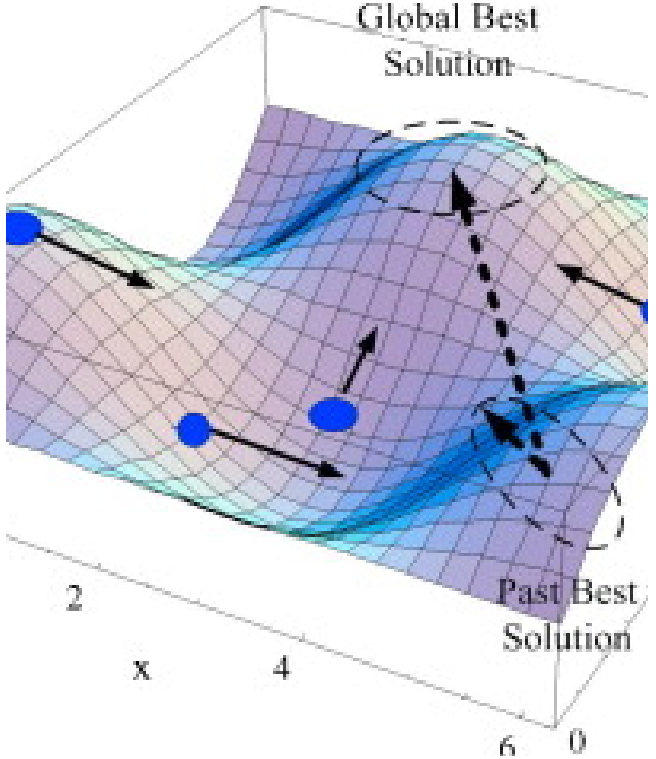
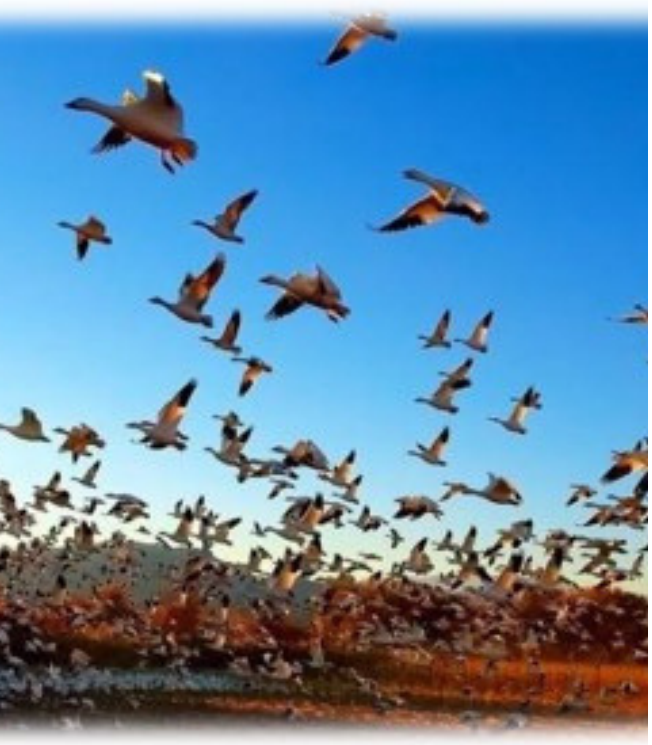


THE PARTICLE SWARM

OPTIMIZATION PROBLEM



CONTENTS

- Origins
- Concept
- PSO Algorithm

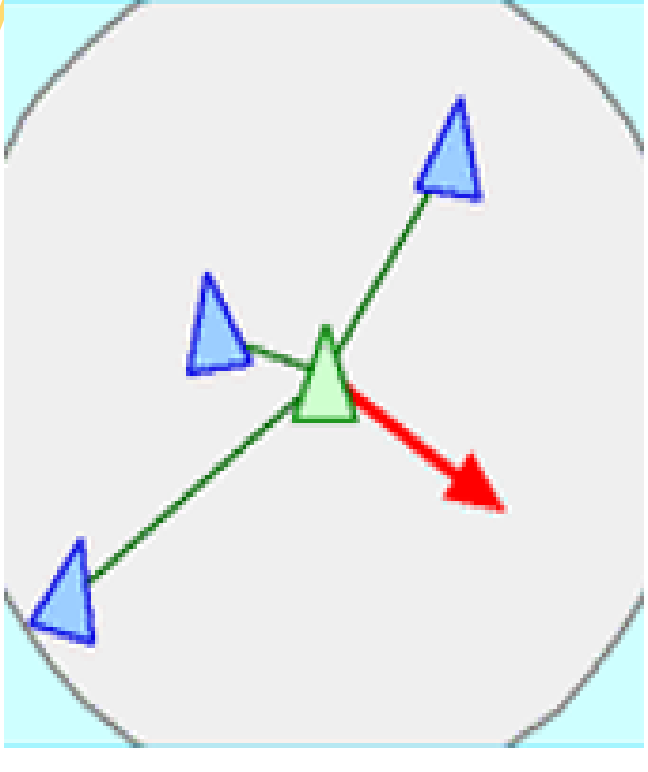
ORIGINS

- Inspired from the nature social behavior and dynamic movements with communications of insects, birds and fish.
- Proposed by James Kennedy & Russell Eberhart (1995)
- Combines self-experiences with social experiences

IN 1986, CRAIG REYNOLDS DESCRIBED THIS PROCESS IN 3 SIMPLE BEHAVIORS:

