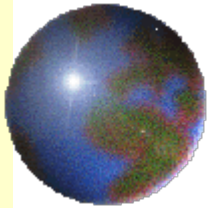
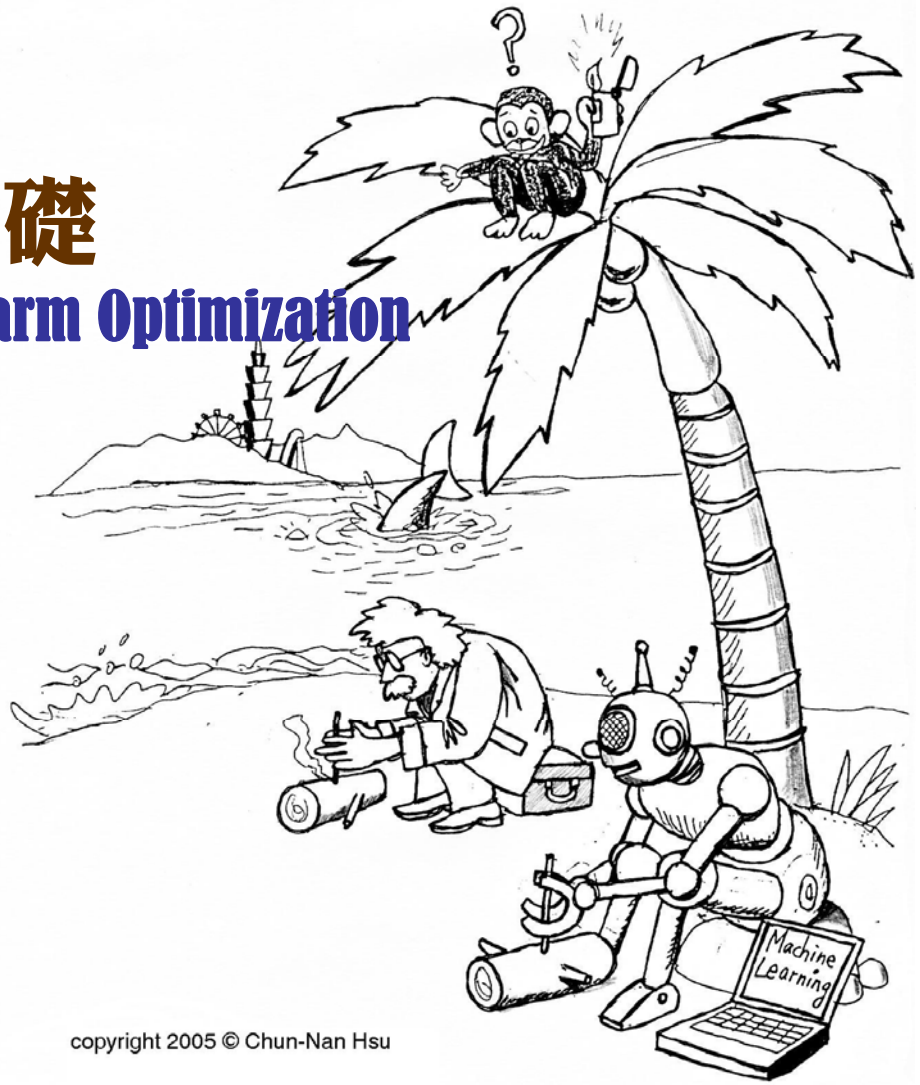


粒子群演算法基礎

Foundations of Particle Swarm Optimization (PSO)



授課教師：陳士杰

國立聯合大學 資訊管理學系

Outlines

- 認識粒子群最佳化演算法(Particle Swarm Optimization; PSO)。
- 了解粒子群最佳化演算法處理問題的程序。
- 了解基本的粒子群最佳化演算法與相關的改進。

Basic Description of PSO

- 自然界生物有時候是以群體形式存在的。而人工生命研究的主流之一是探索這些自然生物是如何以群體的形式生存，並在電腦裡重建這種模型。
  - 生物學家F. Heppner發現鳥群的同步飛行這個整體行為只是建立在每隻鳥對周圍的局部感知上面，而且並不存在一個集中的控制者。也就是說整個群體組織起來但卻沒有一個組織者，群體之間相互協調卻沒有一個協調者。
  - R. Boyd和P. J. Richerson在研究人類的決策過程時，發現人們在決策過程中使用兩類重要訊息：自身的經驗和其他人的經驗。也就是說，人們根據自身的經驗和他人的經驗進行自己的決策。
  - 生物社會學家E. O. Wilson曾說：至少從理論上，在搜尋食物的過程中，群體中個體成員可以得益於所有其它成員的發現和先前自身的經歷。

