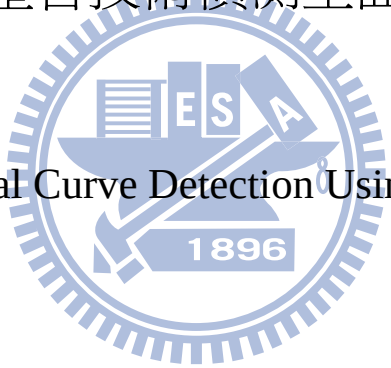
國立交通大學

多媒體工程研究所

碩 士 論 文

使用叢集整合技術偵測主曲線之研究



The Study of Principal Curve Detection Using Cluster Ensembles

研 究 生：林昇毅

指導教授：王才沛　教授

中華民國九十八年八月

使用叢集整合技術偵測主曲線之研究

The Study of Principal Curve Detection Using Cluster Ensembles

研 究 生：林昇毅　　Student：Sheng-Yi Lin

指導教授：王才沛　　Advisor：Tsai-Pei Wang

國 立 交 通 大 學

多 媒 體 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

August 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年八月

使用叢集整合技術偵測主曲線之研究

學生:林昇毅　　指導教授:王才沛

國立交通大學多媒體工程研究所 碩士班

摘要

主曲線是通過資料中心的一條線，因此找到一個圖形的主曲線即可以得到一個圖形的基本形狀。在過去的研究當中，利用叢集化演算法來找尋主曲線在圖形識別的領域中是一個很熱門的議題，其中大部份的演算法大都分爲三個主要的步驟: 叢集化演算法找到分群結果、連接由第一步驟所得到的各個分群獲得一個初始的主曲線、對初始主曲線作平滑化。然而這些過去所提出的演算法中，都因爲叢集化演算法先天的一些限制而有所美中不足的地方，例如如何得到適當的分群個數，初始條件以及雜訊量對於叢集化過程的影響，等等。在此我們想引入一種技術--叢集整合技術，利用叢集整合的特性來降低初始件以及雜訊量對於叢集化過程的影響，以此得到更穩定的叢集化後的分群結果，再將此分群結果利用階層聚合演算法來得到最終的分群結果。

The Study of Principal Curve Detection Using Cluster Ensembles

Student：Sheng-Yi Lin　　Advisor：Tsai-Pei Wang

Institutes of Multimedia Engineering

National Chiao Tung University

ABSTRACT

A principal curve is a curve that passes through the middle of the data distribution. As a result, we can obtain the basic shape of a data distribution by detecting its principal curve. The detection of principal curves through clustering algorithms has been a popular topic in past research. Most of these algorithms consist of three main steps: a clustering algorithm to partition the data, the linking of the clusters to obtain an initial principal curve, and the smoothing of the initial principal curve. However, these algorithms all have some limitations due to the underlying clustering algorithms. Examples of such limitations include how to determine a proper number of initial clusters, initializations, and the effect of noise, etc. In this thesis, our goal is to apply the technique of cluster ensembles to principal curve detection. The benefit of cluster ensembles is the reduced effect of initialization and noise, and this leads to more stable clustering results. The final partition into principal curves are obtained using hierarchical agglomeration algorithms.

誌謝

這篇碩士論文能夠順利完成，首先我要感謝我的指導教授王才沛老師在當完兵後重回學校的這兩年碩士生涯，對於我耐心的指導以及教誨，當所做的研究遇到瓶頸和困難，老師總會給予適當的建議或是可能的解決方向，讓我對於圖形識別有更深一層的領悟與實作的經驗，還有在平時對於論文上專業的學問都能給我們很清楚的觀念和說明，讓我能一步步的建立起完整的觀念以及培養我能有穩固的基礎，另外要特別感謝學校--國立交通大學以及資訊工程研究所的所有教授們，學校對於學生們毫不吝嗇的給予豐富的資源，還有資訊工程研究所的所有教授，以他(她)們在專業領域上的專才以及豐富的知識不 虞餘 力給學生指導，並能在實作的作業上，讓我不只學習到理論與實作並重的觀念，更讓我學習到團隊合作，與一個團隊協調的重要性。

再來要首先要感謝的是我的家人，尤其是我的父母親，就算家中不算富裕仍堅持讓我完成更高的學業，更在就學的途中給我鼓勵以及支持，還有要感謝我的姊姊與妹妹，在就讀碩士的這段期間代替我照顧父母親，跟給予家庭經濟上的幫助，最後要感謝的是實驗室的同學、學長、以及學弟們，讓我在碩士的求學生活裡增添了一份色彩，讓我能在這段期間能不怕辛苦的成功走完，在此衷心的謝謝你們跟指導教授王才沛老師。

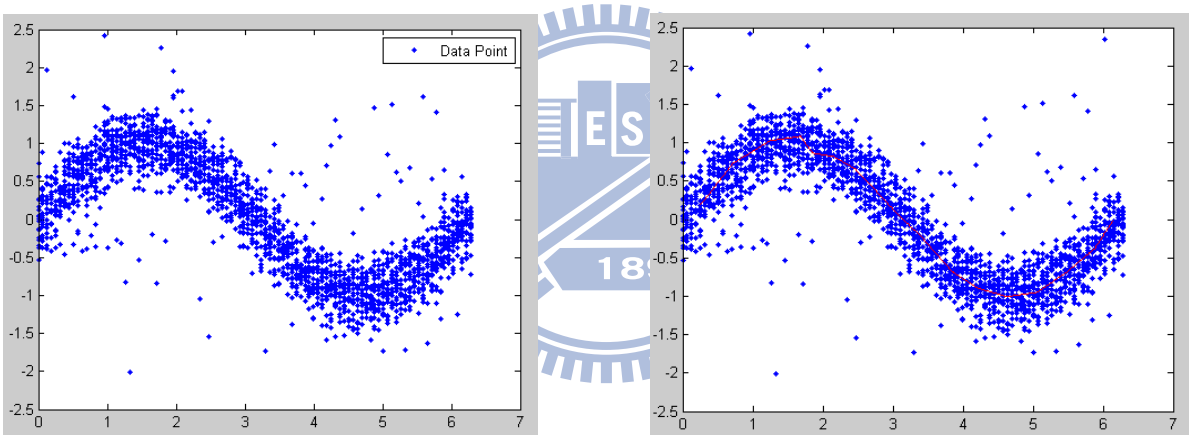
目錄

圖目錄

第一章 簡介

1.1 研究動機

利用叢集化演算法來找尋一個曲線圖形的主曲線，在圖形識別的領域中是一個熱門的研究方向，它可以廣泛的被應用在許多領域上。找到一個圖形的主曲線就等於是找到圖形的主要形狀，例如下圖 1 中的(a)在一堆資料點有一個主要的圖形物件，而一條通過這個圖形物件的線即是它的主曲線，正如圖 1(b)中所視的紅色線段。



(a) (b)

圖 1: 通過圖形物件的主曲線。(a)資料集合、(b)紅色線段爲主曲線。

利用叢集化演算法來找主曲線，有幾個主要的優點: (1) 圖形的資料通常是一個很大的資料集合，利用叢集化演算法可以很容易的將圖形的資料用個數遠小於資料點的叢集來表示，這是個很好的降低資料量的方法。(2) 利用這些叢集做適當的連接即可得到圖形的主曲線。然而由於叢集化演算法先天的一些限制造成了利用叢集化演算法找尋主曲線的一些限制，例如對於一個欲找尋主曲線的圖形，如何去定義出適合它的叢集化過後的重心點數，意即如何去定義出合適此圖形的代表點個數。在自然界中的資料，通常

都有雜訊的影響，因此所提出的叢集化演算法找尋主曲線是否能從一個有雜訊影響的環境中，去找尋到我們所想要的圖形主曲線。還有初始條件對於叢集化過程的影響，可能造成叢集化過後的重心點位置不佳而無法順利的找到主曲線。所以叢集化找主曲線的演算法有許多，但都有其對於資料集合的特性以及能找到的圖形資料集合特徵有所限制。

在 [6] 所提出的演算法中，主要分爲三個階段(1)利用模糊 C 均值叢集化演算法將資料集合分群，得到最後重心點，以這些重心點來代表此一資料集合。(2)分析各個重心點之間的相互關係並且以重心點的相關性來做重心點與重心點的連接，去除掉連接後所產生的迴圈，最後得到圖形的初始形狀。(3)利用所提出的新平滑叢集化演算法對第一個階段所得的重心點位做重新叢集化，得到最後的平滑圖形主曲線。

雖然所提出的演算法能讓圖形資料集合從有雜訊的環境中，正確的取出圖形的初始形狀，但仍然會出現錯誤的情況。當第一階段的由模糊 C 均值叢集演算法所得到的最後分群結果，若無法正確的判斷出此欲找尋的圖形物件是封閉的或是展開的圖形物件，進而造成錯誤判斷的話，則無法從第二階段得到正確的結果。這是由於叢集化演算法受到雜訊的影響，無法正確的收斂到欲找尋的圖形資料上，而造成 所提出的演算法對於有模糊的封閉或是展開觀點的圖形物件會得到有時正確而有時不正確的結果。

[6] 的演算法的另一個問題是無法從具有相交或是有轉角特徵的圖形物件中，或是非單一圖形物件的資料集合中，得到圖形的主曲線。圖 2 爲由 [6] 所取出，圖 2(a)爲一個人字形的資料集合，它包含了一個頂端的轉角。圖 2(b)中爲一個阿拉伯數字的 8，它包含了一個相交的部份。圖 2(c)爲一個中字形的資料集合，它同時包含了轉角以及相交的部份。雖然作者在論文中有提出解決的方法，不過仍然是在實驗的階段。或是在作者的另一篇文獻 [7] 中，有提出更優於 [6] 的叢集化找主曲線的演算法，但在 [7] 中所提出的演算法，仍然只適用於找出單一圖形物件的資料集合。

在本論文中，首先會選用一種叢集化找尋主曲線的演算法，再針對此叢集化演算法其對於某種資料集合的限制做改進。而我們將採用叢集整合技術企圖去找到一個更穩定且更能增加正確性的叢集整合演算法，使得叢集化找尋主曲線能更廣泛的適用於各種資料集合。並且能降低叢集演算法對於初始條件或是參數值的影響，以利去得到一個適用於廣泛情況的叢集化找尋主曲線演算法。

(a) (b) (c)

圖 2: 含有轉角或是相交特徵的圖形資料集合。(a)人字形圖形物件、(b)阿拉伯數字 8 圖形物件、(c)中字形圖形物件。

1.2 章節概要

我們在第二章部份首先介紹現有的主曲線的偵測與叢集整合演算法的文獻，而在第三章介紹本篇論文所提出的利用叢集整合技術偵測主曲線的演算法，第四章爲實驗的部份，第五章爲結論與未來的展望。

第二章 文獻探討

2.1 主曲線的偵測

文獻 [1] 中提出了一個堅固的主曲線偵測演算法，而這個主曲線的偵測是針對存在複雜圖像中的主曲線，在文獻中作者將這樣由軸線所構成的圖像分爲兩個階段來做處理，一開始由點跟邊來表示圖像所形成的爲基本圖形(Basic Graph)而針對基本圖形做去除冗餘邊的處理後所形成的爲超級圖形(Super Graph)，再以超級圖形來做主曲線的偵測。雖然作者在文獻中實驗的部份顯示出了所提出的演算法堅固且有效率的數據，但文獻中假定主曲線爲一段長的線段，這會使得無法去判別出分離的主曲線。

在 [2] 中作者提出了一個新的演算法去偵測主曲線，因爲一個資料點群中其主曲線通常含有自我一致性的性質(Self-Consistency)，所以作者利用投影的方式找到初始的投影點集合，作者稱投影點爲主軸原點(Principal Oriented Point)，再由這些投影點集合產生出主軸圖形(Principal Graph)並且利用最小生成樹(Minimum Spanning Tree)演算法來得到主曲線。在實驗的部份中顯示出了可以正確的找到手寫阿拉伯數字、部份的小寫英文字母、以及中國字，雖然此文獻所提出的演算法在某些步驟並無法確保收斂。但作者強調其不同於其它演算法是在作者是以由下到上(Bottom-Up)的建構方式來找到主曲線，不同於其它的是以由上到下(Top-Down)的建構方式。

而文獻 [3] 中仍是使用資料點的投影來產生主軸點集合，並且使用機率模型(Probability Model)來估計和分類雜訊，使得仍然可以在雜訊的環境中找主曲線。不過作者所提出的方法中有許多自由變數是必須的，造成計算複雜度上過高。而在 [4] 中提

出了利用多邊形邊(Polygonal Lines)來架構出主曲線，並使用最小化資料點與多邊形軸線(Polygonal Curve)之間的距離限制來達到收斂，但與 [2] 相同都存在著某些步驟並無法確保收斂的問題。

另外文獻 [5] 中，是利用增加的方法去找到主曲線，在第一個部份慢慢的去增加多邊形的線段(Polygonal Lines)，直到門檻值被滿足。這些多邊形線段必須滿足區域性的最小值，使得它的成本函數爲最小因此而收斂。雖然能減少傳統叢集化演算法其每一個更新輪迴重新計算所有重心點位置的計算量，但卻無法從有雜訊影響的資料集合中，找出正確的主曲線。

叢集化演算法偵測主曲線可以分爲兩個主要的部份，第一個部份爲利用叢集化演算法求得輸入資料集合的代表點。第二部份爲如何將這些代表點適當的連結起來找到一條或是多條代表此一輸入資料集合的主曲線。大部份的文獻其第一個部份所選用的叢集化演算法，大都以模糊 C 均值叢集演算法爲基礎，例如文獻 [6] 中所提出的叢集化演算法偵測主曲線的演算法，便是使用模糊 C 均值叢集演算法來找到輸入資料集合的代表點。在第二部份提出了兩兩代表點之間的相互影響係數做爲找出適當連結的方法，最後平滑化所找出的初始主曲線來得到最終的主曲線。雖然所提出演算法也能在高維度的資料集合以及在雜訊的環境中得到不錯的結果，但它仍有其限制在，例如它並無法去找出含有轉角或是含有交叉的資料集合，因爲這樣的資料集合無法用文獻所提出的方法去找出代表點的順序，等等。

另外在 [7] 中，作者爲了減少初始設定的叢集演算法重心點對於所輸入資料集合而言過多，以致於所求得的資料集合代表點造成第二部份適當連結的錯誤，因而提出了改進的模糊 C 均值叢集演算法。它可以消除多餘的重心點並且防止重心點太近，因而造成錯誤的重心點相連結的問題。第二部份提出新的代表點之間的相互影響係數的計算