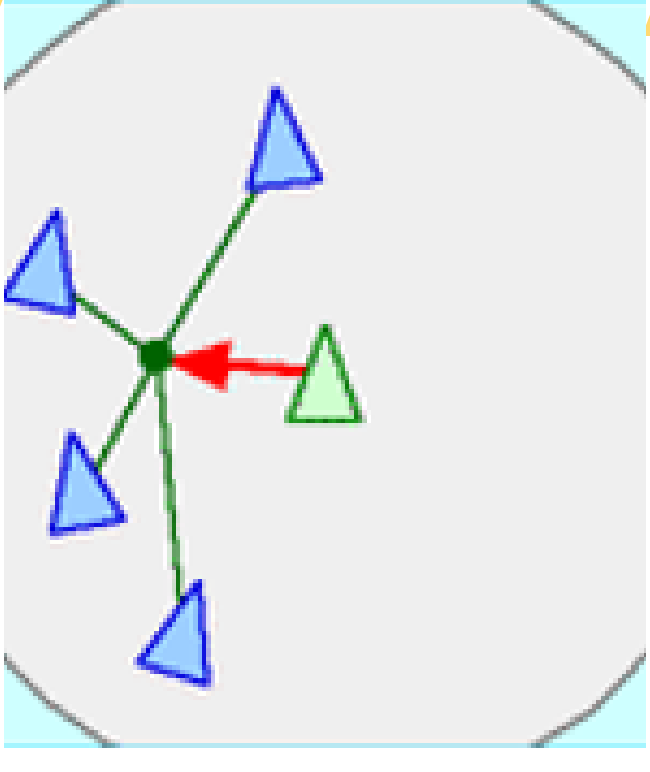
Separation

avoid crowding
local flockmates



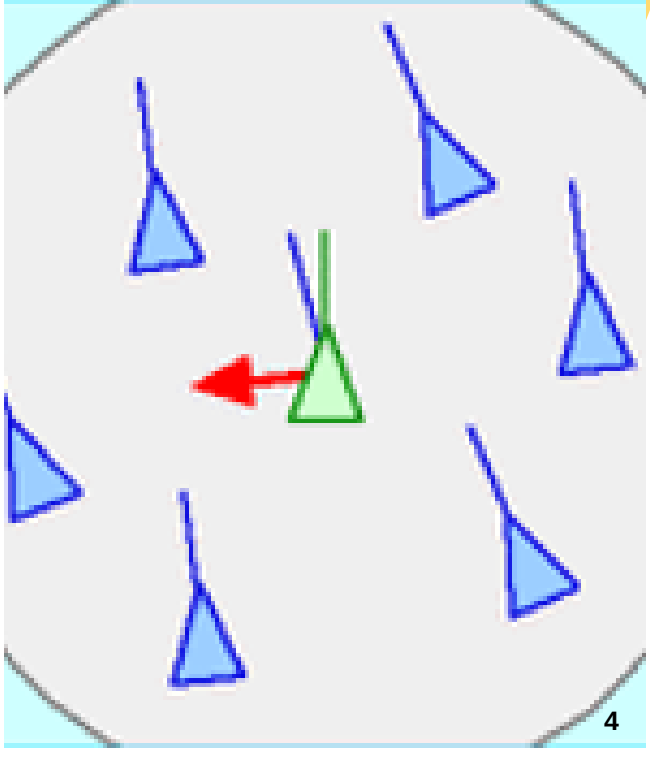
Alignment

move towards the
average heading
of local flockmates



Cohesion

move toward the
average position
of local flockmates



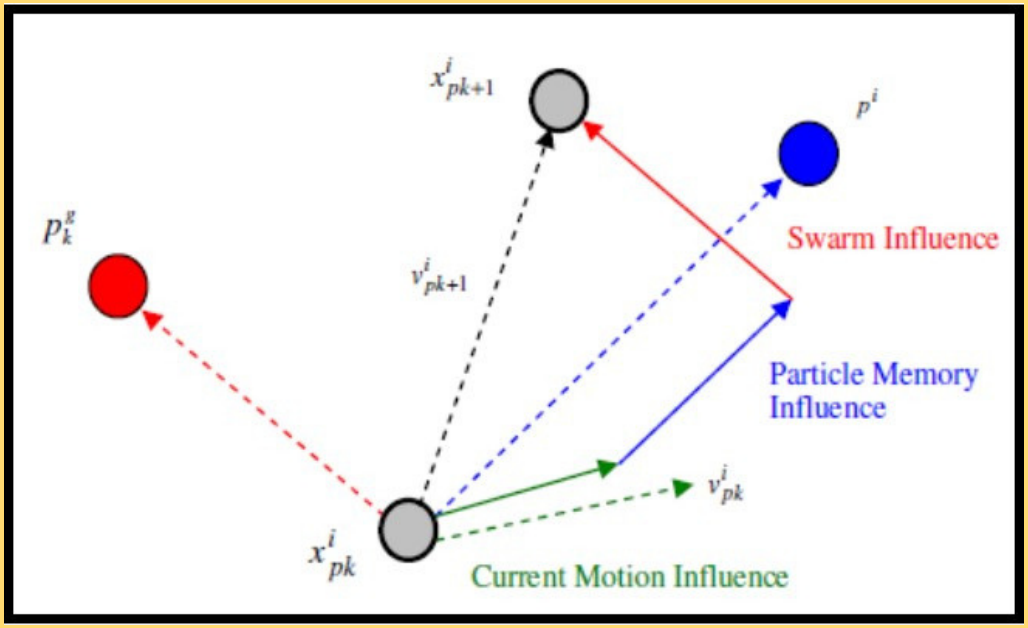
CONCEPT

- Uses a number of agents (particles) that constitute a swarm moving around in the search space looking for the best solution
- Each particle in search space adjusts its “flying” according to its own flying experience as well as the flying experience of other particles

