國立交通大學

多媒體工程研究所

碩 士 論 文

使用強韌叢集演算法的叢集整合技術

Robust clustering for cluster ensemble

研 究 生：石耿維
指導教授：王才沛　教授

中 華 民 國　九 十 八　年 八 月

使用強韌叢集演算法的叢集整合技術
Robust clustering for cluster ensemble

研 究 生：石耿維　　Student：Keng-Wei Shih
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

使用強韌叢集演算法的叢集整合技術

學生：石耿維　　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

本論文研究目的在於探討強韌的叢集演算法（robust clustering）應用在叢集整和（cluster ensemble）技術上的分析。叢集整合演算法包括三個主要的部份：1.產生個別分群的叢集演算法、2.將個別分群用一個資料結構來整合各結果、3.如何由這個整合的資料結構來得出最終的分群。第一部份我們將使用強韌叢集演算法（robust clustering）做分群，本論文中將會使用NC（Noise Clustering）及PFCM（Possibilistic Fuzzy c-Means Clustering）作為個別分群的叢集演算法，第二部份我們將使用代表資料點兩兩關係的co-association矩陣來紀錄各個叢集後的結果，第三部份接著再從co-association矩陣中利用階層式叢集演算法（hierarchical agglomerative clustering algorithm）找出最終分群結果並去除雜訊。最終分群的結果好壞會利用*NMI*（normalized mutual information）做最後分析。測試的資料中我們會用各種資料，包涵高斯、曲線，分析各種叢集演算法後的結果，有雜訊和無雜訊對叢集整合的影響，以及從最後階層樹中分析出最終的分群數目。

Robust clustering for cluster ensemble

Student：Keng-Wei Shih　　Advisor：Tsai-Pei Wang

Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

ABSTRACT

In this paper, we discuss using robust clustering for cluster ensembles. Cluster ensemble algorithms include three main parts: (1) Generate clusters by applying different clustering algorithms; (2) combine multiple results as a data structure; (3) find the clustering result from data structure. In the first part, we use robust clustering algorithm to generate multiple clustering results. Robust clustering algorithms used include noise clustering (NC) and possibilistic fuzzy c-means clustering (PFCM). In the second part, we use co-association matrix to organize clusters. In the third part, we discard noises from the co-association matrix and then use hierarchical agglomerative clustering algorithm to find the final cluster. The quality of combination results can be evaluated with normalized mutual information (NMI). The data sets, used for testing include Gaussian and half rings with or without noise.

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，在這兩年的研究所生涯中，一直很有耐心的教導我們，研究中遇到瓶頸和困難時，老師總是會給予適當的建議和解決方向，讓我的論文研究能夠順利完成，在此要感謝老師的栽培。也要感謝陳玲慧教授和楊敏生教授擔任我們的口試委員，在口試的過程中給了我很多寶貴的建議，使我的論文能夠更完整，特此致謝。也謝謝學校給我們充分的資源，和硬體設備，讓我們在做實驗的過程中能夠更順暢。

我要感謝我的家人，在求學期間一直支持我，給我鼓勵，讓我能夠順利的完成研究所學業。接下來要感謝實驗室的同仁。和我一起口試的劉強和昇毅，有了你們的陪伴，互相勉勵與支持，讓我寫論文的過程不會感到孤獨。也要感謝俞邦、偉誌和崇桂學弟們，有了你們，讓實驗室增加了許多歡笑。再次謝謝你們與王才沛教授。

目錄

圖目錄

表目錄

第一章　簡介

1.1　研究動機

隨著資訊科技的進步，資料越來越多且繁雜，如何能夠有效的將資料分類，或是從一群資料中找出有意義的資訊，一直都是一個很重要的議題。資料叢集（data clustering）從過去到現在一直被廣泛的探討，其應用非常廣，如：資料組織與分類、圖形檔案壓縮、影像分析、機器學習等，叢集的方法有很多種[13]，舉例，像是階層式叢集演算法（hierarchical clustering），階層式演算法主要是根據鄰近的點互相做結合，結合成樹狀結構，無法根據整體資料形狀或是大小當作依據來做結合，而且當資料結合後就不會再改變。另外，分割式叢集演算法（partitional clustering）其中像是清晰式（crisp）的叢集演算法（hard c-means; HCM），其每個資料點只能表示屬於一個叢集，另一種是模糊的叢集演算法（Fuzzy c-means; FCM），每個資料點對每個叢集都有一個歸屬程度。HCM和FCM皆可以依資料的形狀或大小當作依據做分群，使其更有彈性，所以常被用來偵測例如，線段、曲線、圓形、或特定形狀[14][15][16]，但分割式叢集演算法無法預先知道分群的數目，而且當有雜訊時對分群的結果也會有很大的影響。

現實的資料中，雜訊是不可避免的問題，一個好的叢集演算法應該要能夠有處理雜訊的能力，過去已經有許多強韌叢集演算法（robust clustering）用來解決雜訊的問題，像是在FCM中將雜訊視為一個虛擬的“雜訊叢集”，定義雜訊雛型（Noise Prototype）的雜訊叢集法（noise clustering; NC）[1]。利用可能性去分析的可能性C均值（Possibilistic c-Means; PCM）[2]、模糊可能性C均值（Fuzzy Possibilistic c-Means; FPCM）[3]、可能性模糊C均值（Possibilistic Fuzzy c-Means; PFCM）[4]，以上方法共通的目的都是希望能夠降低雜訊對正確叢集的歸屬程度。NC定義將虛擬雜訊叢集和所有點為一個固定距離，依距離值大小來區分雜訊跟資料，距離的大小會決定資料點於虛擬雜訊叢集的歸屬

程度。而可能性的叢集演算法則會受到初始直影響最終分群，所以以上最終分群結果都會受到參數設定的影響。

叢集演算法的種類眾多，且一直都有新的叢集演算法出現，但是各個方法所適用的資料組特性以及所能找出的叢集特性都有其限制的，與其發展出一個應用範圍廣且高效能的叢集演算法，[5]中提到一個有效的方法是使用叢集整合（cluster ensemble）技術，其方法是對同一組資料產生多個不同的叢集結果，再整合這些個別的結果來產生一個更穩定且正確性更高的分群。而且叢集整合非常適合用於平行與分散式系統的資料處理，將資料分散在不同地方計算在加以整合。近幾年有許多叢集整合的相關研究但只有少數的研究有針對雜訊的處理。[7]的做法是事先做雜訊過濾後再做叢集整合，[6]是利用co-association矩陣來辨認雜訊的方法，但還是需由使用者指定要去除的雜訊點比例。

有雜訊的資料做叢集整合在正確性上會有很大的影響，本篇論文中，將使用叢集整合技術，並將強韌叢集演算法（robust clustering）作為叢集整合的叢集演算法，把各個分群後的結果做一個整合，找出最後分群結果。

1.2 章節摘要

在第二章我們會先介紹已知的強韌叢集演算法以及叢集整合技術。第三章我們將使用強韌叢集演算法應用在叢集整合技術，並分析整合後的結果找出雜訊，接著使用single link和average link演算法做最後的分群。第四章為實驗結果及討論。第五章做最後總結及討論未來可發展的方向。

第二章　文獻探討

2.1　強韌叢集演算法

在實際的資料中，往往會受到很多種因素，常使得資料中多了雜訊，這些雜訊容易影響最終的分群結果，所謂的強韌（robust）指的是當使用叢集演算法做分群時，所得到的分群結果能夠減少受雜訊的影響，而降低最終分群的嚴重錯誤[10]。目前常見的強韌叢集演算法[1][2]中，常常在分群過程中對雜訊資料點加入了調整函數（tuning function），藉以希望雜訊對分群的結果等影響能夠降到最低。

FCM 被廣泛的使用在影像處理及資料分類，主要是它透過一個矩陣來紀錄點跟叢集的歸屬值，但在參有雜訊的資料的狀況下，雜訊點的歸屬值依然很高，故FCM的效果很差，[1]提出一個叫雜訊叢集法（Noise Clustering; NC），此方法是在FCM中另外加了一個虛擬的“雜訊叢集”。雜訊點對於虛擬雜訊叢集的歸屬程度較高，藉此降低雜訊點對真正叢集的歸屬程度，另外定義了一個雜訊雛型（noise prototype）：雜訊叢集對所有的點皆為同一距離。表示說這個距離值將會影響雜訊叢集的歸屬程度，若距離太小，則非雜訊的點的最大歸屬值會落到雜訊叢集裡。若距離設定太大，則NC會退化成一般的FCM。對於密度不均的叢集，雜訊叢集的距離選擇上就會有很大的瓶頸。進而也有針對距離選擇的相關研究[8]。

另外，[2]提出了可能性C均值演算法（Possibilistic c-Means; PCM），它在FCM成本函數中另外加一項來避免所有歸屬程度皆為零的叢集化結果，其中加入了”影響範圍”η_j參數，會使得雜訊會有較小的歸屬程度，以減少雜訊對分群結果的影響。比較困難的是，η_j的初始值會影響最終分群結果，且會有產生多個重疊的叢集的頃向。

為了解決PCM重疊叢集的問題，[4]提出可能性模糊C均值演算法（Possibilistic Fuzzy c-Means; PFCM），將PCM與FCM做一個整合，改善了PCM重疊叢集的問題和雜訊影響

FCM歸屬值問題，並增加了*a*和*b*兩個參數來微調PFCM計算中心點的membership和typicality的比重，增加PFCM的彈性。

2.2 叢集整合

近幾年有許多相關的研究在於結合各種分群後的結果，從中分出正確性高且穩定的分群結果。早期關於叢集整合一篇論文[5]中說明了“知識再利用”（knowledge reuse）用於叢集的概念，“知識”指的就是個別分群。另外論文中也指出叢集整合用於平行和分散式運算有很好得效果。[11]中透過理論去驗證叢集整合在穩定度及的正確性的優點。

根據[7]中叢集整合演算法包括三個主要的部份：1.產生個別分群的叢集演算法、2.將個別分群用一個資料結構來整合各結果、3.如何由這個整合的資料結構來得出最終的分群，從圖1可以看清楚叢集整合的架構圖。第一部份需要有多個分群的結果，在[7]中透過k-means演算法並用不同的參數來產生不同的分群結果，[9]對於隨機初始化的k-means做了一個穩定度和正確性的評估，以上所介紹的都是利用k-means去做分群，其屬於硬式叢集（crisp clustering），因為分群結果只屬於一個叢集，而[9]提出了軟式叢集整合，將叢集整合的輸入改成模糊形式的表示方式，另一個方法[12]是用投影的方式將高維度的特徵向量（feature vectors）投影到多個隨機的低維度空間。

關於整合多個分群結果的方式，許多研究最常使用的是co-association矩陣。co-association矩陣是個對稱矩陣，其中矩陣中每個元素代表的是兩資料點被分在同一群的比例，[7]即是把各個k-means的結果整合成co-association矩陣，[9]則是透過模糊的形式，將歸屬程度表示對於叢集的強度來產生co-association矩陣。[8]提出的Cluster-based Similarity Partitioning Algorithm (CSPA) 也是co-association矩陣的實際應用。

得到最終分群的方法裡，對於未知叢集數目的狀況下，[7]中使用階層式聚合叢集

法（Agglomerative hierarchical clustering），包涵了單一連結（single-link; SL）、平均連結（average-link; AL) 和[12]裡所使用的完全連結（complete-link; CL），而最終分群使用大生命期條件（maximum-lifetime criterion）來決定，以得到階層聚合過程中最穩定的一個分群。

