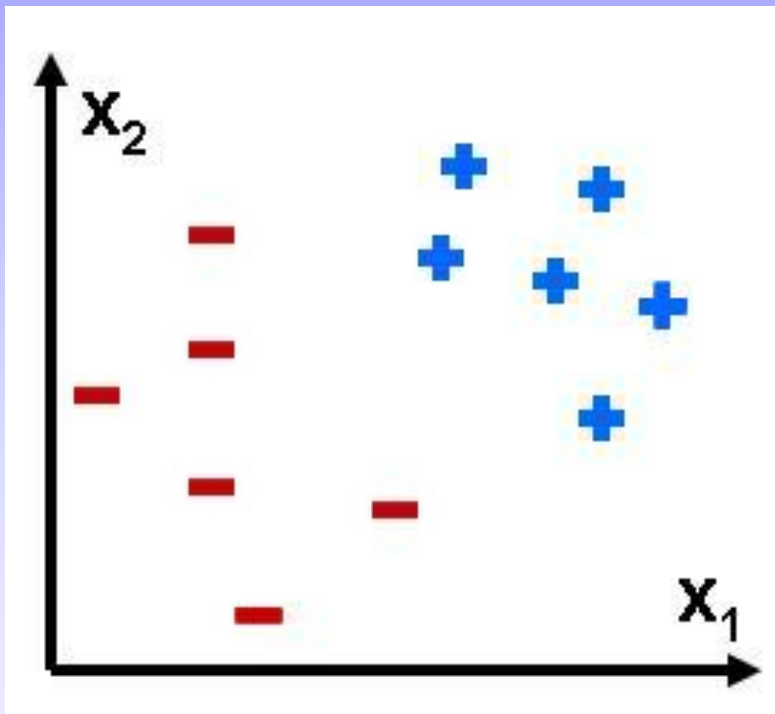


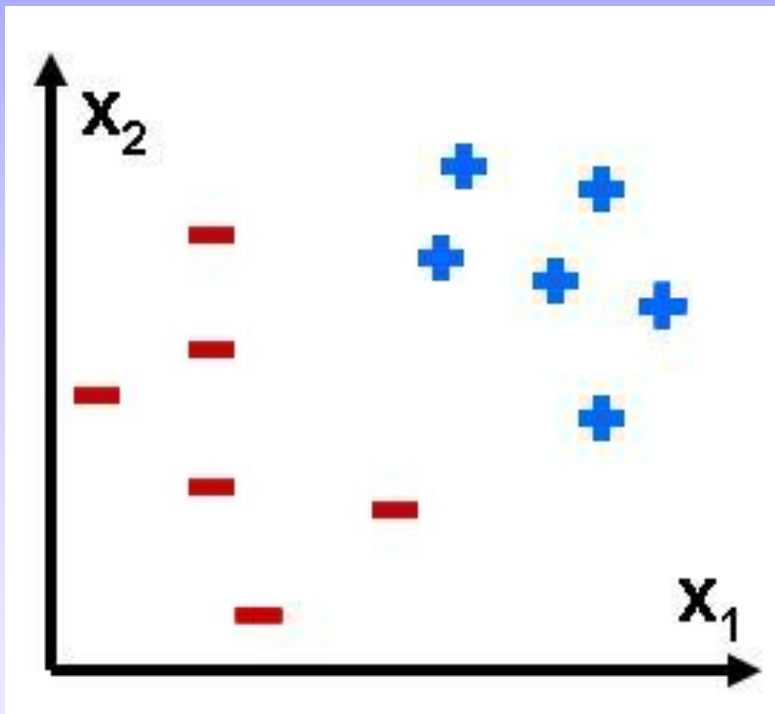
למידה לא מודרכת Unsupervised Learning

עסקנו עד עתה בלמידה מודרכת (Supervised Learning).
בלמידה מודרכת דוגמאות האימון מתוייגות. לדוגמא:



