quantiles are called quartiles.
- We sometimes write things like "the 25% quantile," meaning $\text{quantile}_{100}^{25}$.

## Standard Deviation: Example

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

$$\begin{aligned} A &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14) \\ \text{mean}(A) &= 7 \\ B &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, \color{red}{10'008}) \\ \text{mean}(B) &= 533 \end{aligned}$$

## Standard Deviation: Example

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

$$
\begin{aligned}
A &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14) \\
\mathrm{mean}(A) &= 7 \\
B &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, {\color{red}10'008}) \\
\mathrm{mean}(B) &= 533 \\
\mathrm{var}(A) &= \frac{1}{19-1} \sum_{i=1}^{19} (a_i - \mathrm{mean}(a))^2 = \frac{198}{18} = 11 \\
\mathrm{var}(B) &= \frac{1}{19-1} \sum_{i=1}^{19} (b_i - \mathrm{mean}(b))^2 = \frac{94'763'306}{18} \approx 5'264'628.1
\end{aligned}
$$

## Standard Deviation: Example

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

$$
\begin{aligned}
A &= (1,3,4,4,4,5,6,6,6,6,7,7,9,9,9,10,11,12,14) \\
\text{mean}(A) &= 7 \\
B &= (1,3,4,4,4,5,6,6,6,6,7,7,9,9,9,10,11,12,{\color{red}10'008}) \\
\text{mean}(B) &= 533 \\
\text{var}(A) &= \frac{1}{19-1}\sum_{i=1}^{19}(a_i - \text{mean}(a))^2 = \frac{198}{18} = 11 \\
\text{var}(B) &= \frac{1}{19-1}\sum_{i=1}^{19}(b_i - \text{mean}(b))^2 = \frac{94'763'306}{18} \approx 5'264'628.1 \\
\text{sd}(A) &= \sqrt{\text{var} A} = \sqrt{11} \approx 3.31662479 \\
\text{sd}(B) &= \sqrt{\text{var} B} = \sqrt{\frac{94'763'306}{18}} \approx 2294.477743
\end{aligned}
$$

## Quantiles: Example

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

$$A = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14)$$
$$B = (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10'008)$$

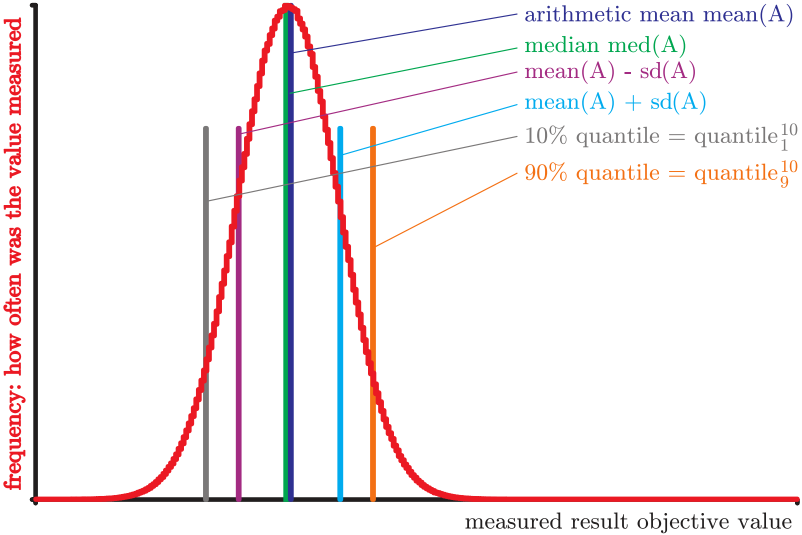
## Quantiles: Example

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

$$
\begin{aligned}
A &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 14) \\
B &= (1, 3, 4, 4, 4, 5, 6, 6, 6, 6, 7, 7, 9, 9, 9, 10, 11, 12, 10'008) \\
\text{quantile}_4^1(A) &= \text{quantile}_4^1(B) = 4.5 \\
\text{quantile}_4^3(A) &= \text{quantile}_4^3(B) = 9
\end{aligned}
$$

