Important Concepts of Modelling

- How to model the problem
  - Representation (Gen string/ Chromosome)
  - Fitness Function
  - Modeling constraints by penalty functions

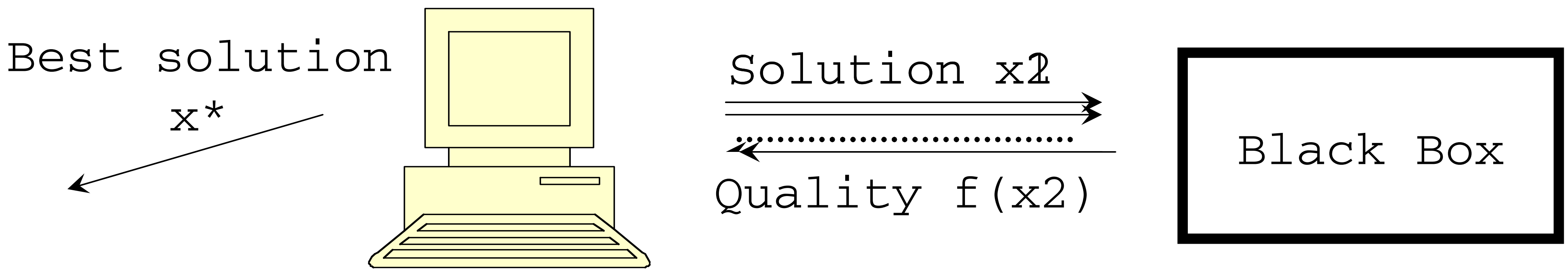
- How to generate new solutions
  - Mutation (one parent)
  - Recombination (two parents)

- What should be kept in memory (= population)

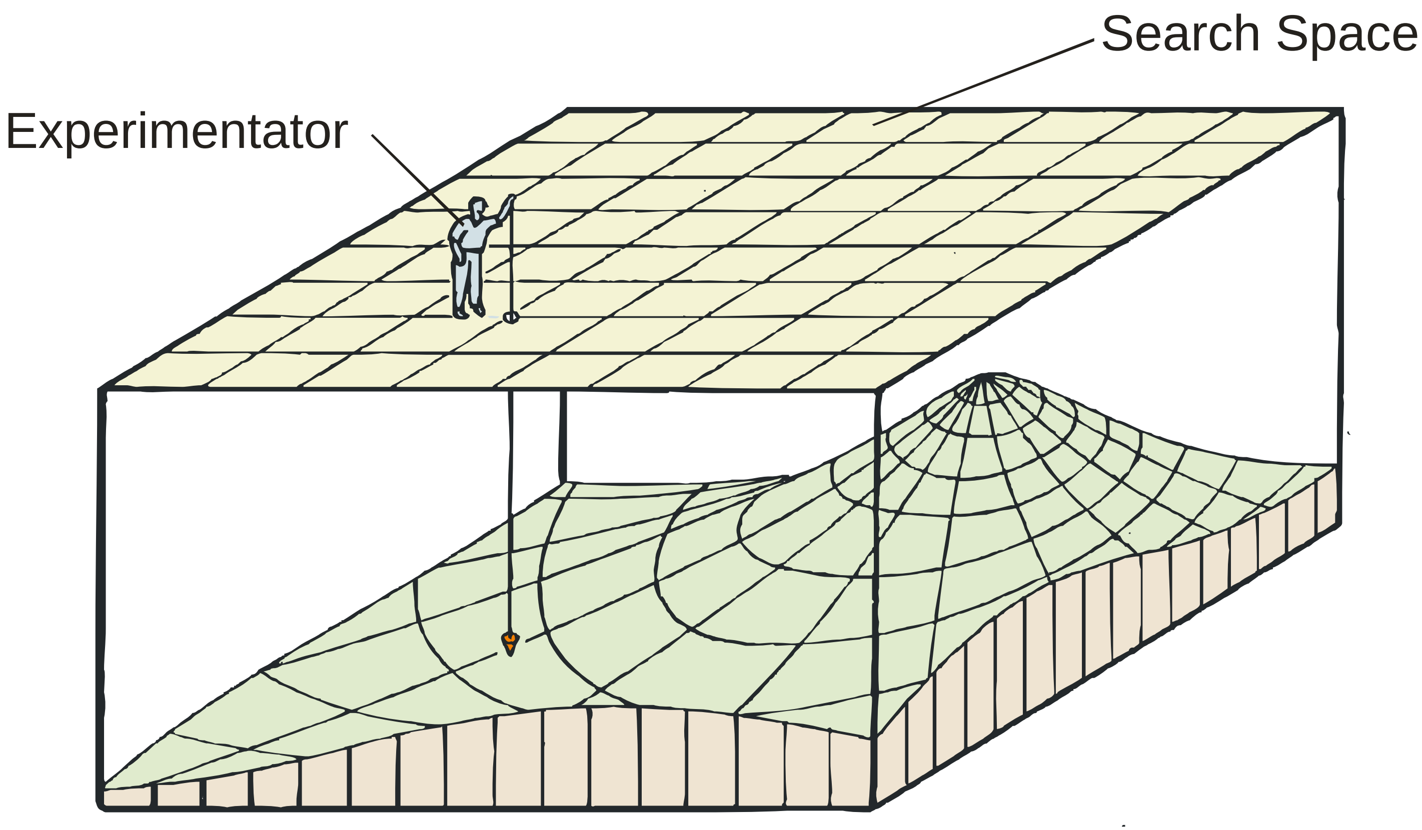
- Performance in comparison to
  - Random Search, Exhaustive Search (Enumeration)
  - Heuristics
    - Greedy Algorithms -> generating a first solution
    - Local Search -> Improving given solutions (Hill Climbing, Gradient Descent)