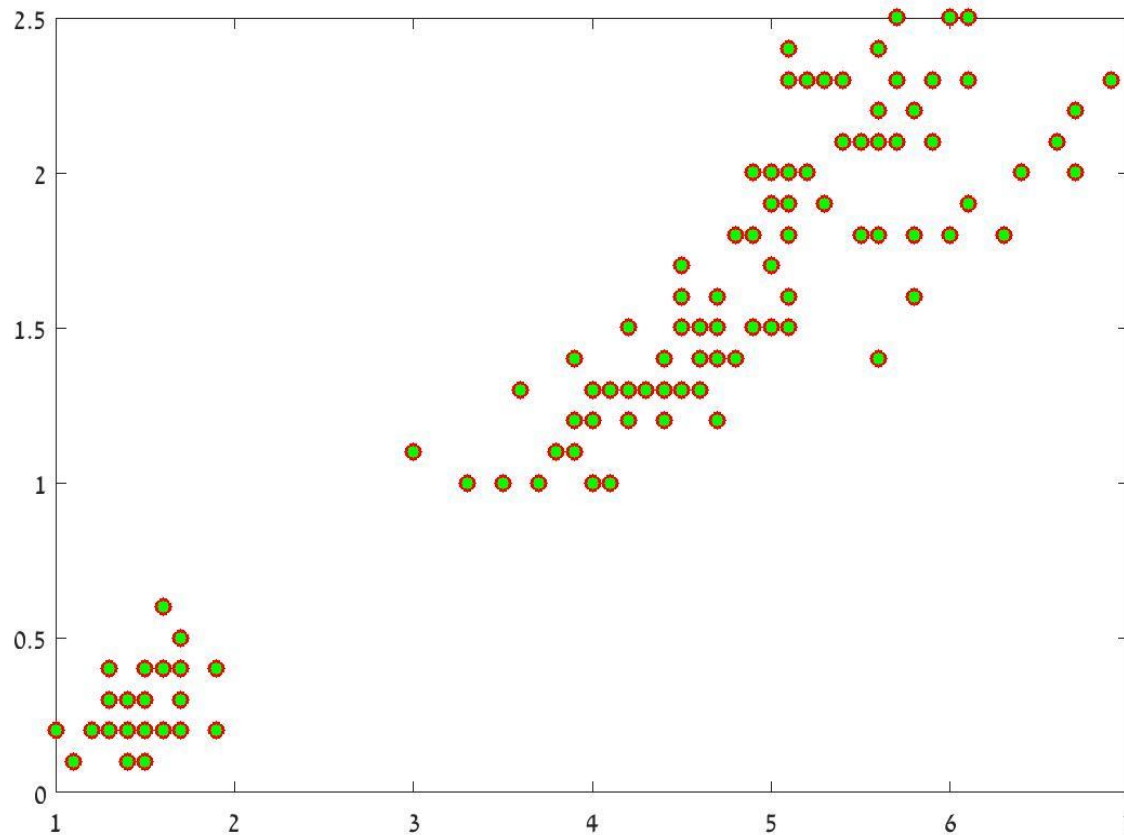
למידה לא מודרכת Unsupervised Learning

עסקנו עד עתה בלמידה מודרכת (Supervised Learning).
בלמידה מודרכת דוגמאות האימון מתוייגות. לדוגמא:



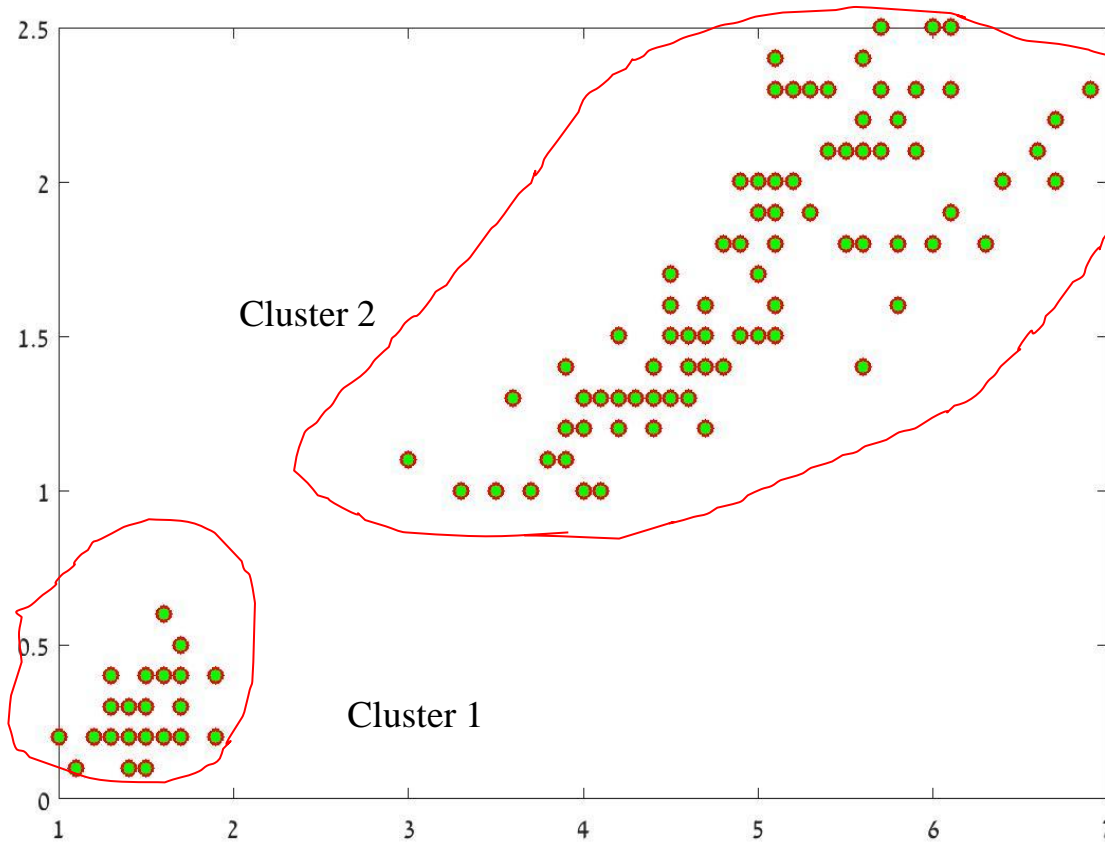
למידה לא מודרכת

- הדוגמאות אינן מתוייגות. או שלכולן אותו תיוג או שהתיוג לא ידוע.
- בהינתן קבוצת הנתונים, האם אפשר למצוא מבנה בנתונים?



למידה לא מודרכת

- לדוגמא, אלגוריתם לא מודרך (unsupervised) עשוי לקבץ את הנתונים לתת קבוצות (צבירים או clusters) באופן קוהרנטי.



למידה לא מודרכת

- אלגוריתם כזה נקרא Clustering algorithm.
- דוגמא לאלגוריתם המבצע Google news :clustering.
- מבצע clustering של אלפי פריטי חדשות ממקורות שונים, ומקבץ אותם לפי הנושא לצבירים שכל אחד מהם מכיל פריטי חדשות דומים.
- <https://news.google.com/>

