Inatel

Otimização por Enxame de Partículas

Disciplina: Inteligência Computacional (C210A/B)

Curso: Engenharia de Computação e Software

Profª. Victoria Dala Pegorara Souto

Otimização por Enxame de Partículas (PSO)

- Promover a compreensão do funcionamento e da aplicação da técnica de Otimização por Enxame de Partículas (PSO) na solução de problemas de otimização.

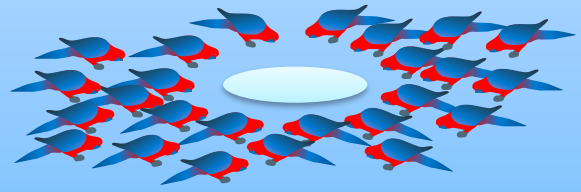
Otimização por Enxame de Partículas (PSO)

- Paradigma da IA

Evolucionista

Metáfora da natureza

Ex. algoritmos genéticos, vida artificial,...



Otimização por Enxame de Partículas (PSO)

- Se os pássaros competem por comida, por que eles indiretamente avisam aos outros quando a encontra?

Porque a convivência de indivíduos não egoístas em colônias, traz, para cada indivíduo, mais vantagens do que desvantagens.

Otimização por Enxame de Partículas (PSO)

- Proposto em 1995 por Kenedy e Eberhart.
- Baseado no comportamento social de algumas espécies animais, sendo inspirado na capacidade de tais espécies trabalharem de forma coletiva na localização de alimentos, esta capacidade pode ser verificada em cardumes de peixes, bandos de pássaros e enxames de abelhas.
- Classificado como Método Evolucionário, apesar de não ser baseado no conceito de seleção natural, pois utiliza a estratégia de colaboração para evoluir. Tal classificação é baseada no fato do PSO fazer uso de populações de soluções individuais do problema e de apresentar aspectos semelhantes aos operadores evolucionários.
- Método de otimização estocástico, o qual demonstra-se eficaz na otimização de funções não-lineares de alta dimensionalidade e com variáveis contínuas.

Otimização por Enxame de Partículas (PSO)

- **Vantagens:**
  - Poucos parâmetros a serem ajustados;
  - Uma única versão com pequenas variações apresenta bons resultados para uma variedade de aplicações;
  - Não é totalmente aleatório: possui componentes aleatórios mas os próximos passos são baseados em informações da população atual;
  - Não é afetado por descontinuidades: não utiliza informações de derivadas na atualização de velocidades e posições.

Otimização por Enxame de Partículas (PSO)

- Versões:
  - Nearest Neighbor Velocity Matching (NNVM);
  - The Cornfield Vector
  - Canonical PSO.

Otimização por Enxame de Partículas (PSO)

Terminologia:

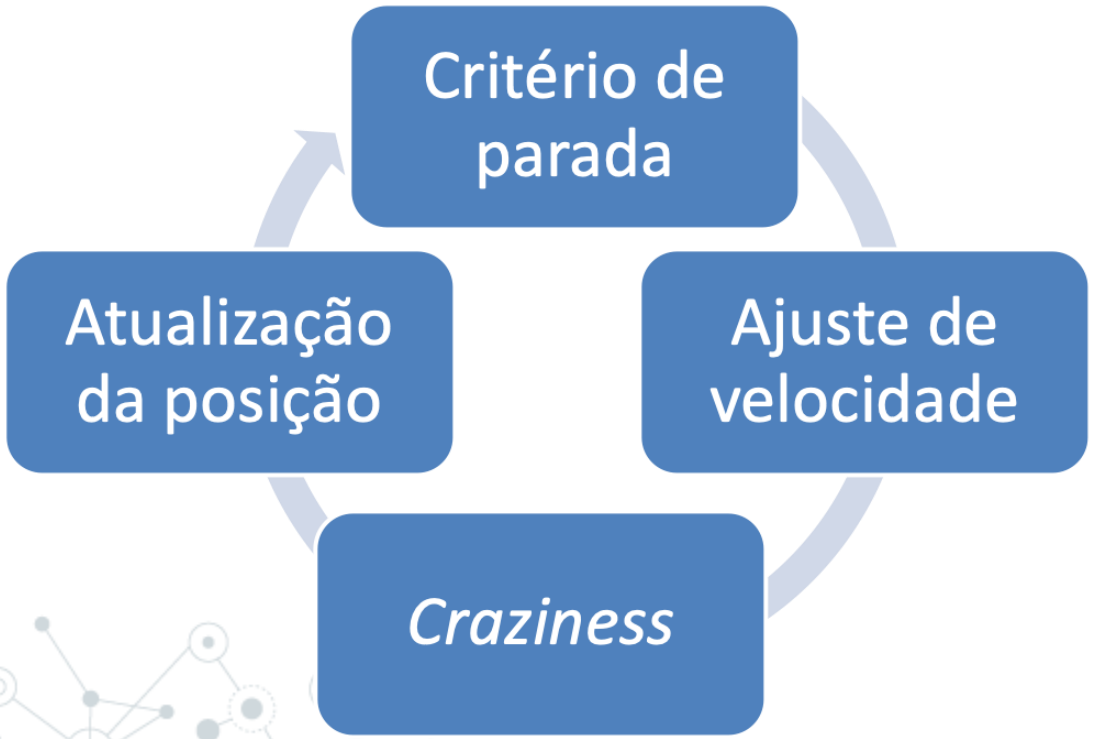
- **Partícula:** Cada **indivíduo do enxame** é chamado de partícula e representa uma **possível solução para o problema** avaliado. Cada partícula atua isoladamente, **atualizando a sua posição e velocidade** para alcançar o seu objetivo.
- **Enxame:** Representa um **grupo de partículas**, o qual tem seu tamanho definido durante a implementação do PSO.
- **Posição:** Representa a **posição da partícula no espaço de busca** do enxame. A posição da partícula é representada em um plano cartesiano (x, y), **os valores limites deste plano são definidos para cada problema a ser otimizado**. Estes valores determinam o espaço de busca do método proposto.
- **Aptidão:** Valor **calculado através da posição de cada partícula**. Este valor é definido através de uma função aptidão, a qual é definida com o objetivo de avaliar a posição de cada partícula, verificando o quão próxima da melhor posição encontra-se cada partícula.
- **Espaço de Busca:** Representa as **posições permitidas para o enxame**, ou seja, como cada posição é representada em um plano cartesiano (x, y). O espaço de busca representa os valores máximo e mínimo de x e y.

Otimização por Enxame de Partículas (PSO)

- ***Nearest Neighbor Velocity Matching***
  - A ideia era simular geograficamente a coreografia do voo do bando de pássaros de forma “graciosa” e “imprevisível”.
  - A simulação inicial foi baseada na “correspondência de velocidade do vizinho mais próximo” e na variável chamada “loucura” (*craziness*).
  - O bando de pássaros voava para uma direção imutável, o que necessitou da introdução de uma variável estocástica denominada “loucura” que causava uma variação aleatória na velocidade e direção de algum pássaro aleatório.

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*



- **Ajuste de velocidade:** ajusta-se a velocidade de cada partícula baseado na partícula mais próxima;
- ***Craziness***: seleciona(m)-se alguma(s) partícula(s) de forma aleatória e altera suas respectivas velocidades de forma aleatória;
- **Atualização da posição:** atualiza-se a posição de cada partícula com base na nova velocidade obtida após a atualização das velocidades;
- **Critério de parada:** verifica-se se o critério de parada é satisfeito. Pode ser por número de iterações, tempo de execução, qualidade da solução, variação da qualidade da solução entre iterações consecutivas, etc.

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

1	Pos.:			
	Vel.:			

3	Pos.:			
	Vel.:			

2	Pos.:			
	Vel.:			

4	Pos.:			
	Vel.:			

- **Problema:** simular e compreender a dinâmica de movimento de um conjunto de indivíduos (partículas).

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

1	Pos.:	+0,8	+0,1	-0,3
	Vel.:	+0,1	-0,5	-0,8

3	Pos.:	-0,7	-0,6	+0,7
	Vel.:	+0,1	-0,1	+0,8

População Inicial

2	Pos.:	+0,4	+0,7	+0,3
	Vel.:	+0,5	+0,8	-0,7

4	Pos.:	+0,1	+0,4	+0,9
	Vel.:	-0,2	+0,6	+1,0

- **População inicial**: neste exemplo foram criadas 4 partículas inicializadas aleatoriamente, cada uma possuindo os vetores:
  - *Pos.* (corresponde a sua posição no espaço tridimensional em análise)
  - *Vel.* (corresponde à velocidade vetorial do movimento da partícula)

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

1	Pos.:	+0,8	+0,1	-0,3
	Vel.:	+0,1	-0,5	-0,8

3	Pos.:	-0,7	-0,6	+0,7
	Vel.:	+0,1	-0,1	+0,8

Ajuste de Velocidade

2	Pos.:	+0,4	+0,7	+0,3
	Vel.:	+0,5	+0,8	-0,7

4	Pos.:	+0,1	+0,4	+0,9
	Vel.:	-0,2	+0,6	+1,0

- **Ajuste de velocidade:** para ajustar a velocidade de cada partícula, o primeiro passo é encontrar qual é a partícula mais próxima. Para isto, pode ser usada a distância euclidiana.

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*

Ajuste de Velocidade

- Considere dois pontos P e Q em um espaço tridimensional:

$$P = (p_x, p_y, p_z) \text{ e } Q = (q_x, q_y, q_z)$$

- A distância euclidiana entre eles é dada por:

$$dist(P, Q) = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2 + (q_z - p_z)^2}$$

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*

Calculo da Distância Euclidiana entre as partículas.