Black-box optimization

- Optimization can be defined as search for an optimal solution.
- We consider NP-hard problems, which can not be solved in polynomial time (unless P=NP), but which allow a solution to be evaluated in polynomial time (e.g. TSP).



Black-Box: Hillclimbing (cf. Blind-Cow-Game)



Model for a simple optimzation landscape
(Rechenberg)

Main Ideas of Black Box Optimization

- Assumption used
  - solutions “in the neighborhood of” good solutions are probably also good
  - solutions “in the neighborhood” of bad solutions are probably also bad
  - ▶ smooth neighborhood
  - ▶ neighborhood is an important concept
- Successful search requires a good balance
  - exploration and exploitation
  - local and global search
- Optimality is difficult to prove
- Problem-specific knowledge can help a lot

Definition: Global Optimum

a **global optimal solution** (***Minimum***) is a solution x* such that

$$\forall x \in S: f(x) \geq f(x^*)$$

Exercise:

- Define global maximum

Definition: Local Optimum

a **local optimal solution** (***Minimum***) is a solution x* such that

$$\forall x \in N(x): f(x) \geq f(x^*)$$

Exercise:

- Define global maximum

Definition: Neighbourhood

The neighborhood $N(x) \subseteq S$ is a specified set of solutions “near” to x

- $\forall x \in S: x \in N(x)$
- $y \in N(x)$ => y is called neighbor of x

Standard Search Spaces: Binär-String

- Bit string (e.g. $x=b_0b_1b_2\ldots b_{n-1}$ with $b_i\in\{0,1\}$)
  - deterministic neighborhood: ***N(x) = {y | H(x,y) ≤1}***
    with $H(x,y)$ being the Hamming distance
  - probabilistic neighborhood function: flip each bit with probability p, i.e.