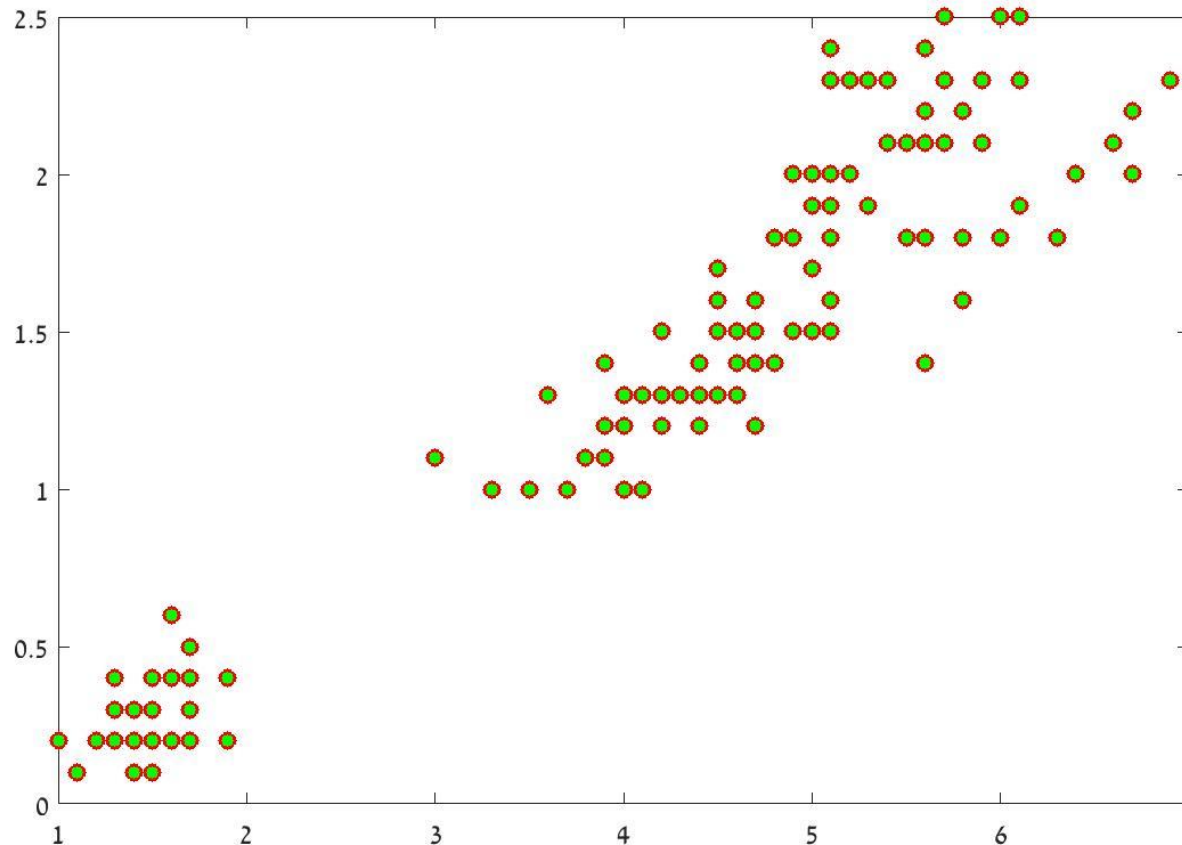
דוגמאות נוספות:

1. אירגון מחשבים ב-clusters. על-ידי clustering של מחשבים הנוטים לעבוד יחד באותו מקום אפשר לחסוך.
2. אנליזה של רשתות חברתיות
3. אנליזה של נתונים אסטרונומיים.



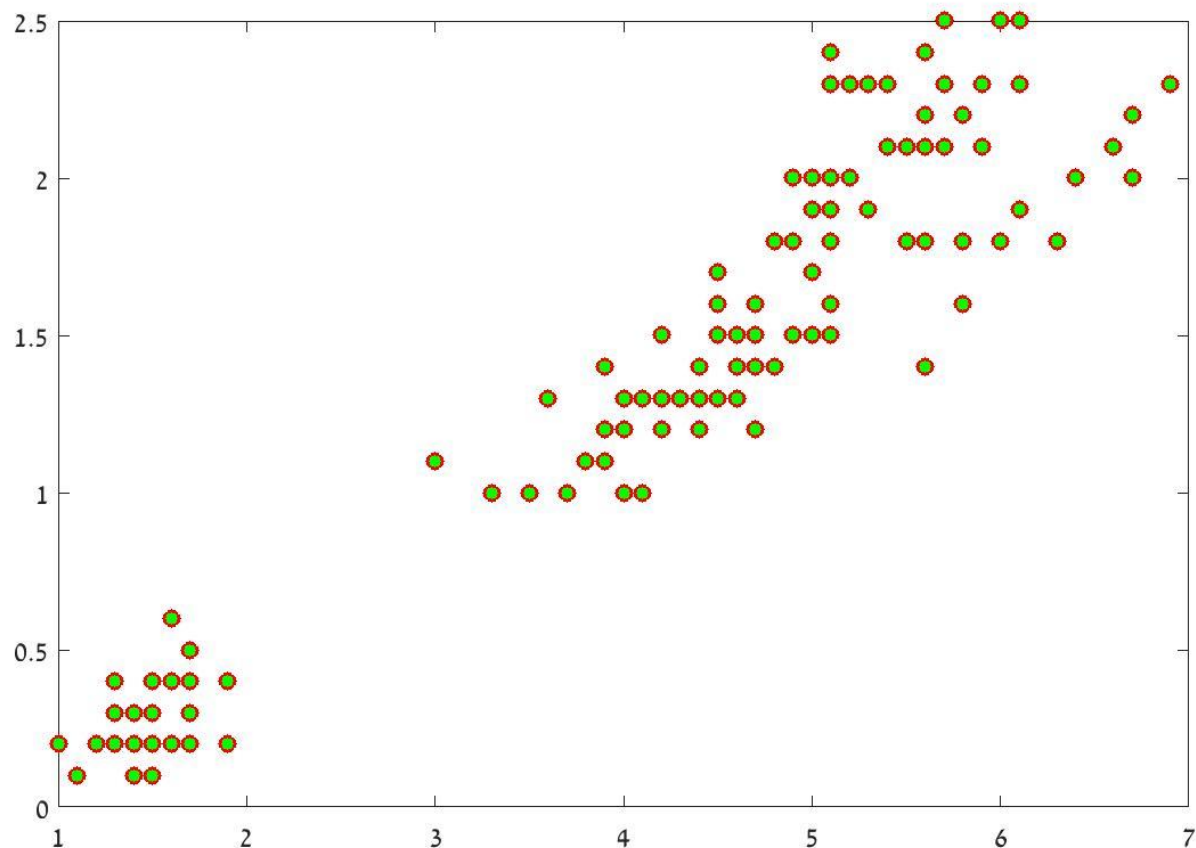
למידה לא מודרכת

- אלגוריתם ה-K-means הוא האלגוריתם הפופולרי ביותר,
- בו משתמשים באופן הנרחב ביותר.



