Data Clustering using Particle Swarm Optimization

Augusto Luis Ballardini

ballardini@disco.unimib.it

Agenda

- Standard K-means and main drawbacks
- The Particle Swarm Optimization [PSO] Algorithm
- Appling PSO to clustering
- Hybrid K-means/PSO algorithm
- Matlab example and some results
- References

K-means and drawbacks

- K-means
  - K-Means is one of the most popular clustering algorithms, **<u>and it is easy to implement</u>**
  - It seeks to minimize the sum of squared errors with an iterative optimization
  - At every iteration, it moves the data centroids toward the closer cluster until no point can move anymore

- Drawbacks
  - It implements a Hill-climbing procedure
  - Highly dependent on the choice of K
  - **<u>Sensitive to initialization: how do we choose the initial partitions?</u>**

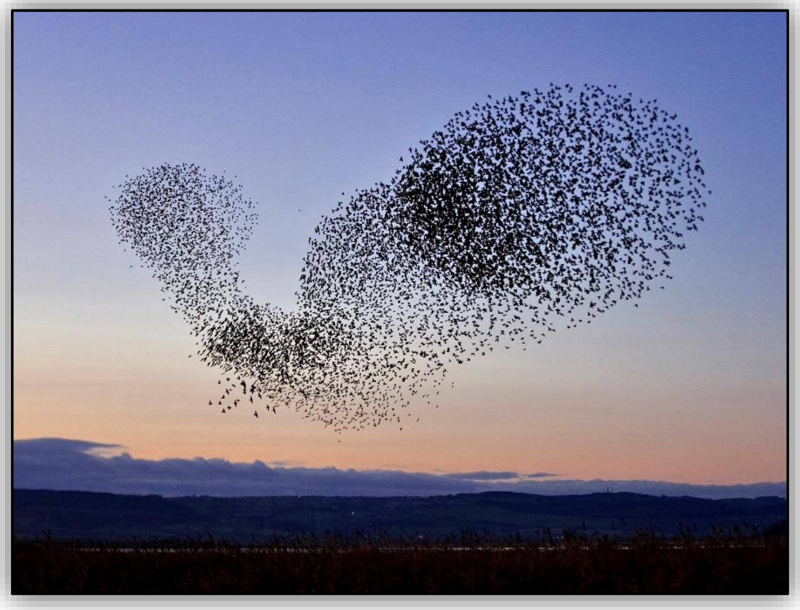
References: Prof. E. Messina, Lesson2 http://www.mind.disco.unimib.it/gallery/index.asp?cat=330&level=2

Particle Swarm Optimization Algorithm

- Optimization Algorithm
  - iterative search process modeled after the social behavior of a bird flock
  - each particle represents a potential solution

- Goal
  - find the "best" particle position i.e. best in the evaluation of a given **fitness (objective) function**

- ***Q: how to cluster using PSO?***



The PSO algorithm code

How to implement the algorithm?

We wrote the full code of PSO using Matlab, <u>in less than 100 lines of code</u>!

The code will be available in my personal page in the IRALAB website
www.ira.disco.unimib.it/people/ballardini-augusto-luis/

Let we see the algorithm pseudo-code

1. Initialize each particle to contain N_c randomly selected cluster centroids.
2. For $t = 1$ to t_{max} do
   (a) For each particle i do
   (b) For each data vector $\mathbf{z}_p$
      i. calculate the Euclidean distance $d(\mathbf{z}_p, \mathbf{m}_{ij}$ to all cluster centroids C_{ij}
      ii. assign $\mathbf{z}_p$ to cluster C_{ij} such that $d(\mathbf{z}_p, \mathbf{m}_{ij}) = \min_{\forall c=1,\cdots,N_c}\{d(\mathbf{z}_p, \mathbf{m}_{ic})\}$
      iii. calculate the fitness using equation (8)
   (c) Update the global best and local best positions
   (d) Update the cluster centroids using equations (3) and (4).

Rules in the PSO algorithm (1)

Each particle *i* mantains:

- x_i, as the *current position* (currently initialized randomly)
- v_i, as the *current velocity* (currently initialized randomly)
- y_i, as the *personal best position*

Each particle's position is adjusted according to

- $v_{i,k}(t+1) = w * v_{i,k}(t) + c_1 * r_{1,k}(t) * (y_{i,k}(t) - x_{i,k}(t)) + c_2 * r_{2,k}(t) * (\hat{y}_{i,k}(t) - x_{i,k}(t))$
- $x_i(t+1) = x_i(t) + v_{i,k}(t+1)$

Rules in the PSO algorithm (2)

- How to calculate the 'best' position?

