CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering
Department of Cybernetics

A0M33EOA:
Differential Evolution.
Other Types of Metaheuristics.

Petr Pošík

Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Cybernetics

Introduction

Contents

Differential evolution (DE):

- Another successful heuristic for optimization in R^D.

Swarm intelligence:

- Particle Swarm Optimization (PSO, optimization in R^D).
- Ant Colony Optimization (ACO, optimization on graphs).

Differential Evolution

Differential Evolution

Developed by Storn and Price [SP97].

- Simple algorithm, easy to implement.
- Unusual breeding pipeline.

Algorithm 1: DE Breeding Pipeline

Input: Population X with fitness in f.
Output: Offspring population X_N.

begin
    $X_N \leftarrow \emptyset$
    foreach $\boldsymbol{x} \in X$ **do**
        $(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3) \leftarrow \texttt{Select}(X, f, \boldsymbol{x})$
        $\boldsymbol{u} \leftarrow \texttt{Mutate}(\boldsymbol{x}, \boldsymbol{x}_1, \boldsymbol{x}_2)$
        $\boldsymbol{x}_N \leftarrow \texttt{Recombine}(\boldsymbol{u}, \boldsymbol{x}_3)$
        $X_N \leftarrow X_N \cup \texttt{BetterOf}(\boldsymbol{x}, \boldsymbol{x}_N)$
    return X_N

- Vectors $\boldsymbol{x}, \boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$ shall all be different, $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$ chosen uniformly.
- For each population member $\boldsymbol{x}$, an offspring $\boldsymbol{x}_N$ is created.
- x_N replaces $\boldsymbol{x}$ in population if it is better.

[SP97] Rainer Storn and Kenneth Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, December 1997.

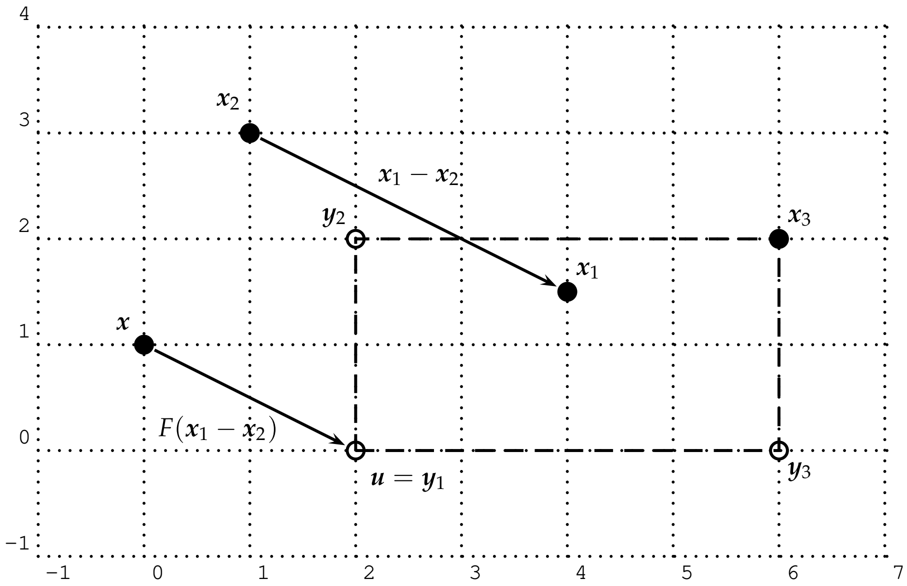
DE Mutation and Recombination

- Mutation and recombination:

$$u \leftarrow x_1 + F(x_2 - x_3), \quad F \in (0, 2)$$

$$x_N, d \leftarrow \begin{cases} u_d & \text{iff } \mathrm{rand}_d \leq CR \text{ or } d = I_{\mathrm{rand}} \\ x_{4,d} & \text{iff } \mathrm{rand}_d > CR \text{ and } d \neq I_{\mathrm{rand}} \end{cases}$$

- $\mathrm{rand}_d \sim \mathcal{U}(0,1)$, different for each dimension
- I_{rand} is a random index of the dimension that is always copied from $\boldsymbol{u}$
- $2^D - 1$ possible candidate points $\boldsymbol{y}$



DE Variants

Small variations of the base algorithm:

- DE/rand vs DE/best: the "best" variant variant uses the best of 4 parent vectors in place of $\boldsymbol{x}$ when generating the offspring.
- DE/./n: n is the number of difference vectors taken into account during mutation.
- DE/./../bin vs DE/./../exp: binomial recombination (described above), exponential recombination (not described here)

DE Variants

Small variations of the base algorithm:

- DE/rand vs DE/best: the "best" variant variant uses the best of 4 parent vectors in place of $\boldsymbol{x}$ when generating the offspring.
- DE/./n: n is the number of difference vectors taken into account during mutation.
- DE/./ ./bin vs DE/./ ./exp: binomial recombination (described above), exponential recombination (not described here)

Many adaptive variants: SaDE, JADE, ...

Swarm Intelligence

Swarm Algorithms

Swarm intelligence:

- In nature: swarm (cz: roj, hejno) of small simple 'units' is able to create very complex behavioral patterns via cooperation.
- **Emergence**: non-linear interactions of simple rules $\longrightarrow$ complex behavior of the whole system.
- Analogy to the behavior of bees, wasps, ants, fish, birds, ...

An engineering view:

- Is it possible to model these systems *in silico* and use that model to solve a practical task?
- How to design the simple units and their interactions such that a practically useful system emerges?

Particle Swarm Optimization

Particle Swarm Optimization

Partice Swarm Optimization (PSO): an optimization algorithm inspired by the behavior of birds.

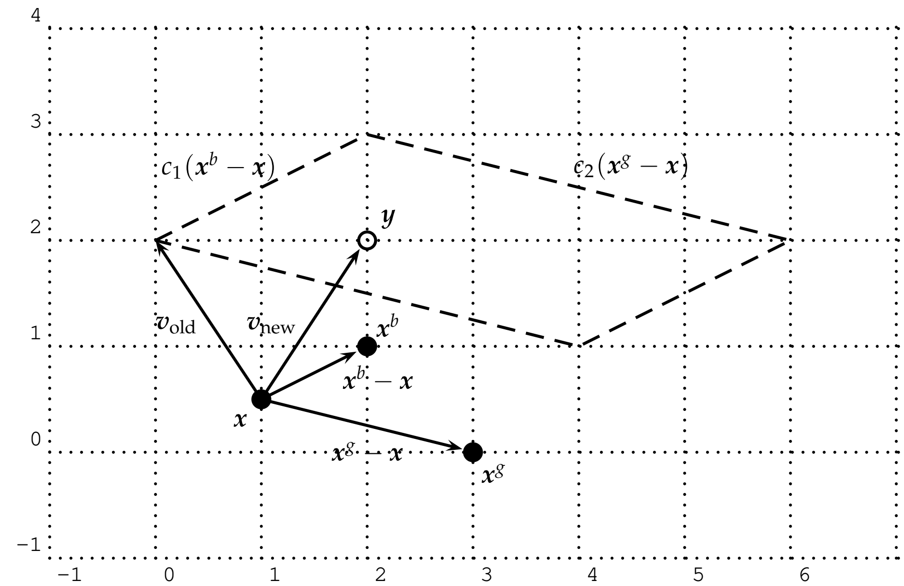
Inspiration:

- Birds fly over the landscape and land on the highest hill.
- Birds are modeled by particles in a multidimensional vector space.
- The particles have their *position* and *speed* (and momentum).
- They remember their own best position (i.e., the highest place of the landscape they flew over), but also
- they communicate and use the best position of their neighboring particles to update their own position and speed.
- The communication is usually of 2 types:
  1. **Globally best position** is known to all particles and is updated as soon as any particle finds an improvement.
  2. **Best position in neighborhood** is shared among a group of neighboring particles.

PSO Algorithm

Algorithm 2: Canonical PSO

1 **begin**
2 Initialize positions $\boldsymbol{x}_i$ and velocities $\boldsymbol{v}_i$.
3 Initialize personal best positions $\boldsymbol{x}_i^b \leftarrow \boldsymbol{x}_i$.
4 Initialize globally best position $\boldsymbol{x}^g \leftarrow \boldsymbol{x}_k, \forall i : f(\boldsymbol{x}_k) \le f(\boldsymbol{x}_i)$
5 **for** $i = 1, \dots, N$ **do**
6 $\boldsymbol{v}_i \leftarrow \omega \boldsymbol{v}_i + c_1 \boldsymbol{r}_1 \circ (\boldsymbol{x}_i^b - \boldsymbol{x}_i) + c_2 \boldsymbol{r}_2 \circ (\boldsymbol{x}^g - \boldsymbol{x}_i)$
7 $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i + \boldsymbol{v}_i$
8 If $f(\boldsymbol{x}_i) < f(\boldsymbol{x}_i^b)$, $\boldsymbol{x}_i^b \leftarrow \boldsymbol{x}_i$.
9 If $f(\boldsymbol{x}_i) < f(\boldsymbol{x}^g)$, $\boldsymbol{x}^g \leftarrow \boldsymbol{x}_i$.
10 If termination condition not satisfied, go to 5.