למידה לא מודרכת

- נניח שרוצים לקבץ את הנתונים לשני **clusters** (אשכולות)



אלגוריתם ה-K-means

- מתחילים על-ידי בחירה של שתי נקודות הנקראות מרכזי הכובד (**צנטרואידים**) של האשכולות, או **clusters centroids**.
- האלגוריתם מורכב משני צעדים.
 - **Cluster assignment step** (צעד השיוך ל-clusters) לפי המרחק הקצר ביותר מאחד הצנטרואידים.
כלומר נשייך כל אחת מהנקודות לקבוצה 1 או 2 לפי המרחק הקצר ביותר של הנקודה לכל אחד מהצנטרואידים.
 - **Group centroid step** הצעד השני הוא צעד מציאת הצנטרואידים של הקבוצות החדשות שנוצרו בצעד השיוך. נזיז את הצנטרואידים לנקודות חדשות לפי הממוצע של כל הנקודות השייכות לאחת מהקבוצות.

אלגוריתם ה-K-means מקטין את השגיאה הריבועית

- אפשר להראות כי אלגוריתם ה-K-means מקטין את השגיאה הריבועית הממוצעת מהצנטרואיד הקרוב, כלומר שואף להביא למינימום את הביטוי:

$$E = \frac{1}{n} \sum_{i=1}^c \sum_{j \in G_i} \|x_j - \mu_i\|^2$$

$$\|x\|_2^2 = \sqrt{\sum_{i=1}^n x_i^2}, \quad x \in R^n \quad \text{כאשר:}$$

נראה כי כל אחד מהצעדים של האלגוריתם, כלומר סווג הדוגמאות לקבוצות לפי המרחק מהצנטרואידים, ומציאת הצנטרואידים ממזער את השגיאה ביחס למשתנים הנקבעים באותו צעד.

אלגוריתם ה-K-means מקטין את השגיאה הריבועית

- הצעד הראשון: סווג הדוגמאות לצנטרואידים: עבור הצנטרואידים הידועים, אותם נסמן ב- μ_i סווג הדוגמאות לצנטרואידים ממזער את השגיאה על-פי הגדרת הצעד, כי

$$i = \arg \min_i \|x - \mu_i\|$$

השגיאה בצעד זה היא:

$$E = \sum_{k=1}^n \left(\min_i \|x_k - \mu_i\|_2 \right)^2$$

אלגוריתם ה-K-means מקטין את השגיאה הריבועית

• בצעד השני יש למצוא $\mu_i, \quad i = 1, 2, \dots, C$

כך ש:

$$E_i = \sum_{j \in G_i} \left(\|x_j - \mu_i\|_2 \right)^2$$

יהיה מינימלי.

נמצא את המינימום של E_i עבור i כלשהו על-ידי גזירה והשוואה ל-0, ונקבל כי הצנטרואיד המביא את השגיאה למינימום הוא הממוצע של הנקודות בקבוצה:

$$\mu_i = \frac{1}{|G_i|} \sum_{j \in G_i} x_j$$

אלגוריתם ה-K-means מקטין את השגיאה הריבועית

בשני צעדי הלימוד המינימום הוא הגלובלי ביחס למשתנים הנקבעים באותו צעד, כאשר המשתנים האחרים הם קבועים. בצעד הראשון – מתוך הגדרה.

בצעד השני – לפונקציית השגיאה נקודת קיצון אחת והיא נקודת המינימום המתקבלת.

הפונקציה $(E_i = \sum_{j \in G_i} (\|x_j - \mu_i\|_2)^2)$ היא ריבועית וקמורה.

למרות זאת לפונקציה $E(G_i, \mu_i)$ מספר נקודות מינימום מקומיות, והאלגוריתם מתכנס לאחת מהן.

קביעת מספר הקבוצות K

כאשר לא ידוע מספר הקבוצות K, ומעוניינים לקבוע K נכון
ו"טבעי" תוך כדי ריצת האלגוריתם.
נגדיר את שגיאת ה-Clustering:

$$E = \frac{1}{n} \sum_{i=1}^c \sum_{j \in G_i} \|x_j - \mu_i\|^2$$

קביעת מספר הקבוצות K

מה המשמעות של שגיאת ה-clustering?

עוברים על כל הקבוצות (אינדקס i), ובכל קבוצה מחשבים את סכום המרחקים מהצנטרואיד μ_i , כלומר עוברים על כל ה- j של אותה קבוצה.

מה קורה אם מגדילים את מספר הקבוצות K?

ככל שמגדילים את K – פחות איברים בקבוצה, כאשר אם מספר הקבוצות $k=n$ כל אחת מנקודות המדידה הופכת לצנטרואיד. השגיאה E תהיה אפס, אך לא נרוויח מידע, המטרה היא לחלק לקבוצות.

שיטה אפשרית: להגדיל את K באופן הדרגתי, לחשב את $E(K)$ באופן הדרגתי, לחשב את $E(K)$ בסוף כל שלב, ולעצור כאשר:

$$1 - \frac{E(k)}{E(k-1)} < \varepsilon$$

קביעת מספר הקבוצות K

כאשר ההנחה היא שהשגיאה בצעד ה- K קטנה יותר מהשגיאה בצעד הקודם (ה- $K-1$), ולכן:

$$\frac{E(K)}{E(K-1)} < 1$$


וכן כאשר K נעשה גדול היחס ישאף ל- 1:

$$\frac{E(K)}{E(K-1)} > 1 - \varepsilon$$

Microarray Data : αμγιτ

- Microarray data are usually transformed into an **intensity matrix** (below)
- The intensity matrix allows biologists to make **correlations** between different genes (even if they are dissimilar) and to understand how genes functions might be related
- Clustering comes into play

Intensity (expression level) of gene at measured time



Time:	Time X	Time Y	Time Z
Gene 1	10	8	10
Gene 2	10	0	9
Gene 3	4	8.6	3
Gene 4	7	8	3
Gene 5	1	2	3

Clustering of Microarray Data

- Plot each datum as a point in N-dimensional space
- Make a distance matrix for the distance between every two gene points in the N-dimensional space
- Genes with a small distance share the same expression characteristics and might be functionally related or similar!
- Clustering reveal groups of functionally related genes