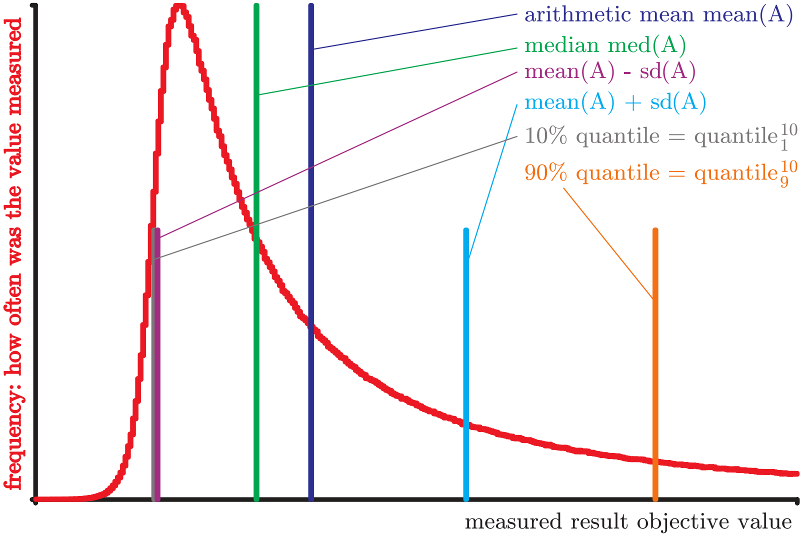
## Further Example

- The implicit assumption that $\mathrm{mean} \pm \mathrm{sd}$ is a meaningful range is not always true!



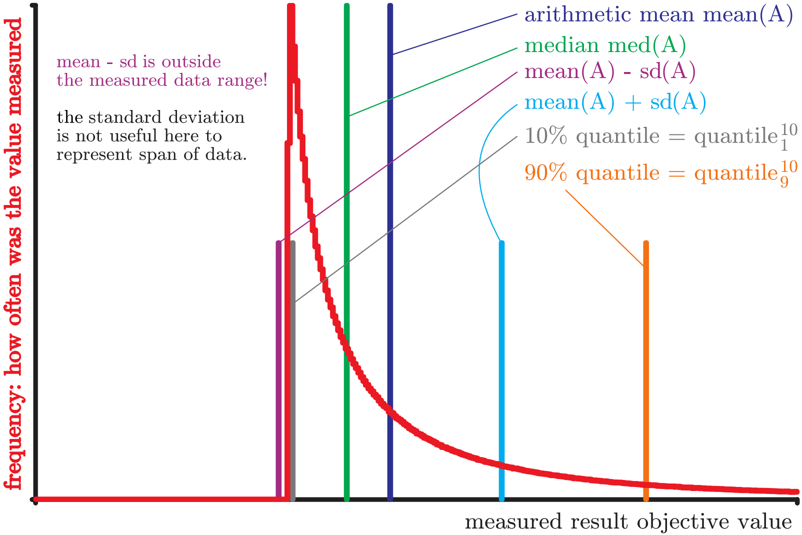
## Further Example

- The implicit assumption that $\mathrm{mean} \pm \mathrm{sd}$ is a meaningful range is not always true!



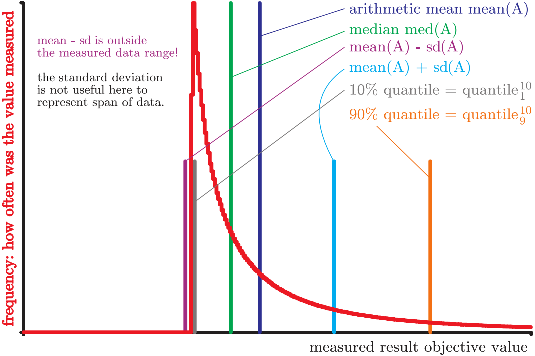
# Further Example

- The implicit assumption that $\mathrm{mean} \pm \mathrm{sd}$ is a meaningful range is not always true!



