第三章 理論介紹

本章節分別對操作規線、類神經網路、遺傳演算法作相關介紹。

3.1 操作規線介紹

在水資源調配中，操作規線為水庫在水資源調配中若每一個時刻(t)皆將整個水庫當下的所有水量(S_t+I_t)拿來運作，就長期營運的觀點這種操作策略不一定最佳。若能考慮水庫當下的可利用水量並配合當地水文豐枯的情形進行操作，應可發揮更大的調節效益，基於此乃有操作規線的操作方式。

一般常用的操作原則是將水庫蓄水空間從呆水位到滿水位分成數個區間如圖 3.1，STi(t)代表水庫至第 i 分層頂部水庫之水位(體積)，以下以石門水庫之規線為例作說明:

1. 水庫水位標高在上限以上時，依據計畫配水量供水，並得視各標的用水需求增加調配之。
2. 水庫水位標高在上限與下限之間時，依據計畫配水量供水。
3. 水庫水位標高在下限與嚴重下限之間時，依據計畫配水量供水為原則，為因應可能之持續枯旱，北水局得邀集相關單位預先協商配水量減供措施。
4. 水庫水位標高在嚴重下限以下時，農業用水依據計畫配水量百分之五十供水為原則，家用及公共給水、工業用水依據計畫配水量百分之八十供水為原則，其實際減供水量由北水局得邀集相關單位協商之。

故每一層皆對應至一供水標的量，若再配合當地水文狀況，則圖 3.1 分層對應之蓄水體積可依時間而變動，如圖 3.2 所示。

3.2 類神經網路

3.2.1 類神經網路簡介

類神經網路是一個可計算之多層網路，它使用大量簡單而具有平行處理能力之人工神經元來模擬人類學習行為，因此本章節將針對生物及人工神經元的構造、類神經網路的組成架構和基本理論，以及類神經網路中之倒傳遞網路(Backpropagation Network)詳加介紹。

3.2.2 生物神經元模型

生物神經網路是由巨量的神經細胞(neuron，又稱神經元)所組成，形成一個高度連結網狀的神經網路，資訊的處理工作即透過上述之連結來進行。以人腦而言，人腦大約由 10^{11} 個神經元所組成，而每一個神經元約有 10^{3} 根連結與其他神經元相連，所以人腦中約有 10^{14} 根連結，因此人腦可以儲存大量而複雜的知識。神經元構造如圖 3.3 所示，其主要構造如下：

1.神經核(soma):神經細胞的核心，為一呈核狀的處理機構。

2.神經軸(axon):神經細胞呈軸索狀的輸送機構。

3.神經樹(dendrites):神經細胞呈樹枝狀的輸出入機構。

4.神經節(synapse):神經細胞神經樹上呈點狀的連結機構。

當神經細胞透過神經節與神經樹從其它神經元輸入脈波訊號後，經神經核處理，產生一個新的脈波訊號，這個訊號再經過神經軸傳送到神經樹，再透過神經節與神經樹成為其它神經元的輸入脈波訊號，如果脈波訊號是經過興奮神經節(excitatory synapse)，則會增加脈波訊號的速率(pulse rate)，如果脈波訊號是經過抑制神經節(inhibitory synapse)，則會減少脈波訊號的速率。因此，脈波訊號的速率是同時取決於輸入脈波訊號的速率，以及神經節的強度。而神經節的強度可視為神經網路儲存資訊之所在，神經網路的學習即在調整神經節的強

度(葉怡成，1994)。

3.2.3 人工神經元模型

根據神經細胞的結構與功能，從 40 年代開始，先後提出的神經元模型有數百種之多，其中對於腦模型、自動機、人工智慧有重大影響的是 1943 年由美國心理學家 McCulloch 和數學家 Pitts 共同提出的形式神經元模型，同常稱之為 MP 模型(謝明富，1999)。此一模型（見圖 3.4 所示）具有將輸入變數與輸出變數間，複雜的內在對映關係充份呈現的功能，其為解決非線性動態問題的最佳工具之一。此模型由許多人工神經元所組成，神經元又稱為節點(node)或臨界值元件(threshold element)，其輸入端輸入各自之訊息，藉由各自權重加權總和後傳入節點，透過閥值的過濾，繼而經由轉換函數轉換後輸出，其數學表示式如下：

