Wrocław University
of Science and Technology

Optimization Methods: Theory and Applications
GA sweetspots
diversity preservation, building block recombination, preconvergence

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)”

Wrocław University of Science and Technology

Problem nature reminder

- Travelling Salesman Problem (TSP) – ***combinatorial*** *in nature* **<- this we are going to solve today**
  - **We want to exchange solution fragments; For instance:**
    - **„good" city sequences (TSP problem)**
    - **„good" item groups (Knapsack problem)**
- Hill – ***topological*** *in nature*
  - 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"

GA Sweetspots

- GA sweespots
  - Diverse population → many building blocks (high-quality *schemata – do you remember the Schema Theorem?*)
  - Building block recombination
- Building block
  - Group of **highly dependent** genes
  - **With values!!!**
  - We can ***assemble*** building blocks to get the ***high-quality solution***
- David Goldberg
  - One of the pioneers of Evolutionary Computation
  - Extraordinary influence on the whole domain
  - His former PhD students are now the world elite (e.g., Kalyanmoy Deb, Dirk Thierens)

D.E. Goldberg, "The Race, the Hurdle, and the Sweet Spot: Lessons from Genetic Algorithms for the Automation of Design Innovation and Creativity", IlliGAL Report No. 98007, 1998

Building blocks

- Standard deceptive function – **remainder**

$$f_{order-k}(\vec{x}) = \begin{cases} u(\vec{x}) & if \quad u(\vec{x}) = k \\ k - u(\vec{x}) - 1 & if \quad u(\vec{x}) < k \end{cases}$$

Where:

- $\vec{x}$ - binary vector (solution)
- $u(\vec{x})$ - unitation (the number of ‚1's)
- *k* – the order of the deceptive function

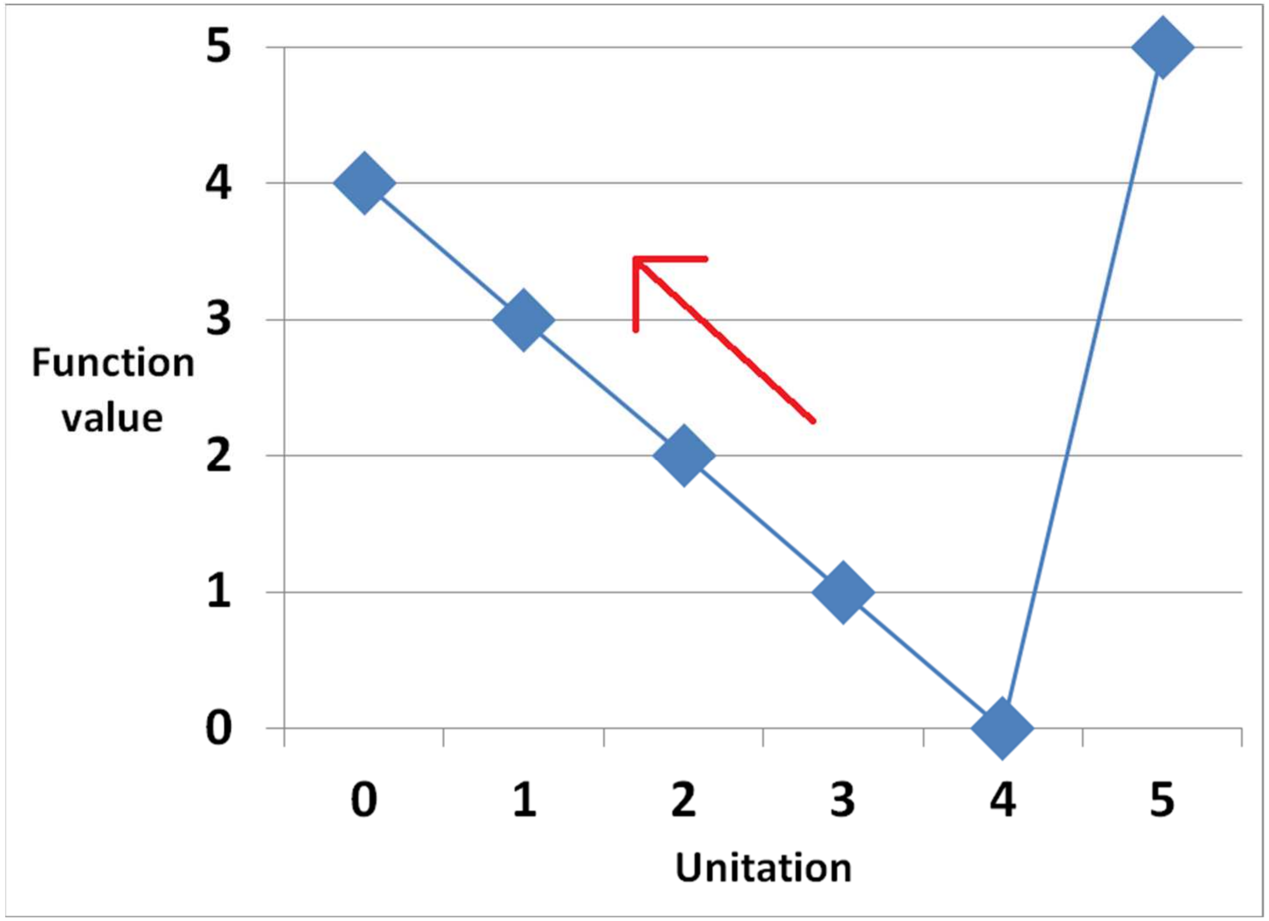
Wrocław University of Science and Technology

Building blocks

Standard deceptive function – **remainder**

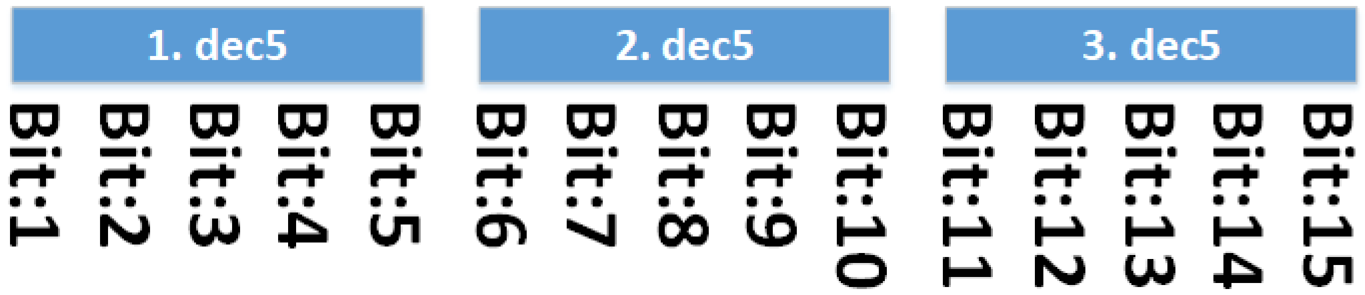
$f_{order-5}(x)$:

Unitation	Wartość
0	5
1	0
2	1
3	2
4	3
5	4



Building blocks

The concatenation of 3 order-5 deceptive functions:

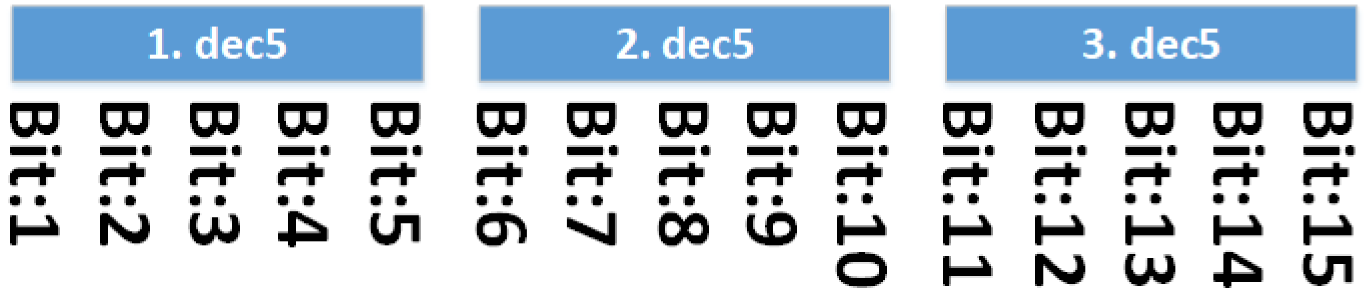


Note that:

- We have three (additively separable) blocks
- You can optimize each block separately…
- …**IF you know**:
  - How many blocks do we have
  - Which genes refer to which block

Building blocks

The concatenation of 3 order-5 deceptive functions:



Building block: the values of highly dependent genes, can build a high-quality solution

QUESTION: Can you give examples of building blocks for the problem defined above?