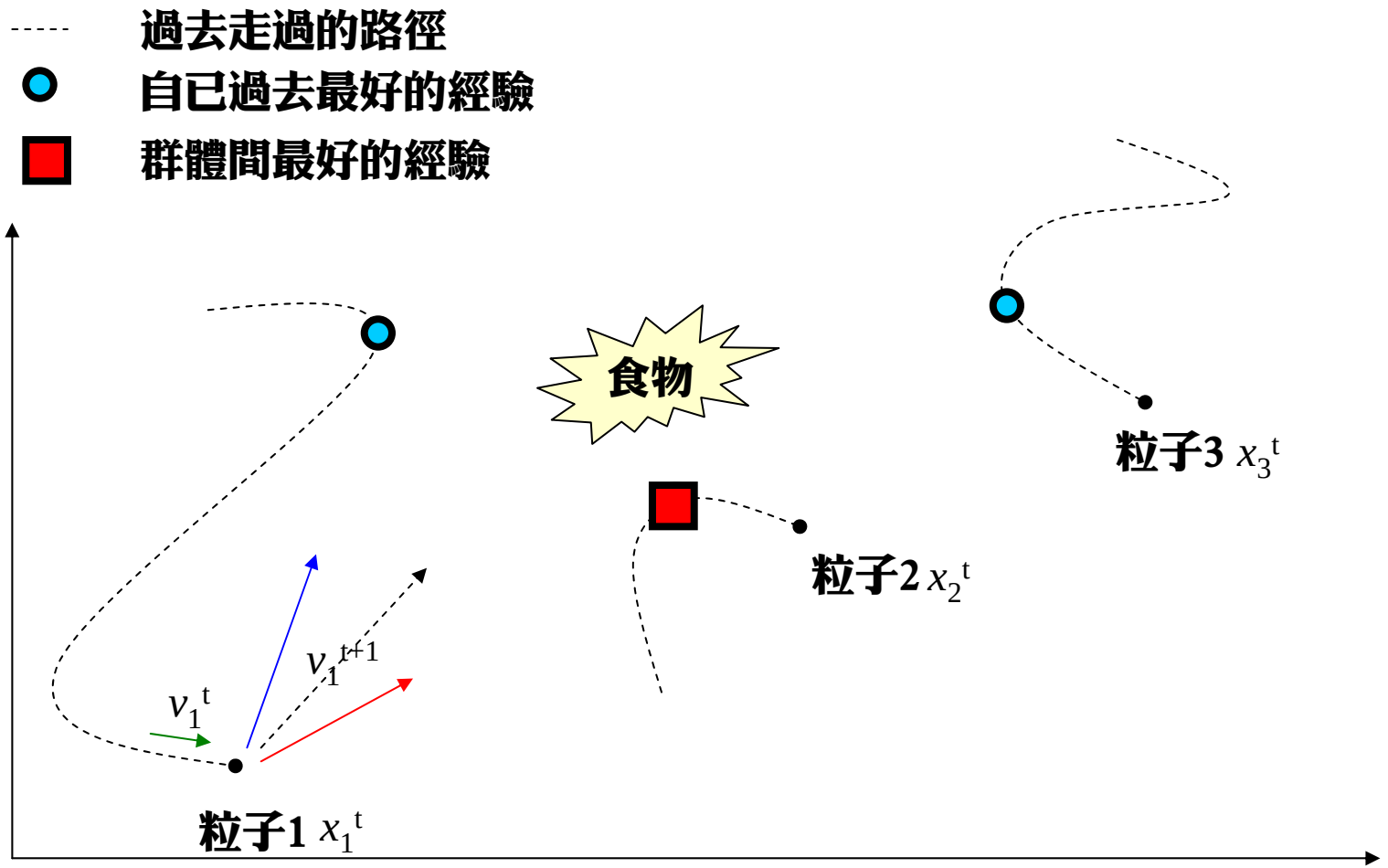
- 粒子群最佳化演算法 (簡稱為PSO)，是一種以群體為基礎的最佳化搜尋技術，由 James Kennedy 和 Russell Eberhart 兩位學者於1995年時所提出。
- Kennedy 和 Eberhart認為：
  - 鳥群中的每個個體能夠通過一個規則估計自身位置的適應性
  - 每個個體能夠記住自已當前所找到的最好位置，稱為局部最佳(pbest)。
  - 此外，還記住群體中所有鳥中找到的最好位置，稱為全局最佳(gbest)。

- 因此，在PSO中：
  - 最佳化問題當中的每一個可能解，都可以被想像成一隻鳥，而每隻鳥被稱之為粒子(Particle)。
  - 每個粒子經由適應函數的衡量而具有一個適應值，以判斷目前所在位置之好壞。
  - 每一個粒子必須賦予記憶性，以記得所搜尋到的最佳位置。
  - 每一個粒子皆以此次的飛行速度與方向來決定下一時間點飛行速度。
- 由於PSO概念簡單，實作方便且參數設置少，近年來受到學術界的廣泛重視。

