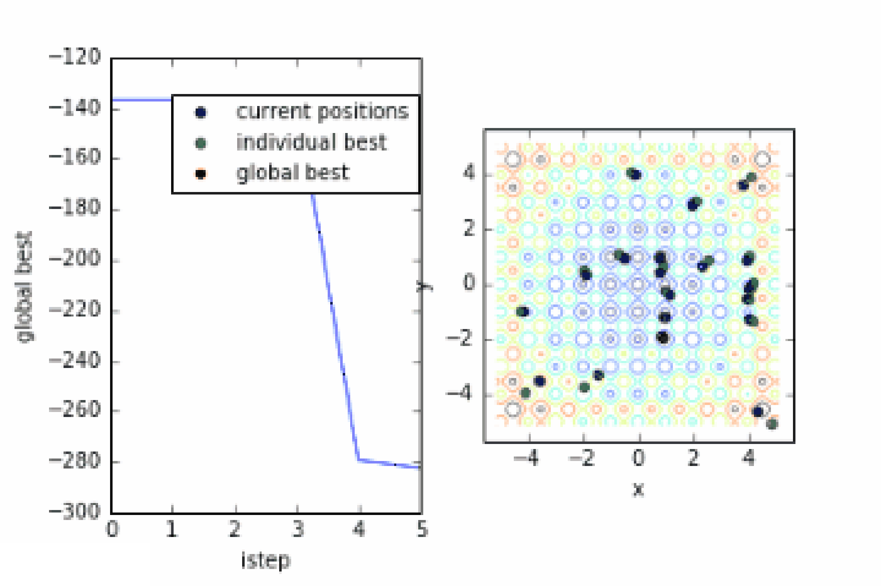
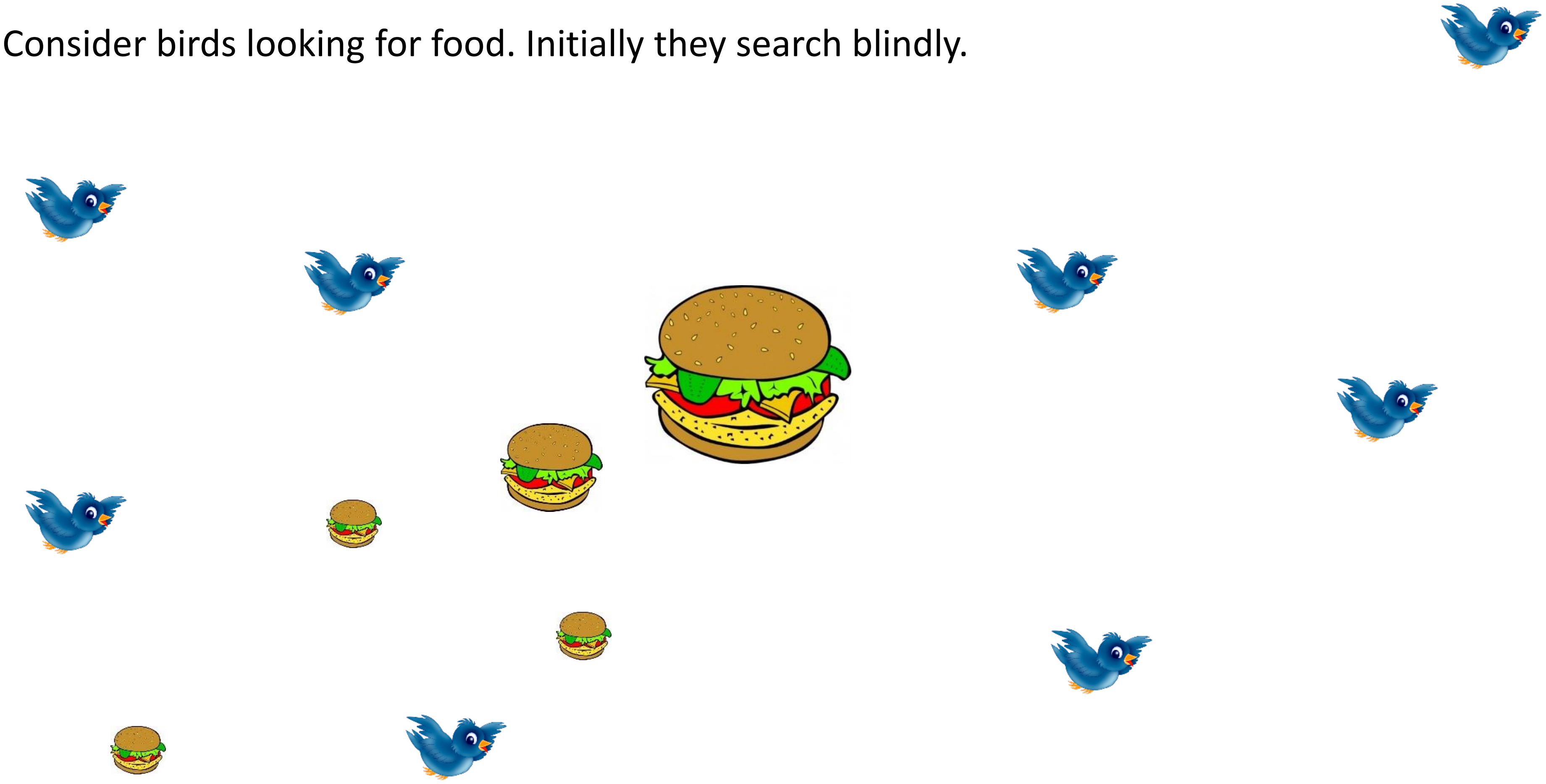
Particle Swarm Optimization