$$p_{xy} = p^{H(x,y)}(1-p)^{n-H(x,y)}$$

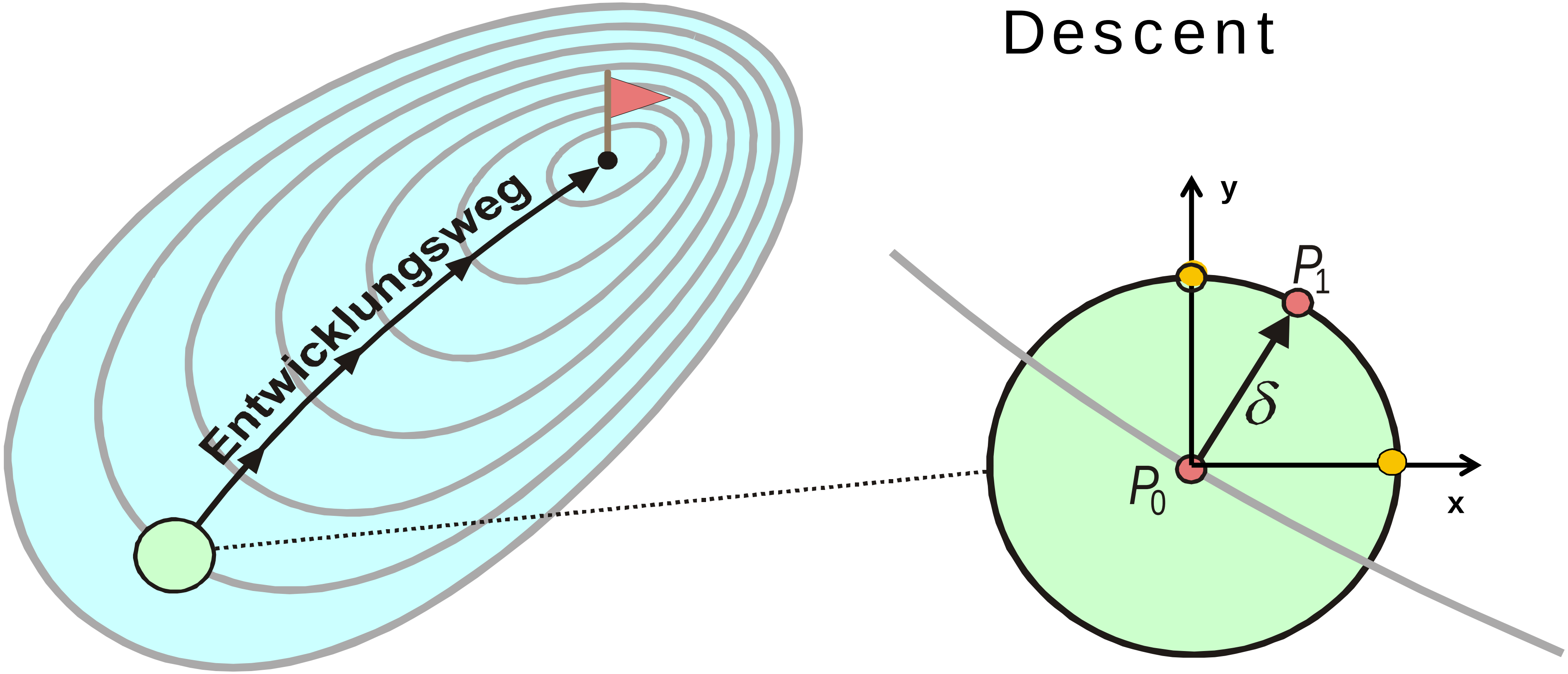
Standard Search Spaces: Real valued vector

- deterministic neighborhood: ***N(x) = {y | d(x,y)*** $\leq \varepsilon$ ***}***
  with $d(x,y)$ being the Euclidean distance.