CONCEPT

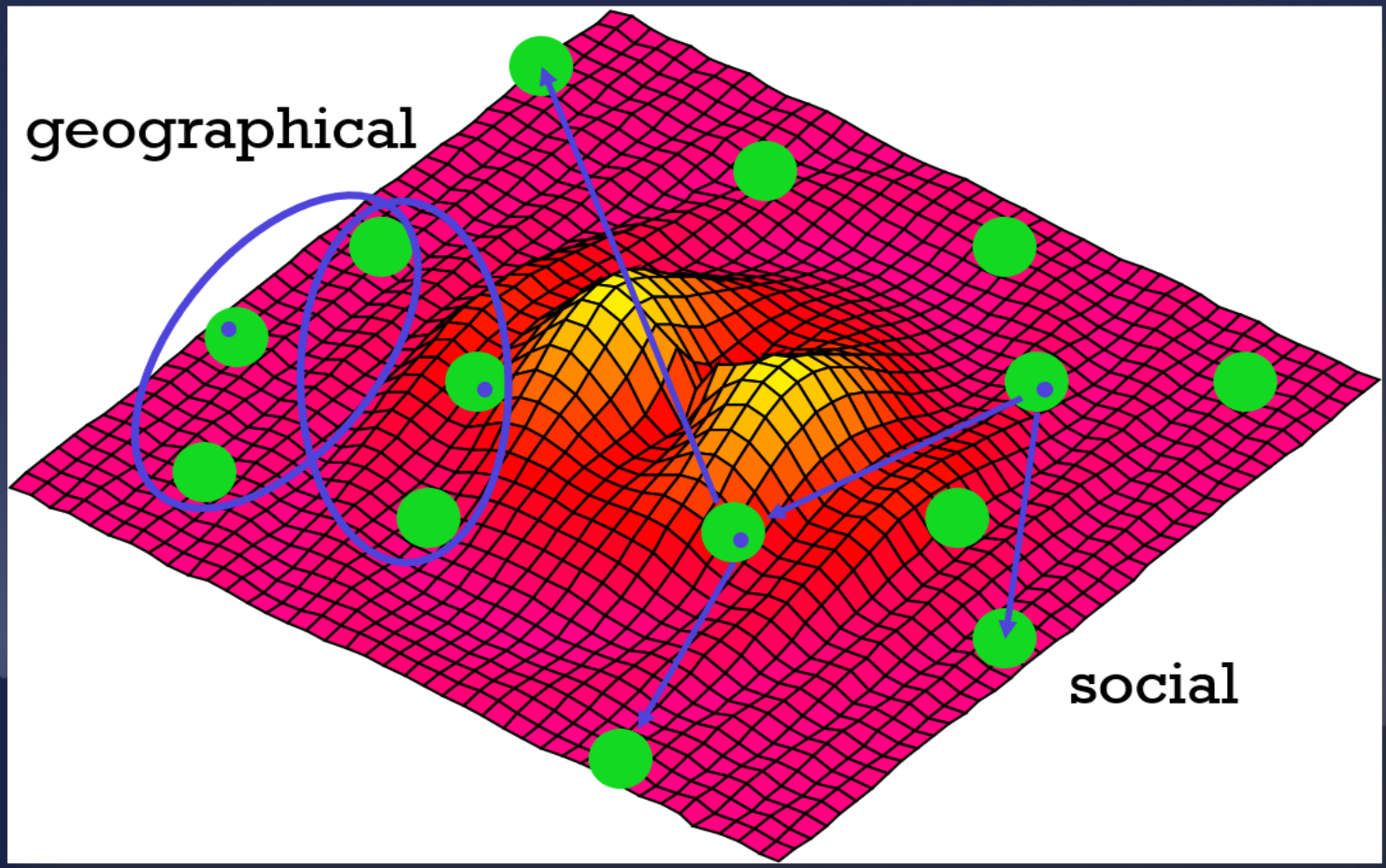
- Search area – Possible solutions
- Movement towards a promising area to get the global optimum
- Each particle keeps track:its best solution, personal best, pbest and the best value of any particle, global best, gbest

CONCEPT

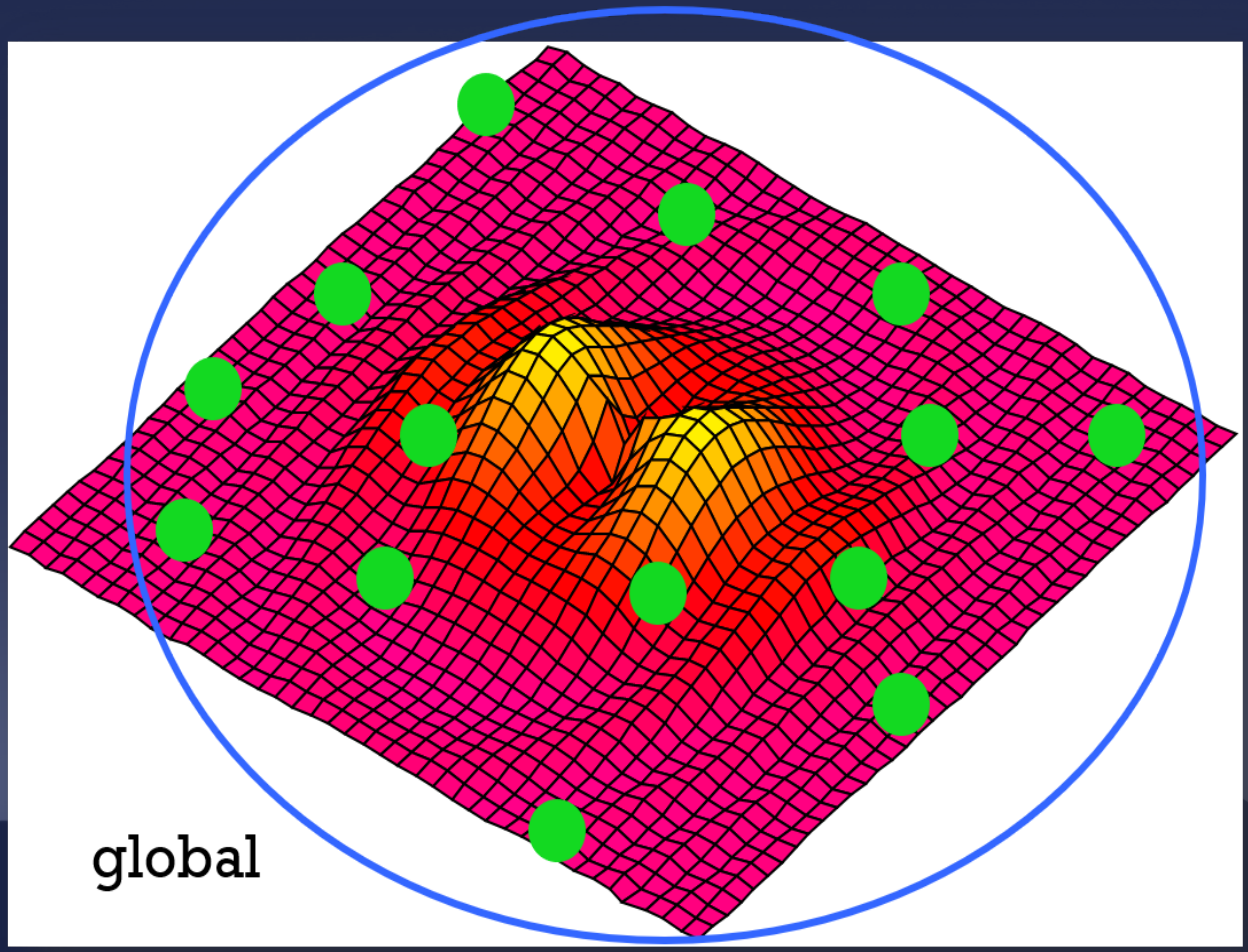


- Each particle adjusts its travelling speed dynamically corresponding to the flying experiences of itself and its colleagues.Each particle modifies its position according to:
- its current position
- its current velocity
- the distance between its current position and pbest
- the distance between its current position and gbest