- Distância euclidiana entre a partícula 1 e as demais partículas:

$dist(P_1, P_2) = 0{,}94\ (Menor\ Distância)$
$dist(P_1, P_3) = 1{,}93$
$dist(P_1, P_4) = 1{,}42$

- Distância euclidiana entre a partícula 2 e as demais partículas:

$dist(P_2, P_1) = 0{,}94$
$dist(P_2, P_3) = 1{,}75$
$dist(P_2, P_4) = 0{,}73\ (Menor\ Distância)$

- Distância euclidiana entre a partícula 3 e as demais partículas:

$dist(P_3, P_1) = 1{,}93$
$dist(P_3, P_2) = 1{,}75$
$dist(P_3, P_4) = 1{,}30\ (Menor\ Distância)$

- Distância euclidiana entre a partícula 4 e as demais partículas:

$dist(P_4, P_1) = 1{,}42$
$dist(P_4, P_2) = 0{,}73\ (Menor\ Distância)$
$dist(P_4, P_3) = 1{,}30$

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

- **Ajuste de velocidade:** Copia-se a velocidade da partícula mais próxima. **Obs.:** isto deve ser feito simultaneamente (ou seja, assume-se que todas as velocidades são copiadas instantaneamente, sem ordem ou precedência).

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

Partícula mais próxima de 1: **2**

1	Pos.:	+0,8	+0,1	-0,3
	Vel.:	+0,5	+0,8	-0,7

Partícula mais próxima de 3: **4**

3	Pos.:	-0,7	-0,6	+0,7
	Vel.:	-0,2	+0,6	+1,0

Ajuste de Velocidade

Partícula mais próxima de 2: **4**

2	Pos.:	+0,4	+0,7	+0,3
	Vel.:	-0,2	+0,6	+1,0

Partícula mais próxima de 4: **2**

4	Pos.:	+0,1	+0,4	+0,9
	Vel.:	+0,5	+0,8	-0,7

- **Ajuste de velocidade:** Copia-se a velocidade da partícula mais próxima. **Obs.:** isto deve ser feito **simultaneamente** (ou seja, assume-se que todas as velocidades são copiadas instantaneamente, sem ordem ou precedência).

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

1	Pos.:	+0,8	+0,1	-0,3
	Vel.:	+0,5	+0,8	-0,7

3	Pos.:	-0,7	-0,6	+0,7
	Vel.:	-0,2	+0,6	+1,0

Craziness

2	Pos.:	+0,4	+0,7	+0,3
	Vel.:	-0,2	+0,6	+1,0

4	Pos.:	+0,1	+0,4	+0,9
	Vel.:	+0,5	+0,8	-0,7

- ***Craziness***: com base em uma probabilidade pequena, alguma(s) partícula(s) altera(m) sua velocidade de forma aleatória.

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

Partícula escolhida aleatoriamente

1	Pos.:	+0,8	+0,1	-0,3
	Vel.:	+0,5	+0,8	-0,7

3	Pos.:	-0,7	-0,6	+0,7
	Vel.:	-0,2	+0,6	+1,0

Craziness

2	Pos.:	+0,4	+0,7	+0,3
	Vel.:	-0,2	+0,6	+1,0

4	Pos.:	+0,1	+0,4	+0,9
	Vel.:	+0,5	+0,8	-0,7

- ***Craziness***: com base em uma probabilidade pequena, alguma(s) partícula(s) altera(m) sua velocidade de forma aleatória.

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

1

Pos.:	+0,8	+0,1	-0,3
Vel.:	+0,5	+0,8	-0,7

3

Pos.:	-0,7	-0,6	+0,7
Vel.:	-0,5	+0,2	+0,8

Velocidade alterada aleatoriamente

Craziness

2

Pos.:	+0,4	+0,7	+0,3
Vel.:	-0,2	+0,6	+1,0

4

Pos.:	+0,1	+0,4	+0,9
Vel.:	+0,5	+0,8	-0,7

- ***Craziness***: com base em uma probabilidade pequena, alguma(s) partícula(s) altera(m) sua velocidade de forma aleatória.

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

1

Pos.:	+0,8	+0,1	-0,3
Vel.:	+0,5	+0,8	-0,7

+

3

Pos.:	-0,7	-0,6	+0,7
Vel.:	-0,5	+0,2	+0,8

+

Atualiza a Posição

2

Pos.:	+0,4	+0,7	+0,3
Vel.:	-0,2	+0,6	+1,0

+

4

Pos.:	+0,1	+0,4	+0,9
Vel.:	+0,5	+0,8	-0,7

+

- **Atualização da posição:** cada partícula tem sua posição atualizada com base no vetor de velocidades ajustado anteriormente.

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*

Atualiza a posição de cada partícula

Atualização da Posição da Partícula *i* é dada por:

$$Pos_i(t+1) = Pos_i(t) + vel_i(t)$$

- **Partícula 1:**
  $Pos_1(t+1) = [0{,}8 \quad 0{,}1 \quad -0{,}3] + [0{,}5 \quad 0{,}8 \quad -0{,}7] = [1{,}3 \quad 0{,}9 \quad -0{,}9]$
- **Partícula 2:**
  $Pos_2(t+1) = [0{,}4 \quad 0{,}7 \quad 0{,}3] + [-0{,}2 \quad 0{,}6 \quad 1{,}0] = [0{,}2 \quad 1{,}3 \quad 1{,}2]$
- **Partícula 3:**
  $Pos_3(t+1) = [-0{,}7 \quad -0{,}6 \quad 0{,}7] + [-0{,}5 \quad 0{,}2 \quad 0{,}8] = [-1{,}2 \quad -0{,}4 \quad 1{,}5]$
- **Partícula 4:**
  $Pos_4(t+1) = [0{,}1 \quad 0{,}4 \quad 0{,}9] + [-0{,}5 \quad 0{,}8 \quad -0{,}7] = [0{,}6 \quad 1{,}2 \quad 0{,}3]$

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

1	Pos.:	+0,8	+0,1	-0,3
	Vel.:	+0,5	+0,8	-0,7

+

3	Pos.:	-0,7	-0,6	+0,7
	Vel.:	-0,5	+0,2	+0,8

+

Atualiza a Posição

2	Pos.:	+0,4	+0,7	+0,3
	Vel.:	-0,2	+0,6	+1,0

+

4	Pos.:	+0,1	+0,4	+0,9
	Vel.:	+0,5	+0,8	-0,7

+

- **Atualização da posição:** cada partícula tem sua posição atualizada com base no vetor de velocidades ajustado anteriormente.

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**

1

Pos.:	+1,3	+0,9	-0,9
Vel.:	+0,5	+0,8	-0,7

3

Pos.:	-1,2	-0,4	+1,5
Vel.:	-0,5	+0,2	+0,8

Atualiza a Posição

2

Pos.:	+0,2	+1,3	+1,2
Vel.:	-0,2	+0,6	+1,0

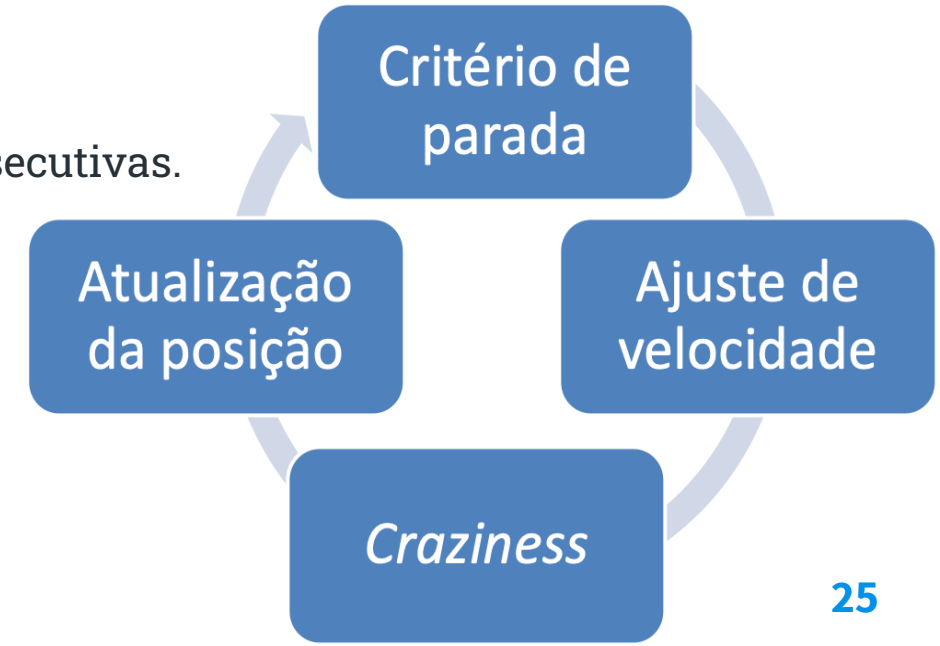
4

Pos.:	+0,6	+1,2	+0,3
Vel.:	+0,5	+0,8	-0,7

- **Atualização da posição:** cada partícula tem sua posição atualizada com base no vetor de velocidades ajustado anteriormente.

Otimização por Enxame de Partículas (PSO)

- *Nearest Neighbor Velocity Matching*
  - **Exemplo:**
    - **Os passos são repetidos até que o critério de parada seja atingido:**
      - Número de iterações;
      - Tempo de execução;
      - Qualidade da solução;
      - Variação da qualidade da solução entre iterações consecutivas.