$$Y_j = f(net_j)\ ,\ net_j = \sum_i W_{ij} X_i - b_j \quad \text{.....(式 3.1)}$$

其中：

Y_j:為模仿生物神經元模型的輸出訊號。

f:為模仿生物神經元模型的轉換函數(transfer function)，將輸入值之加權乘積和轉換成處理單元輸出值。

W_{ij}:為模仿生物神經元模型的神經節強度，又稱連結加權值。

X_i:為模仿生物神經元模型的輸入訊號。

b_j:為模仿生物神經元模型的閥值(bias)。

net_j:為輸入值之加權乘積和。

n：為輸入訊號個數

類神經網路常用之轉換函數有下列四種（見圖 3.5 至 3.8(b)）：

1. 位階臨界轉換函數(Step Threshold Transfer Function)：臨界函數的輸出只隨輸入值的正副號所改變。
2. 線性轉換函數(Linear Transfer Function)：函數輸入值與輸出值呈線性關係。
3. 非線性轉換函數(Nonlinear Transfer Function)：函數輸出的最大值與最小值限制在一個特定的範圍內。
4. 臨界 S 型轉換函數(Sigmoid Threshold Transfer Function)：是最常被應用之函數，因為此種函數型態具有可微分且連續等性質，此特質使網路可以應用到非線性的學習領域中。常用函數有雙彎曲函數(式 3.2)與雙曲線正切函數(式 3.3)兩種。

$$f(x)=\frac{1}{1+e^{-x}} \quad \text{.....(式 3.2)}$$

$$f(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad \text{.....(式 3.3)}$$

3.2.4 類神經網路基本理論

類神經網路的架構乃仿照人類腦部神經網路的模型而發展出來的，一個類神經網路是由許多個人工神經元連結組成，並可以組成各種網路模式(葉怡成，1994)。

類神經網路的總體運作模式有兩種：(1)學習過程(Learning)－網路依學習演算法，從範例中學習，以調整網路權重值的過程，與(2)回想過程(Recalling)－網路依學習演算法，以輸入資料決定網路輸出資料的過程。

此外，學習過程的學習演算法又可分為三類：

(1)監督式學習

從問題領域中取得訓練範例(有輸入變數值，也有輸出變數)，並從中學習輸入變數與輸出變數的內在對映規則，以應用於新的案例(只有輸入變數值，而需推論輸出變數值的應用)。感知機網路、倒傳遞網路、機率神經網路、學習向量量化網路與反傳遞網路等五種類神經網路皆屬監督式學習。

(2)無監督式學習

從問題領域中取得訓練範例(只有輸入變數值)，並從中學習範例的內在集群規則，以應用於新的案例(有輸入變數值，而需推論它與那些訓練範例屬同一集群的應用)。自組織映射圖網路、自適應共振理論網路等兩種類神經網路皆屬無監督式學習。

(3)聯想式學習

從問題領域中取得訓練範例(狀態變數值)，並從中學習範例的內在記憶規則，以應用於新的案例(只有不完整的狀態變數值，而需推論其完整的狀態變數值的應用)。霍普菲爾網路、雙向聯想記憶網路等兩種類神經網路皆屬聯想式學習。

3.2.5 倒傳遞神經網路（BP）

在監督式學習模式中，由 Rumelhart 等學者於 1986 年所發展的誤差向後推導或稱倒傳遞學習演算法（簡稱 BP）是被廣泛使用的一種學習演算法，因為其具有學習及回想的功能，故可進行定率預測。一般倒傳遞網路可分為三部份(如圖 3.9 所示)，輸入層用以接受外在環境的訊息，其神經元數目則依問題而定；輸出層用以輸出訊息給外在環境，其神經元數目同樣依問題而定；隱藏層將輸入與輸出層各處理單元間的相互關係充份地表現出來，其神經元數目並無標準可決定。倒傳遞網路模式學習訓練方式由所探討問題中取得相當數量之訓