ALGORITHM-NEIGHBORHOOD



ALGORITHM-NEIGHBORHOOD



ALGORITHM PARAMETERS

A : Population of agents

pi : Position of agent ai in the solution space

f : Objective function

vi : Velocity of agent's ai

V(ai) : Neighborhood of agent ai (fixed)

The neighborhood concept in PSO is not the same as the one used in other meta-heuristics search, since in PSO each particle's neighborhood never changes (is fixed)

ALGORITHM

```
[x*] = PSO()
P = Particle_Initialization();
For i=1 to it_max
  For each particle p in P do
     fp = f(p);
     If fp is better than f(pBest)
           pBest = p;
     end
  end
gBest = best p in P;
  For each particle p in P do
      v = v + c1*rand*(pBest - p) + c2*rand*(gBest - p);
      p = p + v;
     end
end
```

ALGORITHM

Particle update rule

$$p = p + v$$

with

$$v = v + c1 * rand * (pBest - p) + c2 * rand * (gBest - p)$$

where

p: particle's position

v: path direction

c1: weight of local information

c2: weight of global information

pBest: best position of the particle

gBest: best position of the swarm

rand: random variable

ALGORITHM PARAMETERS

- Number of particles usually between 10 and 50
- C1 is the importance of personal best value
- C2 is the importance of neighborhood best value
- Usually C1 + C2 = 4 (empirically chosen value)
- If velocity is too low → algorithm too slow
- If velocity is too high → algorithm too unstable

ALGORITHM

- Create a ‘population’ of agents (particles) uniformly distributed over X
- Evaluate each particle’s position according to the objective function
- If a particle’s current position is better than its previous best position, update it
- Determine the best particle (according to the particle’s previous best positions)

ALGORITHM

- Update particles' velocities:

$$\mathbf{v}_i^{t+1} = \underbrace{\mathbf{v}_i^t}_{\text{inertia}} + \underbrace{c_1 \mathbf{U}_1^t (\mathbf{pb}_i^t - \mathbf{p}_i^t)}_{\text{personal influence}} + \underbrace{c_2 \mathbf{U}_2^t (\mathbf{gb}^t - \mathbf{p}_i^t)}_{\text{social influence}}$$

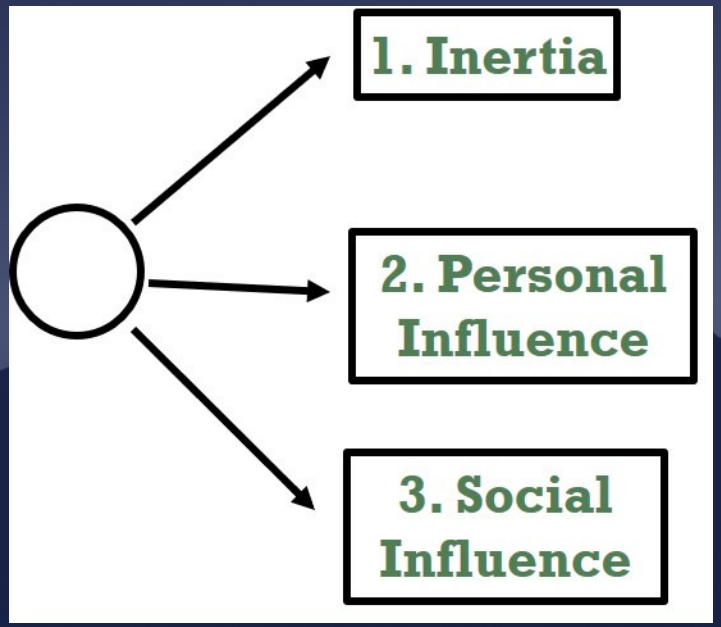
- Move particles to their new positions:

$$\mathbf{p}_i^{t+1} = \mathbf{p}_i^t + \mathbf{v}_i^{t+1}$$

- Go to step 2 until stopping criteria are satisfied.

ALGORITHM

$$\mathbf{v}_i^{t+1} = \underbrace{\mathbf{v}_i^t}_{\text{inertia}} + \underbrace{c_1 \mathbf{U}_1^t (\mathbf{pb}_i^t - \mathbf{p}_i^t)}_{\text{personal influence}} + \underbrace{c_2 \mathbf{U}_2^t (\mathbf{gb}^t - \mathbf{p}_i^t)}_{\text{social influence}}$$



Makes the particle move in the same direction and with the same velocity

Improves the individual,Makes the particle return to a previous position, better than the current,Conservative

Makes the particle follow the best neighbors direction

ALGORITHM

- Intensification: explores the previous solutions, finds the best solution of a given region
- Diversification: searches new solutions, finds the regions with potentially the best solutions

$$\mathbf{v}_i^{t+1} = \underbrace{\mathbf{v}_i^t}_{\text{Diversification}} + \underbrace{\mathbf{c}_1\mathbf{U}_1^t(\mathbf{pb}_i^t - \mathbf{p}_i^t) + \mathbf{c}_2\mathbf{U}_2^t(\mathbf{gb}^t - \mathbf{p}_i^t)}_{\text{Intensification}}$$