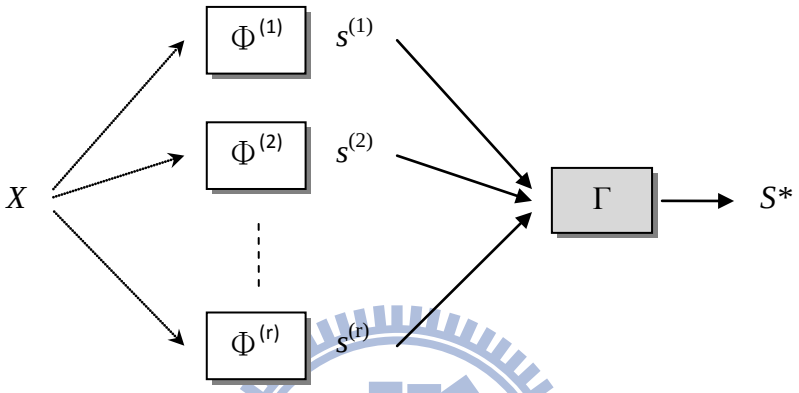


圖 1：叢集整合架構，資料點 X 經過各個演算法 Φ 所得到的最終分群結果 $s^{(r)}$，再經過整合函數 Γ 做整合求得到整合資料 S^*。

第三章　使用強韌叢集演算法的叢集整合技術

本章我們將說明使用強韌叢集演算法應用在叢集整合技術上，我們將依照叢集整合的三本步驟一一說明，3.1節說明使用強韌叢集演算法做分群。3.2節說明將各個分群結果以一個資料結構表示。3.3節說明從此資料表中過濾掉雜訊後計算最後的分群。

3.1　強韌叢集演算法

以下將介紹的強韌叢集演算法都是由FCM為基礎衍生出來的演算法，我們先從FCM開始介紹。接著會介紹雜訊叢集演算法和可能性模糊C均值演算法，並說明每個演算法的特性。

3.1.1　模糊C均值演算法

FCM是一種根據C均值演算法衍生而來的分群法，其透過模糊邏輯的概念，希望能進一步提升分群的效果。FCM不再像HCM每個資料點各個叢集而言只有“屬於”或“不屬於”表示。FCM加入了模糊的概念，故資料點將不再絕對屬於任何叢集，而是以一個介於0～1之間的數字來表示資料點的歸屬程度。假設我們有 k 個分群 $C_1, C_2, \ldots, C_k$，以及 n 個資料點 $X = \{x_1, x_2, x_3, \cdots, x_n\}$，則我們可以以一個 $n \times k$ 的矩陣 U 來表示每個資料點與每個叢集的歸屬程度，我們以 u_{ij} 代表歸屬程度矩陣 U 的元素，指的是 U 中第 i 個資料點對於第 j 個叢集的歸屬值，另外，我們以 V 代表 k 個叢集的中心點，v_i 表示 V 中第 i 個叢集的中心點位置。對一個資料點 x_i 而言，它與各個叢集的歸屬值總和正好等於1：

$$\sum_{j=1}^{k} u_{ij} = 1 \quad , \forall i = 1,2,\ldots,n \tag{1}$$

根據矩陣 U 和中心點矩陣 V，我們可以定義FCM的目標函數（objective function）J：

$$J(U,V)=\sum_{i=1}^{n}\sum_{j=1}^{k}(u_{ij})^{m}\left\|x_i-v_j\right\|^{2} \tag{2}$$

其中，m 是介於$[1,\infty)$之間的模糊化因子。x_i 與 v_j 之間為歐幾里得距離。更新歸屬程度的公式為

$$u_{ij}=\frac{1}{\sum_{c=1}^{k}\left(\frac{\left\|x_i-v_j\right\|^{2}}{\left\|x_i-v_c\right\|^{2}}\right)^{\frac{1}{m-1}}} \tag{3}$$

更新中心點v_j的公式為

$$v_i=\frac{\sum_{j=1}^{n}u_{ji}^{m}x_j}{\sum_{j=1}^{n}u_{ji}^{m}}\quad for\quad i=1,2,..,k \tag{4}$$

我們以[4]裡其中一個範例資料為例，其資料點分佈於圖2，包涵了兩個菱形$\{x_1, x_2, x_3, x_4, x_5\}$、$\{x_6, x_7, x_8, x_9, x_{10}\}$以及兩個雜訊點 x_{12}、x_{11}，其座標在表1。經過FCM（$k=m=2$）計算後的歸屬程度矩陣 U，從表2中很明顯看到雜訊點 x_{12}、x_{11} 對於兩個叢集的歸屬值皆為0.5，就算 x_{12}很遠，但其歸屬值也是0.5，所以FCM對於資料點跟叢集的遠近是無法分辨出來的；而且從圖2中很明顯看到，在有雜訊兩個菱形資料點的歸屬值中，明顯的比無雜訊降低了一些，當雜訊越多，對歸屬程度的影響就更大，故FCM對於雜訊是相當敏感的。

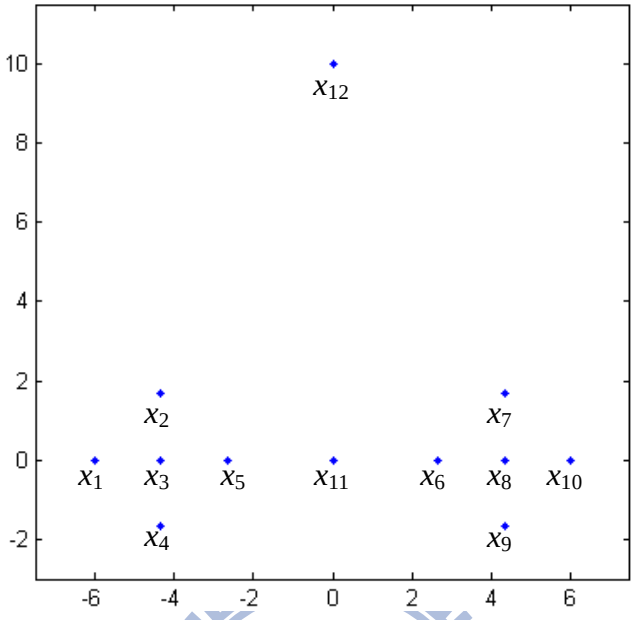


圖 2：範例資料，其中$\{x_1, x_2, x_3, x_4, x_5\}$、$\{x_6, x_7, x_8, x_9, x_{10}\}$為兩個分群，資料點 x_{12}、x_{11} 為雜訊。

表1：範例資料座標

資料點	X座標	Y座標
x_1	-6.00	0.00
x_2	-4.34	1.67
x_3	-4.34	0.00
x_4	-4.34	-1.67
x_5	-2.67	0.00
x_6	2.67	0.00
x_7	4.34	1.67
x_8	4.34	0.00
x_9	4.34	-1.67
x_{10}	6.00	0.00
x_{11}	0.00	0.00
x_{12}	0.00	10.00

表2：表1範例資料經過FCM(m=2)後的歸屬程度，分為有雜訊（有x_{11}, x_{12}）和無雜訊（無x_{11}, x_{12}）

資料點	無雜訊U_{FCM}		有雜訊U_{FCM}	
x_1	0.98	0.02	0.95	0.05
x_2	0.97	0.03	0.98	0.02
x_3	0.99	0.01	0.99	0.01
x_4	0.97	0.03	0.94	0.06
x_5	0.96	0.04	0.96	0.04
x_6	0.05	0.95	0.04	0.96
x_7	0.04	0.97	0.02	0.98
x_8	0.01	0.99	0.01	0.99
x_9	0.03	0.97	0.06	0.94
x_{10}	0.02	0.98	0.05	0.95
x_{11}			0.50	0.50
x_{12}			0.50	0.50

3.1.2 雜訊叢集演算法

雜訊叢集演算法由Dave[1]所發表，在FCM中另外定義一個虛擬的雜訊叢集，而叢集中對每個點為一個固定距離δ，假設資料集合 $X=\{x_1, x_2, x_3, \cdots, x_n\}$ 代表 n 個資料點。假設我們有 k 個真正叢集，我們再外加第 $k+1$ 個叢集為虛擬雜訊叢集。而資料點 x_i 於虛擬叢集的歸屬值 u_{i*} 為：

$$u_{i*} = 1 - \sum_{j=1}^{k} u_{ij} \tag{5}$$

其(5)中 u_{ij} 代表第 i 個資料點於第 j 個叢集的歸屬程度，這也表示資料點對真正叢集的歸屬程度為：

$$0 < \sum_{j=1}^{k} u_{ij} \le 1, \quad \forall i = 1,2,\ldots,n \tag{6}$$

表示說雜訊的資料點對於真正叢集的歸屬值可以是很小的值。而目標函數可以表示為：

$$J(U,V) = \sum_{i=1}^{n}\sum_{j=1}^{k}(u_{ij})^m \left\| x_i - v_j \right\|^2 + \sum_{i=1}^{n}\delta^2\left(1 - \sum_{j=1}^{k} u_{ij}\right)^m \tag{7}$$

歸屬函數的更新公式：

$$u_{ij} = \frac{1}{\sum_{c=1}^{k} \left(\frac{\left\| x_i - v_j \right\|^2}{\left\| x_i - v_c \right\|^2} \right)^{\frac{1}{m-1}} + \left(\frac{\left\| x_i - v_j \right\|^2}{\delta^2} \right)^{\frac{1}{m-1}}} \tag{8}$$

$$v_i = \frac{\sum_{j=1}^{n} u_{ji}^m x_j}{\sum_{j=1}^{n} u_{ji}^m} \quad for \quad i = 1,2,..,k \tag{9}$$

v_i 指的是 u_i 的叢集中心點，另外，Dave [7]建議計算虛擬叢集的距離 δ^2 為：

$$\delta^2 = \frac{\lambda}{kn} \left[\sum_{i=1}^{n} \sum_{j=1}^{k} \left\| x_i - v_j \right\|^2 \right], \tag{10}$$

δ^2 指的是所有資料點與叢集中心點距離平方的總和取平均，λ是一個大於零的數，用來調整 δ^2 的值，而 δ^2 的值將會影響最終雜訊點對於真正叢集的歸屬程度。由(8)的分母第二項可知，所有資料點對於虛擬雜訊的距離只有一個固定值 δ^2，如果 δ^2 值太小會有過多的點會被視為雜訊，若 δ^2 太大會使得虛擬雜訊叢集發揮不了作用，能夠調整 δ^2 大小的參數λ就顯得格外重要。接著我們看λ值的影響，根據表1的資料點經過雜訊叢集演算法後，從表3中可以很明顯看到調整λ值對於歸屬程度*U*的影響。當λ=10 時，虛擬雜訊叢集對真正叢集幾乎沒什麼影響，跟表2 FCM的結果差距不大，對於較遠雜訊點 x_{12} 的歸屬值還是很高 x_{12}=0.42。在λ=1時，可以看出雜訊資料點 x_{12} 對於兩個叢集的歸屬值明顯降低至 x_{12}=0.16，但當λ=0.1時，雖然雜訊資料點 x_{11}、x_{12}的歸屬值雖然很小，但其他資料點的歸屬值明顯的也降低許多，這表示 δ^2 太小，使得所有資料點幾乎都被視為雜訊了，故λ值的選擇將會影響最終分群的結果。

雜訊叢集演算法詳細步驟如下：

步驟一、初始化

i. 隨機初始叢集的中心點 $V^{(0)}$

ii. 設定參數k、m、λ，$1<k<n$，$1<m$

iii. 設定最迴圈數r=1開始，至最大迴圈數 $r_{\max}$

iv. ε：中止條件的臨界值

步驟二、根據公式(10)計算δ^2

步驟三、根據公式(8)計算U^r

步驟四、根據公式(9)更新叢集中心點V^r

步驟五、若$\left(\left\|V^r - V^{r-1}\right\| < \varepsilon\right)$ or $(r > r_{\max})$則離開，否則回到步驟二並遞增r

