Wrocław University of Science and Technology

## Optimization Methods: Theory and Applications

# Optimization in continuous search spaces the basics and evolutionary strategies

**Michał Przewoźniczek**
Department of Systems and Computer Networks
Wroclaw University of Science and technology

Fundusze Europejskie Polska Cyfrowa

Rzeczpospolita Polska

Unia Europejska Europejski Fundusz Rozwoju Regionalnego

*Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego, Program Operacyjny Polska Cyfrowa na lata 2014-2020, Oś Priorytetowa nr 3 "Cyfrowe kompetencje społeczeństwa" Działanie nr 3.2 "Innowacyjne rozwiązania na rzecz aktywizacji cyfrowej" Tytuł projektu: „Akademia Innowacyjnych Zastosowań Technologii Cyfrowych (AI Tech)”*

# Optimization problems

## Definition

- The obejctive function for $n$-dimensional optimization problem:

$$f: D_f \subseteq R^n \rightarrow R$$

where: $D_f$ – solution search space

- Obejctive: find the best $\vec{x}^*$

$$\vec{x}^* = \underset{\vec{x} \in D_{\vec{x}} \subseteq D_f}{\arg\min} f(\vec{x}) \quad \text{or} \quad \vec{x}^* = \underset{\vec{x} \in D_{\vec{x}} \subseteq D_f}{\arg\max} f(\vec{x})$$

where: $D_{\vec{x}}$ – the feasible set

Wrocław University of Science and Technology

# Optimization problems

## Continuous search space

- The domain of the optimized function:

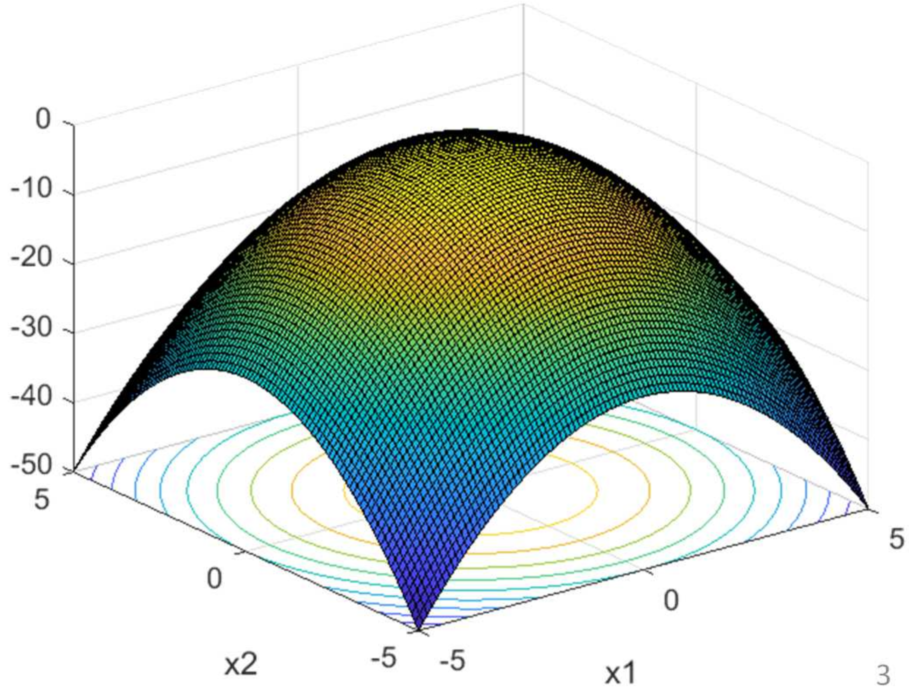
$$D_f \subseteq R^n$$

- Example: two-dimensional sphere function

$\vec{x} = [x_1, x_2]$
$f: D_f = R^2 \rightarrow R$
$f(\vec{x}) = -x_1^2 - x_2^2$

We limit the search space to

$D_{\vec{x}} = [-5, 5]^2 \subseteq D_f$



# Problem nature reminder

- Travelling Salesman Problem (TSP) – ***combinatorial*** *in nature*
  - We want to exchange solution fragments
  - For instacne – the „good" city sequences

- Hill – ***topological*** *in nature* **<- this we are going to solve today**
  - We want to search for the better solution in the neighbourhood of the best solutions found that far
  - We shift „slightly left", or „ slightly right"

Wrocław University of Science and Technology

# Continuous search space

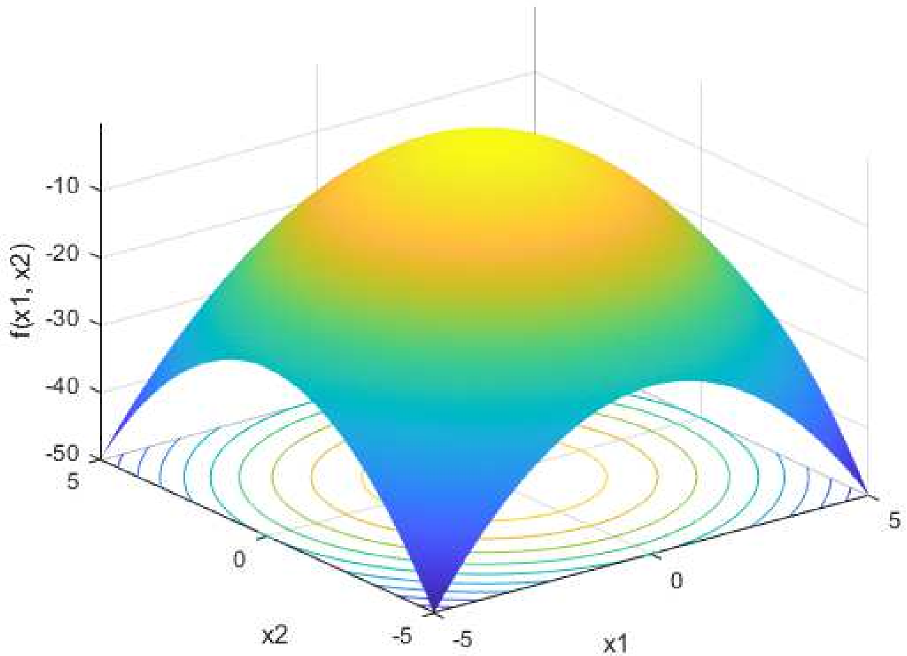
## Unimodal and multi-modal problems

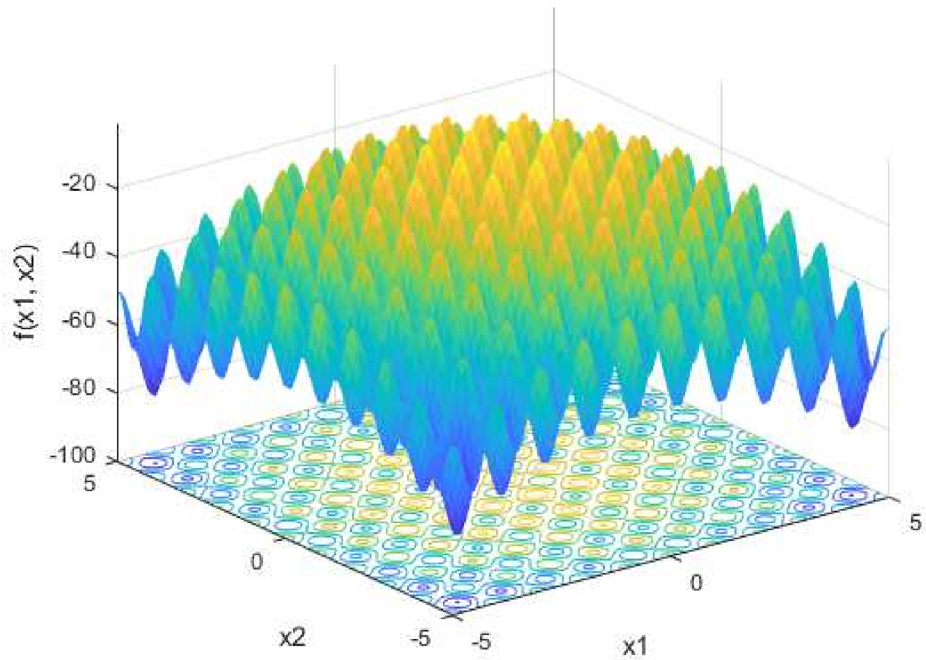
**Unimodal**

- Signle local optimum (global)
- Example: sphere function

**multi-modal**

- Many local optima
- Example: Rastrigin function



# Continuous search space

## Real-world problems

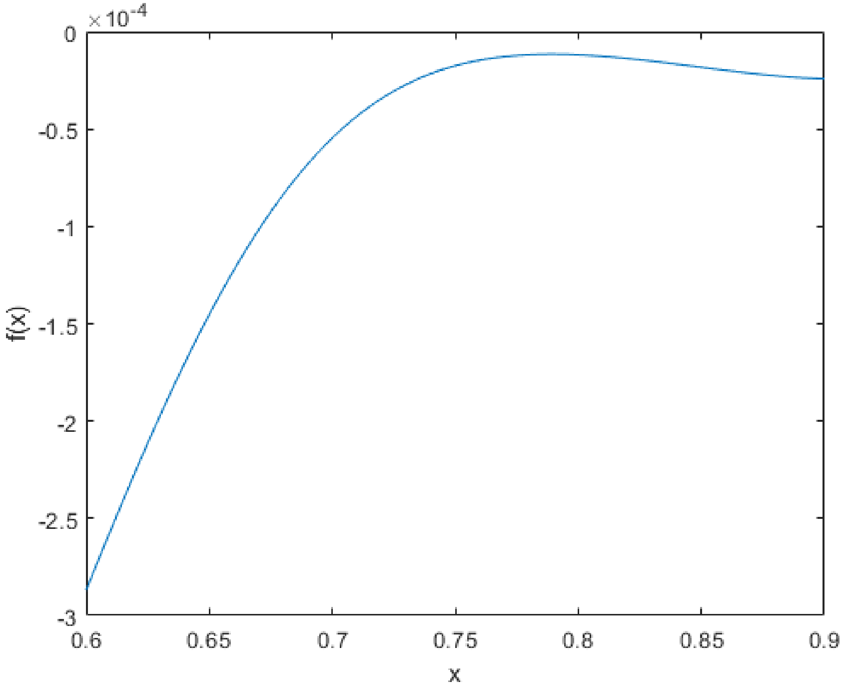
- Chemistry
  - Chemical processes parameters optimization
  - Minimization of the molecule energy
- Power engineering
  - Load division
  - Scheduling in hydrotermal power plants
- Astronautics
  - Spaceship trajectory optimization

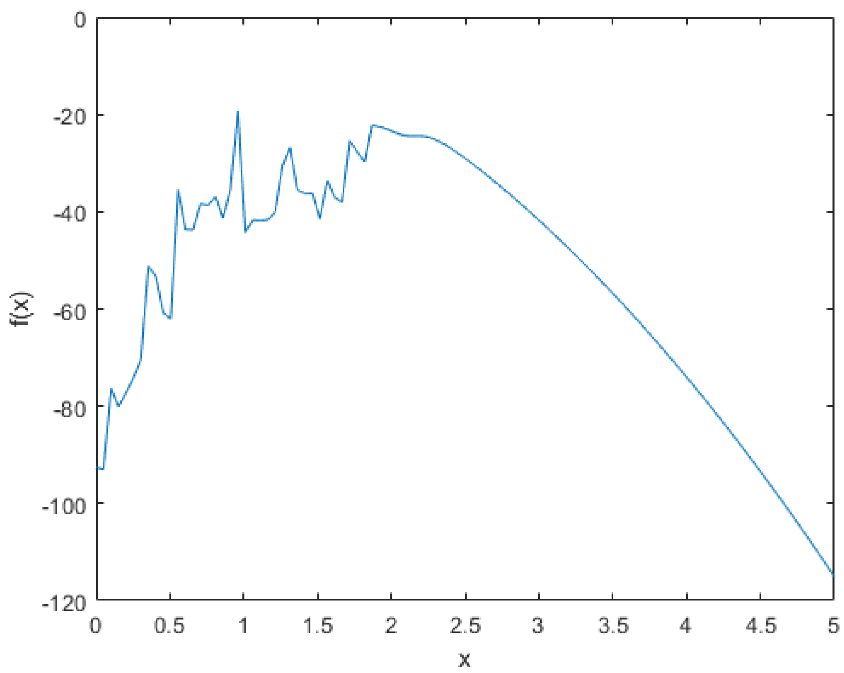
Swagatam Das i Ponnuthurai N. Suganthan. 2010. Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems. *Jadavpur University, Nanyang Technological University, Kolkata*.

# Continuous search space

## Real-world problems

- One-dimensional problems




- Real-world problems have **different** characteristics
- Conclusion: the perfect optimizer – it may **not exist**

# How to find the optimum?

- Let's consider the following function

$$f(x) = -x^2, \qquad D_f = R$$

- Set $D_x = [-5, 5]$
- Using the basic mathematics – we can check when the first derivative equals 0

$$f'(x) = -2x$$

$$f'(x) = 0 \leftrightarrow x = 0$$

$$f(-5) = -25, \qquad \boldsymbol{f(0) = 0}, \qquad f(5) = 25$$

# How to find the optimum?

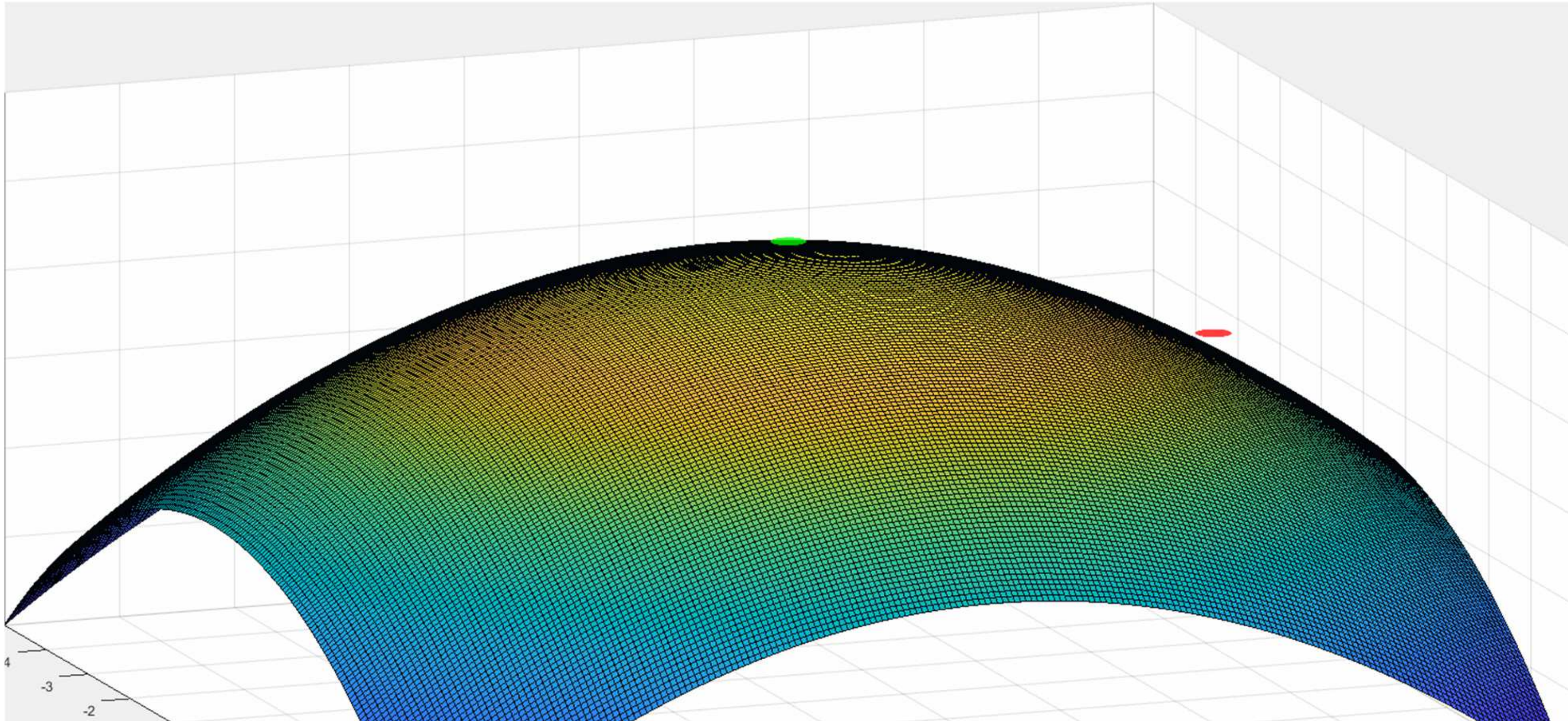
- In case of many variables – check when gradient equals 0
- The set, of frequently non-linear, equations
- The exact method
- Is it easy to implement?
- It is hard to solve sometimes
- If we can implement = then we can automate

# Searching through search space

- Easy to implement, but there is no guarantee we will find the global optimum
- Brute force search
  - Check all available solutions
  - Possible only in very narrow binary/discrete search spaces
  - In continuous search spaces – you can do it only at some certain precision level
- Random Search
  - Check the finite number of available solution, frequently, much smaller than the number of all solutions
  - The subsequent random generations are **not** dependent on the previous ones

Wrocław University
of Science and Technology

# Random Search



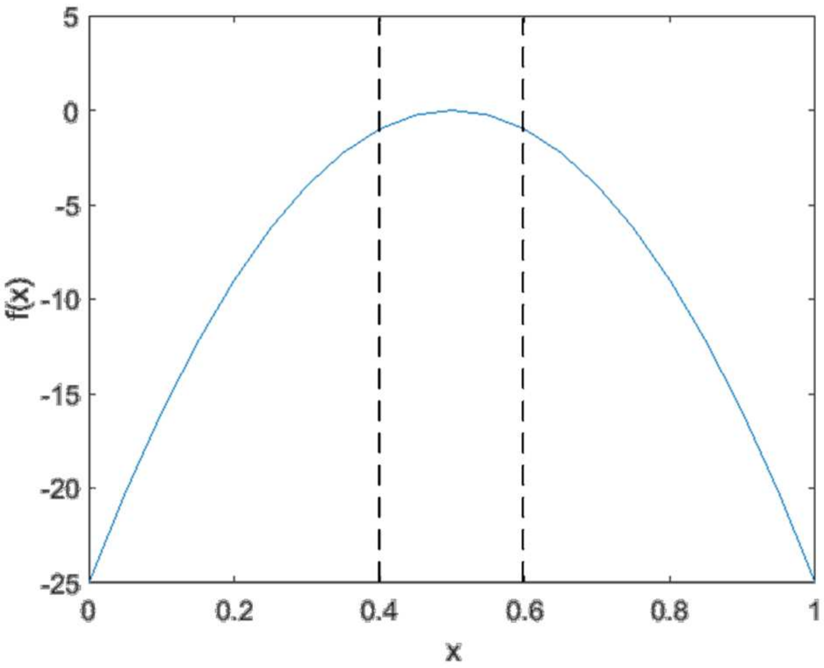

The green point – the optimum

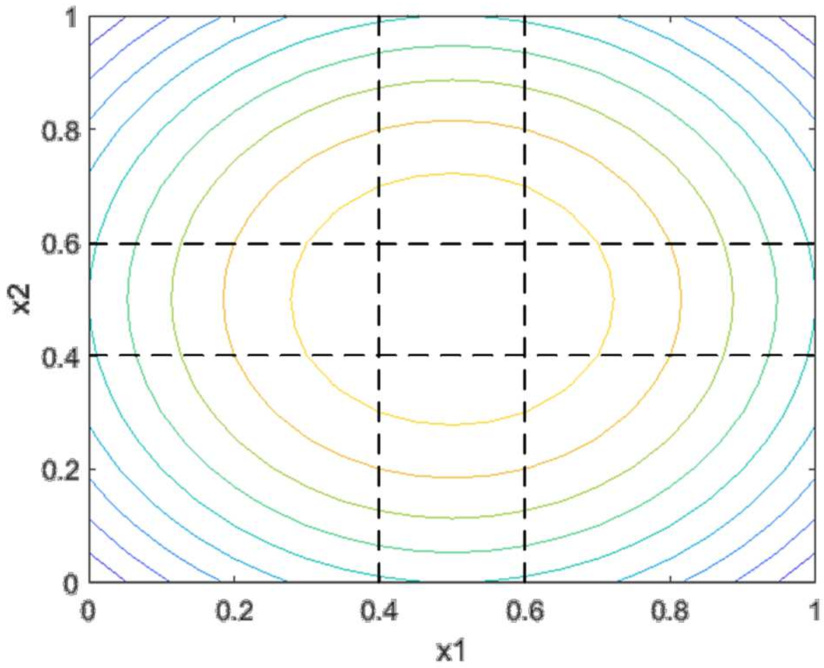
The black point – the best-found solution

The red point – the current solution

# Random Search

Is it easy to find a promising solution?




$$\frac{0.2}{1} = 20\%$$

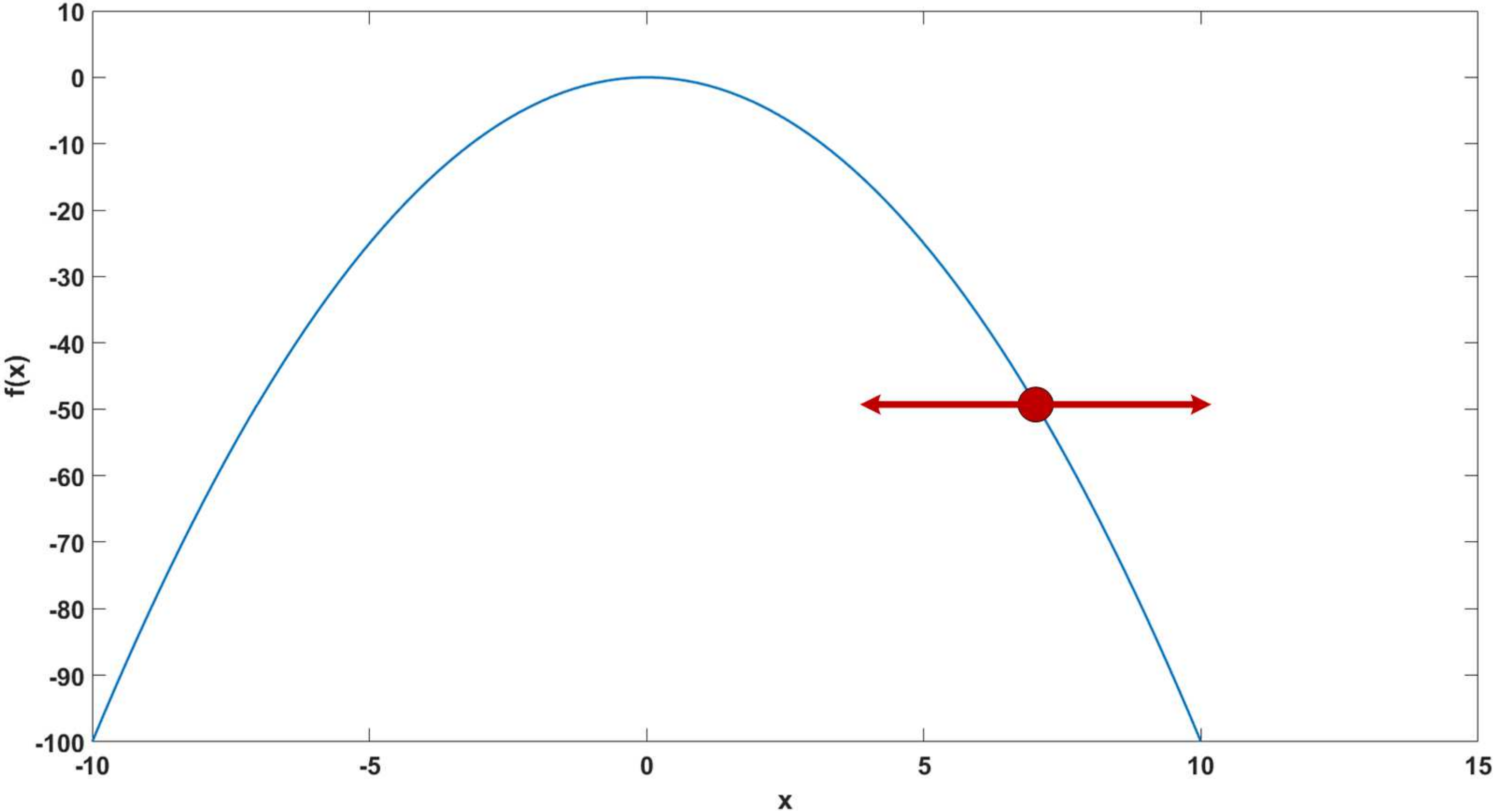
$$\frac{0.2 \cdot 0.2}{1 \cdot 1} = 4\%$$

How about 10 dimensions?