方式，以及利用最小生成樹來找到適當的連結，改進了文獻 [6] 中所提出的演算法無法找到具有轉角或是含有交叉的資料集合。

但文獻 [7] 所提出的演算法在初始的假定所有的輸入資料集合中都只含有一個圖形的物件，它並無法去決定出含有兩個以上的圖形物件。下圖由文獻 [7] 中取出，由圖 3: (a)可以看出來此一資料集合正確的是包含了兩個圖形物件，而由圖 3 (b)是從文獻 [7] 中所提出的演算法所得到的結果，輸入的資料集合只視爲一個圖形物件被找出。

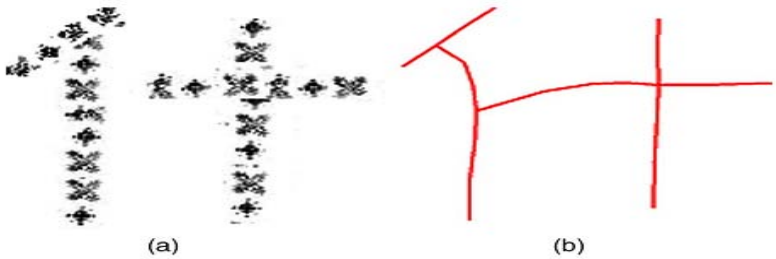


圖 3: (a)含有多重物件的資料集合。(b)由[2]所提出的演算法所得到的結果。

文獻 [8] 中，其第一個部份所使用的叢集化演算不同於之前文獻所提出的，它所使用的爲 GK 叢集化演算法，它與模糊 C 均值最大的不同點在於其每一個雛型都包含了一個重心點位置以及一個相關性變異矩陣，所以每一個重心點雛型都有其各自的體積，使得每一個重心點雛型都包含有其區域性的資料散佈資訊。而在第二部份利用 Bhattacharya 距離函數來當作兩兩重心雛型之間的距離測量公式，並在最後使用了最小生成森林演算法來找出重心雛型之間適當的連結。它可以成功從一個含有多個圖形物件的資料集合中找出所有的圖形物件，但必須在這些圖形物件是分離的狀況下，若圖形物件是重疊的狀態時，則無法正確的決定出所包含的圖形物件。

2.2 叢集整合演算法

叢集整合大致上可分爲兩種，硬性的叢集整合演算法以及軟性的叢集整合演算法，

其中軟性的叢集整合演算法又可以視爲硬性叢集整合演算法的一個延伸。軟性的叢集整合演算法與硬性叢集整合演算法最主要的差異是在軟性的叢集整合演算法，其整合性的資料結構是以整合機率式的個別分群的方式來獲得。

叢集整合演算法主要可以分爲三個步驟: 產生分群的方法、將數個個別的分群結果使用一個資料結構來表示、利用此一整合性的資料結構來得到最終的分群。文獻 [9] 中提出如何去整合一個軟性的叢集化演算法。首先利用模糊 C 均值來當作爲產生分群的方法，並且提出兩兩重心點之間相似值的測量公式，利用此一公式來得到一次個別分群，其所有重心點的相互關係矩陣，由此來整合數次個別分群的方法爲將數次個別分群的相互關係矩陣總合起來取平均，而所得到的這個平均的相互關係矩陣，即爲整合性的相互關係矩陣。最後再選擇一個一致性的函數來從這一個整合性的相互關係矩陣來得到最終分群的結果。在這一篇文獻中，主要是在提出一個軟性的叢集整合的方法，並且在文獻中針對軟性與硬性的叢集整合方法做比較，顯示出軟性的叢集整合優於硬性的叢集整合。

文獻 [10] 中，提出了硬性的叢集整合演算法，首先利用 C 均值來當作產生個別分群的方法，並且使用統計的方式來測量兩兩重心點之間的相似性，以此來得到個別分群的相互關係矩陣。最後作者提出了一個新的一致性函數: K 型演算法(K-Modes Algorithm)來求得整合性的相互關係矩陣其最終分群，並且在最後實驗的部份比較了 K 型演算法跟其它數個現有的一致性函數的效率，在實驗中顯示出 K 型演算法比現有的一致性函數有較佳的時間複雜度。

文獻 [11] 提出了一個基於 C 均值的證據累加的叢集整合演算法，它是屬於硬性的叢集整合演算法，利用 C 均值叢集演算法來得到單一次的個別分群、並提出一個新的叢集整合方法: 證據累加去求得整合性的相互關係矩陣，最後利用階層聚合中的單一連

結演算法，並且使用最大生命週期條件來得到最終的分群。在此文獻中，作者更使用了NMI (normalized mutual information)來測量在不同的整合次數下其所求得結果的正確率。並且在實驗的部份取了數種叢集整合演算法，以及叢集化演算法，針對數種資料集合，做由叢集化演算法所得到個別分群與叢集整合演算法的比較，顯示出叢集整合演算法有較好的正確率。

另外在 [12] 中提出了數個模糊式的叢集整合演算法，說明模糊式的叢集整合演算法，不僅保有模糊式叢集化演算法的彈性，同時也保有叢集整合的穩定性與較高的正確性。而文獻 [13] 說明了如何在高維度的資料集合下做叢集整合演算法。文獻 [14] 說明叢集整合演算法非常適合於資料量大的資料集合，因爲叢集整合演算法非常適合用來做分散式計算。

第三章 叢集整合技術偵測主曲線

在下圖 4 中可以看到我們所提出的叢集整合技術偵測主曲線主要的流程。首先我們先由 GK 叢集演算法獲得代表資料集合的雛型，獲得第一階段的分群的結果，再從這些代表資料集合的雛型找出它們之間的相似值，並利用階層聚合演算法得到第二階段的分群，而這個第二階段分群即爲叢集整合產生個別分群的方法。下一個利用叢集整合技術整合數次這樣的個別分群來得到第三階段的分群結果。最後取出第三階段分群其個別分群的資料集合，將這個資料集合當成一次 FCT(fuzzy curve tracing) [6] 的輸入，利用修改的 FCT 演算法獲得單一個別分群的主曲線。最後整合數段的單一個別分群的主曲線來獲得最終的結果。

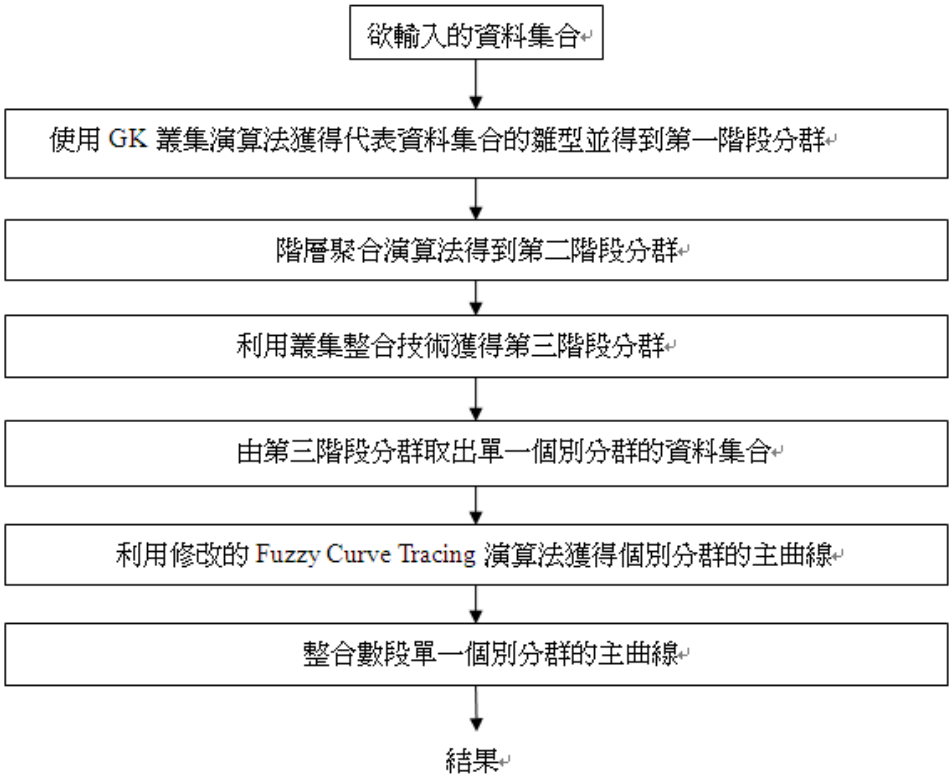


圖 4: 流程圖

下圖 5 是圖 4 流程圖每階段所產生的結果的一個例子，(a)爲由 GK 叢集演算所得到的第一階段的分群、(b)爲階層聚合所產生的第二階段的分群、(c)叢集整合後所得到的第三階段分群、(d)爲個別分群的資料集合、(e)爲(d)的主曲線、(f)爲整合主曲線後的結果。

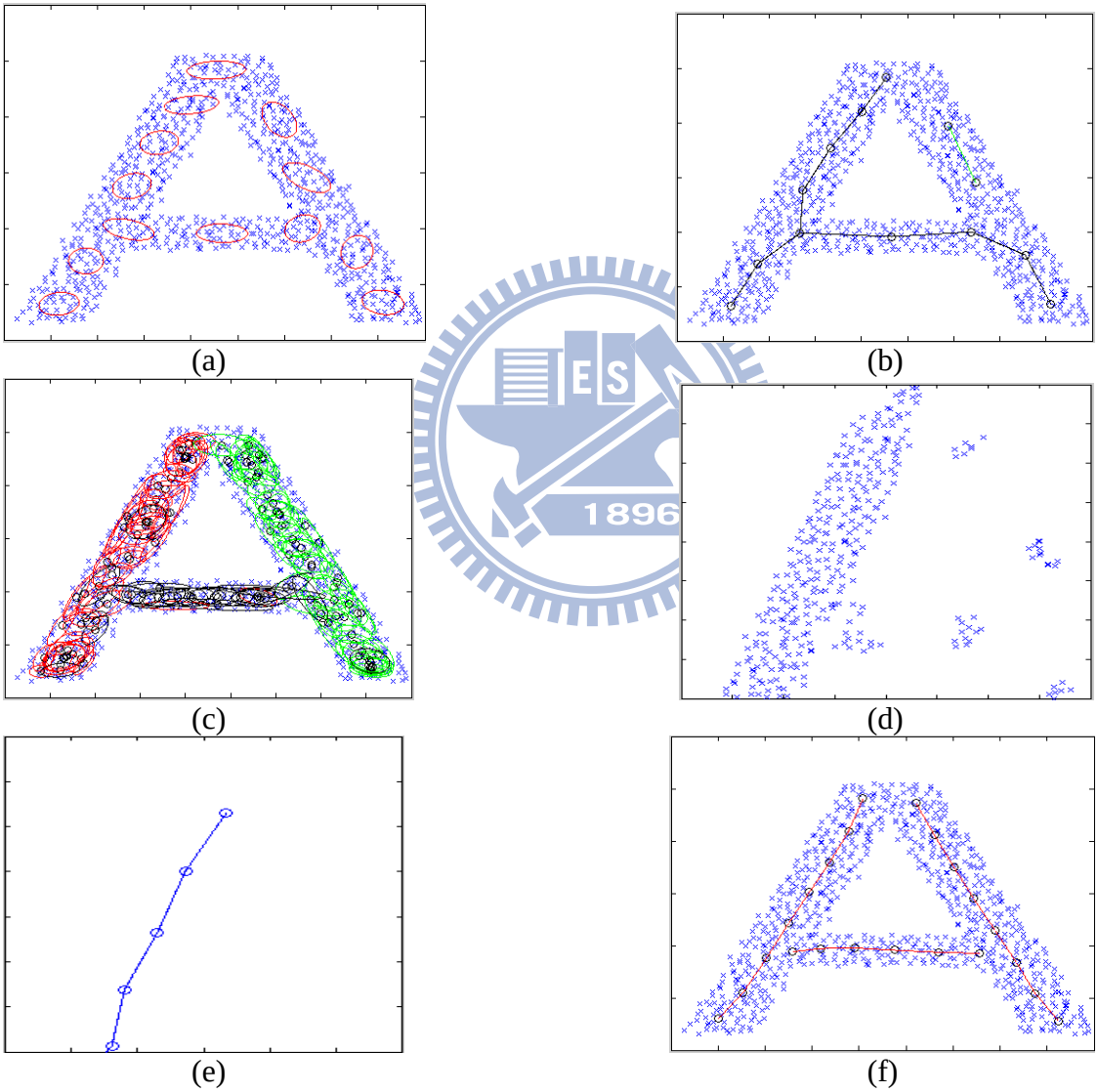


圖 5: 流程圖 4 各個步驟所產生的結果。(a)第一階段分群結果、(b)第二階段分群結果、(c)第三階段分群結果、(d)紅色個別分群的資料集合、(e)由 FCT 從圖(d)所得到的主曲線、(f)整合數段所得到的結果。

3.1 Gustafson-kessel 叢集化演算法

Gustafson-Kessel(GK)叢集化演算法可以視爲 C 均值或模糊 C 均值叢集化演算法的一個加強版本，它仍然可以分爲硬性的以及軟性的 GK 叢集化演算法，而 GK 叢集化演算法與模糊 C 均值叢集化演算法最大的不同點在於模糊 C 均值叢集化演算法的雛型(cluster prototypes)爲一個重心點位置。但 GK 叢集化演算法的雛型是一個具有體積(volume)的雛型，對於一個由 GK 叢集演算法所得到每一個雛型來說，都包含了一個相關變異矩陣(covariance matrix)以及重心位置。在 GK 叢集化演算法的每一個步驟的更新將會同時去更新重心點位置以及相對應的相關變異矩陣，這樣使得每一個雛型對資料集合具有分群的重心點位置，以及在這個重心點位置上資料集合的分佈資訊。

有了資料集合在重心點位置上的資料分佈資訊，便可以得到資料集合在此重心點位置上的資料的分佈方向，以利我們去分辨含有多重主曲線的資料集合。GK 叢集化演算法與模糊 C 均值叢集化演算法一樣，是屬於成本函數最佳化的叢集化演算法，其成本函數如下:

$$J(Z;U,V,\{A_i\}) = \Sigma_{i=1}^{K}\Sigma_{k=1}^{N}(u_{ik})^m D_{ikA_i}^2 \tag{1}$$

其中各項的參數定義如下：$Z=[x_1,...,x_N]\in R^{n\times N}$ 爲一個資料集合，其中變數 $x_k \in R^n, 1\le k\le N$ 爲資料點、$U=[u_{ik}]\in[0,1]^{K\times N}$ 爲資料集合的模糊分割矩陣，其中 u_{ik} 爲資料點 x_k 對於重心點 v_i 的模糊相關係數、而叢集化分群的重心點定義爲 $V=[v_1,v_2,...,v_k], v_i\in\Re^n, 1\le i\le k, 1\le k\le N$ 、$m\in[1,\infty)$ 爲模糊運算子、A_i 爲重心點位置 v_i 的距離函數矩陣，其距離函數的定義如下:

$$D_{ikA_i}^2 = (x_k - v_i)^T A_i (x_k - v_i) \tag{2}$$

而

$$v_i = \frac{\sum_{k=1}^{N} (u_{ik}^{(l-1)})^m x_k}{\sum_{k=1}^{N} (u_{ik}^{(l-1)})^m}, 1 \le i \le k \tag{3}$$