表3：NC根據不同λ所產生的歸屬程度矩陣

資料點	U_{NC}：λ=10		U_{NC}：λ=1		U_{NC}：λ=0.1	
x_1	0.95	0.04	0.89	0.03	0.58	0.02
x_2	0.97	0.02	0.92	0.03	0.60	0.02
x_3	0.99	0.01	0.99	0.0	0.99	0.01
x_4	0.93	0.06	0.89	0.04	0.60	0.02
x_5	0.96	0.04	0.91	0.04	0.62	0.03
x_6	0.04	0.96	0.04	0.91	0.03	0.62
x_7	0.02	0.97	0.03	0.92	0.02	0.60
x_8	0.01	0.99	0.01	0.99	0.01	0.99
x_9	0.06	0.93	0.04	0.89	0.02	0.60
x_{10}	0.04	0.95	0.03	0.89	0.02	0.58
x_{11}	0.49	0.49	0.42	0.42	0.16	0.16
x_{12}	0.44	0.44	0.21	0.21	0.03	0.03

3.1.3 可能性模糊C均值演算法

可能性模糊C均值演算法（Possibilistic Fuzzy c-means; PFCM）[4]是一個整合了PCM和FCM的強韌模糊叢集演算法，改善了FCM對於雜訊點影響的問題，並且改善了PCM會有多個重疊叢集的問題。PFCM的目標函數為：

$$J_{m,n}(U,T,V)=\sum_{i=1}^{n}\sum_{j=1}^{k}(au_{ij}^{m}+bt_{ij}^{n})\times\left\|x_i-v_j\right\|^2+\sum_{j=1}^{k}\eta_j\sum_{i=1}^{n}(1-t_{ij})^n \tag{11}$$

其中U為FCM的歸屬程度，T為PCM的歸屬程度。u_{ij}和t_{ij}計算如下：

$$u_{ij}=\left(\sum_{c=1}^{k}\left(\frac{\left\|x_i-v_j\right\|}{\left\|x_i-v_c\right\|}\right)^{\frac{2}{(m-1)}}\right)^{-1},1\le i\le n;\quad 1\le j\le k \tag{12}$$

$$t_{ij}=\frac{1}{1+\left(\frac{b}{\eta_j}\left\|x_i-v_j\right\|^2\right)^{\frac{1}{(n-1)}}},1\le i\le n;\quad 1\le j\le k \tag{13}$$

$$v_i=\frac{\sum_{j=1}^{n}\left(au_{ji}^{m}+bt_{ji}^{n}\right)x_j}{\sum_{j=1}^{n}\left(au_{ji}^{m}+bt_{ji}^{n}\right)},1\le i\le k \tag{14}$$

η_j是一個正整數值，根據[2]式子可以表示為

$$\eta_j=K\frac{\sum_{i=1}^{n}u_{ij}^{m}\left\|x_i-v_j\right\|^2}{\sum_{i=1}^{n}u_{ij}^{m}} \tag{15}$$

其中K為一個大於零的正整數，一般都是設定K=1。PFCM分群的結果決定於η_j的值，就像雜訊叢集演算法的δ^2，而NC中δ^2在每一個分群中皆相同，但在PFCM中η_j對於每個資料叢集都是不一樣，因此可以解釋為，在NC中，只有一個雜訊叢集，而在PFCM中每一個真正叢集都有一個相對應的雜訊叢集。

從[3]中提到在PCM中，要得到正確的結果，η_j 計算方式是先利用FCM做分群，接著得到FCM的歸屬值，和叢集中心點，再計算出 η_j，帶入PCM演算法，在此PFCM也會採用一樣的方法取得 η_j 的初始值。以下介紹可能性C均值演算法的詳細步驟。

可能性C均值演算法：

步驟一. 初始化：

1. 隨機初始叢集的中心點 $V^{(0)}$
2. 設定參數 c、m、n，$1<k<n$，$1<m$，$1<n$
3. 設定參數 a、b
4. 設定公式(15)中的參數 K，$K>0$
5. 設定最迴圈數 $r=1$開始，至最大迴圈數 $r_{\max}$
6. ε：中止條件的臨界值

步驟二. 計算 η_i

1. 執行FCM求得 U^{FCM}，V^{FCM}
2. U^{FCM}，V^{FCM} 帶入公式(15)計算出 η_j

步驟三. 根據公式(12)計算歸屬程度 U^r

步驟四. 根據公式(13)計算 T^r

步驟五. 根據公式(14) 更新叢集中心點 V^r

步驟六. 若 $\left(\left\|V^r - V^{r-1}\right\| < \varepsilon\right)$ or $\left(r > r_{\max}\right)$ 則離開，否則回到步驟三並遞增 r

表4是根據表1的資料點經過PFCM後的U和T，表4 中的歸屬程度U和表2 FCM的歸屬程度差不多，而T中很明顯看出雜訊點對於兩個叢集的歸屬值皆很低(x_{12}=0.07)，故能

夠有效的把雜訊分辨出來。

我們將把雜訊叢集演算法和PFCM所求得的歸屬程度U為主要分析雜訊的來源。依照不同參數產生不同的結果，(參數包涵了分群個數、雜訊叢集演算法的δ^2、可能性C均值演算法的η_j)，做為叢集整合的叢集來源。

表4：PFCM的兩個歸屬程度矩陣U_{PFCM}、T_{PFCM}

資料點	U_{PFCM}		T_{PFCM}	
x_1	0.04	0.96	0.64	0.08
x_2	0.03	0.97	0.811	0.11
x_3	0.01	0.99	0.961	0.11
x_4	0.06	0.94	0.66	0.10
x_5	0.04	0.96	0.84	0.16
x_6	0.96	0.04	0.16	0.84
x_7	0.97	0.03	0.11	0.81
x_8	0.99	0.00	0.11	0.96
x_9	0.94	0.06	0.10	0.66
x_{10}	0.96	0.04	0.08	0.64
x_{11}	0.5	0.5	0.35	0.35
x_{12}	0.5	0.5	0.07	0.07

3.2 使用強韌叢集法來獲得co-association 矩陣

我們將強韌叢集演算法所得到的結果計算出co-association矩陣，假設資料集合$X=\{x_1,x_{2,}x_3,\cdots,x_n\}$表示$n$個資料點。一個$X$的分群可由$k$個子集合$C_1,C_2,\ldots,C_k$來表示。我們用$P=\{C_1,C_2,\ldots,C_k\}$代表一個分群，執行多次叢集演算法可以得到多個不同的P，總共有m個分群將被整合：$P_1,P_2,\ldots,P_m$。每個分群當中的叢集個數都可以是不同的，我們以k_q來代表分群$P_q(1\le q\le m)$當中的叢集個數。

為了讓m個分群結果有一致的表示方式，先針對每個個別叢集P_q計算個別$n\times n$的co-association矩陣，以$\boldsymbol{S}^{(q)}$來表示。$s_{ij}^{(q)}$代表$\boldsymbol{S}^{(q)}$的第(i,j)個元素，式子如下：

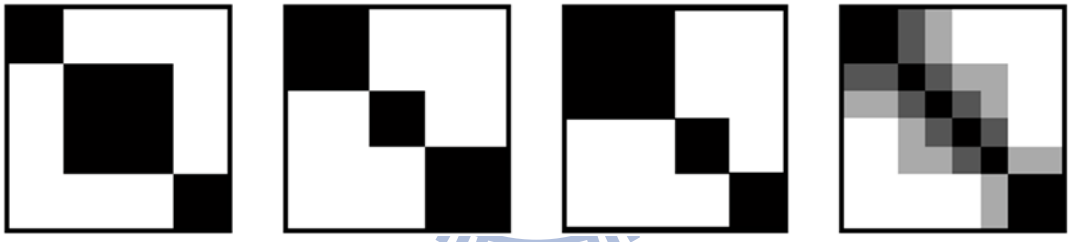
$$s_{ij}^{(q)} = \begin{cases} 1, & c_i^{(q)} = c_j^{(q)} \\ 0, & otherwise \end{cases} \tag{16}$$

$c_i^{(q)}$ 表示叢集 P_q 中，資料點 x_i 所屬的叢集。整合個別分群的co-association矩陣 S^*：

$$S^* = \frac{1}{m}\sum_{k=1}^{m} s_{ij}^{(k)} \tag{17}$$

S^* 代表m個個別分群的co-association矩陣加總的平均。

在此我們舉個例子。假設有八個資料點，執行三次叢集演算法後分群為 $P_1 = \{\{x_1, x_2\}, \{x_3, x_4, x_5, x_6\}, \{x_7, x_8\}\}$ ， $P_2 = \{\{x_1, x_2, x_3\}, \{x_4, x_5\}, \{x_6, x_7, x_8\}\}$ ， $P_3 = \{\{x_1, x_2, x_3, x_4\}, \{x_5, x_6\}, \{x_7, x_8\}\}$，其co-association矩陣分別為圖3(a)、圖3 (b)、圖3 (c)，圖3(d)表示這三個叢集及整合後的co-association矩陣。



(a) (b) (c) (d)

圖 3：(a)、(b)、(c)為 P_1、P_2、P_3 個別的 co-association 矩陣，(d)為整合後的 S^* 矩陣

個別co-association矩陣內只有0或1表示，如此不夠有彈性，我們若要針對模糊式的分群計算co-association矩陣，我們的分群P將以 $n \times k$ 的矩陣U來代表，而U代表的是n個資料點對於 k 個分群的歸屬程度（membership），以FCM來說，它的U滿足以下式子：

$$\sum_{j=1}^{k} u_{ij} = 1, \quad \forall i = 1,2,\ldots,n \tag{18}$$

以FCM來說，個別叢集 P_q 計算個別 $n \times n$ 的co-association矩陣$\boldsymbol{S}^{(q)}$式子可以改成：

$$s_{ij}^{(q)} = \sum_{c=1}^{k} u_{ci}^{(q)} u_{cj}^{(q)}, \quad 1 \le i \le n, \quad 1 \le j \le n \tag{19}$$

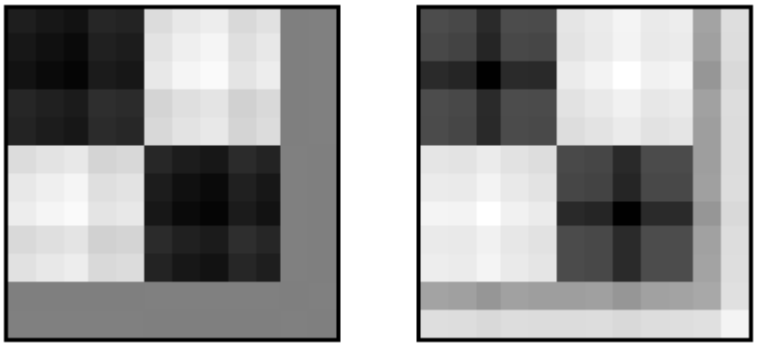
，由原本只有0或1的表示方式改成0～1的表示方式，如此可以使資料更有彈性，提高正確性。

對於強韌叢集演算法，由於資料點對於叢集的歸屬程度可以很小，它的U滿足以下式子：

$$0 < \sum_{j=1}^{k} u_{ij} \le 1, \quad \forall i = 1,2,\ldots,n \tag{20}$$

表示說，若是雜訊點，其對每個叢集的歸屬值可以很小，所以在使用(19)計算個別叢集的co-association矩陣中可能會有某個資料點與其他所有資料點的值都會非常小，我們可以依這些資訊分辨出雜訊。