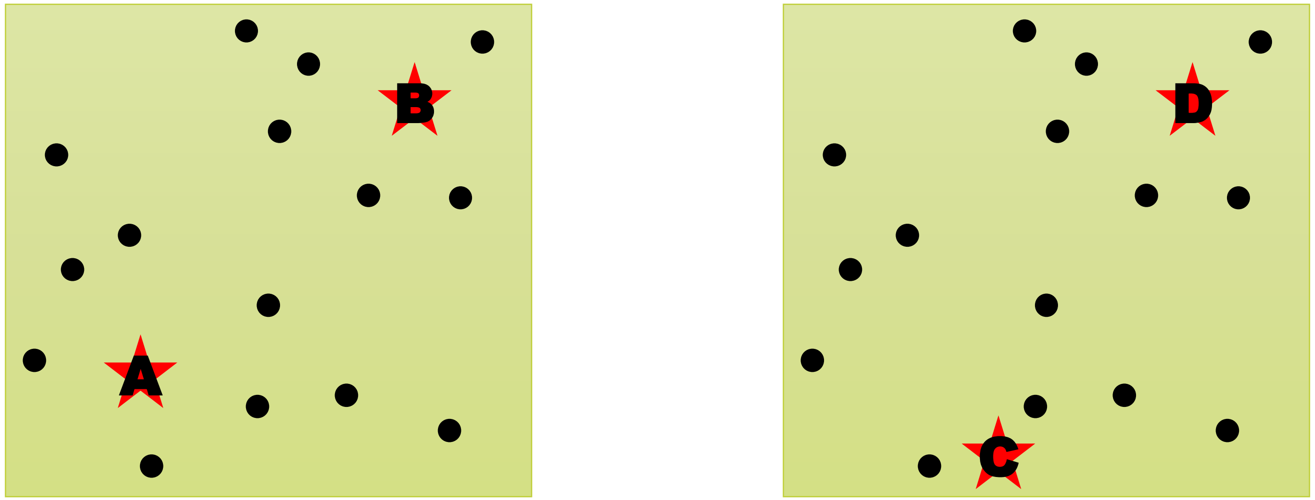
$$\mathbf{y}_i(t+1) = \begin{cases} \mathbf{y}_i(t) & \text{if } f(\mathbf{x}_i(t+1)) \geq f(\mathbf{y}_i(t)) \\ \mathbf{x}_i(t+1) & \text{if } f(\mathbf{x}_i(t+1)) < f(\mathbf{y}_i(t)) \end{cases}$$

- A swarm represents a number of candidate clusterings (cluster centroids) for the input; the fitness of these particles is measured as the quantization error (over all the cluster centroids of the particle)!

$$J_e = \frac{\sum_{j=1}^{N_c}[\sum_{\forall \mathbf{z}_p \in C_{ij}} d(\mathbf{z}_p, \mathbf{m}_j)/|C_{ij}|]}{N_c}$$

Rules (*explained*) in the PSO algorithm

this is the swarm: n=2 particles, each one has k=2 centroids with 2 classes (x and y)



Rules (*explained*) in the PSO algorithm

Position vectors of the swarm

$$\begin{bmatrix} x_a & y_a \\ x_b & y_b \end{bmatrix} \qquad \begin{bmatrix} x_c & y_c \\ x_d & y_d \end{bmatrix}$$

Velocity vectors of the swarm

$$\begin{bmatrix} v_{ax} & v_{ay} \\ v_{bx} & v_{by} \end{bmatrix} \qquad \begin{bmatrix} v_{cx} & v_{cy} \\ v_{dx} & v_{dy} \end{bmatrix}$$

BEST position of the swarm

$$\begin{bmatrix} x_a & y_a \\ x_b & y_b \end{bmatrix} \qquad \begin{bmatrix} x_c & y_c \\ x_d & y_d \end{bmatrix}$$