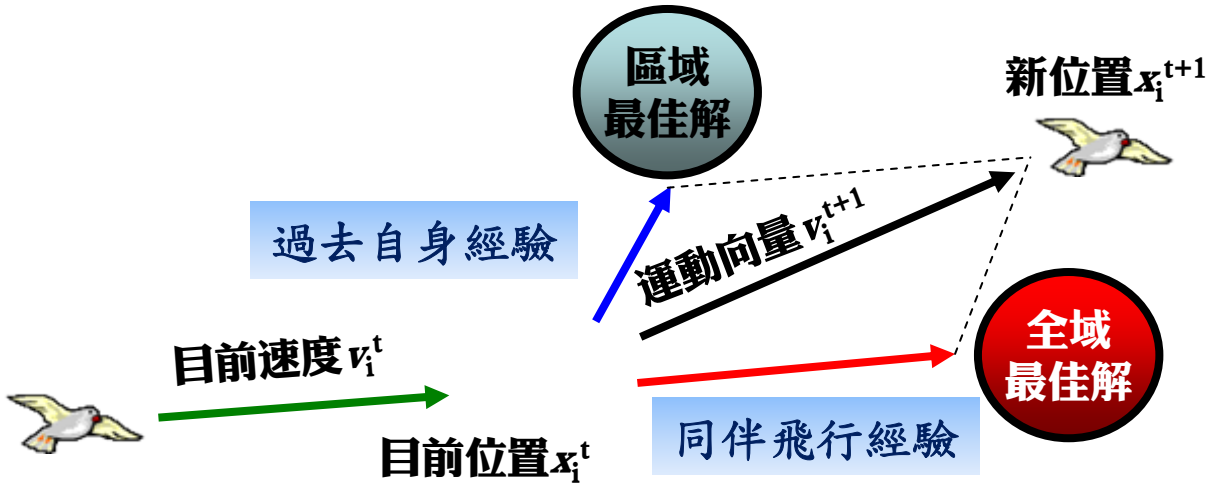
■ 基本粒子群演算法

- PSO演算法的基本概念是**隨機產生一群粒子**，而每個粒子可視為最佳化問題的一個潛在之可能解。
- 假設一個由m個粒子組成的群體在D維的搜索空間以一定的速度飛行。m也被稱為群體規模，過大的m會影響演算法的運算速度和收斂性。
- 粒子 i 在時刻 t 的狀態屬性設置如下: $(1 \le d \le D, 1 \le i \le m)$
  - **粒子 i 當前的位置向量**：$x_i^t = (x_{i1}^t, x_{i2}^t, \ldots, x_{id}^t \ldots, x_{iD}^t)$。其中 $x_{id}^t \in [L_d, U_d]$， L_d 與 U_d 分別為搜索空間的下限與上限。
  - **粒子 i 的飛行速度**：$v_i^t = (v_{i1}^t, v_{i2}^t, \ldots, v_{id}^t \ldots, v_{iD}^t)$，即粒子移動的距離。其中 $v_{id}^t \in [v_{min}, v_{max}]$， v_{min} 與 v_{max} 分別為最小和最大速度。
  - **粒子 i 迄今搜索到的最佳位置**：$p_i^t = (p_{i1}^t, p_{i2}^t, \ldots, p_{iD}^t)$。(局部最佳)
  - **粒子群迄今搜索到的最佳位置**：$p_g^t = (p_{g1}^t, p_{g2}^t, \ldots, p_{gD}^t)$。(全域最佳)

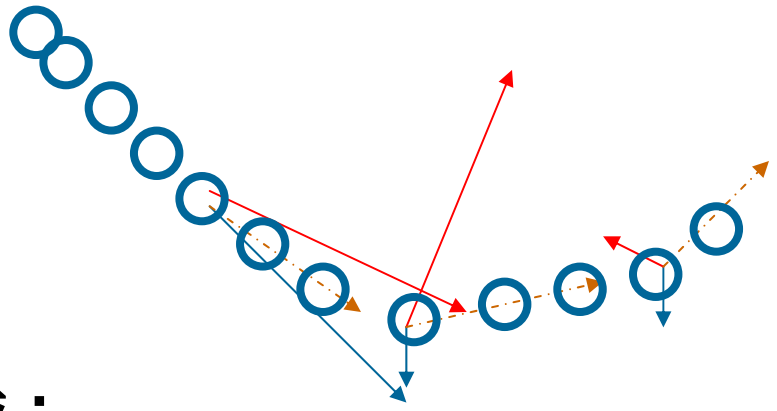
- 在某時刻 t 之下，每個粒子的狀態：



- **在t+1時刻，粒子 i 根據以下公式更新速度和位置：**
  - $v_i^{t+1} = v_i^t + c_1 r_1 (p_i^t - x_i^t) + c_2 r_2 (p_g^t - x_i^t)$ **(公式 1)**
  - $x_i^{t+1} = x_i^t + v_i^{t+1}$ **(公式 2)**
- **其中：**
  - r_1**和**r_2**是介於** [0, 1]**之間的隨機值，用來保持群體的多樣性。**
  - c_1**和**c_2**稱為學習因子，也可稱為加速因子，可以讓粒子 i 向自已的歷史最佳點(認知學習因子**c_1**)以及群體內歷史最佳點(社會學習因子**c_2**)靠近。通常取**$c_1 = c_2 = 2$**。**