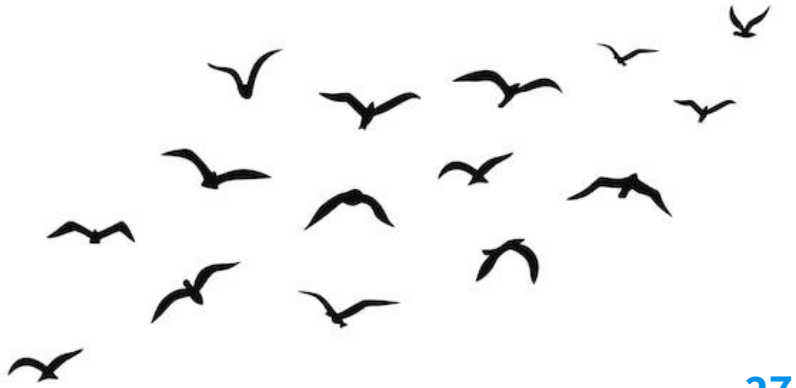
Otimização por Enxame de Partículas (PSO)

The Cornfield Vector

- Foi introduzida a figura do **"poleiro"** (***roost***), uma posição que atraía os pássaros até que eles finalmente pousassem lá. Estabeleceu-se, então, a figura do **"objetivo".**
- **Com a introdução do "poleiro", não havia mais necessidade da variável "loucura" (*craziness*), já que a simulação "assumia vida própria".**
- Também foram introduzidos os termos ***"pbest"*** e **"gbest"**, uma espécie de memória sobre as melhores posições pessoais (de cada indivíduo) e global (entre todos indivíduos).
- Desta forma, a dinâmica dos membros de um bando lhes permitia **aprender partindo do conhecimento do outro.**

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Terminologia**
    - **pbest:** Representa a melhor posição experimentada pela partícula até o momento. Este parâmetro representa a memória individual de cada partícula do enxame.
    - **gbest:** Representa a melhor posição experimentada pelo enxame. Tal parâmetro representa a memória coletiva do enxame. Todas as partículas do enxame possuem o mesmo gbest, diferentemente do pbest, o qual cada partícula possui o seu.



Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*

- **Avaliação da população**: módulo onde é avaliado cada partícula da população, atualizando os valores pbest e gbest usados nas etapas posteriores.
- **Ajuste de velocidade (pessoal/cognitivo):** atualiza-se a velocidade de cada partícula com base na posição passada em que ela esteve com a melhor avaliação em relação ao objetivo (pbest);
- **Ajuste de velocidade (global/social):** atualiza-se a velocidade de cada partícula com base na partícula que retornou a melhor avaliação em relação ao objetivo (gbest);
- **Atualização da posição:** atualiza-se a posição de cada partícula com base na nova velocidade obtida após as atualizações das velocidades;
- **Critério de parada:** verifica-se se o critério de parada é satisfeito. Pode ser por número de gerações, qualidade da solução, variação da qualidade da solução, etc.

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

Objetivo		
+1,2	+3,7	-0,9

Problema: um objetivo (ou seja, uma posição em um espaço tridimensional) foi criado para exemplificar o funcionamento do algoritmo.

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

- **População inicial**: neste exemplo foram criadas 4 partículas inicializadas aleatoriamente, cada uma possuindo os vetores:

Pos. (corresponde a sua posição no espaço tridimensional em análise)
Vel. (corresponde à velocidade vetorial do movimento da partícula)

1

Pos.:	+0,8	+0,1	-0,3
Vel.:	+0,1	-0,5	-0,8

3

Pos.:	-0,7	-0,6	+0,7
Vel.:	+0,1	-0,1	+0,8

Objetivo		
+1,2	+3,7	-0,9

2

Pos.:	+0,4	+0,7	+0,3
Vel.:	+0,5	+0,8	-0,7

4

Pos.:	+0,1	+0,4	+0,9
Vel.:	-0,2	+0,6	+1,0

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

- **População inicial**: também são introduzidas as seguintes variáveis:

pbest (melhor posição, bem como a distância até o objetivo, que **um indivíduo em particular** do grupo já esteve ao longo das iterações)

gbest (melhor posição, bem como a distância até o objetivo, que **a melhor partícula da população** já esteve esteve ao longo das iterações)

gbest		
dist = ?		
?	?	?

1

pbest	dist = ?		
	?	?	?
Pos.:	+0,8	+0,1	-0,3
Vel.:	+0,1	-0,5	-0,8

3

pbest	dist = ?		
	?	?	?
Pos.:	-0,7	-0,6	+0,7
Vel.:	+0,1	-0,1	+0,8

Objetivo		
+1,2	+3,7	-0,9

2

pbest	dist = ?		
	?	?	?
Pos.:	+0,4	+0,7	+0,3
Vel.:	+0,5	+0,8	-0,7

População Inicial

4

pbest	dist = ?		
	?	?	?
Pos.:	+0,1	+0,4	+0,9
Vel.:	-0,2	+0,6	+1,0

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

- Nesta etapa, deve-se comparar a posição atual de cada partícula com sua melhor posição individual até o momento **(pbest)** e, se for melhor, deve ser atualizada.
- Também, deve ser encontrada a melhor posição dentre todos os indivíduos **(gbest)**.

pbest e gbest serão usadas em etapas posteriores (nos ajustes de velocidade).

gbest		
dist = ?		
?	?	?

1

pbest	dist = ?		
	?	?	?
Pos.:	+0,8	+0,1	-0,3
Vel.:	+0,1	-0,5	-0,8

3

pbest	dist = ?		
	?	?	?
Pos.:	-0,7	-0,6	+0,7
Vel.:	+0,1	-0,1	+0,8

Objetivo		
+1,2	+3,7	-0,9

Avaliação da População

2

pbest	dist = ?		
	?	?	?
Pos.:	+0,4	+0,7	+0,3
Vel.:	+0,5	+0,8	-0,7

4

pbest	dist = ?		
	?	?	?
Pos.:	+0,1	+0,4	+0,9
Vel.:	-0,2	+0,6	+1,0

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

- Como esta é a primeira iteração, a melhor posição em que cada partícula já esteve (pbest) é a própria posição inicial, com distância a ser calculada a seguir.
- **Para encontrar gbest**, deve-se encontrar a **distância de todas as partículas em relação ao objetivo, e escolher a menor delas**.

gbest		
dist = ?		
?	?	?

1

pbest	dist = ?		
	+0,8	+0,1	-0,3
Pos.:	+0,8	+0,1	-0,3
Vel.:	+0,1	-0,5	-0,8

3

pbest	dist = ?		
	-0,7	-0,6	+0,7
Pos.:	-0,7	-0,6	+0,7
Vel.:	+0,1	-0,1	+0,8

Objetivo		
+1,2	+3,7	-0,9

Avaliação da População

2

pbest	dist = ?		
	+0,4	+0,7	+0,3
Pos.:	+0,4	+0,7	+0,3
Vel.:	+0,5	+0,8	-0,7

4

pbest	dist = ?		
	+0,1	+0,4	+0,9
Pos.:	+0,1	+0,4	+0,9
Vel.:	-0,2	+0,6	+1,0

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

Calculo da Distância Euclidiana entre cada partícula e o objetivo.

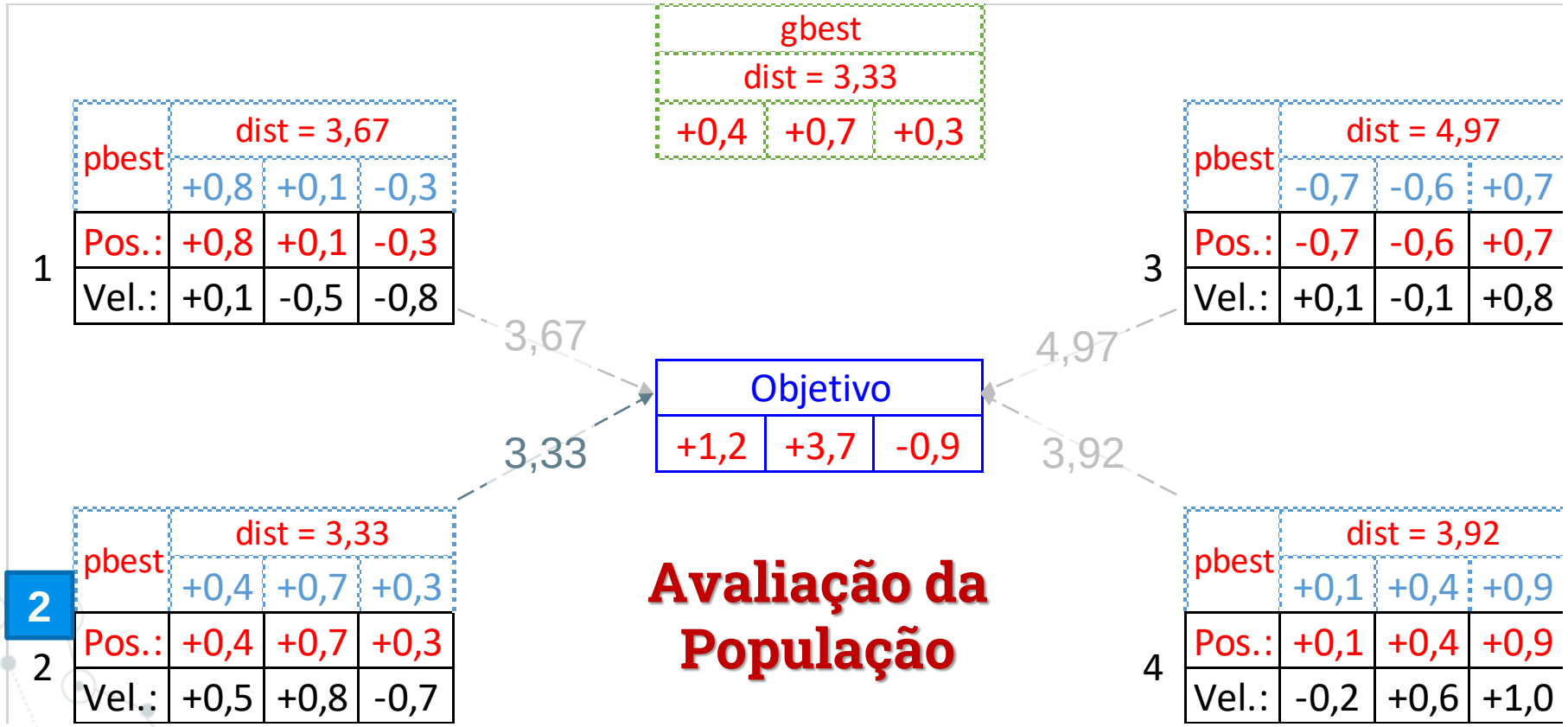
- Distância euclidiana entre a partícula 1 e o objetivo:
  $dist(P_1, \text{Objetivo}) = 3{,}67$
- Distância euclidiana entre a partícula 2 e o objetivo:
  $dist(P_2, \text{Objetivo}) = 3{,}33\ (Menor\ Distância)$
- Distância euclidiana entre a partícula 3 e o objetivo:
  $dist(P_3, \text{Objetivo}) = 4{,}97$
- Distância euclidiana entre a partícula 4 e o objetivo:
  $dist(P_4, \text{Objetivo}) = 3{,}92$
- **Distância Média ("avaliação global da população")**