ALGORITHM CHARACTERISTICS

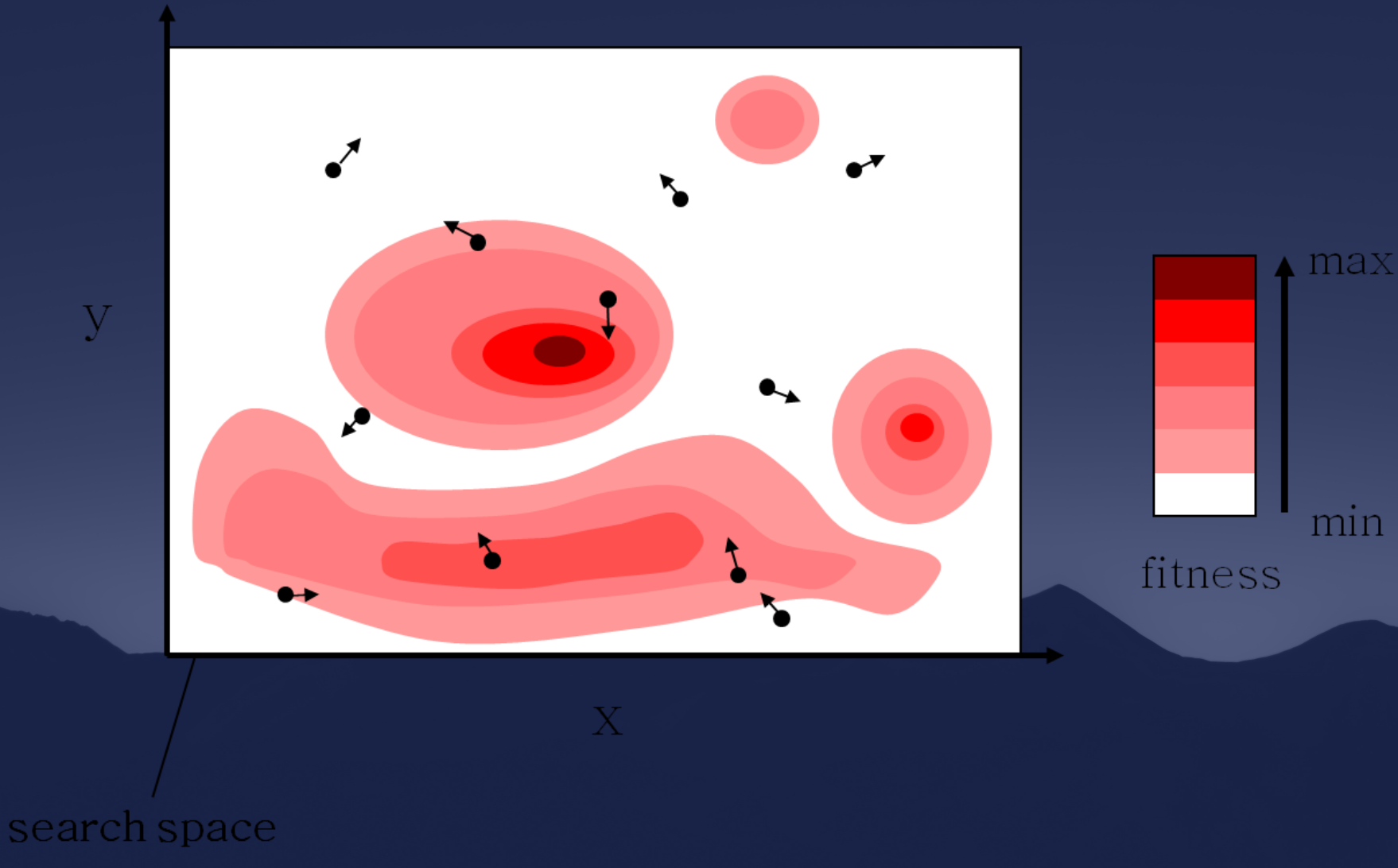
ADVANTAGES

-
- Insensitive to scaling of design variables
- Simple implementation
- Easily parallelized for concurrent processing
- Derivative free
- Very few algorithm parameters
- Very efficient global search algorithm

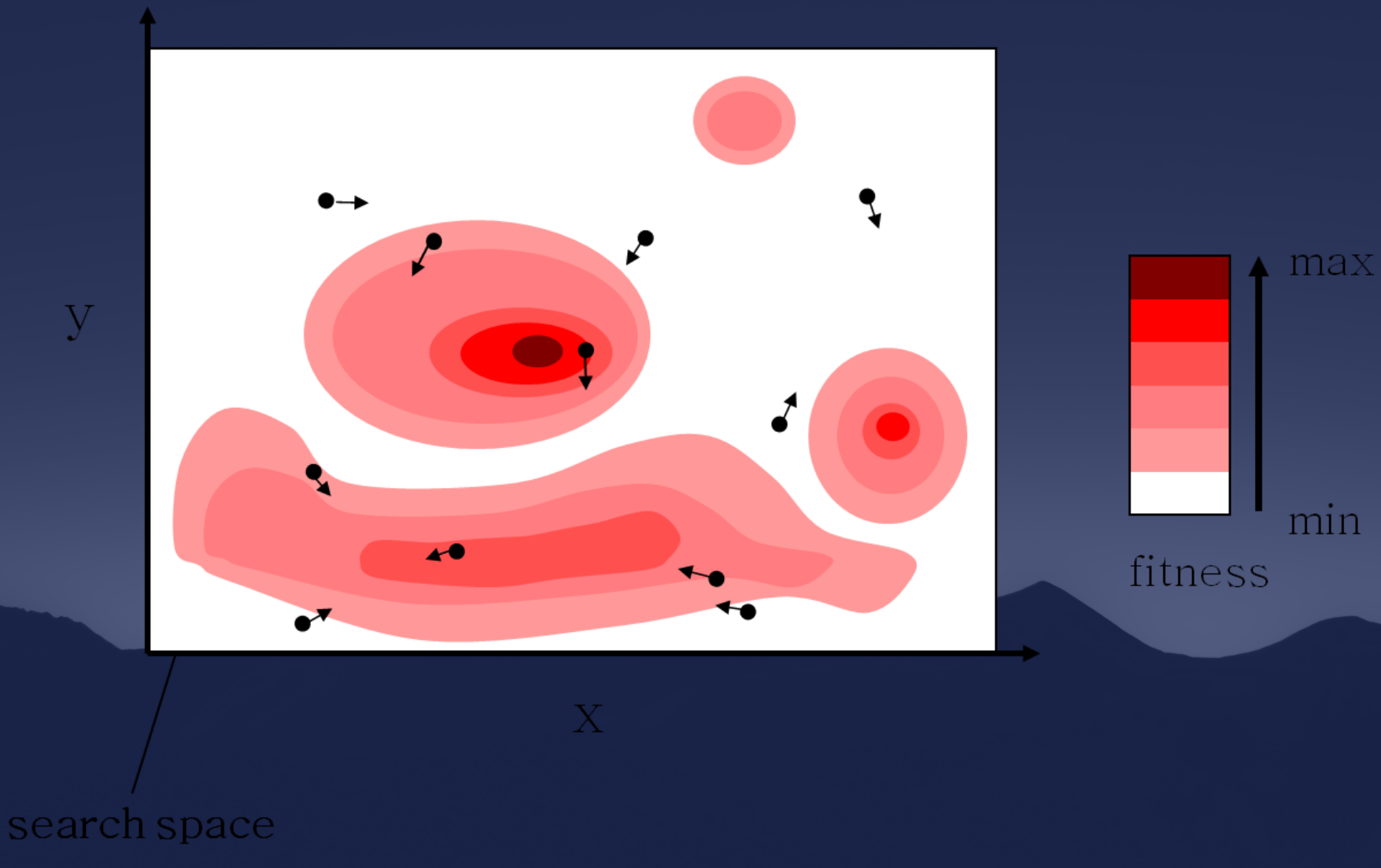
DISADVANTAGES

- Tendency to a fast and premature convergence in mid optimum points
- Slow convergence in refined search stage (weak local search ability)