# Directed local saerch

- Input: initial solution
- Output: the **local** optimum
- How to get it: at every iteration we wish to approach the local optimum

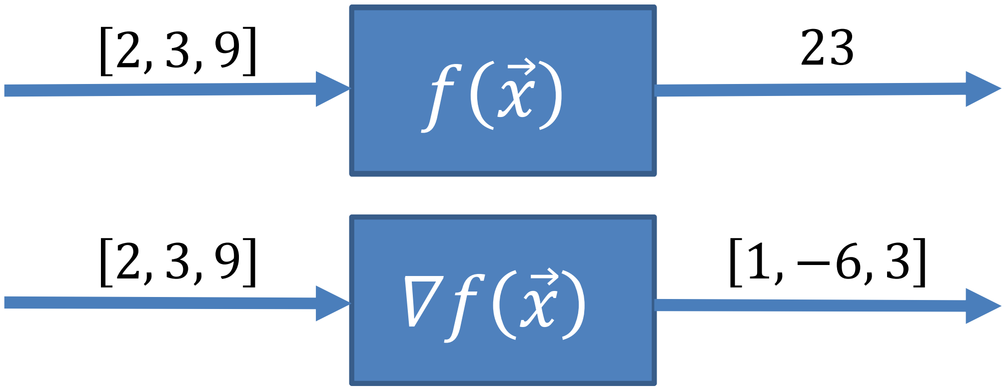
# Left or right?




- We see the chart -> we know where to go
- How optimizer should gues it?

# Gradient methods
## Gradient descent

- We consider the optimized function and its gradient as black-boxes




- Gradient shows us where to go

# Gradient methods
## Gradient descent

- Single step – generate new solution in the direction towards the nearest local optimum

$$\vec{x}_{i+1} = \vec{x}_i + \alpha \cdot \nabla f(\vec{x}_i)$$

where $\alpha$ is the step-length coefficient

- The value of $\alpha$ is usually low
- In the case of a single variable

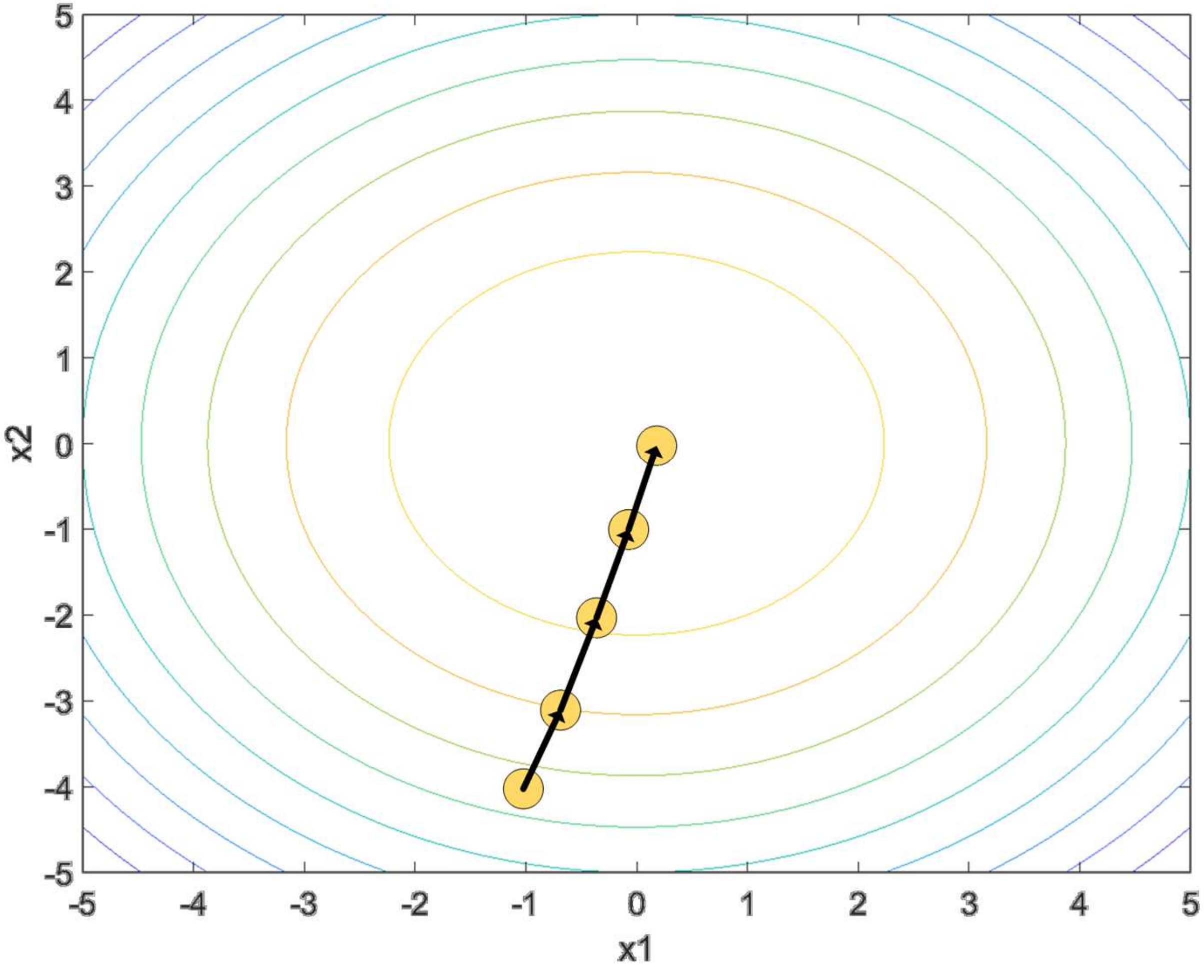
$$x_{i+1} = x_i + \alpha \cdot f'(x_i)$$

Wrocław University of Science and Technology

# Gradient descent

## Sphere funciton example

$$f(\vec{x}) = -x_1^2 - x_2^2, \qquad \nabla f(\vec{x}) = [-2x_1, -2x_2]$$



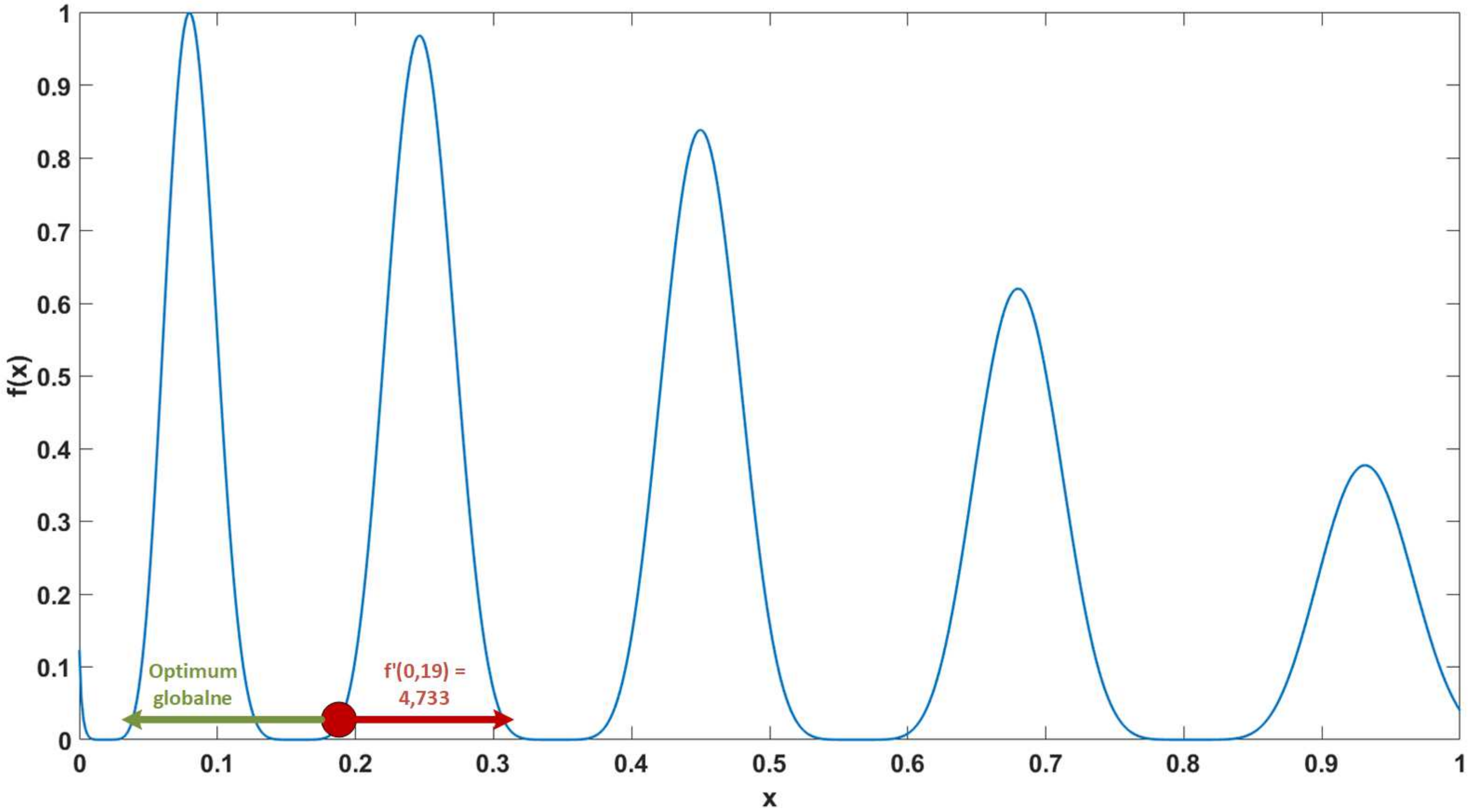
# Gradient descent
## Multi-modal problems

Uneven Decreasing Maxima problem

$$f(x) = \exp\left(-2\log(2)\left(\frac{x - 0{,}08}{0{,}854}\right)^2\right)\sin^6\left(5\pi\left(x^{3/4} - 0{,}05\right)\right)$$

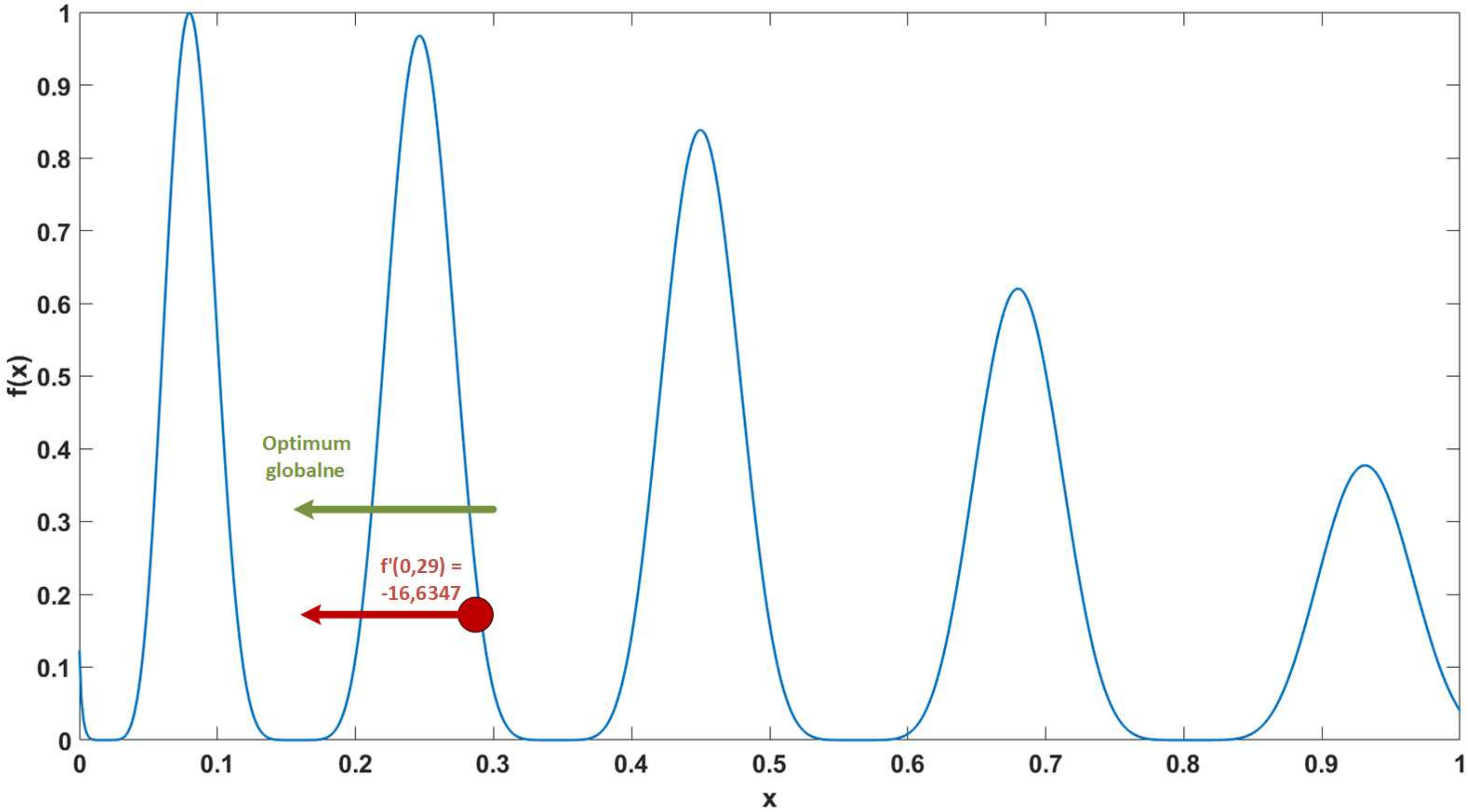
$$f'(x) =$$
$$= \frac{1}{\sqrt[4]{x}} e^{-1{,}90081(x-0{,}08)^2} \sin^5\left(5\pi\left(x^{3/4} - 0.05\right)\right)\left(70{,}6858\sin\left(2{,}35619\right.\right.$$
$$\left.\left.- 15{,}708x^{3/4}\right) + \sqrt[4]{x}(0{,}30414 - 3{,}80163x)\sin\left(5\pi\left(x^{3/4} - 0{,}05\right)\right)\right)$$

# Gradient descent
## Multi-modal problems




So close to the optimum… but do not celebrate too early…

# Gradient descent
## Multi-modal problems

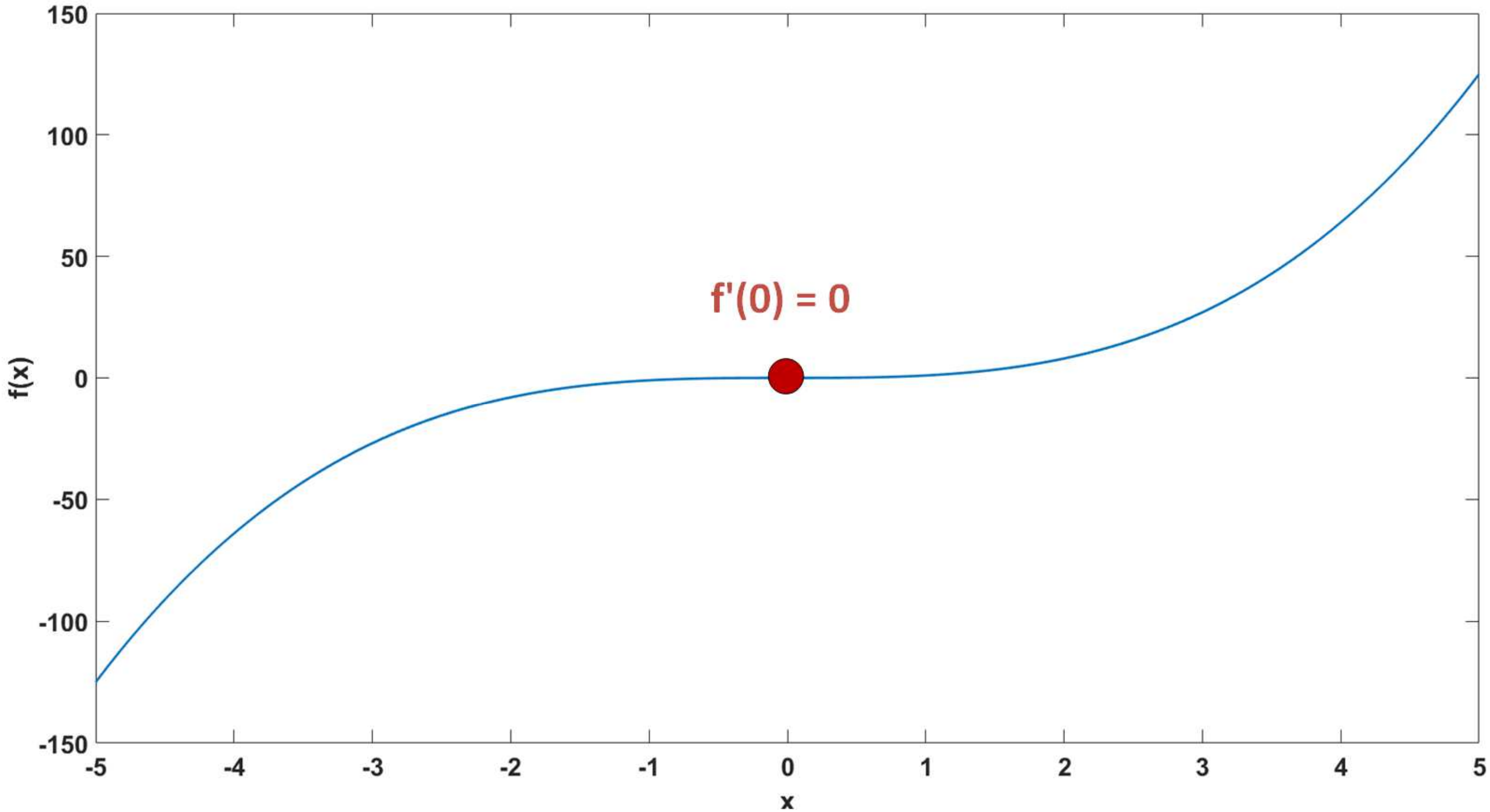



We move in the right direction… So maybe we can make a large step?