$$dist_{média} = 3{,}97$$

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

- A partícula 2 apresentou uma menor distância euclidiana em relação ao objetivo, portanto **ela é a melhor da população (gbest).**
- O valor de gbest é atualizado com a posição desta partícula.
- Neste momento, já temos determinados pbest e gbest.



Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

- Cada partícula tem sua velocidade incrementada em um valor de **magnitude aleatória** na **direção da melhor** posição em que **ela** já esteve (**pbest**).
- Como é a primeira iteração, a posição atual coincide com **pbest**, então não há ajuste.

Exemplo no eixo x:
Se a posição$_x$ atual estiver à esquerda de pbest$_x$, então a velocidade é incrementada (+).
Se a posição$_x$ atual estiver à direita de pbest$_x$, então a velocidade é decrementada (-).

gbest		
dist = 3,33		
+0,4	+0,7	+0,3

1

pbest	dist = 3,67		
	+0,8	+0,1	-0,3
Pos.:	+0,8	+0,1	-0,3
Vel.:	+0,1	-0,5	-0,8

3

pbest	dist = 4,97		
	-0,7	-0,6	+0,7
Pos.:	-0,7	-0,6	+0,7
Vel.:	+0,1	-0,1	+0,8

Objetivo		
+1,2	+3,7	-0,9

2

pbest	dist = 3,33		
	+0,4	+0,7	+0,3
Pos.:	+0,4	+0,7	+0,3
Vel.:	+0,5	+0,8	-0,7

Ajuste da Velocidade (pessoal/cognitivo)

4

pbest	dist = 3,92		
	+0,1	+0,4	+0,9
Pos.:	+0,1	+0,4	+0,9
Vel.:	-0,2	+0,6	+1,0

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

- Cada partícula tem sua velocidade incrementada em um valor de **magnitude aleatória** na **direção da melhor posição** em que **todas** as partículas já estiveram (**gbest**).

Exemplo no eixo x:
Se a posição$_x$ atual estiver à esquerda de gbest$_x$, então a velocidade é incrementada (+).
Se a posição$_x$ atual estiver à direita de gbest$_x$, então a velocidade é decrementada (-).

gbest		
dist = 3,33		
+0,4	+0,7	+0,3

Objetivo		
+1,2	+3,7	-0,9

1

pbest	dist = 3,67		
	+0,8	+0,1	-0,3
Pos.:	+0,8	+0,1	-0,3
Vel.:	+0,1	-0,5	-0,8

2

pbest	dist = 3,33		
	+0,4	+0,7	+0,3
Pos.:	+0,4	+0,7	+0,3
Vel.:	+0,5	+0,8	-0,7

3

pbest	dist = 4,97		
	-0,7	-0,6	+0,7
Pos.:	-0,7	-0,6	+0,7
Vel.:	+0,1	-0,1	+0,8

4

pbest	dist = 3,92		
	+0,1	+0,4	+0,9
Pos.:	+0,1	+0,4	+0,9
Vel.:	-0,2	+0,6	+1,0

Ajuste da Velocidade (global/social)

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

- Cada partícula tem sua velocidade incrementada em um valor de **magnitude aleatória** na **direção da melhor posição** em que **todas** as partículas já estiveram (**gbest**).

Exemplo no eixo x:
Se a posição$_x$ atual estiver à esquerda de gbest$_x$, então a velocidade é incrementada (+).
Se a posição$_x$ atual estiver à direita de gbest$_x$, então a velocidade é decrementada (-).

gbest		
dist = 3,33		
+0,4	+0,7	+0,3

1

pbest	dist = 3,67		
	+0,8	+0,1	-0,3
Pos.:	+0,8	+0,1	-0,3
Vel.:	+0,1	-0,5	-0,8

3

pbest	dist = 4,97		
	-0,7	-0,6	+0,7
Pos.:	-0,7	-0,6	+0,7
Vel.:	+0,1	-0,1	+0,8

Objetivo		
+1,2	+3,7	-0,9

2

pbest	dist = 3,33		
	+0,4	+0,7	+0,3
Pos.:	+0,4	+0,7	+0,3
Vel.:	+0,5	+0,8	-0,7

4

pbest	dist = 3,92		
	+0,1	+0,4	+0,9
Pos.:	+0,1	+0,4	+0,9
Vel.:	-0,2	+0,6	+1,0

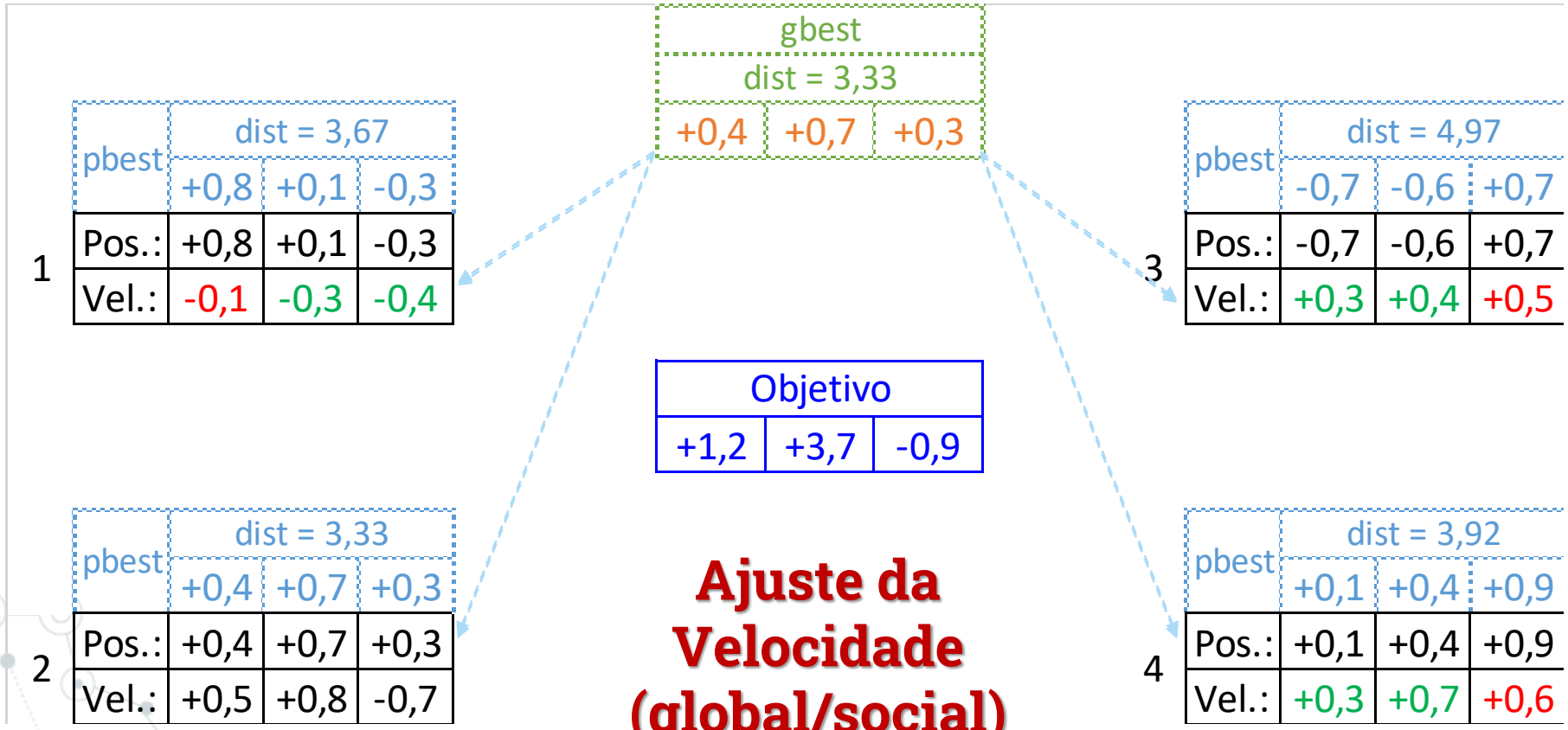
Ajuste da Velocidade (global/social)

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

- Cada partícula tem sua velocidade incrementada em um valor de **magnitude aleatória** na **direção da melhor posição** em que **todas** as partículas já estiveram (**gbest**).