Presented by: Yubo “Paul” Yang



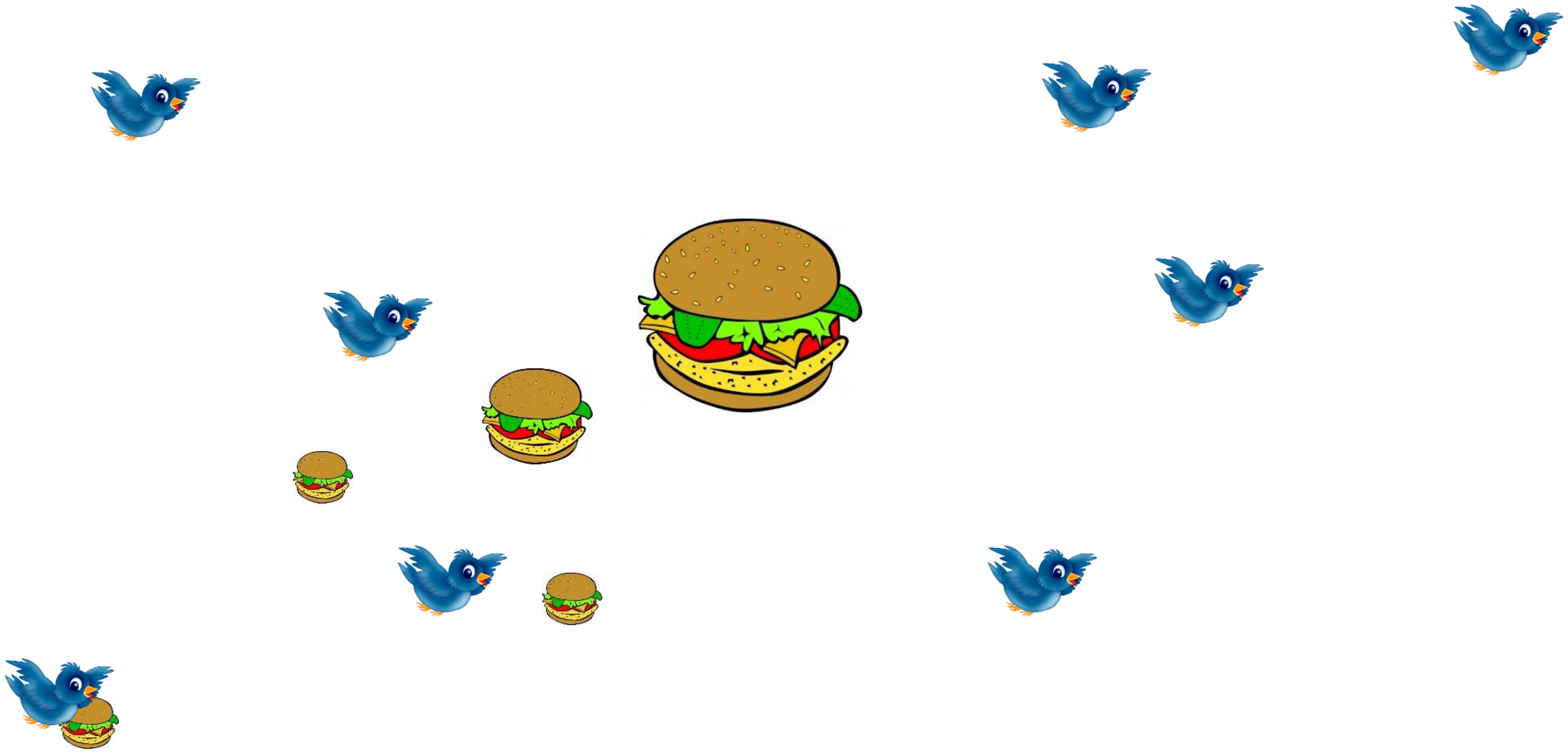
Motivation: Swarm Intelligence

Consider birds looking for food. Initially they search blindly.



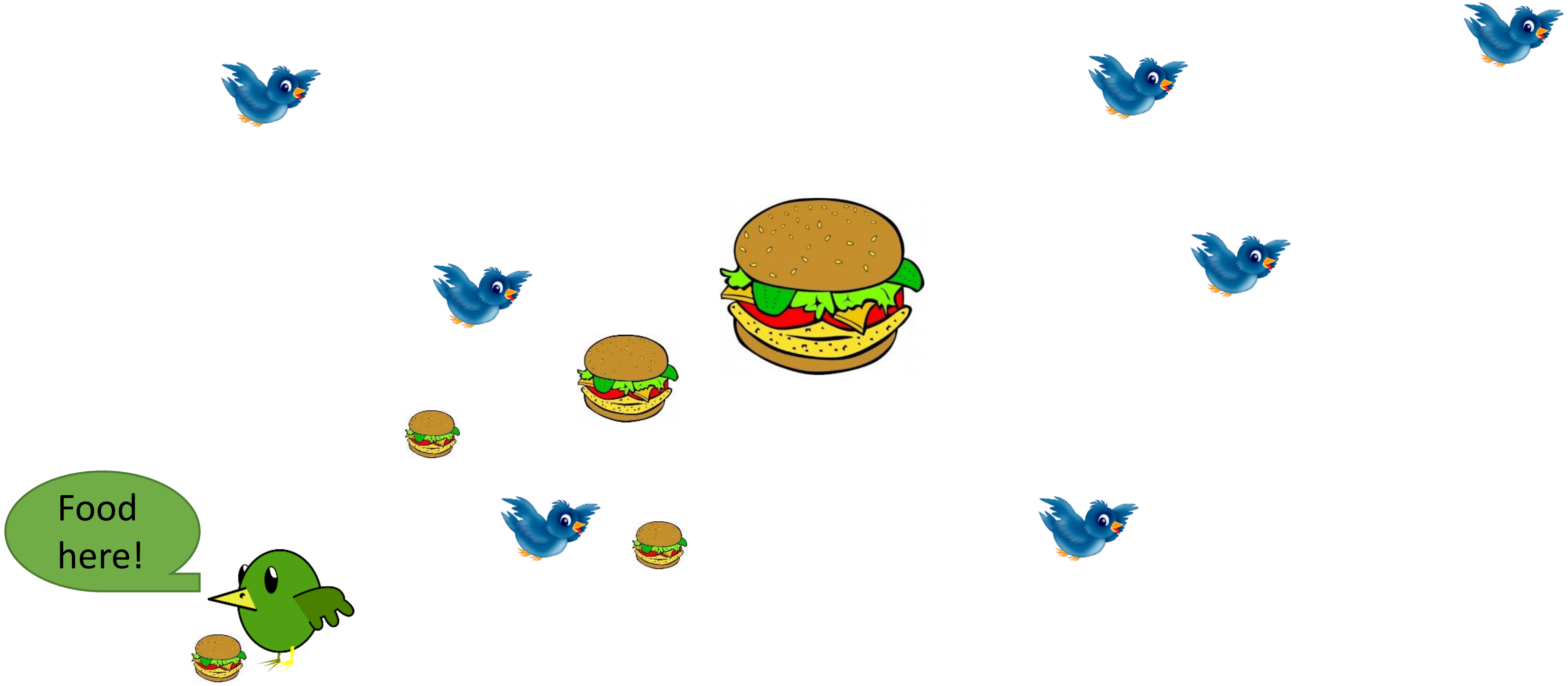
Motivation: Swarm Intelligence

Consider birds looking for food. Initially they search blindly.