練樣本，並從樣本中應用向前饋入與誤差向後推導兩步驟推求輸入變數與輸出變數的內在對映規則，再應用回想功能，進行新案例之輸出變數值推估。下列為倒傳遞類神經網路（BP）之學習與回想過程建立步驟，並對本研究訓練倒傳遞類神經網路所採用之演算法作說明：

◆倒傳遞網路學習過程的建立

(1)令輸入層、隱藏層與輸出層節點分別以 i、j、k 為下標符號，建立一輸入層、隱藏層與輸出層節點數分別為 m、n、o 個，以均勻分佈隨機亂數設定加權值矩陣 W_{ij}、W_{jk} 與閥值 b_j、b_k 的初始值。

(2)輸入一個訓練案例之輸入向量與目標輸出向量。

輸入向量: $\vec{X} = [x_i]$ ， $i = 1,2,3...m$

目標輸出向量: $\vec{T} = [T_k]$ ， $k = 1,2,3...o$

(3)計算推估輸出向量 $\vec{O}$ 。

(a)輸入層向量

$\vec{X} = [x_i] \quad i = 1,2,....m$

(b)計算隱藏層輸出向量 $\vec{H}$

$$H_j = f_h(net_j) \text{ ， } net_j = \sum_{i=1}^{m} x_i w_{ij} - b_j \qquad j = 1,2,3.....n \qquad \text{.....(式 3.4)}$$

其中 $f_h(x)$ 為隱藏層轉換函數，可依問題型態挑選適當轉換函數。

(c)計算輸出層輸出向量 $\vec{O}$

$$O_k = f_o(net_k) \text{ ， } net_k = \sum_{j=1}^{n} H_j w_{jk} - b_k \qquad k = 1,2,3...o \qquad \text{.....(式 3.5)}$$

其中 $f_o(x)$ 為輸出層轉換函數，可依問題型態挑選適當轉換函數。

(4)計算加權值矩陣修正量 Δw ，及閥值修正量 Δb 。

因為監督式學習的目的在降低網路輸出單元目標輸出值與推論

輸出值之差距，所以一般以能量函數（又稱誤差函數）表示學習的品質：

$$E = \frac{1}{2}\sum_{k=1}^{o}(O_k - T_k)^2 \qquad \text{.....(式 3.6)}$$

因此網路的學習過程即為使能量函數最小化的過程，通常以最陡坡降法來使能量函數最小化，即每次所有輸入值和目標都提供給網路後，網路便更新權重值和偏權值，調整的幅度和誤差函數對該加權值的敏感程度成正比，即與誤差函數對加權值的偏微分值大小成正比：

$$\Delta w = -\eta \frac{\partial E}{\partial w} \qquad \text{.....(式 3.7)}$$

式中：

Δw:加權值調整的幅度。

η:學習速率(learning rate)，控制每次以最陡坡降法最小化誤差函數的步幅，其範圍為$0 \le \eta \le 1$，可視所需狀況自行設定範圍容許內之值。

經由一連串的代入及演算可推得以下各值：

(a)輸出層

$$\Delta w_{jk} = \eta(T_k - O_k)\cdot df_o(net_k)\cdot H_j \qquad \text{.....(式 3.8)}$$

$$\Delta b_k = -\eta(T_k - O_k)\cdot df_o(net_k) \qquad \text{.....(式 3.9)}$$

其中$df_0(x)$為輸出層轉換函數之一階導函數，舉例說明：若以雙曲線正切函數$f_o(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$為輸出層之轉換函數，其一階導函

數則為 $df_o(x)=\frac{4}{(e^x+e^{-x})^2}$ 。

(b)隱藏層

$$\Delta w_{ij}=\eta(T_k-O_k)\cdot df_o(net_k)\cdot w_{jk}\cdot df_h(net_j)\cdot x_i \quad \text{.....(式 3.10)}$$

$$\Delta b_j=-\eta(T_k-O_k)\cdot df_o(net_k)\cdot w_{jk}\cdot df_h(net_j) \quad \text{.....(式 3.11)}$$