Exemplo no eixo x:
Se a posição$_x$ atual estiver à esquerda de gbest$_x$, então a velocidade é incrementada (+).
Se a posição$_x$ atual estiver à direita de gbest$_x$, então a velocidade é decrementada (-).

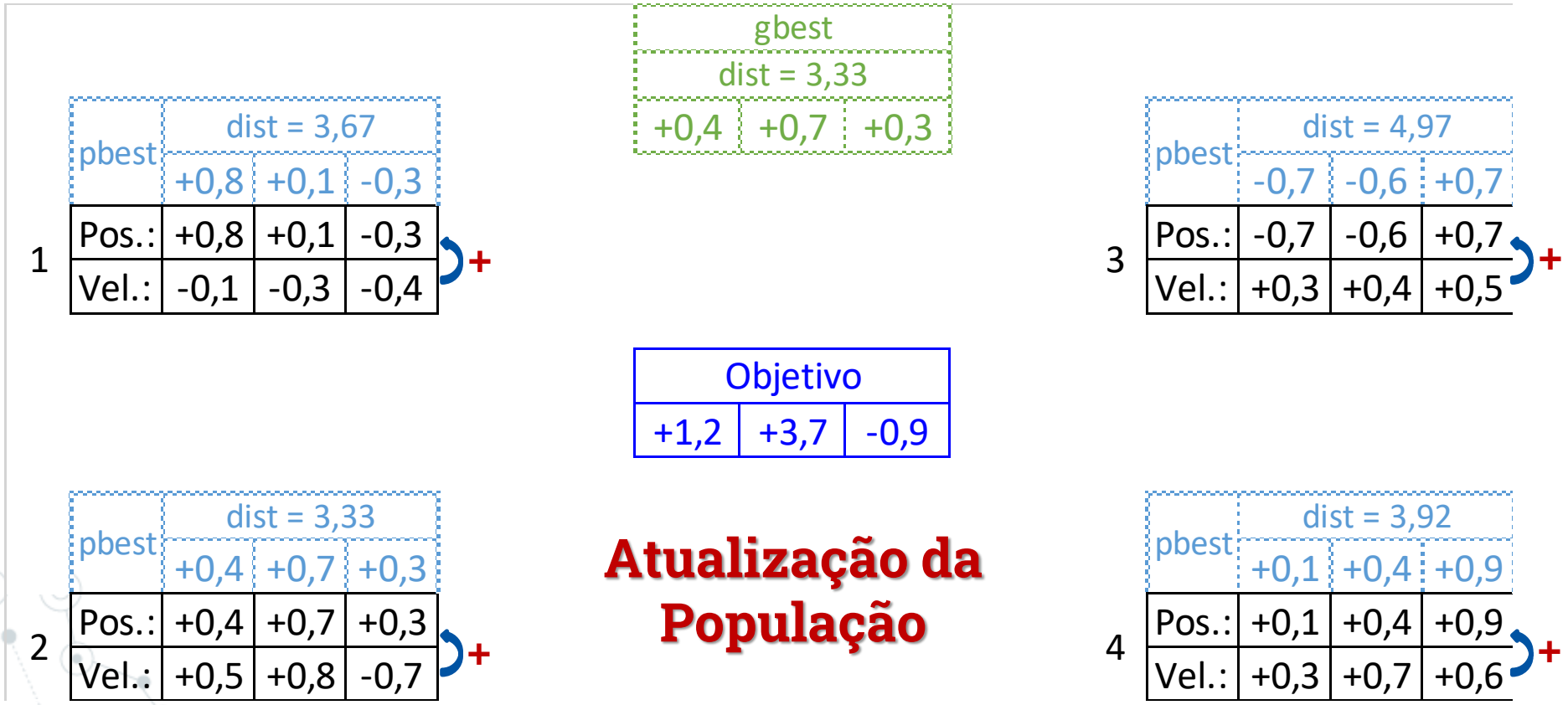


Otimização por Enxame de Partículas (PSO)

The Cornfield Vector

Exemplo

- **Atualização da posição:** cada partícula tem sua posição atualizada com base no novo vetor de velocidade ajustado anteriormente.

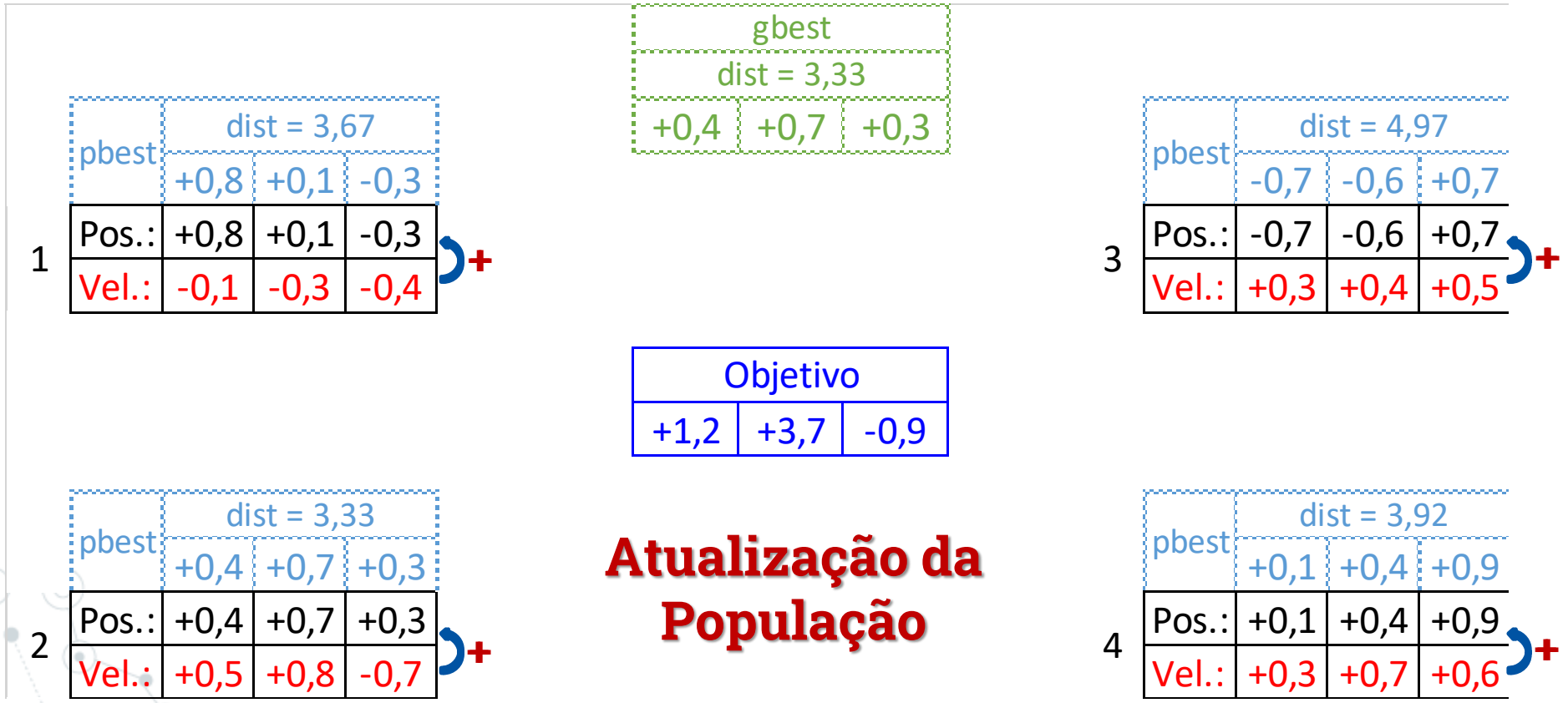


Otimização por Enxame de Partículas (PSO)

The Cornfield Vector

Exemplo

- **Atualização da posição:** cada partícula tem sua posição atualizada com base no novo vetor de velocidade ajustado anteriormente.



Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

- **Atualização da posição:** cada partícula tem sua posição atualizada com base no novo vetor de velocidade ajustado anteriormente.

gbest		
dist = 3,33		
+0,4	+0,7	+0,3

	pbest	dist = 3,67		
		+0,8	+0,1	-0,3
1	Pos.:	+0,7	-0,2	-0,7
	Vel.:	-0,1	-0,3	-0,4

	pbest	dist = 4,97		
		-0,7	-0,6	+0,7
3	Pos.:	-0,4	-0,2	+1,2
	Vel.:	+0,3	+0,4	+0,5

Objetivo		
+1,2	+3,7	-0,9

Atualização da População

	pbest	dist = 3,33		
		+0,4	+0,7	+0,3
2	Pos.:	+0,9	+1,5	-0,4
	Vel.:	+0,5	+0,8	-0,7

	pbest	dist = 3,92		
		+0,1	+0,4	+0,9
4	Pos.:	+0,4	+1,1	+1,5
	Vel.:	+0,3	+0,7	+0,6

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo**

A População Melhorou?

Calculo da Distância Euclidiana entre cada partícula e o objetivo.