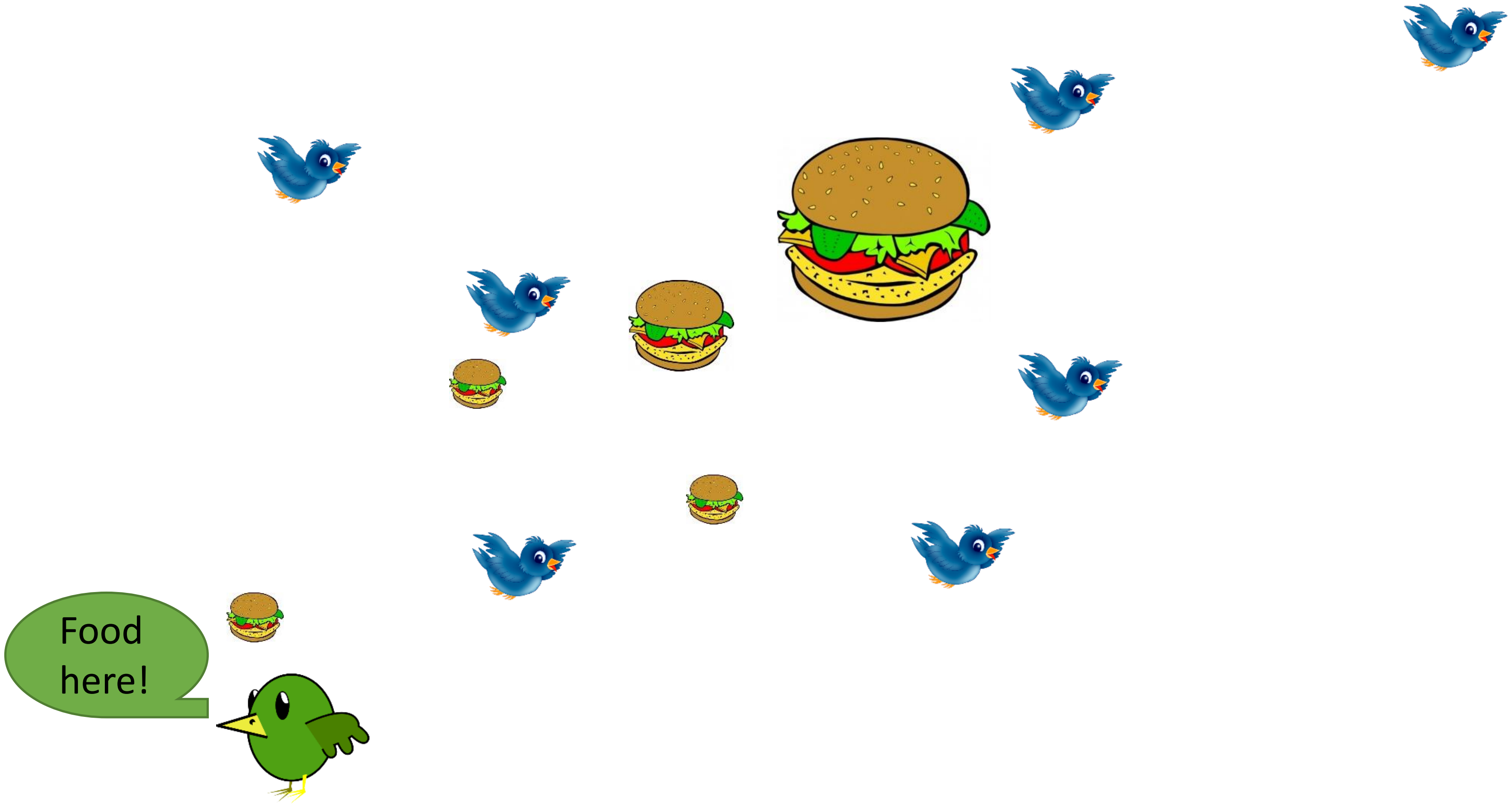
Motivation: Swarm Intelligence

As soon as one of them find food, it circles the food, and maybe yell and get fatter.



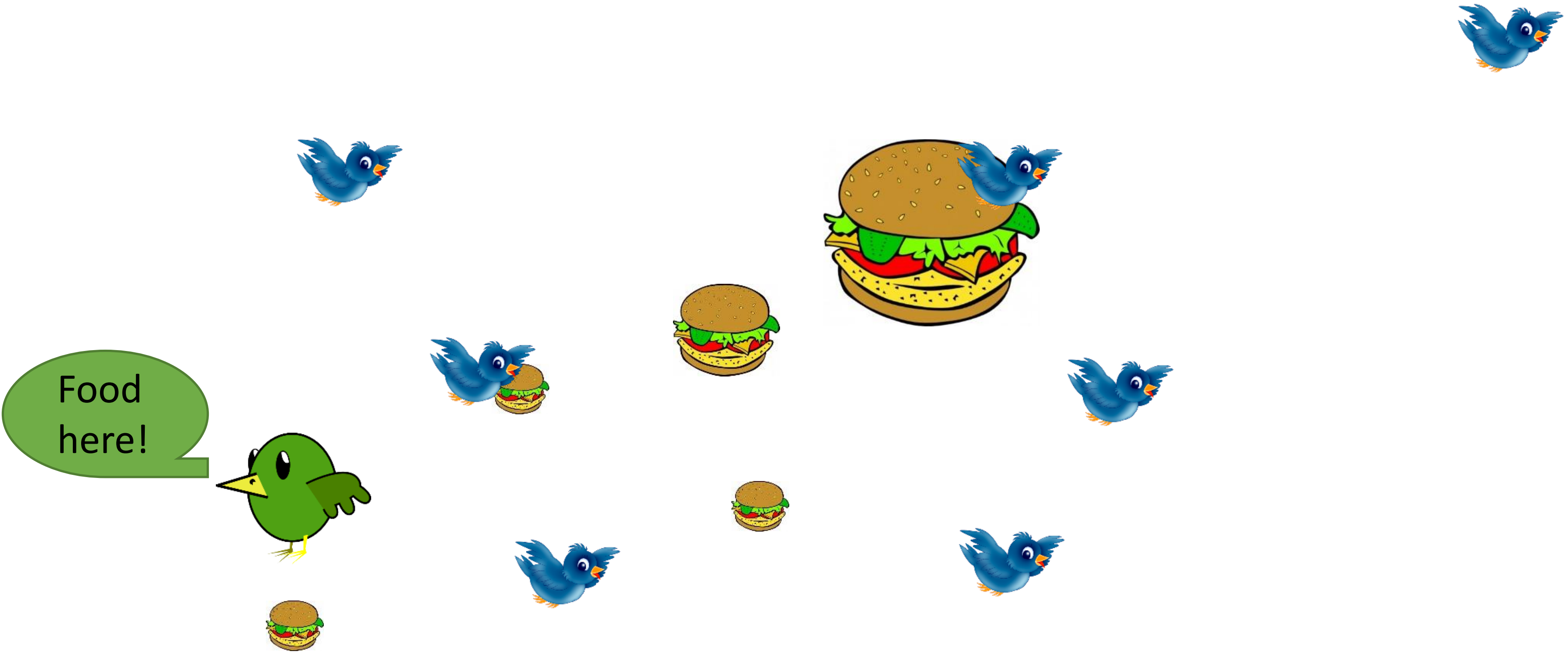
Motivation: Swarm Intelligence

Other birds then flock towards the noisy fat birdy.