其中 $df_h(x)$ 為隱藏層轉換函數之一階導函數。

(5)更新加權值矩陣與閥值

$$w_{new}=w_{old}+\Delta w \quad \text{.....(式 3.12)}$$

$$b_{new}=b_{old}+\Delta b \quad \text{.....(式 3.13)}$$

(6)重覆步驟 2 至步驟 6，直至收斂，即誤差不再有明顯變化且符合所要求的精度。

◆倒傳遞網路回想過程的建立

(1)讀入加權值矩陣與閥值矩陣。

(2)輸入一個欲作檢定或驗證的案例之輸入向量

$\vec{X}=[x_i] \quad i=1,2,....m$ 。

(3)計算推估輸出向量 $\vec{O}$ 。

(a)輸入層向量

$\vec{X}=[x_i] \quad i=1,2,....m$

(b)計算隱藏層輸出向量 $\vec{H}$

$$H_j=f_h(net_j)\ ,\ net_j=\sum_{i=1}^{m}x_i w_{ij}-b_j \quad \text{.....(式 3.14)}$$

其中 $f_h(x)$ 為隱藏層轉換函數，可依問題型態挑選適當轉換函數。

(c)計算輸出層輸出向量$\vec{O}$

$$O_k = f_o(net_k)\ ,\ net_k = \sum_{j=1}^{n} H_j w_{jk} - b_k \qquad k = 1,2,3...o \ \ldots\ldots\ldots\text{(式 3.15)}$$

其中 $f_0(x)$ 為輸出層轉換函數，可依問題型態挑選適當轉換函數。

◆ **訓練倒傳遞類神經網路之演算法**

在訓練最佳化技巧上可分兩大類，第一大類為一階梯度法，其中包含(1)梯度下降(2)最陡步階下降下(3)共軛梯度下降;第二大類為二階梯度法，其中包含(1)牛頓法(2)高斯牛頓法(3)擬牛頓法(4)Levenberg-Marquardt (LM)法。二階梯度的訓練法比一階梯度訓練法快 10~100 倍，在函數逼近問題中，Levenberg-Marquardt 法會有最快的收斂，同時具有最低的均方誤差，因此，本研究採用 Levenberg-Marquardt 演算法。

Levenberg-Marquardt 演算法不必計算牛頓法中的 Hessian 矩陣，當性能函數具有平方和的形式時，Hessian 矩陣 H 可以逼近成下式:

$$H = J^T J \ \ldots\ldots\ldots\text{(式 3.16)}$$

並用下式計算梯度

$$f = J^T e \ \ldots\ldots\ldots\text{(式 3.17)}$$

其中 J 為 Jacobian 矩陣，它包含網路誤差對於權重值和偏權值的一階微分，e 是網路誤差向量，定義如下:

$$\|e\|^2 = \sum_i \left[\left(X_i - \hat{X}_i \right)^2 + \left(Y_i - \hat{Y}_i \right)^2 \right] \ \ldots\ldots\ldots\text{(式 3.18)}$$

Jacobian 矩陣若用到傳遞技巧計算會比 Hessian 矩陣複雜，Levenberg-Marquardt 演算法是將 Hessian 矩陣逼近使用，即將牛頓法的步驟改成如(式 3.19)

$$W_{k+1} = W_k - \left[J^T J + \mu I\right]^{-1} J^T e \quad \text{(式 3.19)}$$

其中 W_k 代表在第 k 次疊代的參數向量(包含所有權重與偏權值)，I 為單位矩陣，μ 為常數。當 μ 越大，表示(式 3.19)具有較小的步階大小之梯度下降; μ=0 時，則剛好是使用近似化 Hessian 矩陣的牛頓法。

3-3 遺傳演算法

3.3.1 遺傳演算法之介紹

遺傳演算法(genetic algorithm)的理論基礎可回溯自 1859 年達爾文(Charles Darwin)的「物種起源」(On the Origin of Species by Means of Nature Selection)書中之「物競天擇，適者生存」演化及淘汰觀念。在這種由自然選擇的演化機制中，生物界中的任一個體會將其特徵繁衍到下一代，而生物特徵是由細胞內的染色體來決定的，也就是由基因所組成的基因鏈，由於不同特徵的生物對於環境的適應力不大一樣，基於自然生態之淘汰，適應力較高的個體會有較高機率與異性交配繁衍，就如同自然界中求偶期生物往往必須藉由爭鬥來展示自身的強健與否，方有機會得到另一半的青睞。基於此一理論，具有較佳適應力之染色體亦有較高之機率繁衍至下一代。