Fitness Particle 1

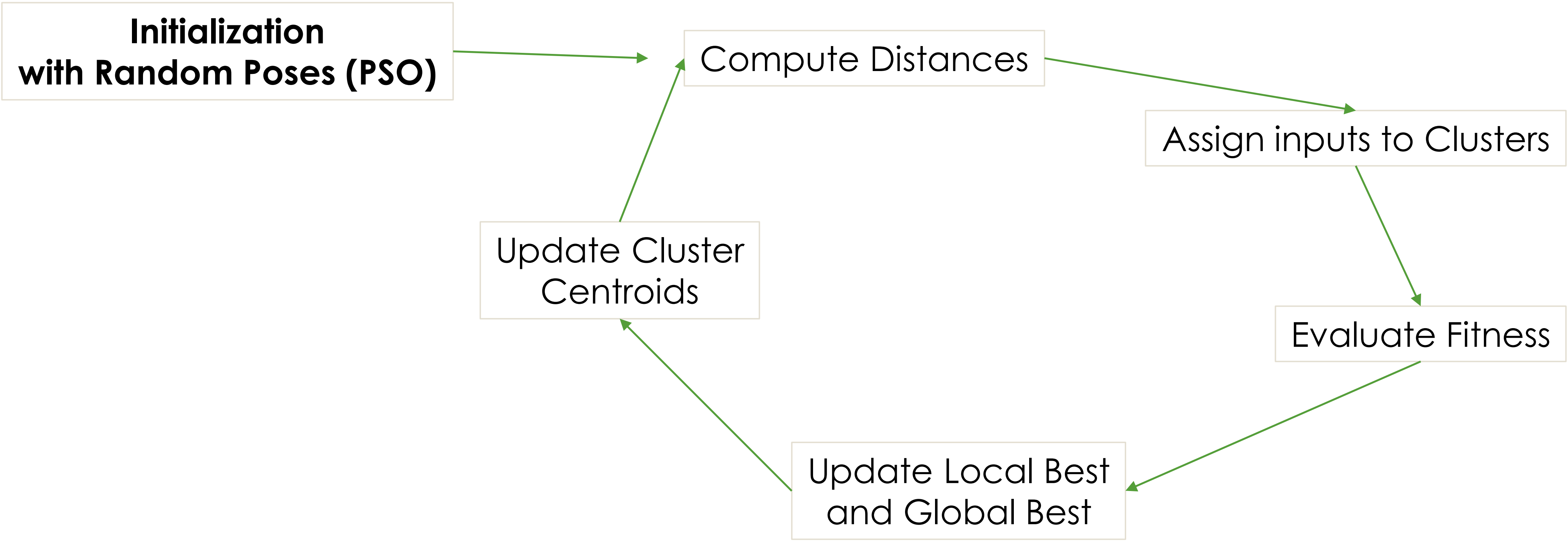
Fitness Particle 2

n particles…

GLOBAL BEST position of the swarm

$$\begin{bmatrix} x_a & y_a \\ x_b & y_b \end{bmatrix}$$

Rules (*explained*) in the PSO algorithm



Rules (*explained*) in the PSO algorithm

How to calculate the position of the particle (already initialized with random values)

Please, consider {w,c1,c2} as constants and {r1,r2} as values sampled from U(0,1)

- *1st calculate new velocity of the particle*

$$v_{i,k}(t+1) = \underbrace{w * v_{i,k}(t)}_{\textit{inertia}} + \underbrace{c_1 * r_{1,k}(t) * (y_{i,k}(t) - x_{i,k}(t))}_{\textit{cognitive component}} + \underbrace{c_2 * r_{2,k}(t) * (\hat{y}_{i,k}(t) - x_{i,k}(t))}_{\textit{social component}}$$

cognitive = distance of the particle from its personal best position
social = distance of the particle from the best particle found so far (i.e. the personal bests)

$v_{i,k}(t)$ and $x_{i,k}(t)$ are *actual velocity / position*
$y_{i,k}(t)$ is the *local best* position
$\hat{y}_{i,k}(t)$ is the *global best* position

Rules (*explained*) in the PSO algorithm

How to calculate the position of the particle (already initialized with random values)

Please, consider {w,c1,c2} as constants and {r1,r2} as values sampled from U(0,1)

- *1st calculate new velocity of the particle*

  $$v_{i,k}(t+1) = \underbrace{w * v_{i,k}(t)}_{\textit{inertia}} + \underbrace{c_1 * r_{1,k}(t) * (y_{i,k}(t) - x_{i,k}(t))}_{\textit{cognitive component}} + \underbrace{c_2 * r_{2,k}(t) * (\hat{y}_{i,k}(t) - x_{i,k}(t))}_{\textit{social component}}$$