- ****1** **00* **101 NO**
- ******* 01010 **101 NO**
- **11100 ***** ***** NO**

- ******* 11111 ***** YES**
- ******* ***** 00000 YES**
- **11111 11111 ***** NO**

Building blocks

- Note that:
  - Buidling block is a **high-quality schema**
  - But it's as short schema as possible
    - e.g., **11111 11111** ***** → blocks 1-5 i 6-10 are not connected → it **is NOT a** building block
    - BUT: if **schema 11111 11111** ***** is represented in the population → **that's good!**
  - ***** ***** **00000** is a building block, although it **IS NOT** a part of the optimal solution

Building blocks

- Building blocks – why do we talk about it?
  - You have something to talk about on the lecture…
  - You want to solve the problem?
    → **exchange building blocks!**
  - You want to exchange building blocks?
    → **you need to know where they are**
  - **How many schema do we have in the population?**
    - From 2^n (all individuals are the same)
    - Up to $pop \cdot 2^n$
  - You want to have building blocks?
    → **you need to have a diverse population**

Population diversity

- Population diversity preservation – chosen techniques
  - Global mutation
  - Fitness sharing
  - The Baldwin effect (but **not** the Lamarck effect)
  - Island Models
  - Population-sizing
  - New population management techniques $\rightarrow$ population pyramid **(next lecture)**
- Objective: obtain and preserve the set of valueable building blocks

Global mutation

- We optimize…
- …at some point we get stuck…
- …after some time we detect that we got stuck and…
- …we perfom the global mutation…

Global mutation

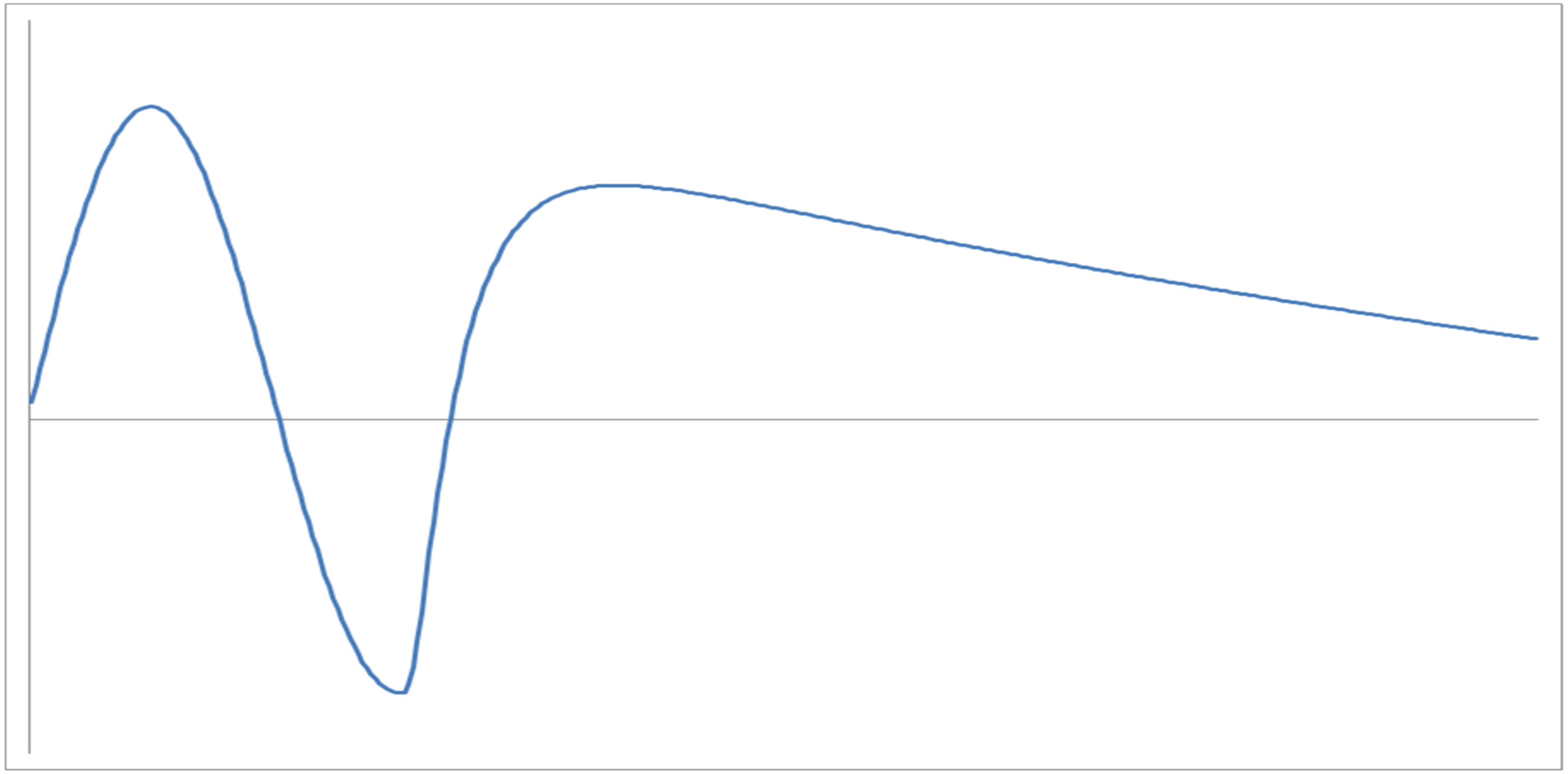
- Global mutation
  - We choose some part of the population (usually a large part, e.g., 50%)
  - For every chosen individual → we randomly reinitialize a large part of the genotype
- Objective: intrudce new schemata (i.e., building blocks) into the population
- Advantage: simplicity

Global mutation

- Disadvantages
  - Blind-luck mechanism
    - We do not know how many genes should be exchanged
    - We do not know which genes should be exchanged
  - Individuals (globally) mutated → significantly lower fitness (when comapred to the non-mutated individuals)
    - Non-mutated individuals will quickly dominated the (globally) mutated ones (Shall we keep them in one population?)
    - Introduce a quarantine?
    - Separate (globally) mutated and non-mutated individuals?
  - Simple mechanism, **but**:
    - It may get complicated if you want to use it effectively
    - It may be ineffective anyway