此外，多數生物之繁衍均以採用有性生殖方式，因此子代是由父母雙方之染色體組合而成，使得子代染色體具備與父代不同之適應力，可能會出現「歹竹出好筍」之情況。由於較能適應環境之生物，有較高之繁衍機率，因此可將整體族群引導向更適合生存之方向。經由長時間之繁衍，最後將演變至最能適應於此生態環境下的種族。最早是由密西根大學的 John Holland 教授將 這種自然界的選擇方法系統化並發展一可用之演算法，其在 1975 年於 Adaption in Natural and

Artificial System 文中所提出，發展出遺傳演算法搜尋技術的基本架構，並且由其學生 David Goldberg 成功地運用在工程問題上。之後，有許多研究亦證實了遺傳演算法在最佳化問題的求解上是十分有效率的，其有以下幾個優點：

1. 可優選連續（continuous）及不連續（discrete）的問題。
2. 優選的過程中，無須求得目標函數的導數。
3. 不同於單點搜尋方式，而是採用多點搜尋，其演算機制可跳脫局部最佳解（local optimum）。
4. 可以處理多變數的優選問題。
5. 具有隱平行運算的能力，若在平行電腦中，可大量節省運算的時間。
7. 演算優選的結果，可提供一組最佳解，而非只有單一最佳解。
8. 參數優選機制係以演化方式(複製交配突變)進行，此一方式不受問題函數型態之影響

以上的優點，使得我們發現當傳統的最佳化方法無法解決一個問題或得到令人滿意的優選結果時，遺傳演算法便是一個很有趣且擁有很大潛力去替代部份傳統的優選法。

對所有的問題而言，遺傳演算法並非都是一個最佳的方法，例如當在處理一具有凸函數型態，且僅有少量變數之問題時，一般傳統以微積分為基礎的搜尋法，即可比遺傳演算法快速的找到最佳解，對於一些簡單優選的問題，傳統的演算法亦都能很快的解決，然而當我們在處理實際的問題時，經常會遇見的是非凸函數且多變數型態或更複雜的問題，這是一般演算法不易解決的，而遺傳演算法就有解決此類問題的能力，且可得到近以全域最佳解。

3.3.2 遺傳演算法之架構

在利用 GA 求解最佳化問題前，需先決定目標函數(objective function)、設計變數(Design variables)和搜尋空間。接著將設計變數編碼成類似生物染色體的字串，字串中的各個位元相當於基因(Gene)，經由三大運算元─複製(reproduction)、交配(crossover)及突變(mutation)過程產生適合度(fitness)較佳的新群集，直到找到最佳值或達到收斂條件為止。其流程如圖 3.10 所示：

A. 定義編碼方式與族群數

使用者應依據變數的範圍與所需精度，定義染色體長度與變數上下限，若以下列範例為例，其變數 x 範圍為 0 與 31 之間，變數 y 範圍為 0 與 63 之間，染色體長度設定為 5 與 6。

以變數 x 為例，其染色體長度為 5，因此最小與最大之染色體分別為 00000 與 11111，二進位轉換為十進位之最大與最小分別為 0 與 31(=2^5-1)，代表染色體設定方式將整個變數區段切割為 31 段，因此在本問題中，變數精度為 1.0(=(31-0) / (2^5-1))。由此類推，變數 y 之染色體長度為 6，其最小與最大之染色體分別為 000000 與 111111，其十進位之最小與最大值分別為 0 與 63(2^6-1)，因此其變數精度亦為 1.0(=(63-0) / (2^6-1))。

因此若變數 x 之染色體長度設定為 10，亦即將解空間切割為 1023(2^{10}-1)個區段，變數精度約為 0.03(= (31-0) / (2^{10}-1))，因此我們可以知道，當染色體長度越長時(即 bit 數越多)，該變數之精度亦越大，然而這亦代表待搜尋之解空間亦越大，將耗費更多之計算時間與資源。染色體族群數訂為 4，整體族群如表 3.1 所示。