爲叢集重心點的位置更新公式，在每一次的更新循環後因爲模糊相關係數的改變，因此叢集重心點的位置也要隨著目前的模糊相關係數改變到適當的位置。

$$A_i = \frac{\sum_{k=1}^{N} (u_{ik}^{(l-1)})^m (x_k - v_i^{(l)})(x_k - v_i^{(l)})^T}{\sum_{k=1}^{N} (u_{ik}^{(l-1)})^m}, 1 \le i \le k \tag{4}$$

爲計算各別叢集重心點的相關性矩陣，

$$D_{ikA_i}^2 = (x_k - v_i^{(l)})^T \left[\rho_i \det(A_i)^{1/n} A_i^{-1} \right] (x_k - v_i^{(l)}), 1 \le i \le k, 1 \le k \le N \tag{5}$$

爲 GK 叢集演算法所使用的距離計算公式，其中的 n 爲資料點的維度、ρ_i 爲叢集重心的體積(它的值通常設爲 1)，而

$$u_{ik}^{(l)} = \frac{1}{\sum_{j=1}^{k} \left(\frac{D_{ikA_i}}{D_{jkA_j}} \right)^{2/(m-1)}} \tag{6}$$

爲利用(5)所得到的距離來求算出叢集重心點與資料點之間的相關係數。GK 叢集化演算法如下:

For $l = 1,2,\ldots$

步驟 1: 使用(3)計算 v_i

步驟 2: 使用(4)計算 A_i

步驟 3: 使用(5)計算 $D_{ikA_i}^2$

步驟 4: 更新分割矩陣

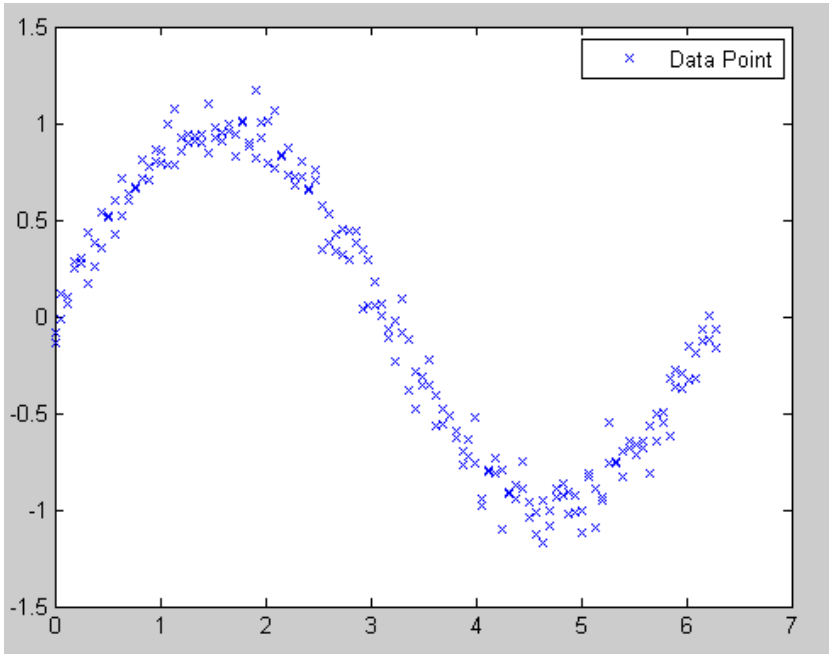
For $1 \le k \le N$,

使用(6)計算 $u_{ik}^{(l)}$

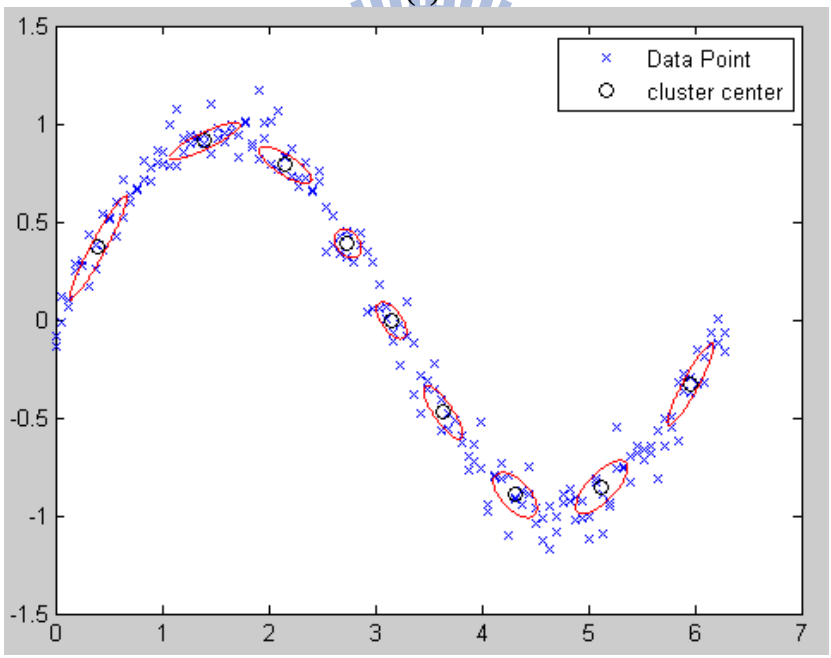
End

Until $\left\| U^{(l)} - U^{(l-1)} \right\| < \varepsilon$

虛擬碼其中的 ε 爲收斂的門檻值，而 $\left\| U^{(l)} - U^{(l-1)} \right\|$ 爲目前減掉前一次的模糊係數的變化量，當此一變化量小於門檻值則停止迴圈表示已達收斂狀態。下圖 6(a)爲所輸入的資料集合，而(b)爲利用 GK 叢集化演算法所得到的第一階段分群的結果，可看出每一個雛型不僅包含了重心點位置也包含了體積。



(a)



(b)

圖 6: 利用 GK 叢集化演算法求得資料集合代表點。(a)輸入的資料集合。(b)第一階段分群的結果。每一個 GK 雛型都含有資料散佈的方向性。

當相關性變異矩陣爲 singularity 時，則在上述所提到的演算法中的步驟 3，必須要去求得相關性變異矩陣的行列式，將會造成計數上的錯誤，這是因爲相關性變異矩陣是 singularity 矩陣時，並無法去求得此相關性變異矩陣的行列式。而造成相關性變異矩陣

爲 singularity 矩陣有幾個可能的問題: 當所輸入的資料集合的資料點呈現線性的時候，或者當重心點的初始條件給的太多時會造成過度對應，使得某些重心點只能分配到極少的資料點，造成此一重心點的相關性變異矩陣變成爲 singularity 的矩陣。

在此我們使用文獻 [15] 所提出的方法，當相關性變異矩陣爲 singularity 矩陣時，則它的最大的特徵根與最小的特徵根的差值會相當的大，所以將以此來判斷某一個相關性變異矩陣是否爲 singularity 矩陣，並且在計算相關性變異矩陣時，多加了整體性的相關性變異矩陣。這樣是爲了減少個別的相關性變矩陣太區域性造成某一個重心點只被分配到極少數的資料點，進而造成相關性變異矩陣爲 singularity 矩陣的情況。因此在 [15] 所提出的改進 GK 叢集化演算法中，當某一個相關性變異矩陣被判斷爲 singularity 矩陣時，此一相關性變異矩陣將被重新設定它的最大的特徵根與最小特徵根的比值。所提出的改進 GK 叢集化演算法其主要的修改部份如下:

新的參數定義如下: β 爲判斷最大的特徵根與最小的特徵根的比值是否造成相關性變異矩陣爲 singularity 矩陣的條件參數、γ 爲所加入的整體性相關性變異矩陣的比值、A_0 爲整體性的相關性變異矩陣。最主要的修改部份在於演算法的步驟二多加上一個比率的單位矩陣:

$$A_i = (1-\gamma)A_i + \gamma \det(A_0)^{1/n} I, \tag{7}$$

從 A_i 取出 eigenvalues λ_{ij} 和 eigenvectors ϕ_{ij}

對於 $\lambda_{i\max}/\lambda_{ij} > \beta$ 找出 $\lambda_{i\max} = \max_j \lambda_{ij}$ 和 設定: $\lambda_{ij} = \lambda_{i\max}/\beta \quad \forall j$

利用下列式子重新計算 A_i

$$A_i = [\phi_{i1}...\phi_{in}]diag(\lambda_{i1},...,\lambda_{in})[\phi_{i1}...\phi_{in}]^{-1}$$

3.2 階層聚合演算法獲得第二階段分群

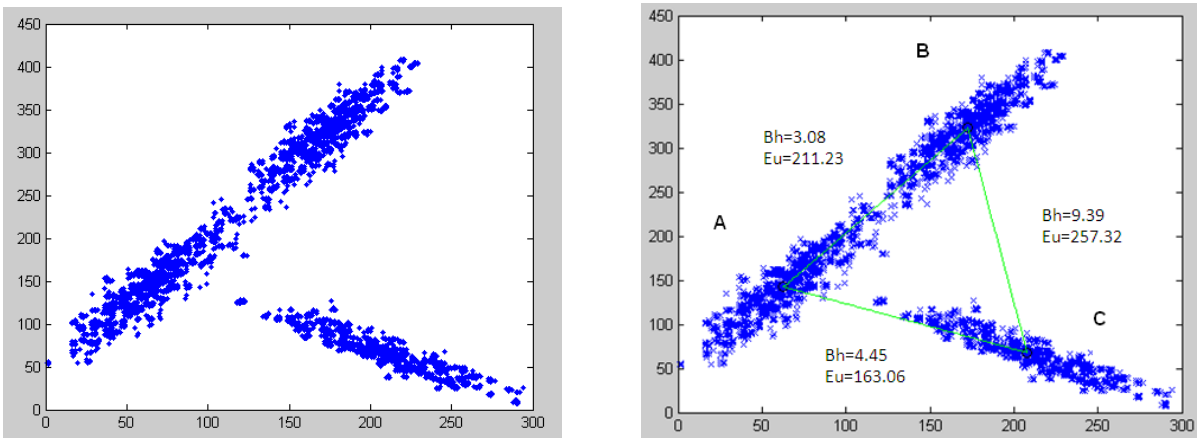
在這一小節中我們將去找出第一階段分群結果中雛型兩兩之間的距離係數，並由此距離係數來判別雛型之間相關性的多寡，再利用階層聚合演算法來獲得第二階段分群的結果。

在利用 GK 叢集化演算法後所得到的每一個雛型都具有重心點位置以及其相關性變異矩陣，所以勢必要去找到一個比歐基理得距離更適合於這樣的重心點之間的距離計算公式，在此採用的是 Bhattacharya distance [8]

$$B(i,j)=\frac{1}{8}(v_i-v_j)^T\left(\frac{A_i+A_j}{2}\right)^{-1}(v_i-v_j)+\frac{1}{2}\ln\left(\frac{\left|(A_i+A_j)/2\right|}{\sqrt{|A_i|}\sqrt{|A_j|}}\right) \tag{8}$$

Bhattacharya distance 可以被視爲兩個橢圓重心點之間的方向性的測量，當某兩橢圓重心點之間的方向性越一致，則此兩橢圓之間的由(8)所計算出的值越小，反之如果某兩橢圓重心點之間的方向性越不一致，則所計算出的值越大，因此由(8)所得到的 $B(i, j)$ 是一個非相似值矩陣。

下圖 7 是一個圖例，圖 7(a)中爲三個利用高斯分佈所產生的資料集合、而在(b)中爲三個高斯分佈資料兩兩之間的 Bhattacharya distance 與歐基里德距離的距離數值。首先我們以高斯分佈 A 爲基礎點來觀察，觀察圖(b)可清楚得知高斯分佈 B 應該比高斯分佈 C 更適合與高斯分佈 A 做連結。但如果我們是使用歐基里德距離的距離計算方式，則高斯分佈 C 比高斯分佈 B 更適合與高斯分佈 A 做連結，造成錯誤的判斷。而利用 Bhattacharya distance 來當成距離的計算方式，由圖(b)中的距離數值來看，Bhattacharya distance 可正確的判斷出高斯分佈 A 與高斯分佈 B 做連結。



(a) (b)

圖 7: Bhattacharya distance 與歐基里德距離的比較。(a)高斯分佈資料集合(b) Bhattacharya distance 與 歐基里德距離。

有了 GK 叢集化演算法所得到的代表資料集合的重心點以及上述的(8)所得到非相似值矩陣，則我們想利用階層聚合叢集化演算法對這些重心點做最佳化的分群，而階層聚合最常用的有單一連結 (single-link)、平均連結 (average-link)、或是完全連結 (complete-link) 。在此將採用單一連結的最大生命週期條件 (maximum-lifetime criterion) 來決定最穩定的分群，其階層聚合叢集化演算法參數與演算法定義如下:

若 P 爲一相似值矩陣，其 P_{ij} 爲雛型 i 與雛型 j 之間的相似值，則每一個步驟都需取出最大相似值的兩個雛型 C_i 與 C_j ，將其合併成一個雛型 C_q ，而合併完所產生的新雛型對於其它的舊雛型 C_s 其相似值的更新公式如下:

單一連結:

$$d(C_q, C_s) = \max\{d(C_i, C_s), d(C_j, C_s)\} \tag{9}$$

完全連結:

$$d(C_q, C_s) = \min\{d(C_i, C_s), d(C_j, C_s)\} \tag{10}$$

平均連結:

$$d(C_q, C_s) = \frac{1}{2}(d(C_i, C_s) + d(C_j, C_s)) \tag{11}$$

若 P 爲一非相似值矩陣，其 P_{ij} 爲雛型 i 與雛型 j 之間的相似值，則每一個步驟都需取出最小相似值的兩個雛型 C_i 與 C_j，將其合併成一個雛型 C_q，而合併完所產生的新雛型對於其它的舊雛型 C_s 其非相似值的更新公式如下:

單一連結:

$$d(C_q, C_s) = \min\{d(C_i, C_s), d(C_j, C_s)\} \tag{12}$$

完全連結:

$$d(C_q, C_s) = \max\{d(C_i, C_s), d(C_j, C_s)\} \tag{13}$$

平均連結:

$$d(C_q, C_s) = \frac{1}{2}(d(C_i, C_s) + d(C_j, C_s)) \tag{11}$$

演算法如下:

輸入 ： P 是一個 $n \times n$ 的相似／非相似矩陣 ，

輸出 ： n 個資料點被分配到 k 個叢集聚合分割中.