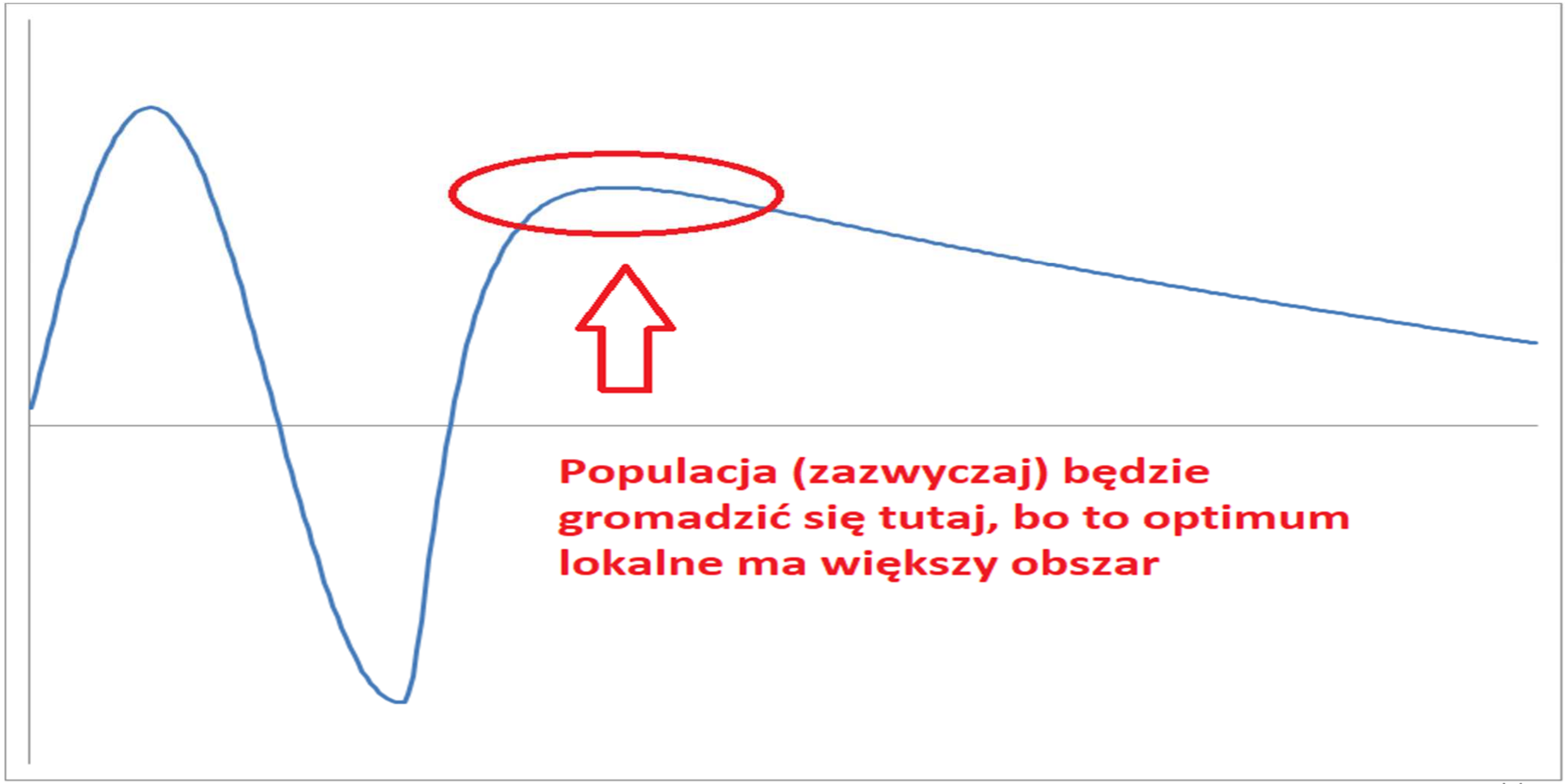
Fitness sharing

- Task: maximization
- Domain: 1 dimension (coordinate X)
- Question: to which optimum the population is most likely to converge?



Fitness sharing

How to push the population to the other hill?



Fitness sharing

- How about:
  - Scaling fitness according the individual’s *originality*?
  - Individual’s *originality* (*rarely* met) → receives *normal* fitness
  - *Typical* individual (there are many similar or the same individuals) → we **decrease** fitness
- Then:
  - Individuals from hill of small size (but high quality) will have a higher chance to become parents…
  - …because although **their number is lower**, **they are more original**

Fitness sharing

- We modify fitness:

$$f(x_i, P) = \frac{fitness(x_i)}{\sum_{ind \in P} sh(x_i, ind)}$$

where:

x_i - oceniany osobnik

$sh(x_i, ind)$- sharing function

$$sh(x_i, ind) = \max\left\{0{,}1 - \left(\frac{d(x_i, ind)}{\sigma}\right)^{\alpha}\right\}$$

Fitness sharing

- We modify fitness:

$$f(x_i, P) = \frac{fitness(x_i)}{\sum_{ind \in P} sh(x_i, ind)}$$

$sh(x_i, ind)$- sharing function

$$sh(x_i, ind) = \max\left\{0; 1 - \left(\frac{d(x_i, ind)}{\sigma}\right)^{\alpha}\right\}$$

$d(x_i, ind)$ - the distance between x_i and *ind*

σ- sharing range

Fitness sharing

$$sh(x_i, ind) = \max\left\{0; 1 - \left(\frac{d(x_i, ind)}{\sigma}\right)^{\alpha}\right\}$$

$d(x_i, ind)$ - the distance between x_i and *ind*

σ- sharing range

- **If** $d(x_i, ind) < \sigma \rightarrow sh(x_i, ind) > 0$

Fitness sharing

- We modify fitness:

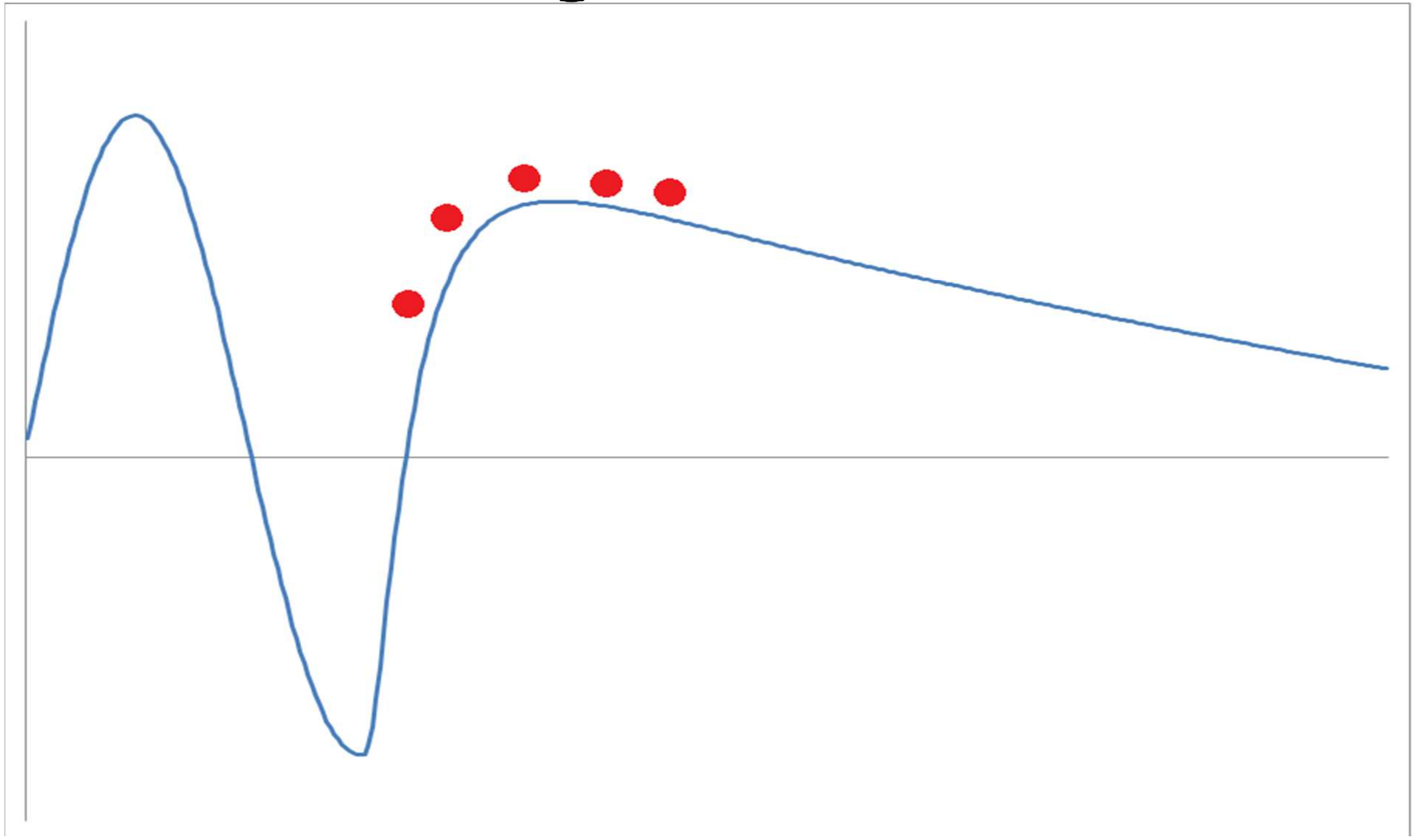
$$f(x_i, P) = \frac{fitness(x_i)}{\sum_{ind \in P} sh(x_i, ind)}$$

$$sh(x_i, ind) = \max\left\{0, 1 - \left(\frac{d(x_i, ind)}{\sigma}\right)^{\alpha}\right\}$$