初始族群則以隨機函數決定每條染色體每個位元，因此各位元 0 與 1 之機率應為 50%與 50%。

例題：

$$\max \quad \sin\left(\frac{x}{3}\right)\times e^{-\frac{x}{10}}\times\cos\left(\frac{y}{4}\right)+\frac{x}{50} \qquad \text{.....(式 3.20)}$$

$$s.t. \quad 0\le x\le 31$$
$$0\le y\le 63$$

B. 選取

「物競天擇」理論中最重要的是較佳適應力的個體會有較高機率被選取，較差適應力的個題則有較低機率被選取，但是被選取機率差異又不至於過大，因此以下有三種最為常見之方法表現選取機制。

◆ 輪盤法(roulette Wheel Method)

如同射飛鏢至輪盤上，因此輪盤上所佔面積較大者，有較高機率被選取，但較低機率之染色體又不至於完全沒有機會被選取如圖 3.11 所示。其方法如下，將所有染色體適合度加總作為分母，因此下式之 P_i 為染色體 i 被選取之機率，PP_i 為累積機率，因此若 i=N 時，$PP_N = 1$，如表 3.2 所示。

$$P_i = \frac{fit_j}{\sum_{k=1}^{N} fit_k} \qquad \text{.....(式 3.21)}$$

$$PP_i = \sum_{j=1}^{i} P_j \qquad \text{.....(式 3.22)}$$

以均勻分布之隨機函數產生一個 0 至 1 之間的隨機值，若隨機值位於 PP_i 與 PP_i –1之間，即表示選定第 i 條染色體，如表 3.3 所示，將選定之染色體複製到子代，直到滿足該複製之總數。

此法之缺點為若部份染色體之與其他染色體適合度差距過大，則造成較小染色體被選取之機率趨近於 0，部份染色體之被選取機率極大，容易使得族群會有均質化的現象。另外，輪盤法也無法直接適用

於最小值問題與目標函數處於負值之問題，若欲使用輪盤法，必須適度轉換適合度。

◆ 排序選取法(rank selection)

為了改進輪盤法不適用於族群間適合度差異過大的缺失，因此以各染色體適合度排名為基礎，再配合輪盤法的概念進行"物競天擇"之選取。表 3.4 是以上列題目作為範例：

1.計算各染色體之目標函數值。

2.依據目標函數值之大小，計算各染色體之排名。

3.依據排名計算候選機率與累積機率。

4.產生 0 至 1 之均勻分布隨機函數值，依據該數值座落於輪盤之位置，即表示該染色體被選取。

◆比較選取法(Tournament selection)

比較選取法與前述兩種方法不同，不用計算輪盤機率，其是模仿自然界的生物彼此競爭情形，當某一個體的適應值愈高，其經由比較選取後，存活下來而被複製的機會愈高，此選取法有一好處，即是染色體被複製下來的機率與染色體間適合度的相對值大小無關而是取決於相對大小，因此較適合於個體間適合度值相對變化很大之問題。此外，若使用此一選取方法，最小值問題與涵蓋負值值域之問題均適合，無須修改其適合度函數，其演算步驟如下:

1.依每代染色體數目，設定一個合理的比較個數，假設為 2 個。

2.每次從母代隨機選取 2 個染色體，比較其適合度較優者則表示被選取，如表 3.5 所示。

C. 複製

將舊有群體依其對環境的適應程度加以繁殖，高適應力者有較高之機會被複製至子代，使下一代能更適應環境。本運算方式係模仿生物界之無性生殖，父代與子代之染色體型式不變，但是配合前述較佳染色體有較高機率被選取之特性，可以將較佳染色體在整體族群中，變為強勢染色體，如圖 3.12 所示。

◆ **菁英政策**

為確保已搜尋之族群最佳染色體必能延續至下一代，遺傳演算法內可以施以菁英政策，確保當代最佳之數個染色體必將被複製。若不施行菁英政策，當代族群最佳解雖有較高機率被選取，但是並不保證一定會被複製，因此有可能使得族群最佳解反而隨著演進而變差。