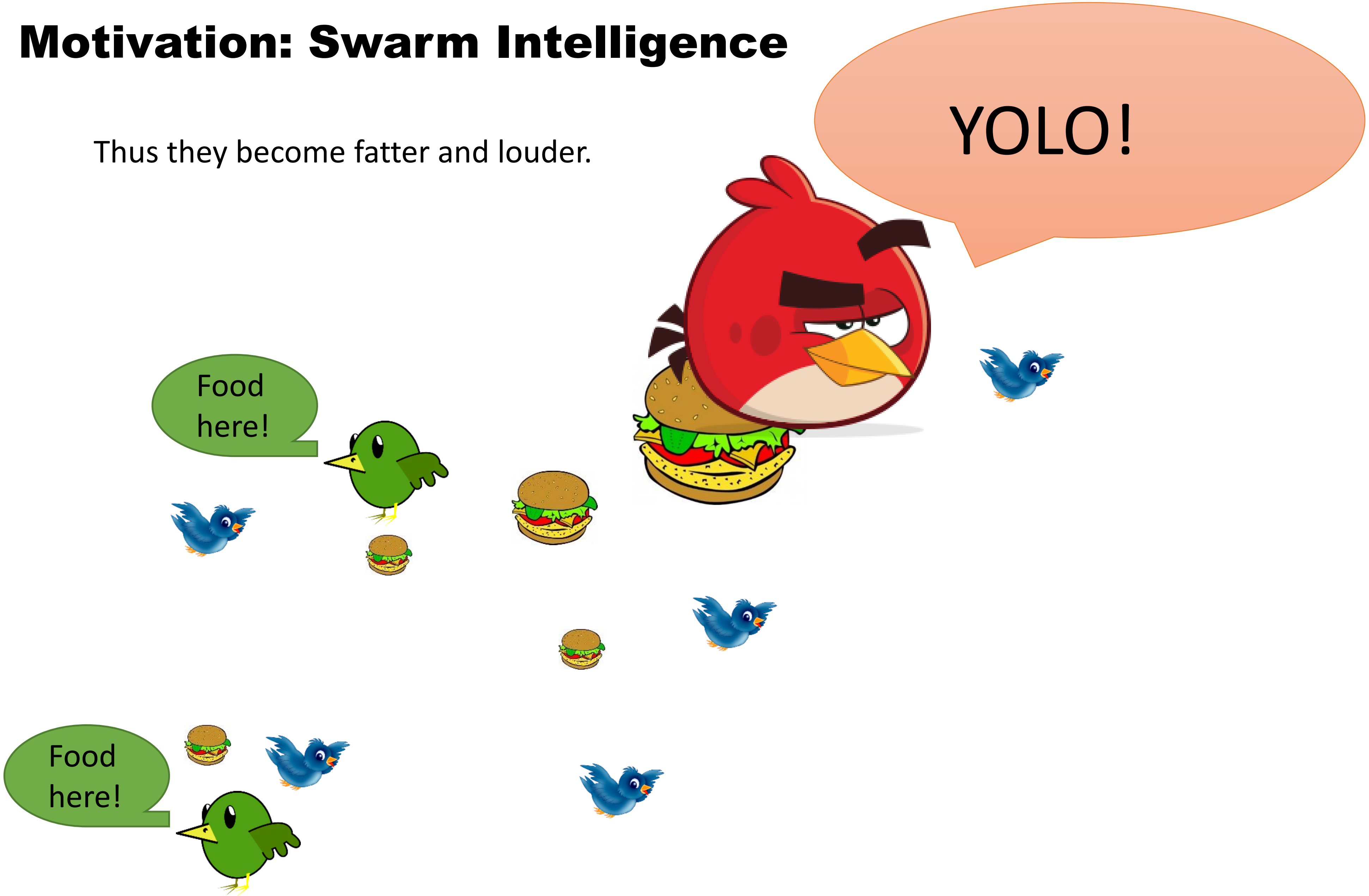
Motivation: Swarm Intelligence

On their way, they may find even more food.



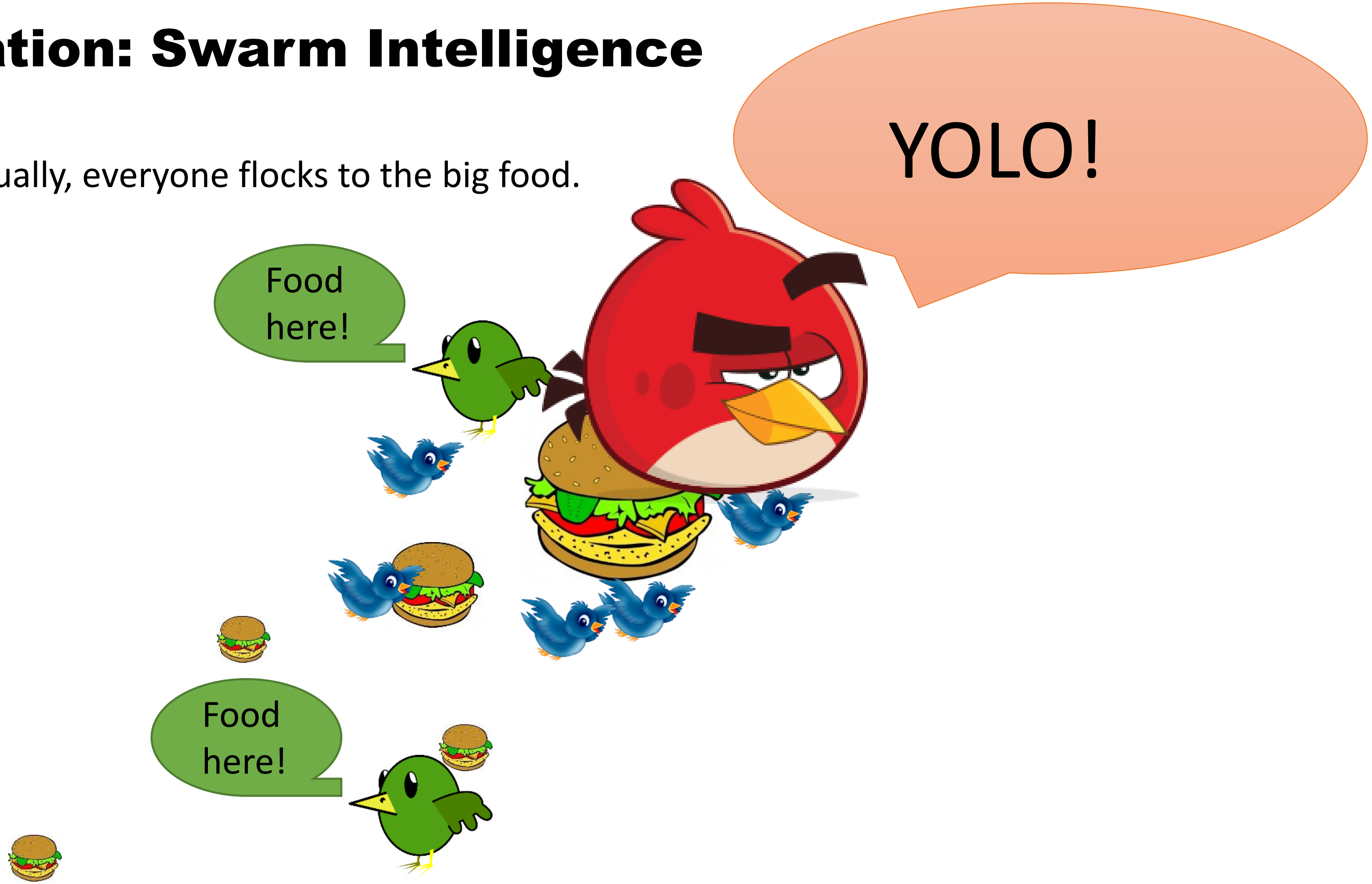
Motivation: Swarm Intelligence

Thus they become fatter and louder.

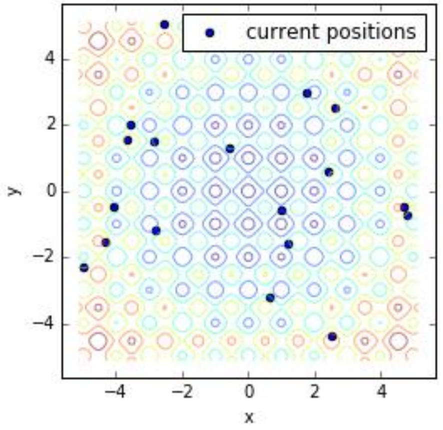


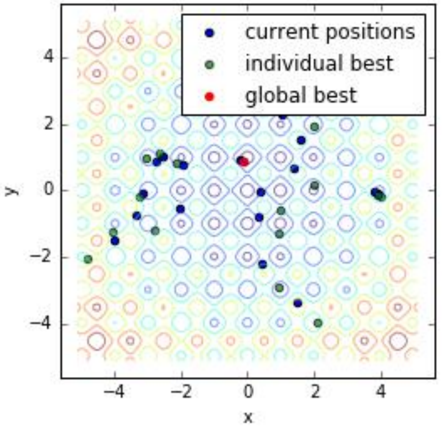
Motivation: Swarm Intelligence

Eventually, everyone flocks to the big food.