- Distância euclidiana entre a partícula 1 e o objetivo:
  $dist(P_1, \text{Objetivo}) = 3{,}94$ (piorou, era 3,67)
- Distância euclidiana entre a partícula 2 e o objetivo:
  $dist(P_2, \text{Objetivo}) = 2{,}28$ (melhorou, era 3,33)
- Distância euclidiana entre a partícula 3 e o objetivo:
  $dist(P_3, \text{Objetivo}) = 4{,}71$ (melhorou, era 4,97)
- Distância euclidiana entre a partícula 4 e o objetivo:
  $dist(P_4, \text{Objetivo}) = 3{,}63$ (melhorou, era 3,92)
- **Distância Média ("avaliação global da população")**

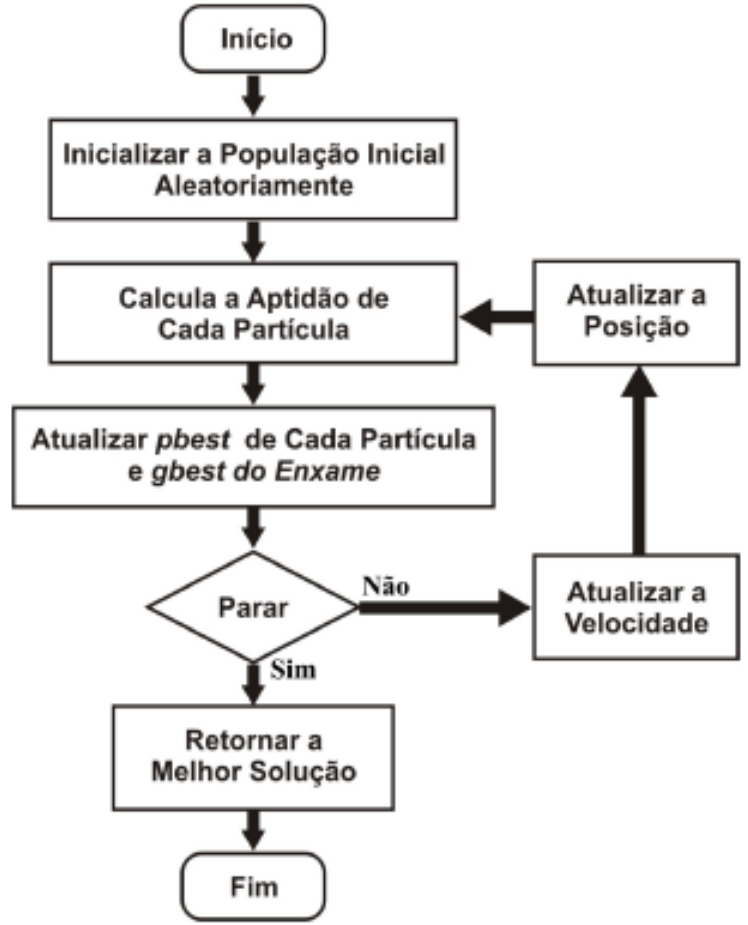
$dist_{média} = 3{,}64$ (era, 3,97)

Otimização por Enxame de Partículas (PSO)

- *The Cornfield Vector*
  - **Exemplo:**
    - **Os passos são repetidos até que o critério de parada seja atingido:**
      - Número de iterações;
      - Tempo de execução;
      - Qualidade da solução;
      - Variação da qualidade da solução entre iterações consecutivas.

Otimização por Enxame de Partículas (PSO)

- *Canonical PSO → PSO Clássico*
  - Nesta técnica, cada pássaro é visto como uma partícula e representa uma **possível solução para o problema** a ser otimizado.
  - Estas partículas percorrem o **espaço de busca** com uma **velocidade variável.**
  - Seu movimento no espaço depende de duas variáveis, a **experiência individual (pbest) e a experiência do enxame (gbest)**. Essas variáveis são obtidas através da avaliação da posição de cada partícula através de uma **função aptidão**.
  - Por conseguinte, a posição da partícula é calculada empregando a melhor posição (pbest) experimentada pela partícula até a iteração atual e a melhor posição (gbest) experimentada pelo enxame até a iteração atual



Otimização por Enxame de Partículas (PSO)

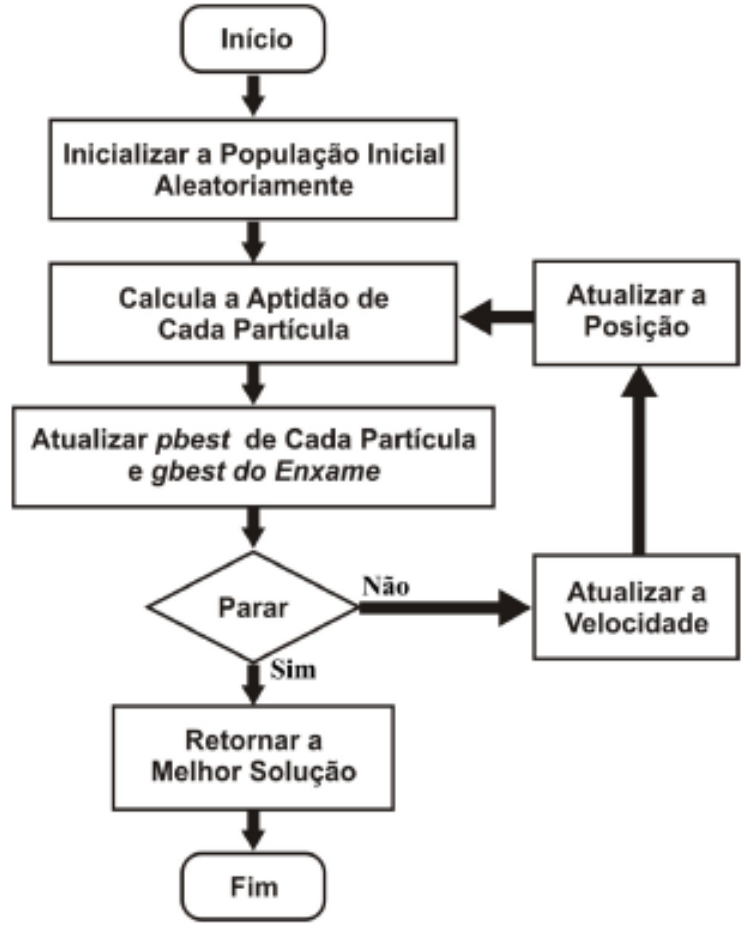
Funcionamento Básico:

1. **Inicializar a População Inicial Aleatoriamente:**

O primeiro passo do PSO consiste em inicializar a sua população incial, para fazer esta inicialização é necessário **definir o espaço de busca de cada partícula do PSO**.

Para isto, deve-se saber o **limite superior e inferior permitido para a posição de cada partícula**.

Após definido o espaço de busca das partículas, define-se aleatoriamente a **posição incial e a velocidade inicial** de cada partícula do PSO.



Otimização por Enxame de Partículas (PSO)

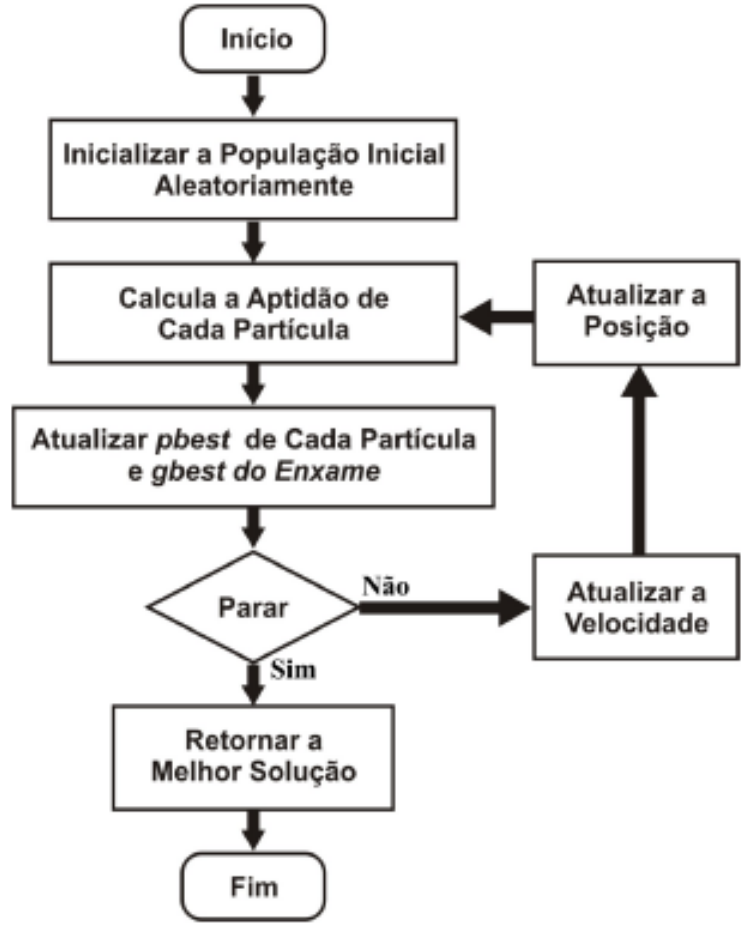
Funcionamento Básico:

2. **Calcula a Aptidão de cada Partícula:**

A posição de cada partícula é avaliada através da **função aptidão**.

A função aptidão deve ser definida cuidadosamente, pois ela irá determinar a direção para onde a partícula irá se locomover.