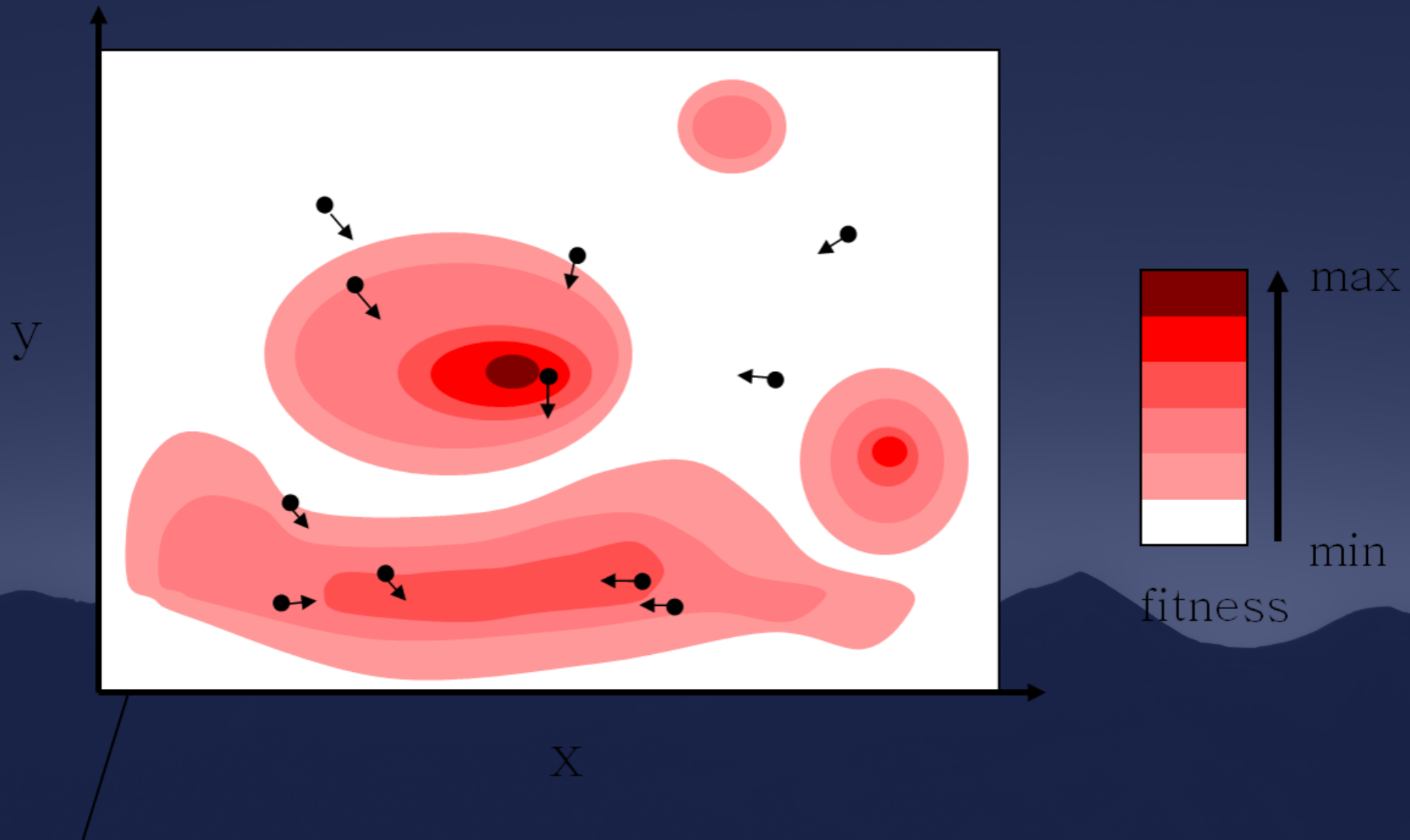
EXAMPLE



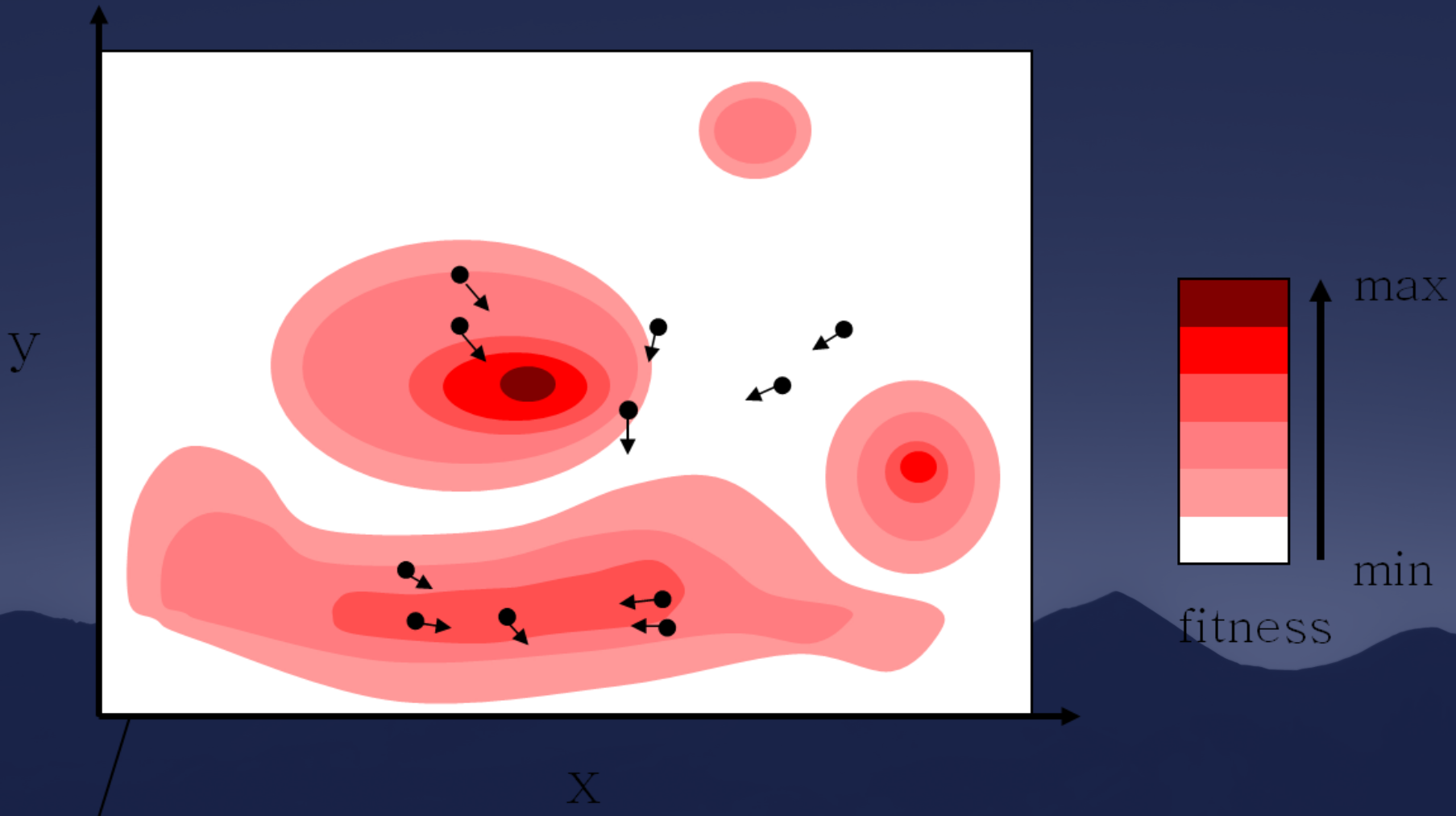