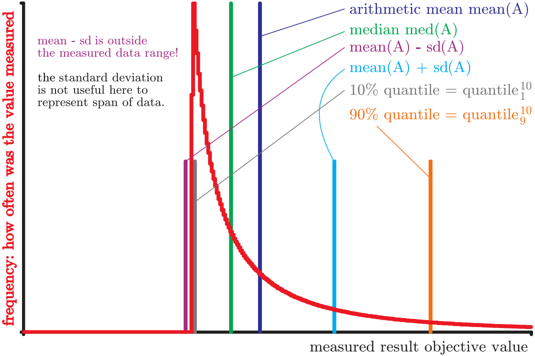
# Further Example

- The implicit assumption that $\mathrm{mean} \pm \mathrm{sd}$ is a meaningful range is not always true!
- Such a shape is possible in optimization!



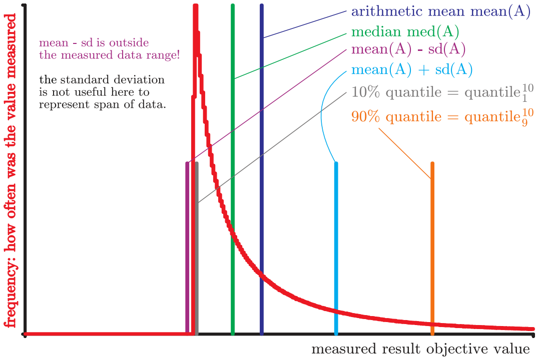
# Further Example

- The implicit assumption that $\mathrm{mean} \pm \mathrm{sd}$ is a meaningful range is not always true!
- Such a shape is possible in optimization:
  - The global optimum marks a lower bound for the possible objective values.



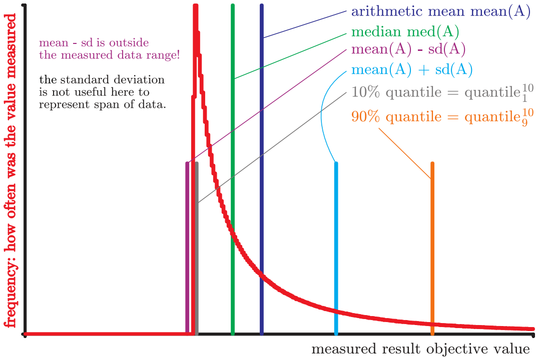
# Further Example

- The implicit assumption that $\mathrm{mean} \pm \mathrm{sd}$ is a meaningful range is not always true!
- Such a shape is possible in optimization:
  - The global optimum marks a lower bound for the possible objective values.
  - A good algorithm often returns results which are close-to-optimal.



# Further Example

- The implicit assumption that $\mathrm{mean} \pm \mathrm{sd}$ is a meaningful range is not always true!
- Such a shape is possible in optimization:
  - The global optimum marks a lower bound for the possible objective values.
  - A good algorithm often returns results which are close-to-optimal.
  - There may be a long tail of few but significantly worse runs.



谢谢

Thank you

# References I


1. Thomas Weise. *An Introduction to Optimization Algorithms*. Institute of Applied Optimization (IAO) [应用优化研究所] of the School of Artificial Intelligence and Big Data [人工智能与大数据学院] of Hefei University [合肥学院], Hefei [合肥市], Anhui [安徽省], China [中国], 2018–2020. URL http://thomasweise.github.io/aitoa/.
2. Thomas Weise. *Global Optimization Algorithms – Theory and Application*. it-weise.de (self-published), Germany, 2009. URL http://www.it-weise.de/projects/book.pdf.
3. 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 http://hal.inria.fr/inria-00462481. inria-00462481.
4. 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.
5. 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.
6. 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.
7. Ke Tang, Xiaodong Li, Ponnuthurai Nagaratnam 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.
8. 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.
9. Frank E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11:1–21, 1969. doi:10.1080/00401706.1969.10490657.
10. Gangadharrao Soundalyarao Maddala. *Introduction to Econometrics*. MacMillan, New York, NY, USA, second edition, 1992. ISBN 978-0-02-374545-4.