- For each individual: $\sum_{ind \in P} sh(x_i, ind)$ >= 1
  - Becasue the distance to itself = 0
- **IF**:
  - x_i, → high quality solution
  - Many idnividuals similar to x_i, → fitness may be low

Fitness sharing

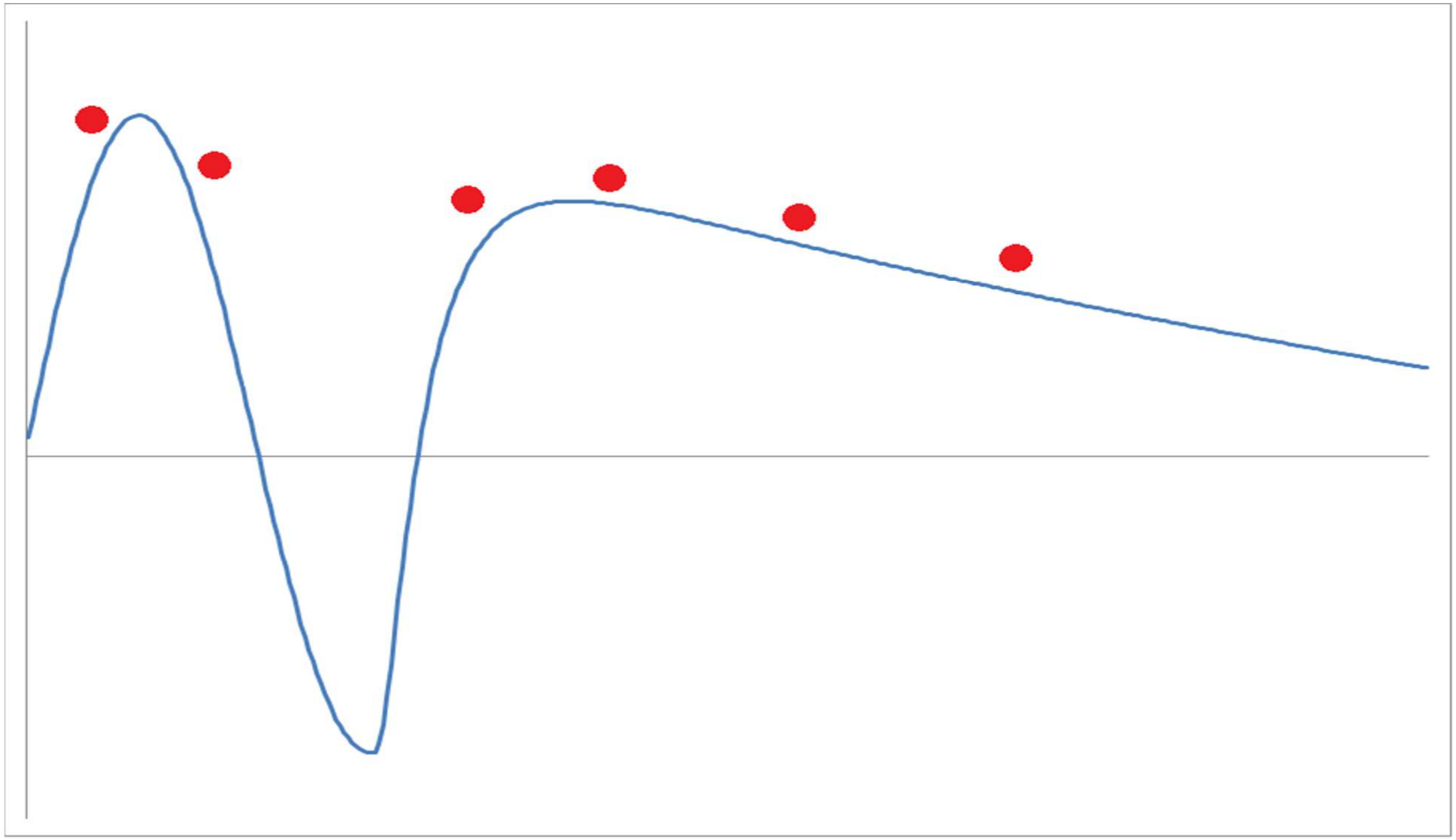
No fitness sharing…



Wrocław University of Science and Technology

Fitness sharing

With fitness sharing…



Fitness sharing

- Implementation → how to cache fitness?
  - We can cache only the unmodified fitness
  - **NOTE: fitness in this case → significantly different than the optimized problem**
- Advantage: increases population diversity
- Disadvantages:
  - High computational cost
  - Additional parameters to tune and set

Wrocław University
of Science and Technology

Baldwin effect

- Local optimization and the individual

  We modify the genotype:

  $$y_i = lopt(x_i)$$

  where

  x_i - the genotype of the *i*th individual

  y_i - the genotype of the *i*th individual **after local optimization**

Baldwin effect

- Baldwin effect:

$$y_i = lopt(x_i)$$
$$fitness(x_i) = fitness(y_i) = fitness(lopt(x_i))$$

  - fitness → from the optimized genotype
  - Original genotype → remains unmodified

- Features:
  - **Significant diversity improvement**
  - **High computational cost**

Lamarck effect

- Lamarck effect:

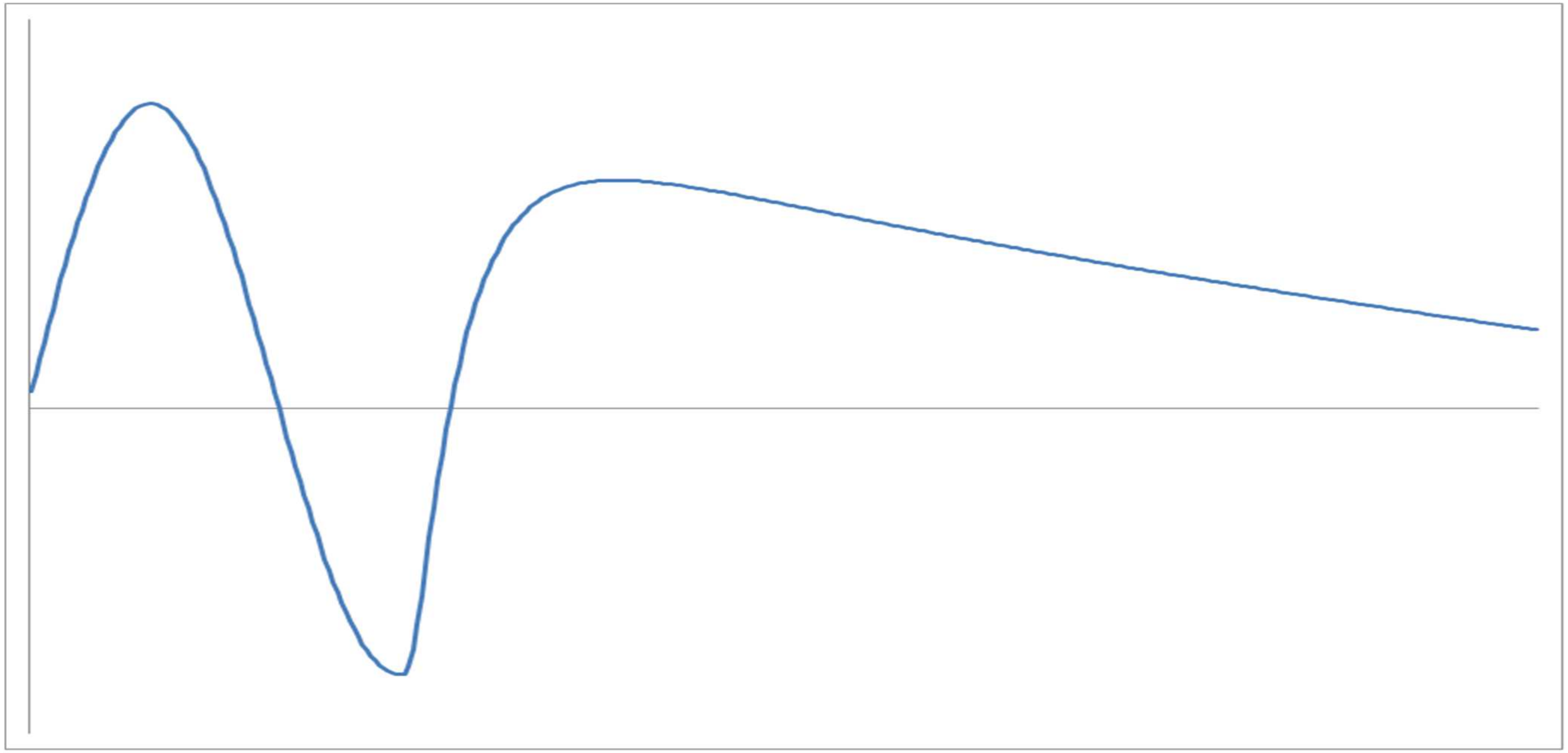
$$x_i = lopt(x_i)$$

$fitness(x_i)$ **- x_i is already optimized**

  - Fast convergence
  - Frequently too fast (**preconvergence**)