圖4中是將表1的範例資料經過FCM(a)和NC(b)計算出歸屬程度U後再依(19) 做10次取平均後所算出的co-association矩陣，圖以灰階顯示表示0～1，矩陣內的值表示兩點在同一個叢集關係程度，越高表示兩點皆在同一叢集的程度就越高，圖為了明顯表示故以1－ co-association表示。co-association是一個對稱矩陣，而最右邊兩排是雜訊資料點x_{11}和x_{12}，從(a)和(b)圖比較可以明顯看出兩個分群，而且雜訊點NC顏色比FCM來的淺，就點x_{12}來說，x_{12}對於其它點的關係都很低，表示資料點x_{12}不屬於任何叢集，所以只要是雜訊，在co-association矩陣裡，它對所有點的關係值都很小，我們可以根據這個資訊來做雜訊的辨別，藉此將雜訊過濾掉。



(a) (b)

圖 4：表 1 範例資料的歸屬程度 U 經過 (19) 做 10 次取平均的 co-association 矩陣，(a)FCM(k=m=2)，(b)NC(λ=0.8, k=m=2)

另外，PFCM演算法會有兩個歸屬程度矩陣U和T，我們用T的歸屬程度來計算co-association矩陣，$\boldsymbol{S}^{(q)}$可以改成：

$$s_{ij}^{(q)} = \sum_{c=1}^{k} t_{ci}^{(q)} t_{cj}^{(q)}, \quad 1 \le i \le n \quad 1 \le j \le n \tag{21}$$

圖5是根據表1的資料由PFCM演算法計算的歸屬程度，每個圖都是計算10次後取平均的co-association矩陣。圖5(a)和圖5(b)則是U和T根據公式(19)和(21)計算後所建立的co-association矩陣。圖5(a)雜訊點的關係值明顯有過高的現象，圖5(b)中雖然雜訊點的關係較低，但是兩個分群並沒有很明顯，故我們將U和T相乘求得 $\boldsymbol{S}^{(q)}$：

$$s_{ij}^{(q)} = \sum_{c=1}^{k} u_{ci}^{(q)} t_{ci}^{(q)} u_{cj}^{(q)} t_{cj}^{(q)}, \quad 1 \le i \le n \quad 1 \le j \le n \tag{22}$$

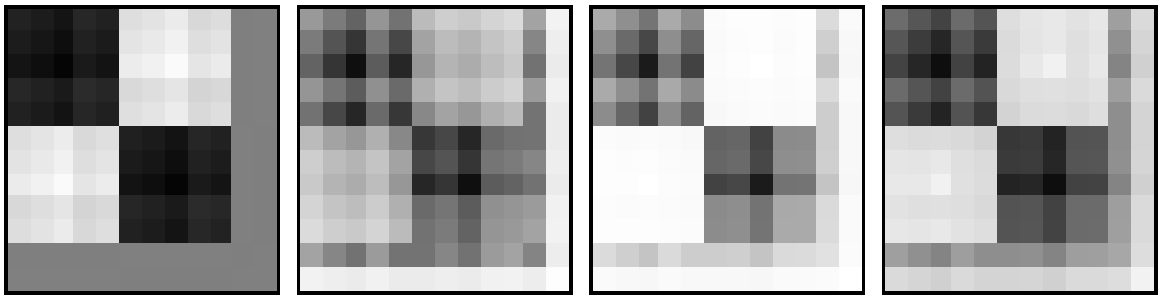
求得的co-association矩陣顯示於圖5(c)。 這兩個分群變得比較清晰。為了增加對比，所以再做開根號求得 $\boldsymbol{S}^{(q)}$：

$$s_{ij}^{(q)} = \sqrt{\sum_{c=1}^{k} u_{ci}^{(q)} t_{ci}^{(q)} u_{cj}^{(q)} t_{cj}^{(q)}}, \quad 1 \le i \le n \quad 1 \le j \le n \tag{23}$$

計算出的矩陣顯示於圖5(d)，明顯看出兩個叢集有變得較明顯。

由於FCM、NC、PFCM分群結果會依初始的叢集中心點的不同而使得分群結果不同，所以我們針對各種演算法產生多個分群結果再將每個個別的co-association矩陣根據

(17)將資料做一個整合。



(a) (b) (c) (d)

圖 5：表 1 範例資料經過 PFCM(a=b=K=1, m=n=2, 叢集數=2)做 10 次取平均的 co-association 矩陣，(a)U_{PFCM} 式子(19)，(b)T_{PFCM} 式子(21)，(c)式子(22)，(d)式子(23)

3.3 從co-association矩陣取得最終分群

上一章節我們介紹了如何把各個叢集演算法的歸屬程度矩陣轉換成co-association矩陣再做整合，本章節將介紹如何從co-association矩陣中得到最終的分群。根據(19)所算出每個分群結果的U的co-association矩陣，再由(17)將結果做整合得到最終的co-association矩陣，其co-association矩陣是一個對稱矩陣，代表的是資料點與資料點之間的關係，越高表示兩點皆在同一叢集的程度就越高，從(19)可知，$S_{ij}^{(q)}$ 大表示資料點 x_{i}、x_{j} 的歸屬程度（membership）相似，$S_{ij}^{(q)}$ 小表示x_{i}、x_{j}的歸屬程度不相似。

在我們對最終分群個數未知的情況下，將使用階層聚合（hierarchical clustering）的單一連結（single-link; SL）或平均連結（average-link; AL）來做分群，由於階層聚合是根據co-association矩陣去做分群，故在做分群之前我們必須先把雜訊去除掉，我們可以從圖4(b)中看到，雜訊點與其他點的關係值都非常的小，若是資料點是在真正叢集裡，那它一定會跟某些點會有很高的關聯性，像圖4(b)較黑的部份。我們判斷資料點x_{i}是否是雜訊的方法為：

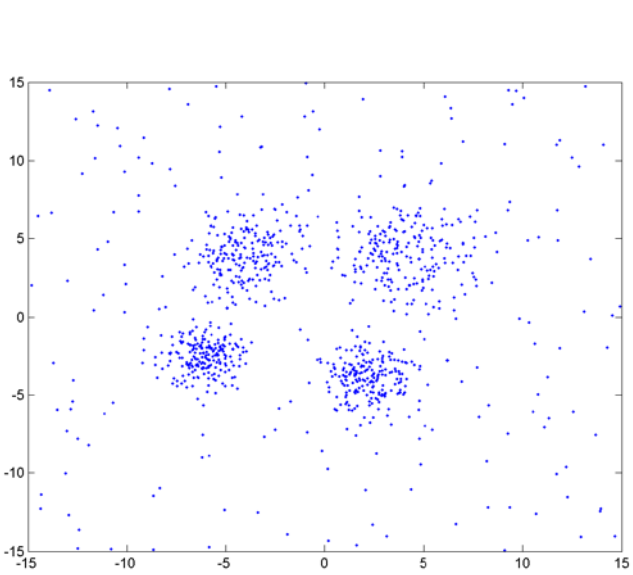
$$\omega_i = \begin{cases} 1, & \max(x_i) - \min(x_i) < threshold \\ 0, & \max(x_i) - \min(x_i) >= threshold \end{cases} \quad ,1 \le i \le n \quad threshold > 0 \qquad (24)$$

ω_i是個0或1的數，用來表示資料點x_i是否是雜訊點，max(x_i)指的是co-association矩陣中資料點x_i與其他資料點最大的關係值，min(x_i)指的是co-association矩陣中與x_i與其他資料點最小的關係值，*threshold*則是一個臨界值。非雜訊的資料點x_i必定會跟某些資料點的關聯性會特別高故max(x_i)大且對其他叢集的歸屬程度很小，所以min(x_i)會很小，所以max(x_i)-min(x_i)的值一定會很大。若是雜訊的資料點，由於雜訊不屬於任何叢集，故max(x_i)很小，當然min(x_i)也會很小，所以max(x_i)-min(x_i)一定會很小。既然min(x_i)都很小，為何不直接用max(x_i)做篩選呢，原因是取min(x_i)則是希望能夠把叢集與叢集之間模糊地帶的雜訊與真正資料點區分出來，像是表1的資料點x_{11}則是介於兩個叢集之間的雜訊點，經過NC演算法後於圖4(b)中可以發現，資料點x_{11}的關係值中，它與其他點的關係都介於0.5左右，若只取max(x_i)則值會太大，但若是計算max(x_i)-min(x_i)則值會變很小，故可以更準確的把雜訊與真正資料區分出來。所以正確叢集的資料點的max(x_i)-min(x_i)會大，而雜訊資料點的max(x_i)-min(x_i)會小，我們定一個臨界值來分出雜訊或是叢集資料點，因此我們可以從這些資訊去辨別出是否是雜訊。

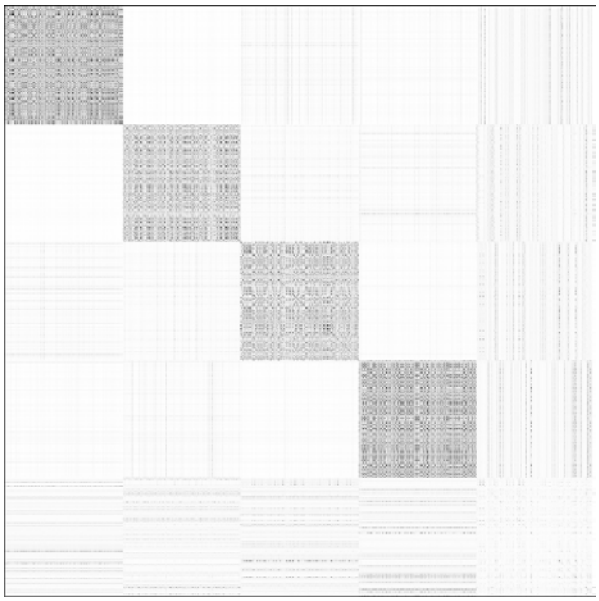
舉例而言，圖6(a)是四個200個點的高斯分佈，另外加入200個雜訊點。圖6(b)是使用NC演算法（k=15, λ=0.1）求得的歸屬程度經過(19)計算出個別的co-association矩陣，做10次分群取平均後得到平均後的co-association矩陣。圖6(d)是根據圖6(b) co-association矩陣求得max(xi)-min(xi)後的結果，其中801～1000為雜訊資料點，很明顯看出雜訊點比真正資料叢集的值來的低，故我們可以訂定一個臨界值（threshold）將雜訊與真正資料給區隔開來。另外，圖6(c)是使用FCM計算max(xi)-min(xi)後的結果，圖中的雜訊點的值和真正資料點的值差距並不大，所以不容易把雜訊給區隔開來。