But will we get to the top? **Not really…**

# Gradient descent

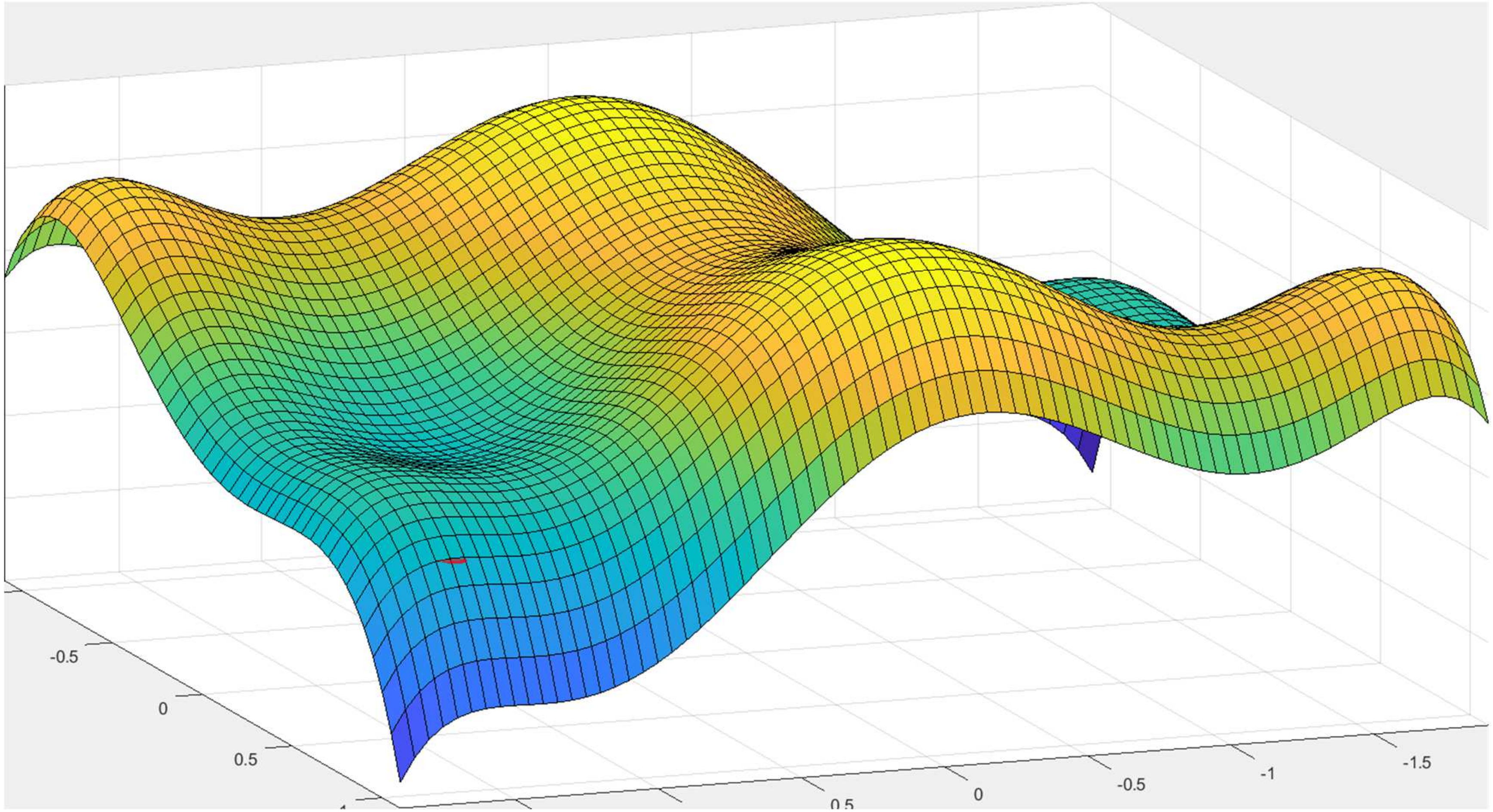
## Getting stuck




Unfortunatelly, we got stuck – the first derivative equals 0…

# Gradient descent

## Small step

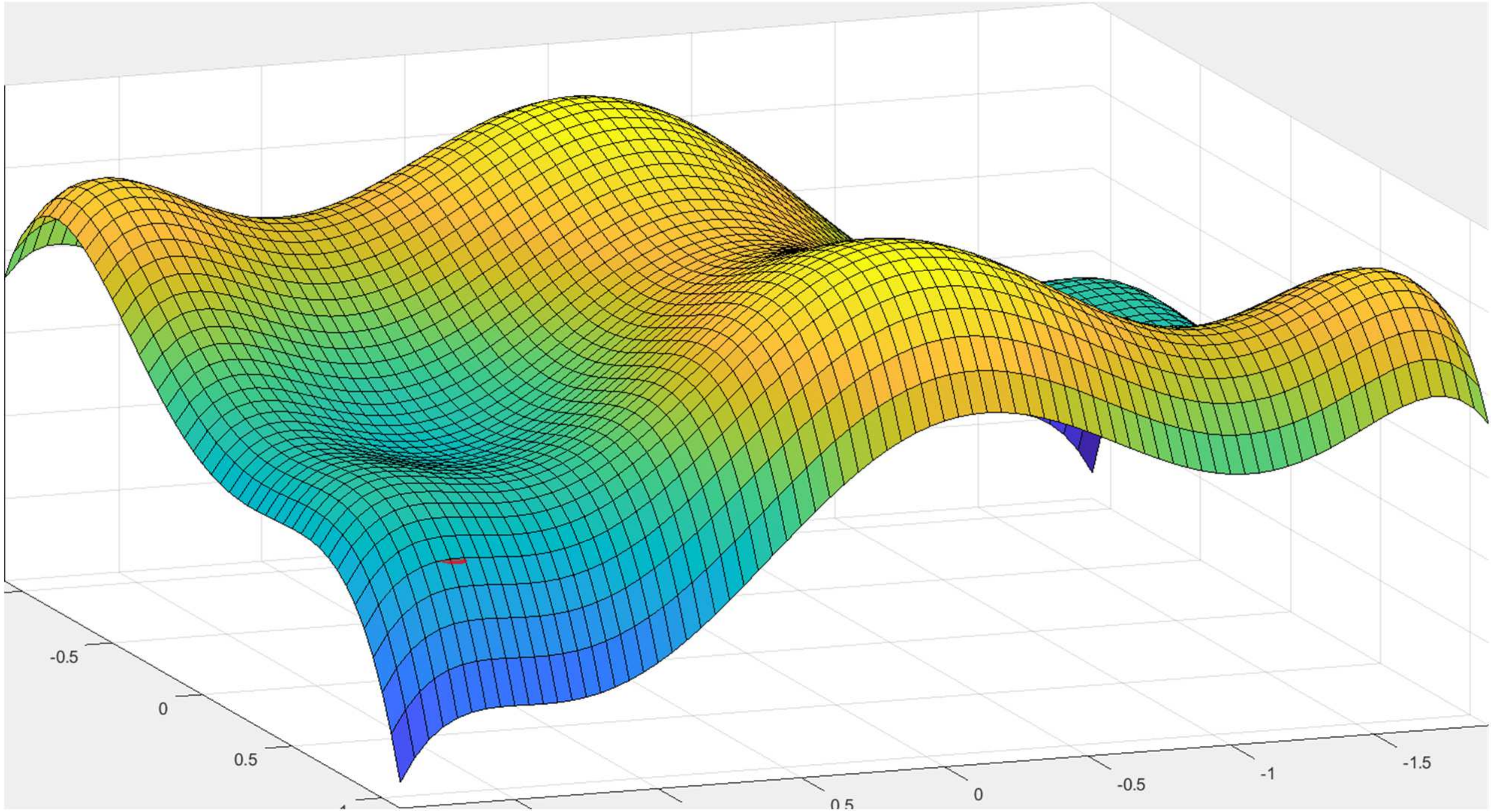



We quickly find the local optimum
And run around it

Wrocław University of Science and Technology

# Gradient descent

## Large step




We jump around the whole space We have found the subspace with global optimum

**But we do not know how to climb up…**

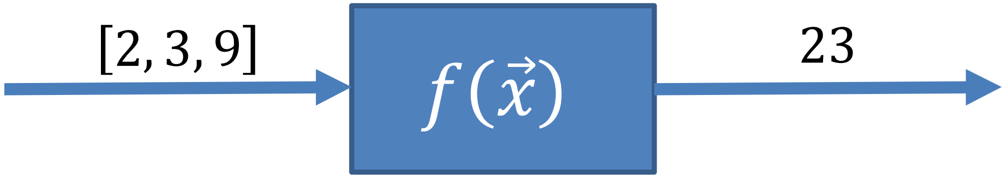
Wrocław University of Science and Technology

# Gradient methods
## Summary

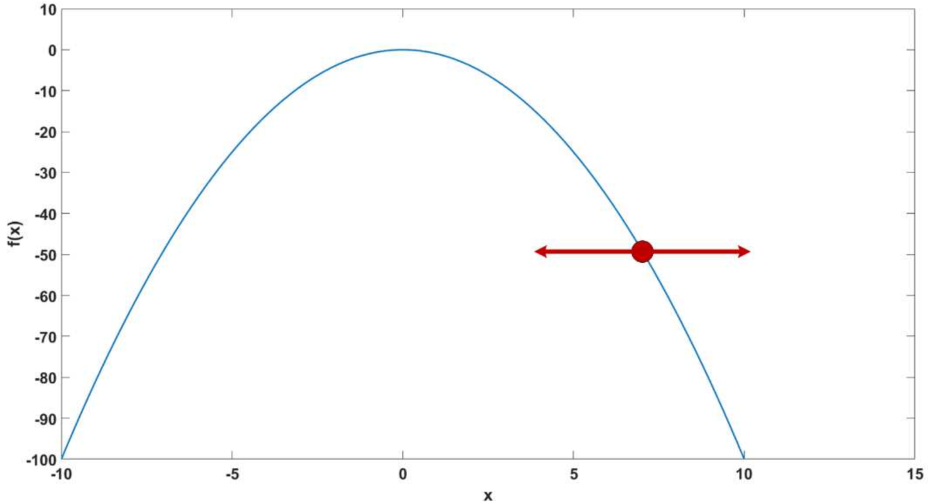
- We must know at least the first derivative – **significant limitation**
- Small optimization step – **we can easily get stuck**
- Higly effective for **convex problems**
- Examples
  - Gradient descent
  - Newton method (it can handle many issues but requires the second dericative)

# Non-gradient methods

- We only have a black-box with the optimized function




- On this base we must find the appropriate direction



# Non-gradient methods

## Hill-Climbing

- Search the neighbourhood to find better solutions
- Neighbourhood is frequently defined using the normal distribution

$$\vec{x} = [x_1, \dots, x_n] \text{ - initial solution}$$

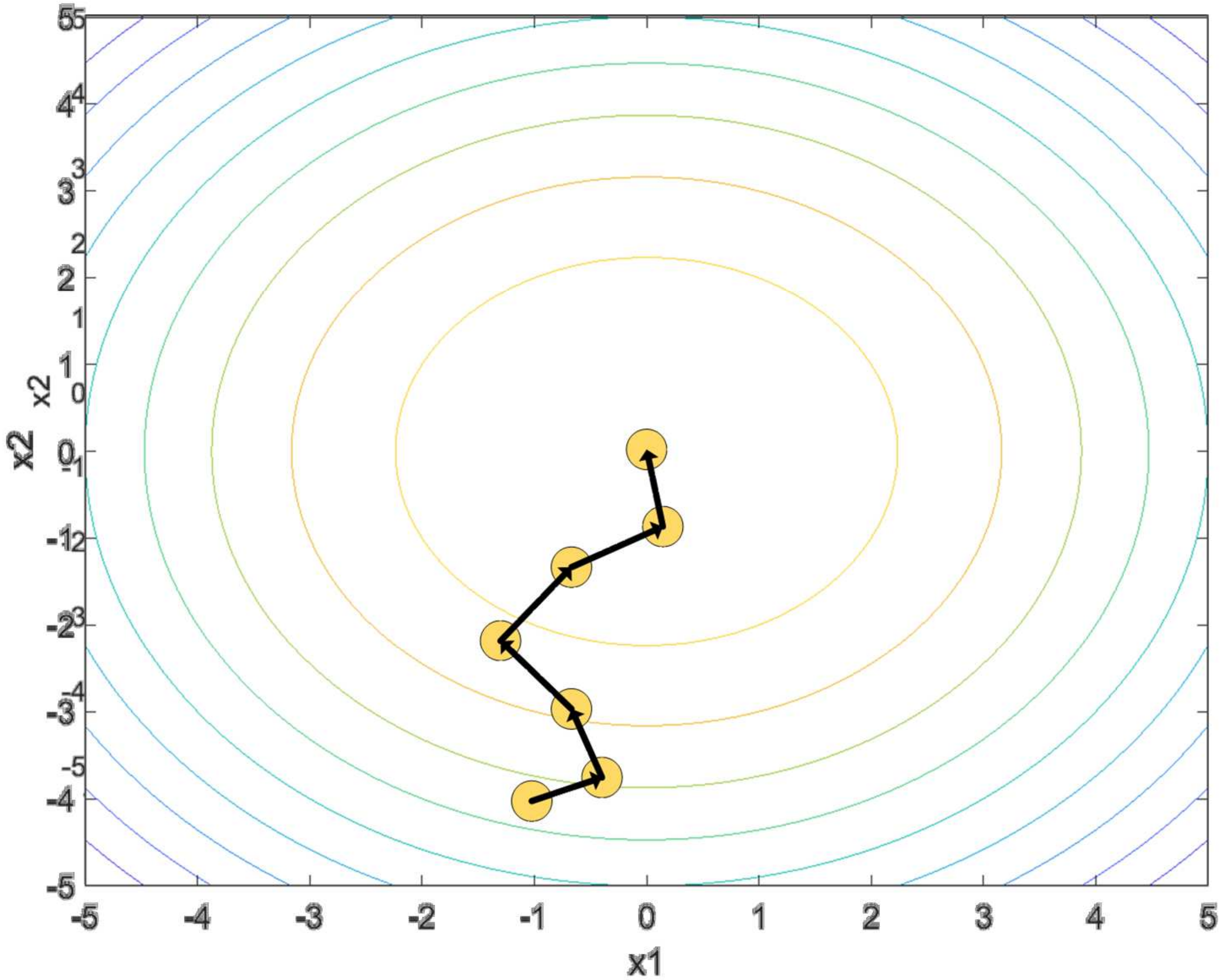
$$\overrightarrow{x'} = [x_1', \dots, x_n'] \text{ - neighbouring solution}$$

$$x_i' = x_i + \sigma \cdot N(0,1)$$

where $\boldsymbol{\sigma}$ defines the neighbourhood size

- The best solution found in the current iteration becomes the staring point for the next generations

# Hill-Climbing

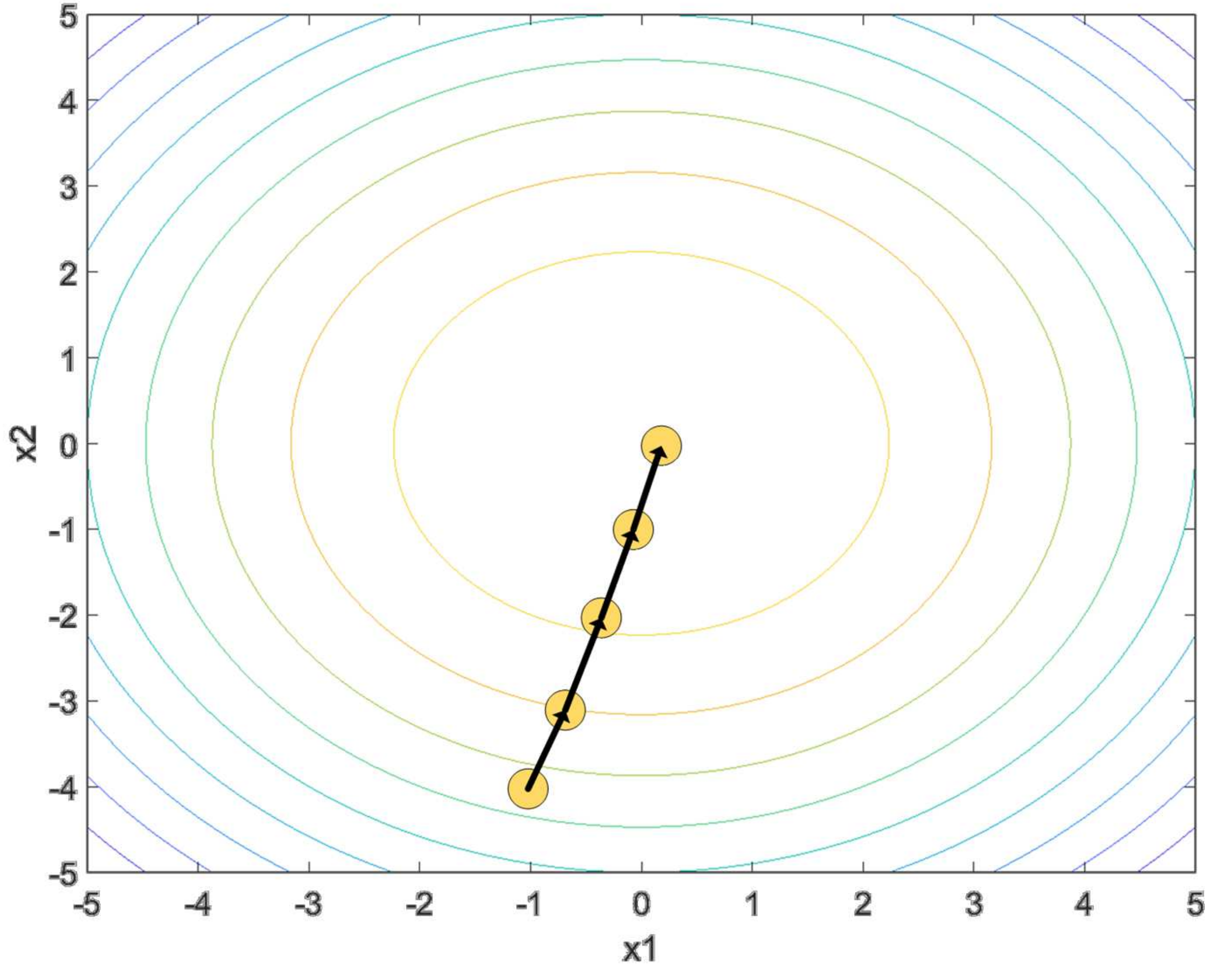


Wrocław University of Science and Technology

# Gradient descent

## Reminder

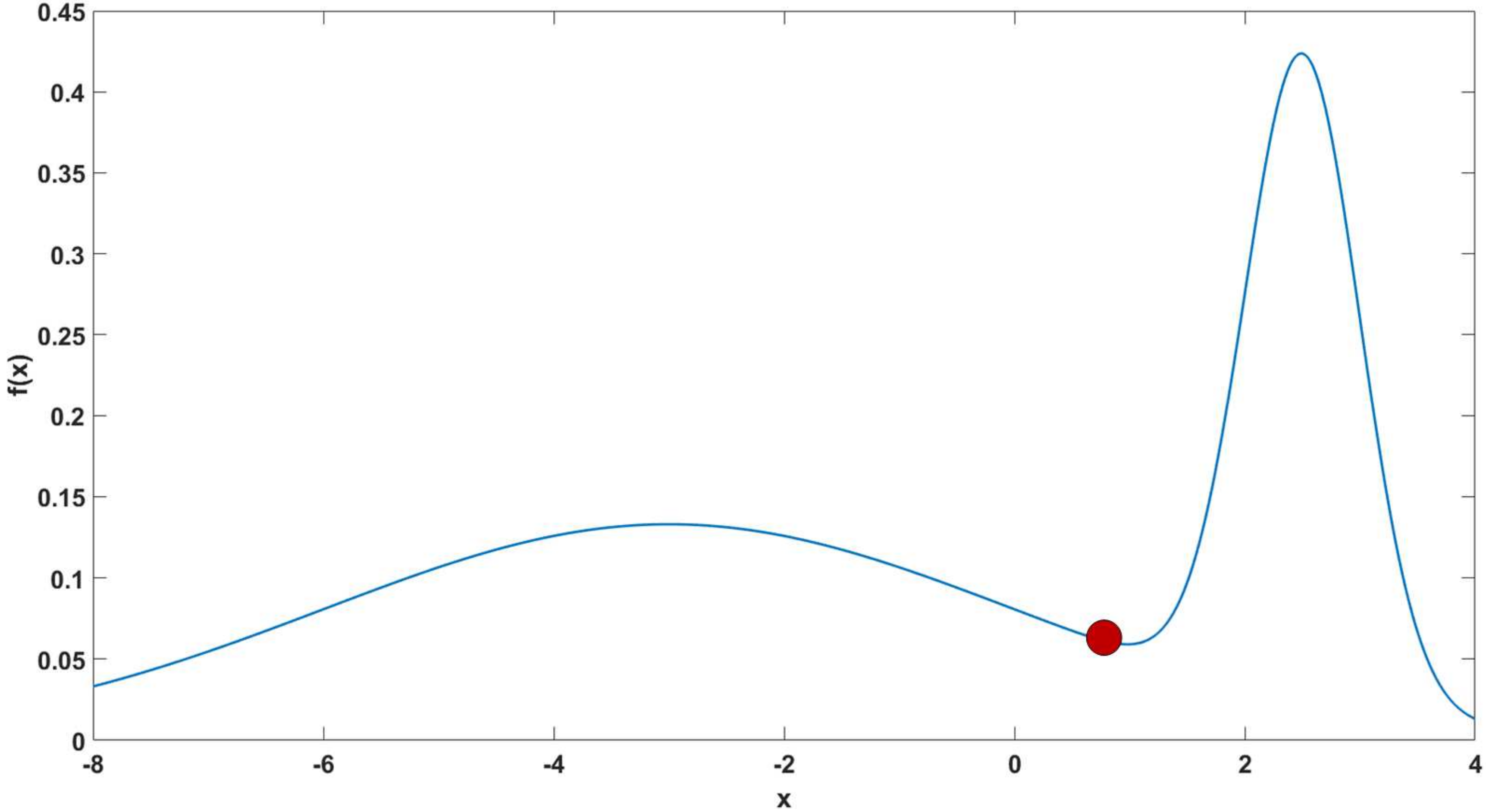
$$f(\vec{x}) = -x_1^2 - x_2^2, \qquad \nabla f(\vec{x}) = [-2x_1, -2x_2]$$



# Hill-Climbing

- Gradient descent – right to the optimum
- Hill-climber – non-convincing „zigzag”
- **BUT:**
  - **„Zigzag” got where it should**
  - **I do not have to know any derivative**
  - **The derivative does not evan have to extist!!!**

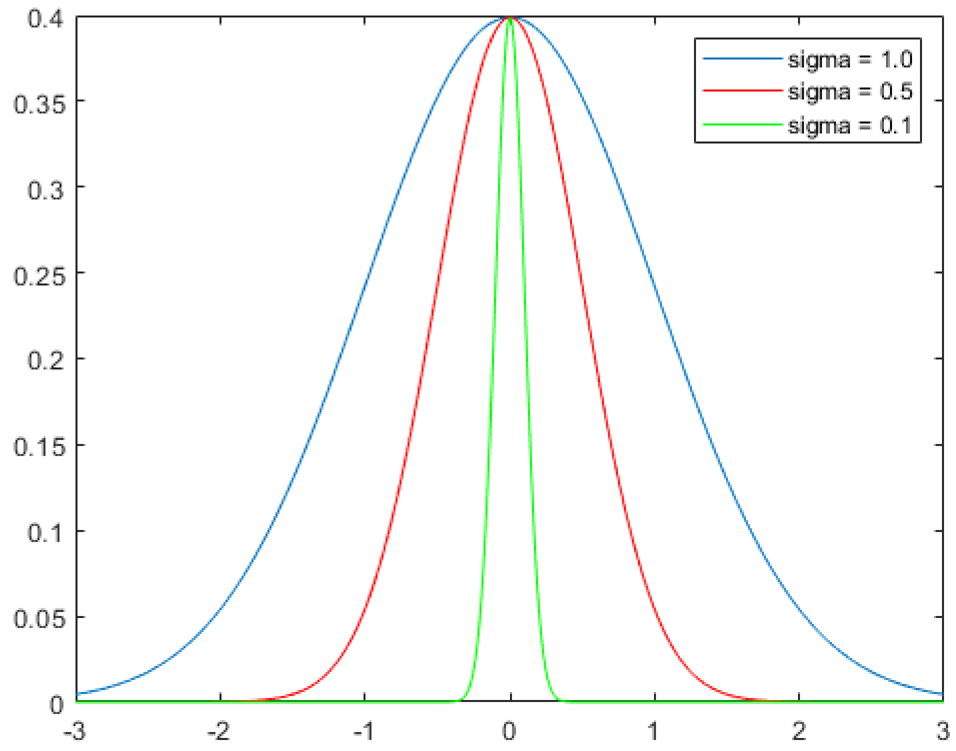
# Attraction basins




- Gradient-using optimizer – where will it go?
- How about the Hill-climber?