- **粒子飛行狀態(時刻 t)：**



- **公式 1 主要由三部份組成：**
  - **粒子依據自身目前的速度進行慣性運動。**
  - **認知部份**(Cognition part)，**即粒子本身的思考。**
  - **社會部份**(Social part)，**即粒子間的資訊共享與相互合作。**
- **公式 1 正是利用粒子根據它上一次迭帶的速度、當前位置和自身最好的經驗、群體最好經驗之間的向量距離來更新速度，然後粒子根據公式 2 飛向新的位置。**

- 由於粒子群演算法中沒有實際的機制來控制粒子的速度，所以有必要對速度的最大值進行限制。
  - 在此，採用[v_{min}, v_{max}]做為粒子飛行速度的上下限。尤其是 v_{max} 這個參數被証明是非常重要的，若值太大會導致粒子跳過最佳解，但太小的話又會導致對搜索空間的不充份搜索。
- 如果適當調整加速因子c_1和c_2 這兩個參數的話，可以減少局部最小值的困擾，當然也會使收斂速度變快。

■PSO演算法參數分析

- 由於PSO演算法中的參數較少，所以每個參數的設置對演算法性能皆有其一定影響。
- PSO演算法的參數改進主要是針對速度迭代公式(即：公式 1)中，涉及以下兩方面：
  - 慣性權重的調節
  - 學習因子的調節

慣性權重的調節

- **為了改善基本粒子群演算法的收斂性能，**Shi**與**Eberhart**在**1998**年導入了慣性權重，且這個改進後的演算法反而成為標準的粒子群演算法。**
- Shi**與**Eberhart**將更改速度的公式 (即：公式** 1) **修改如下，而位置公式 (即：公式** 2) **則保持不變：**

$$v_i^{t+1} = w\, v_i^t + c_1\, r_1\, (p_i^t - x_i^t) + c_2\, r_2\, (p_g^t - x_i^t) \qquad \text{(公式 3)}$$

  - w **稱為慣性權重：**
    - **當慣性權重較大時，**PSO**傾向於開發者 (**Exploitation**) ，大範圍地探索新的區域，此時的**PSO**就像是全域搜索的方法。**
    - **當慣性權重較小時，**PSO**傾向於探測者 (**Exploration**) ，在局部範圍內找尋最佳值，此時的**PSO**就像是局部搜索的方法。**

- 因此，如果在迭代計算的過程中，以線性遞減慣性權重，則PSO在一開始時具有良好的全域搜索能力，能夠迅速定位到接近全域最佳解的區域，而在後期具有良好的局部搜索能力，能夠精確地得到全域最佳解。
- 線性遞減公式如下：

$$w = w_{star} - (w_{star} - w_{end}) / T_{max} \times t \qquad \text{(公式 4)}$$

  - T_{max} 為最大迭代次數
  - t 為當前迭代次數
  - w_{star} 與 w_{end} 分別為初始慣性權重與終止慣性權重
- Shi等人在2000年經過多組反覆實驗後，建議採用從0.9線性遞減到0.4的策略。

學習因子的調節

- 在PSO演算法中，學習因子c_1和c_2決定了粒子本身經驗與群體的經驗對粒子運動軌跡的影響。在理想狀態下，搜索初期要使粒子盡可能地探索整個空間。而在搜索末期，粒子應避免陷入局部極值。
  - M. Clerc (1999) 推導出c_1和c_2取2.5。一般的設置是$c_1 = c_2 \in [1, 2.5]$
  - A. Ratnawecra等人 (2004) 提出利用線性調整學習因子：

$$c_1 = c_{1s} + (c_{1e} - c_{1s}) / T_{max} \times t \quad \text{(公式 5)}$$

$$c_2 = c_{2s} + (c_{2e} - c_{2s}) / T_{max} \times t \quad \text{(公式 6)}$$

    - c_{1s} 與 c_{2s} 分別表示c_1和c_2的迭代初始值
    - c_{1e} 與 c_{2e} 分別表示c_1和c_2的迭代終值

    在搜索初期粒子飛行主要參考粒子本身的歷史訊息，到了後期則注重社會訊息。

■操作步驟 (以最小化問題為例)

1. 初始化及參數設定

- 隨機產生 m 個粒子，初始化每個粒子 i 的位置向量 x_i^t 和速度 v_i^t 。
- 搜索空間的下限與上限L_d與 U_d、學習因子c_1和c_2 、最小和最大飛行速度v_{min}與v_{max}、迭代次數 t = 1、最大迭代次數 T_{max} 、慣性權重 w 與收斂精度 ξ