我們將小於 *threshold* 的資料點從co-association矩陣中去除掉，剩下的再做階層聚

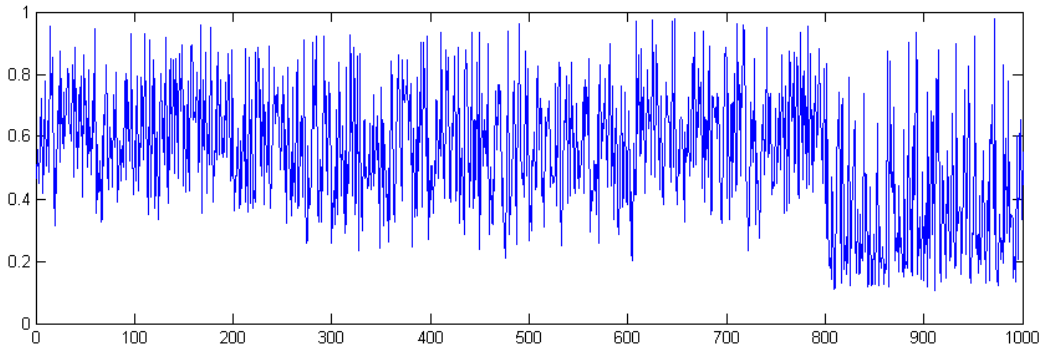
合分群，要找到最終分群的方法是根據階層聚合分群的lifetime[7]。定義k個叢集的lifetime指的是階層圖中選擇到k個叢集的範圍，如圖7(b)的階層圖，2個叢集的lifetime是l_2，而3個叢集的lifetime的l_3，4個叢集的lifetime是l_4，即是每個階層的生命期（lifetime）。就l_4來說，最大值是0.8657，最小值是0.7927，lifetime 則是最大減最小l_4=0.8657-0.7927=0.0838。圖7(b)中，l_2=0.071、l_3=0.0393、l_4=0.0838，其中l_4最大，這也表示階層聚合在4個叢集的時候，穩定度最高，所以圖7(b)我們的最大生命期（maximum-lifetime criterion）則是l_4=0.0838，4個叢集數，根據最大生命期的叢集數當作最終的分群數目，圖7(a)為最終的分群結果。



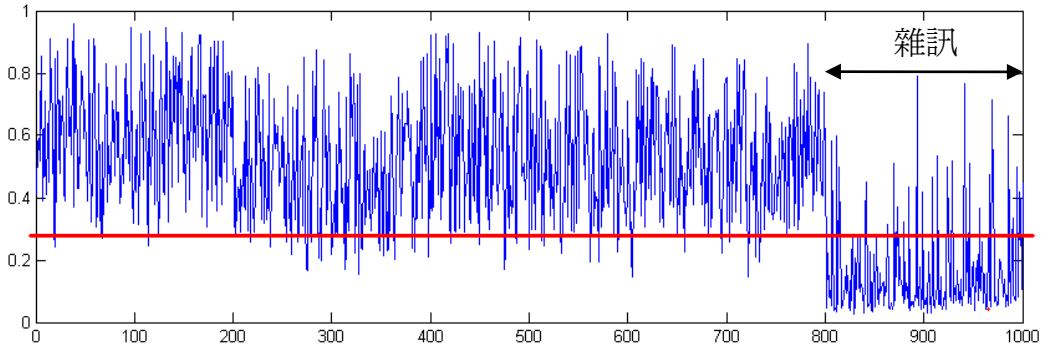
(a)



(b)NC: co-association



(c)FCM



(d)NC

圖 6：(a)資料點總共 1000 個，4 個 n=200 的 gaussian 外加 200 個雜訊，(b)經過 NC 做 10 次取平均後的 co-association 矩陣。(c)和(d)是經過 FCM 和 NC 後的 co-association 矩陣，再計算 max(x_i)-min(x_i)後的結果，其中橫座標是資料點。

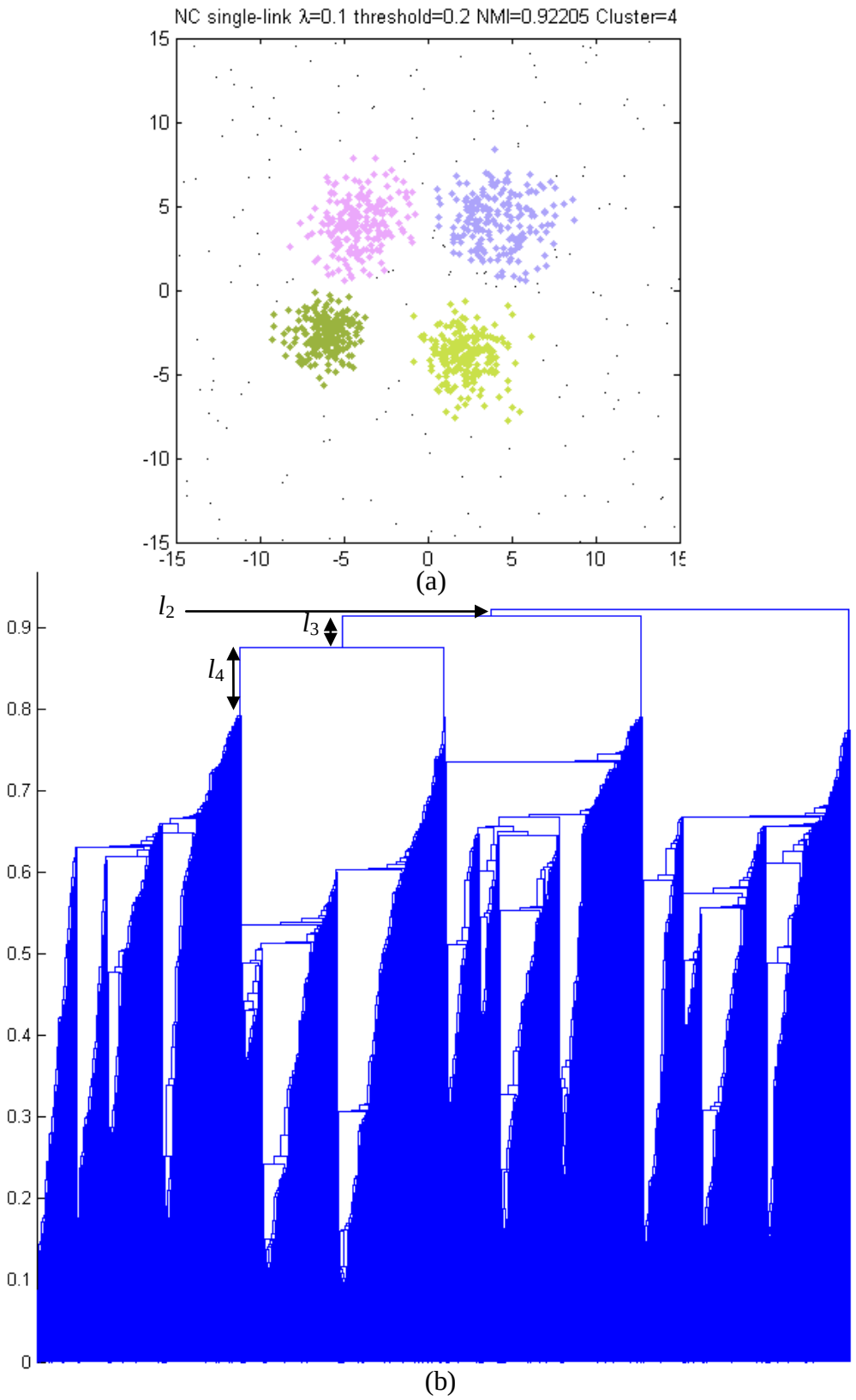


圖 7：將圖 6 的 co-association 矩陣去除雜訊後剩下的資料做 single-link 後的結果，(a)即是最大生命期=4 的最終分群結果，(b)是 single-link 產生的樹狀圖，lifetime l_2、l_3、l_4 中，其中 l_4 的 lifetime 最長。

3.4 最終分群結果的正確度分析

為了能夠分辨出分群結果的正確性，我們將採用[5][7]所介紹的正規劃共同資訊量（normalized mutual information; *NMI*）來評估分群結果的正確性。*NMI*式子表示如下：

$$NMI\left(P^a,P^b\right)=\frac{I\left(P^a,P^b\right)}{\sqrt{H\left(P^a\right)H\left(P^b\right)}} \tag{25}$$

其中 $H(P^a)$ 指的是分群 P^a 的熵（entropy）：

$$H(P^a)=-\sum_{i=1}^{k_a}\frac{n_i^a}{n}\log\left(\frac{n_i^a}{n}\right) \tag{26}$$

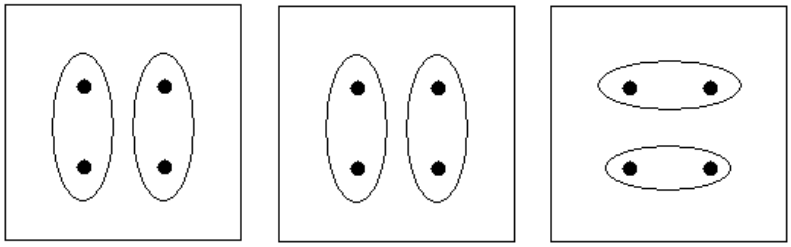
k_a 表是叢集數目，n_i^a 表示分群 P^a 中的屬於第 C_i^a 個叢集的資料點數目，n 為資料點總數。另外 $I\left(P^a,P^b\right)$ 是 P^a 和 P^b 的共同資訊量（mutual information），式子如子下：

$$I\left(P^a,P^b\right)=\sum_{i=1}^{k_a}\sum_{j=1}^{k_b}\frac{n_{ij}^{ab}}{n}\log\left(\frac{\frac{n_{ij}^{ab}}{n}}{\frac{n_i^a}{n}\cdot\frac{n_j^b}{n}}\right) \tag{27}$$

n_{ij}^{ab} 指的是資料點屬於分群 P^a 中的叢集 C_i^a 和分群 P^b 中的叢集 C_j^b 的數目，完整的*NMI*式子如下：

$$NMI\left(P^a,P^b\right)=\frac{\sum_{i=1}^{k_a}\sum_{j=1}^{k_b}n_{ij}^{ab}\log\left(\frac{n\cdot n_{ij}^{ab}}{n_i^a\cdot n_j^b}\right)}{\sqrt{\left(\sum_{i=1}^{k_a}n_i^a\log\left(\frac{n_i^a}{n}\right)\right)\left(\sum_{j=1}^{k_b}n_j^b\log\left(\frac{n_j^b}{n}\right)\right)}} \tag{28}$$

很明顯的 $NMI\left(P^a,P^a\right)=1$，當$P^a$與$P^b$中的分群彼此都不相同時則*NMI*=0都是，從圖8可以看到圖8(a)是理想分群結果，圖8(b)的分群結果與圖8(a)故*NMI*=1，圖8(c)則與圖8(a)的分群結果都不相同，故*NMI*=0。我們將帶入實際分群結果和理想分群結果計算*NMI*來辨別結果的好壞。



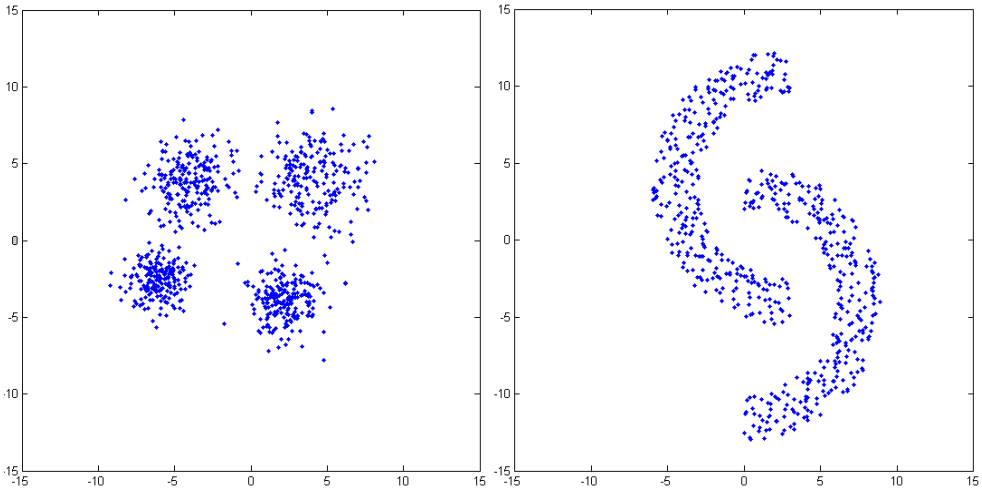
(a) (b) (c)

圖 8：(a)是理想分群結果，(b)與(a)相同，故 *NMI*=1，(c)分群的結果明顯與(a)都不相同，故 *NMI*=0

由於資料中會加入雜訊，而雜訊點應不屬於真正叢集的一部分，但在計算*NMI*時必須知道每個資料點屬於哪個叢集，如果把所有雜訊點視為一個叢集，這樣有點不合理，因為雜訊點之間是沒有任何關係的，所以我們將把每個雜訊資料點視為一個單一的叢集，假如一組資料中包涵了兩個叢集和一百個雜訊，那理想分群結果將會有2個真正叢集加100個雜訊點叢集。.