程序 ： 階層聚合演算法

$l = n$

For i = 1 to n

Let $C_i = \{x_i\}$ for $i = 1,...,n$

Repeat

從 P 中找出最相似／不相似的兩個叢集， C_i 和 C_j

合併 C_i 和 C_j 並且將 l 減一

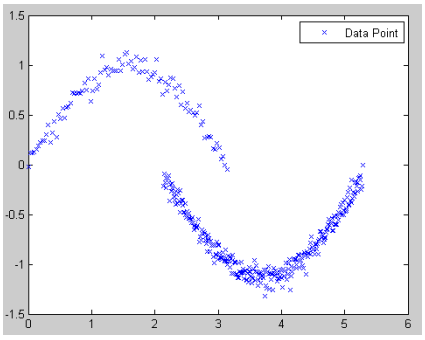
Until $l \le k$

回傳所有非空的叢集聚合分割

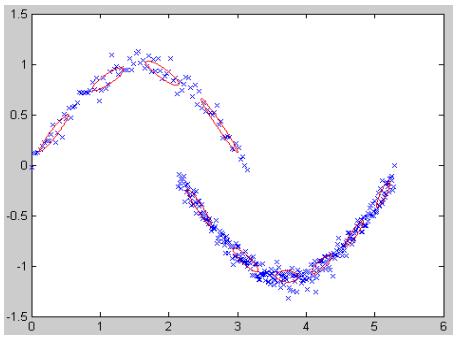
上述演算法的 k 值是最後所形成的個別分群的個數，在此我們是採用最大生命週期限制來決定最後個別分群的個數，而利用最大生命週期限制所得到的這些雛型，可以被視爲所輸入的資料集合其最穩定的分群。這個所求得的最穩定分群爲我們第二階段的分群結果，當然這裡所說的輸入資料集合是指由 GK 叢集化演算法所得到的雛型，當成爲階層聚合叢集化演算法的輸入資料。

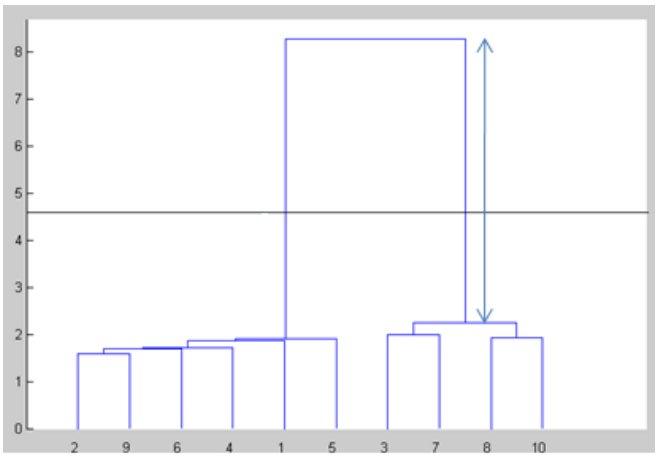
所謂的最大生命週期限制爲在階層聚合叢集化演算法的聚合過程中取出最穩定的分割，例如: 在階層聚合的過程中某一個時間點因爲聚合的動作存在著大大小小的分群，假設目前的相似值爲 v_1 ，而到下一次聚合動作發生的相似值爲 v_2 ，當 $v_2 - v_1 = v$ 中的 v 爲最大時，則此時便是一個最穩定的分割狀態。

下圖 8 中我們針對圖(a)的資料集合做 GK 叢集化演算法得到的第一階段的分群，即圖(b)。再將第一階段分群的這些雛型利用階層聚合叢集化演算法使用最大生命週期限制得到的最穩定的分割。其中圖(c)爲階層聚合過程中的樹狀圖，而圖(c)中的黑線爲標示出在此時候的分割爲聚合過程中最穩定的狀態。



(a)

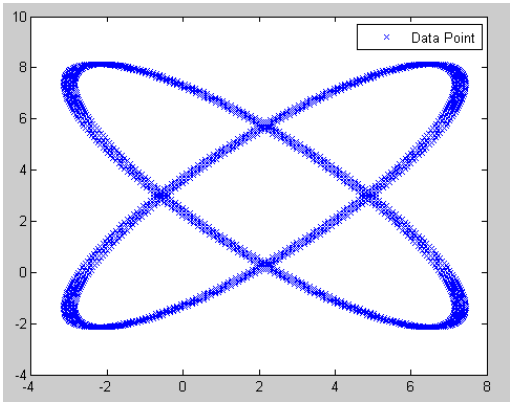

(b)



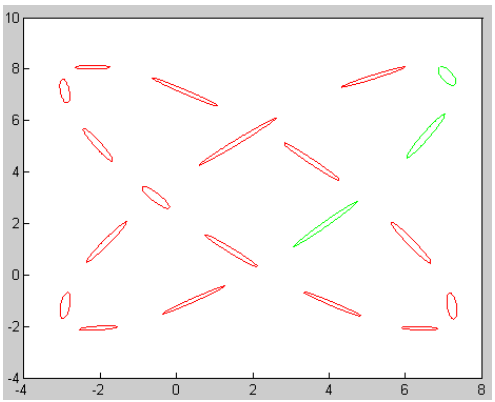
(c)

圖 8: 階層聚合演算法單一連結使用最大生命週期限制。(a)兩個半圓形的資料集合、(b)第一階段分群結果、(c)層階聚合樹狀圖。

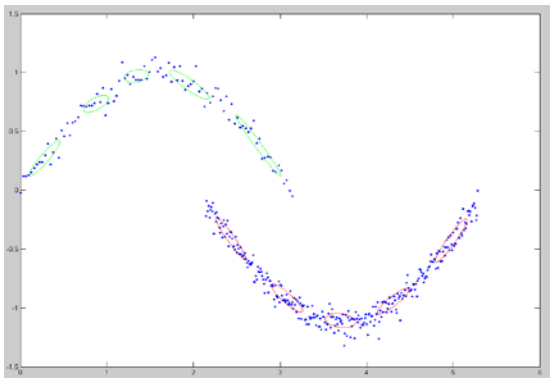
下圖 9 的(a)是(b)的輸入資料集合、(b)跟(c)是我們使用階層聚合叢集演算法中的單一連結並使用最大生命週期限制所得到的分群結果，而我們所產生的是一個非相似值矩陣，其圖中相同顏色的雛型是代表在最大生命週期的限制下所得到的同一群的聚合雛型。由單一連結的聚合更新公式可以得知每次聚合的時候都去找到最小的非相似值的兩個雛型來做聚合的動作，且更新的時候是以最小非相似值來當成對其它非同一群的雛型的非相似值。所以單一連結非常適合用來偵測軸線形的資料集合。下圖 9(b)、9(c)中同一顏色的雛型代表聚合後屬於同一群，所以(b)跟(c)在最大生命週期限制下都被分爲兩群。



(a)



(b)



(c)

圖 9: 階層聚合叢集演算法使用最大生命週期限制。(a)兩個相交的橢圓、(b)爲(a)的第二階段分群結果、(c)兩個半圓形的第二階段分群結果。

3.3 使用叢集整合技術得到第三階段的分群

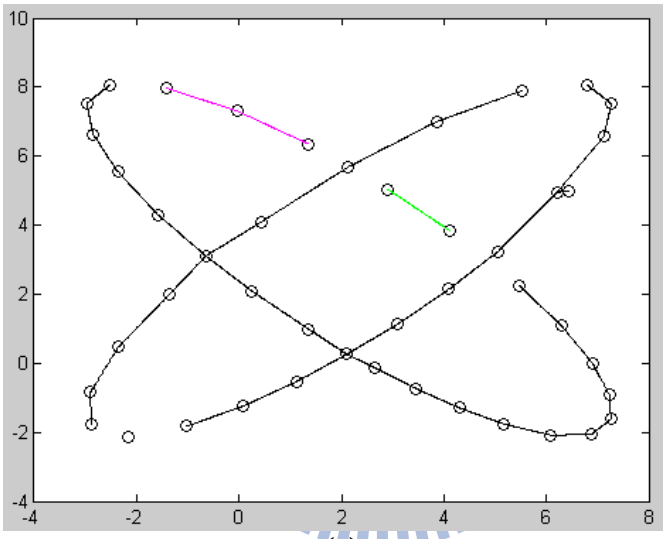
在 3.2 節中所得到的是一個第二階段的分群，並由 3.2 節可以得知對於含有交叉性質物件的資料集合，我們並無法從第二階段的分群得到正確的分群結果。所以在此章節中我們將利用叢集整合技術來獲得正確的分群結果，也就是我們所提出的第三階段的分群。

叢集整合演算法包含了三個主要的部份: 產生分群的方法、將數個個別的分群結果使用一個資料結構來表示、利用此一整合的資料結構來得到最終的分群。在此我們產生分群的方法爲先由上一小節所得到輸入資料集合的第二階段分群，但爲了確保在叢集整合的過程中能夠有最佳的整合效果，所以在使用第二階段分群來當叢集整合中產生分群的方法之前，我們將對第二階段分群做連結最佳化的動作。

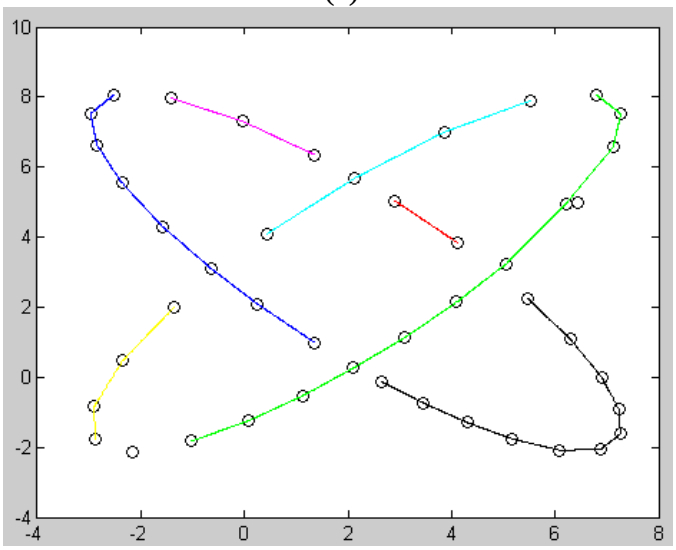
這些第二階段分群中的每一個單一分群可能是數個從 GK 叢集化演算法所得到的代表資料集合的雛型所組成。我們將針對每一個單一分群做連結最佳化的動作，主要有兩個步驟: 一、去除掉大於兩個連結的雛型其多餘的連結，直到每一個雛型都只保有最多兩個連結。二、對於由步驟一所得只保有兩個連結的雛型的此兩連結作判斷，當如果此兩連結其夾角小於所設定的門檻值的話，則去除掉這兩連結讓此一雛型保持沒有連結

的狀態。當去除掉某一個連結時則這個單一分群將被分裂爲二。

圖 10 是一個去除過程的例子，首先從圖 10(a)中每一個同一顏色的連線包含了數個個數不一的雛型。一個同一顏色的連結中所包含的所有雛型，我們將它視爲一個分群，即是我們所稱的單一分群。而在(a)中所包含的雛型最多的黑色連結線段的單一分群，經過連結最佳化的動作在(b)中已經被分爲數個單一分群。而圖 10(a)是第二階段分群的結果以連結的方式來表示。



(a)



(b)

圖 10: 第二階段分群其連結最佳化。(a)第二階段分群結果、(b)第二階段分群其連結最佳化。

由上述的兩個動作後所得到的連結最佳化第二階段分群，被當作爲叢集整合演算法的產生分群的方法，並且在這裡我們所說的一個分群指的是連結最佳化第二階段分群中的一個單一分群。

下圖 11 爲數次連結最佳化第二階段分群的結果，我們將以單一分群爲單位來整合數次分群的結果。

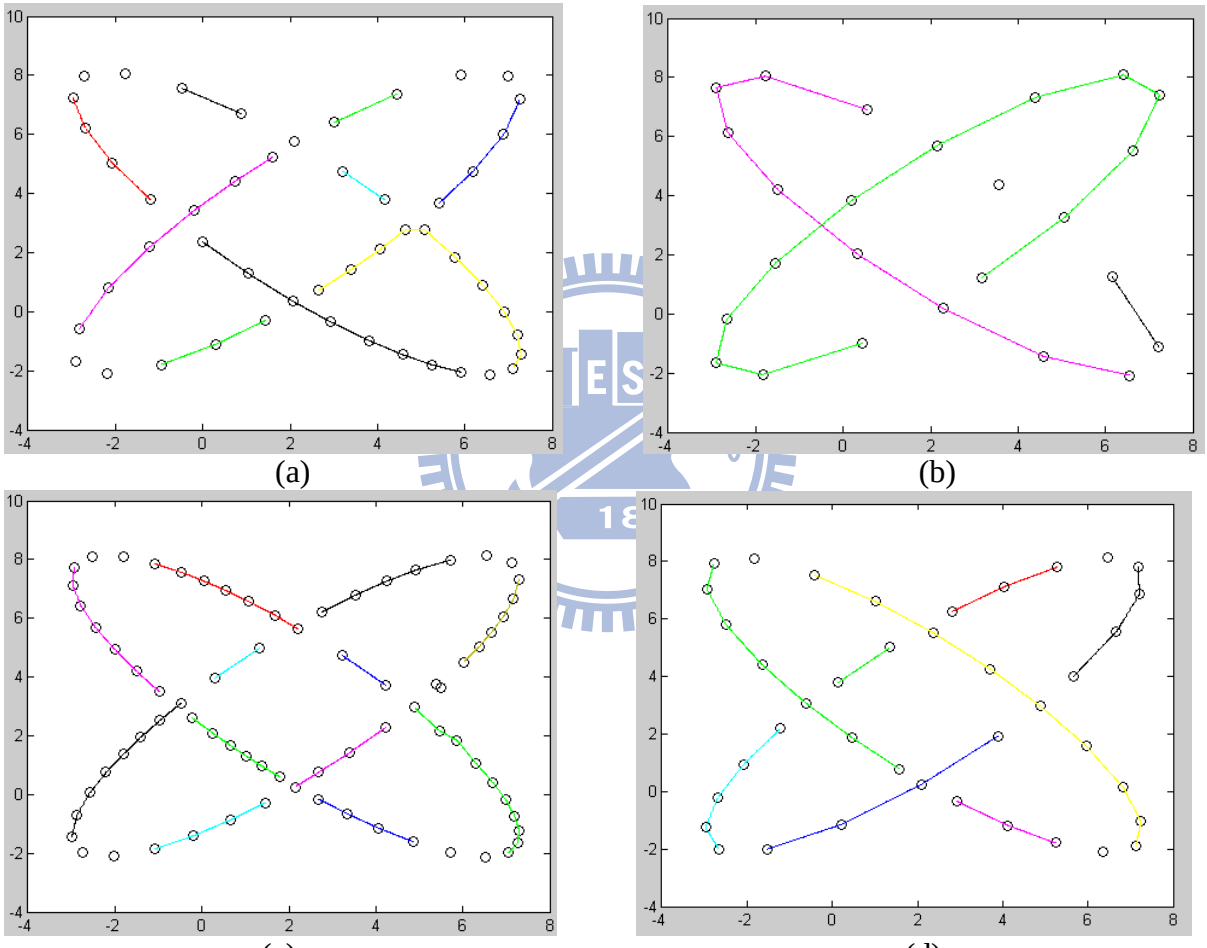


圖 11: 數次連結最佳化第二階段分群的結果。(a)雛型個數爲 50 個的分群狀態、(b) 雛型個數爲 25 個的分群狀態、(c) 雛型個數爲 75 個的分群狀態、(d) 雛型個數爲 40 個的分群狀態。