EXAMPLE



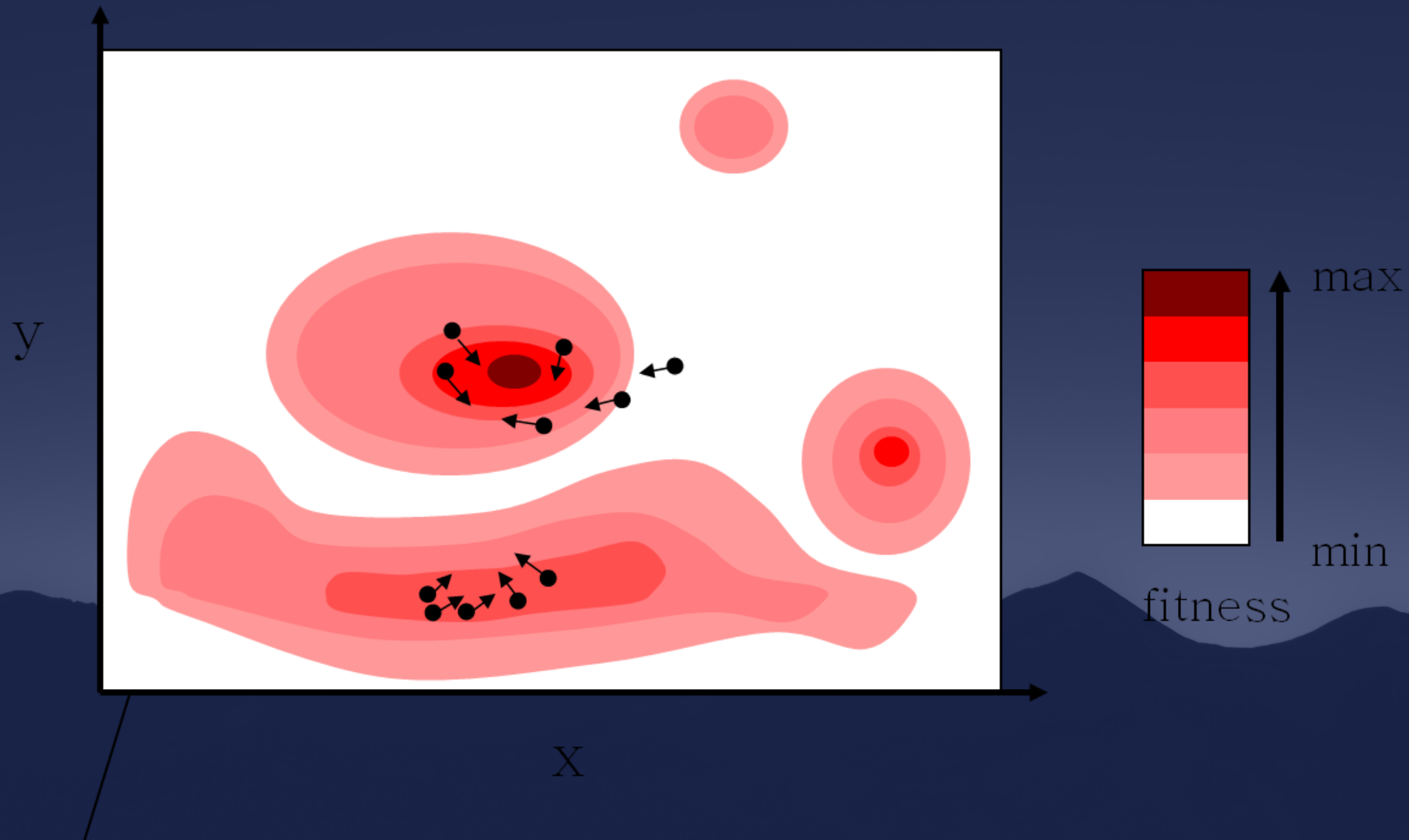
EXAMPLE



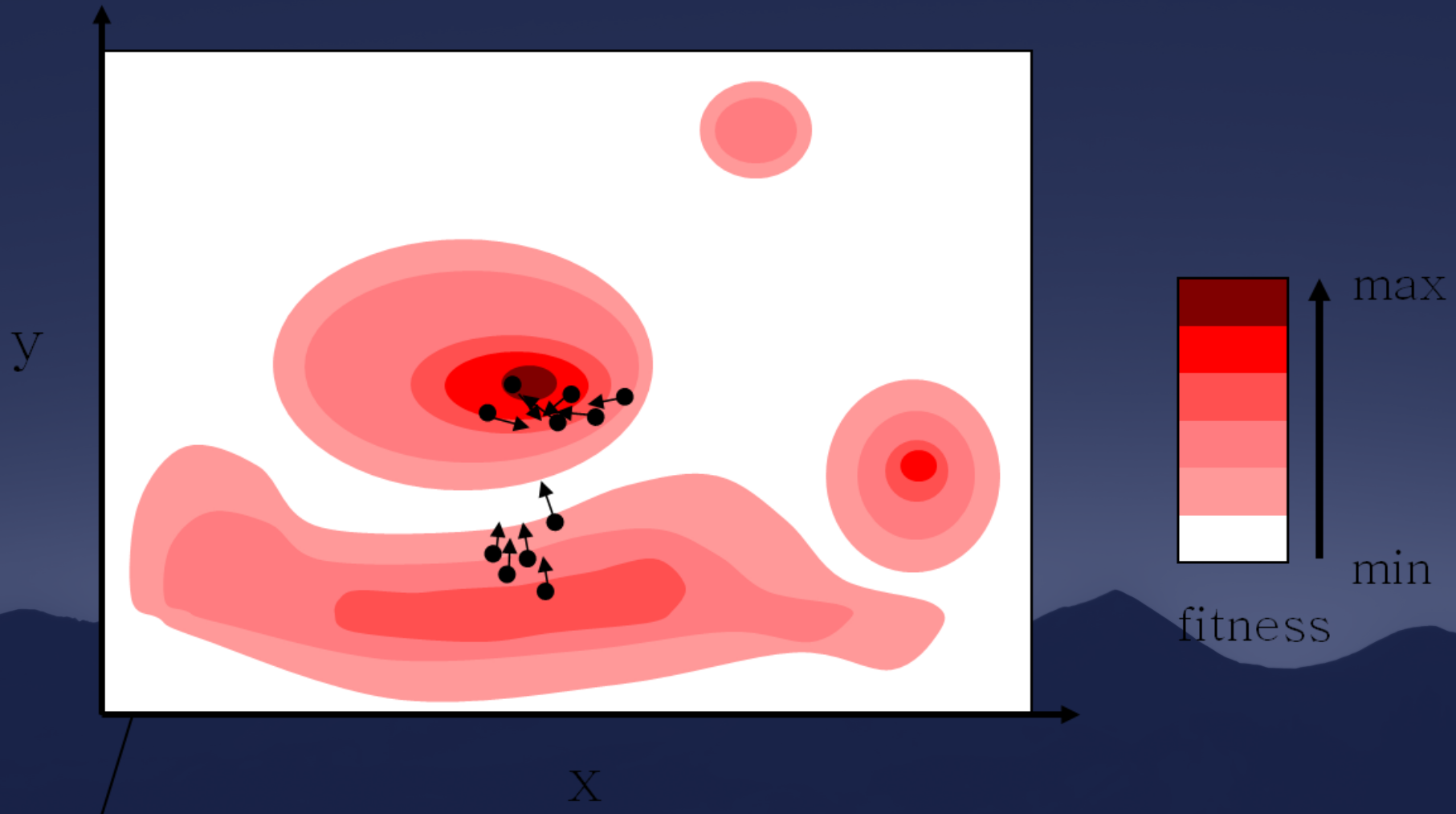