Algorithm: Flock to Past Best



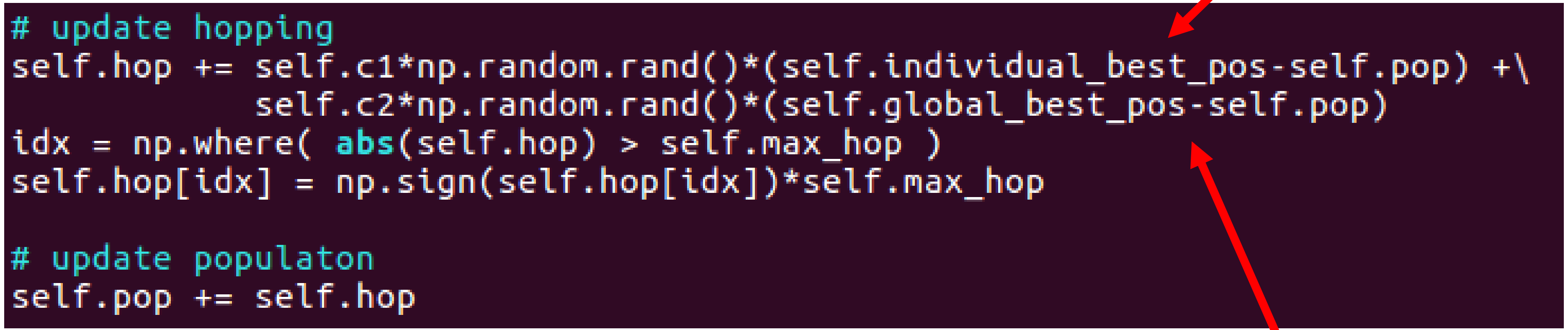


1. Initialize a number of samples from solution space.
2. Before some termination criteria is met:
   1. Evaluate “fitness” of each sample.
   2. Register “individual best” solutions.
   3. Select “global best” solution.
   4. Update each sample according to its individual best, the global best or a linear combination.
   5. Check convergence criteria.

Algorithm: Individual vs. Global Best

- c1 is the degree of individuality of each particle/sample - loner cowboy behavior
- c2 is the degree of submissiveness of each particle/sample - mindless minion behavior

Follow own experience



```
# update hopping
self.hop += self.c1*np.random.rand()*(self.individual_best_pos-self.pop) +\
            self.c2*np.random.rand()*(self.global_best_pos-self.pop)
idx = np.where( abs(self.hop) > self.max_hop )
self.hop[idx] = np.sign(self.hop[idx])*self.max_hop

# update populaton
self.pop += self.hop
```

Follow flock leader

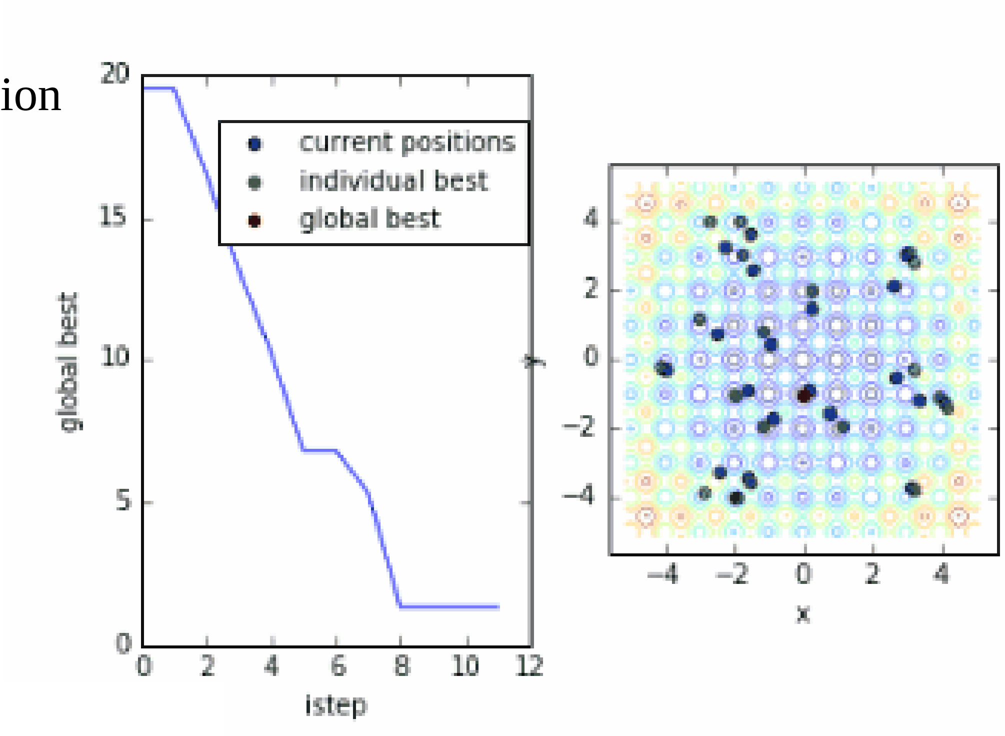
Example: Minimize 2D Rastrigin Function

- The Rastrigin function is multimodal and highly oscillatory function

$$\mathrm{Ras2D}(x,y) = 10\left[2 - \cos(2\pi x) - \cos(2\pi y)\right] + x^2 + y^2$$

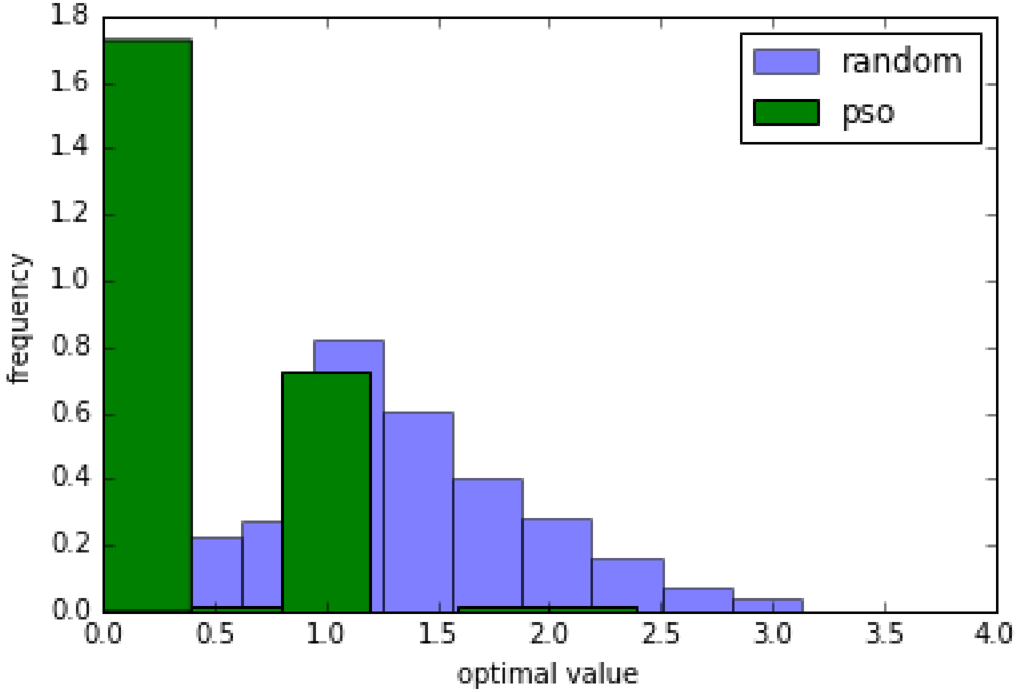
- Global minimum is at (0,0) with a value of 0

- Many local minima surround the global minimum.



Example: Minimize 2D Rastrigin Function

It’s better than random!