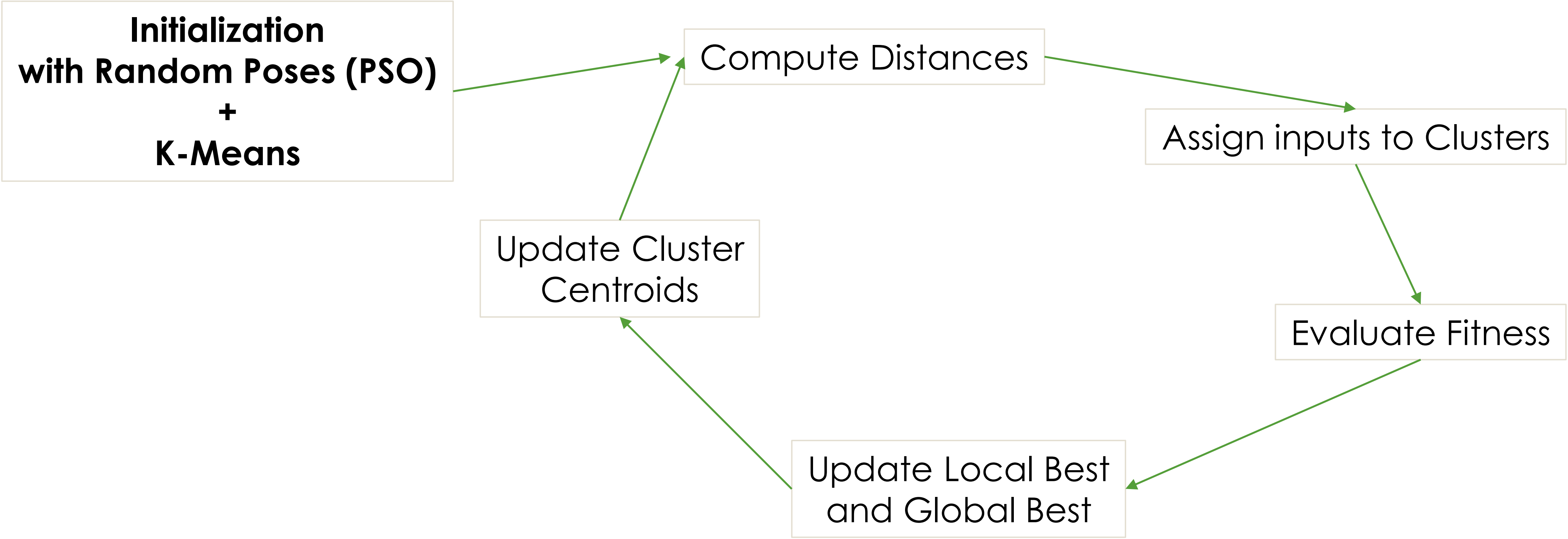
- *2nd calculate the new position of the particle*

  $$x_i(t+1) = x_i(t) + v_{i,k}(t+1)$$

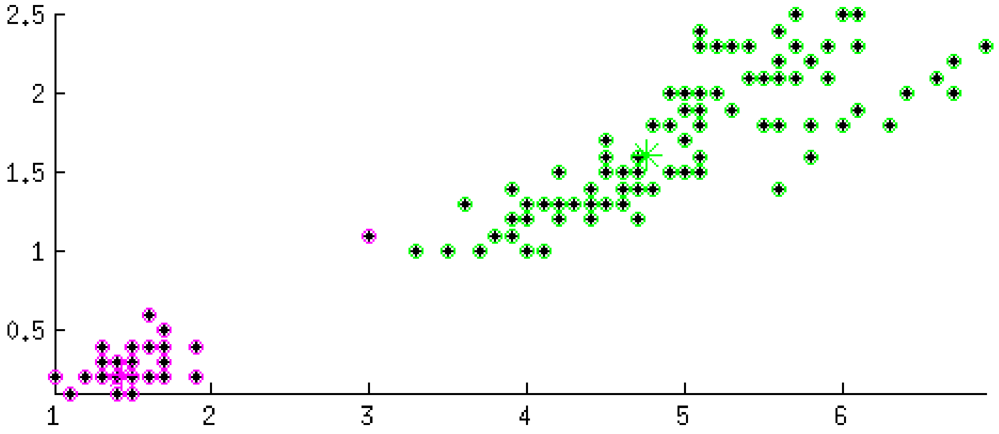
Summarizing and going further!