2. 評估每一個粒子 i

- 測量每個粒子所在位置的適應值F_i^t 。
  - 若是初始情況，則將每個粒子目前的適應值F_i^t設成該粒子的局部(個體)最佳值F_i^*；且將每個粒子的目前初始位置設成其局部最佳位置 p_i^t 。
  - 否則，將當前粒子的適應值F_i^t和該粒子目前的局部最佳值F_i^*做比較，如果優於該粒子的局部(個體)最佳值F_i^* ，則將局部最佳位置 p_i^t 更新為該粒子目前的位置，且更新個體最佳值F_i^* 。
- 從所有粒子中找出當前個體最佳極值 $\min(F_1^*, F_2^*, \ldots, F_m^*)$ 。若此極值較目前的全域最佳極值F_g^t為優，則將p_g^t設定為此極值所表示之粒子的所在位置，且更新全域最佳值F_g^t 。

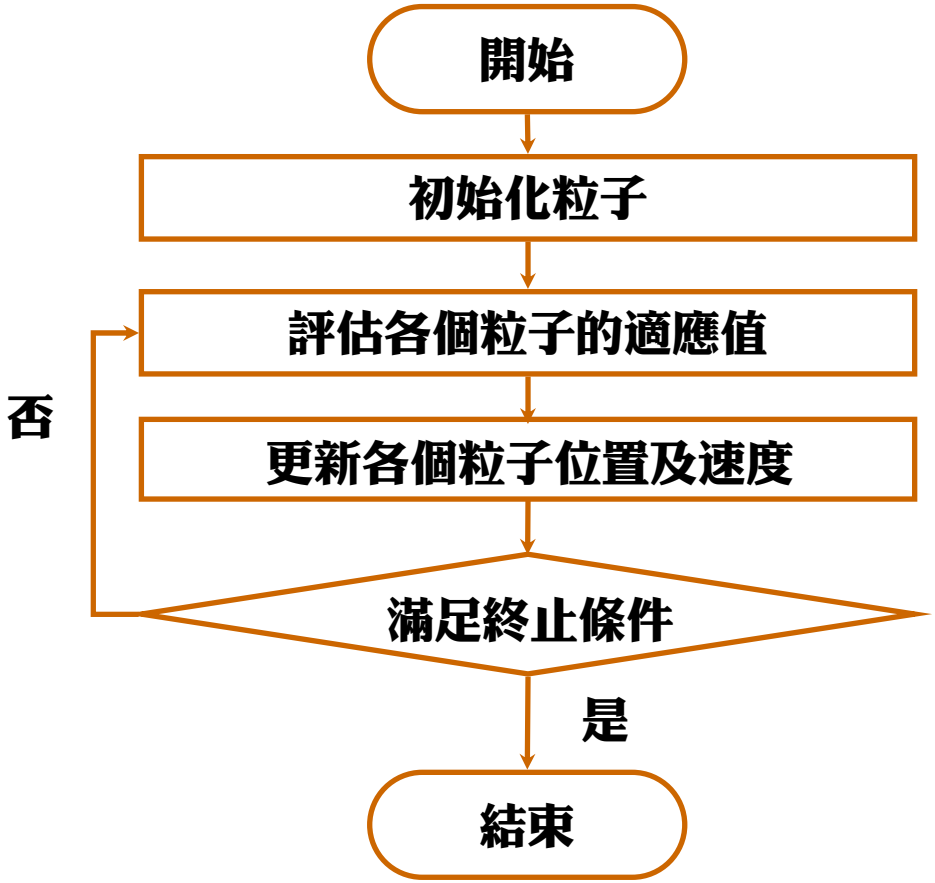
3. 粒子的狀態更新

- 利用公式 3和公式 2對每一個粒子的速度和位置進行更新，並設t = t+1。
  - (飛行速度)：如果速度v_i^{t+1}於空間座標當中的某一維度之移動距離v_{id}^{t+1} > v_{max} ，則將其設置為 v_{max} ；如果v_{id}^{t+1} < v_{min} ，則將其設置為 v_{min} 。
  - (新位置)：如果新位置x_i^{t+1}於空間座標當中的某一維度值x_{id}^{t+1} > U_d ，則將其設置為 U_d ；如果x_{id}^{t+1} < L_d ，則將其設置為 L_d 。