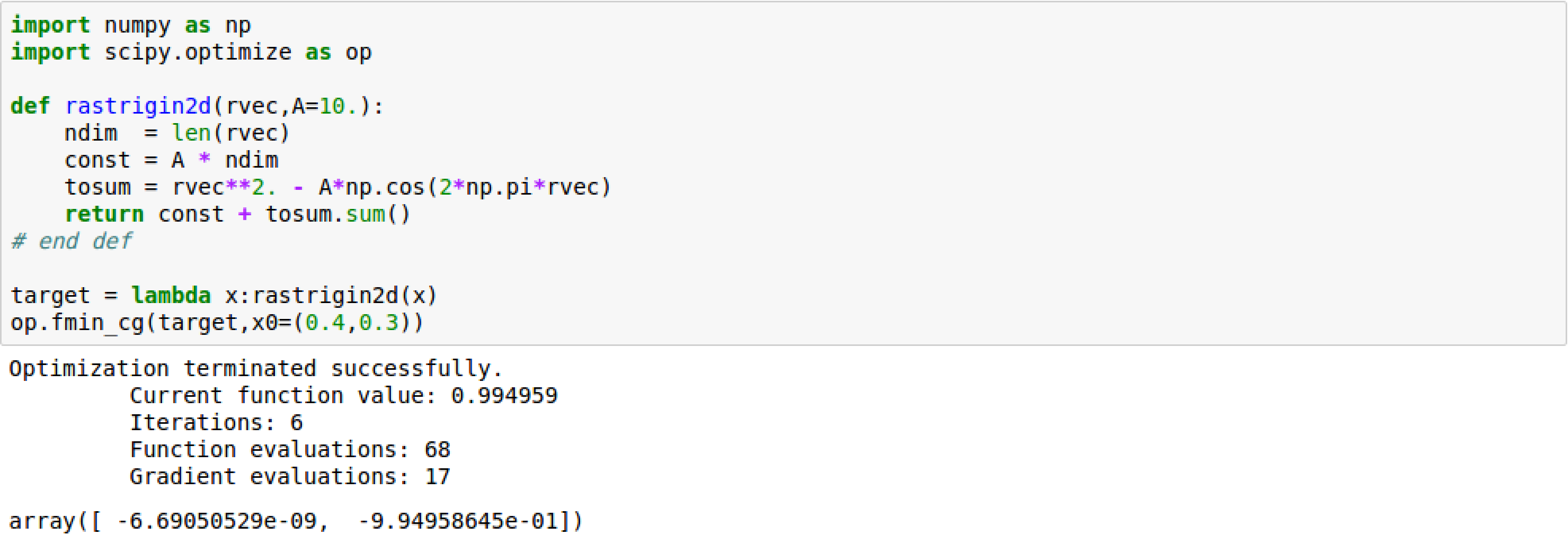
Why Particle Swarm Optimization (PSO) ?

- Easy to implement

- Does not require gradient

- Less likely to get stuck in a local minimum than deterministic algorithms

Example: Conjugate Gradient gets stuck in a local minimum of the 2D Rastrigin function.

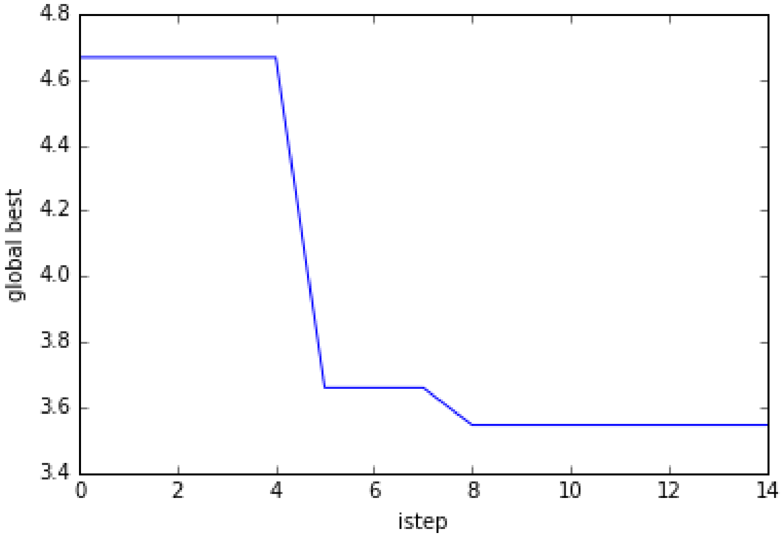


```
import numpy as np
import scipy.optimize as op

def rastrigin2d(rvec,A=10.):
    ndim  = len(rvec)
    const = A * ndim
    tosum = rvec**2. - A*np.cos(2*np.pi*rvec)
    return const + tosum.sum()
# end def

target = lambda x:rastrigin2d(x)
op.fmin_cg(target,x0=(0.4,0.3))
```

```
Optimization terminated successfully.
         Current function value: 0.994959
         Iterations: 6
         Function evaluations: 68
         Gradient evaluations: 17

array([ -6.69050529e-09,  -9.94958645e-01])
```

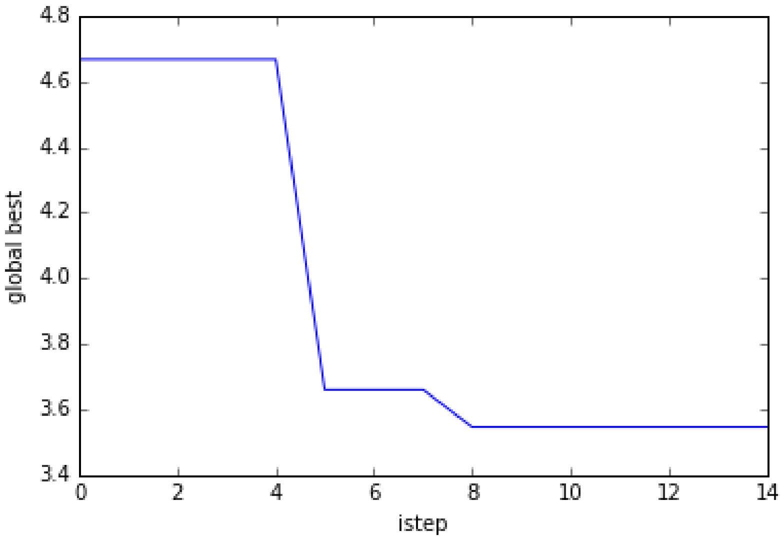
Gotcha! How to Determine Convergence?

Is this converged?

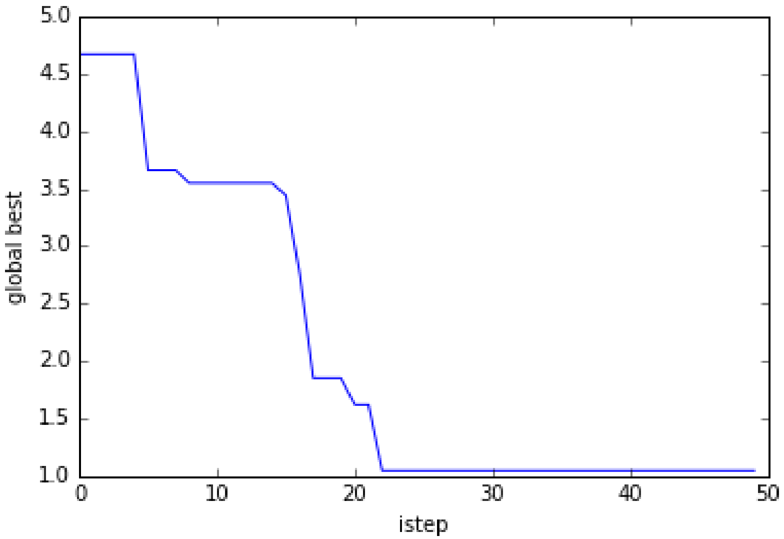


Gotcha! How to Determine Convergence?