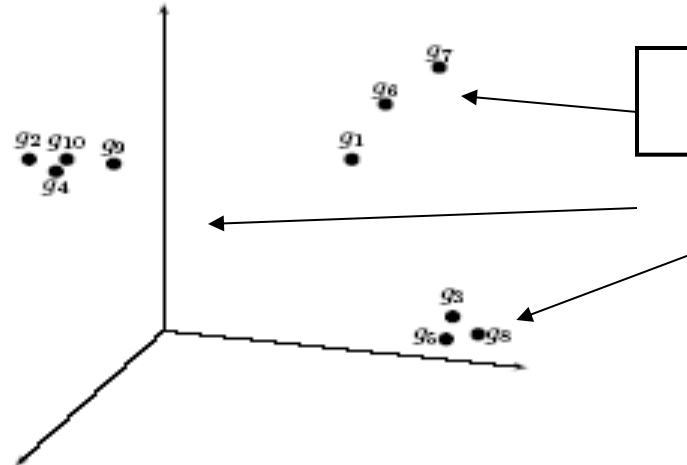
Clustering of Microarray Data (cont'd)

Time	1 hr	2 hr	3 hr
g_1	10.0	8.0	10.0
g_2	10.0	0.0	9.0
g_3	4.0	8.5	3.0
g_4	9.5	0.5	8.5
g_5	4.5	8.5	2.5
g_6	10.5	9.0	12.0
g_7	5.0	8.5	11.0
g_8	2.7	8.7	2.0
g_9	9.7	2.0	9.0
g_{10}	10.2	1.0	9.2

(a) Intensity matrix, I

	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9	g_{10}
g_1	0.0	8.1	9.2	7.7	9.3	2.3	5.1	10.2	6.1	7.0
g_2	8.1	0.0	12.0	0.9	12.0	9.5	10.1	12.8	2.0	1.0
g_3	9.2	12.0	0.0	11.2	0.7	11.1	8.1	1.1	10.5	11.5
g_4	7.7	0.9	11.2	0.0	11.2	9.2	9.5	12.0	1.6	1.1
g_5	9.3	12.0	0.7	11.2	0.0	11.2	8.5	1.0	10.6	11.6
g_6	2.3	9.5	11.1	9.2	11.2	0.0	5.6	12.1	7.7	8.5
g_7	5.1	10.1	8.1	9.5	8.5	5.6	0.0	9.1	8.3	9.3
g_8	10.2	12.8	1.1	12.0	1.0	12.1	9.1	0.0	11.4	12.4
g_9	6.1	2.0	10.5	1.6	10.6	7.7	8.3	11.4	0.0	1.1
g_{10}	7.0	1.0	11.5	1.1	11.6	8.5	9.3	12.4	1.1	0.0

(b) Distance matrix, d



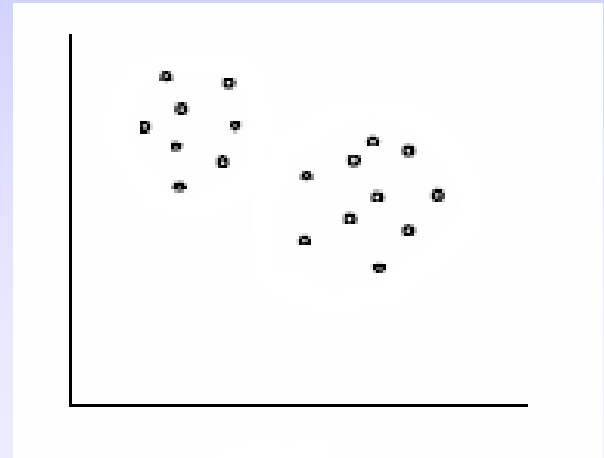
(c) Expression patterns as points in three-dimensional space.

Clusters

Homogeneity and Separation Principles

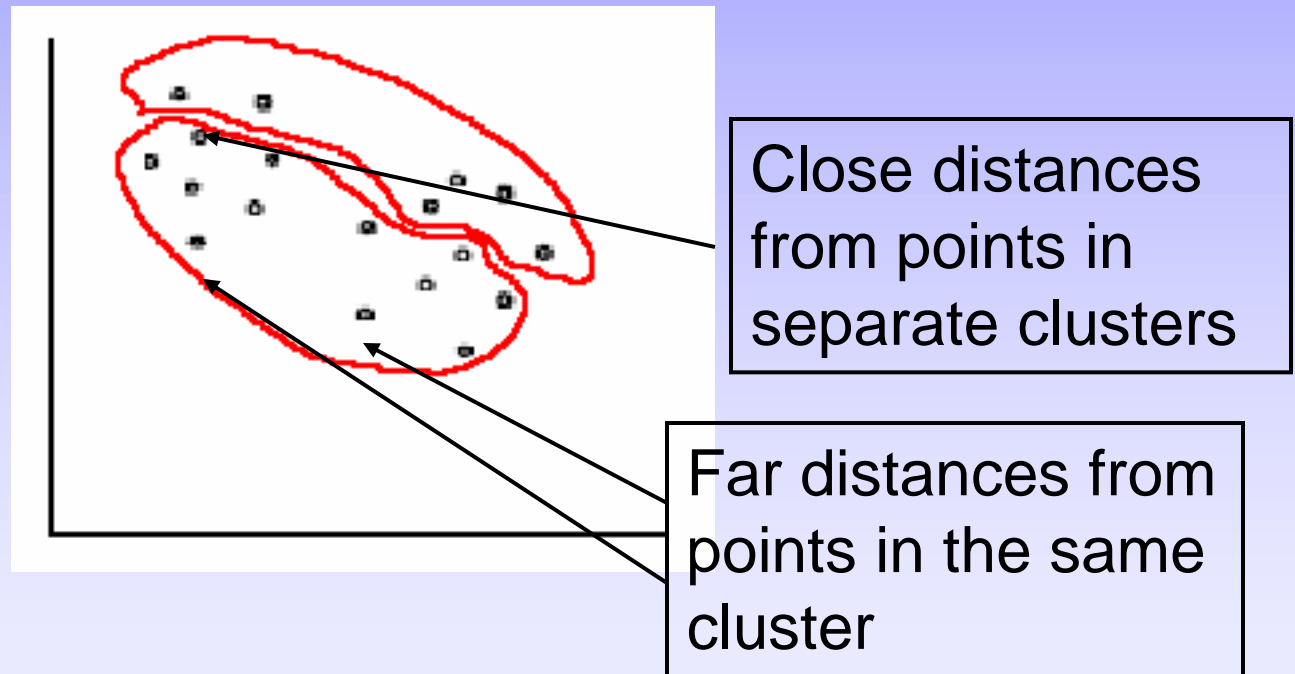
- **Homogeneity:** Elements within a cluster are close to each other
- **Separation:** Elements in different clusters are further apart from each other
- ...clustering is not an easy task!