4. 檢驗是否符合結束條件

- 如果當前的迭代次數 t 達到預先設定的最大次數 T_{max} ，或是最終結果小於預定的收斂精度 ξ ，則停止迭代，輸出最佳解；否則跳到步驟 2。

PSO演算法流程圖



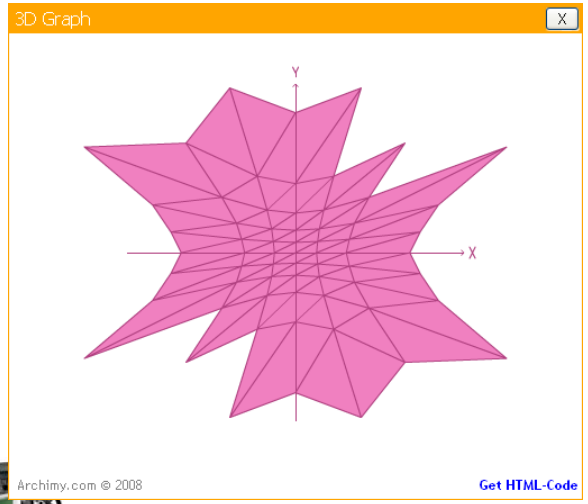
※範例※

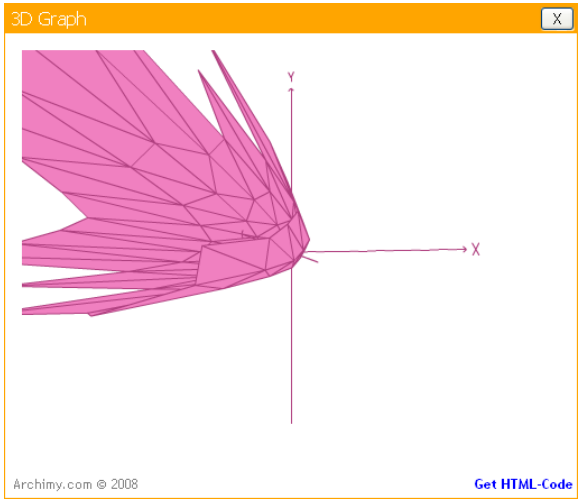
- **以PSO演算法求解以下極小化問題：**

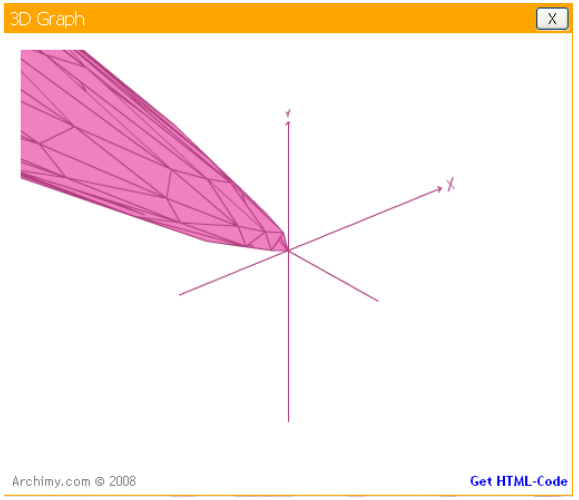
$$\text{Min} \quad Z = x_1^2+2x_2^2-0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2)+0.7$$

$$\text{subject to} \quad x_1, x_2 \in [-100, 100]$$

此問題的最佳解為$(x_1, x_2) = (0, 0)$，其$Z = 0$。此問題具有許多區域最佳解，因此並不容易找到全域最佳解。







來源：http://www.archimy.com/

1. 初始化及參數設定

- **搜索空間的下限與上限**$[L_d, U_d]=[-100, 100]$，**學習因子**$c_1=c_2=2$、**最小和最大飛行速度**$[v_{min}, v_{max}]=[-4, 4]$、**迭代次數** $t = 1$、**最大迭代次數** $T_{max}=500$ 、**慣性權重** $w=0.9$ 。
- **隨機產生** 10 **個粒子，初始化每個粒子的位置向量** x_i^t **和速度** v_i^t 。

位置向量x_i^1	速度v_i^1	當代適應值F_i
x_1^1 =(-99.7497, -61.3392)	v_1^1 =(0.5087, 2.4699)	F_1=17476.10
x_2^1 =(17.0019, -29.9417)	v_2^1 =(-0.1610, 3.1677)	F_2=2082.78
x_3^1 =(64.5680, -65.1786)	v_3^1 =(1.9728, 2.8715)	F_3=12666.20
x_4^1 =(42.1003, -39.2010)	v_4^1 =(0.1083, -3.8801)	F_4=4846.72
x_5^1 =(-81.7194, -70.5374)	v_5^1 =(-1.0844, 2.6728)	F_5=16629.70
x_6^1 =(97.7050, -76.1834)	v_6^1 =(-0.4345, -3.9627)	F_6=21155.30
x_7^1 =(-98.2177, -6.3326)	v_7^1 =(-0.9770, 0.5695)	F_7=9727.96
x_8^1 =(20.3528, -66.7531)	v_8^1 =(0.8573, 1.3044)	F_8=9327.58
x_9^1 =(-9.84222, -88.5922)	v_9^1 =(-1.1830, 0.8615)	F_9=15794.50
x_{10}^1 =(56.6637, 3.9976)	v_{10}^1 =(2.4209, -1.5844)	F_{10}=3242.42

2. 評估每一個粒子

- 測量每個粒子所在位置的適應值F_i^t。
  - 現為初始情況(t = 1)，將每個粒子目前的適應值設成該粒子的局部最佳值F_i^*；且將每個粒子的目前初始位置設成該粒子的局部最佳位置 p_i^t。