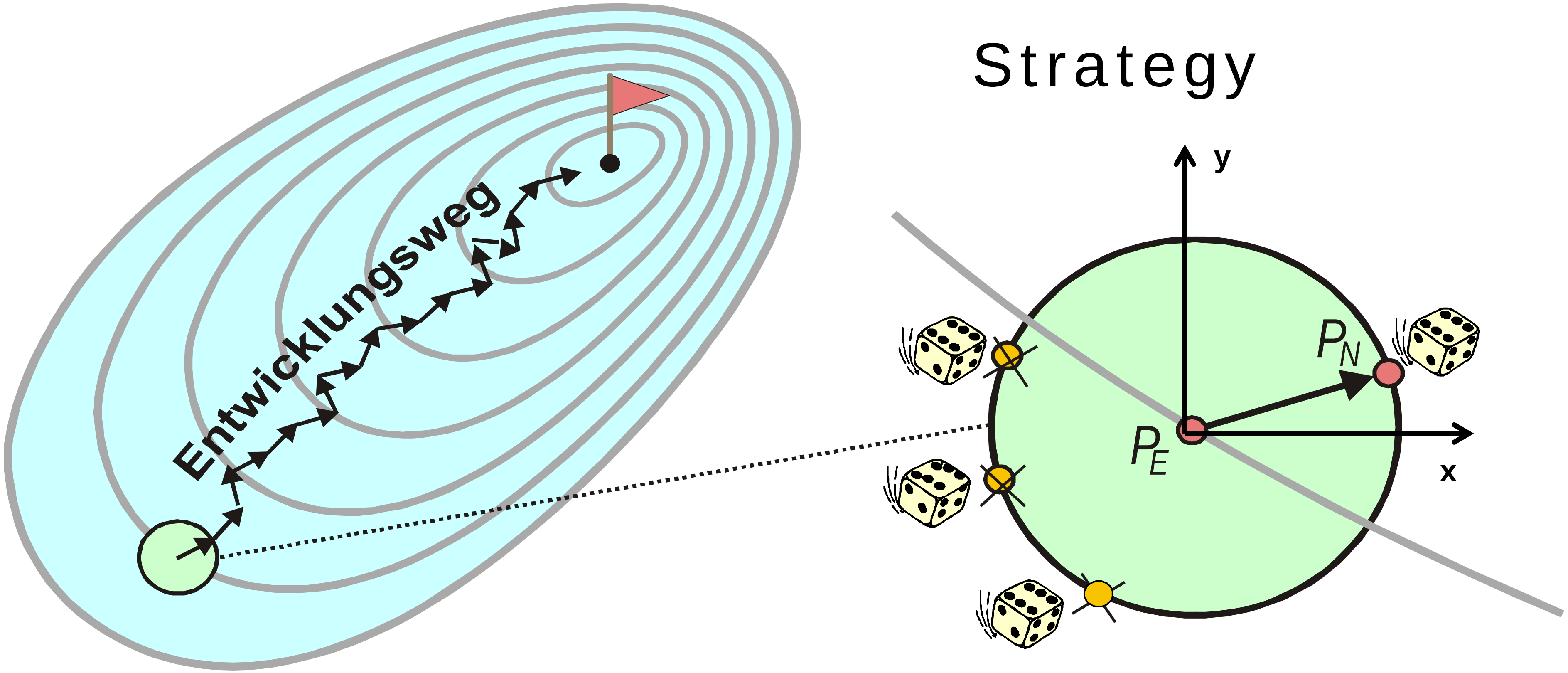
- probabilistic neighborhood:
  - Input: $x = (x_1, x_2, ..., x_n)$ with $x_i \in \Re$
  - Output: $y = (y_1, y_2, ..., y_n)$ with $y_i \in \Re$
    generated in the following way:
    for i=1..n do

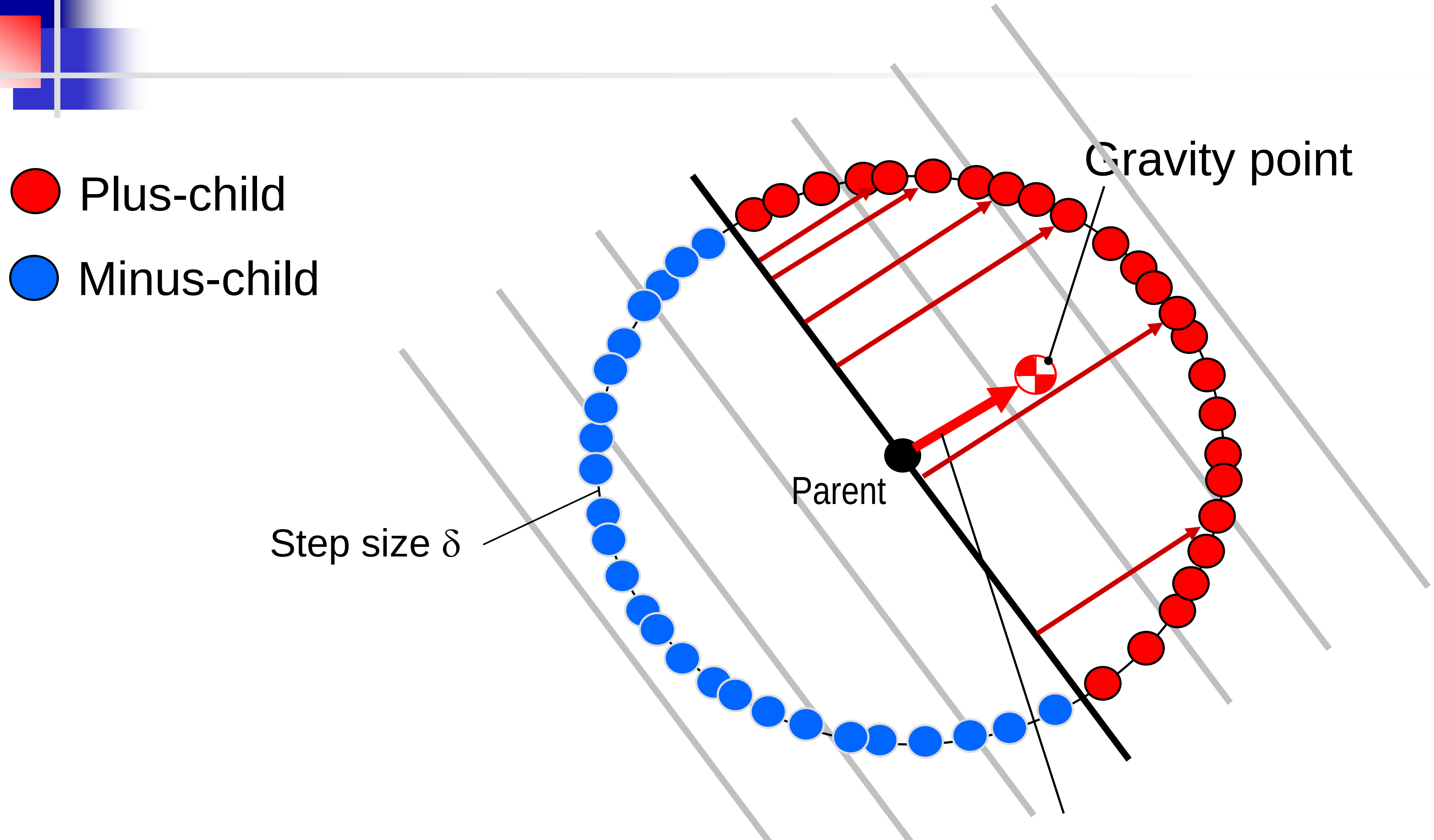
$$y_i = \begin{cases} x_i + v_i,\ v_i \in N(0, \sigma^2) & \text{with probability } p_i \\ x_i & \text{otherwise} \end{cases}$$