而在叢集整合演算法的步驟二: 將數個個別的分群結果使用一個資料結構來表示，所提出的方法如下: 由於每一個單一分群是由數個從 GK 叢集化演算法所得到的代

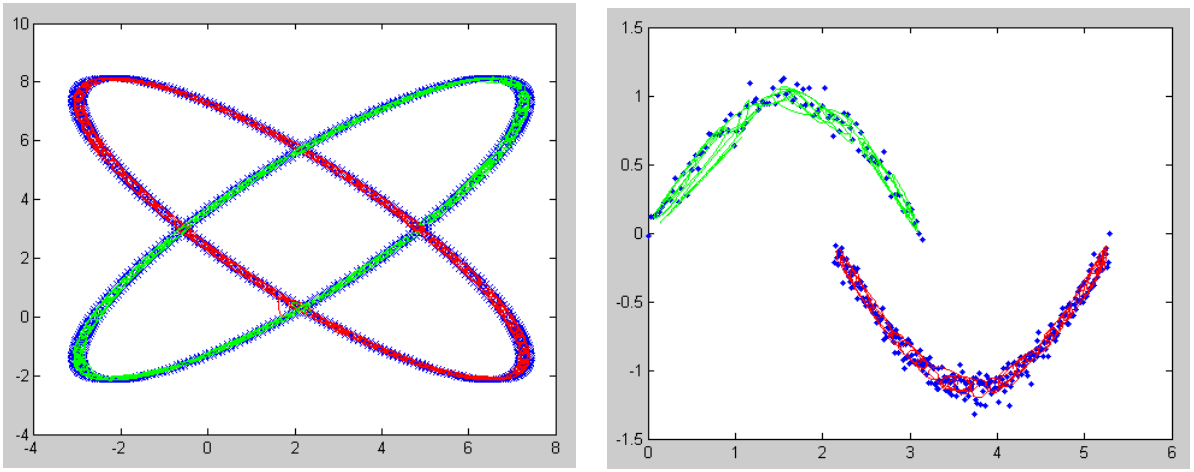
表資料集合的雛型所組成，我們將某一個資料點與單一分群之間的模糊相關係數定義爲資料點對於這一個單一分群中所有 GK 雛型之間模糊相關係數的總合。

有了單一分群的模糊相關係數後，我們將去整合數次連結最佳化第二階段分群的結果，並且計算每一個單一分群其兩兩之間的相似值，所使用的是 [16] 中的:

$$P_{ij} = \frac{\sum_{k=1}^{N} \min(u_{ik}, u_{jk})}{\min(\sum_{k=1}^{N} u_{ik}, \sum_{k=1}^{N} u_{jk})} \tag{14}$$

這裡的 u_{ik} 指的是單一分群 i 與所有資料點之間的模糊相關係數所構成的矩陣，而 P_{ij} 即爲單一分群 i 與單一分群 j 之間的相似值。當兩個單一分群內的雛型重疊度增加時，則 P_{ij} 的值會隨著增加。反之，則 P_{ij} 的值會隨著減少。經由(14)來代表整合數次的個別連結最佳化第二階段分群結果，其所求得的整合矩陣爲一個整合數次個別連結最佳化第二階段分群後，所有單一分群的相似矩陣。在此將使用階層聚合演算法中固定分群個數的平均連結來從這一個整合性的相似矩陣中，獲得第三階段的分群結果。

下圖 12 的(a)、(b)分別爲相交雙橢圓資料集合整合十次以及兩個半圓形資料集合整合五次個別分群後，利用叢集整合技術所得到的最終分群的結果，圖中所展現出來的是包含每次個別分群的所有 GK 雛型。經由所提出的叢集整合技術我們可以正確的決定出資料集合所包含的物件，例如: (a)中所有紅色的雛型代表同一群，它可以正確的決定出一個橢圓資料集合、(b)中所有紅色的雛型代表同一群，它決定出(b)資料集合的下面半圓形資料集合。



(a) (b)

圖 12: 叢集整合演算法所得到的最終分群。(a)整合 10 次個別分群後所得到的第三階段分群結果，(b)整合 5 次個別分群後所得到的第三階段分群結果。(a)、(b)中相同顏色表示爲同一分群。

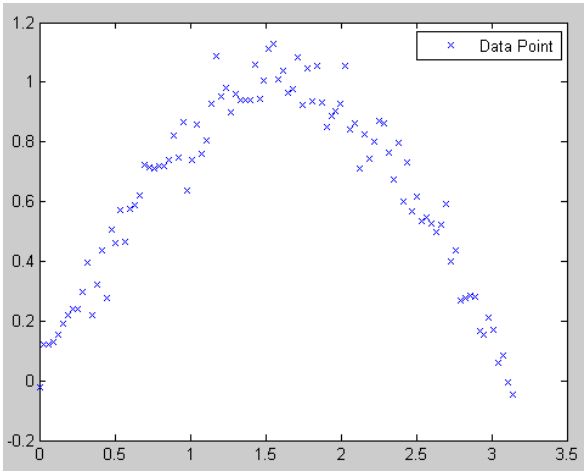
3.4 獲得單一分群的資料集合

輸入的資料集合可能是由數個多重的分離圖形物件、或是交叉的圖形物件所組成，我們想從這樣的資料集合中自動的去找到正確的分群，進而能夠自動化的找到資料集合中所包含的多重圖形物件、或是交叉的圖形物件。

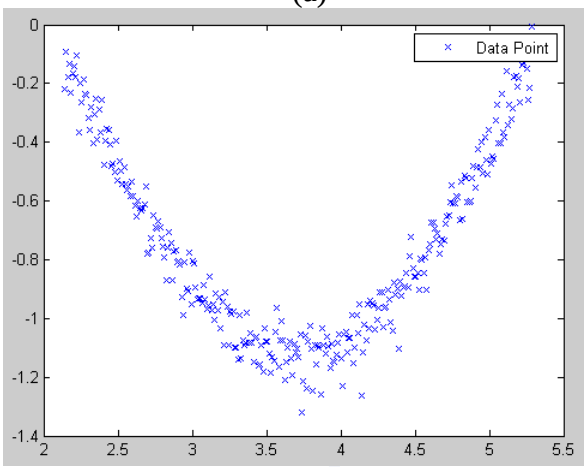
我們將個別去取出屬於一個第三階段分群的資料集合，再以這資料集合去找到它的主曲線。對於由 3.3 所得到的第三階段分群結果的一個分群而言，它是由數次的個別第二階段分群所整合的結果。而每一次的個別分群是由數個從模糊 GK 叢集所組成，即爲第一階段分群所得到的雛型。因此我們對一個第三階段分群它所包含的所有由模糊 GK 叢集做去模糊化(defuzzification) ，將每一個輸入的資料集合中的資料點歸屬於最大相關性值的 GK 叢集所屬的第三階段分群，由此來得到屬於每一個第三階段分群的資料集合。

下面圖 13 的(a)跟(b)是利用圖 12(b)的第三階段分群所得到的兩個個別分群的資料集合。(a)的部份爲圖 12 的(b)中綠色的個別分群所收集到的資料點集合，形成上半部的半圓形，(b)的部份爲圖 12 的(b)中紅色的個別分群所收集的到的資料集合，形成下半部

的半圓形。



(a)



(b)

圖 13: 單一個別分群的資料集合。(a)兩個半圓形的上半部、(b)兩個半圓形的下半部。

3.5 使用修改的 Fuzzy Curve-Tracing 演算法獲得主曲線

文獻 [6] 中，有討論關於模糊主曲線追蹤演算法的優點與其最主要的限制，它並無法找到含有多重物件的資料集合。但由前一小節所得到的資料集合是屬於一個單一物件的資料集合，因此我們可以使用文獻 [6] 所提出的模糊主曲線追蹤演算法，來獲得此一資料集合的主曲線。其演算法介紹如下:

首先使用模糊 C 均值叢集化演算法來得到輸入資料集合的代表點，因爲模糊 C 均值叢集化演算法只是一個 GK 叢集化演算法的特例，其主要的差異是距離計算方式的不

同。距離函數定義如下:

$$D_{ik} = (x_k - v_i)^T (x_k - v_i) \tag{15}$$

即爲將相關性矩陣取成爲單位矩陣，而各項參數的定義與其演算法，如同 3.1 的模糊 GK 叢集化演算一樣。

在 [6] 中作者提出了一個影響係數的計算方式，對於每一個叢集重心點都與其最大影響係數跟第二大影響係數的重心點相連，形成一個重心點的相關性圖形。而這一個所形成的相關性圖形可能會包含迴圈，所以必須去除掉連結時所產生的迴圈，才能得到初始的主曲線，在此作者提出了兩個方法去決定出所產生的迴圈是一個終點的迴圈，還是一個位於相關形圖形中間的迴圈。但如果在一個複雜的情況下，這個相關性圖形有可能會形成較複雜的迴圈，例如一個邊數大於三個邊的多邊形迴圈，可是作者所提出的去除迴圈方法只假定形成的迴圈是一個三角形的迴圈，這樣的方法並無法去除掉多邊形迴圈，所以我們在這提出一個更適合於廣泛情況的去除迴圈的方法。而這裡所指的迴圈是由連結大於兩個的重心點所形成的，我們想要讓每一個重心點的連結個數最多不超過兩個。

首先我們利用(14)來計算兩兩叢集重心點之間的相似值，並且設定一個門檻值，當兩個重心點的相似值大於這一個所設定的門檻值時，則它們就應該被連結。我們的方法與 [6] 有所不同的是對於一個叢集重心點而言，我們不僅僅是與其相似值最大與第二大的叢集重心點相連結，而是一個叢集重心點可能跟所有其它的叢集重心點相連，也可能與其它的都不相連，由這樣來產生一個相關性圖形。而這個相關性圖形並不一定形成一個單一的連通圖，所以我們從這一個相關性圖形中找到最大的連通子圖，並且去除掉不屬於這一個連通子圖的所有叢集重心點以及它的相關連結，由這一個最大的連通子圖來代表新的相關性圖形。

再來針對這一個所取出來的連通子圖中，其連結大於兩個邊的叢集重心點做去除多餘邊的動作。去除的方法爲找到大於兩個邊的叢集重心點，對於它所擁有的所有連結中取出相似值最小的那一個連結，而再要去除掉這個相似值最小的那個邊時必須去檢查如果去掉這一個邊是否會形成兩個連通子圖，若是，則放棄去除此邊，改取出相似值第二小的邊重覆一樣的檢查是否會形成兩個分離的連通子圖，若不是則去除掉此一連結邊。重覆這樣的動作直到這一個連通子圖沒有大於兩個邊的叢集重心點並且仍然是維持一個連通圖形，由此我們便可得到初始主曲線。下圖 14 爲所提出方法的流程圖。