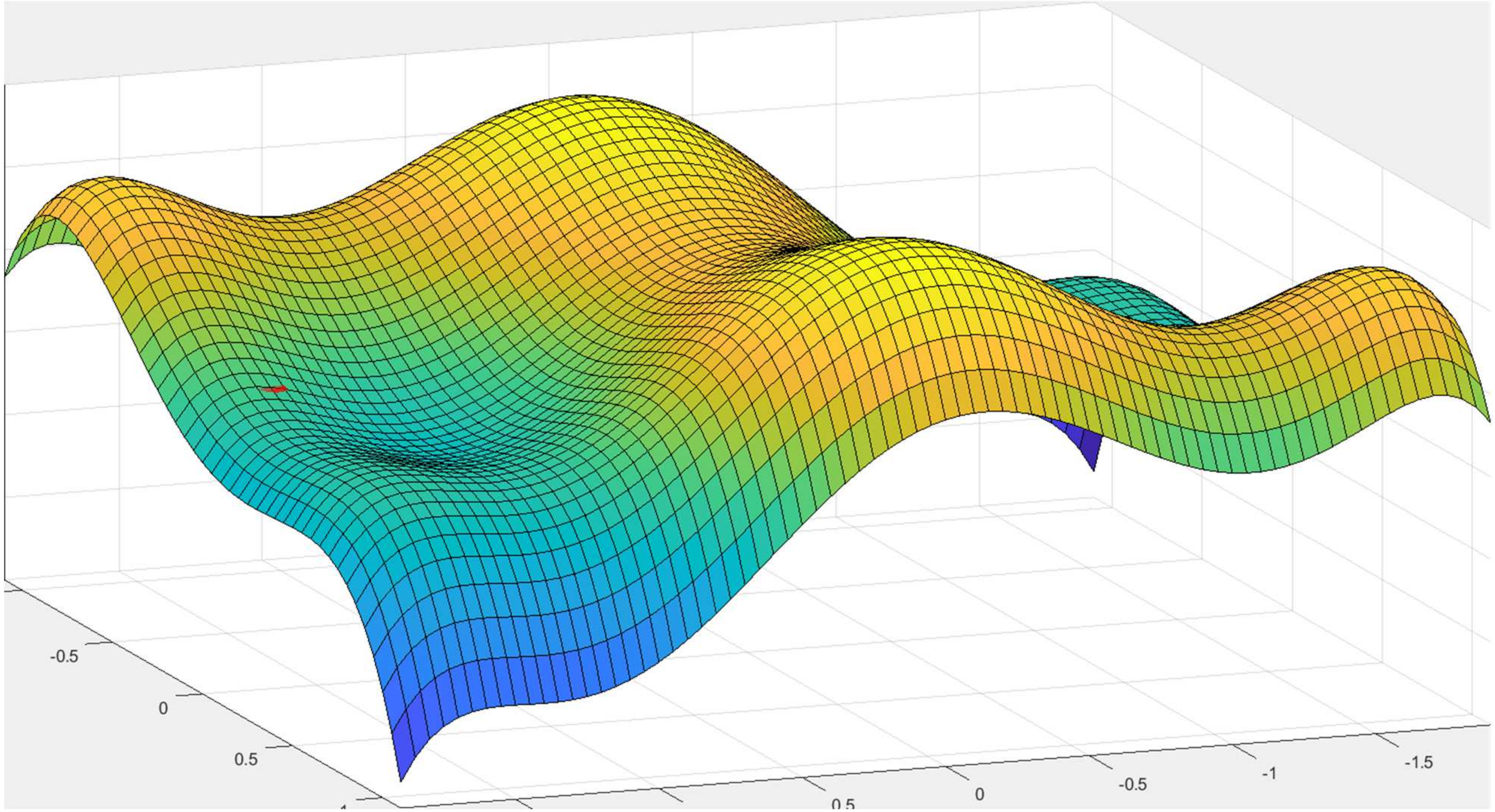
# Hill-Climbing
## Neighbourhood

- Small $\sigma$ – only the neighbourhood search
- High $\sigma$ – search in the close and far neighbourhood
- **How this is different to the step in gradient methods?**



# Hill-Climbing

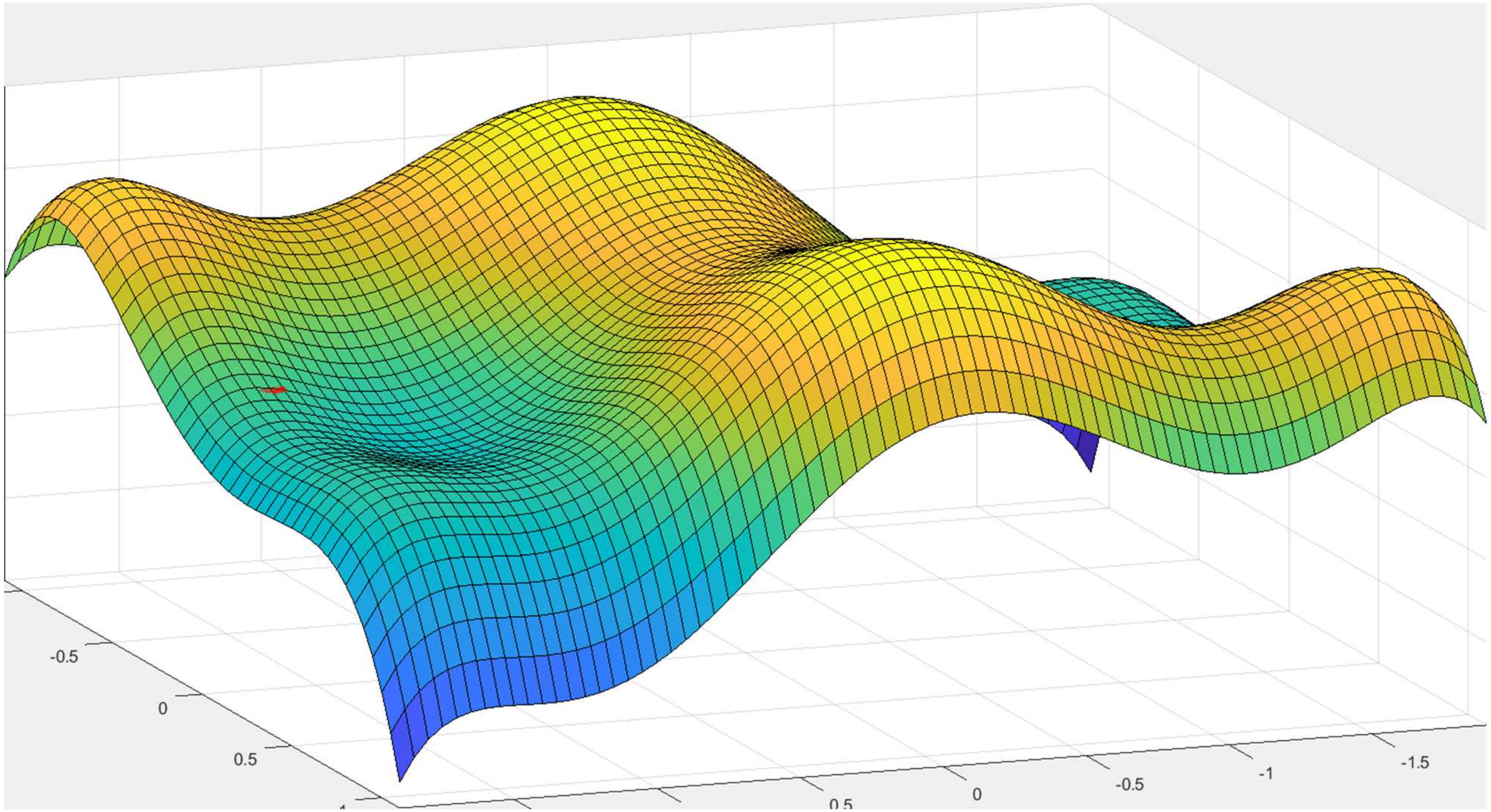
## Small step



Wrocław University of Science and Technology

# Hill-Climbing

## Large step



# Exploration and exploitation

- Large steps – **search for** attraction basins with high-quality solutions (exploration)
- Small steps – **climb up** in a given attraction basin (exploitation)
- Local optimization – small steps are favorable
- Global optimization
  - Small/large step better? We do not know
  - Different step size on a different optimization stage (np. large at the beginning, small at the end)
  - Step size should be **adapted during** the optimization

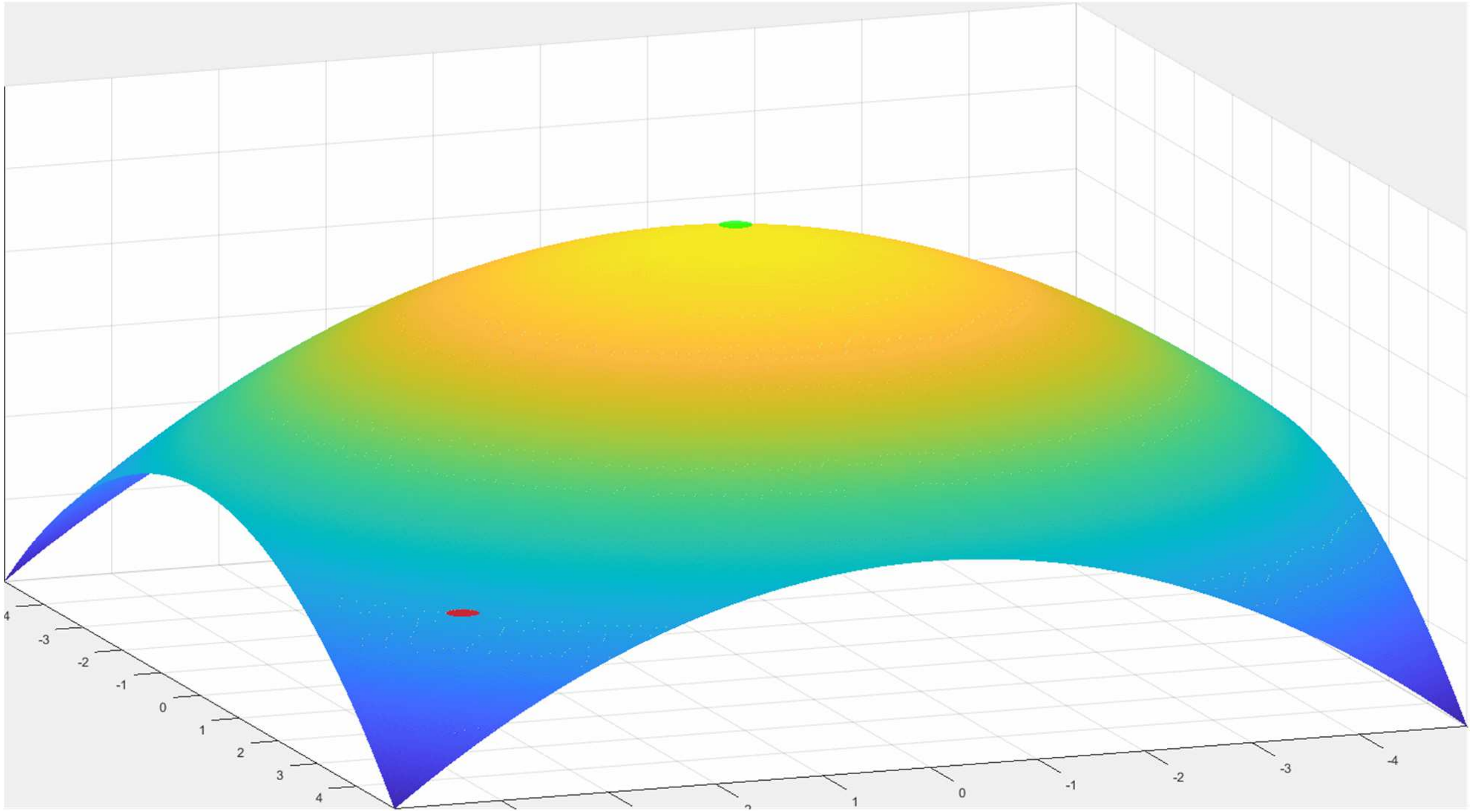
# Evolutionary strategies

# Evolutionary strategy (1 + 1)

- Subclass of Hill-Climbers
- **Single** optimizer iteration – check **one** solution from the neighbourhood
- How to generate solution from the neighbour? – use **mutation**!

# Evolutionary strategy (1 + 1)

## Small step

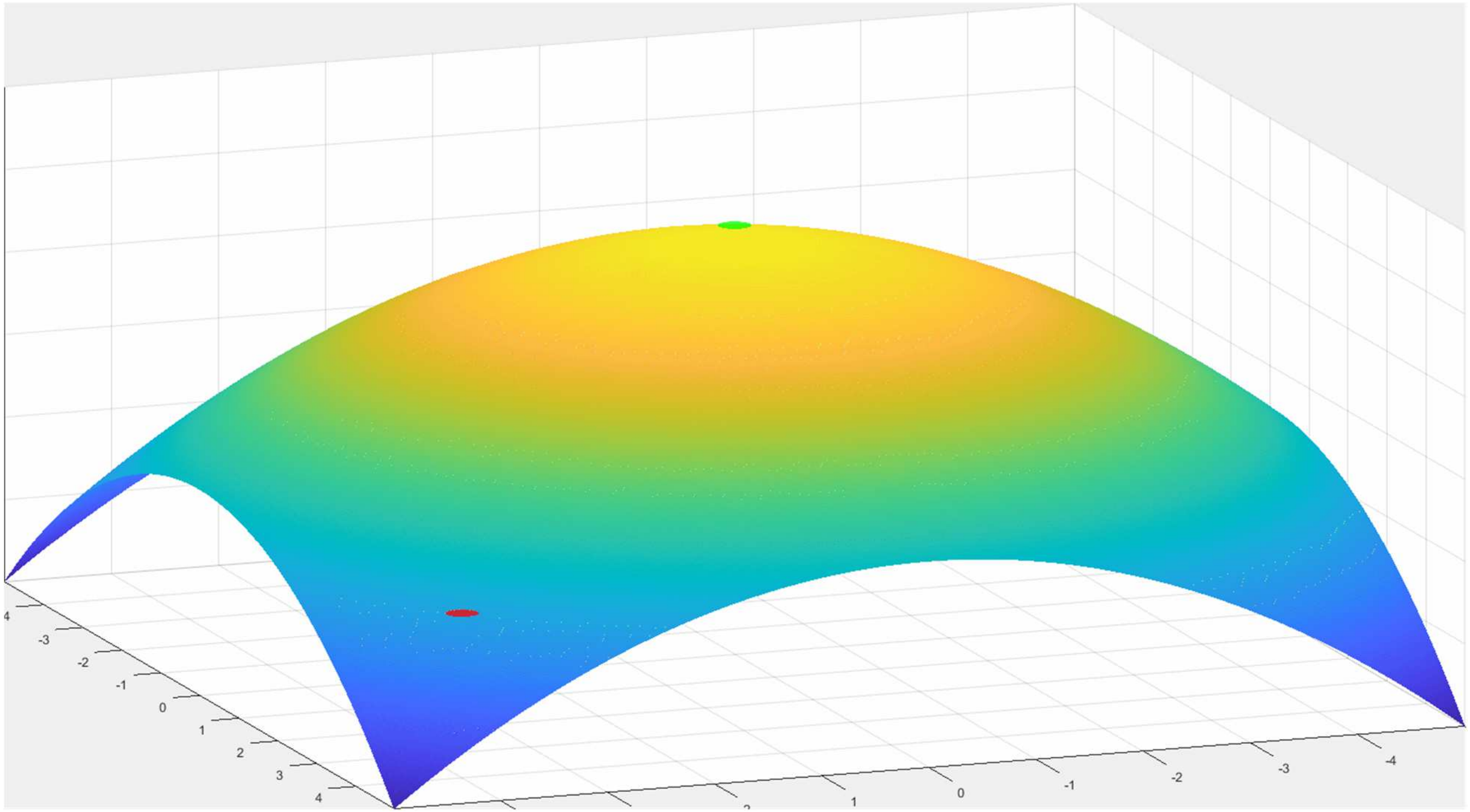


# Evolutionary strategy (1 + 1)

- Small step
  - Unimodal functions – we **WILL FIND** the global optimum…
  - …but **slowly**
- Proposition – let’s investigate the large step

# Evolutionary strategy (1 + 1)

## Large step

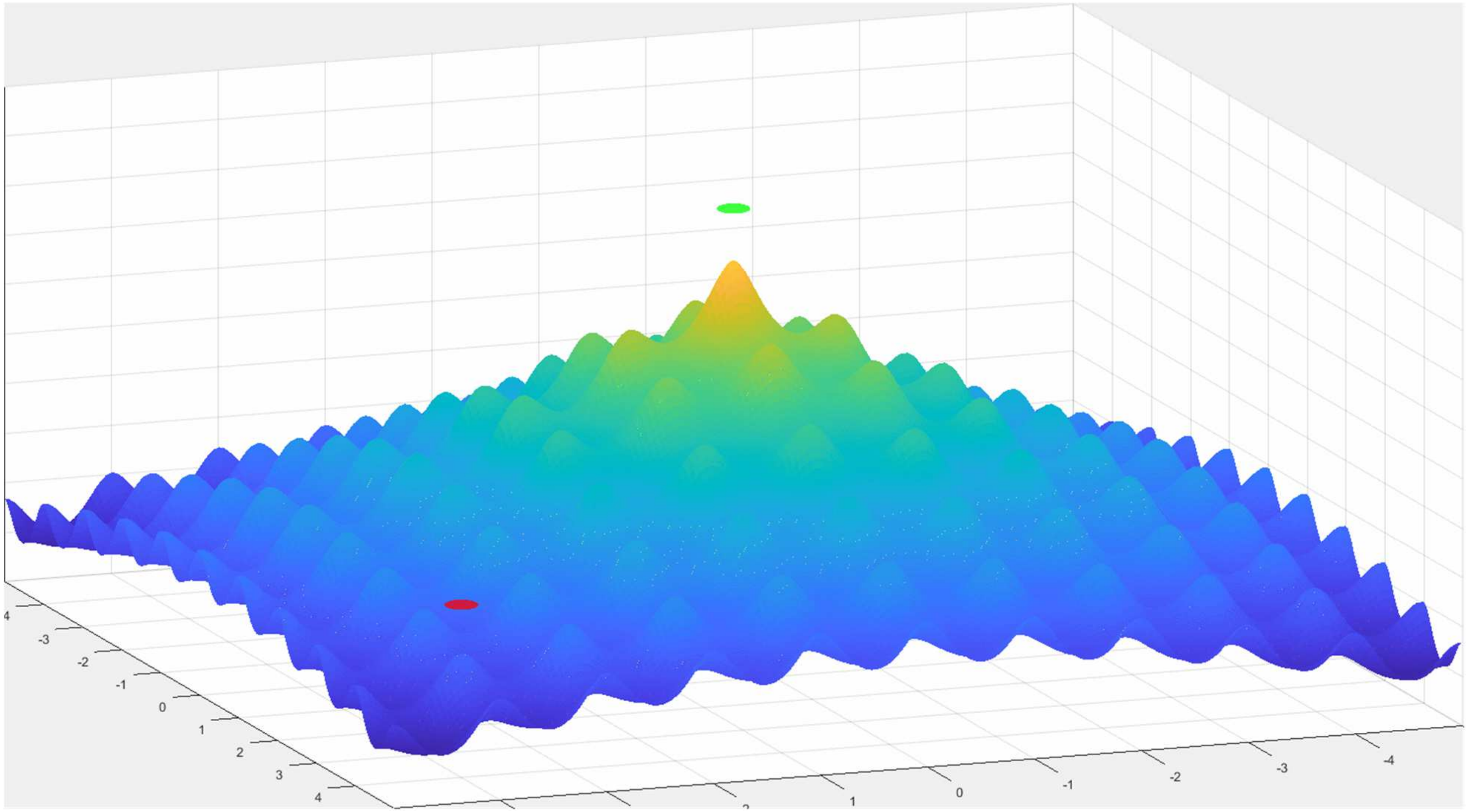


# Strategia ewolucyjna (1 + 1)

- Large step
  - Unimodal functions – we **quickly find** the **attraction basin** of the global optimum (exploration)
  - **No exploitation –** the chance to **get close** to the optimum is **small**
- Maybe the small step is not **THAT** bad?
  - We move **slowly**…
  - …but **surely**…
- Let’s check it for the multi-modal problems!