D. 交配

在「物競天擇」之世界中，前述複製機制佔有極大之比重，具有將較佳染色體變成更為強勢，但是此一運算機制並無法改變族群之成員，因為其特性僅單純複製父代之所有特性。然而交配機制類似生物界的有性生殖，父母雙方染色體之結合可以產生不同之組合，可能產生「歹竹出好筍」的情況，進而將族群整體進化提升。

◆ **單點交配**

在進行單點交配時，程式會先依亂數決定一個切斷點，利用這個切斷點，將原先挑選出欲進行交配兩個的染色體切成兩部分，再將切開的部分重新組合成一對新的染色體，範例說明詳見表 3.6。

◆ **雙點交配**

雙點交配的步驟與單點交配類似，唯一的不同處是在進行雙點交配時，程式會先依亂數決定兩個切斷點，利用這兩個切斷點，將原先

挑選出欲進行交配的染色體切成三部分，再將切開的部分重新組合成新的染色體，範例說明詳見表 3.7。

◆ **均勻交配**

使用均勻交配時，首先必須先產生一個和染色體長度相等的二位元向量，在這個二位元向量中，每一個位元均以均勻分佈之隨機函數決定此位元的值為 0 或 1，此一二進位陣列可稱之為模版(mask)。利用這個模版，可以決定染色體交配與否。若是在模版中的位元值為 1 時，此一位元之染色體則互換；反之，在模版中的位元值為 0 時，則不進行互換，範例說明詳見表 3.8。

E. 突變

在生物界中，突變機制在演化上亦扮演一個重要的角色，此一機制可能形成跳躍式地進化。在最佳化領域中，當最佳解侷限於某個局部解時，突變機制可以以飛越式的方式跳脫。在遺傳演算法的運作過程中，程式需要先定義一個突變率，代表整體染色體發生突變之機率，如同生物界中亦有其突變機率一般，因此透過隨機函數產生一組向量，向量長度為族群之染色體數目，亦即若向量中有數值小於定義之突變率，代表所屬染色體便會進行突變的程序。當選定特定染色體進行突變時，仍透過均勻分佈之隨機函數，隨機選定染色體特定位元，將此位元的值作 0 與 1 的互換，因此被染色體數值立即變換，等於將染色體體質作一立即的改變。

突變率之給定不宜過大，若給予過大之突變率則使得整體族群不穩定；反之，若突變率給予過小，則缺少了飛越性地跳脫行為，容易過早收斂。表 3.9 為染色體突變變化表，首先必須先決定突變率，為避免過早收斂與族群不穩定兩種極端狀況，本案例選定 0.1，由於本案例之族群數為 4，因此隨機產生四個 0 至 1 之間之隨機函數，由表

上所示僅有第一條染色體之突變隨機值小於突變率，因此僅有第一條染色體要突變。而突變之位元仍以隨機函數來決定，由於本案例之染色體長度為 11，因此從 1 至 11 之間隨機挑選任意位元，本表中挑選第 6 個位元，其原始數值為 1，透過突變機制後，則變換為 0；反之，若原始數值為 0，則變換為 1。

F. 解碼與計算適合度

當完成複製、交配與突變三項機制後，遺傳演算法已形成另一組子代族群，在此步驟則需對子代族群之二進位染色體，透過解碼動作轉換為實數數值，在此根據前述之解碼動作，將二進位染色體轉換為變數值，其為實數值。將此實數變數值代入式 3.20 函數中，可以求得最後一欄之目標函數值，結果如表 3.10。

G. 收斂條件

遺傳演算法屬於啟發式演算法之一員，一般而言，若尚未求得全域最佳解，啟發式演算法永遠有機會往全域最佳解收斂，因此最常見之收斂條件，各代族群最佳解若維持一定代數不變，可視為求得全域最佳解，停止遺傳演算流程。

另外，設定一最大運算代數亦為常見之收斂條件，通常若遺傳運算雖不斷有所改善，但其改善幅度十分微小，即可避免過多無謂之計算時間。