第四章　實驗結果

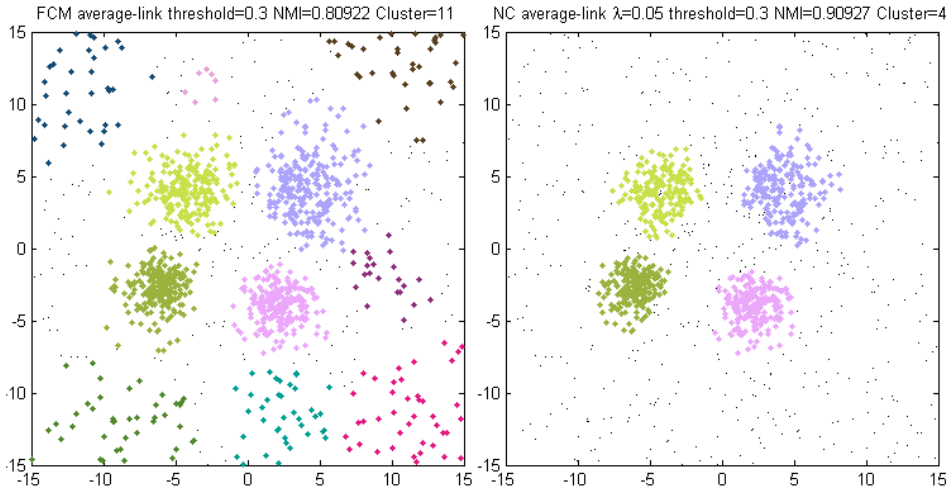
這一章，我們將針對兩組資料做強韌叢集演算法搭配叢集整合技術，一個資料組包涵了高斯分佈，另一個則是兩個半圓，如圖 9。圖 9(a)是四個高斯產生的資料群，各 200 個資料點，圖 9(b)則是兩個半圓，各 200 個資料點。



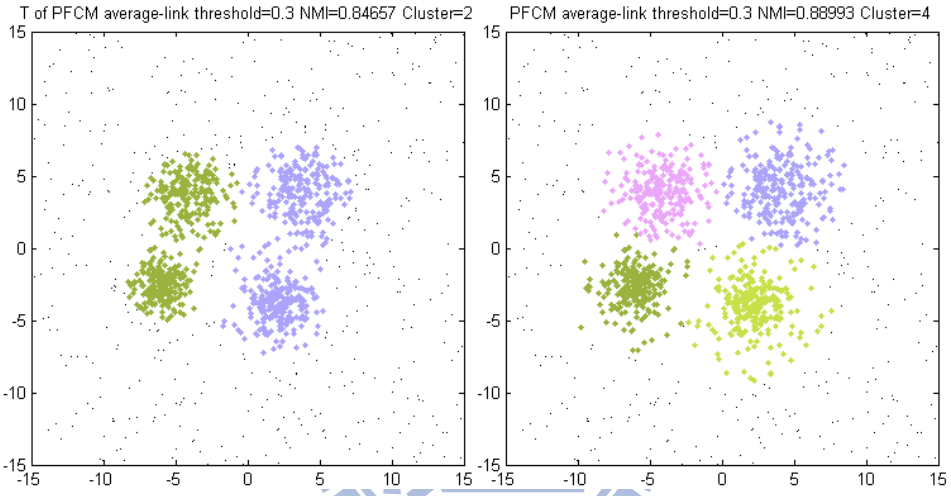
(a)高斯分佈　　(b)半圓

圖 9：兩組測試資料，(a)4 個高斯的群聚每個叢集都是 200 個點，(b)是兩個半圓各 200 個點

我們將第三章的流程套用再 FCM、NC、PFCM 上，由於我們由階層聚合中找出最終分群數，所以我們對於這三個演算法分群的叢集數目會大於理想的叢集數。圖 10 是把圖 9(a)的資料點加入 500 個雜訊後做 FCM、NC、PFCM 的 T(21)和(23)的結果，分群數都是 k=15，做 10 次得到個別的 co-association 矩陣再取平均求得整合後的 co-association 矩陣後 *threshold* 取 0.3 將雜訊去除，接著再用 average-link 做最後分群。圖 10(a)中可以看到由 FCM 分群後，只有去除掉部份雜訊，而圖 10(b)(c)則明顯的把雜訊給去除並得到正確的最終分群。



(a)FCM (b)NC



(c) T_{PFCM} (d)PFCM

圖 10：四個高斯分佈和 500 個雜訊的資料做雜訊叢集整合後的的結果。(a)FCM、(b)NC(λ=0.05)、(c)T_{PFCM} 公式(21)、(d)PFCM 公式(23)，*threshold*=0.3。

在過濾雜訊步驟中，我們是以(24)計算出的值當作 *threshold* 的參考，故取不同的 *threshold* 造成最終分群的結果也是不同的。圖 11 則是將半圓圖加入 300 個雜訊後使用 NC 演算法，叢集數 k=15、λ=0.05，做 10 次得到個別的 co-association 矩陣再取平均求得整合後的 co-association 矩陣，*threshold* 分別取 0.5、0.4、0.3、0.2、0.1 後使用 single-link 做最後分群。從圖 11 可見 *threshold* 越小，越少資料點被視為雜訊點，如圖 11(a)*threshold* 取 0.1 和圖 11(e)*threshold* 取 0.5 相比，圖 11(a)較少資料點被視為雜訊。

圖 11 各個分群結果*threshold*從 0.1～0.5 的*NMI*分別為圖 11(a) *NMI*=0.61311，圖 11(b) *NMI*=0.84286，圖 11(c) *NMI*=0.88059，圖 11(d) *NMI*=0.81027，圖 11(e) *NMI*=0.67276。在去除雜訊點後的分群結果中，圖 11(a)的最大生命期所選到的叢集數k=768，這很顯然是受到雜訊資料點的影響。接著當*threshold*漸漸增加後，可以看到正確的分群。從圖 11(d)中，這兩條半圓曲線開始被截成很多段，圖 11(e)半圓被截成很多段的狀況更明顯，而且最終分群數目也變成個別線段的分群數*k*=15。在過度去除雜訊的狀況下會出現這種現象是因為我們的co-association矩陣的資料是取自叢集演算法的歸屬程度，而在群跟群分界之間的資料點的歸屬程度就沒有特別明顯的表示歸屬哪一個叢集，就像表 3 的資料點 x_{11}的歸屬值在表中只介於 0.5～0.3 之間。故當*threshold*值增加時，介於群跟群之間的資料點可能就會被當作雜訊，故會產生一段一段的現象。

另外，圖 12 跟圖 11 使用一樣的資料集，一樣使用 NC 演算法。差別在於，NC 的叢集數是亂數介於 5～20 區間，其效果並沒有比圖 11 來的好。

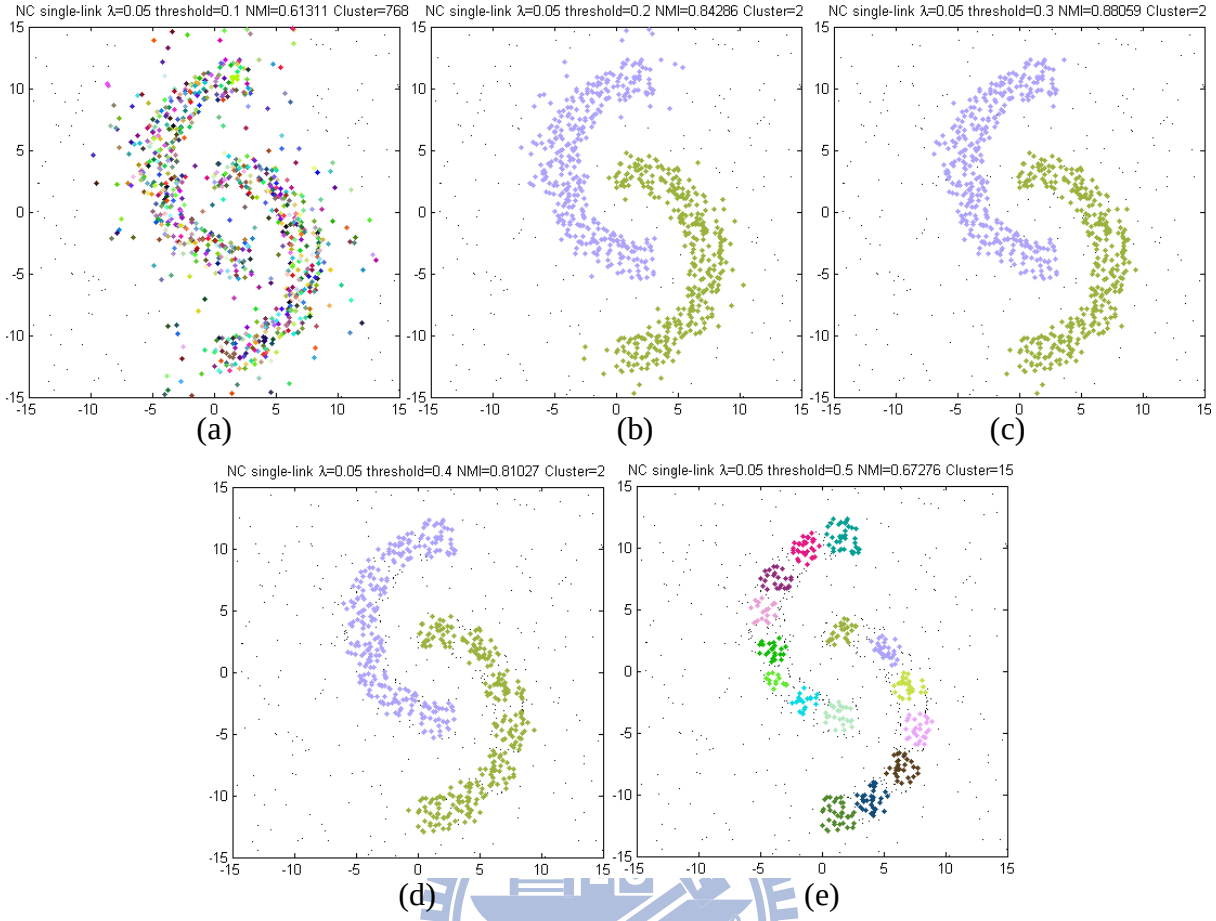


圖 11：兩個半圓加入 300 個雜訊，使用 NC 演算法(叢集數=15，λ=0.05)做整合後不同 threshold 的結果。(a)*threshold*=0.1，(b)*threshold*=0.2，(c)*threshold*=0.3，(d)*threshold*=0.4，(e)*threshold*=0.5