Gradient Descent



Evolution Strategy

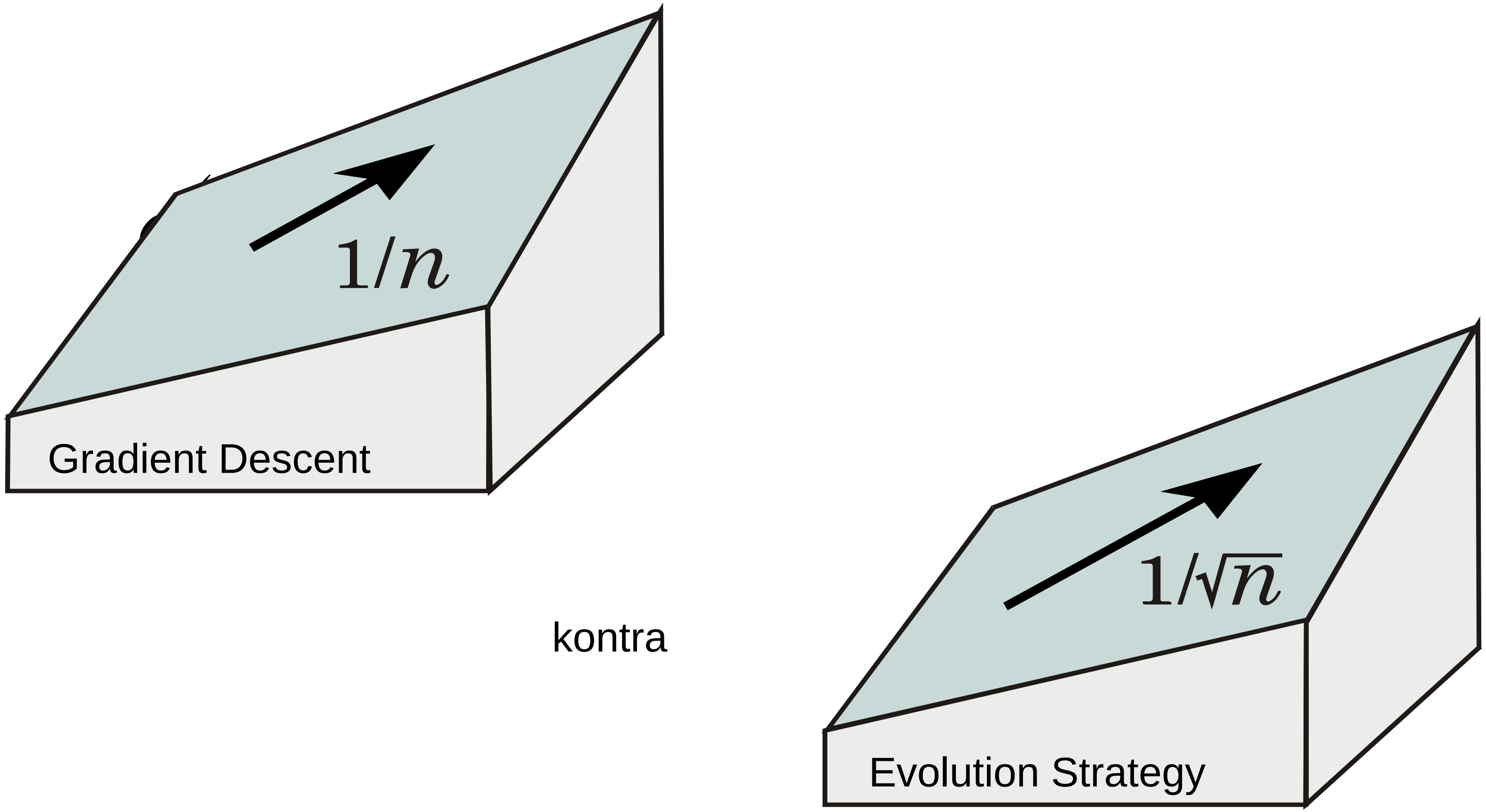




Progress on average $\frac{1}{2}\times\frac{2}{\pi}\times\delta = \frac{\delta}{\pi}$

50% of children are worse!

Comparison of progress on average

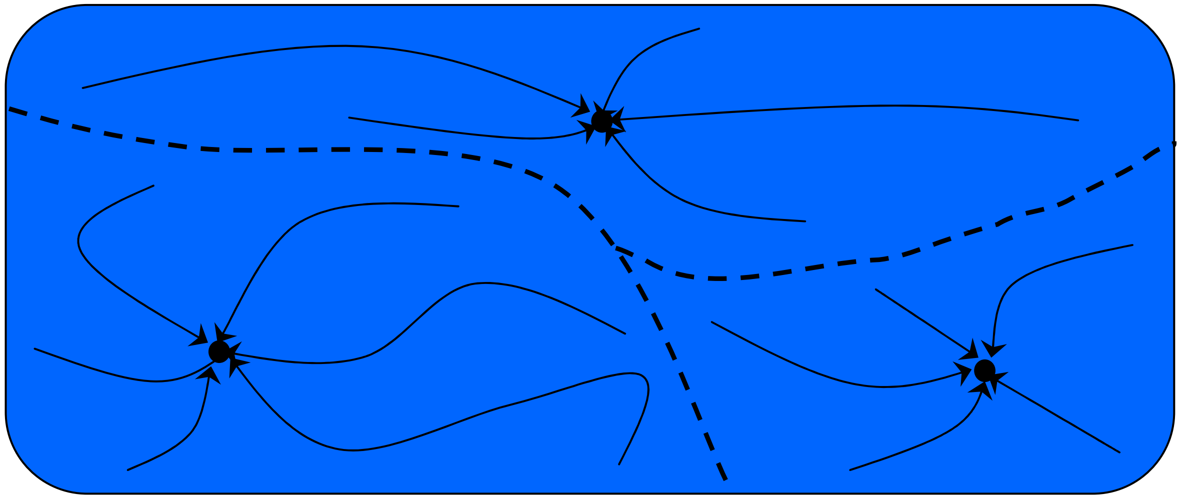


Fitness Landscapes

- Often, it is intuitive to think of search as a walk in a "fitness landscape".
- The landscape concept only makes sense in the context of an associated neighborhood structure, which depends on the representation and the search operators

Definition: Let $t{:}S\text{->}S$ be a transformation function. A state $s \in S$ is called **attractor** if $t(s)=s$. The set of states B that are mapped to an attractor s by repeated applications of t is called **basin of attraction**.