# Strategia ewolucyjna (1 + 1)

## Small step and multi-modal function



# Strategia ewolucyjna (1 + 1)

- Small step and multi-modal problems
  - Randomly chosen local optimum – quickly found
  - **No exploration**
  - Similar to greedy algorithm
  - For **multi-modal** problems – **ineffective**
- **Some idea** for small step and multi-modal problems
  - Frequent restarts
  - Expected effectiveness increase - small

# Strategia ewolucyjna (1 + 1)

## Step size – summary

- Small step
  - Precisely **exploitates**
  - Effective in finding local optima
  - Multi-modal problems – ineffective (**no exploration**)
- Large step
  - Effective **exploration**
  - **No exploitation**
- Small/large step ratio?
  - **What is the perfect ratio?**
  - **Dependens on the problem!**
- **Idea – adaptation!**

# Step size adaptation
## 1/5 success rate rule

- Consider the last $\boldsymbol{k}$ mutations
- Decision based on the number successful iterations
  - Success = mutation that improved the solution quality
  - Success rate **is higher** than $1/5 \cdot k$, **increase** the step size

$$\sigma^{(g+1)} = \sigma^{(g)} \cdot 1/c_d$$

  - Success rate **is lower** than $1/5 \cdot k$, de**crease** the step size

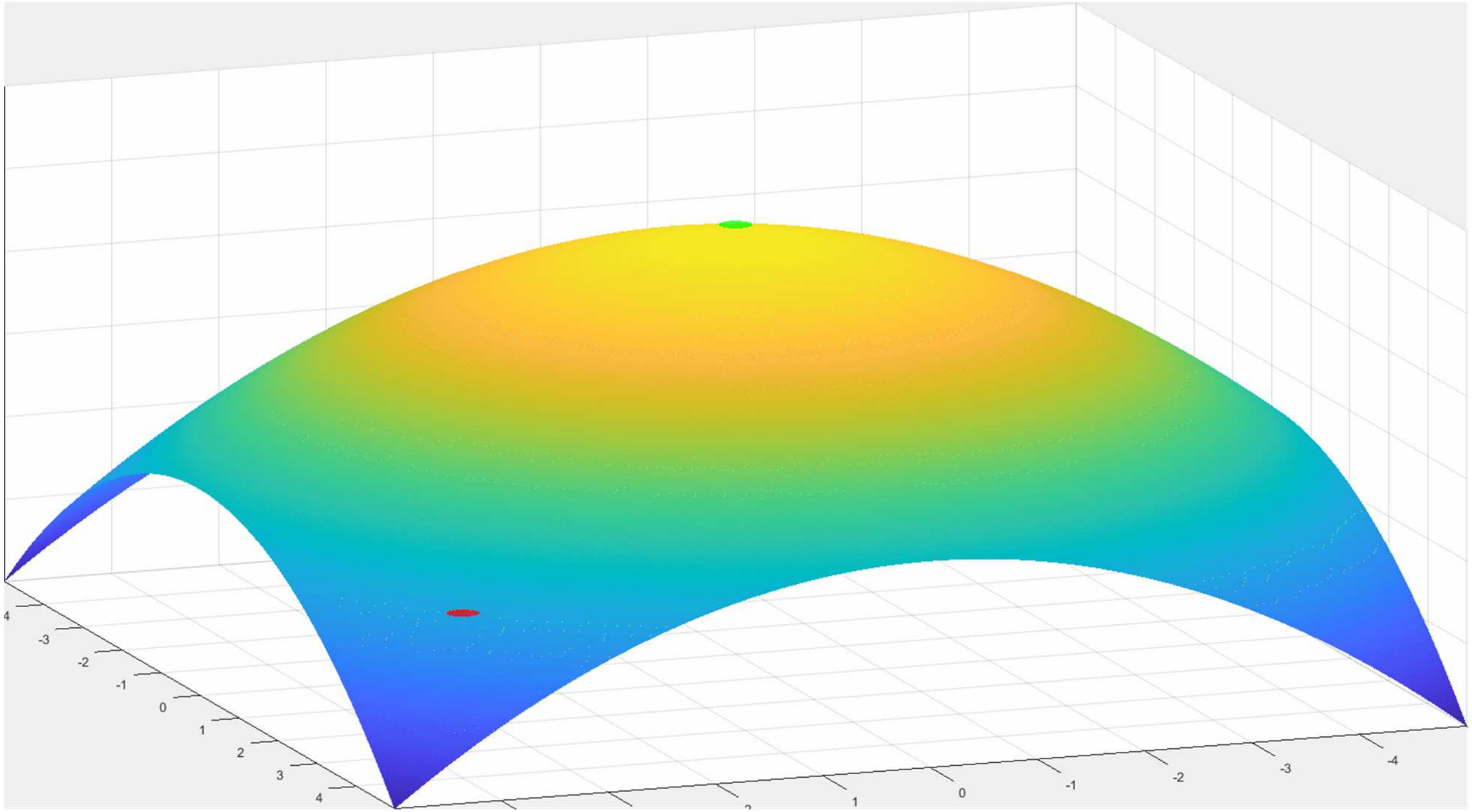
$$\sigma^{(g+1)} = \sigma^{(g)} \cdot c_d$$

  - Success rate **is equal** to $1/5 \cdot k$, **do not modify** the step size
- It frequent to use $\boldsymbol{c_d = 0,82}$
  - The general value range $c_d \in (0, 1)$
  - „Good" $c_d$ value – dependent on the problem

Wrocław University of Science and Technology

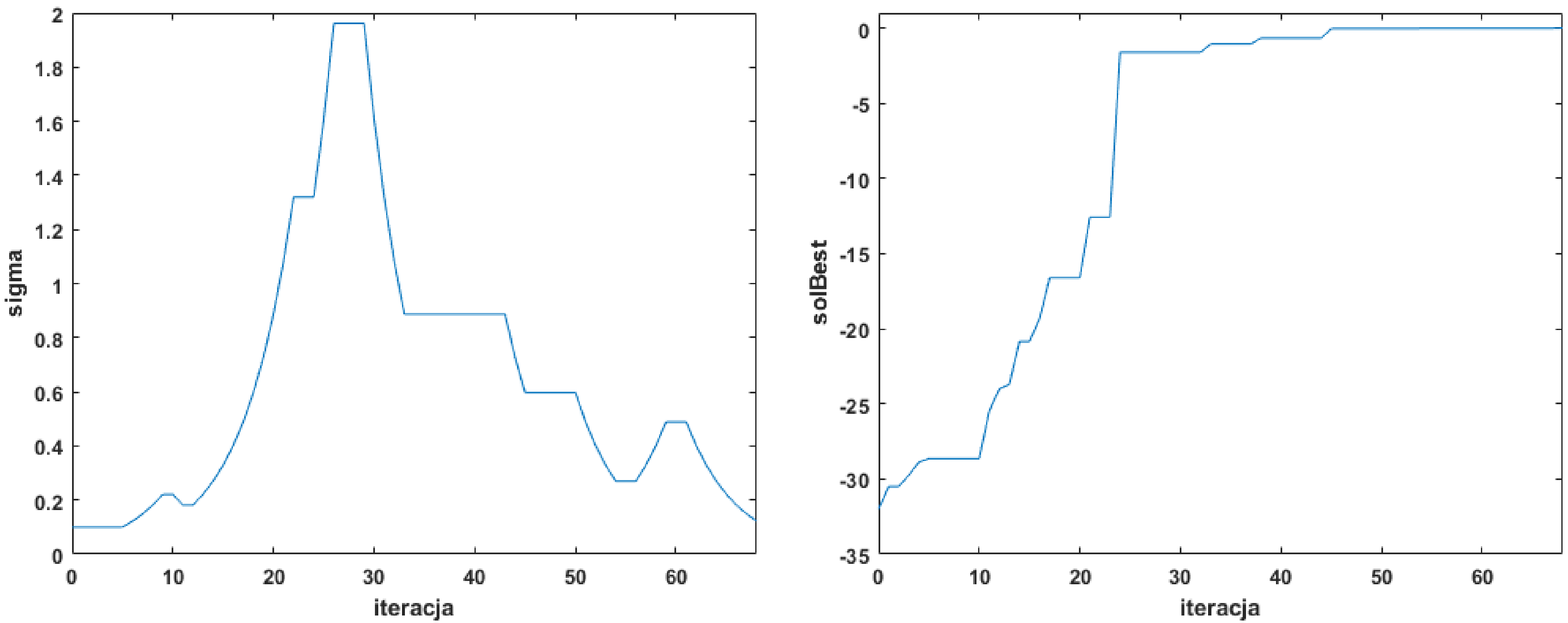
# 1/5 success rate rule

## Small step at the beginning



# 1/5 success rate rule

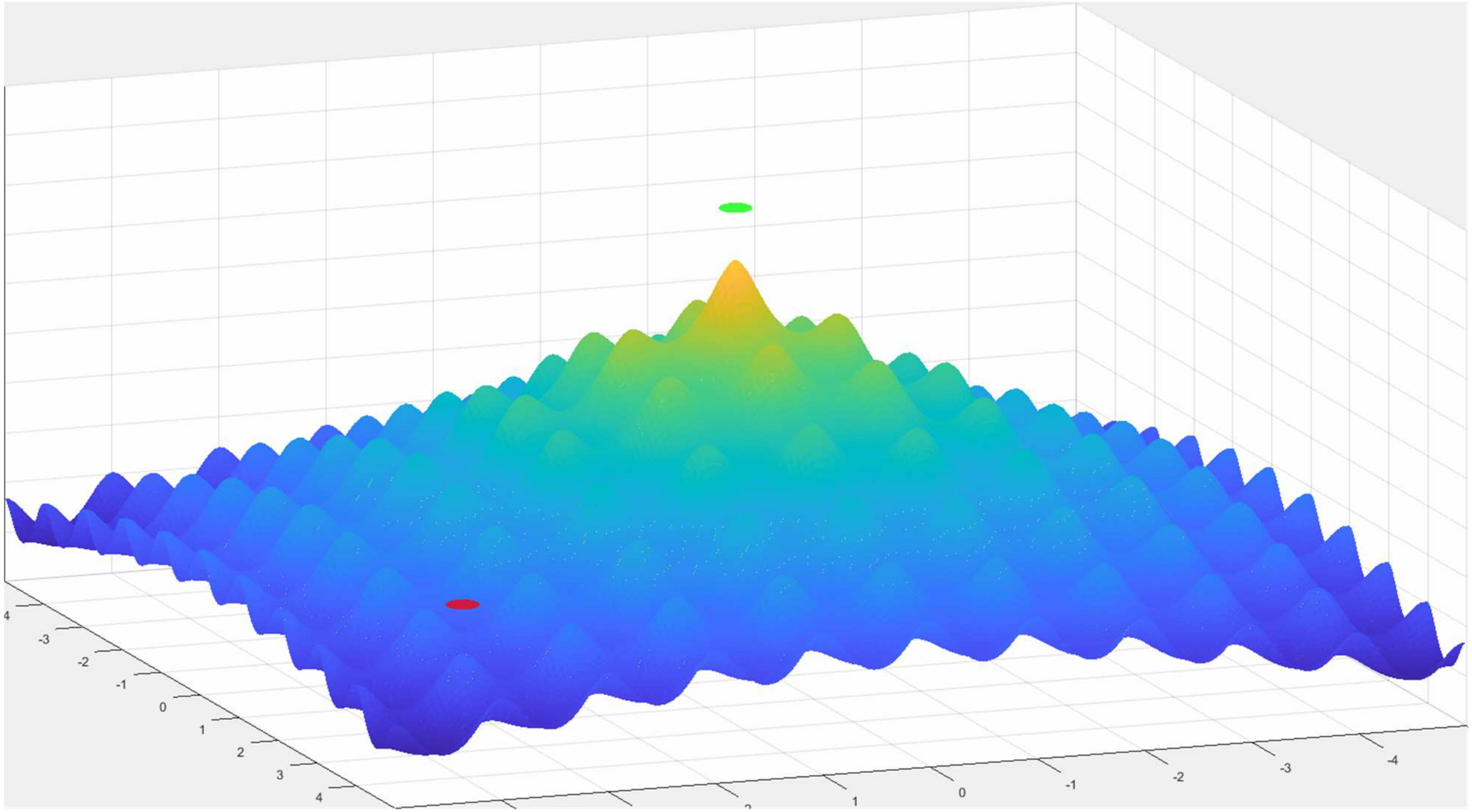
## Small step at the beginning



Wrocław University of Science and Technology

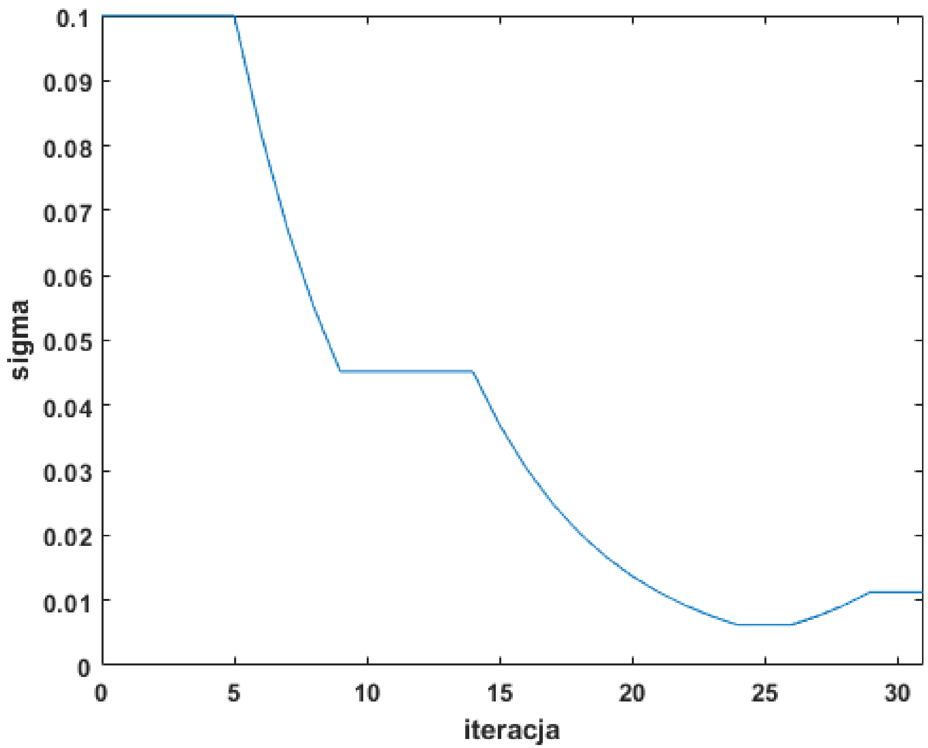
# 1/5 success rate rule

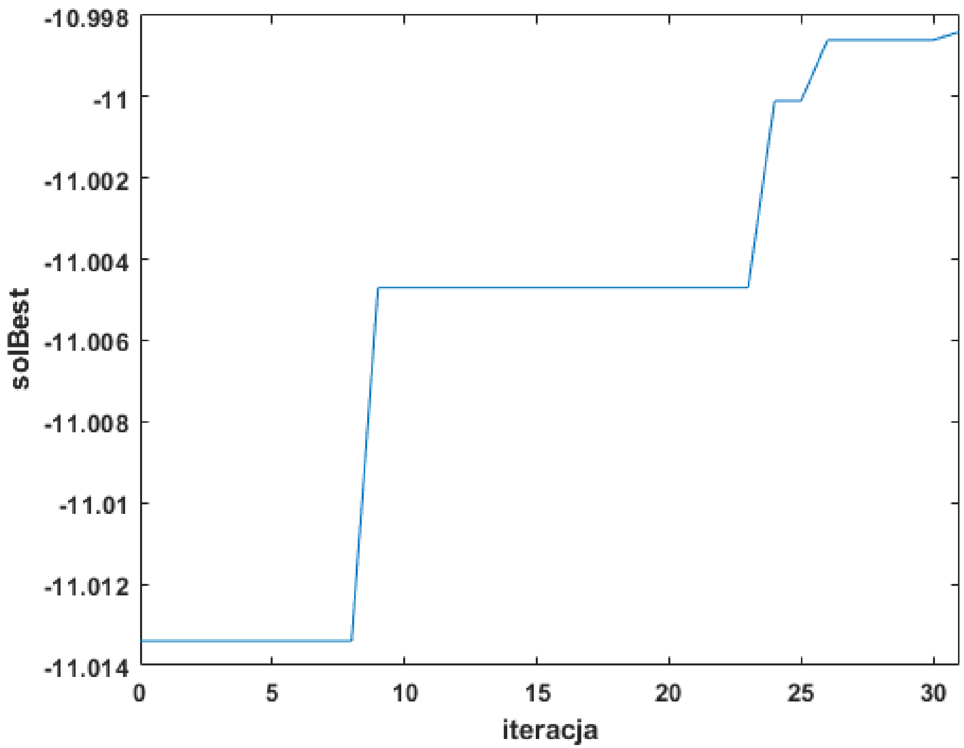
## Small step at the beginning – multi-modal function



# 1/5 success rate rule

## Small step at the beginning



# 1/5 success rate rule

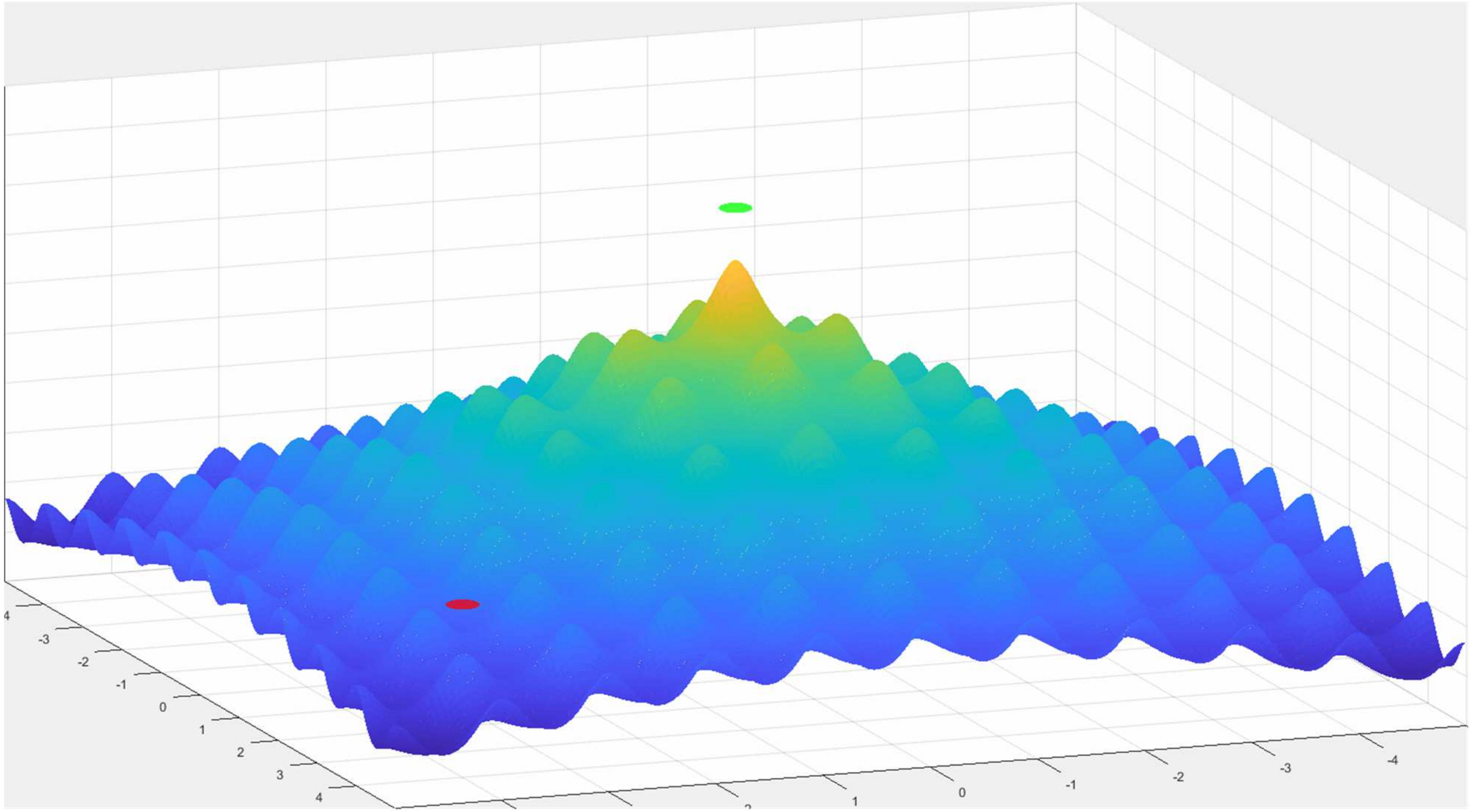
## Small step at the beginning

- Unimodal function
  - At first we climb up slowly…
  - …then the climbing speed increases…
  - …finally, we slow down to find the global optimum
- Multi-momda function
  - Local optimum is precisely exploited (again)
  - After some time the step size will increase but it may take a lot of time
- Conclusion: let’s start from the **large** step (**exploitation**)