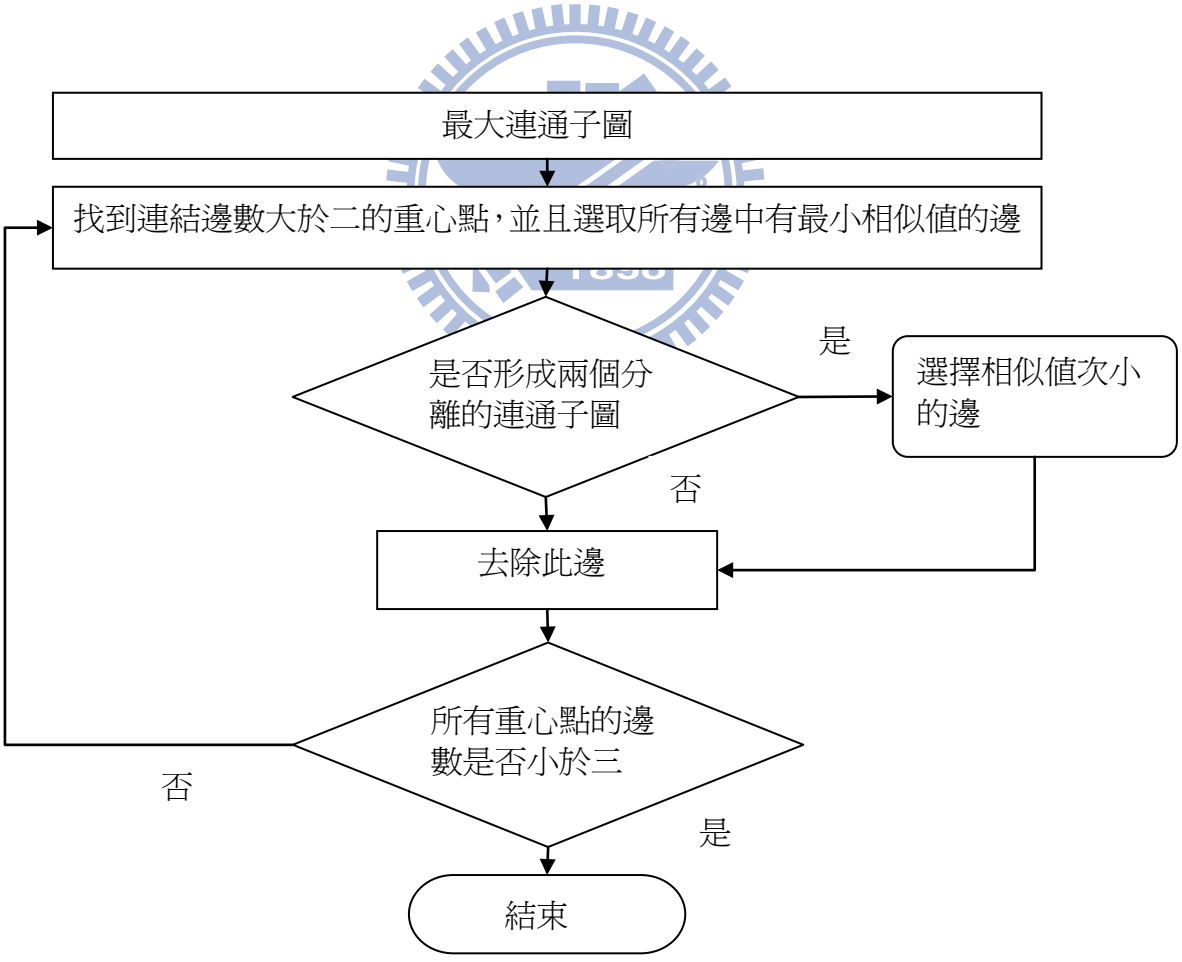
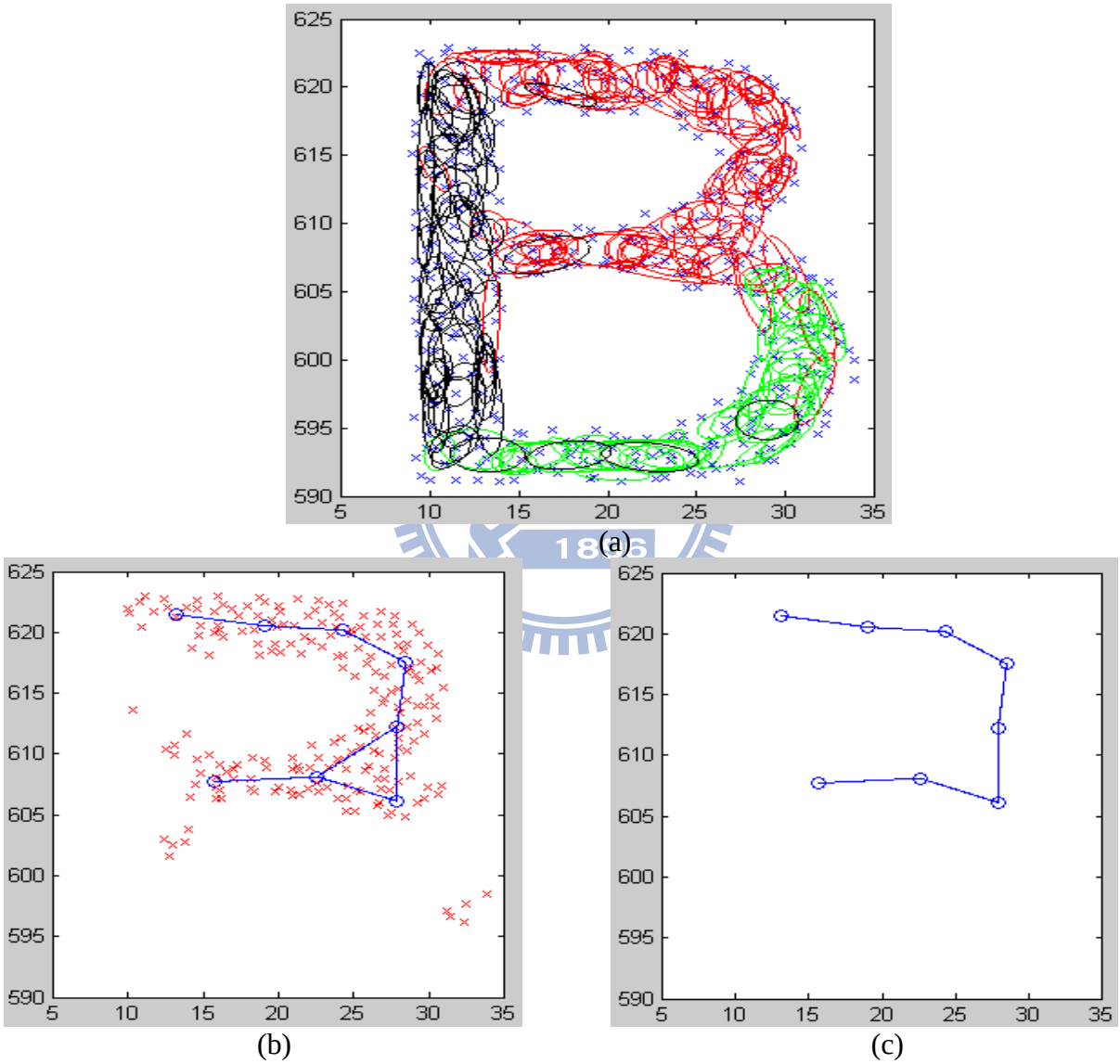


圖 14: 去除多餘邊流程圖

下圖 15 爲利用我們所提出去除多餘邊的方法所得到的初始主曲線的結果，其中(a)爲由 3.3 所得到的第三階段分群的結果。而(b)爲對於(a)中紅色雛型所包含的資料點所形成的資料集合做找初始主曲線的動作，圖中有由模糊 C 均值演算法得到的重心點所形成的相關性圖形，且它也是最大的連通子圖。(c)爲去除掉多餘邊所得到的初始主曲線。



(a)　(b)　(c)

圖 15: 去除多餘邊獲得初始主曲線。(a)第三階段分群結果、(b)紅色個別分群資料集合的最大連通子圖、(c)去除多餘邊後所得到的結果。

在上述的方法後，我們可以得到一個初始的主曲線，由這一個初始的主曲線可以去找出叢集重心點的排列順序。我們下一個步驟就是平滑化初始的主曲線。由 [6] 中作

者提出的方法是去修改模糊 C 均值叢集演算法，在它的成本函數中多加了一條限制條件，當某一個叢集重心點要趨於收斂的時候，它必須去考慮到與其相連的叢集重心的位置。所以利用叢集重心點的排列順序慢慢的去調整叢集重心點的之間的相對位置，使得初始主曲線凹凸不平的地方因爲叢集重心點拉近的關係變的較平滑，而且這一個平滑化的步驟也是讓所找到的主曲線能在雜訊的環境中，得到良好的結果最主要的原因。

[6] 所提出的平滑化演算法的成本函數定義如下:

$$J(Z;U,V)=\sum_{i=1}^{K}\sum_{k=1}^{N}(\mu_{ik})^{m}\left|x_k-v_i\right|^2+\alpha\sum_{i=2}^{K-1}\left|v_{i+1}-2v_i+v_{k-1}\right|^2 \tag{16}$$

其演算法以及各項參數定義如同模糊 C 均值叢集演算法，而有所不同的只有其步驟一重心點的更新與α值，此一α值爲平滑化程度值，當所得到的初始主曲線其所有的雛型的連結都爲二時，則所求的是封閉的主曲線，則其成本函數定義應更改爲:

$$J(Z;U,V)=\sum_{i=1}^{K}\sum_{k=1}^{N}(\mu_{ik})^{m}\left|x_k-v_i\right|^2+\alpha\sum_{i=1}^{K}\left|v_{MOD(i+1,K)}-2v_i+v_{MOD(i-1,K)}\right|^2 \tag{17}$$

3.6 整合數段主曲線獲得最終結果

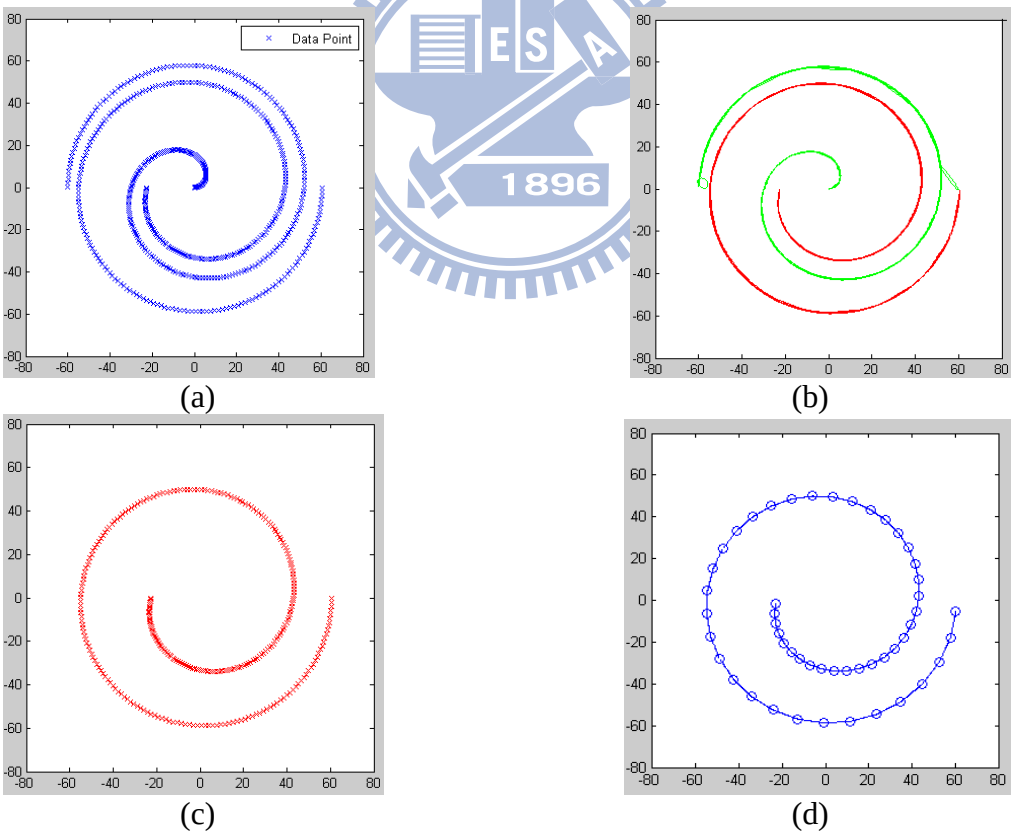
我們由 3.4 所得到的單一分群的資料集合，可能是輸入資料集合中的某一個分段的資料，例如: 假若輸入的資料集合可能含有多重的圖形物件，而我們所取出的可能是一個圖形物件的資料、若輸入的資料集合是含有交叉的圖形物件，而我們所取出的可能就是這個圖形物件的一段不含交叉的資料段。

有了這些原本輸入資料集合的資料分段集合，我們再把這些資料分段集合經由 3.5 的模糊主曲線追蹤演算法來求得各自資料分段集合的主曲線。因此在我們找出輸入資料集合的主曲線最後的一個步驟，便是把這些各自從資料分段集合所求得的各自的主曲線整合起來並由這一個整合的主曲線來代表輸入資料集合所求得的主曲線。

第四章 實驗結果

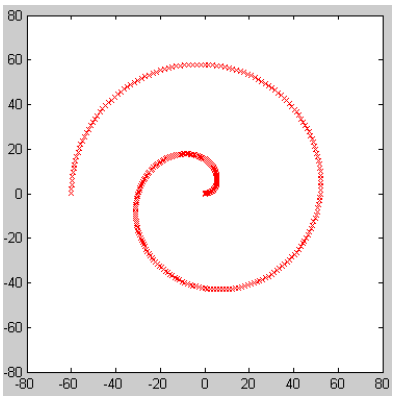
4.1 具有多重物件的資料集合偵測

首先我們將對於含有多重物件的資料集合做主曲線的偵測，下圖 16(a)爲一個包含有兩個物件的資料集合，(b)爲所得到的第三階段的分群結果，可看出我們所提出的方法能正確的分辨出兩個物件。而(c)與(e)分別爲取出綠色個別分群與紅色個別分群的資料集合，其(d)與(f)爲對應的主曲線。

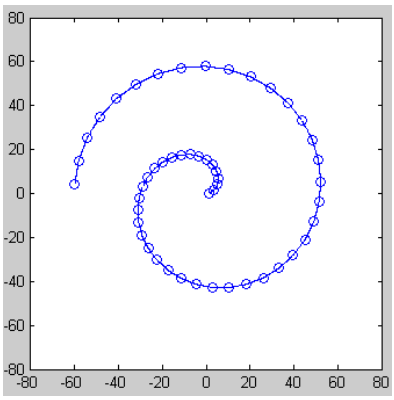


(a) (b)

(c) (d)



(e)

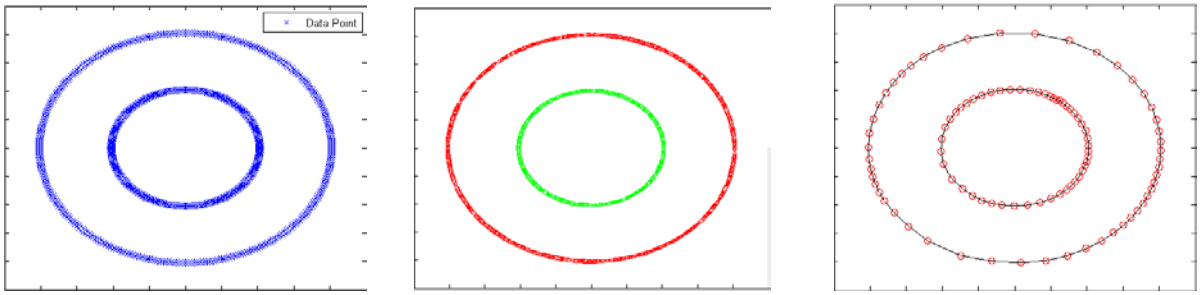


(f)

圖 16: 雙螺旋主曲線的偵測。(a)資料集合、(b)第三階段分群結果、(c)與(e)綠色個別分群與紅色個別分群的資料集合、(d)與(f)爲對應的主曲線。

在上圖16的實驗中我們所使用到的參數值設定如下: 其修改的模糊GK 叢集演算法的β爲10的15次方、γ爲0.0002、收斂門檻值ε爲1、模糊運算子m爲2、其整合個別分群的次數爲10次。在整合個別分群中，每一次個別分群所選取的雛型個數我們是以在一個雛型個數範圍內，利用隨機的方式去挑選目前這一次所需要的雛型個數。所使用的雛型個數範圍爲: 50、55、60、65、70、75、80。而模糊主曲線追蹤演算法的叢集重心點個數爲70個、連結門檻值爲0.01、平滑化程度α值爲2。

再來我們去測試另一個含有多重物件的資料集合，其所用的參數值與上述有所不同的是模糊主曲線追蹤演算法的叢集重心點個數爲50個。下圖17中(a)是兩個不同半徑的同心圓所形成的資料集合，(b)爲第三階段的分群結果，可看出仍然可以正確的將這兩不同半徑的同心圓分成兩個分群。因此我們便可從第三階段的分群結果取出正確的個別分群的資料集合，並且針對個別的資料集合去找出它的主曲線。(c)爲最後所得到的主曲線結果。