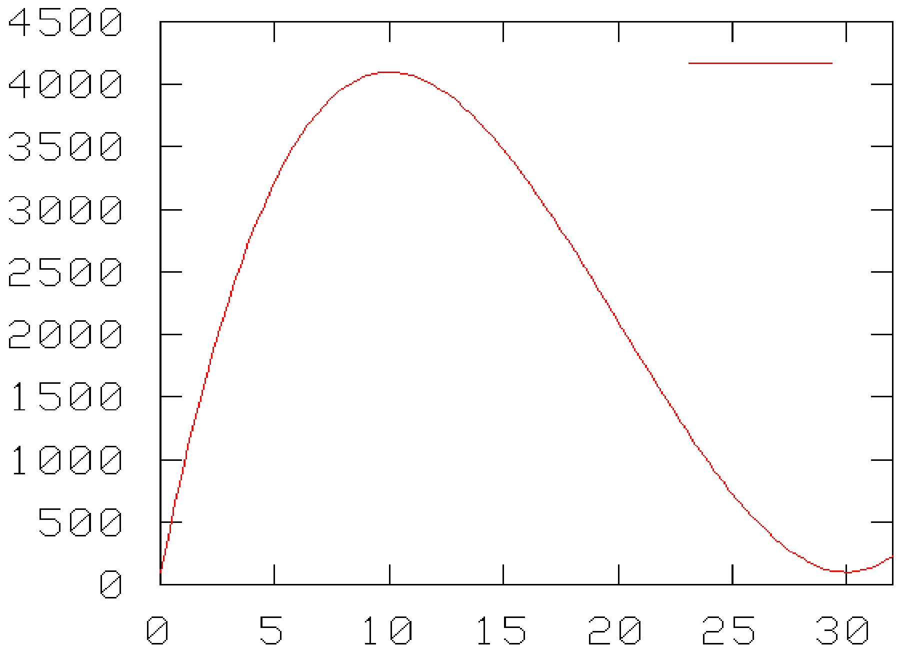
Search space with three attractors and corresponding basins of attraction



- A simple example:

$$\max\ f(x) = x^3 - 60x^2 + 900x + 100$$

$x \in \{0,1,\ldots,31\}$



- x shall be encoded
  as bit-string of length 5
- standard deterministic 1-bit neighborhood
- two transformation operators: steepest ascent (best neighbor) or first ascent (first neighbor better than current solution) hill-climbing

Simple binary coding, single bit flip

steepest ascent <-> next ascent (back to front)

Local Optima			
01010	01100	00111	10000
Basin of attraction			
00000 00001 00010 00011 00101 01000 01001 01010 01011 01101 01110 01111 10101 11000 11001 11010 11011 11101 11110 11111	00100 01100 11100	00110 00111 10110 10111	10000 10001 10010 10011 10100

Local Optima			
01010	01100	00111	10000
Basin of attraction			
01000 01001 01010 01011	01100 01101 01110 01111	00000 00001 00010 00011 00100 00101 00110 00111	10000 10001 10010 10011 10100 10101 10110 10111 11000 11001 11010 11011 11100 11101 11110 11111

Conclusion: Representation / Encoding

- Search space and search operators together define the search “landscape”
  - *Exists* $x \in S$ such that the decoded solution $d(x)$ is a global optimum
  - Every solution should be “reachable” from every other solution (i.e. by mutation)

- A search landscape should be smooth with only few traps
  - **Similar solutions should have similar fitness**
  - **A move in the neighborhood should preserve important characteristics (“locality”)**

- Best Neighbour achieve better local optima than Next Ascent
  - All neighbors too inefficient
  - set of neighbors ideal balance quality of local optimum and computing time

- Evolutionstrategy compares favorable to Gradient Descent
  - Progress
    - $O(1/n^{1/2})$ evolution strategy
    - $O(1/n)$ gradient descent
  - No gradient information necessary

Iterated local search

```
procedure iterated hill-climber
begin
   t ← 0; initialize best
   repeat
      local ← FALSE
      select a current solution x_c at random
      repeat
         select  x_n = argmin_{y ∈ N(x_c)\x_c} (f(y))
         if f(x_n) ≤ f(x_c)
            then x_c ← x_n
            else local ← TRUE
      until local
      t ← t+1
      if f(x_c) ≤ f(best)
         then best ← x_c
   until termination-condition
end
```

$$x_n = \underset{y \in N(x_c) \backslash x_c}{\arg\min} \left(f(y) \right)$$

Simple local search (hill-climber)

```
procedure hill-climber
begin
   local ← FALSE
   select a current solution x_c at random
   repeat
        select   x_n = argmin_{y ∈ N(x_c)\x_c} (f(y))
        if f(x_n) ≤ f(x_c)
              then x_c ← x_n
              else local ← TRUE
   until local
end
```

$$x_n = \arg\min_{y \in N(x_c) \setminus x_c} (f(y))$$

Problem: How can we avoid to get stuck in local minima?