Given these points a clustering algorithm might make two distinct clusters as follows



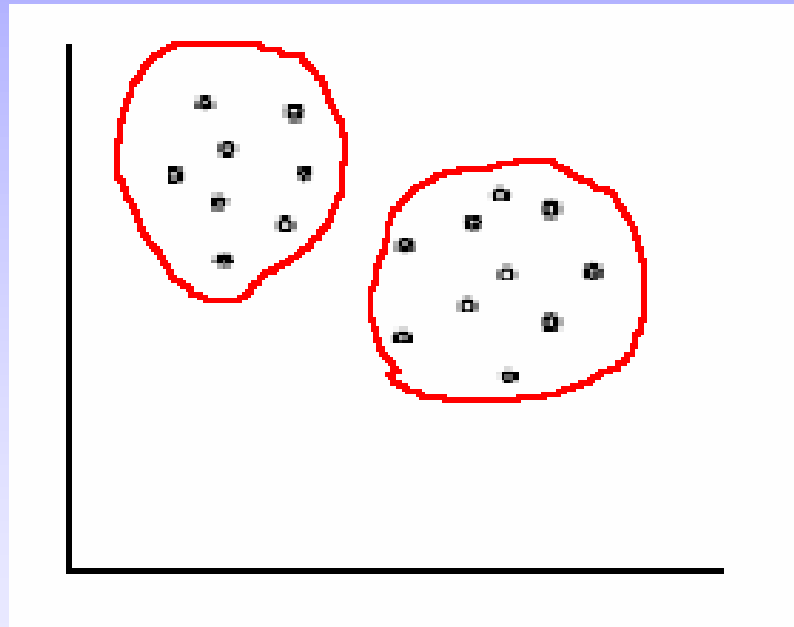
Bad Clustering

This clustering violates both Homogeneity and Separation



Good Clustering

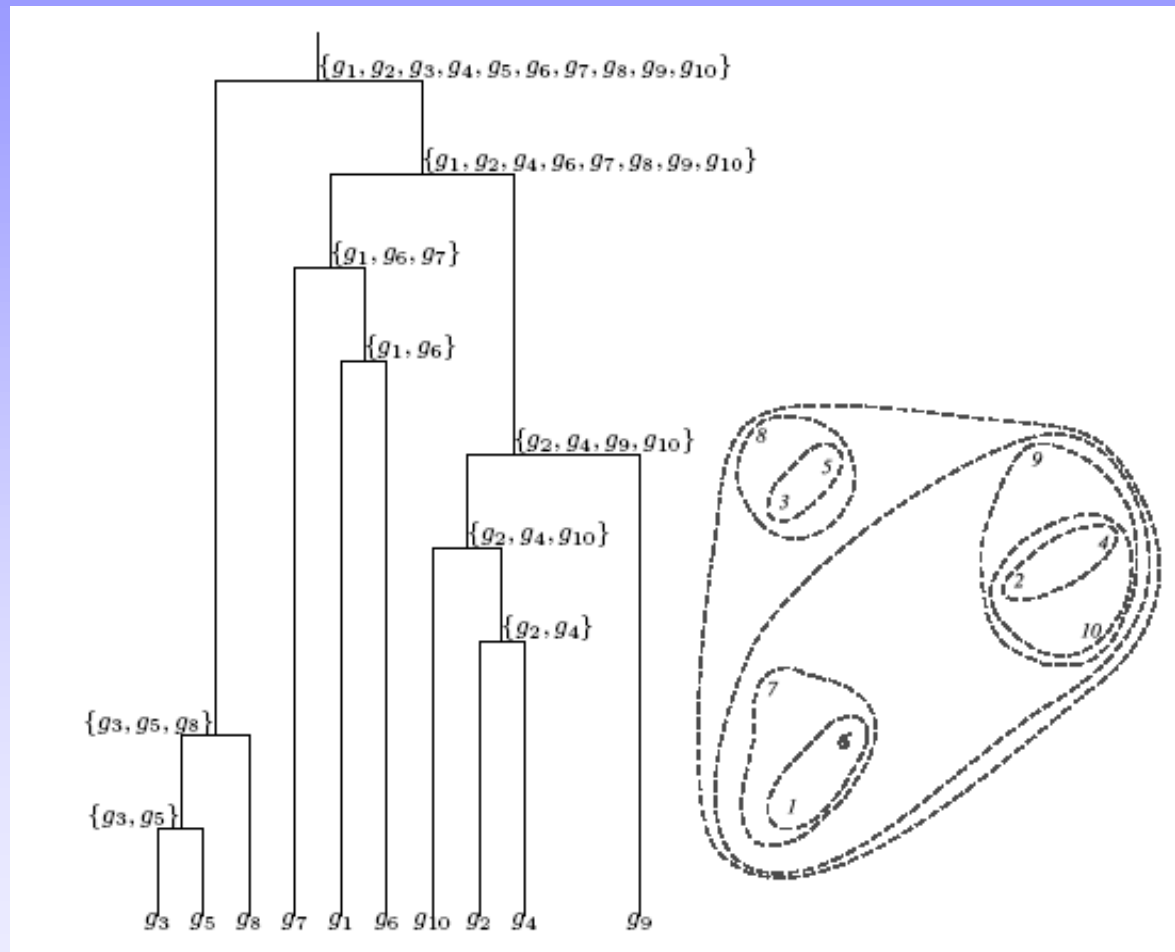
This clustering exhibits both **good** Homogeneity and Separation