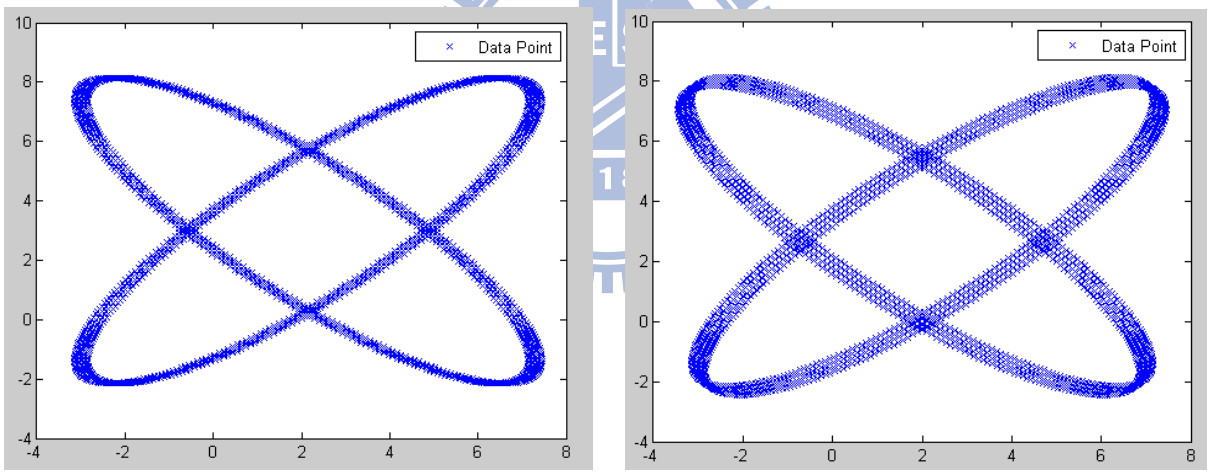


(a) (b) (c)

圖17: 不同半徑的同心圓其主曲線的偵測。(a)同心圓資料集合、(b)第三階段分群結果、(c)整合後個別分群主曲線所得到的結果。

4.2 測試具有交叉的資料集合

這裡我們所使用資料集合含有兩個橢圓，並且此兩橢圓互相相交產生四個交點。在這裡我們將產生兩組這樣的資料集合，如下圖 18 的(a)跟(b)。



(a) (b)

圖 18: 相交橢圓的資料集合。(a)較窄的相交橢圓、(b)較寬的相交橢圓。

這兩組資料集合主要的不同點在於橢圓圓周的寬度，而橢圓圓周的寬度會決定兩個橢圓其相交部份對於找到其主曲線的複雜程度。這是因爲如果圓周寬度較寬的橢圓相對於圓周寬度較窄的橢圓其在交叉的部份所包含不同橢圓的資料點較多，而且交叉的部份較大，所以造成由 GK 演算法所得到的雛型會有較大的機會落在交叉的部份上。因此這些落在交叉部份上的雛型會有很大的機會包含到不同橢圓的資料點，造成在階層聚合的

過程中有較多的情況造成錯誤的連結。

由下圖 19 可以看出圓周寬度較寬有較差的連結情況，其中(a)與(b)爲輸入的橢圓圓周較寬在不同的雛型個數下所得到的第二階段分群結果。而(c)與(d)爲輸入的橢圓圓周較窄在不同的雛型個數下所得到的第二階段分群結果。

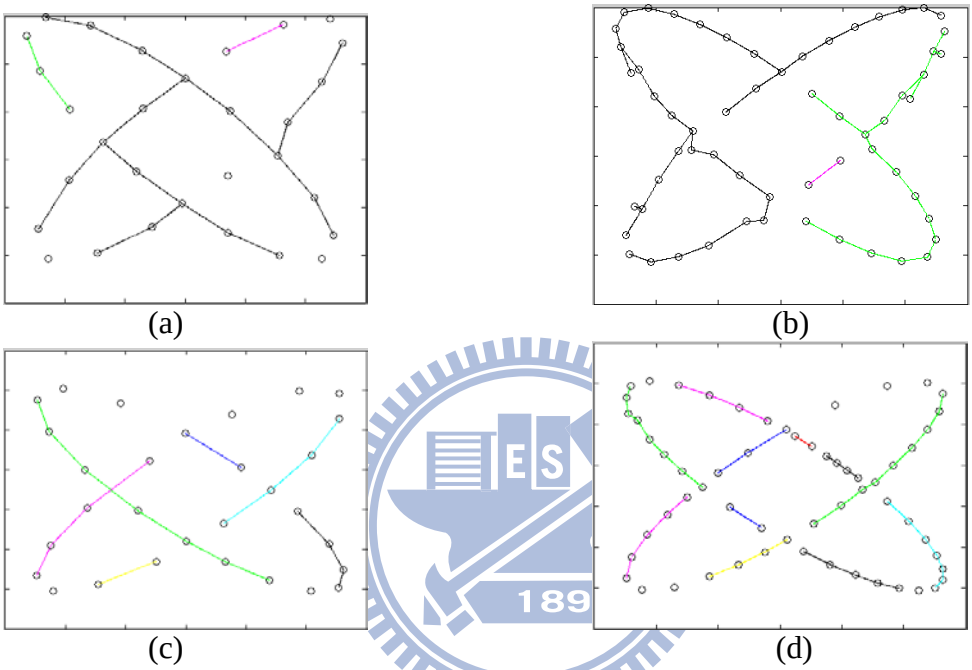


圖 19: 不同資料集合的第二階段分群結果。(a)與(b)分別爲雛型個數爲 30 與 60，由圖 18(b)所得到的第二階段分群的結果。(c)與(d)分別爲雛型個數爲 30 與 60，由圖 18(a)所得到的第二階段分群的結果。(a)跟(b)是在最佳化之前。

我們使用這兩組資料集合來測試是否能決定出兩個橢圓的主曲線。首先我們將先對橢圓圓周寬度較小的圖 18(a)做實驗。實驗中各項的參數數值設定如下：修改的模糊 GK 叢集演算法的 β 爲 10 的 15 次方、γ 爲 0.0002、收斂門檻值 ε 爲 0.5、模糊運算子 m 爲 2、其整合個別分群的次數爲 10 次，所使用的雛型個數範圍爲: 20、25、30、35、40、45、50、55、60、65、70、75、80。而模糊主曲線追蹤演算法的叢集重心點個數爲 20 個、連結門檻值爲 0.02、平滑化程度 α 值爲 2。圖 20 是圓周寬度窄的橢圓。

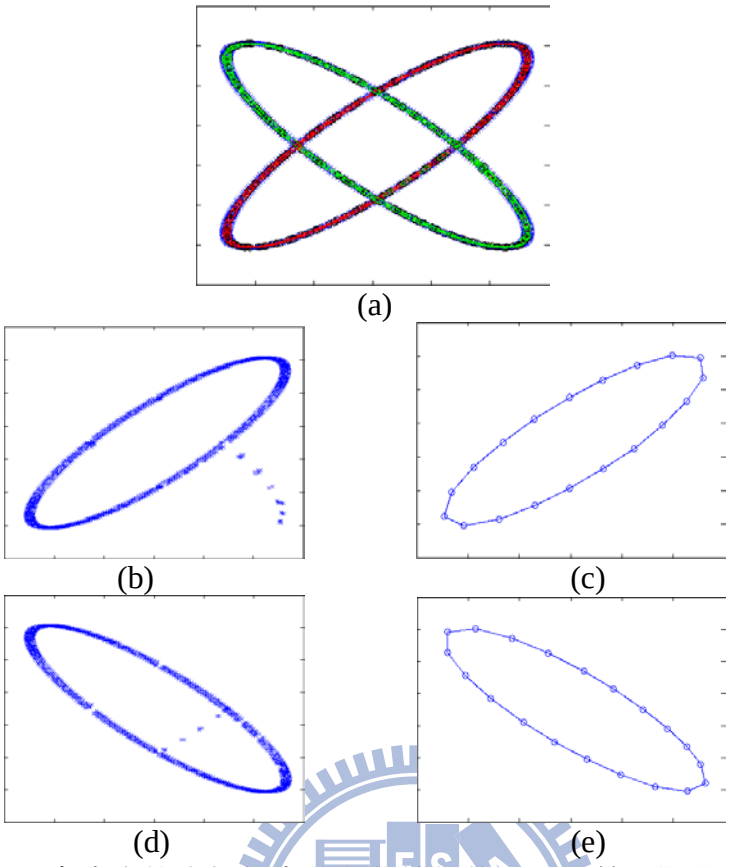
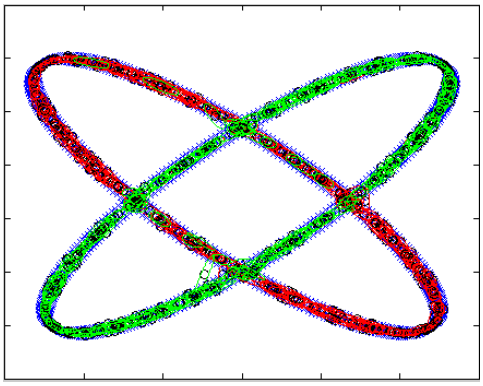


圖 20: 橢圓圓周寬度窄的資料集合其主曲線的偵測。(a)第三階段分群結果、(b)紅色個別分群資料集合、(c)紅色個別分群的主曲線、(d)綠色個別分群資料集合、(e)綠色個別分群的主曲線。

而橢圓圓周較寬的各項參數數值設定與上例有所不同的爲: 其整合個別分群的次數爲 15 次。圖 21 是圓周寬度寬的橢圓。



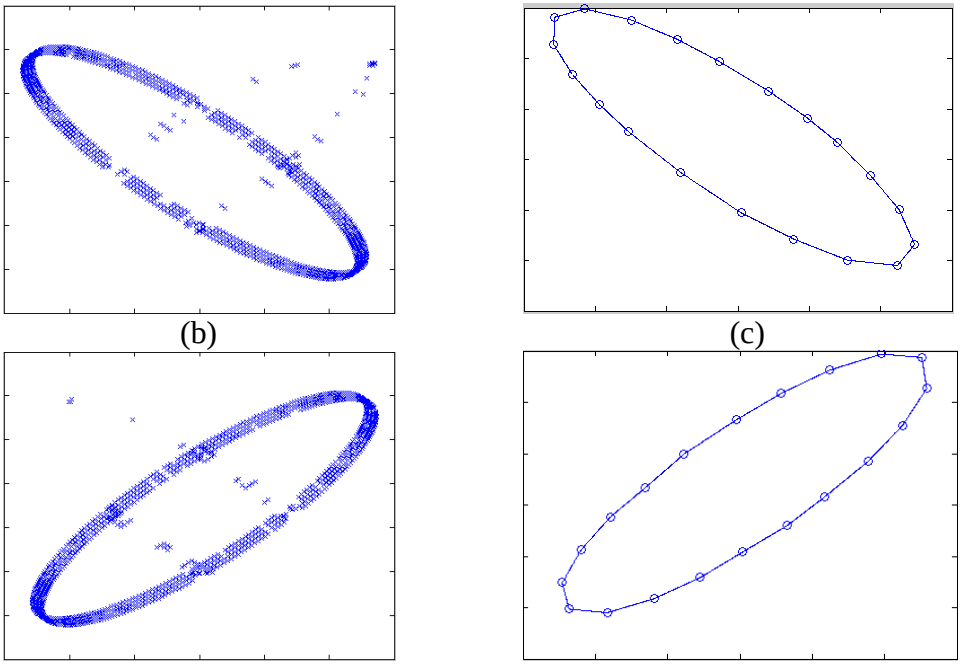


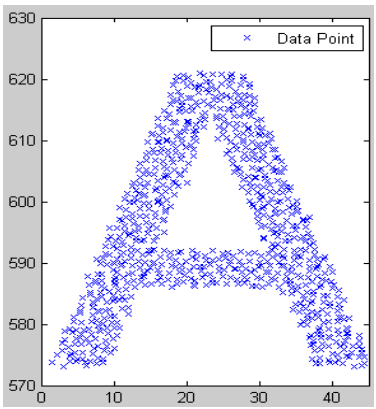
圖 21: 橢圓圓周寬度寬的資料集合其主曲線的偵測。(a)第三階段分群結果、(b)紅色個別分群資料集合、(c)紅色個別分群的主曲線、(d)綠色個別分群資料集合、(e)綠色個別分群的主曲線。

由圖 20 與圖 21 的(a)可以知道因爲橢圓圓周寬度寬的資料集合有較高的困難度，所以有較差的第三階段的分群結果。因此在取出個別分群的資料集合時會造成圖 21(b)與(d)有少數的資料點是取到不正確的橢圓資料點，形成類似有雜訊的情況。不過我們所利用修改的模糊主曲線追蹤演算法可以解決這種具有輕微雜訊的情況。

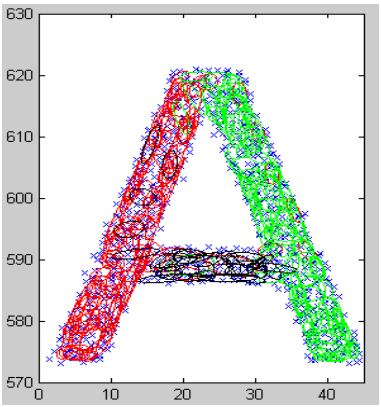
4.3 英文字母主曲線的偵測

在 4.1 與 4.2 的實驗中，可以知道我們所提出的方法不僅可以在一個含有多重物件的資料集合裡找到多重的物件，若這些多重的物件是交叉的也可以正確的被找到，所以在此小節中我們將對英文字母做主曲線偵測的動作，結果爲下面的圖 22。下圖 22 是我們所提出的演算法每步驟的結果，其中(b)爲叢集整合技術所得到的第三階段分群的結果，而(c)、(e)、(g)爲屬於單一個別分群的資料集合，與其相對應的(d)、(f)、(h)爲利用

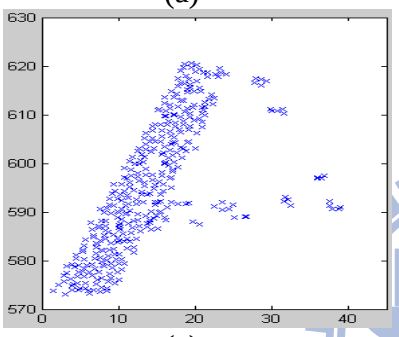
修改的 FCT 所得到的個別分群的主曲線，而(i)爲最後整合各段主曲線所得到的結果。



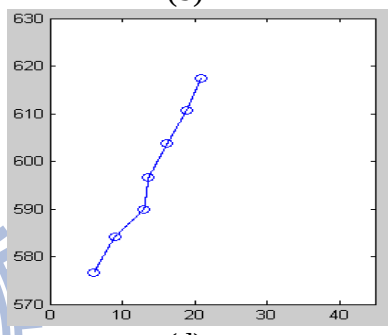
(a)



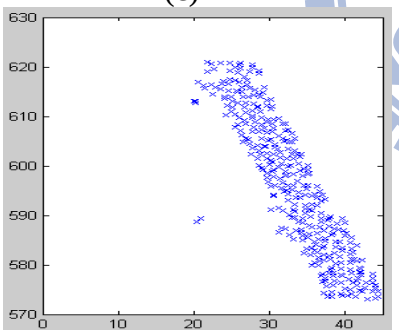
(b)