Wrocław University of Science and Technology

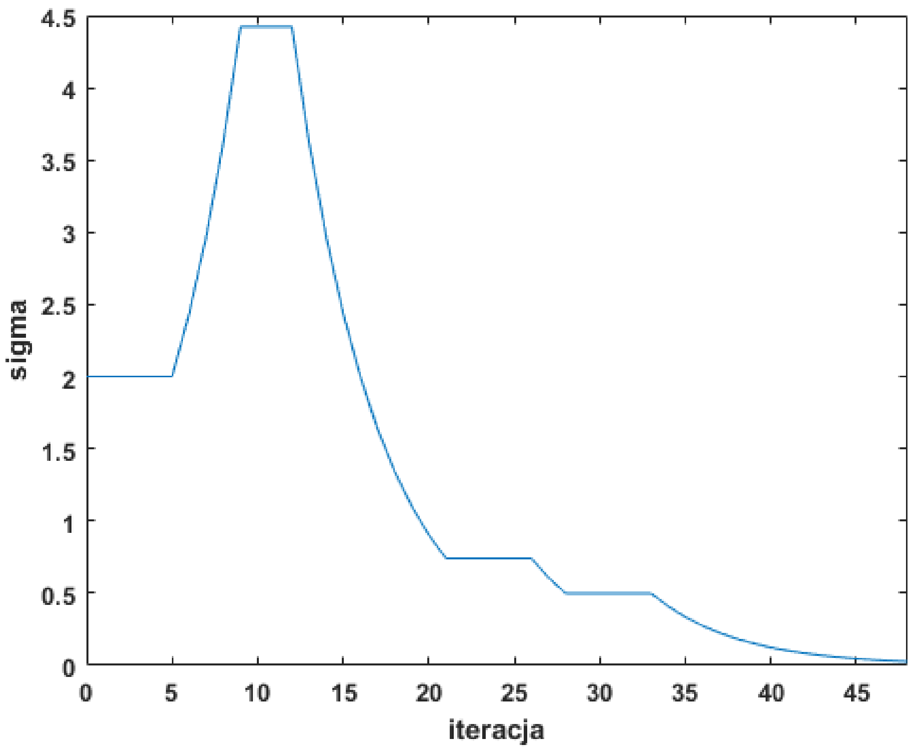
# 1/5 success rate rule

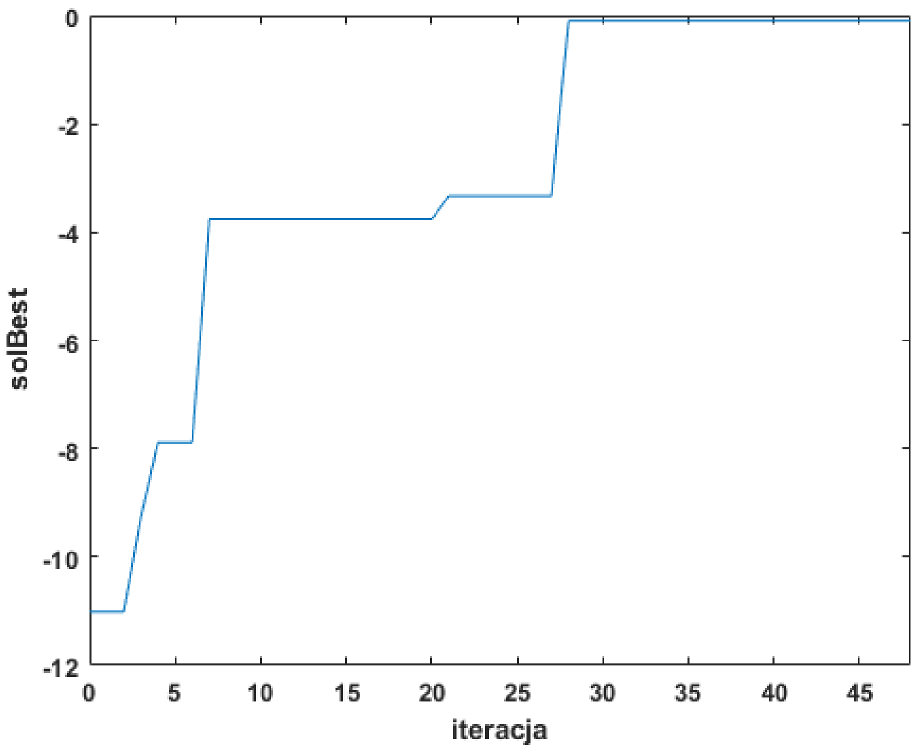
## Large step at the beginning



# 1/5 success rate rule

## Large step at the beginning



# 1/5 success rate rule

## Summary

- Small step size at the beginning
  - We still can can get stuck
  - A lot of time will pass before the optimizer will start exploration
- Reasonable strategy
  - **Large step** at the beginning **(exploration)**
  - Step size will automatically decrease later on **(exploitation)**
- Switching between exploration, and exploitation
  - Possible
  - But expensive (we have to wait long before step size will be large again)

# Estimation of distribution algorithm (EDA)

- Main idea
  - We **model** the solution space
  - We generate new (candidate) solution using the model
- Single iteration:
  - Create the new solution using the **model**
  - Evaluate the new solution
  - Update the **model** on the base of rated solutions (attention: the better fitting solutions may influence the **model** more)
- Examples
  - Bayesian optimization algorithm (BOA) **(discrete problems)**
  - Covariance matrix adaptation evolution strategy (CMA-ES)

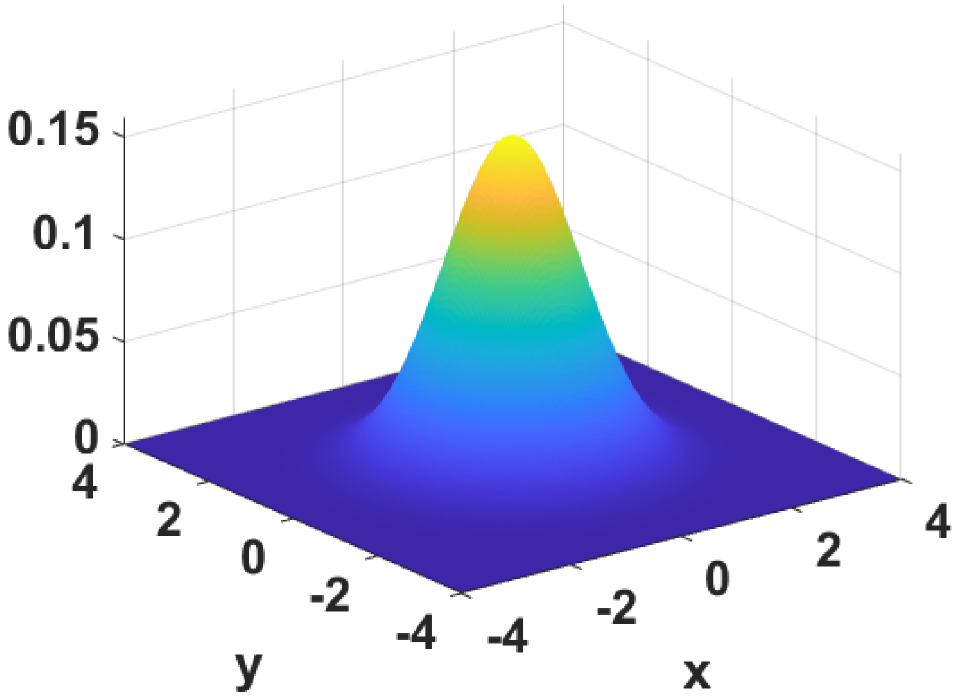
# Covariance matrix adaptation evolution strategy (CMA-ES)

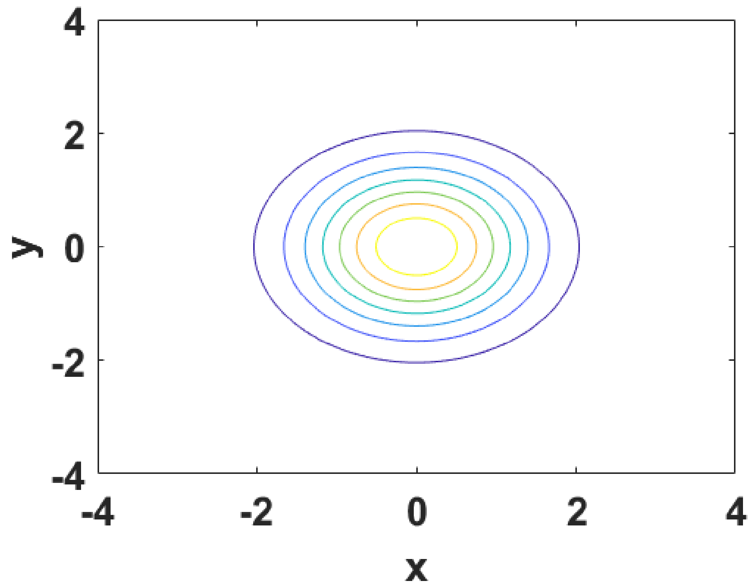
- EDA → we need a **model**
- Smapling using multi-dimensional normal distribution
- **Population-based** optimizer
  – Maintains the population of solutions
  – Generates the candidating solutions using one of the solutions from the population