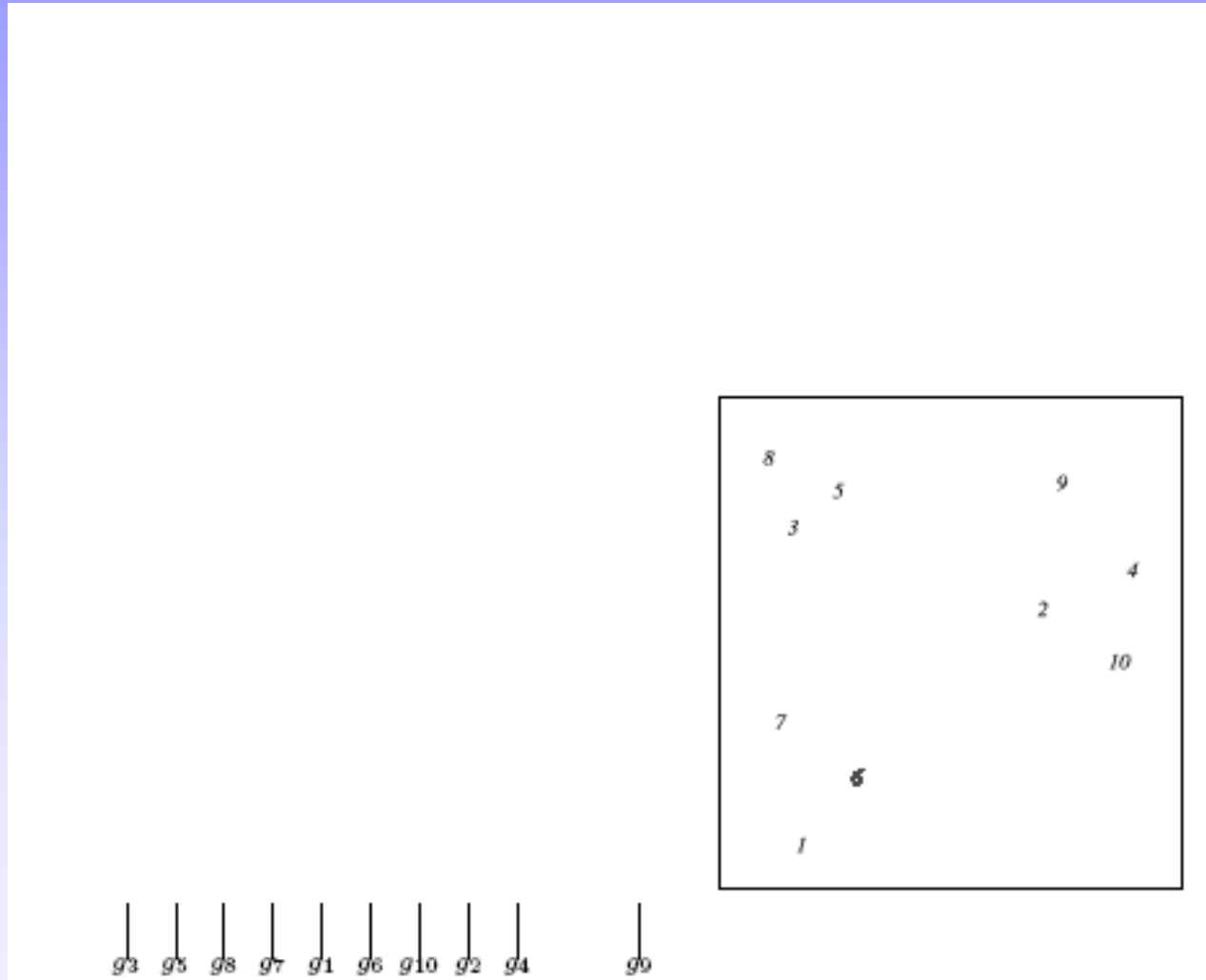
Clustering Techniques

- **Agglomerative:** Start with every element in its own cluster, and iteratively join clusters together
- **Divisive:** Start with one cluster and iteratively divide it into smaller clusters
- **Hierarchical:** Organize elements into a tree, leaves represent genes and the length of the branches represent the distances between genes. Similar genes lie within the same subtrees