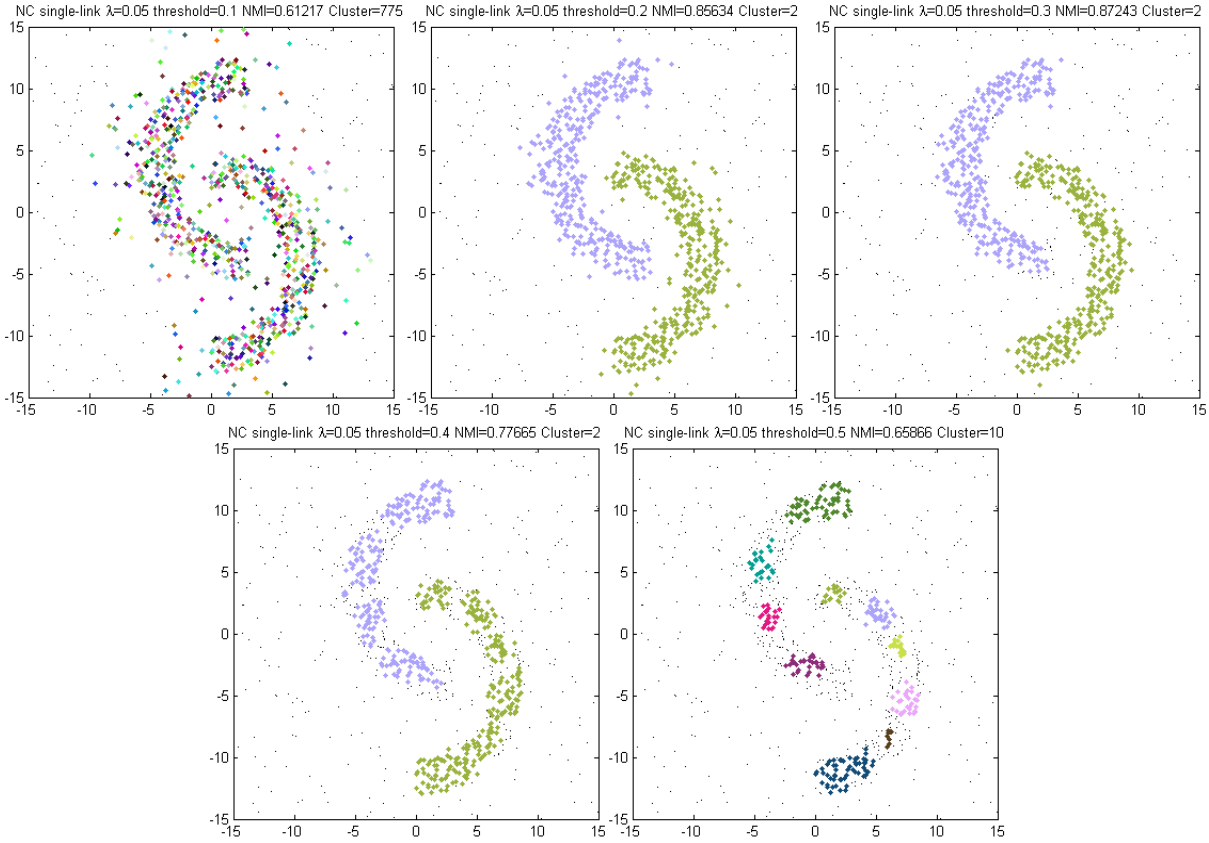


圖 12：兩個半圓加入 300 個雜訊，使用 NC 演算法(叢集數=[5,20]區間，λ=0.05)做整合後不同 threshold 的結果。(a)*threshold*=0.1，(b)*threshold*=0.2，(c)*threshold*=0.3，(d)*threshold*=0.4，(e)*threshold*=0.5

我們再進一步看各種 *threshold* 對不同雜訊量不同資料形狀以及不同叢集數去做叢集整合後的影響，圖 14 為四個高斯分佈資料點和圖 15 為兩個半圓的資料點，各別加入 0、100、300、500 個雜訊，使用 2 到 20 個叢集數於 FCM、NC、PFCM 演算法，計算 10 次結果再做叢集整合。每個圖表中，橫座標為初始的叢集數目，縱座標是最終分群的 *NMI*。*NMI* 值越高，表示分群結果越準確。我們觀察到雜訊很少的時候 *threshold* 越小 *NMI* 曲線越高，而當雜訊增加後反而是 *threshold* 大的 *NMI* 值高。我們以圖 11 中使用 NC 演算法做分群的當例子，圖 14(b)是無雜訊的資料點使用 NC 後的結果，其 *NMI* 最高的曲線是 *threshold*=0.1 時最佳。當雜訊點增加到 300 時，圖 14(h)*NMI* 最高曲線是在 *threshold*=0.2。雜訊點增加到 500 時，*threshold*=0.3 的 *NMI* 曲線也越來越高。這現象

表示雜訊點比例的增加影響到所有資料點的歸屬程度，使得雜訊資料點與真正資料點之間的歸屬不再那麼明確。但 NC 與 FCM 比起來，NC 的 *NMI* 曲線確實比 FCM 高。另外，NC 比 PFCM 來的穩定且正確性更高。

在雜訊很多情況下，似乎 FCM 和 NC 以最佳 *NMI* 曲線來比較感覺差距並不大。就以圖 15(j)和圖 15(k)來說，FCM 最佳的是叢集數=16，*threshold*=0.4，跟 NC 的 *threshold*=0.3，看似 *NMI* 只有差 0.05 左右。實際結果於圖 13(i)(f)中。圖 13 中，FCM 的 NMI 值最高是圖 13(i) threshold=0.4，而 NC 圖 13(f)的 threshold=0.3，雖然 FCM 的結果跟理想分群差很多，但 NMI 值依然是 0.781 而 NC 的是 0.878 只差 0.097。但為什麼只差這麼少，原因在於計算 NMI 時也有考慮雜訊，每個雜訊點視為單一個叢集，雜訊的比例，像圖 10 中真正資料是 4 個高斯分佈共 800 個點另外加上 500 個雜訊點，雜所以以這些圖看來，至少 NMI=0.85 以上才屬於正確分群。

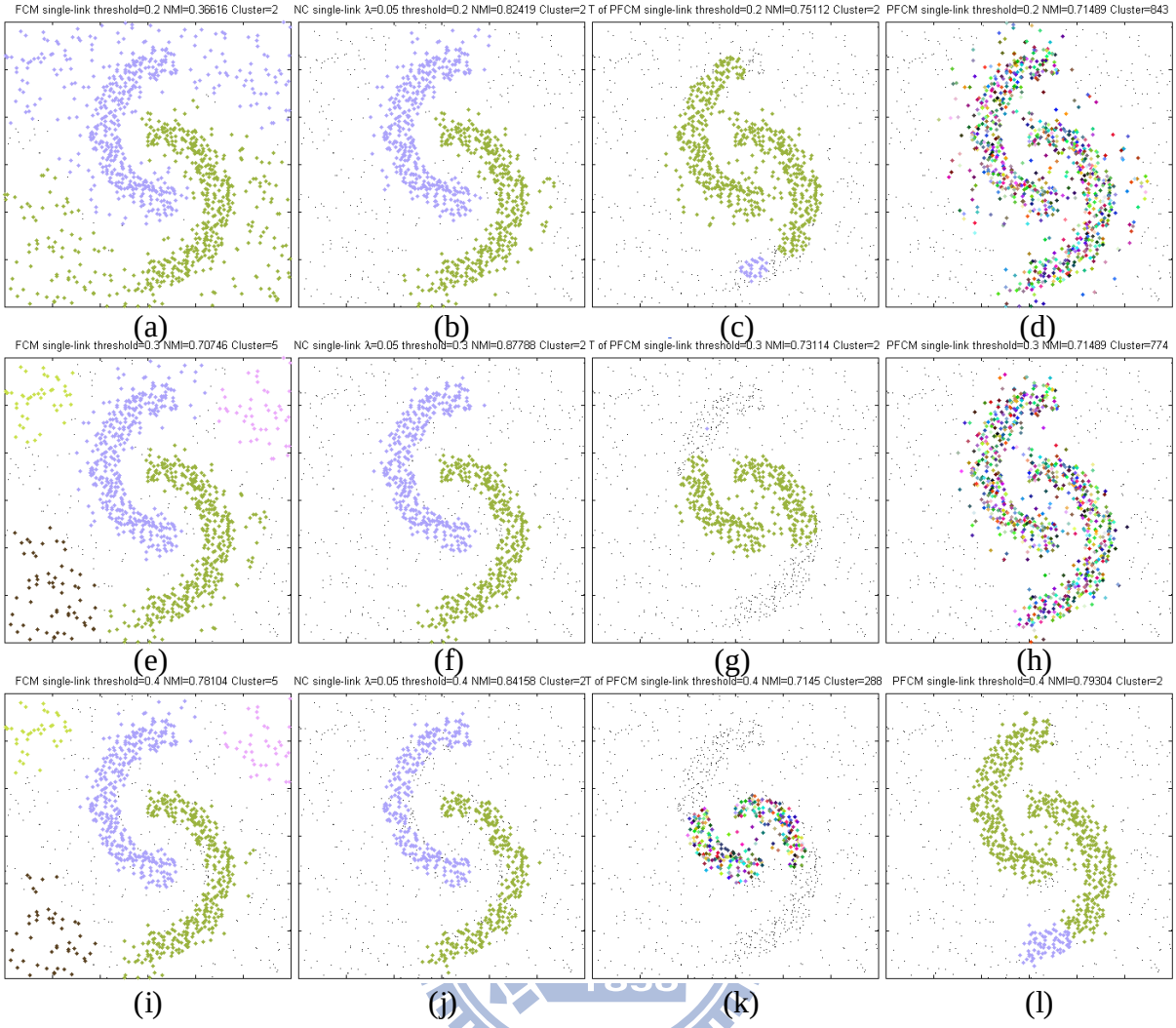


圖 13：雙半圓加入 500 個雜訊點，分別使用 FCM、NC、T_{PFCM}、PFCM，叢集數=16，threshold 分別取 0.2、0.3、0.4 的結果，其中(a)(e)(i)是 FCM，(b)(f)(j)是 NC，(c)(g)(k)是 T_{PFCM}，(d)(h)(l)是 PFCM。