EXAMPLE



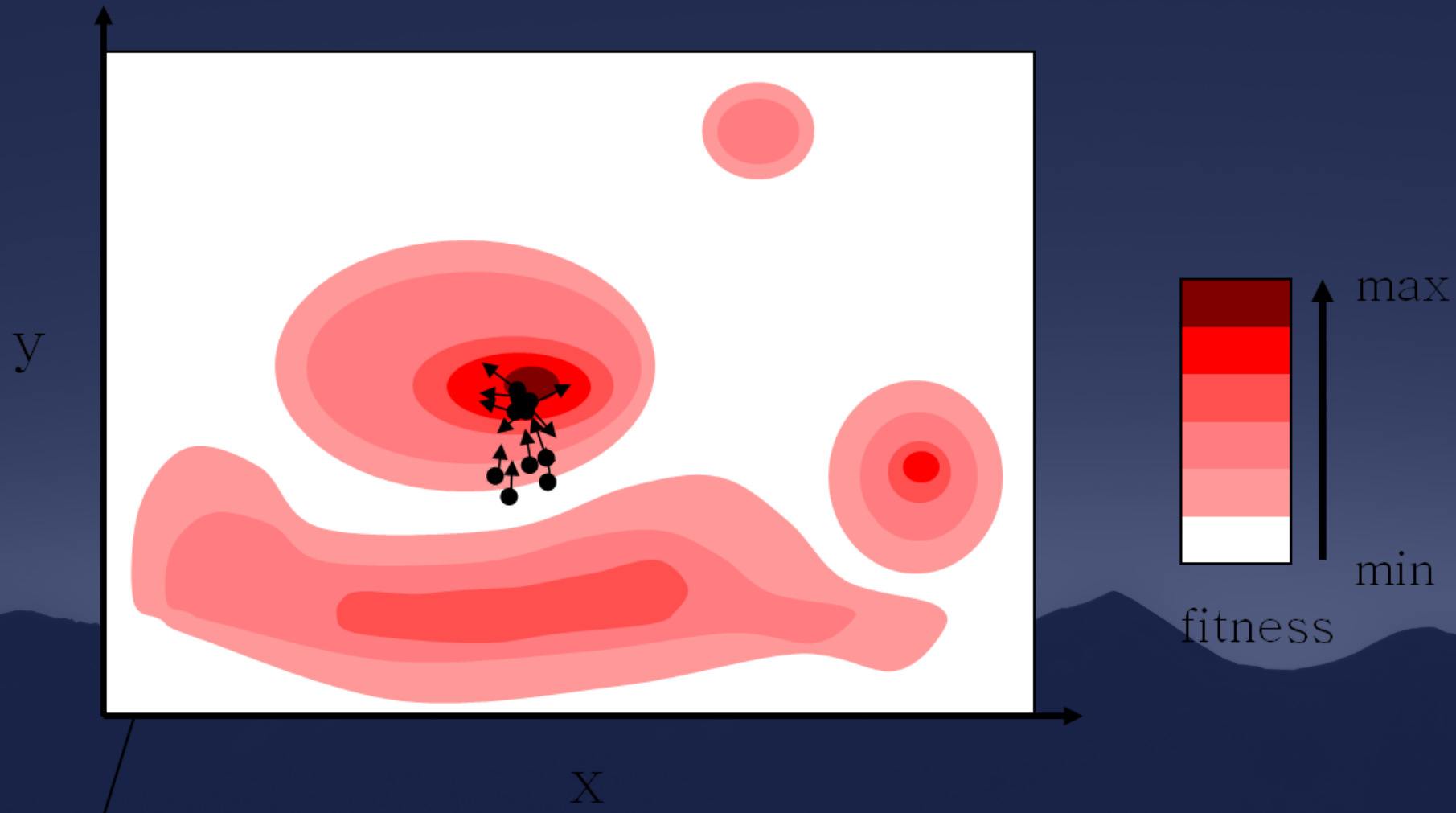
EXAMPLE



EXAMPLE



EXAMPLE



EXAMPLE