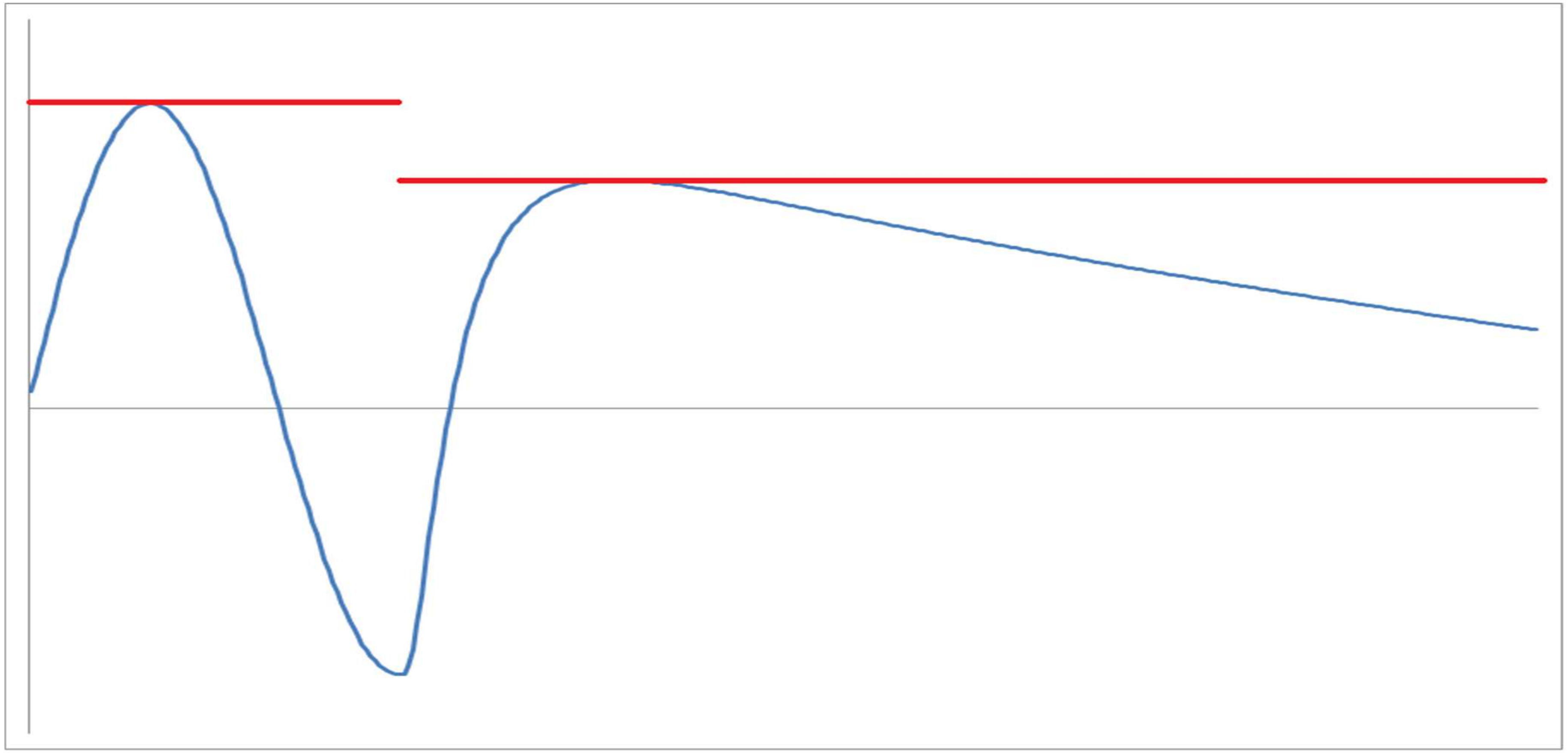
Baldwin effect

Solution space without the Baldwin effect:



Baldwin effect

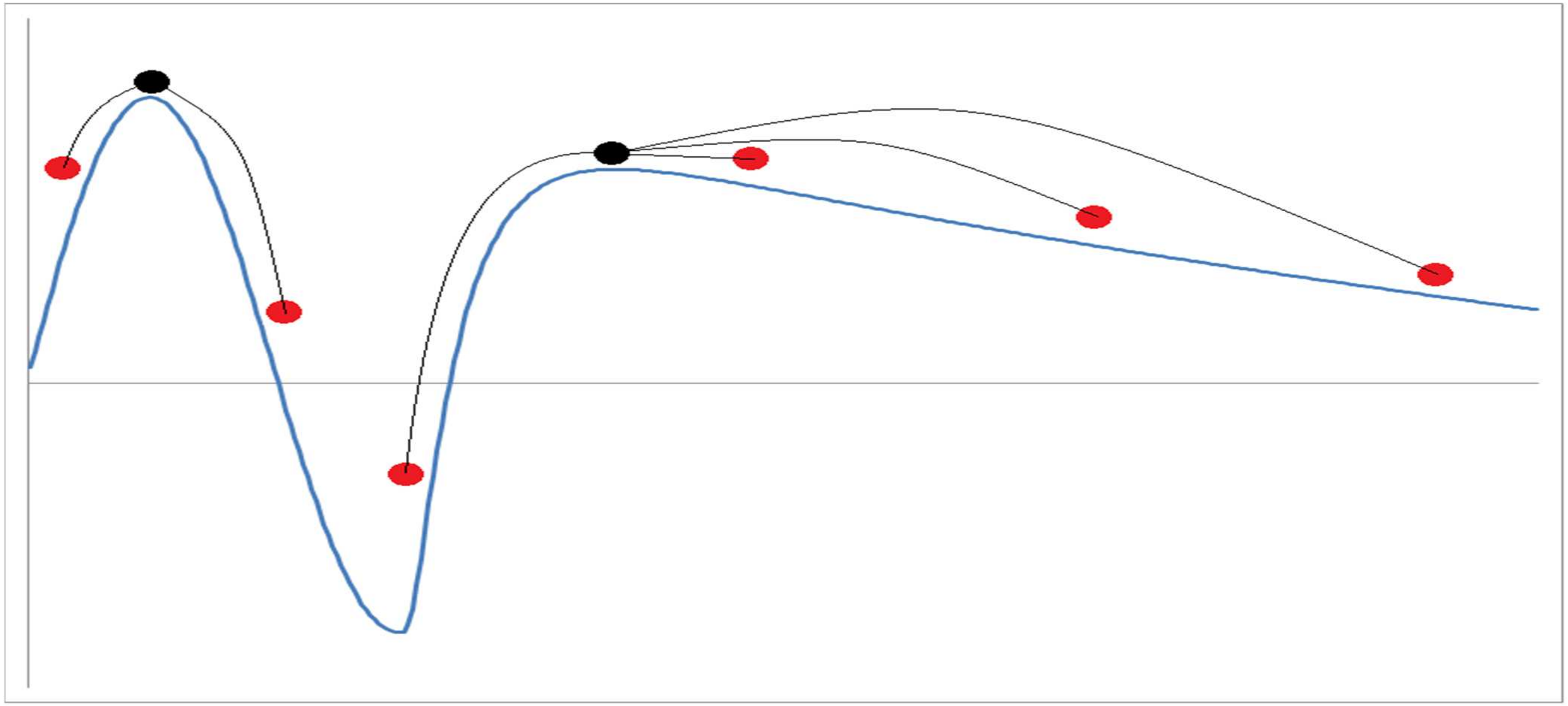
Solution space **with** Baldwin effect (red line):