# Multi-dimensional normal distribution

$$\mathcal{N}([0, \dots, 0], \mathbf{I})$$

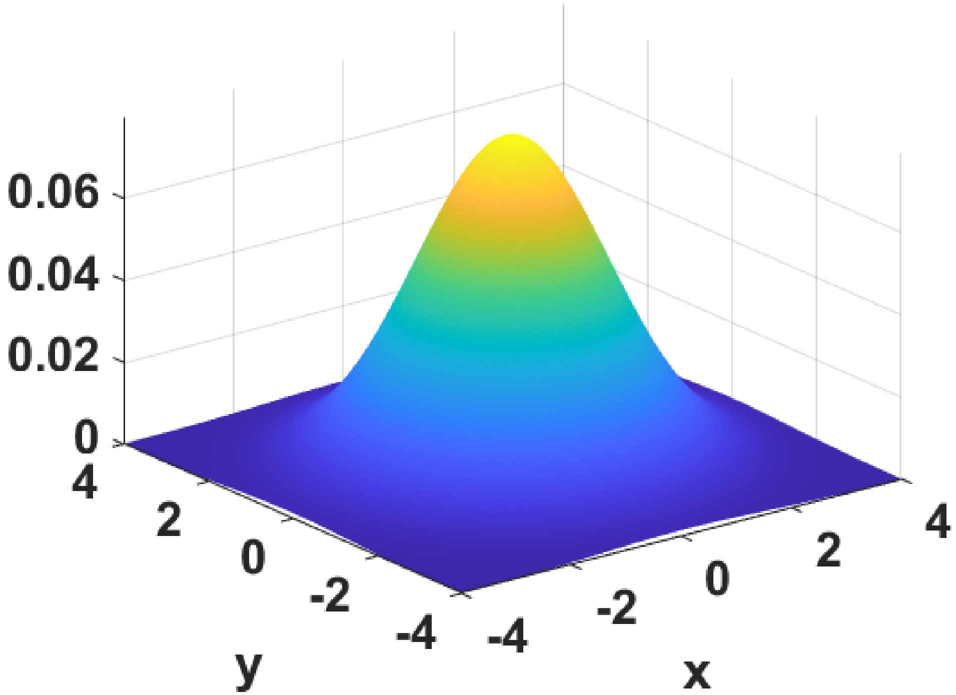
macierz jednostkowa

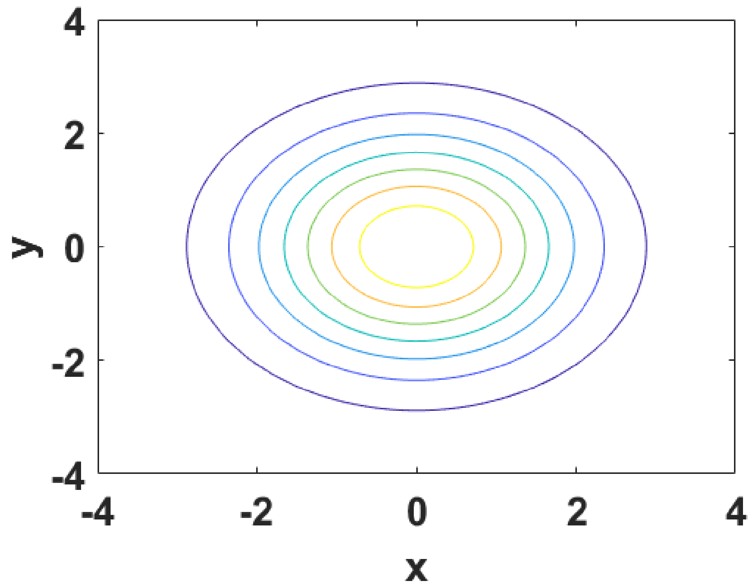


# Multi-dimensional normal distribution

$$\mathcal{N}([0, \ldots, 0], \sigma^2 \cdot \mathbf{I})$$

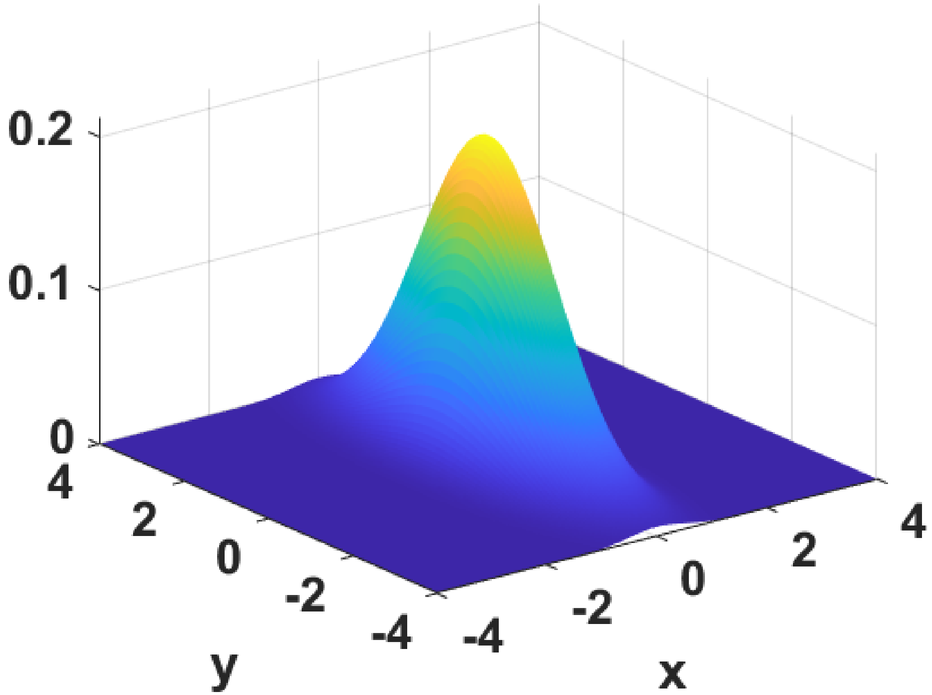


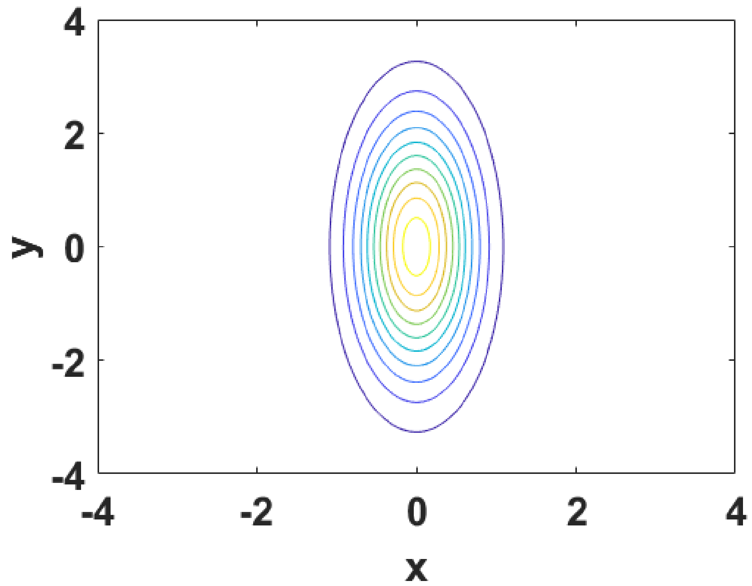



# Multi-dimensional normal distribution

$$\mathcal{N}([0,\ldots,0],\mathbf{D}^2)$$

macierz diagonalna

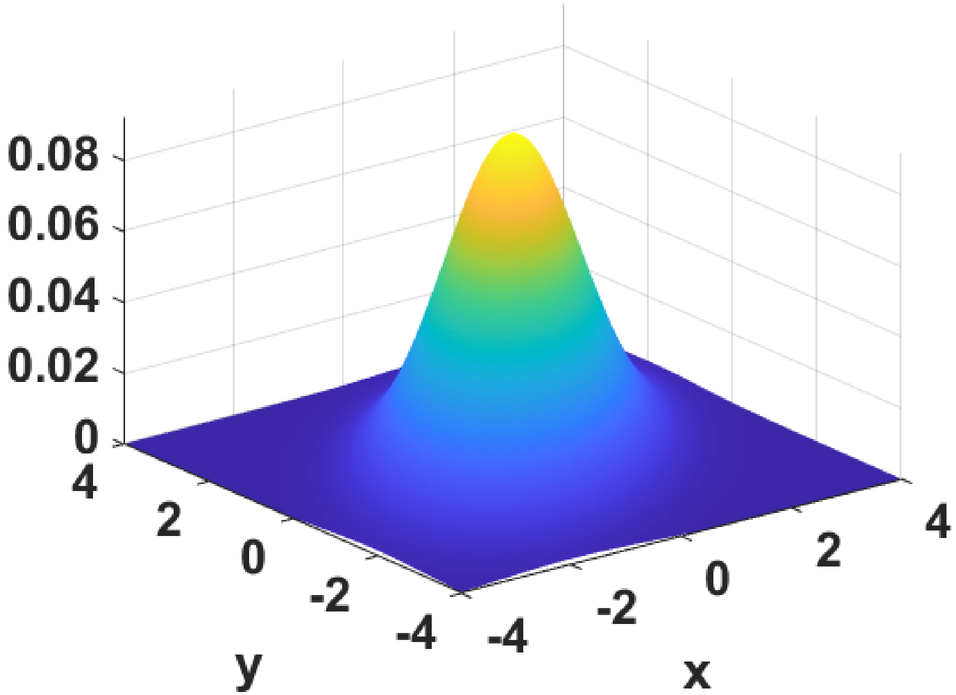


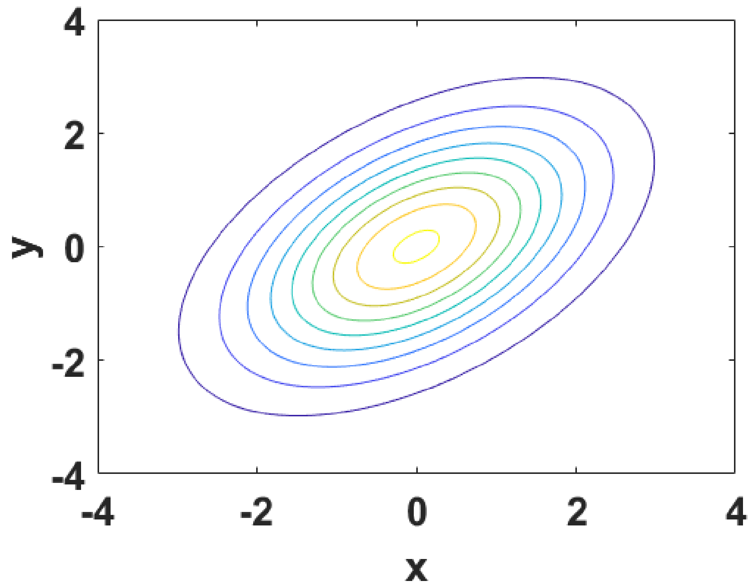



# Multi-dimensional normal distribution

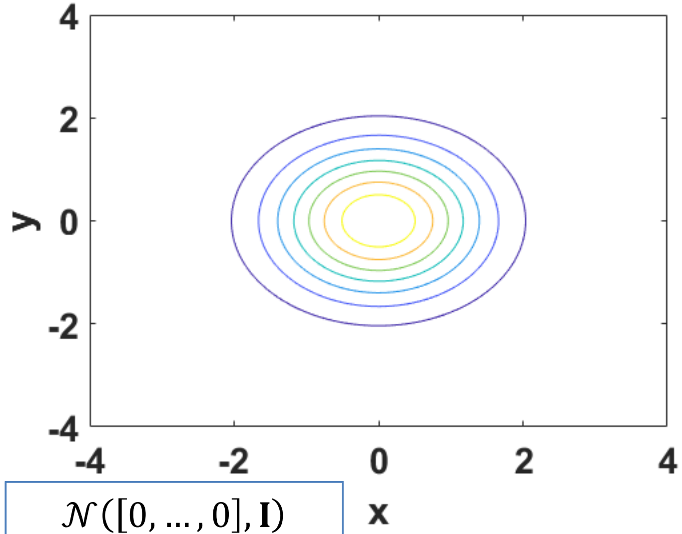
$$\mathcal{N}([0,\dots,0],\mathbf{C})$$

pełna macierz kowariancji

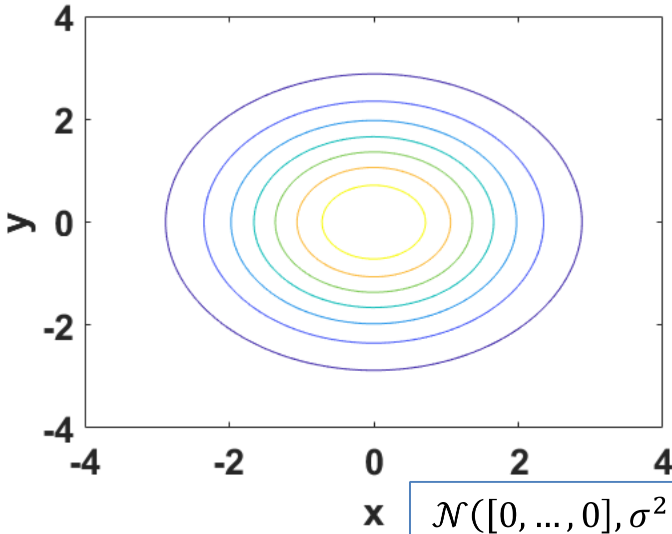


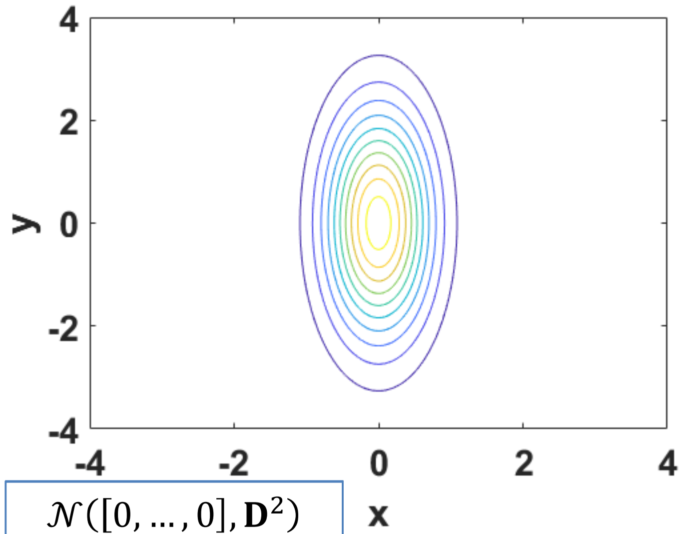
# Multi-dimensional normal distribution



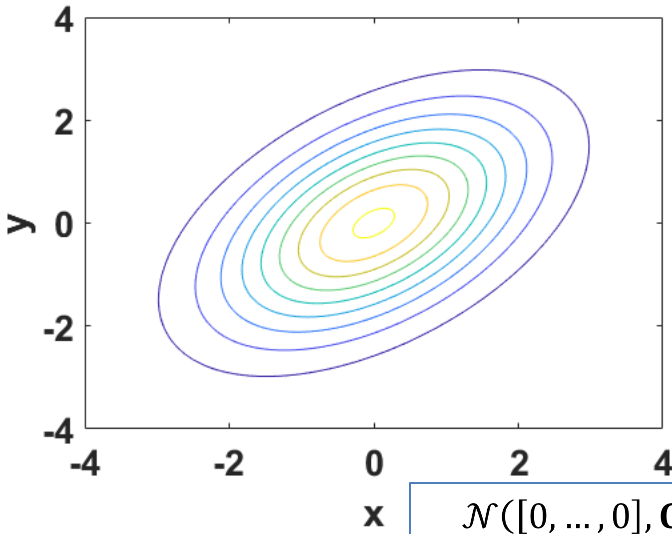

$\mathcal{N}([0, \ldots, 0], \mathbf{I})$




$\mathcal{N}([0, \ldots, 0], \sigma^2 \cdot \mathbf{I})$



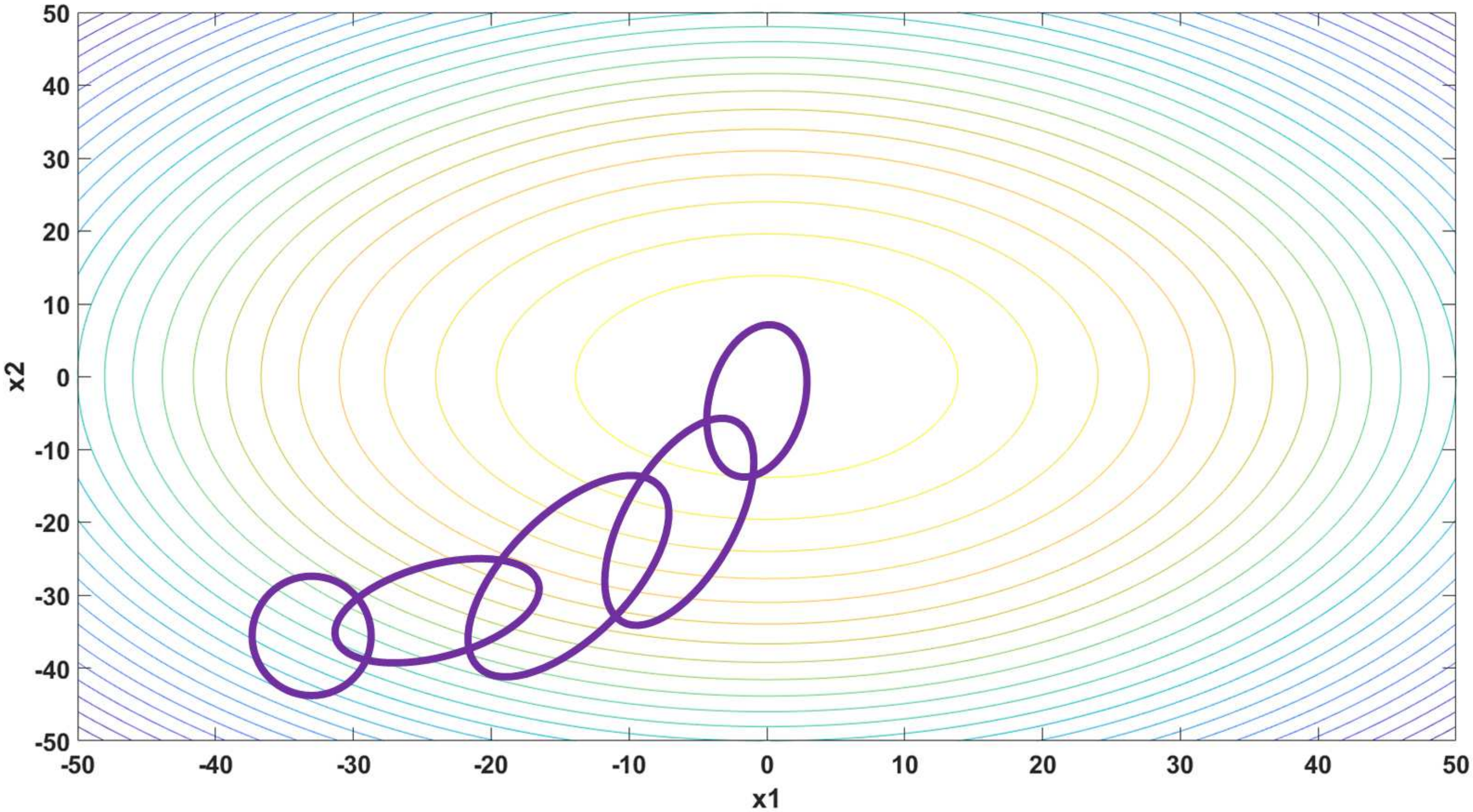

$\mathcal{N}([0, \ldots, 0], \mathbf{D}^2)$




$\mathcal{N}([0, \ldots, 0], \mathbf{C})$

# CMA-ES

## Intuicje



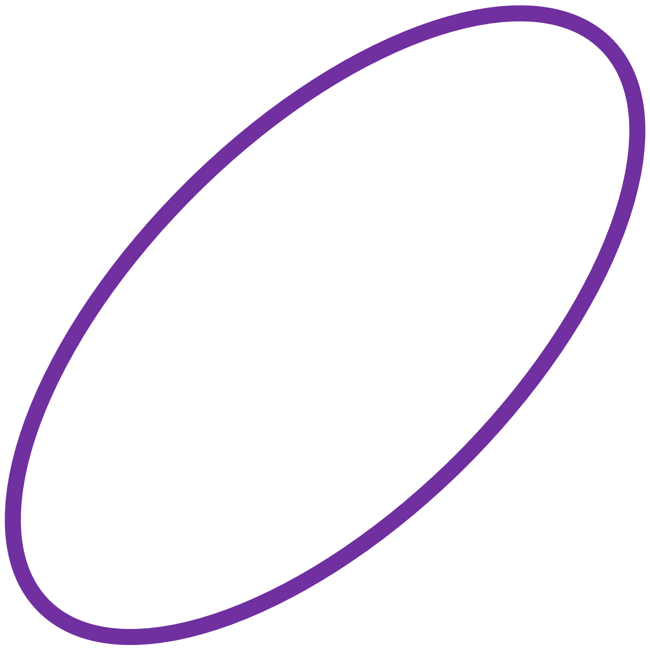
# CMA-ES

- **Model**: multi-dimensional normal distribution
- **Creation** of a single solution $\vec{x}$:

$$\vec{x} = \vec{m} + \sigma \cdot \mathcal{N}([0, \dots, 0], \mathbf{C})$$

- Model parameters (**change** during the run):
  – Average vector ($\vec{m}$)
  – Step size ($\sigma$)
  – Covariance matrix ($\mathbf{C}$)

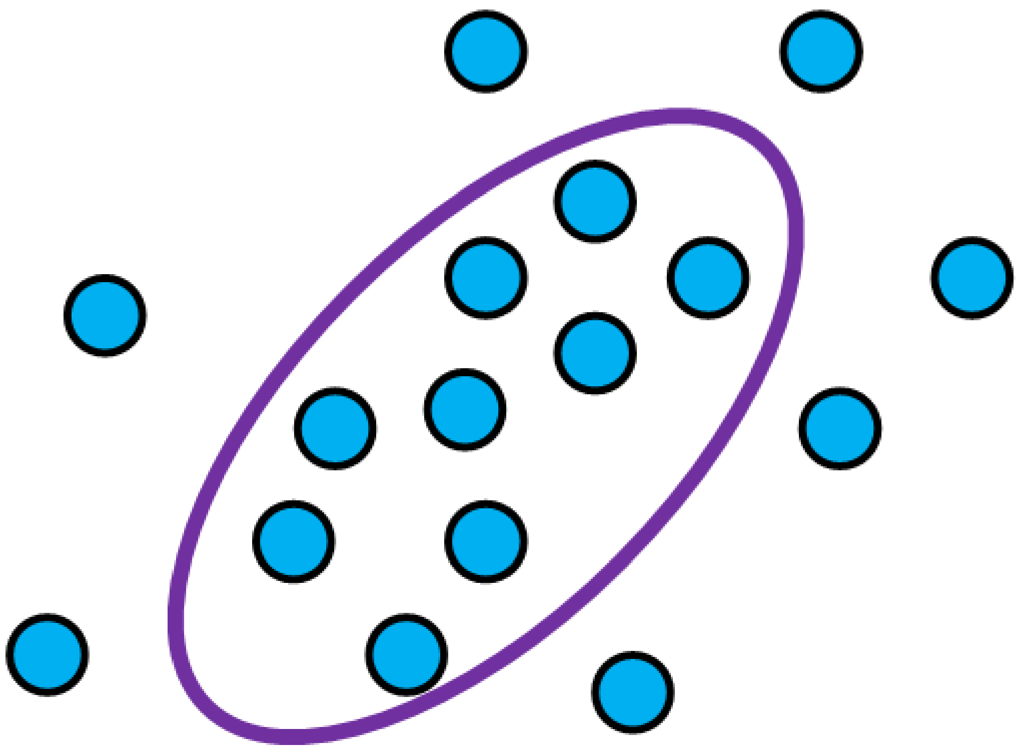
# CMA-ES
## Model parameters



- Average vector is the middle
- Shape defined by
  - Step size
  - Covariance matrix

# CMA-ES
## Adaptation $\vec{m}$ and C

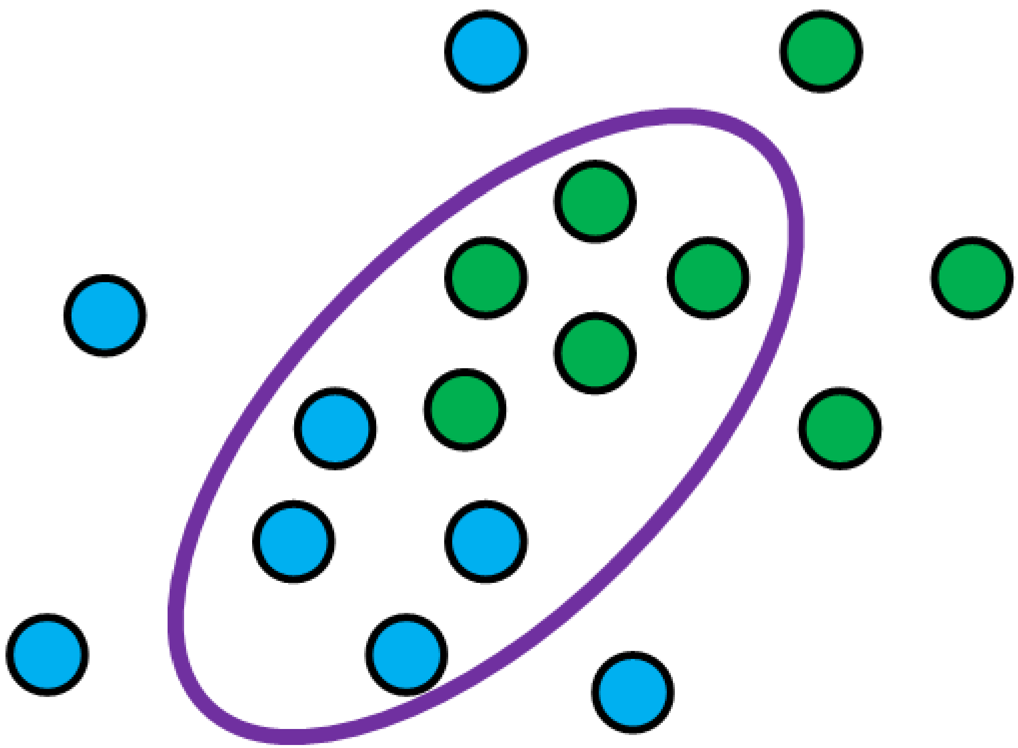
Generate $\lambda$ solutions using the model



# CMA-ES

## Adaptation $\vec{m}$ and C

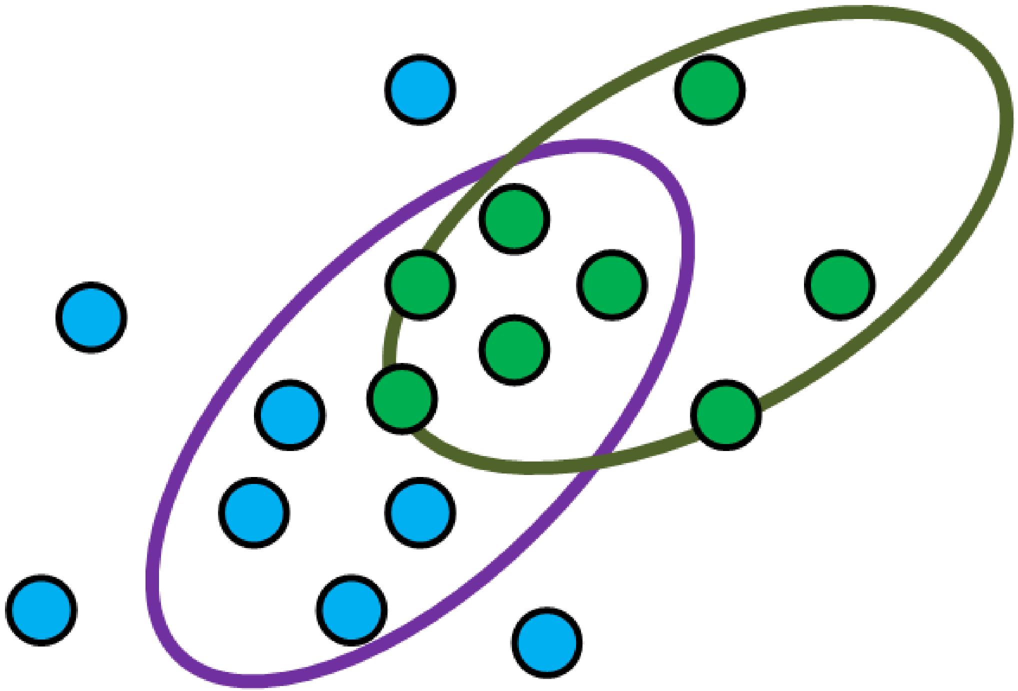
Find $\mu = \lambda/2$ best solutions



# CMA-ES

## Adaptation $\vec{m}$ and C

Update $\vec{m}$ i $\mathbf{C}$ using $\mu$ best solutions – better fitting solutions are more influencive

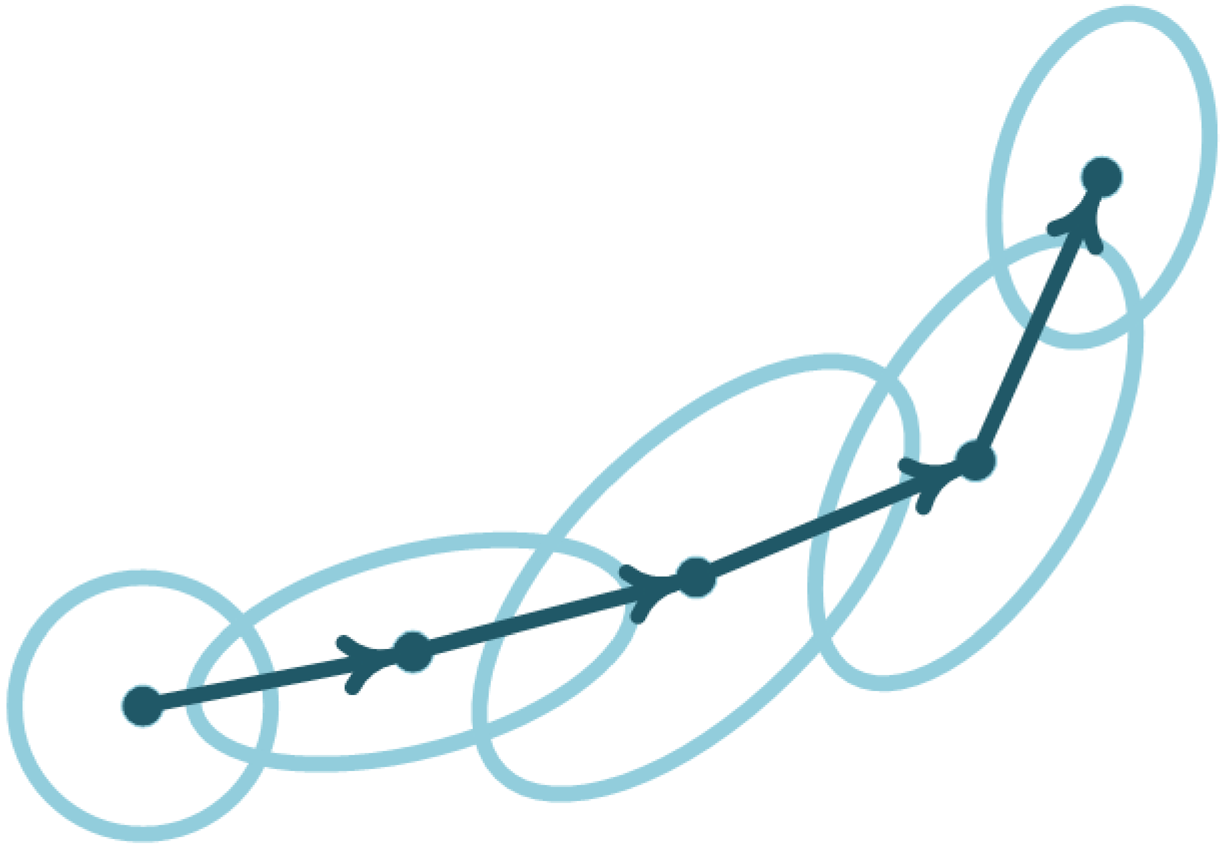


Wrocław University of Science and Technology

# CMA-ES

## Evolutionary path

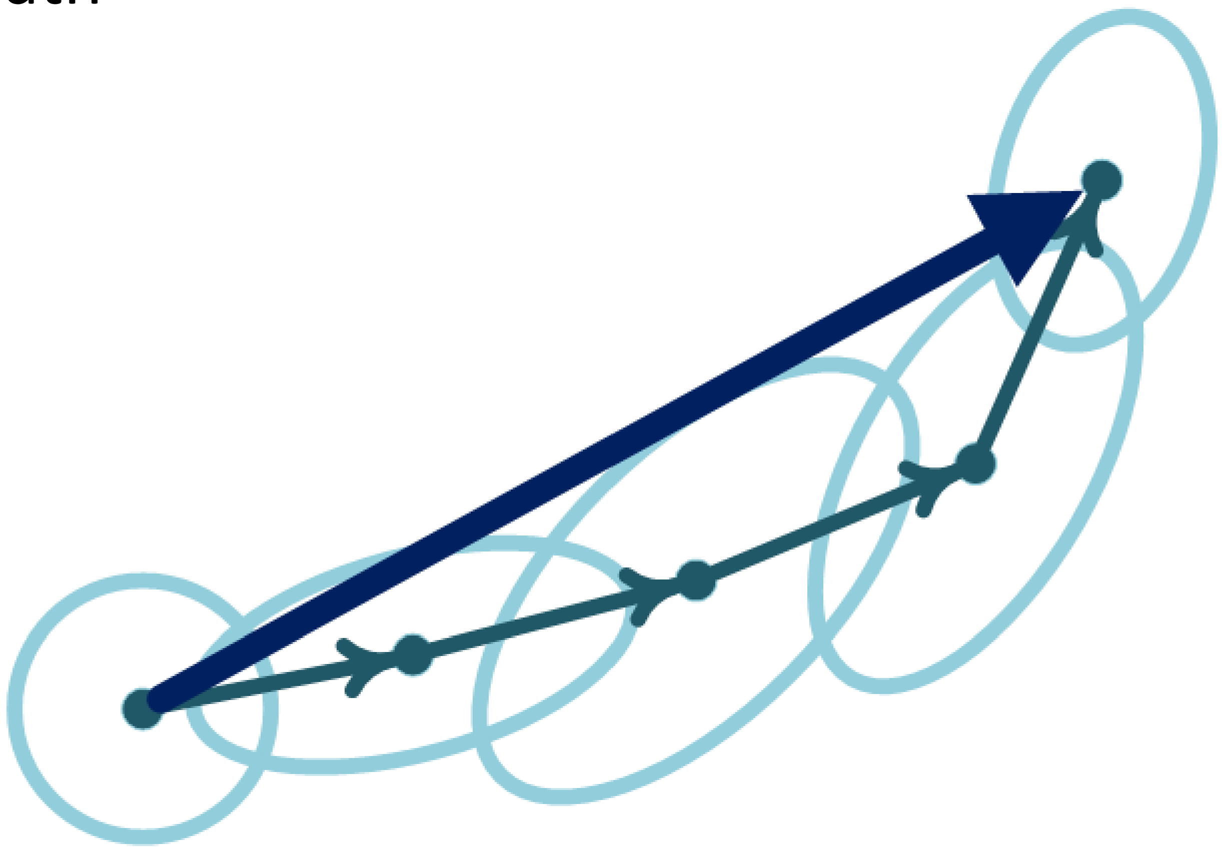
The path joining subsequent models



# CMA-ES

## Evolutionary path

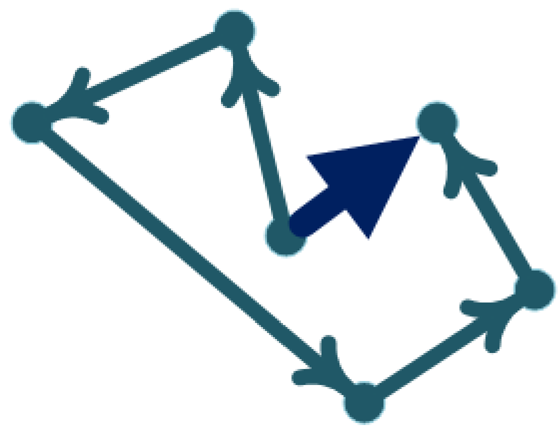
Summarized path connects the beginning and the end of the path