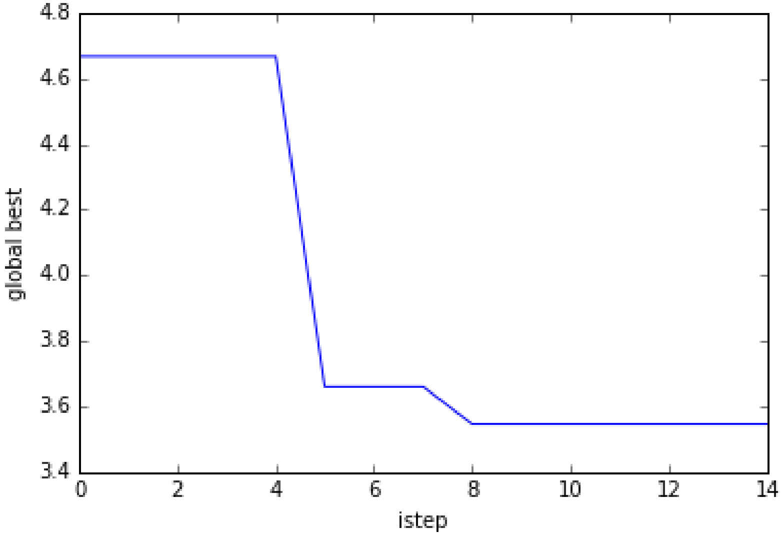
Psyche! No!

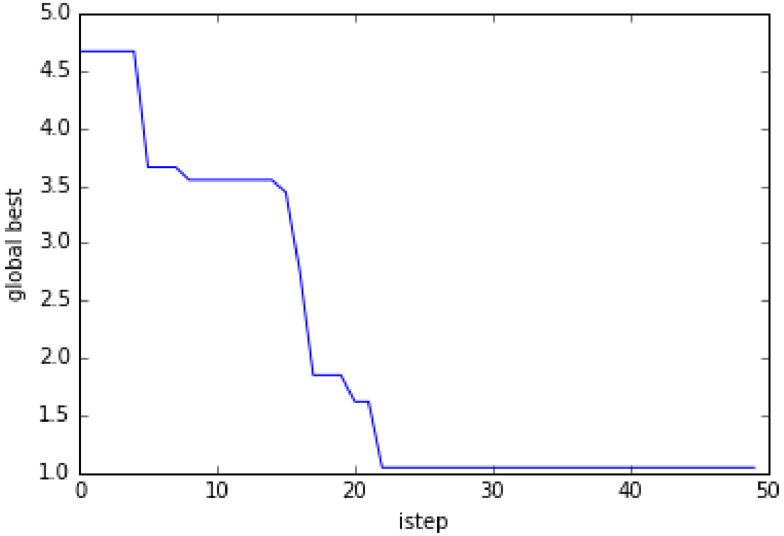


Is this converged?

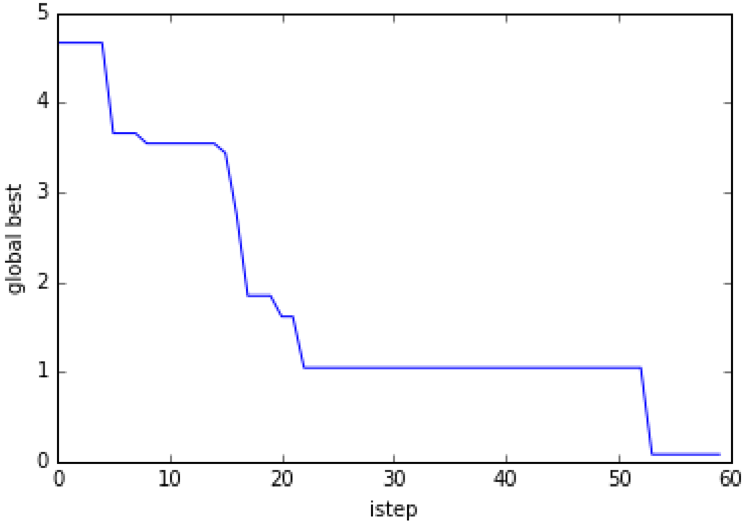


Gotcha! How to Determine Convergence?





Is this converged?

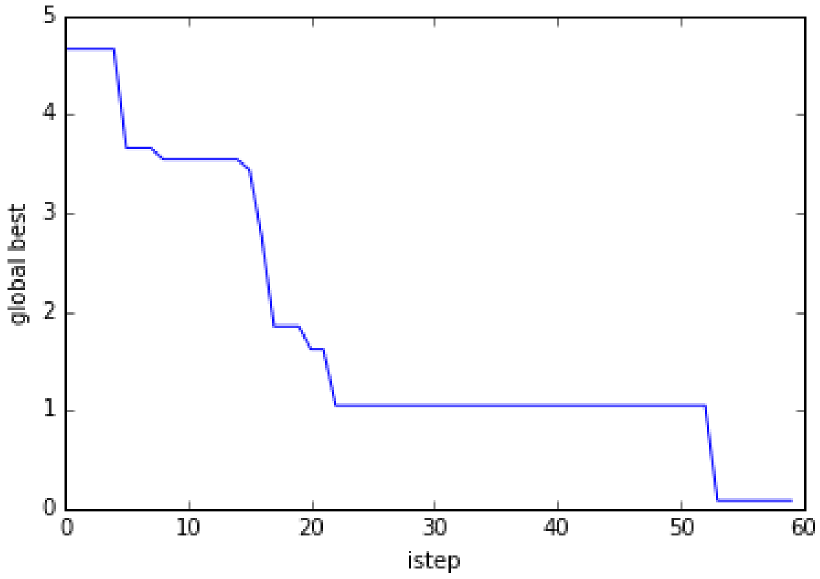


How to Determine Convergence? Sign Test? Hop Trace?

Feed global best trace into a sign test

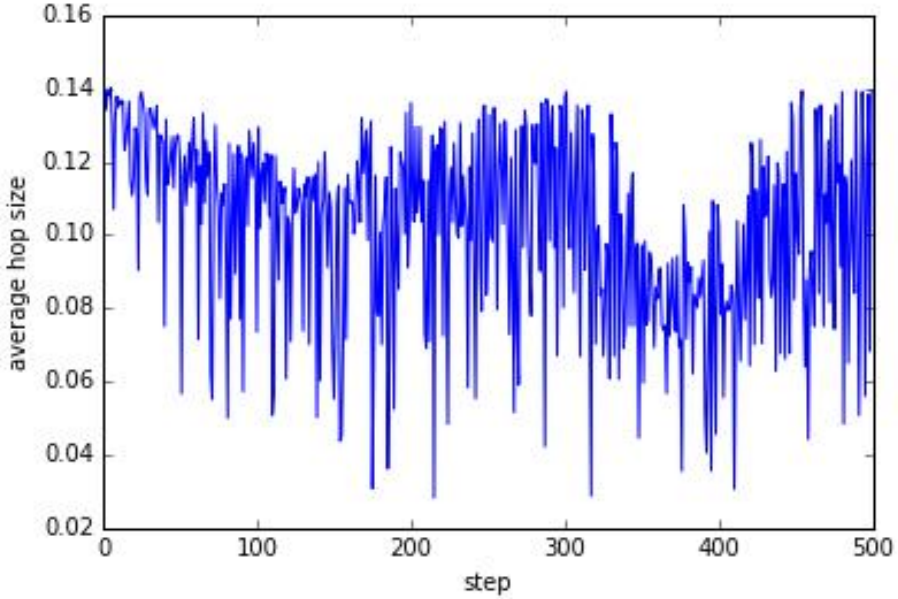
Kwok et. al., IEEE, CEC (2007)

This basically counts the number of times global best is not improved.