Wrocław University
of Science and Technology

Lamarck effect

If we are lucky, then **we find the optimum very quickly**…

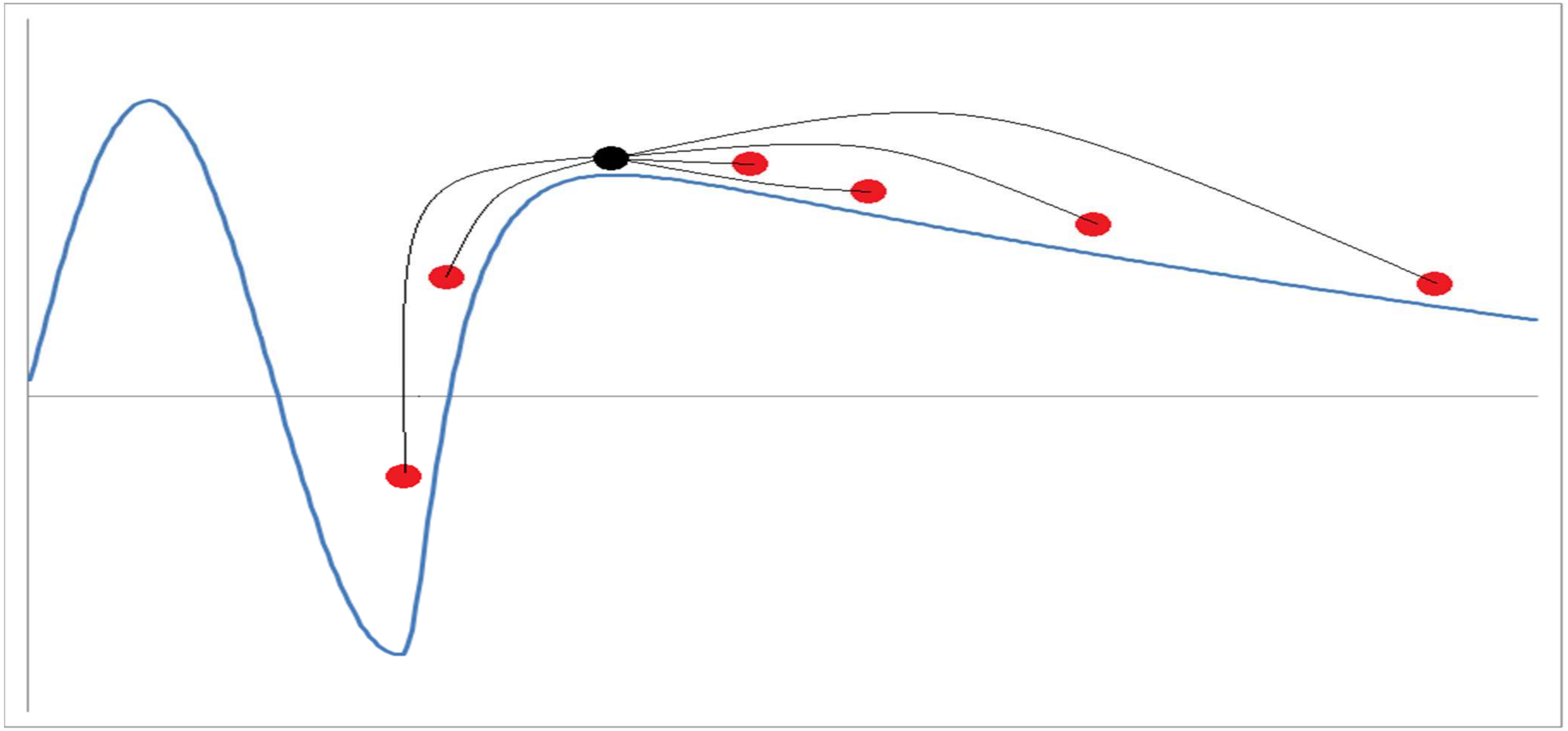


…and we ride towards the setting sun…

Efekt Lamarcka

However, if we are **un**lucky…

…and we usually are…



…then we are stuck for good…

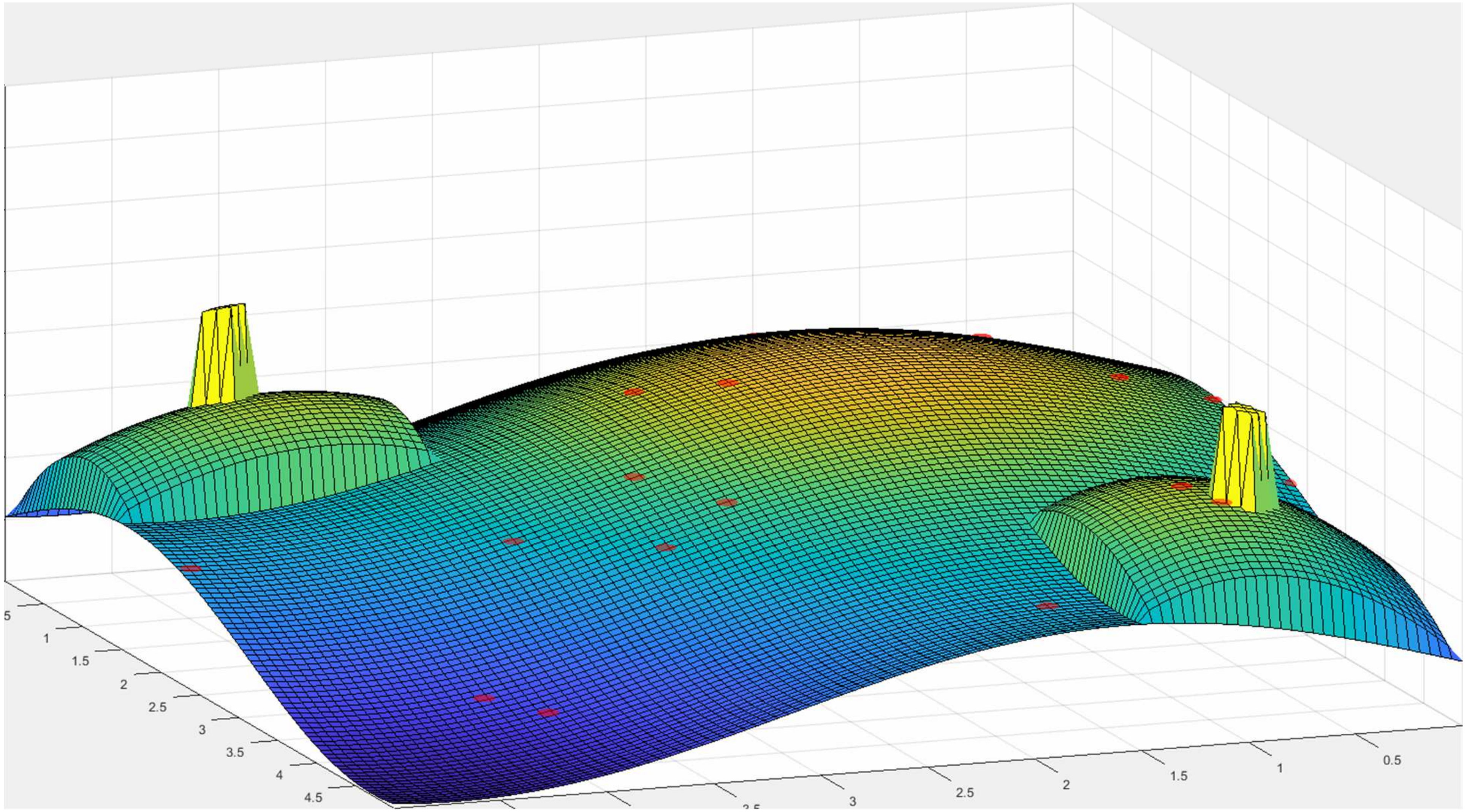
Efekt Baldwina
(ang. *Baldwin effect*)

- Advantages: excellent diversity preservation mechanism
- Disadvantage: highly expensive
- You can join Baldwin and Lamarck effects:
  - One part of the population → Baldwin effect
  - The other part of the population → Lamarck effect
  - Quite frequently used
  - Proportions are highly dependent on the problem

Island Model

- Reminder:
  - Optimizer: Genetic Algorithm
  - Population: 20 individuals
  - Crossover: 0.4
  - Mutation: 0.3
  - Selection: tournament (size: 2)
- Objective: function value maximization