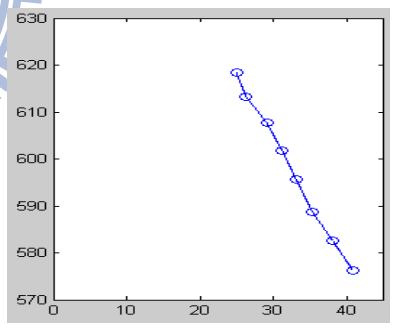
(c)



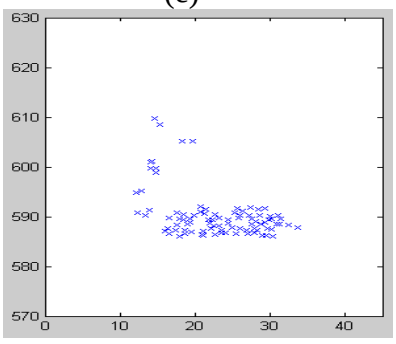
(d)



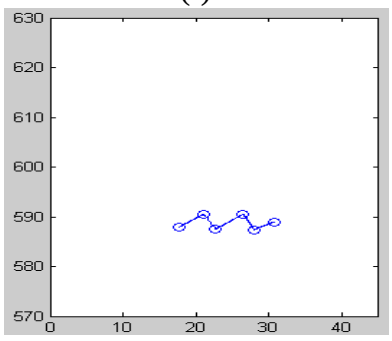
(e)



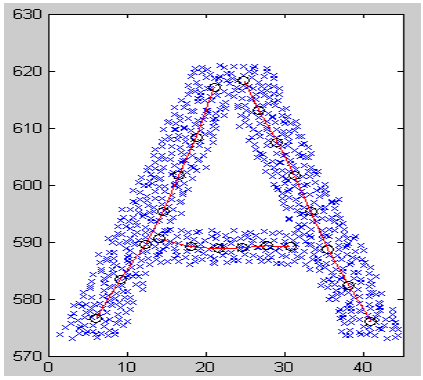
(f)



(g)



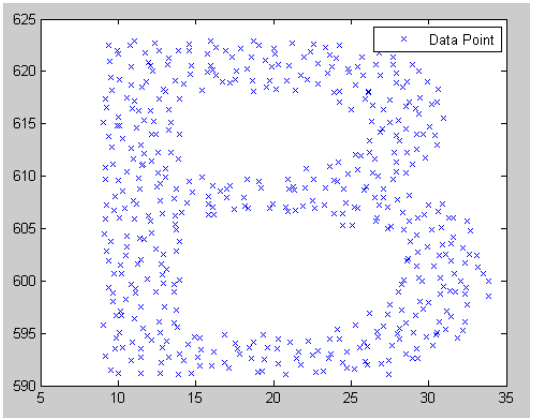
(h)



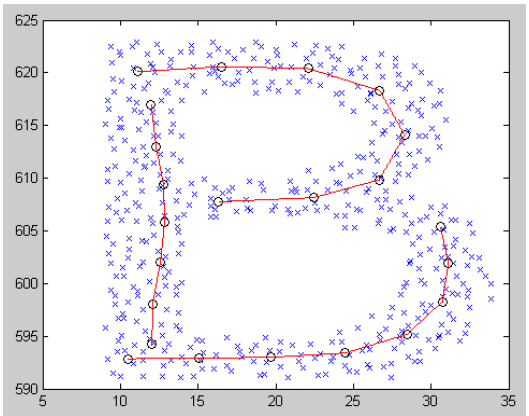
(i)

圖 22: 偵測字母 A。(a)字母 A 的資料集合、(b)叢集整合後的第三階段分群結果、(c)紅色個別分群的資料集合、(d)紅色個別分群的主曲線、(e)綠色個別分群的資料集合、(f)綠色個別分群的主曲線、(g)黑色個別分群的資料集合、(h)黑色個別分群的主曲線、(i)整合所有個別分群主曲線的結果。

對於圖 22 所使用到的各項參數爲: 其修改的模糊 GK 叢集演算法的 β 爲 10 的 15 次方、γ 爲 0.0002、收斂門檻值 ε 爲 0.5、模糊運算子 m 爲 2、其整合個別分群的次數爲 10 次，所使用的雛型個數範圍爲: 10、15、20、25、30、35、40。而模糊主曲線追蹤演算法的叢集重心點個數爲 8 個、連結門檻值爲 0.05、平滑化程度 α 值爲 2。而在圖 22 的(h)中，初始主曲線呈現出彎彎曲曲的形式，但在經過模糊主曲線追蹤演算法的平滑化後可以得到很好的平滑結果。在下面的圖 23 中我們對其它的英文字母展現出相同的實驗結果，而所使用的參數設定如上述。



(a)



(b)

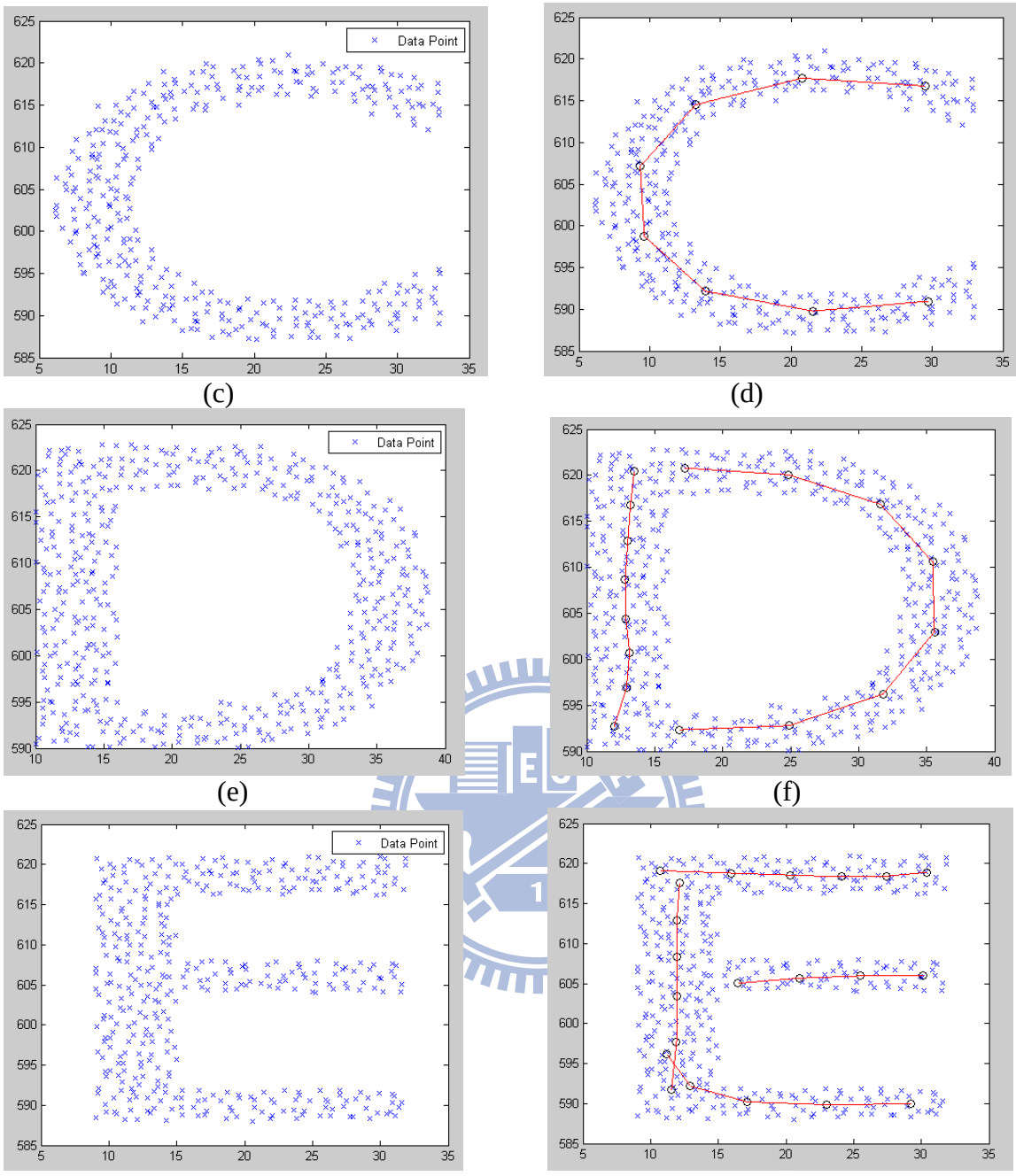


圖 23: 不同英文字母的主曲線偵測。由左到右分別爲左圖爲所採用的字母資料集合(a)、(b)、(e)、(g)，相對應的右圖爲利用本篇論文所提出的方法所得到的結果(b)、(d)、(f)、(h)，而由上到下分別爲針對不一樣的字母所做的測試結果。

第五章 結論與未來展望

由於近年來利用叢集整合技術是一種趨勢，在許多的文獻中談到了叢集整合的許多好處，例如: 能夠獲得一個較穩定而且正確的結果、或者是非常適合分散式的計算。它能夠在不同時間或者是不同的地點各別去得到各別的結果，最後再將所有各別的結果整合起來得到最終的結果。因此隨著目前資訊量的倍增，其分散式的計算更顯得的重要，而且叢集整合的技術提供了我們一個穩定且較正確結果，所以在這樣的前提之下，叢集整合技術非常適合我們欲發展一個更穩定且更適合於廣泛情況的叢集化演算法偵測主曲線。

因此我們在本論文中提出了一個叢集整合技術偵測主曲線的演算法，與一般叢集化演算法偵測主曲線較不同的是在一開始我們挑選了一個重心點具有體積的 GK 叢集化演算法，並且利用階層聚合演算法來得到最佳化的分群。最後由叢集整合技術來得到正確的最終分群。對於一個欲輸入的資料集合來說，我們不僅僅能夠找出含有多重物件的主軸線，更能找到具有交叉性質物件的主曲線。而且在我們的第四章的實驗中顯示出，我們能夠將兩個相互交叉的橢圓正確的把它們分離開來，還有我們可以很正確的去找出英文字母筆畫的邊界，並將這些筆畫所屬於的資料集合給取出來，找到它們的主曲線。

而在偵測主曲線時我們使用修改的模糊主曲線追蹤演算法，因爲在原本的模糊主曲線追蹤演算法的形成相關性圖形跟去除多餘邊所形成的迴圈步驟中，並有沒很嚴謹的去制訂出一個廣泛的去除迴圈的方法。因此我們自己定義一個形成相關性圖形和去除多餘邊所形成的迴圈的方法，由此來得到最佳且最適合對於所取出來個別分群的資料集合，它的初始主曲線。

在未來我們想加強我們所提出的演算法，讓它不只是能夠找到英文字母的筆畫主曲

線，我們更希望在將來可以找到中國字的筆畫主曲線。並希望能去找出一個資料集合中包含有中國字、英文字母、和含有符號字元等多重物件、或者是不規則相交叉性質的複合式多重物件，或是能夠去辨識不同國家的語言文字。在最後我們希望在一個有雜訊影響的資料集合中，我們所提出的叢集整合技術偵測主曲線仍然能夠順利的去找出資料集合中所包含的交叉物件以及非交叉的多重物件。

參考文獻

[1] M. Chen, Z. Cheng, and Y. Liu, “A Robust Algorithm of Principal Curve Detection,” *Proc. Int’l Conf. Pattern Recognition*, Vol. I, pp. 429-432, 2004.

[2] X. Liu, and Y. Jia, “A Bottom-up Algorithm for Finding Principal Curves with Applications to Image Skeletonization”, *Pattern Recognition*, Vol. 38, pp. 1079-1085, 2005.

[3] D.C. Stanford, A.E. Raftery, “Finding Curvilinear Features in Spatial Point Patterns: Principal Curve Clustering with Noise,” *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 22, pp. 601-609, 2000.

[4] B. Kegl, A. Krzyzak, T. Linder, and K.Zeger, “A Polygonal Line Algorithm for Construction Principal Curves,” *Neural Information Processing Systems*, Vol. 11, pp. 501-507, 1998.

[5] J.J. Verbeek, N. Vlassis, and B. Krose, “A K-segments Algorithm for Finding Principal Curves,” *Pattern Recognition Lett.*, Vol. 23, pp. 1009-1017, 2002.

[6] H. Yan, “Fuzzy Curve-Tracing Algorithm,” *IEEE Trans. Syst., Man, Cybern., Part B: Cybern.*, Vol. 31, pp. 768-780, 2001.

[7] B. Lam and H. Yan, “Complex Curve Tracing Based on A Minimum Spanning Tree Model and Regularized Fuzzy Clustering”, *IEEE int’l Conf. on Image Processing*, Vol. 3, pp. 2091-2094, 2004.

[8] M. S. Baghshah and S. B. Shouraki, “A Fuzzy Clustering Algorithm for Finding Arbitrary Shaped Clusters”, *IEEE/ACS int’l Conf. on Computer Systems and Applications*, Vol. 31, pp. 559-566, 2008.

[9] L. Yang, H. Lv., and W. Wang, “Soft Cluster Ensemble Based on Fuzzy Similarity Measure”, *Proc. IMACS Multiconfereceon*, pp. 1994-1997, 2006.

[10] H. Luo, F. Kong, and Y. Li, “Combining Multiple Clusterings via K-modes Algorithm,” *Proc. Advanced Data Mining and Applications*, pp. 308-315, 2006.

[11] A. L. Fred, and A. K. Jain, “Combining Multiple Clusterings Using Evidence Accumulation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 27, pp. 835-850, 2005.

[12] R. Avogadri, and G. Valentini, “Ensemble Clustering with A Fuzzy Approach,” *Supervised and Unsupervised Ensemble Methods and their Applications, Studies in Computational Intelligence*, vol. 126, 2008.

[13] X. Z. Fern and C. E. Brodley., “Random Projection for High Dimensional Data Clustering: A Cluster Ensemble Approach,” *Proceedings of the International Conference on Machine Learning*, pp. 186-193, 2003.

[14] P. Hore, L. Hall, and D. Goldgof., “A Cluster Ensemble Framework for Large Data Sets,” *IEEE Int’l Conf. Systems, Man, and Cybernetics*, Vol. 4, pp. 3342-3347, 2006.

[15] R. Babuska, P.J. van der Veen, U. Kaymak, ”Improved Covariance Estimation for Gustafson-Kessel Clustering”, *IEEE Int’l Conf. Fuzzy Syst.*, Vol. 2, pp. 1081-1085, 2002.

[16] U. Kaymak and M. Sentnes, “Fuzzy Clustering with Volume Prototypes and Adaptive Cluster Merging,” *IEEE Trans. Fuzzy Syst.*, Vol. 10, pp. 705-712, 2002.