Calculate moving correlation for average hop trace

Yang (2016) ?



Application to the Bin Packing Problem

Ingredients in PSO: naïve application

- Population of Solutions ✓ : a collection of greedy solutions
- Individual and Global Best ✓ : highest packing fraction solution
- Hopping update ???? : How to hop “towards” individual or global best?

Application to the Bin Packing Problem

Ingredients in PSO: modified PSO

- Population of Solutions ✓ : a collection of greedy solutions
- Individual and Global Best ✓ : highest packing fraction bin
- Hopping update: Liu et. al., IEEE, CEC (2006)

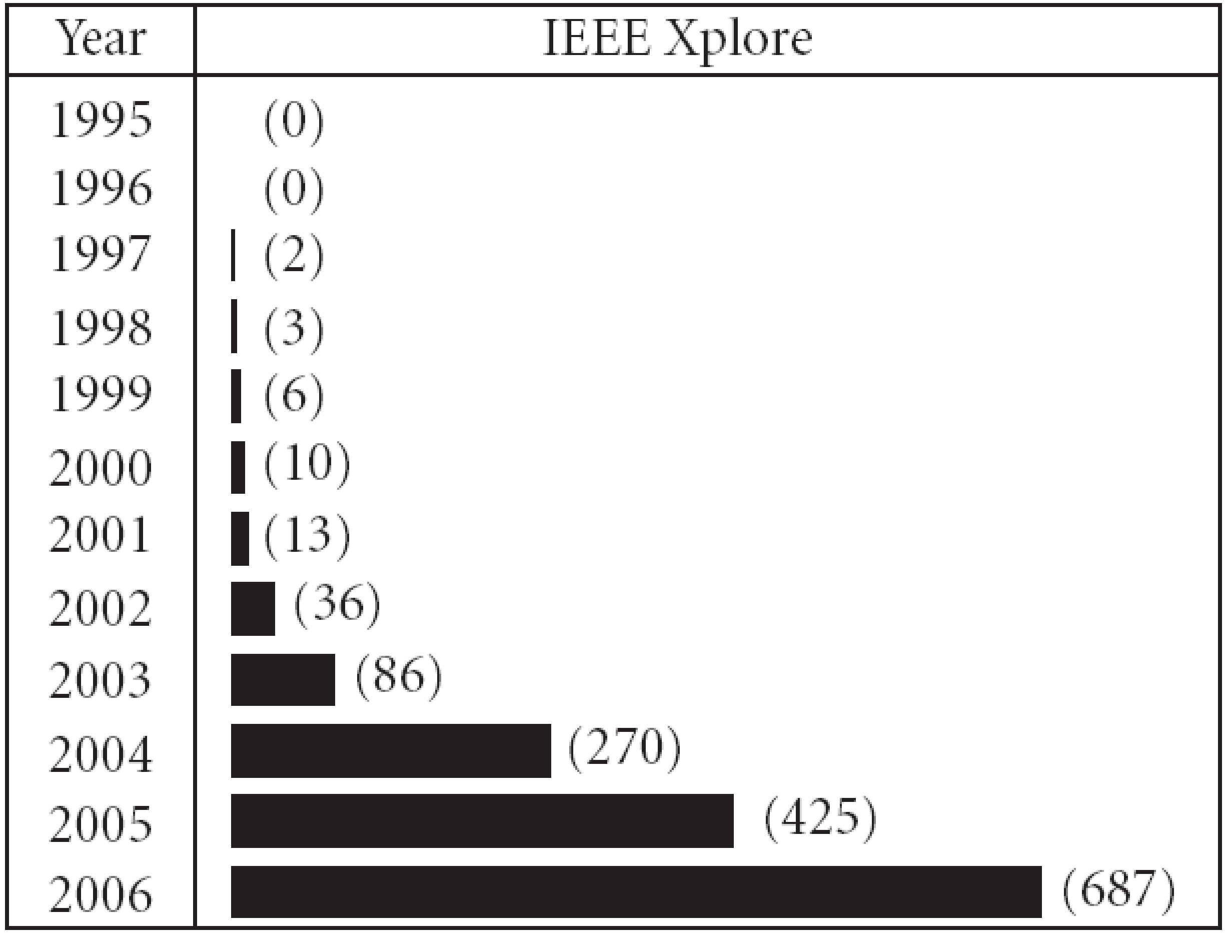
Hop towards individual best: use best bin from personal history
Hop towards global best: use best bin from global history

Application to the Bin Packing Problem

Coming “Soon”

Real World Applications:

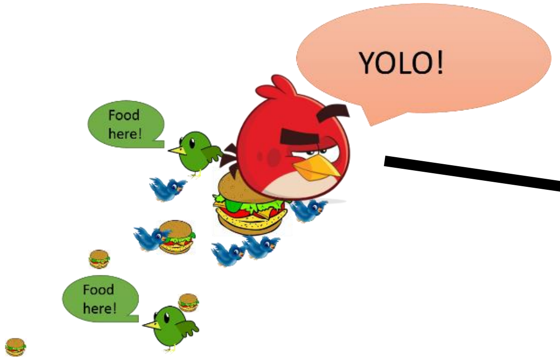
- Antenna Array Design
- Biomedical
- Communication Networks
- Clustering and Classification
- Combinatorial Optimization
- Distribution Networks
- Electronics and Electromagnetics
- Engines and Motors Efficiency Optimization
- Fuzzy and Neurofuzzy: fuzzy control, fuzzy classification
- Graphics and Visualization
- Scheduling



Poli, JAEA, **2008**, 685175 (2008)

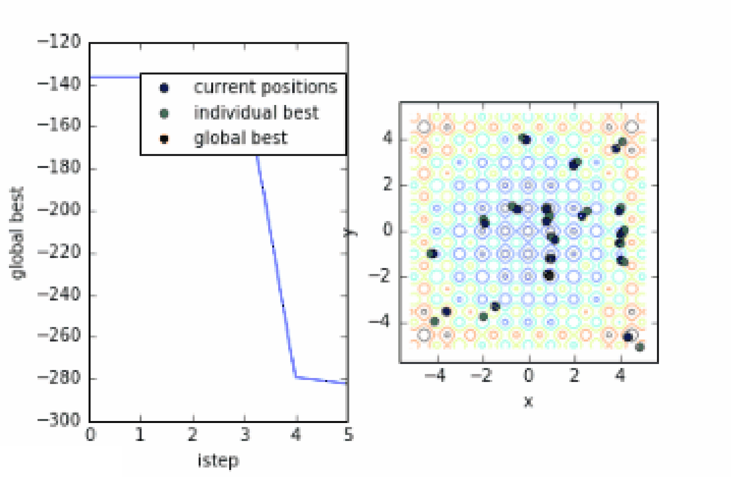
Conclusions:

- PSO is a nature (swarm intelligence) inspired optimization algorithm



nature

algorithm



- PSO is easy to implement, requires no gradient, and tend to get out of local minima

- PSO has many applications and enjoys a rising level of interest