- The algorithm still needs a K value
- The population-based search of the PSO algorithm reduces the effect that initial conditions has, as opposed to the K-Means algorithm the search starts from multiple positions in parallel
- K-means algorithm tends to converge faster (after less evaluations) than the PSO, but usually with a less accurate clustering [4]
- The performance of PSO can further be improved by seeding the initial swarm with the result of the K-Means algorithm (used as one of the particles, while the rest of the swarm is initialized randomly).
  This is know as Hybrid PSO and K-Means Clustering Algorithm [1]

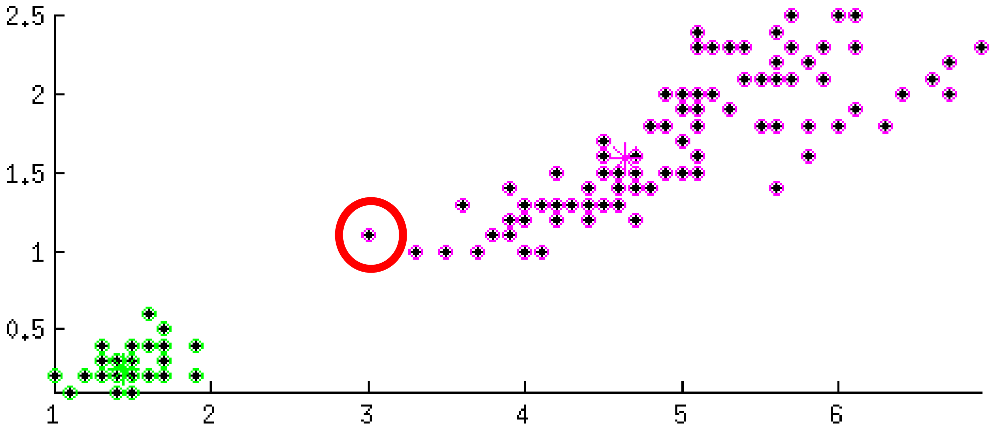
Rules (*explained*) in the PSO algorithm



A *static* example



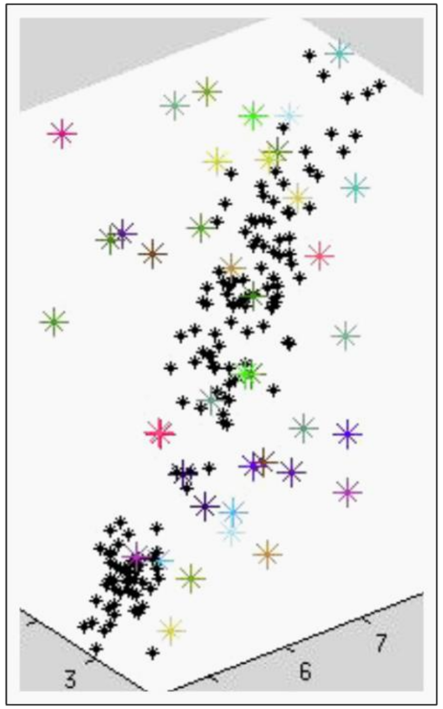
Resulting clustering with only k-means

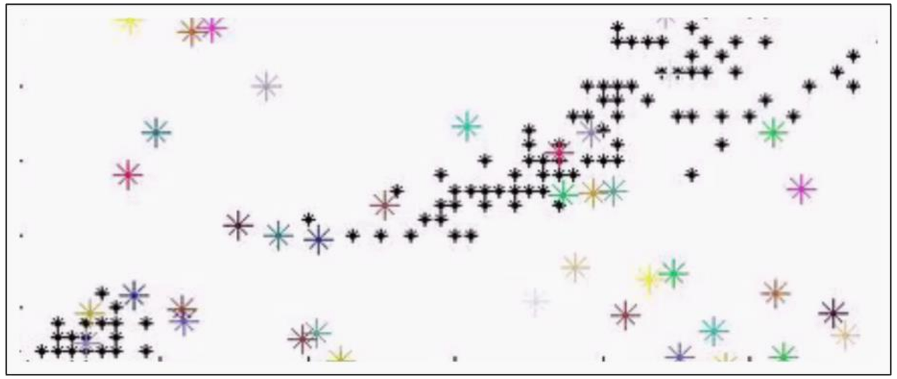


Resulting clustering using PSO with k-means initialization.

Look at the blinking circle!

A *dynamic* example





References

- [1] Van Der Merwe, D. W.; Engelbrecht, AP., "Data clustering using particle swarm optimization," *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on* , vol.1, no., pp.215,220 Vol.1, 8-12 Dec. 2003
- [2] J Kennedy, RC Eberhart, "Particle Swarm Optimization", Proceedings of the IEEE International Joint Conference on Neural Networks, Vol. 4, pp 1942-1948, 1995.
- [3] J Kennedy, RC Eberhart, Y Shi, "Swarm Intelligence", Morgan Kaufmann, 2002.
- [4] M Omran, A Salman, AP Engelbrecht, "Image Classification using Particle Swarm Optimisation", Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning, Singapore, 2002.