# CMA-ES

## Step size adaptation

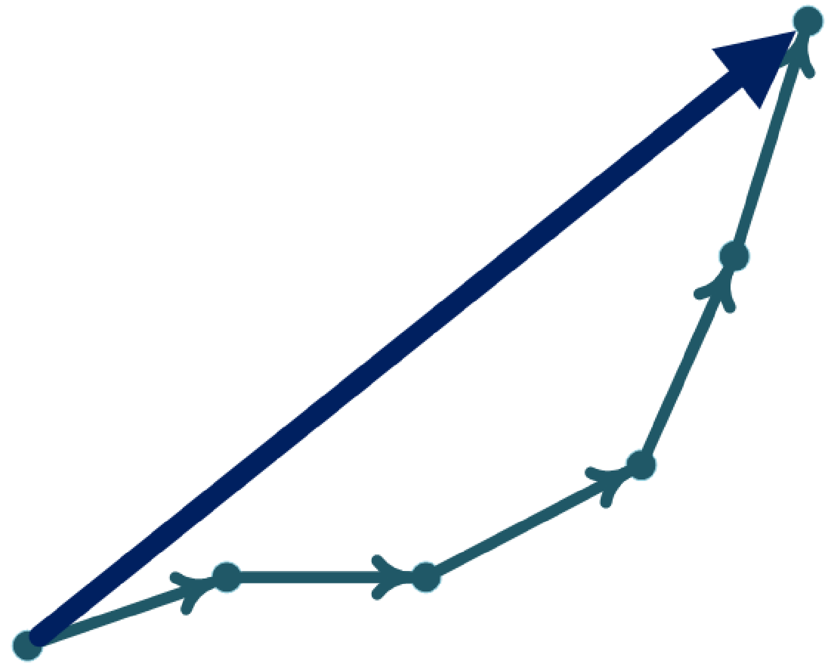
- **Short** summary path ➔ **decrease** the step size
- Subsequent steps are against each other – they lead in different directions



# CMA-ES

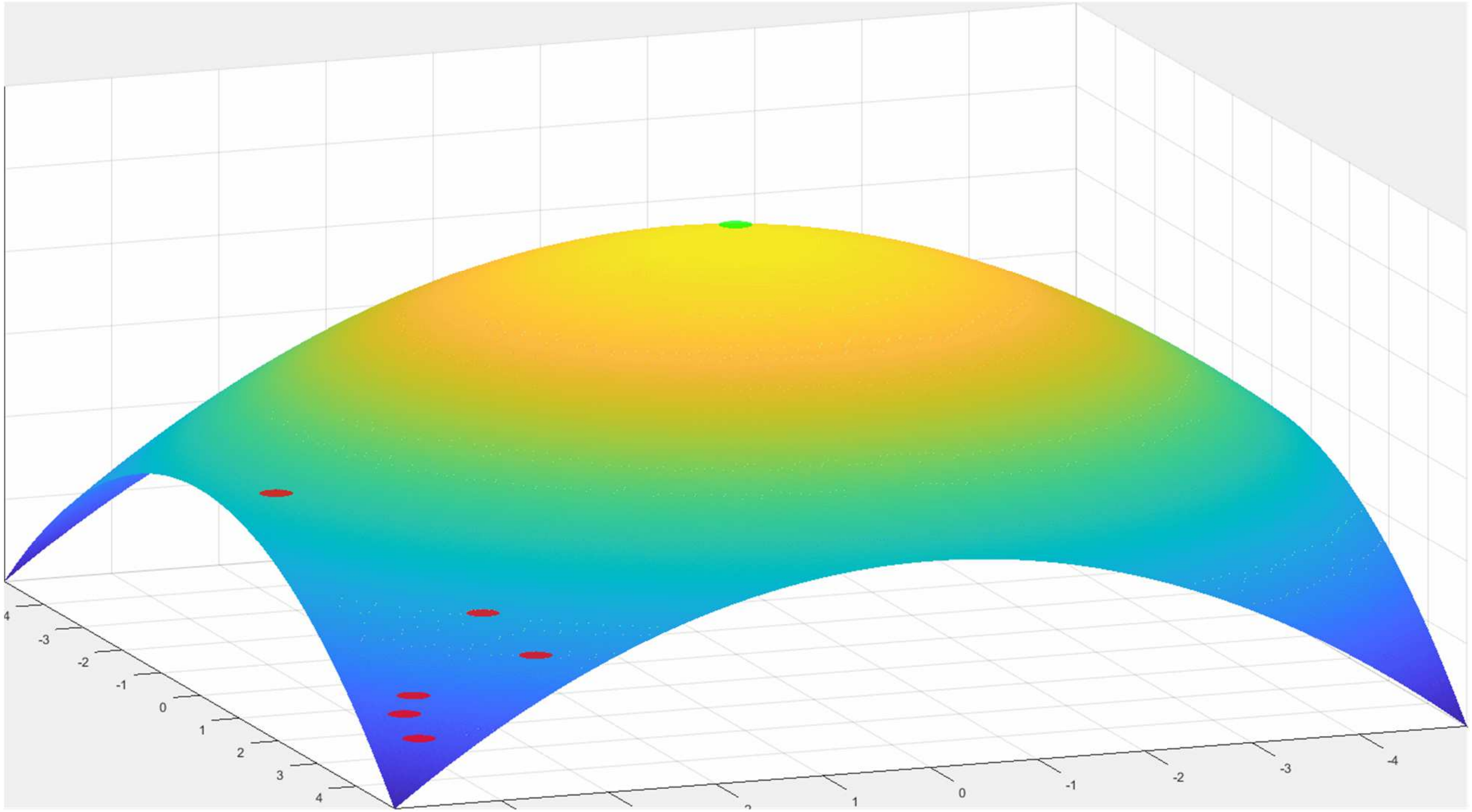
## Adaptacja wielkości kroku

- **Long** summary path → **increase** step size
- All steps made in the similar direction → they can be replaced with the larger one

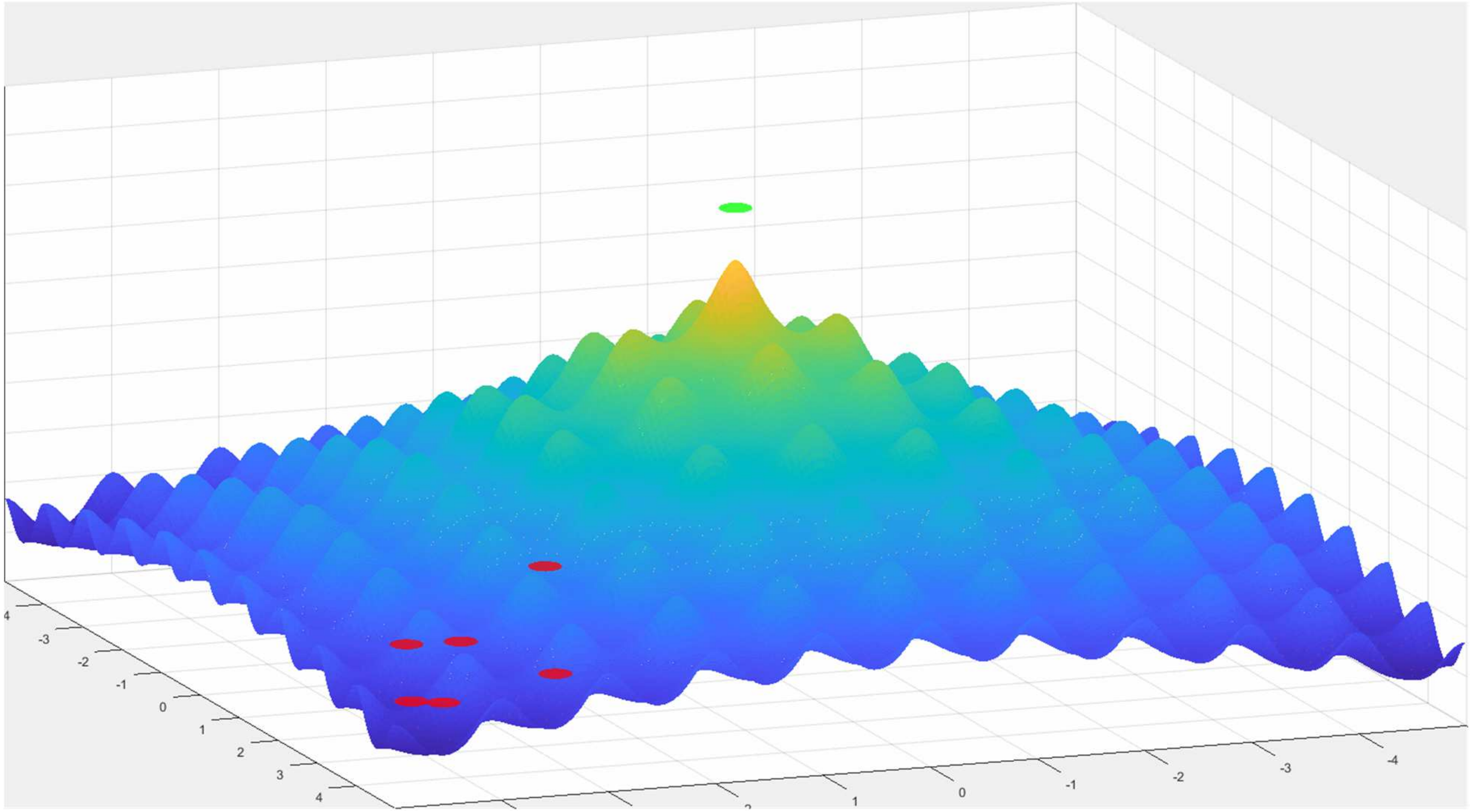


# CMA-ES

## Sphere function

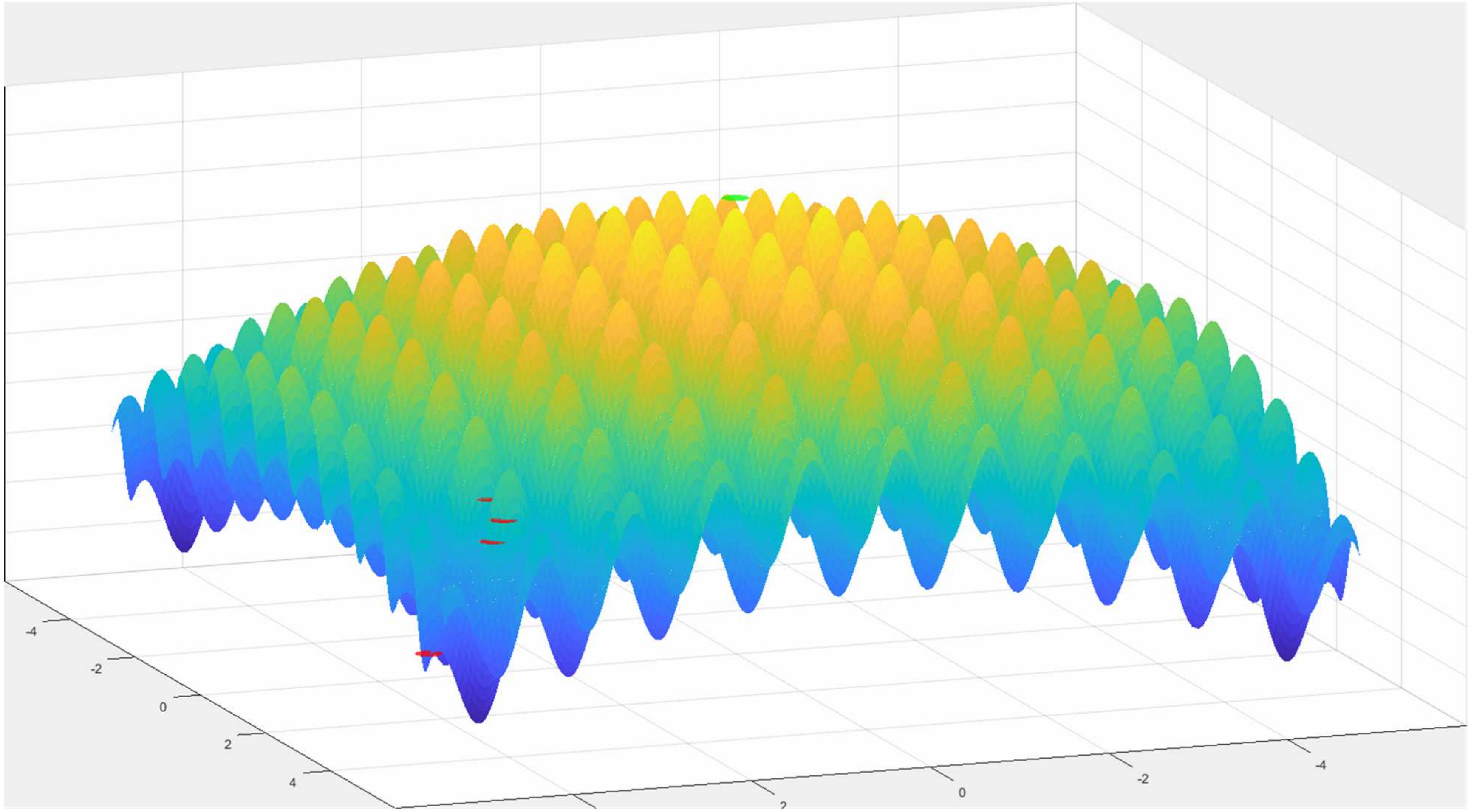


# CMA-ES
## Ackley's function



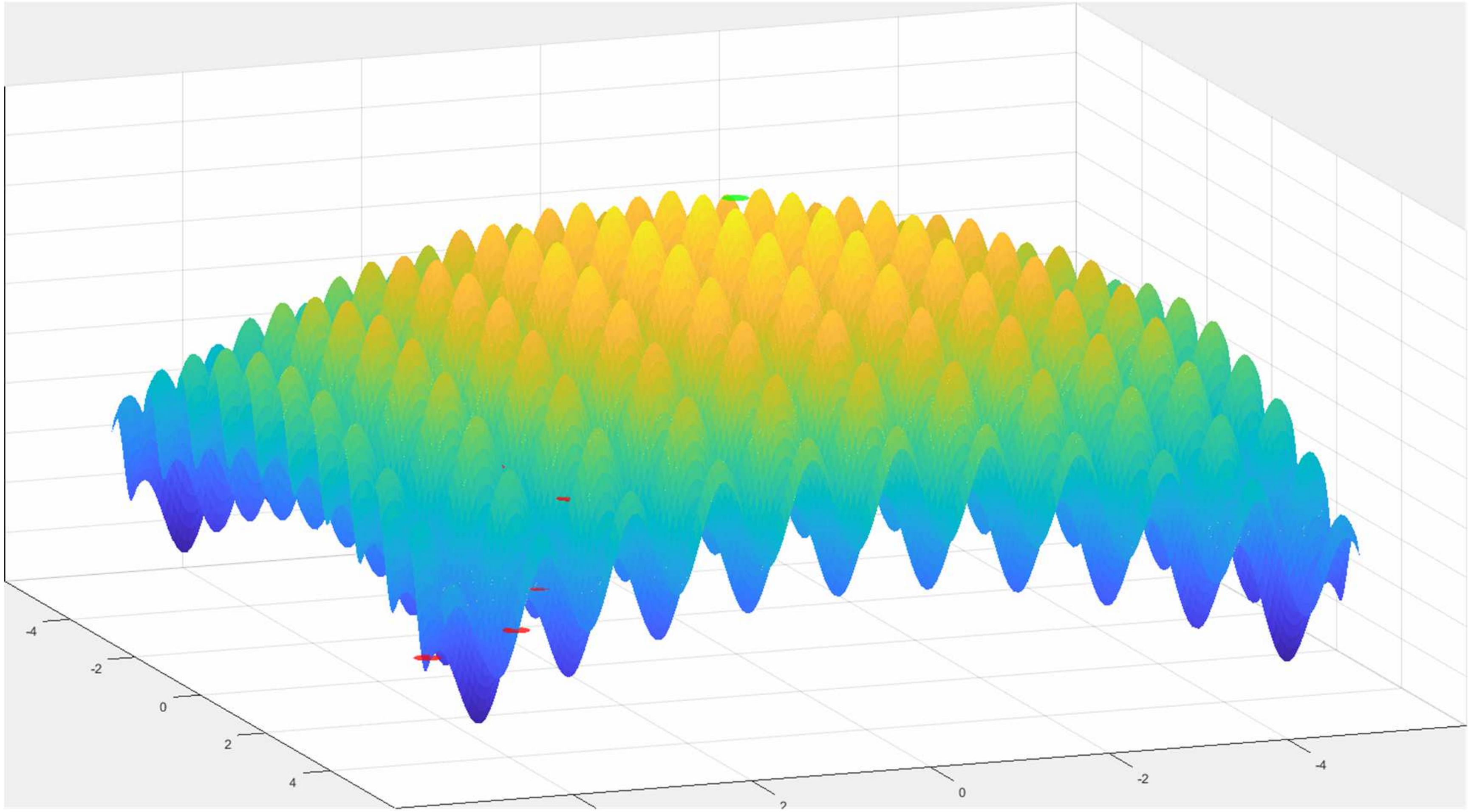
# CMA-ES

## Rastrigin's function



# CMA-ES (stucked)

## Rastrigin's function



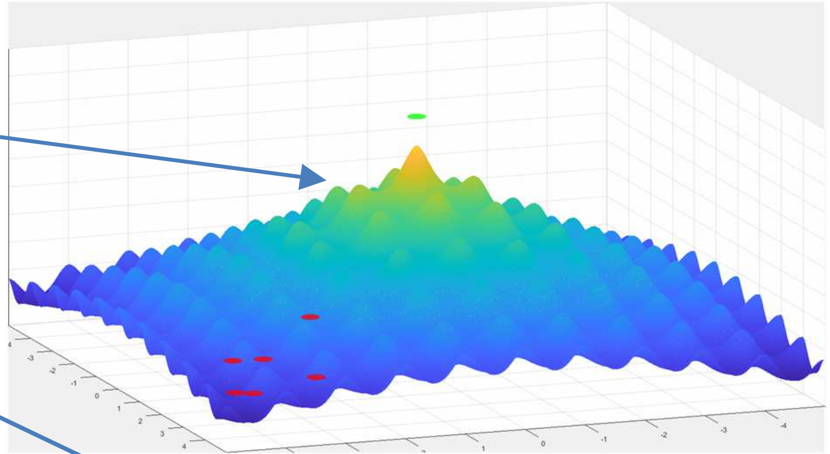
# CMA-ES

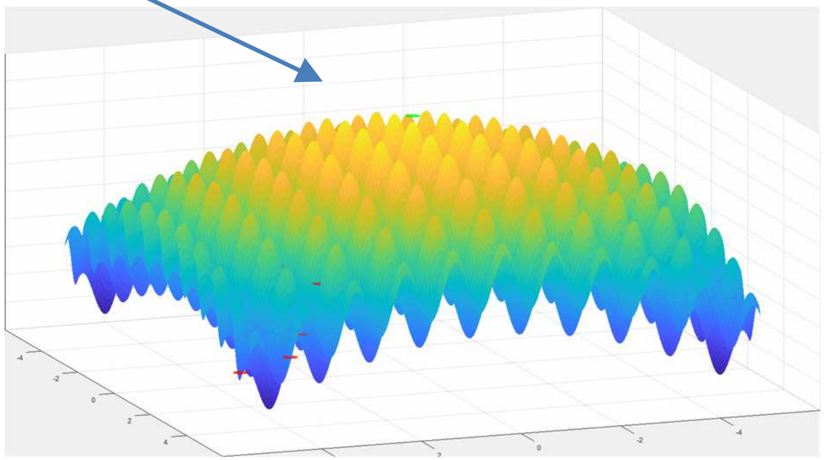
## Why did it stuck?

- Ackley's – **does not stuck**
- Rastrigin's – **it stucks**

**WHY?**

- **Ackley – the „hills" of different height**
- **Rastrigin – the „hills" heights are almost identical**
- **In effect, for Rastrigin: exploration in CMA-ES does not work**





# CMA-ES

## What to do when we're stuck?

- Restarts
  - Primitive…
  - …yet frequently effective
- **CMA-ES – super-cool local optimizer**
  - Adaptation
  - Model-based neighbourhood analysis
  - Effective
  - High exploration capability
  - Disadvantages: intuitions behind CMA-ES – strongly inspired by local optimization

# CMA-ES

## Summary

- EDA – sampling fro mmulti-dimensional normal distribution
- Directed search using using adaptation of shape a model
- It can stuck in local optimum – more recent versions use restarts to mitigate this issue

# CMA-ES
## Summary

- CMA-ES:
  - Continuously developed
  - State-of-the-art for many continuous problems

H. Beyer and B. Sendhoff, "Simplify Your Covariance Matrix Adaptation Evolution Strategy," in *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 5, pp. 746-759, 2017.

J. Arabas and D. Jagodziński, "Toward a Matrix-Free Covariance Matrix Adaptation Evolution Strategy," in *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 84-98, 2020.

# Summary

- Intuitions behind modern optimizers
  - Frequently based on simple observations
  - **Easy to understand, but very hard to guess without thorough analysis**
- Key-terms – intuitions + understanding
  - Exploitation
  - Exploration
  - Adaptaion – usually much better choice than user-based parameter tuning
- Problem features – every problem is different (has unique features)
- Perfect (optimal) optimizer – **forget it!**

# Comming soon…

- **Discrete serach spaces strike back…**
- Genetic Algorithms – introduction
- Population-based optimizers – main intuitions
- Most important improvements for Gas
- Why does it work – the theory