Island Model



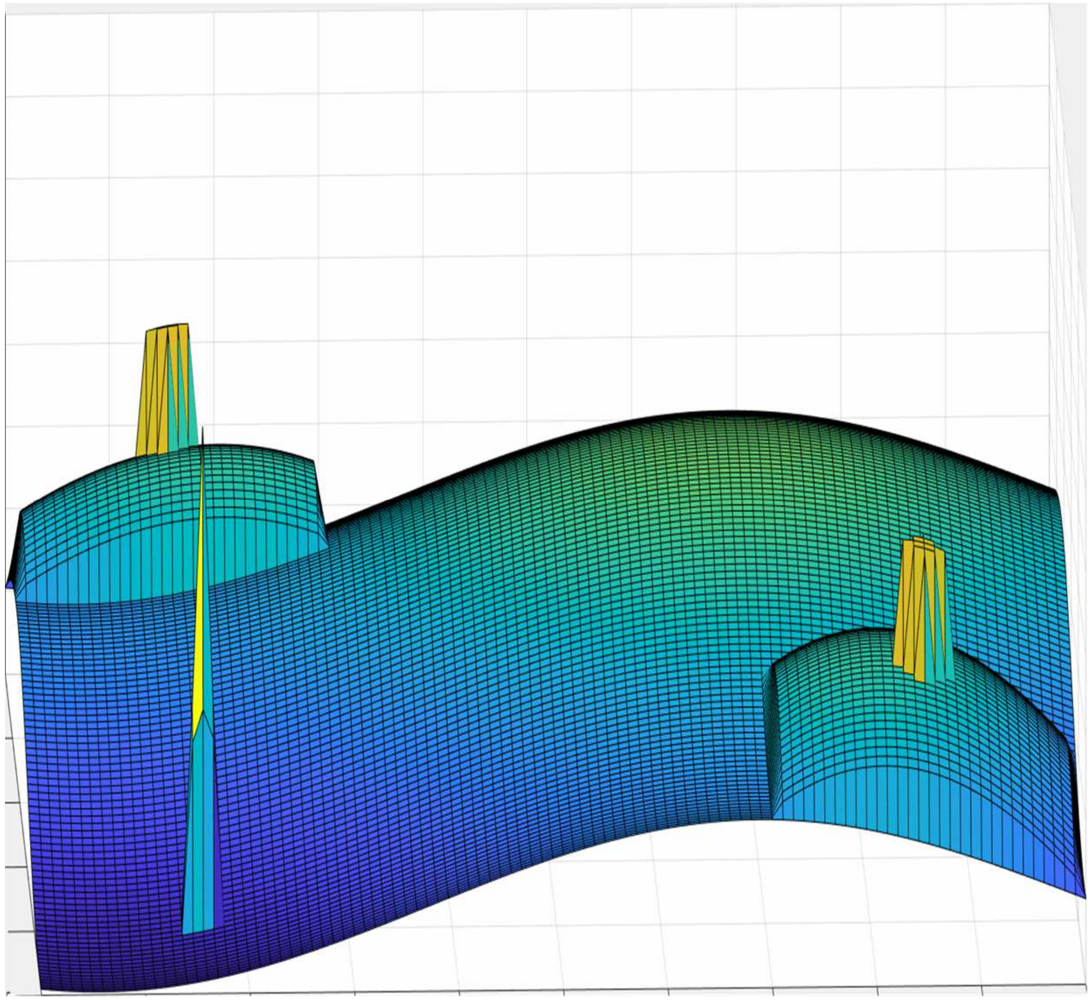
Island Model

- What happened?
  - The optimum was found…
  - …but only one (there are two)…
- Do we need to find both hills?
  - **Yes**
  - Finding two hills will improve the solution quality?

    No
  - **Then what for?**

Island Model

BECAUSE OF THIS!!!



- **There is another hidden hill**
- We didn't expect it
- Its neighbourhood is highly inattractive
- And the optimum is there
- **This new hill is the compilation of coordinates of the two lower hills**
- **A typical situation in real-world problems optimization**

Island Model

- **On searching the hidden hills**
  - Narrow size → the Chance to randomly hit it → negligible
  - Disguting neighbourhood→ **nothing** will converge there
- **What to do?**
  - Find the two lower hills
  - Mix two individuals from the two lower hills
- Global optimum → joins features of two other local optima → typical in real-world

Island Model

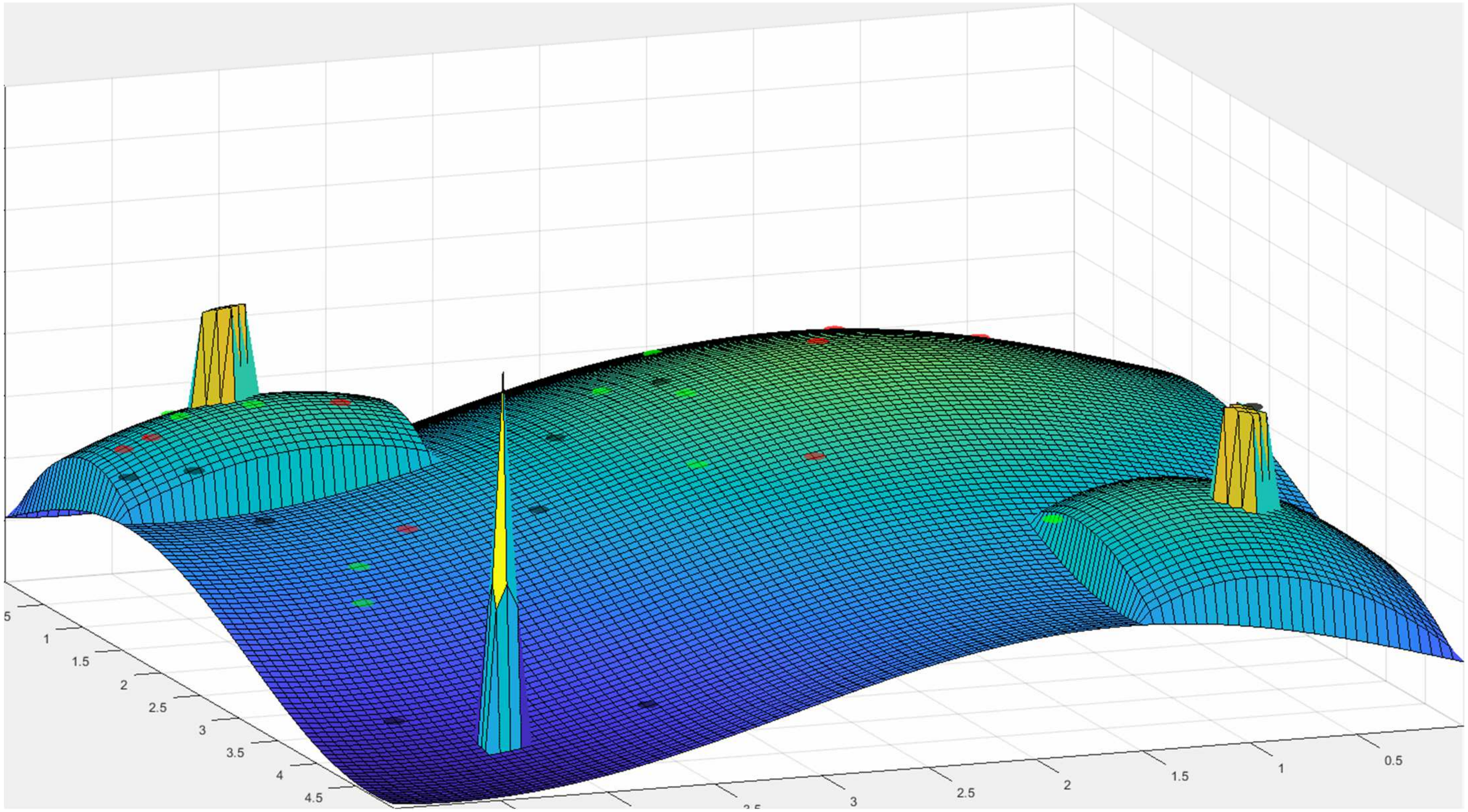
- **How to find the two lower hills**
  - One population → manager only to find one hilll
  - And how about…
  - …using many subpopulations
  - Subpopulations are isolated
- The effect:
  - You may get two subpopulations (each subpopulation finds one of the two lower hills)?
  - Then: we mix individuals from the two separate subpopulations → and we have a Chance to find the hiddem hills