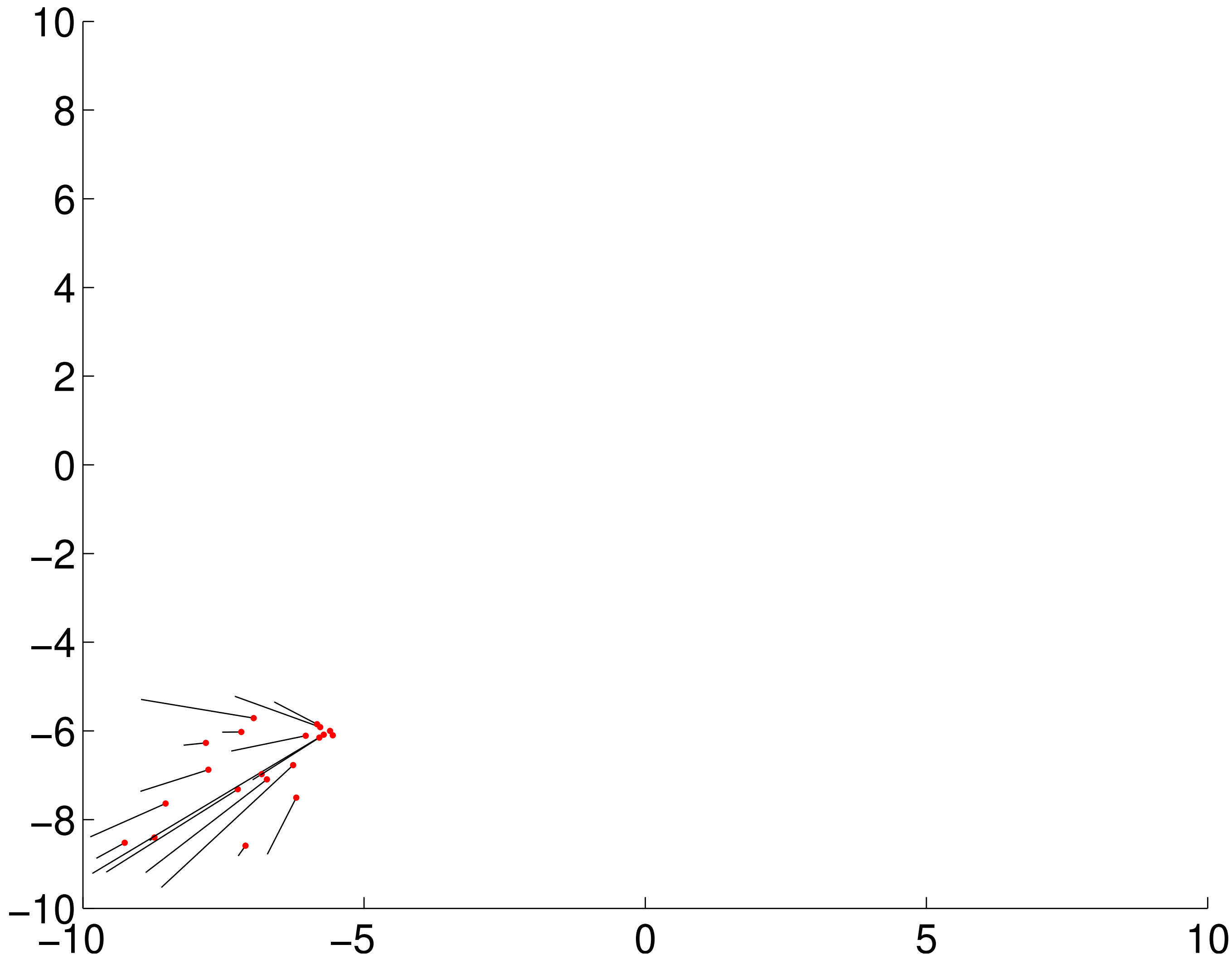


Meaning of symbols:

f	objective function (landscape) $f : \mathcal{R}^D \rightarrow \mathcal{R}$
N	the number of particles
$\boldsymbol{x}_i$	particle positions, $\boldsymbol{x}_i \in \mathcal{R}^D$
$\boldsymbol{v}_i$	particle velocities, $\boldsymbol{v}_i \in \mathcal{R}^D$
$\boldsymbol{x}_i^b$	personal best position
$\boldsymbol{x}^g$	globally best position
ω	particle momentum, suitable value is e.g. 0.9, sometimes it decreases during simulation e.g. to 0.4.
c_1, c_2	attraction constants, 'cognitive' and 'social' componments, suitable values between 1 and 2
$\boldsymbol{r}_1, \boldsymbol{r}_2$	random vectors from $U(0,1)^D$
$\circ$	vector multiplication by items

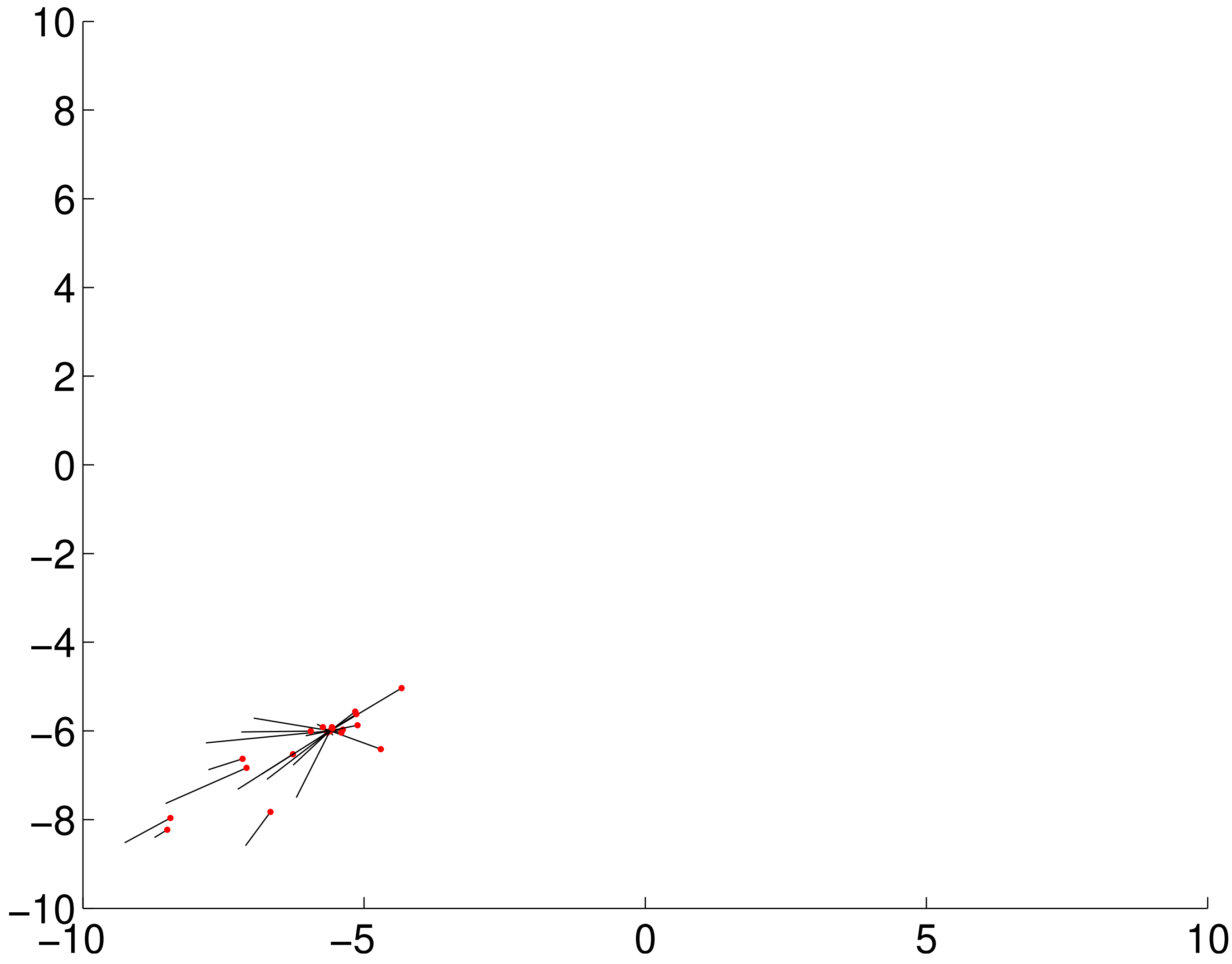
PSO: Demo

PSO on 2D Sphere function:



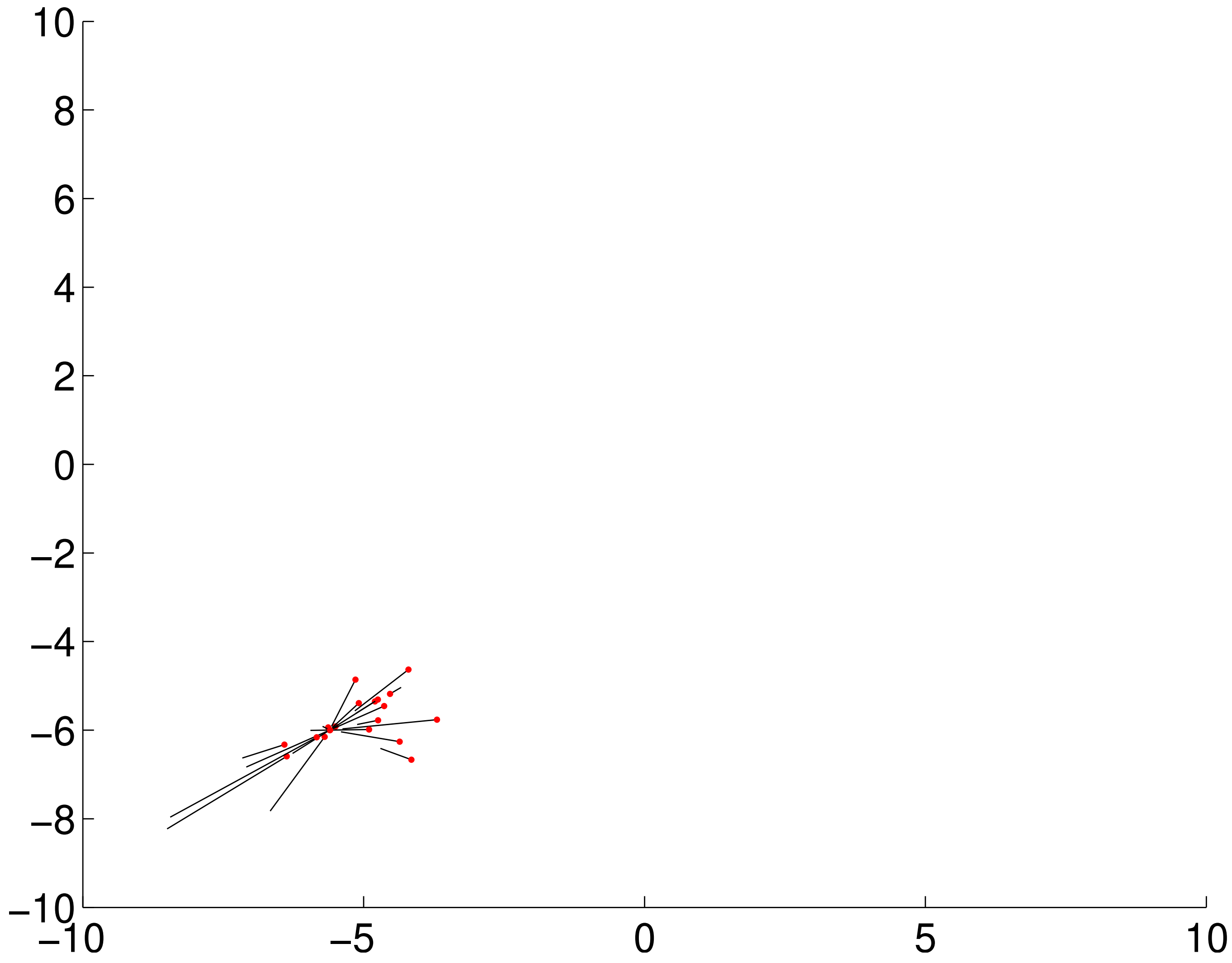
PSO: Demo

PSO on 2D Sphere function:



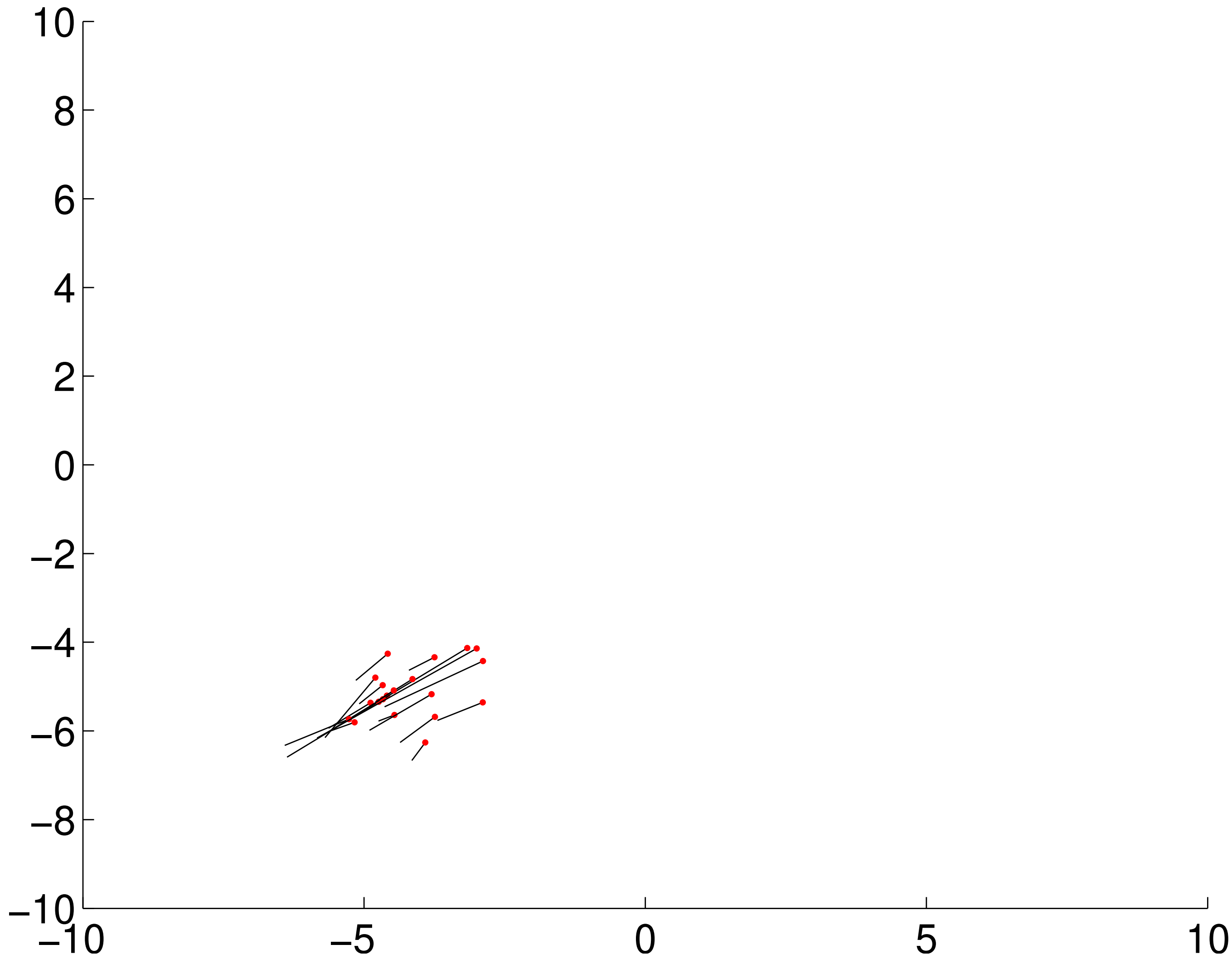
PSO: Demo

PSO on 2D Sphere function:



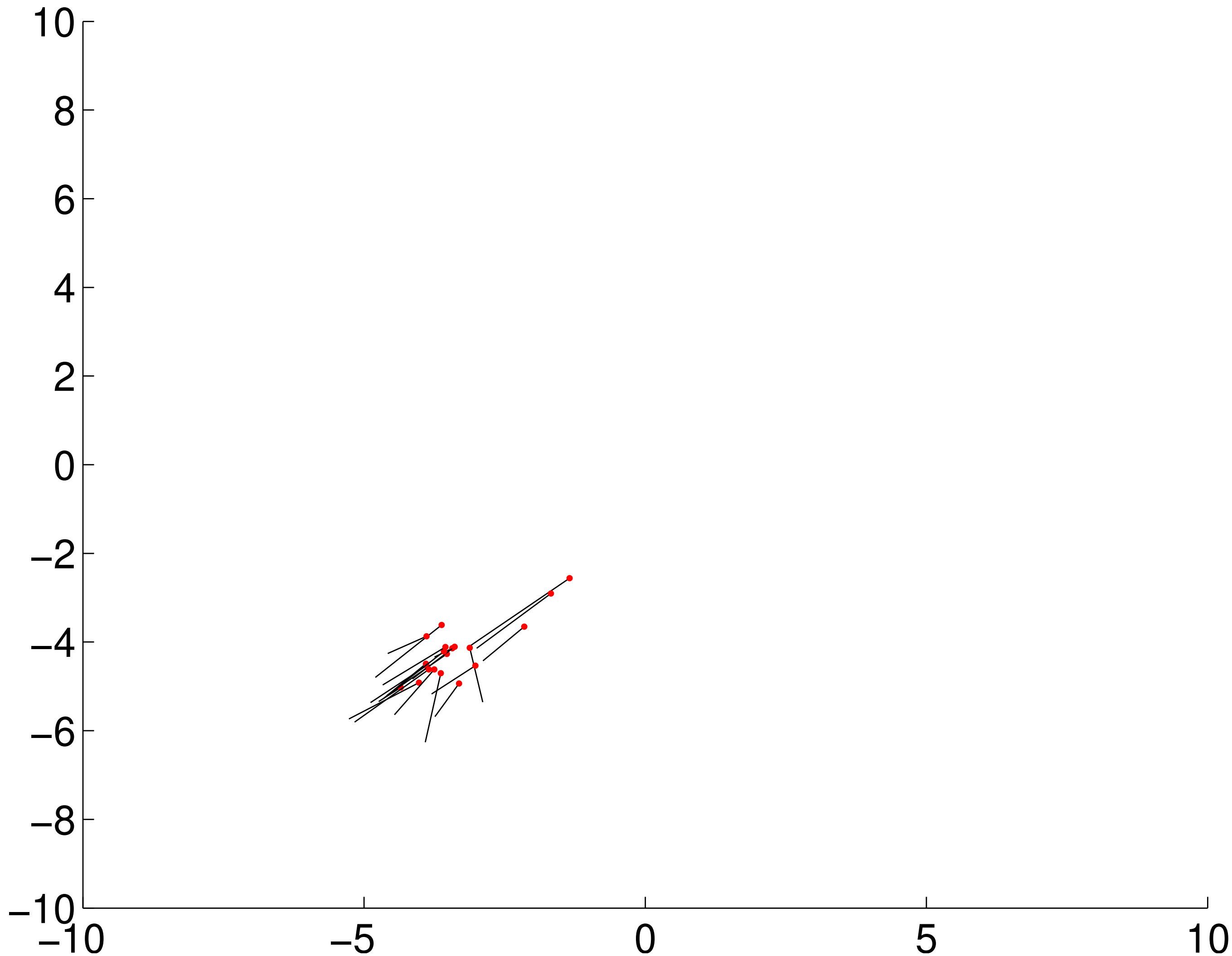
PSO: Demo

PSO on 2D Sphere function:



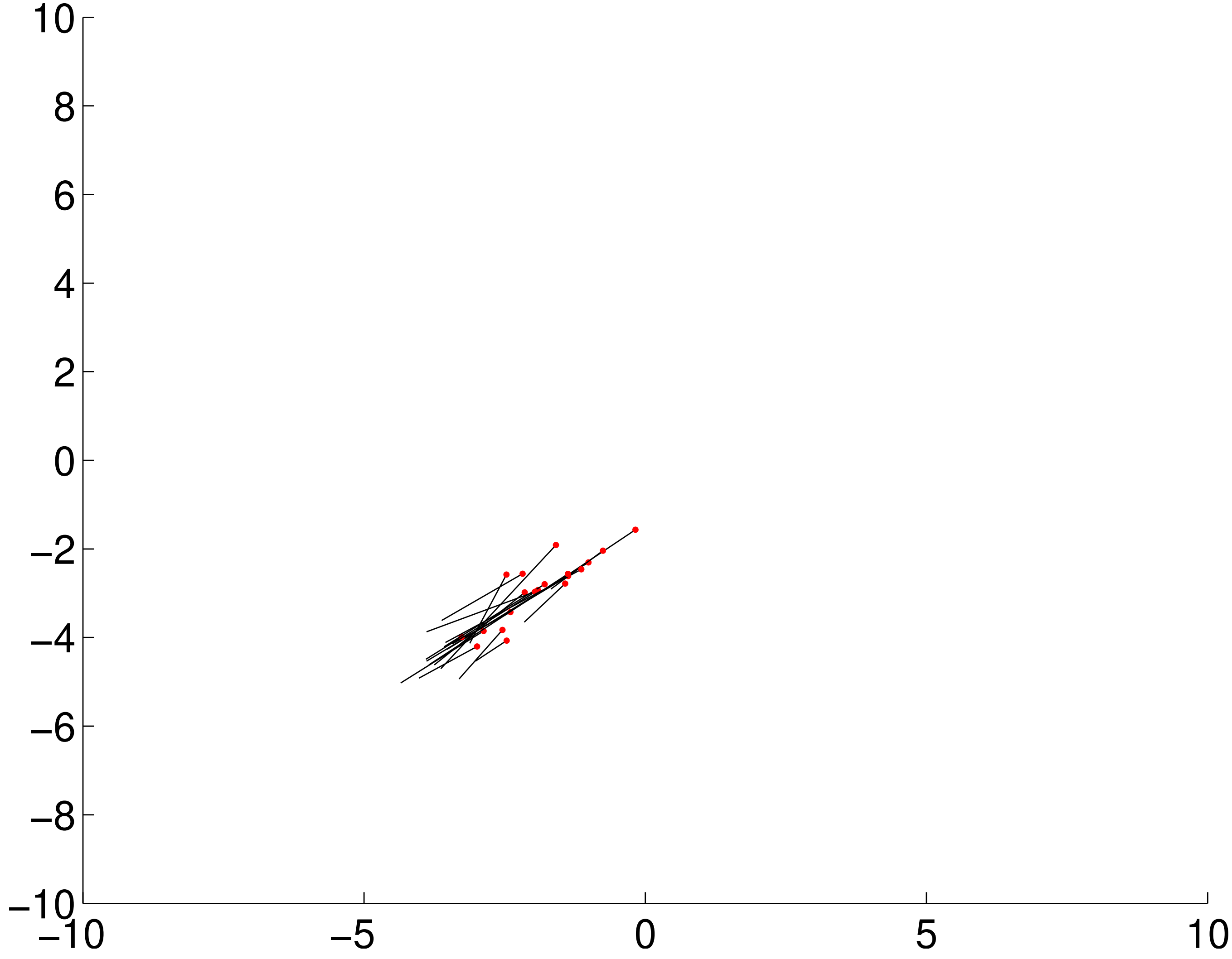
PSO: Demo

PSO on 2D Sphere function:



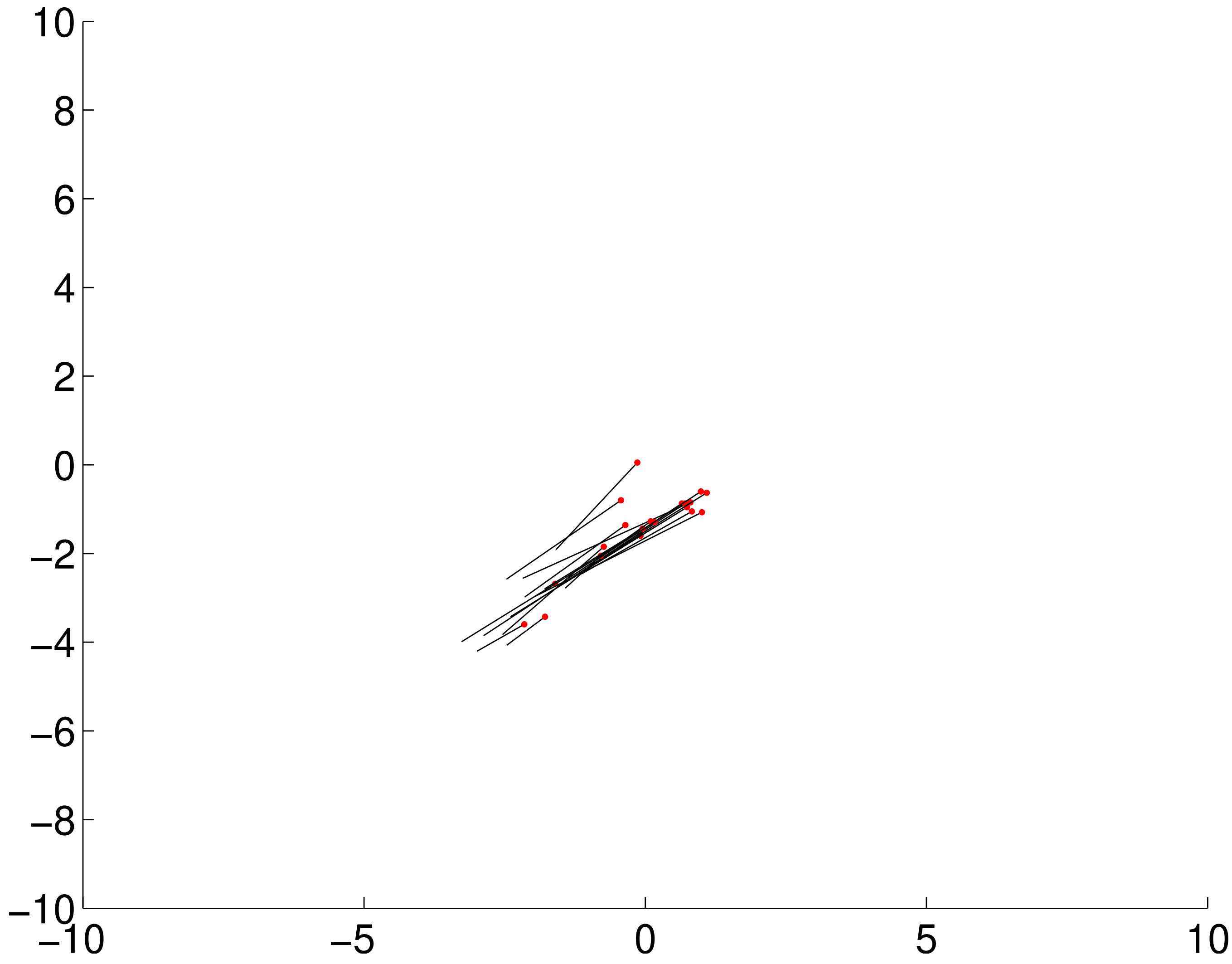
PSO: Demo

PSO on 2D Sphere function:



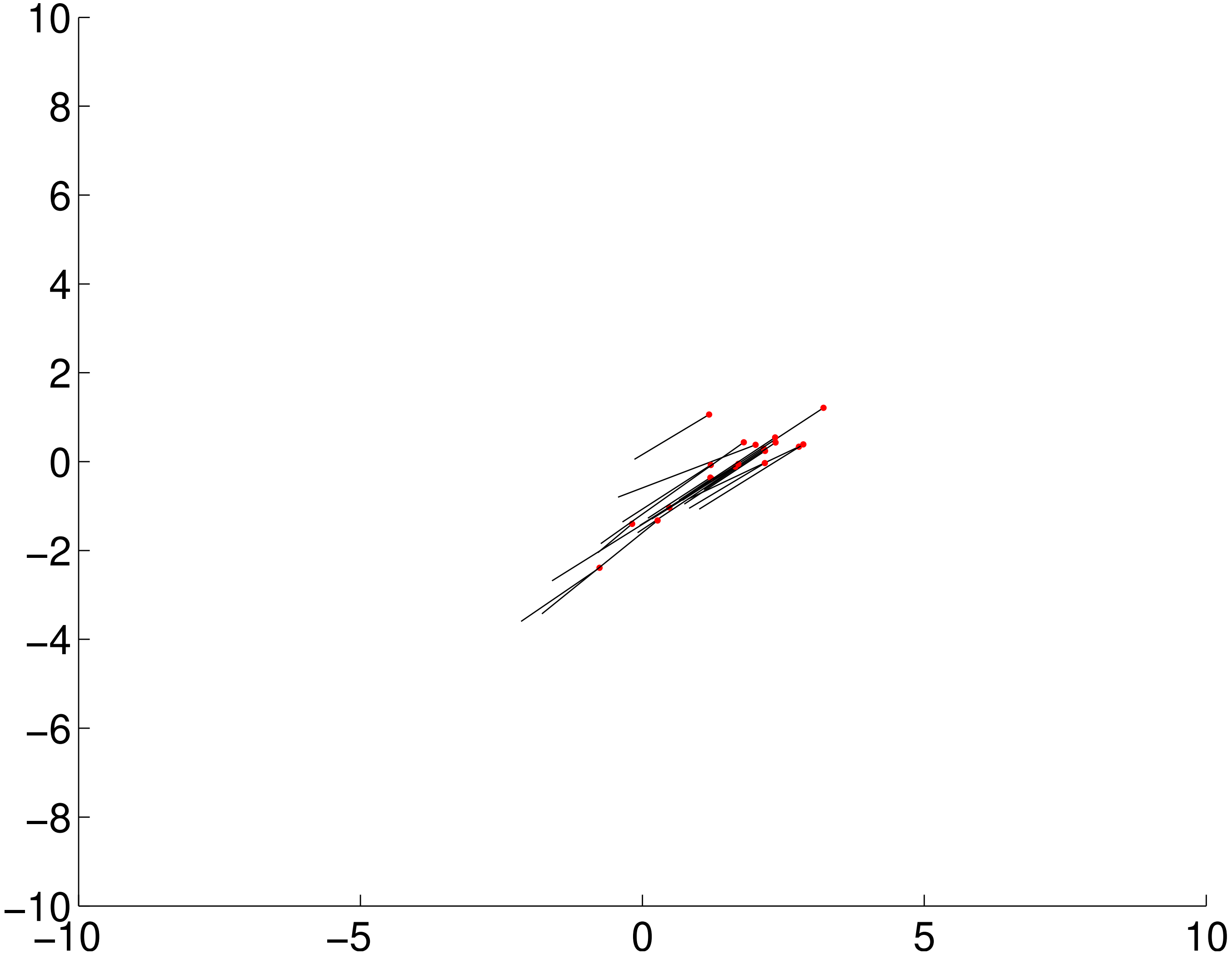
PSO: Demo

PSO on 2D Sphere function:



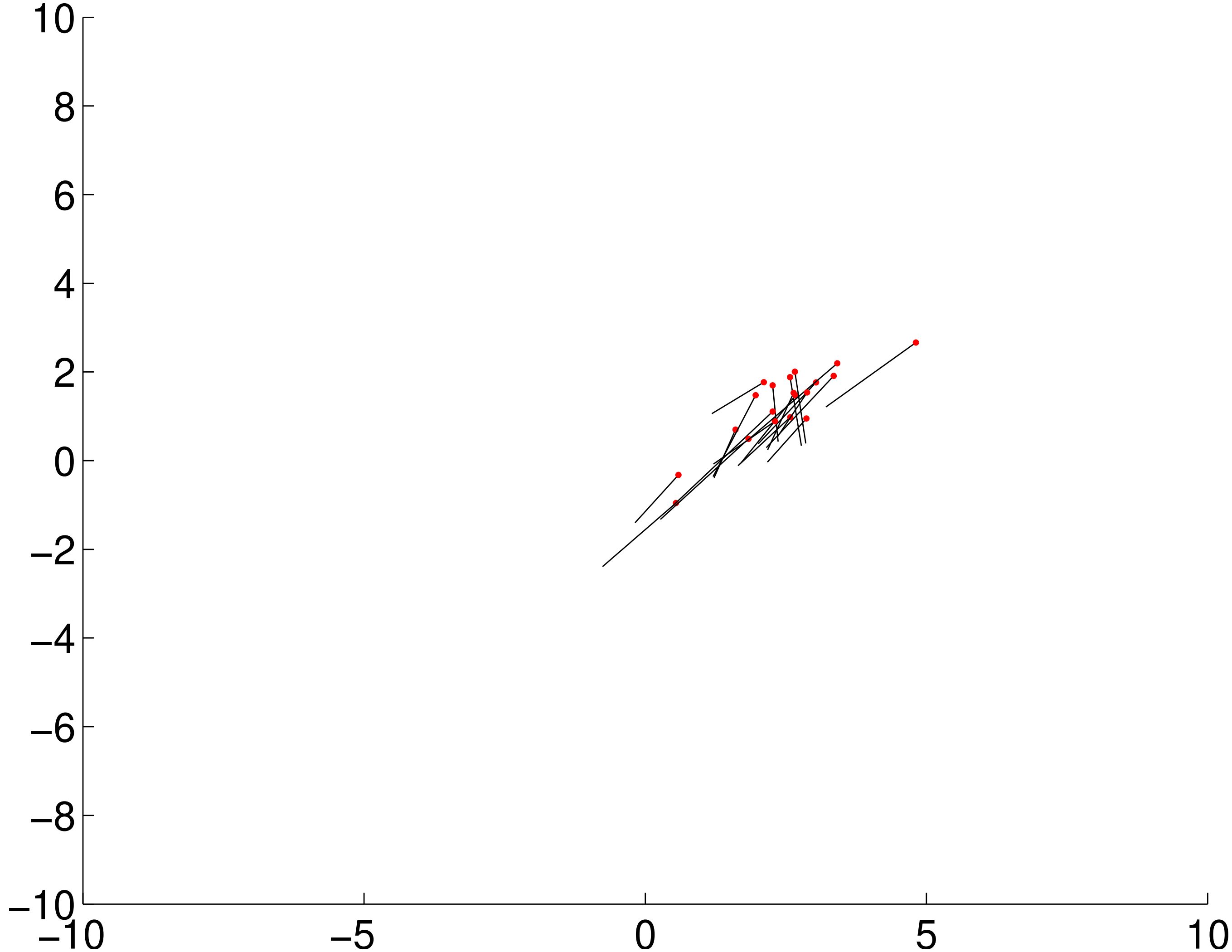
PSO: Demo

PSO on 2D Sphere function:



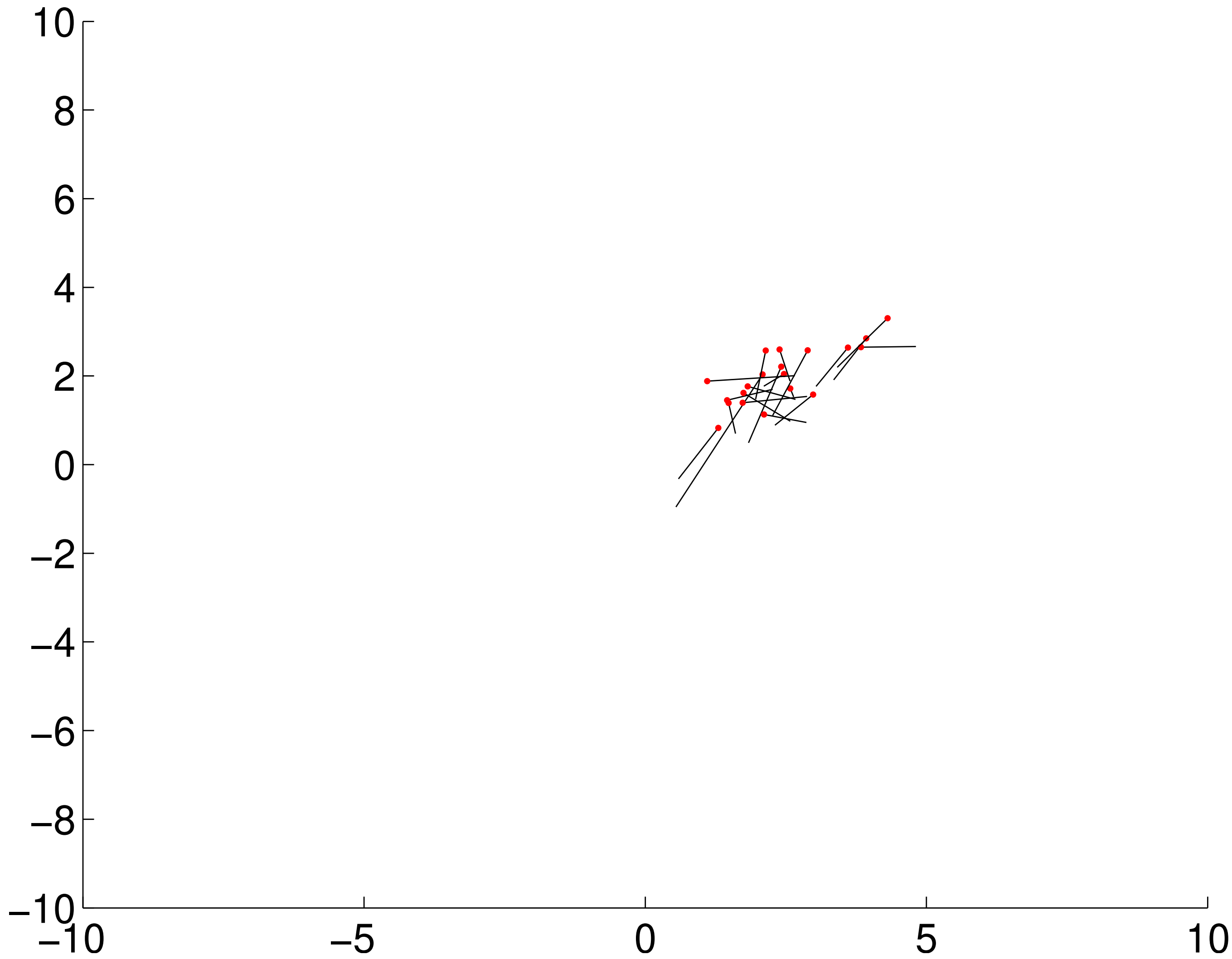
PSO: Demo

PSO on 2D Sphere function:



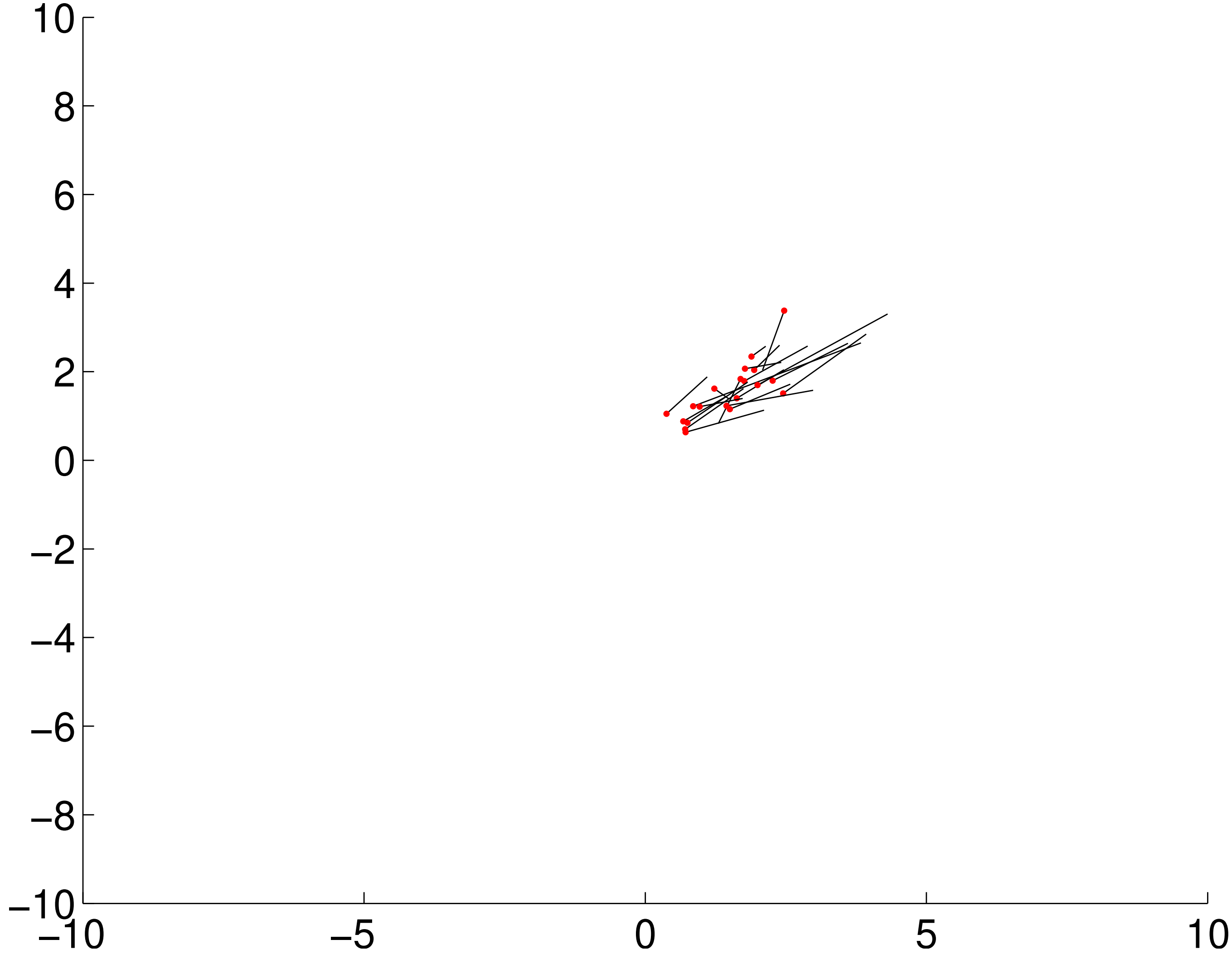
PSO: Demo

PSO on 2D Sphere function:



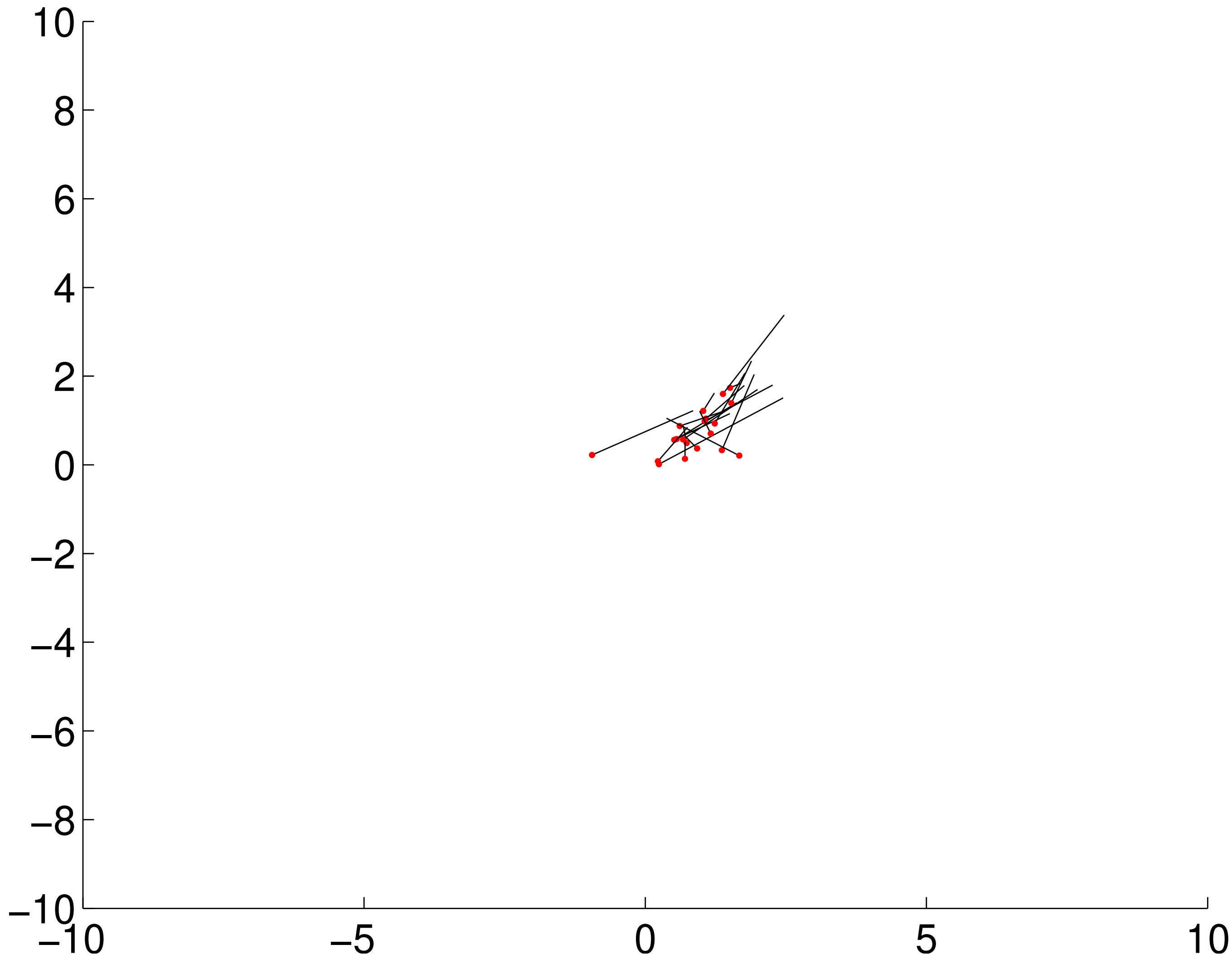
PSO: Demo

PSO on 2D Sphere function:



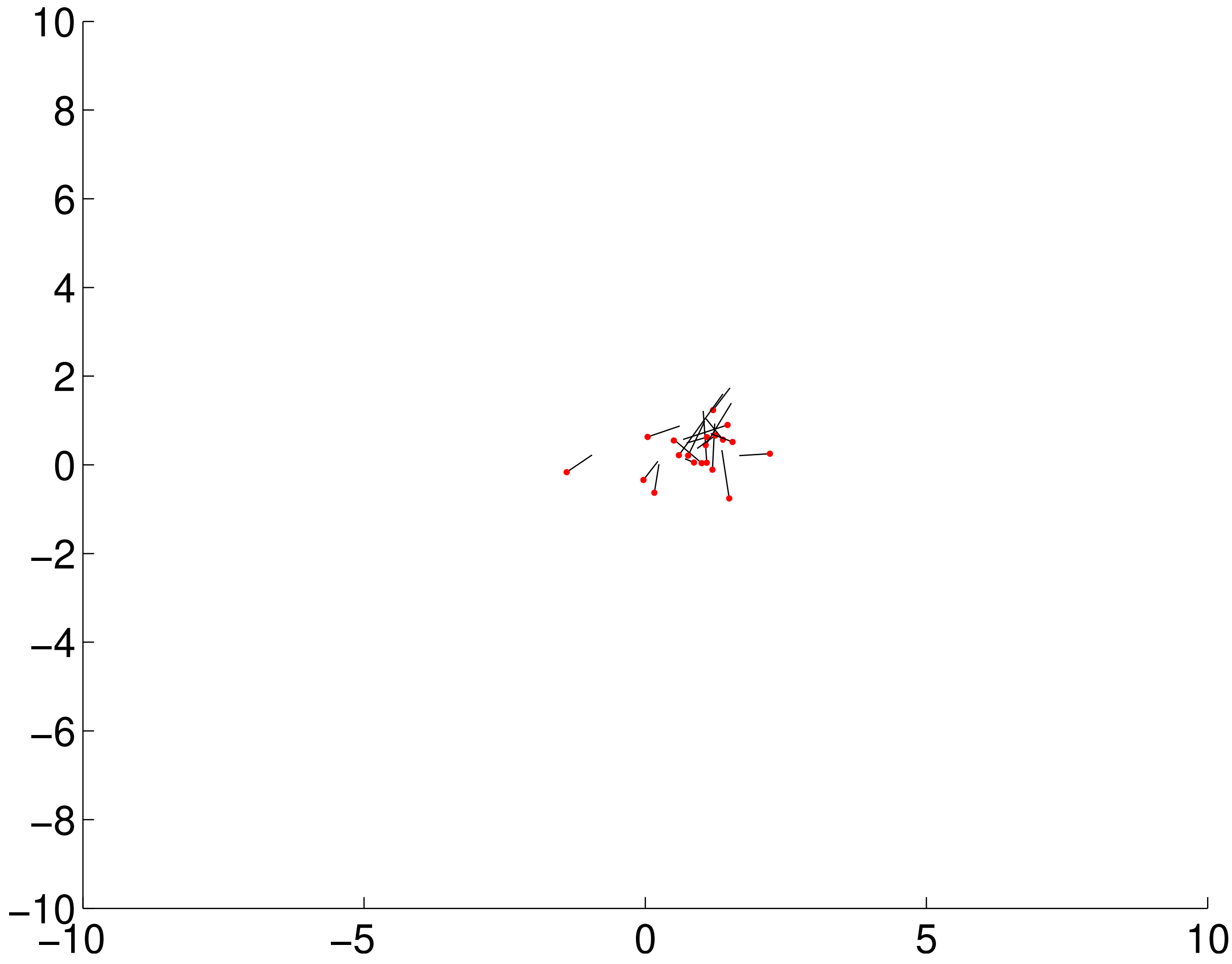
PSO: Demo

PSO on 2D Sphere function:



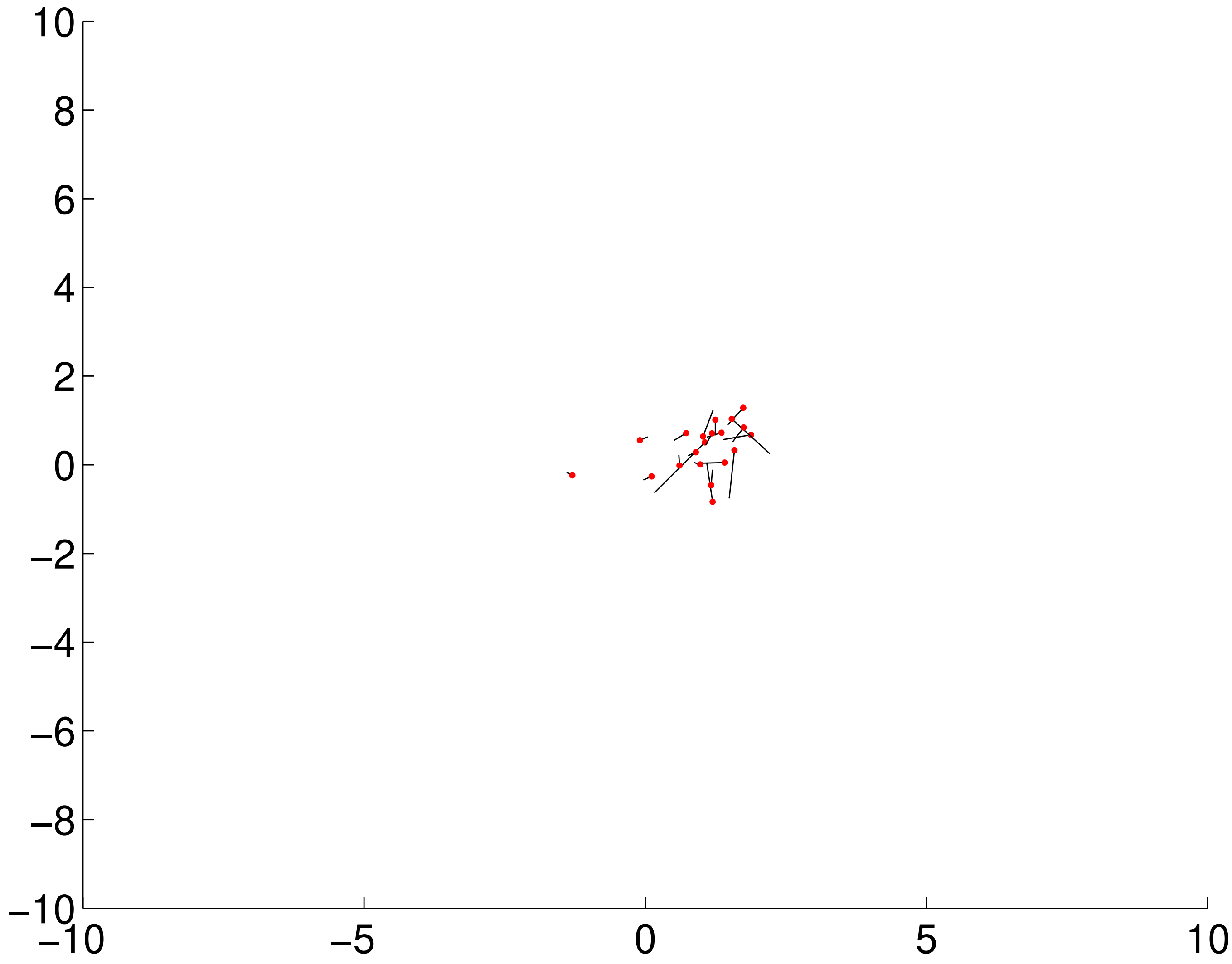
PSO: Demo

PSO on 2D Sphere function:



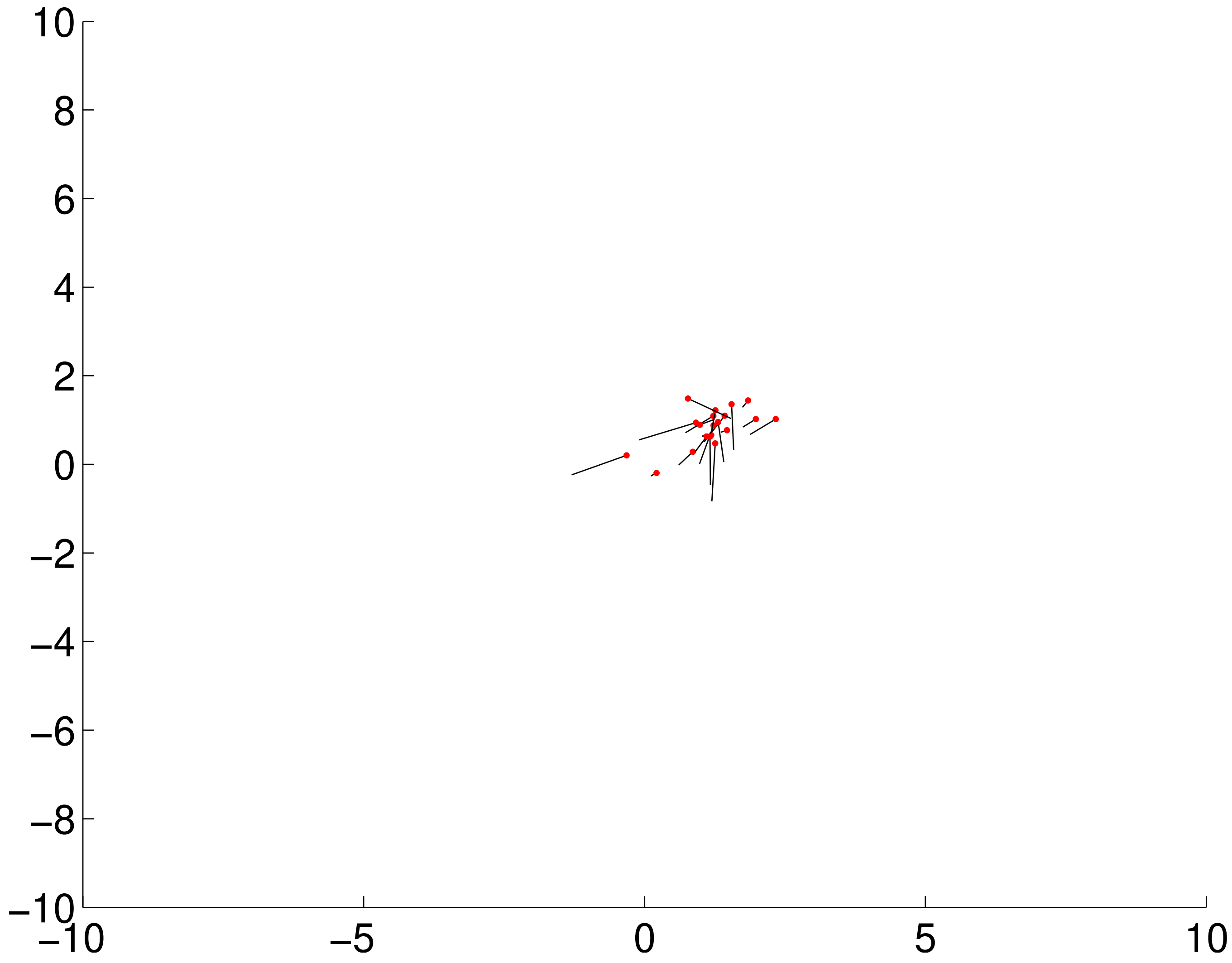
PSO: Demo

PSO on 2D Sphere function:



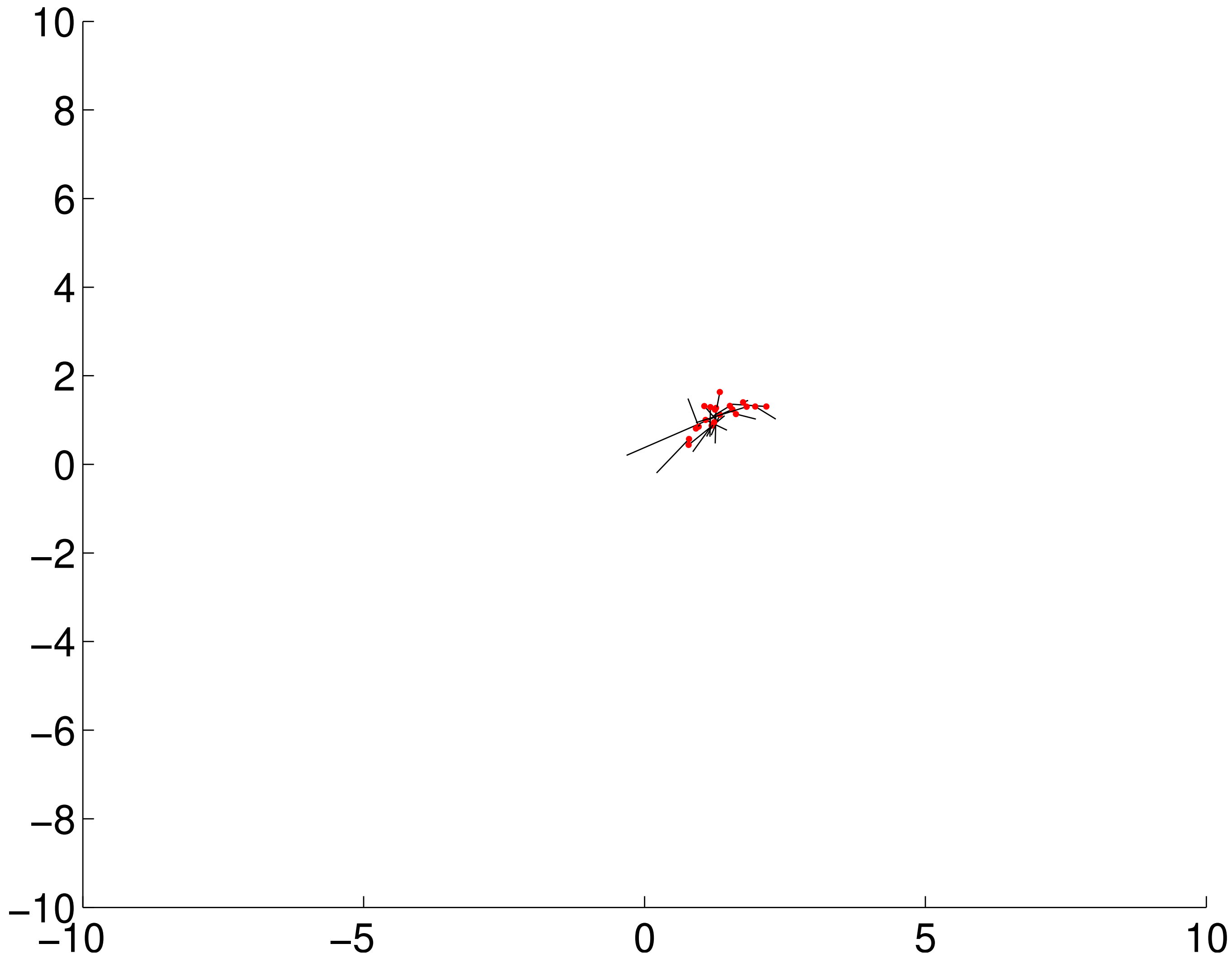
PSO: Demo

PSO on 2D Sphere function:



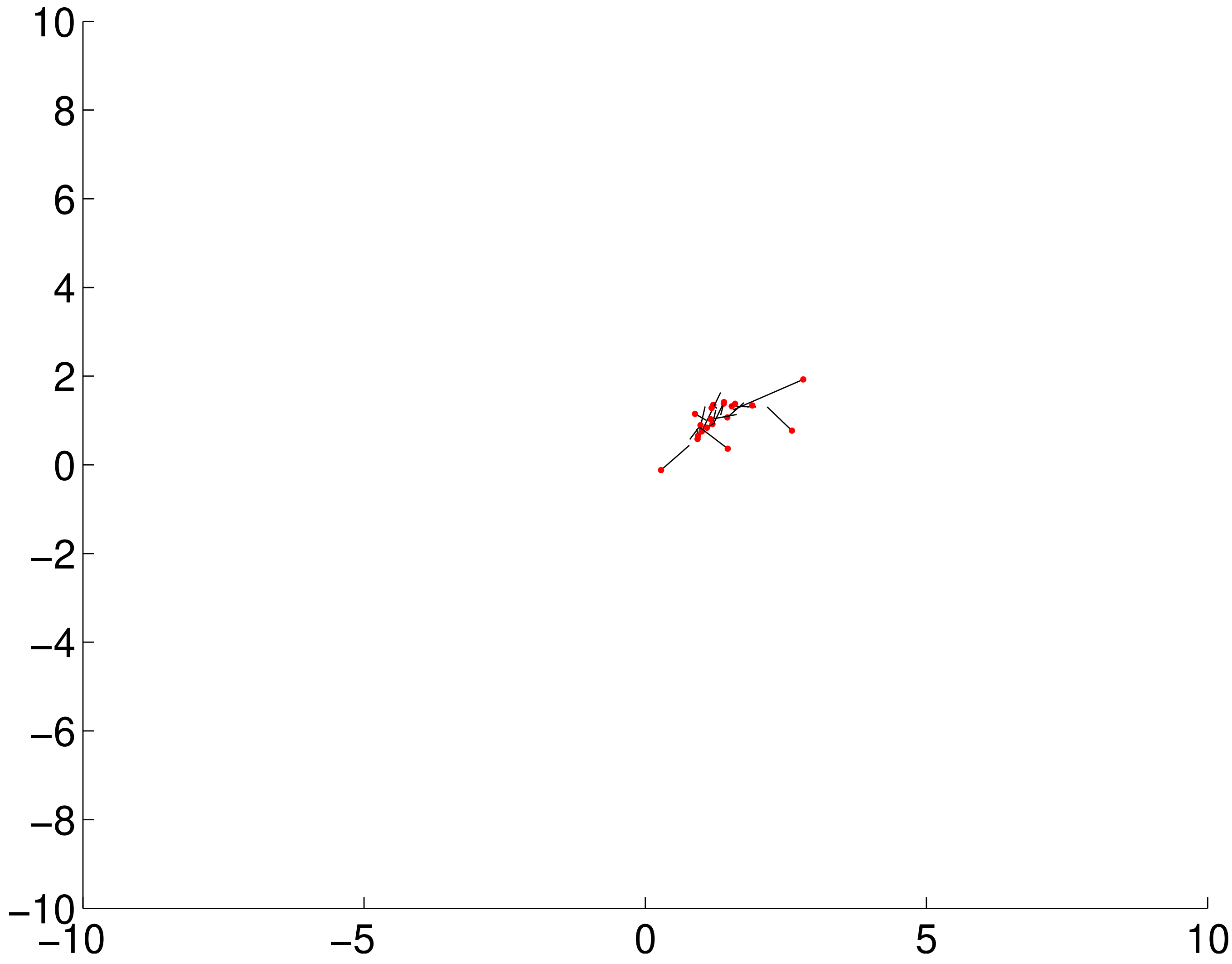
PSO: Demo

PSO on 2D Sphere function:



PSO: Demo

PSO on 2D Sphere function:



Ant Colonies

Ant colonies

Ants:

- Social insects
- Ant colonies exhibit an intelligent behavior:
  - labor division, work coordination
  - complex nests
  - ability to find 'low-energy' path between the nest and a food source
- They communicate by
  1. physical contact (they touch with their antennas)
  2. interaction with the environment (pheromone trails)

"In nature, ants first search their environment randomly, until they find a source of food. Then, they return to the nest and lay a pheromone trail behind. Other ants are able to sense this pheromone trail and are able to follow it, and thus make it stronger. The pheromone evaporates; after the food source is exhausted, the random foraging reemerges."

Ant Colony Optimization

Ant Colony Optimization (ACO) is a class of stochastic optimization algorithms for solving combinatorial problems.

Similarities with the real ants:

- a colony of cooperating individuals
- pheromone trail
- indirect communication via pheromone (stigmergy)
- probabilistic decision making, local strategies

Differences from the real ants:

- (usually) discrete world (a graph)
- inner state, memory
- the amount of pheromone train can depend on the solution quality
- may use several types of pheromones

Ant Colony Optimization

Ant Colony Optimization (ACO) is a class of stochastic optimization algorithms for solving combinatorial problems.

Similarities with the real ants:

- a colony of cooperating individuals
- pheromone trail
- indirect communication via pheromone (stigmergy)
- probabilistic decision making, local strategies

Differences from the real ants:

- (usually) discrete world (a graph)
- inner state, memory
- the amount of pheromone train can depend on the solution quality
- may use several types of pheromones

Algorithm 3: ACO

begin
    Initialize the pheromone trails on graph edges: $\tau_{ij}(0) = \tau_0$.
    Set the initial position of ants in the graph.
    while *not termination condition* **do**
        foreach *ant* **do**
            Build a solution.
            Apply local search. `// Optional, but used very often.`
        Update pheromone trails.

Algorithm parts

Ant k constructs a solution:

- Probability the ant will move from the current node i to neighboring node j is

$$p_{ij}^k(t) = \frac{(\tau_{ij}(t))^\alpha (\eta_{ij})^\beta}{\sum_{l \in \mathcal{N}_i^k} (\tau_{il}(t))^\alpha (\eta_{il})^\beta}, \text{kde } j \in \mathcal{N}_i^k,$$

  where

τ_{ij}	the amount of pheromone on edge $i \to j$,
$\eta_{ij} = \frac{1}{d_{ij}}$	known heuristic information,
α, β	the influence of pheromone and heuristic information, respectively,
$\mathcal{N}_i^k$	a set of graph nodes accessible to ant k from node i.

- If $\alpha = 0$, only the heuristic information has an effect, and the solution construction reduces to greedy algorithm (nearest neighbor heuristic).
- If $\beta = 0$, only the pheromone trail has an effect. The paths found in the first iteration have a big influence. Moreover, if $\alpha > 1$, stagnation occurs very fast, i.e. all ants use the same (not optimal) path.
- Suggested values of parameters:

$$\alpha = 1 \qquad \beta = 2 \text{ až } 5 \qquad \rho = 0.5 \qquad m = n \text{ (TSP)} \qquad \tau_0 = m/C^{nn} \text{ (TSP)}$$

  m is the number of ants, n is the number of cities, C^{nn} is the length of the path constructed by the nearest neighbor heuristic.

Algorithm parts (cont.)

Pheromone update on all edges

- Done after all ants find their solution.
- Pheromone evaporation: $\tau_{ij} \leftarrow (1-\rho)\tau_{ij}$.
  ρ is the evaporation rate, allows to 'forget' bad paths.
- Pheromone deposition from all ants: $\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^k$, where

$$\Delta\tau_{ij}^k = \begin{cases} 1/C^k & \text{if ant } k \text{ used edge } i \rightarrow j \\ 0 & \text{otherwise,} \end{cases}$$

  C^k is the length of the path of ant k.

Other options:

- The best path is reinforced the most.
- The amount of deposited pheromone is proportional to the ant rank according to the path lengths (i.e., not directly proportional to path lengths).
- Update of pheromone trails as soon as an ant uses and edge.
- More types of pheromones can be used:
  - Ants can start from both the nest and the food source.
  - We can have more types of ants.
- ...

Applications

ACO was able to find good solutions in the following tasks:

- Traveling salesperson problem
- Network routing, vehicle routing
- Scheduling
- Quadratic assignment problem
- Shortest common supersequence
- Classification rule learning
- ...

Advantages:

- The graph topology can change in time (e.g. in routing problems)

Demo: ant foraging

Conclusions

Summary

- There are plenty of nature-inspired techniques, other than EAs.
- Swarm intelligence takes advantage of the emergent swarm behavior which is a result of simple interactions among individual swarm members.
- Particle swarm optimization primarily aims at real-parameter optimization, but there are also variants suitable for discrete spaces.
- Ant colonies are used to solve problems which can be reduced to search for the shortest path in a graph (combinatorial problems). Again, variants for real-parameter optimization exist (but are somewhat 'unnatural').