O valor calculado pela função aptidão será utilizado para determinar os valores de **pbest e gbest.**



Otimização por Enxame de Partículas (PSO)

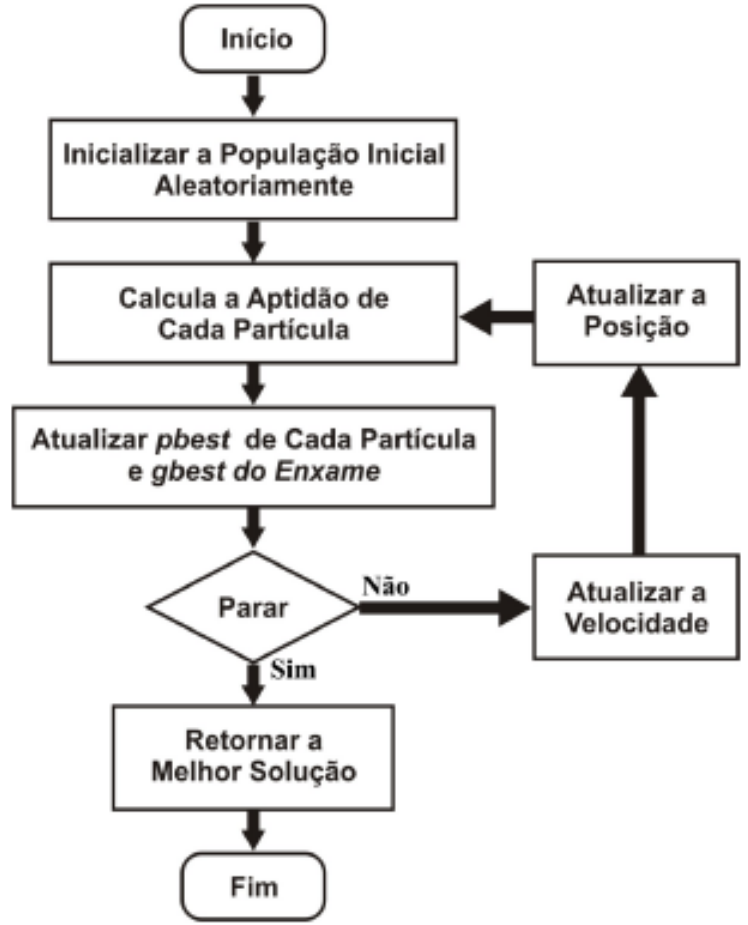
Funcionamento Básico:

3. **Atualizar *pbest* para cada partícula e *gbest* para o enxame:**

   Após determinado o valor de aptidão de cada partícula, deve-se atualizar os valores de pbest e gbest.

   Se o valor de aptidão encontrado na iteração atual for superior ao registrado em tais parâmetros. Ou seja, caso a partícula encontre uma posição melhor que as posições por ela visitada até a iteração anterior o valor de pbest será atualizado, caso contrário o valor de pbest será mantido inalterado.

   Caso o enxame encontre uma posição melhor que a melhor posição visitada até a iteração anterior o valor de gbest será atualizado, caso contrário, gbest será mantido inalterado.



Otimização por Enxame de Partículas (PSO)

Funcionamento Básico:

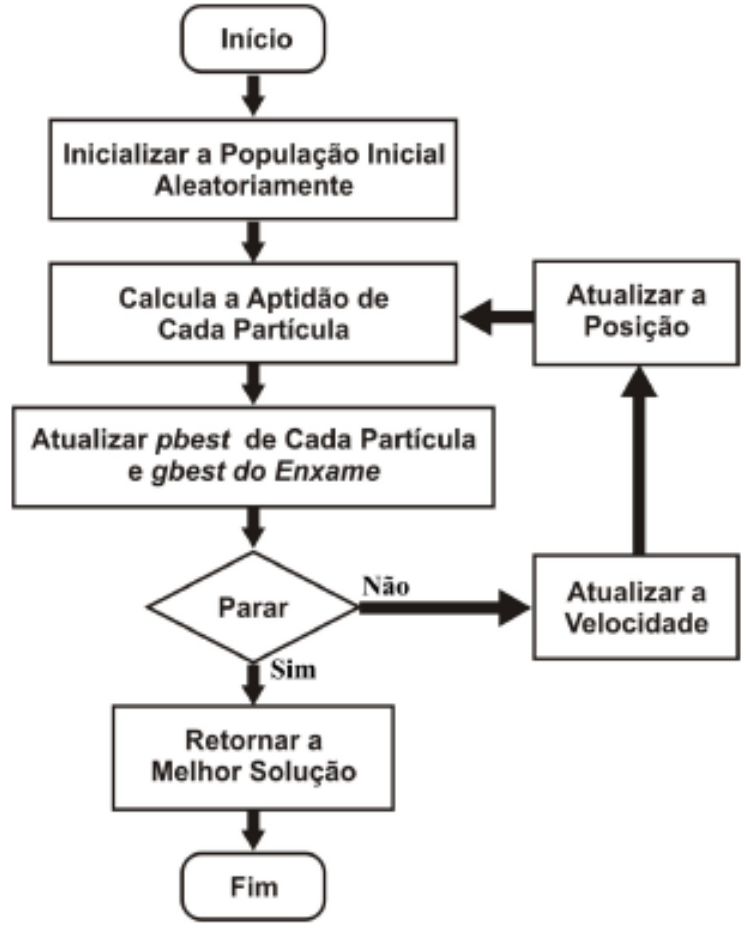
4. **Atualizar *pbest* para cada partícula e *gbest* para o enxame:**

   O critério de parada define se o algoritmo deve ser encerrado ou deve continuar executando, voltando ao passo 2.

   Caso o critério de parada seja satisfeito o algoritmo deve retornar a posição de gbest, ou seja, a melhor solução para o problema em questão.

   Caso contrário, a posição da partícula deve continuar sendo atualizada até que tal critério seja satisfeito.

   Para cada aplicação pode-se definir um critério de parada distinto.



Otimização por Enxame de Partículas (PSO)

Funcionamento Básico:

5. Atualiza Velocidade:

Caso o critério de parada não seja satisfeito é necessário determinar a velocidade de cada partícula no espaço de busca, a qual é dada por:

$$V_i^{k+1} = \omega \cdot V_i^k + C_1.\text{rand}_1\left(pbest_i^k - X_i^k\right) + C_2 \cdot \text{rand}_2(gbest^k - X_i^k)$$

k → Iteração atual da partícula	X_i^k → Posição atual da partícula i;	$pbest_i^k$ → Melhor posição da partícula i até a iteração atual ;
V_i^{k+1} → Velocidade da partícula i na iteração $k + 1$;	V_i^k → Velocidade da partícula i na iteração atual;	$gbest_k$ → Melhor posição do enxame até a iteração atual;
C_1 e C_2 → Fatores de aprendizagem da partícula, os quais definem a influência da experiência individual e coletiva no movimento das partículas;	$rand_1$ e $rand_2$ → Valor aleatório entre [0,1];	ω → Coeficiente de inércia.

Otimização por Enxame de Partículas (PSO)

Funcionamento Básico:

5. Atualiza Velocidade:

Caso o critério de parada não seja satisfeito é necessário determinar a velocidade de cada partícula no espaço de busca, a qual é dada por:

$$V_i^{k+1} = \omega \cdot V_i^k + C_1.\text{rand}_1\left(pbest_i^k - X_i^k\right) + C_2 \cdot \text{rand}_2(gbest^k - X_i^k)$$

O coeficiente de inércia da partícula é responsável por ponderar a influência da velocidade atual na atualização da velocidade de cada partícula. Este coeficiente é determinado por:

$$\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{k_{total}} \cdot k$$

Onde:

- k_{total} é o número total de iterações;
- ω_{max} e ω_{min} representam os valores máximo e mínimo do coeficiente de inércia, respectivamente.

Otimização por Enxame de Partículas (PSO)

Funcionamento Básico:

5. Atualiza Velocidade:

Caso o critério de parada não seja satisfeito é necessário determinar a velocidade de cada partícula no espaço de busca, a qual é dada por:

$$V_i^{k+1} = \omega \cdot V_i^k + C_1.\text{rand}_1\left(pbest_i^k - X_i^k\right) + C_2 \cdot \text{rand}_2(gbest^k - X_i^k)$$

O coeficiente de inércia da partícula é responsável por ponderar a influência da velocidade atual na atualização da velocidade de cada partícula. Este coeficiente é determinado por:

$$\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{k_{total}} \cdot k$$

→ A escolha de valores máximo e mínimo do coeficiente de inércia deve ser cuidadosa, pois valores muito altos podem fazer com que a partícula seja afastada do valor ótimo, ou seja, se perca do enxame, porém valores muito baixos fazem com que ocorra uma convergência prematura da população, sendo atingido um ótimo local, fato este que não é desejado

Otimização por Enxame de Partículas (PSO)

Funcionamento Básico:

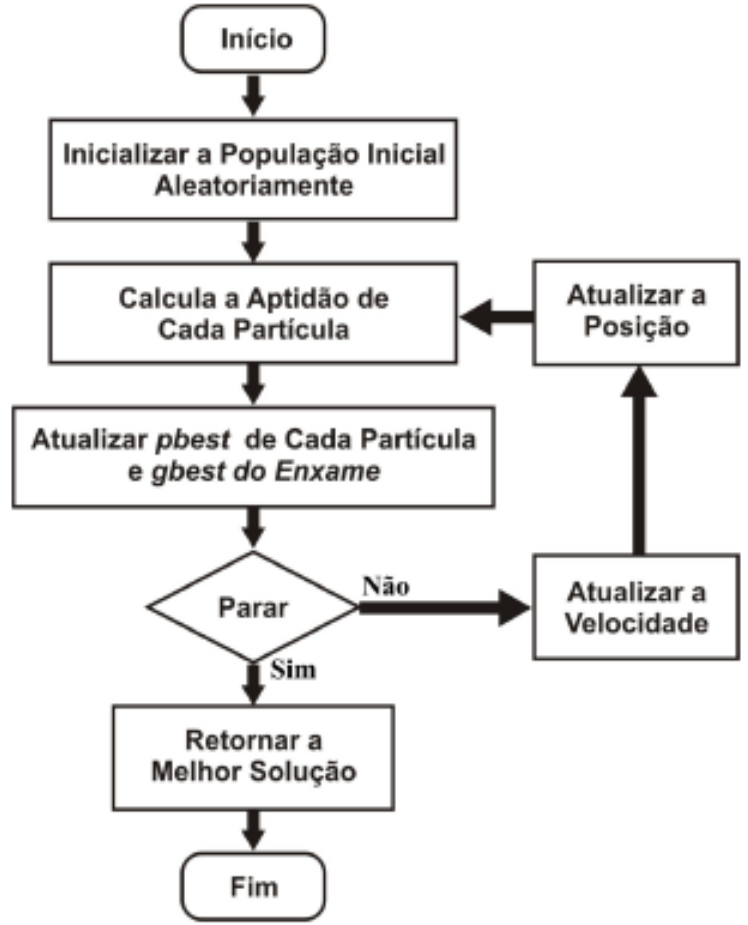
5. **Atualiza Velocidade:**

Caso o critério de parada não seja satisfeito é necessário determinar a velocidade de cada partícula no espaço de busca, a qual é dada por:

$$V_i^{k+1} = \omega \cdot V_i^k + C_1.\text{rand}_1\left(pbest_i^k - X_i^k\right) + \\ + C_2 \cdot \text{rand}_2\left(gbest^k - X_i^k\right)$$