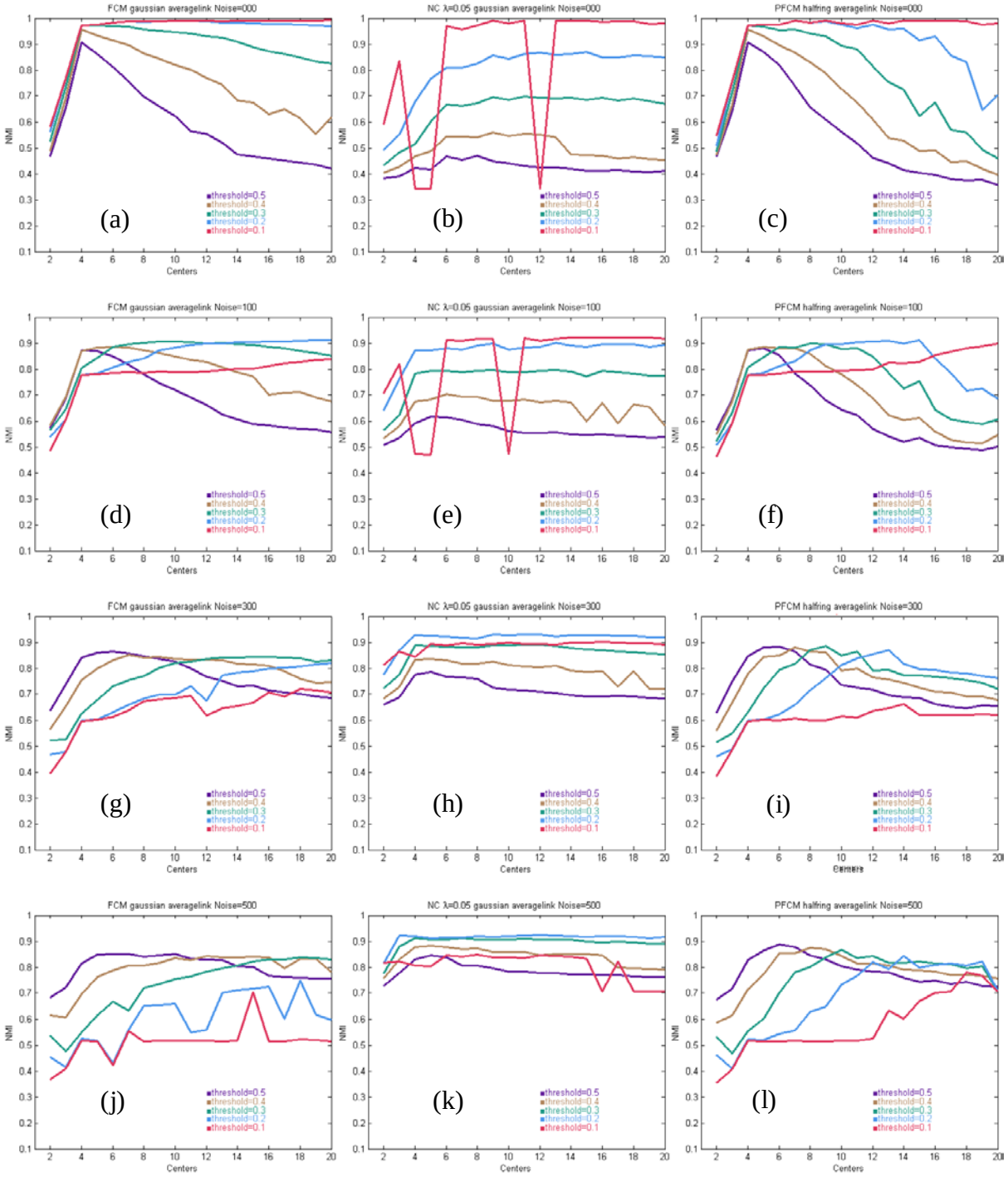


圖 14：4 個高斯分佈資料點，FCM、NC、PFCM 整合，不同 *threshold* 和不同雜訊比例的 *NMI* 曲線圖。(a)(b)(c)分別是 FCM、NCP、FCM 無雜訊的 *NMI* 曲線圖，(d)(e)(f)分別是 FCM、NCP、FCM 加入 100 個雜訊的 *NMI* 曲線圖，(g)(h)(i) 分別是 FCM、NCP、FCM 加入 300 個雜訊的 *NMI* 曲線圖，(j)(k)(l) 分別是 FCM、NCP、FCM 加入 500 個雜訊的 *NMI* 曲線圖。

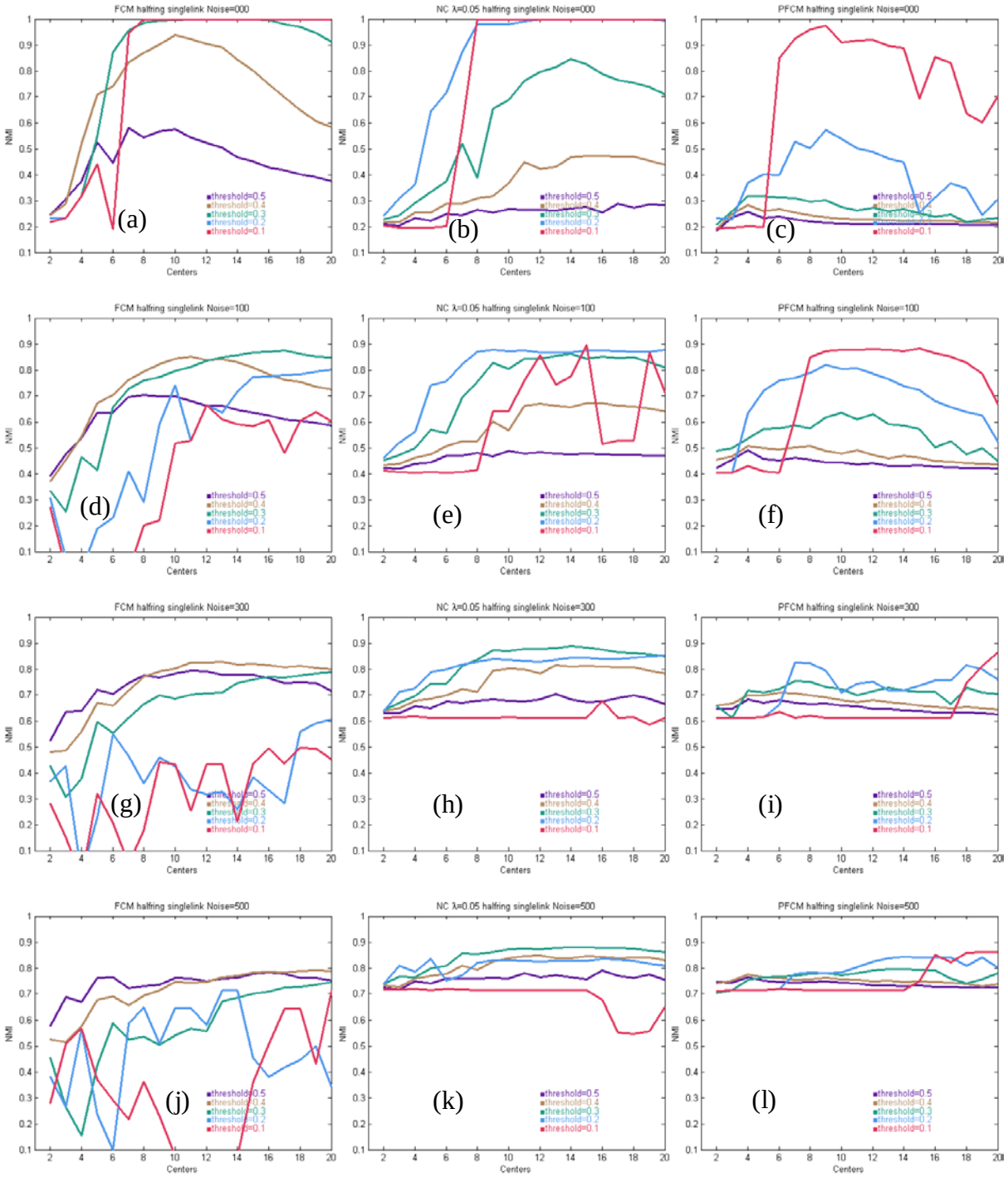


圖 15：雙半圓資料點，FCM、NC、PFCM 整合，不同 *threshold* 不同雜訊的 *NMI* 曲線圖。(a)(b)(c)分別是 FCM、NCP、FCM 無雜訊的 *NMI* 曲線圖，(d)(e)(f)分別是 FCM、NCP、FCM 加入 100 個雜訊的 *NMI* 曲線圖，(g)(h)(i) 分別是 FCM、NCP、FCM 加入 300 個雜訊的 *NMI* 曲線圖，(j)(k)(l) 分別是 FCM、NCP、FCM 加入 500 個雜訊的 *NMI* 曲線圖。

第五章　未來展望

本篇論文中，我們將強韌叢集演算法用在叢集整合中，過程中包涵了強韌叢集演算法的選擇，叢集結果的整合，和從整合的結果中找出最終分群。本篇論文的結果顯示，將強韌叢集演算法用在叢集整合與使用一般叢集演算法做叢集整合，相較之下較不受雜訊比例影響，將雜訊去除並找出正確的結果，但各種資料的最佳分群結果的參數設定像是叢集法的初始分群個數，NC 的 λ、PFCM 的 *a* 和 *b* 參數，都是不相同的，我們必須針對不同資料型態使用不同的參數，不容易找到一個共通的參數對各種資料都是最佳的。另外，雜訊 *threshold* 參考資料的計算是依照 co-association 的值，但它也是會依雜訊和資料分佈而使得 *threshold* 選擇不一。本論文只針對個別的強韌叢集演算法做整合，未來的發展希望能夠試著加入不同的強韌叢集演算法做整合，能夠找出一個共通的參數在不受資料或雜訊影響有好的分群結果，增加演算法的彈性。

現實的資料對於雜訊的影響一直是個很大的問題，本篇論文的測試資料只有測試高斯跟是半圓，未來再推廣到真實的資料，像是應用在影像處理、資料處理、或是辨識系統上，都一定會有很好的效果。

參考文獻

[1] R.N. Dave, "Characterization and detection of noise in clustering", *Pattern Recognition Letters*, vol. 12, pp. 657-664, 1991.

[2] R. Krishnapuram and J.M. Keller, "A possibilistic approach to clustering", *IEEE. Trans. Fuzzy Systems*, vol. 1, pp. 98-110, 1993.

[3] M. Barni, V. Cappellini, and A. Mecocci, "Comments on 'A Possibilistic Approach to Clustering' ", *IEEE Trans. Fuzzy Systems*, vol. 4, pp. 393-396, 1996.

[4] Pal, N.R., Pal, K., Keller, J.M., Bezdek, J. C., "A possibilistic fuzzy c-means clustering algorithm." *IEEE Trans. Fuzzy Systems* vol. 13, pp. 517- 530. 2005

[5] A. Strehl and J. Ghosh "Cluster ensembles -- a knowledge reuse framework for combining multiple partitions", *J. Machine Learning Research*, vol. 3, pp. 583-617, 2002.

[6] Y. Hong, S. Kwong, Y. Chang, and Q. Ren, "Unsupervised data pruning for clustering of noisy data", *Knowledge-Based Systems*, vol. 21, pp. 612-616, 2008.

[7] A.L.N. Fred and A.K. Jain, "Combining multiple clusterings using evidence accumulation", *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 27, pp. 835-850, 2005.

[8] Frank Rehm, Frank Klawonn, Rudolf Kruse "A novel approach to noise clustering for outlier detection", *Soft Compute 2007*, pp 489–494, 2006

[9] K. Punera and J. Ghosh, "Soft Cluster Ensembles", in *Advances in Fuzzy Clustering and its Applications*, Ed. J. Valente de Oliveira and W. Pedrycz (Wiley, 2007).

[10] R. N. Dave and R. Krishnapuram, "Robust Clustering Methods: A Unified View", *IEEE Trans. on Fuzzy system*, vol. 5, pp. 270-293, 1997.

[11] A.P. Topchy, M.H.C. Law, A.K. Jain, and A.L. Fred, "Analysis of consensus partition in cluster ensemble", *Proc. 4th IEEE Int'l Conf. Data Mining* (ICDM), pp. 225-232, 2004.

[12] X.Z. Fern and C.E. Brodley, "Random projection for high dimensional clustering: A cluster ensemble approach", *Proc. 20th Int'l Conf. Machine Learning* (ICML), 2003.

[13] A.K. Jain, M.N. Murty, and P.J. Flynn, "Data Clustering: A Review", *ACM Computing*

Surveys, vol. 31, pp. 264-323, 1999.

[14] R.N. Davé, "Use of the Adaptive Fuzzy Clustering Algorithm to Detect Lines in Digital Images", *Intelligent Robots and Computer Vision VIII*, vol. 1192, pp. 600-611, 1989.

[15] H. Frigui and R. Krishnapuram, "A Comparison of Fuzzy Shell-Clustering Methods for the Detection of Ellipses", *IEEE Trans. Fuzzy Systems*, vol. 4, no. 2, pp. 193-199, May 1996.

[16] T. Wang, "Possibilistic clustering of generic shapes derived from templates", *Proc. 2008 IEEE Int'l Conf. Fuzzy Systems* (FUZZ-IEEE), pp. 1721-1728, 2008.