Island Model

- 3 subpopulations
- 10 individuals in each subpopulation
- MIGRATION (information exchange)
  - Every 15 iterations
  - Best individual from each subpopulation is migrated to other subpopulation
- Subpopulation settings – the same as before:
  - Crossover: 0.4
  - Mutation: 0.3

Wrocław University of Science and Technology

Island Model



Wrocław University
of Science and Technology

Island Model

- At the beginning each subpopulation climbs up the ofther hill
- First migration
  - Black and green subpoplation – no change
  - Red subpopulation – converged to the same hill as the green subpoplulaion
- Second migration
  - Black population contain individuals from both hills
  - Mixing between the black subpopulation
- **Applaud!**

Island Model

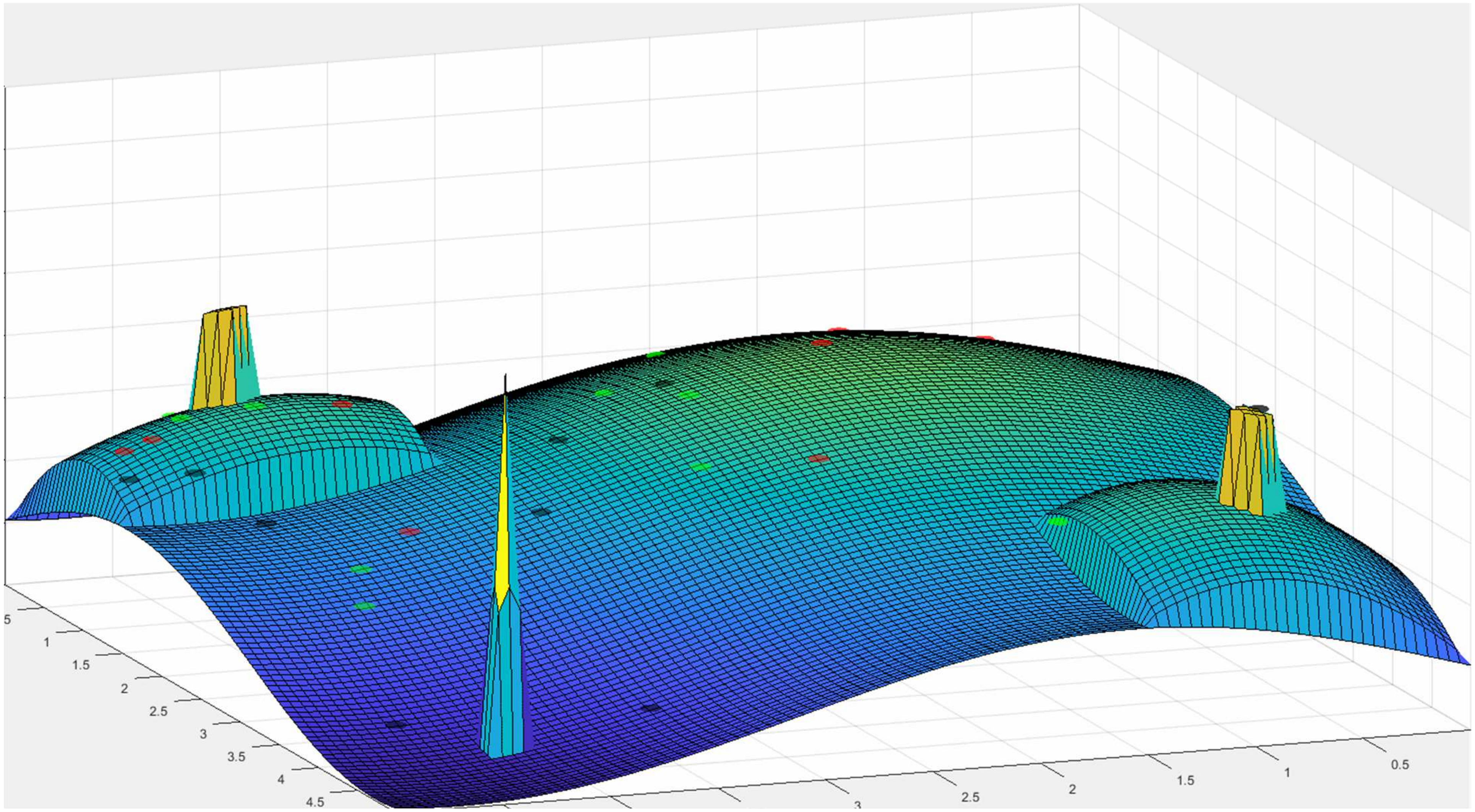
- Island model – disadvantages

Execution for *n* populations -> *n* Times higher computation load

- Additional parameters:
  - Subpopulation number
    - Too low – expensive, but no effect
    - Too high – so expensive that GA may not have enough time to converge, i.e., no effect
  - Migration frequency
    - Too rare – too low information exchange rate = no effect
    - Too often – no diversity preservation effect = no effect

Island Model

Too frequent migration (every 6 iterations instead of every 15)



Island Model

- Initially each subpopulation finds each high hill
- After subsequent migrations

  Diveristy loss (green and black subpopulation are on the same hill)
- At the end:

Red subpopulation is also attracted to the area with green and black subpopulations

- **A VERY EXPENSIVE failure…**

Population-sizing

- **Which population size is the best one?**
  - It depends on the problem
  - It depends on constraints

    Example:
    - Computation budget dependent on *Fitness Function Evaluations* (FFE)
    - High FFE number → we can use large population (we have a lot of time to converge, there is no hurry)
    - Low FFE number → smaller population may be favourable, because we want GA to converge faster

Population-sizing

- **Which population size is the best one? (continued)**
  - It depends on the optimizer

    Example

    - Non-modified GA-based optimizer → it may require a larger population toi preserve diveristy
    - GA-based optimizer with Baldwin effect →
      - Better diversity preservation → lower population size is enough
      - A single GA iteration is expeinsive → we may be unable to afford a large population size

Population-sizing

- SO, which population size is the best one?

The only fair answer: NOBODY KNOWS

- So, how about…
- …adjusting it?

Population-sizing

- We start with the subpopulation of size 2
- 4 iterations for subpopulation of size 2
- Create a new subpopulation of size 4
- For every 4 iterations of the subpopulation of size 2, perform 1 iteration of the population of size 4
- After 4 itertions of the population of size 4, add a new subpopulation of size 8
- For each iteration of the subpopulation of size 8 do:
  - 4 iterations of the subpopulation of size 4
  - 16 iterations of the subpopulation of size 2
- And so on…

Population-sizing

Populacja
pop=2

Utwórz i zrób iterację 1
Iteracja 2
Iteracja 3
Iteracja 4

4 iteracje
4 iteracje
4 iteracje

4 iteracje
4 iteracje
4 iteracje

Populacja
pop=4

Utwórz i zrób iterację 1

Iteracja 2
Iteracja 3
Iteracja 4
Iteracja 5
Iteracja 6
Iteracja 7
Iteracja 8

Populacja
pop=8

Utwórz i zrób iterację 1

Iteracja 2

Population-sizing

- Subpopulationnumber increases quickly
- Correction mechanism
  - **IF** for any subpopulation x there exists a subpopulation y **such that**:
    - Size(*x*) < Size(*y*)
    - AvrFit(*x*) < AvrFit(*y*)

    Where:

    Size(*x*) – subpopulation size

    AvrFit(*x*) – average fitness in a given subpopulation
  - **THEN**:

    Delete subpopulation *x* and all subpopulations smaller than *x*

Population-sizing

- Quite old – 20 years is a lot in this domain
- Adaptive $\rightarrow$ applicable in many cases (problems)
- Simple
- Decreases the parameter number
- Employed in some of the **state-of-the-art. optimizers** (see it next week)

Diversity preservation

- Now you know how to do it
- **IF** you have a diverse poplation
  - →1st GA Sweetspot – **GAINED!**
  - → You have a variety of building blocks
- **But how to exchange building blocks, and not the annoying rest?**
  - **You should remember** → **GA** was **ineffective** after **gene order shuffling** because it was **unable to exchange building blocks**
  - **Thus: we must discover the building blocks**

Comming soon…

Problem structure decomposition

Gene dependency discovery

(some) State-of-the-art GAs