Onde

$$\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{k_{total}} \cdot k$$



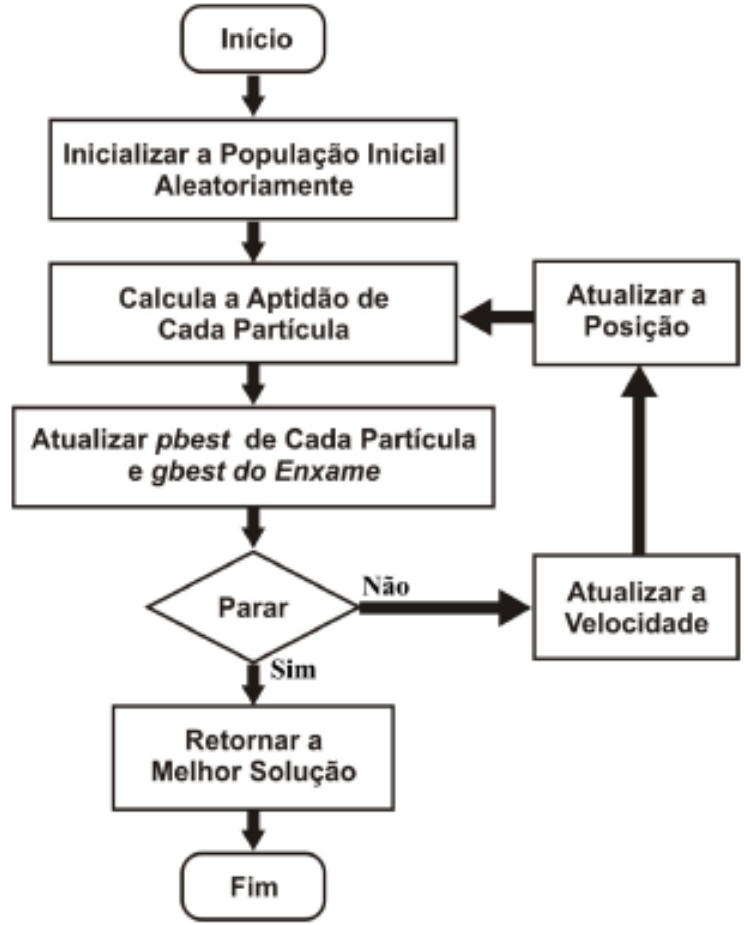
Otimização por Enxame de Partículas (PSO)

Funcionamento Básico:

5. Atualiza Posição:

Através da posição atual da partícula e da nova velocidade calculada é possível atualizar a posição da partícula no espaço de busca. Para realizar esta atualização o PSO utiliza a equação:

$$X_i^{k+1} = X_i^k + V_i^{k+1}$$



Otimização por Enxame de Partículas (PSO)

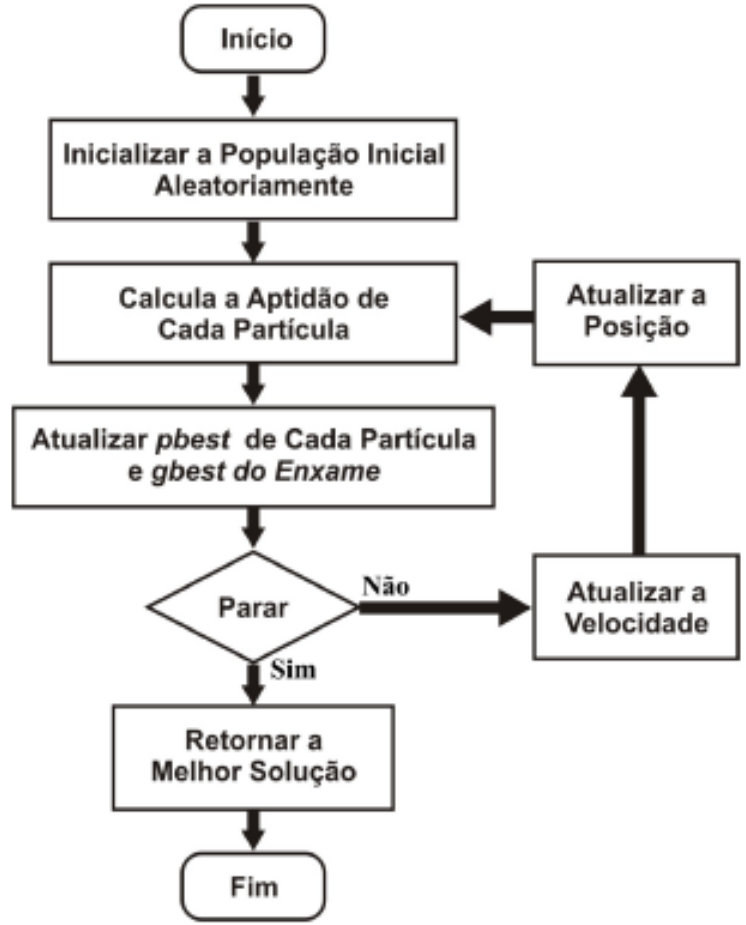
Observações:

- O PSO pode ser utilizado na resolução de problemas complexos, os quais não possuem soluções analíticas praticáveis. Estes problemas em sua grande maioria necessitam considerar diversas variáveis para determinar a sua solução.

- Por este motivo, o PSO deve ser capaz de codificar tais variáveis, com o objetivo de encontrar a melhor solução para o problema. O PSO foi projetado, inicialmente, considerando **variáveis reais contínuas.**

- Para realizar a implementação do PSO é necessário definir o **critério de parada** do algoritmo.

- No algoritmo do PSO existem diversos **parâmetros que devem ser ajustados**, visando produzir variações no modo como o algoritmo executa a busca no espaço de busca do problema, ou seja, tais parâmetros definem a movimentação das partículas pelo espaço de busca.



Otimização por Enxame de Partículas (PSO)

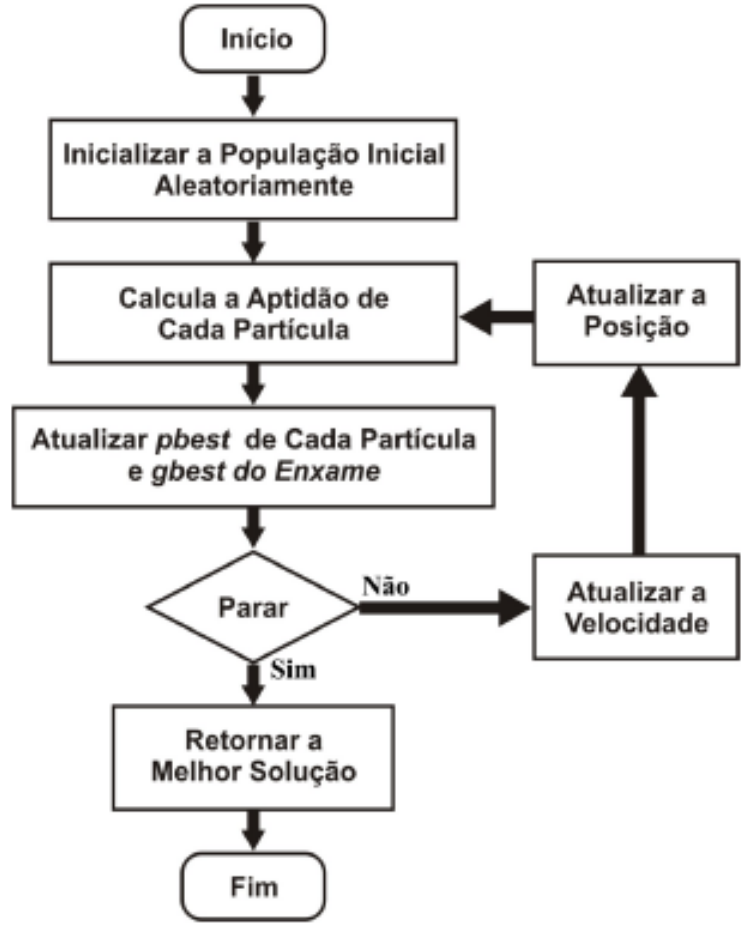
PSO para Variáveis Binárias:

- Versão do PSO utilizada para a otimização de problemas de otimização **DISCRETOS.**
- Nesta versão do PSO a atualização da posição das partículas é realizada através da equação abaixo:

$$X_i^{k+1} = \begin{cases} 1 & se \quad \text{rand}() < S(V_i^{k+1}) \\ 0 & , \quad Caso\ Contrário \end{cases}$$

- $S(V_i^{k+1})$→ Valor da função sigmóide para o valor da velocidade calculado.

As demais etapas do PSO são as mesmas apresentadas anteriormente.



Inatel

Obrigada pela Atenção!

Dúvidas?

victoria.souto@Inatel.br