當代適應值F_i^1	局部最佳值F_i^*	局部最佳位置p_i^1
F_1^1=17476.10	F_1^* =17476.10	p_1^1 =(-99.7497, -61.3392)
F_2^1=2082.78	F_2^* =2082.78	p_2^1 =(17.0019, -29.9417)
F_3^1=12666.20	F_3^* =12666.20	p_3^1 =(64.5680, -65.1786)
F_4^1=4846.72	F_4^* =4846.72	p_4^1 =(42.1003, -39.2010)
F_5^1=16629.70	F_5^* =16629.70	p_5^1 =(-81.7194, -70.5374)
F_6^1=21155.30	F_6^* =21155.30	p_6^1 =(97.7050, -76.1834)
F_7^1=9727.96	F_7^* =9727.96	p_7^1 =(-98.2177, -6.3326)
F_8^1=9327.58	F_8^* =9327.58	p_8^1 =(20.3528, -66.7531)
F_9^1=15794.50	F_9^* =15794.50	p_9^1 =(-9.84222, -88.5922)
F_{10}^1=3242.42	F_{10}^* =3242.42	p_{10}^1 =(56.6637, 3.9976)

- 從所有粒子中找出當前群體最佳適應值極值 $\min(F_1^*, F_2^*, \ldots, F_m^*)$。若此極值較目前的全域最佳極值$F_g^t$為優，則將$p_g^t$設定為此極值所表示之粒子的所在位置，且更新全域最佳值。目前的群體最佳適應值為 F_g^1 =2082.78， 群體最佳位置為p_g^1 =(17.0019, -29.9417)。

3. 粒子的狀態更新

- 隨機產生亂數 $r_1 = 0.875973$與 $r_2 = 0.726676$。利用公式 3和公式 2對第一個粒子的速度和位置進行更新，並設 t = t+1。
  - 利用公式3可得知粒子1的新速度 v_1^2= (170.1390, 47.8545)。因為超過了最小和最大飛行速度[v_{min}, v_{max}]=[-4, 4]，故修正為[4.0, 4.0]。
  - 利用公式2可得知粒子1的新位置$x_1^2 = x_1^1 + v_1^2$ =(-95.7497, -57.3392)。
- 其它粒子的更新如表所示。

(r_1, r_2)	速度v_i^2	位置向量x_i^2
(0.875973, 0.726676)	v_1^2=(4.0000, 4.0000)	x_1^2 =(-95.7497, -57.3392)
(0.955901, 0.925718)	v_2^2 =(-0.1449, 2.8509)	x_2^2 =(16.8570, -27.0908)
(0.539354, 0.142338)	v_3^2=(-4.0000, 4.0000)	x_3^2 =(60.5680, -61.1784)
(0.462081, 0.235328)	v_4^2 =(-4.0000, 0.8659)	x_4^2 =(38.1003, -38.3351)
(0.862239, 0.209601)	v_5^2=(4.0000, 4.0000)	x_5^2 =(-77.7194, -66.5374)
(0.779656, 0.843654)	v_6^2=(-4.0000, 4.0000)	x_6^2=(93.7050, -72.1834)
(0.996796, 0.999695)	v_7^2 =(4.0000, 4.0000)	x_7^2 =(-94.2177, 2.3326)
(0.611499, 0.392438)	v_8^2=(-1.8585, 4.0000)	x_8^2=(18.4943, -62.7531)
(0.266213, 0.297281)	v_9^2=(4.0000, 4.0000)	x_9^2=(-5.8422, -84.5922)
(0.840144, 0.023743)	v_{10}^2=(0.2954, -3.0366)	x_{10}^2 =(56.9591, 0.9400)

4. 檢驗是否符合結束條件

- 當前的迭代次數 t =1，尚未達到預先設定的最大次數 T_{max} ，故跳回步驟 2。
  - 再執行一次步驟2，可求得t = 2的每個粒子之適應值F_i^2，同時並與該粒子先前的局部最佳值F_i^*做比較。若$F_i^2 < F_i^*$，則將該粒子的目前位置設成該粒子的局部最佳位置 p_i^t ，且局部最佳適應值$F_i^* = F_i^2$ 。如下表所示。
  - 從所有粒子中找出當前群體最佳適應值極值 $\min(F_1^*, F_2^*, \ldots, F_m^*)$。若此極值較目前的全域最佳極值$F_g^t$為優，則將$p_g^t$設定為此極值所表示之粒子的所在位置，且更新全域最佳值。目前的群體最佳適應值為F_g^2 =1752.58， 群體最佳位置為p_g^2 =(16.8570, -27.0908)。

(t=1)

局部最佳值F_i^*
F_1^* =17476.10
F_2^* =2082.78
F_3^* =12666.20
F_4^* =4846.72
F_5^* =16629.70
F_6^* =21155.30
F_7^* =9727.96
F_8^* =9327.58
F_9^* =15794.50
F_{10}^* =3242.42

(t=2)

當代適應值F_i^2	局部最佳值F_i^*	局部最佳位置p_i^2
F_1^2= 15744.70	F_1^* =15744.70	p_1^2 =(-95.7497, -57.3392)
F_2^2= 1752.58	F_2^* =1752.58	p_2^2 =(16.8570, -27.0908)
F_3^2= 11154.80	F_3^* =11154.80	p_3^2 =(60.5680, -61.1784)
F_4^2= 4391.51	F_4^* =4391.51	p_4^2 =(38.1003, -38.3351)
F_5^2= 14895.40	F_5^* =14895.40	p_5^2 =(-77.7194, -66.5374)
F_6^2= 19202.80	F_6^* =19202.80	p_6^2 =(93.7050, -72.1834)
F_7^2= 8888.90	F_7^* =8888.90	p_7^2 =(-94.2177, 2.3326)
F_8^2= 8219.06	F_8^* =8219.06	p_8^2 =(18.4943, -62.7531)
F_9^2= 14346.30	F_9^* =14346.30	p_9^2 =(-5.8422, -84.5922)
F_{10}^2=3246.79	**F_{10}^* =3242.42**	**p_{10}^2 =(56.6637, 3.9976)** ←**未改變**

- **上述步驟僅執行到PSO的第二次迭代。本範例執行到第359次迭代時，會找到此問題的最佳解p_g = (0, 0)， F_g = 0。但本範例無設定收斂精度ξ，故PSO不知道此解為最佳解，所以會繼續執行到T_{max}=500為止。**

■PSO特性

- 分散式多點搜尋
- 粒子具記憶性
- 結合廣域搜尋和區域搜尋
- 粒子的特點為位置與速度
- 適合在連續性的範圍內搜尋
- 只有少數的參數需要調整
- 可以被應用來解決大多數的最佳化問題

■ PSO與GA比較

- **相同**
  1. 屬於多點搜尋
  2. 群體隨機初始化
  3. 對群體內每一個體計算適應值(FitnessValue)
  4. 群體根據適應值進行複製
- **相異**
  1. PSO沒有遺傳操作-交換（Crossover）、突變（Mutation）而是根據自己的速度來決定搜尋
  2. PSO有記憶性 – 即根據粒子自身過去經驗與群體最佳解，交換正向資訊
  3. 較適合連續性最佳化問題，但亦適合離散性最佳化問題
  4. 所需的參數設定較少

■ PSO的應用領域

- 例如:系統設計、多目標最佳化、分類、型樣識別、生物系統模擬、排程、遊戲、機器人應用、決策制定、網路安全及路由選擇、神經網路訓練、模擬和識別等
- 而其相關的實例則包含了模擬控制器設計、工作排程、影像分割、語音識別、時間頻率分析、燒燙傷診斷、手勢姿勢識別和自動目標偵測等問題上

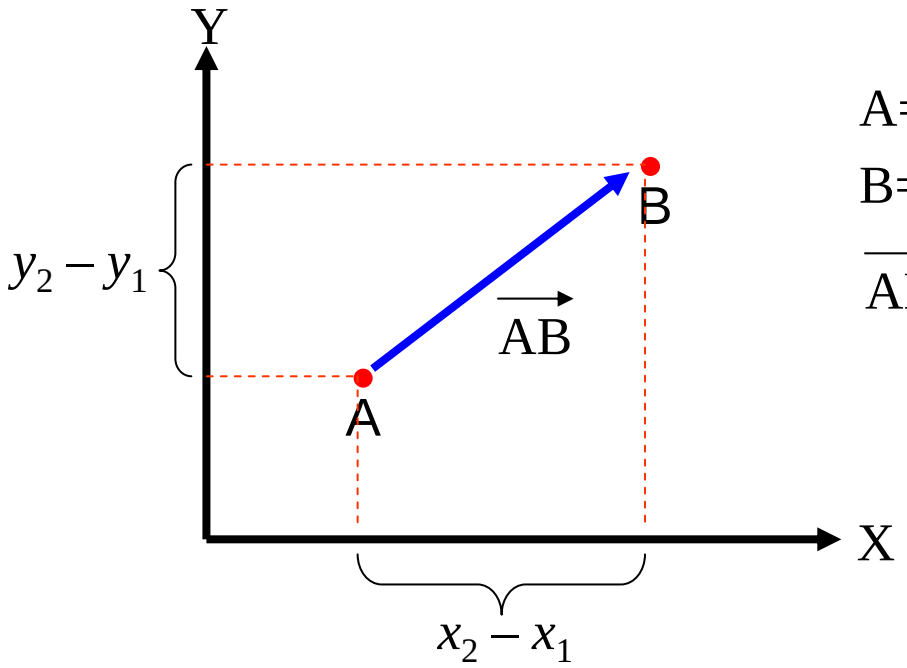
補　充

■向量概念回顧

- 在n維座標空間中，任兩點之間的向量為何？

Ans:

- (以二維座標空間為例)



$A=(x_1, y_1)$

$B=(x_2, y_2)$

$\overrightarrow{AB}$ = B-A

$= (x_2 - x_1, y_2 - y_1)$

- 向量：具有方向性與大小