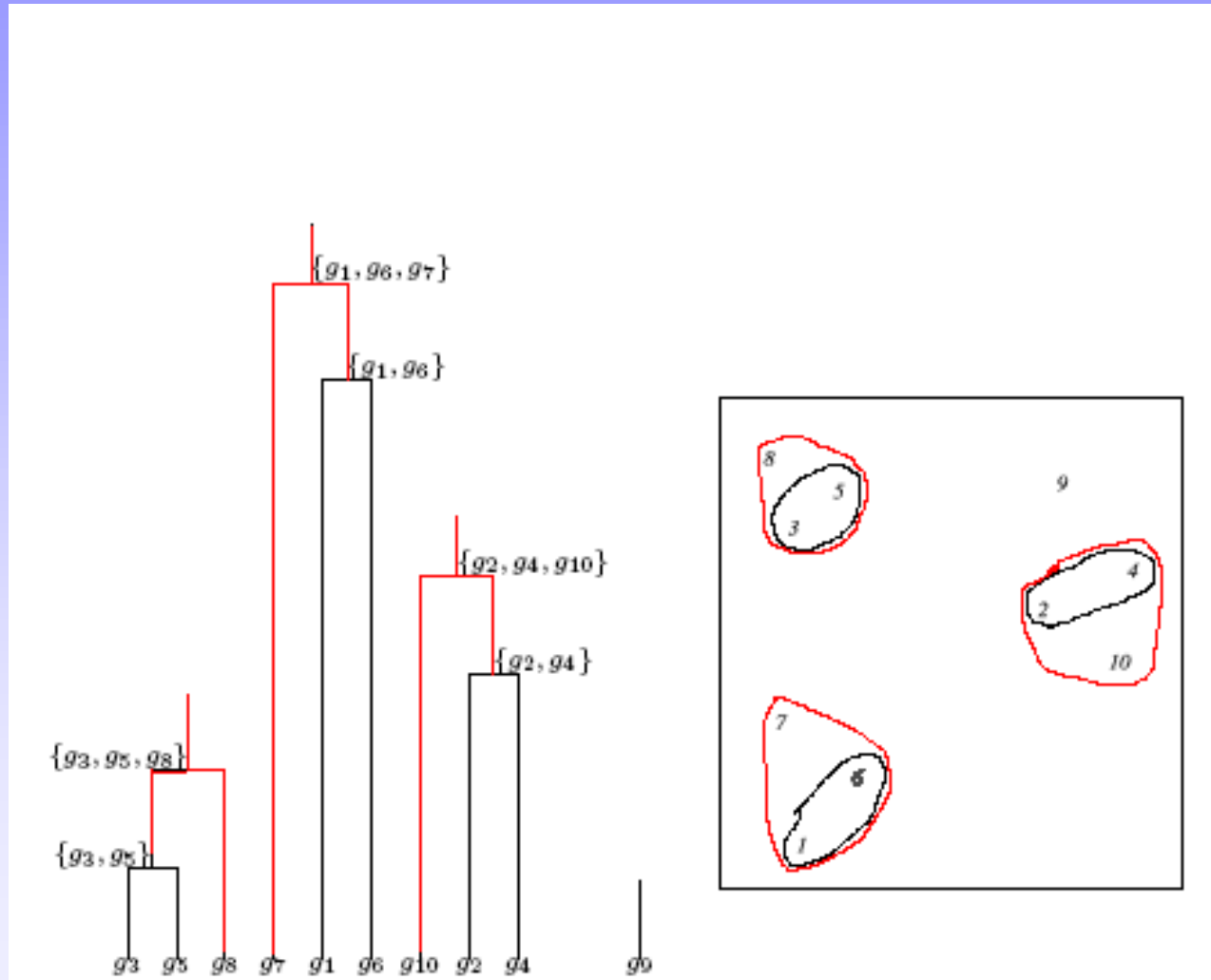
Hierarchical Clustering



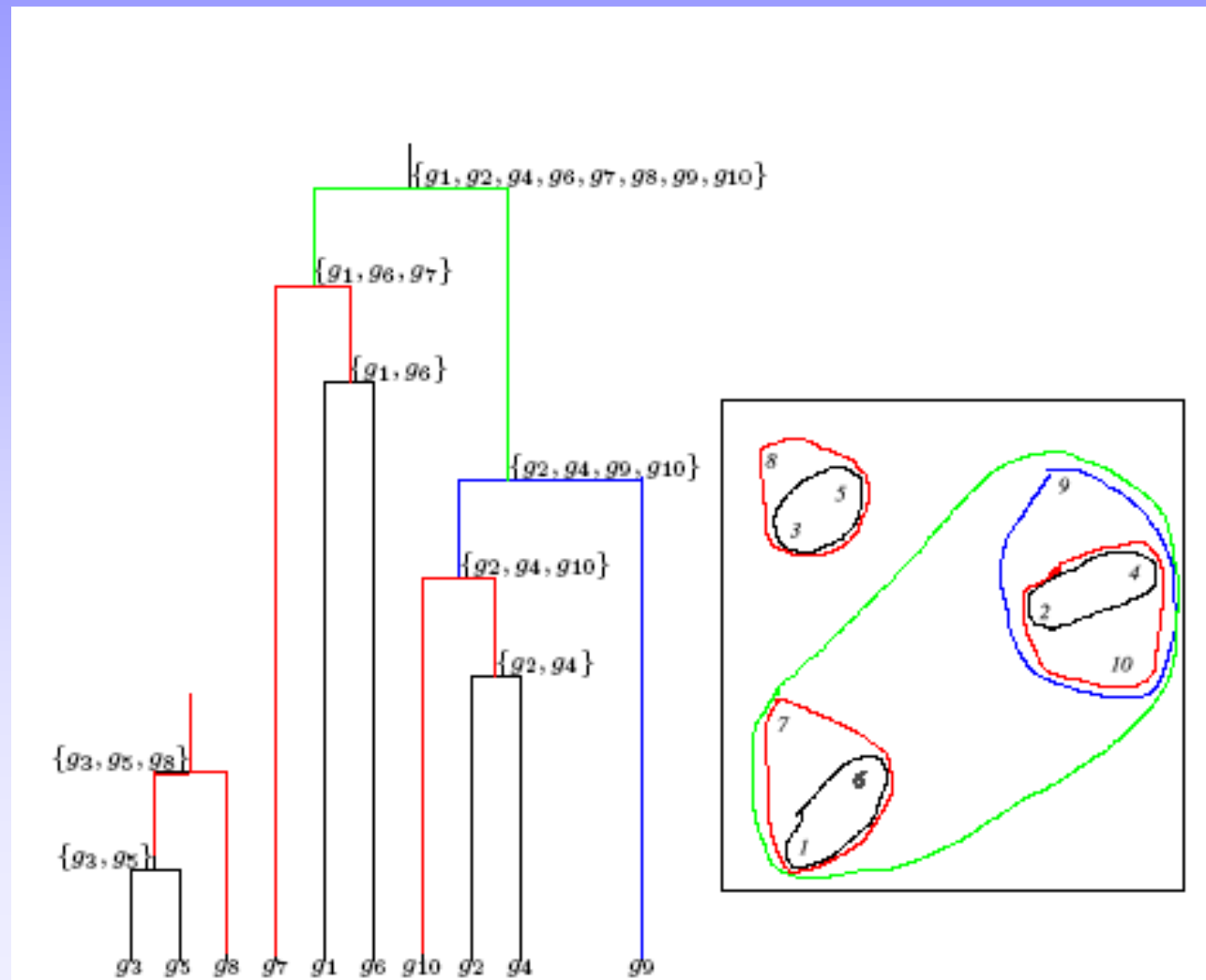
Hierarchical Clustering: Example



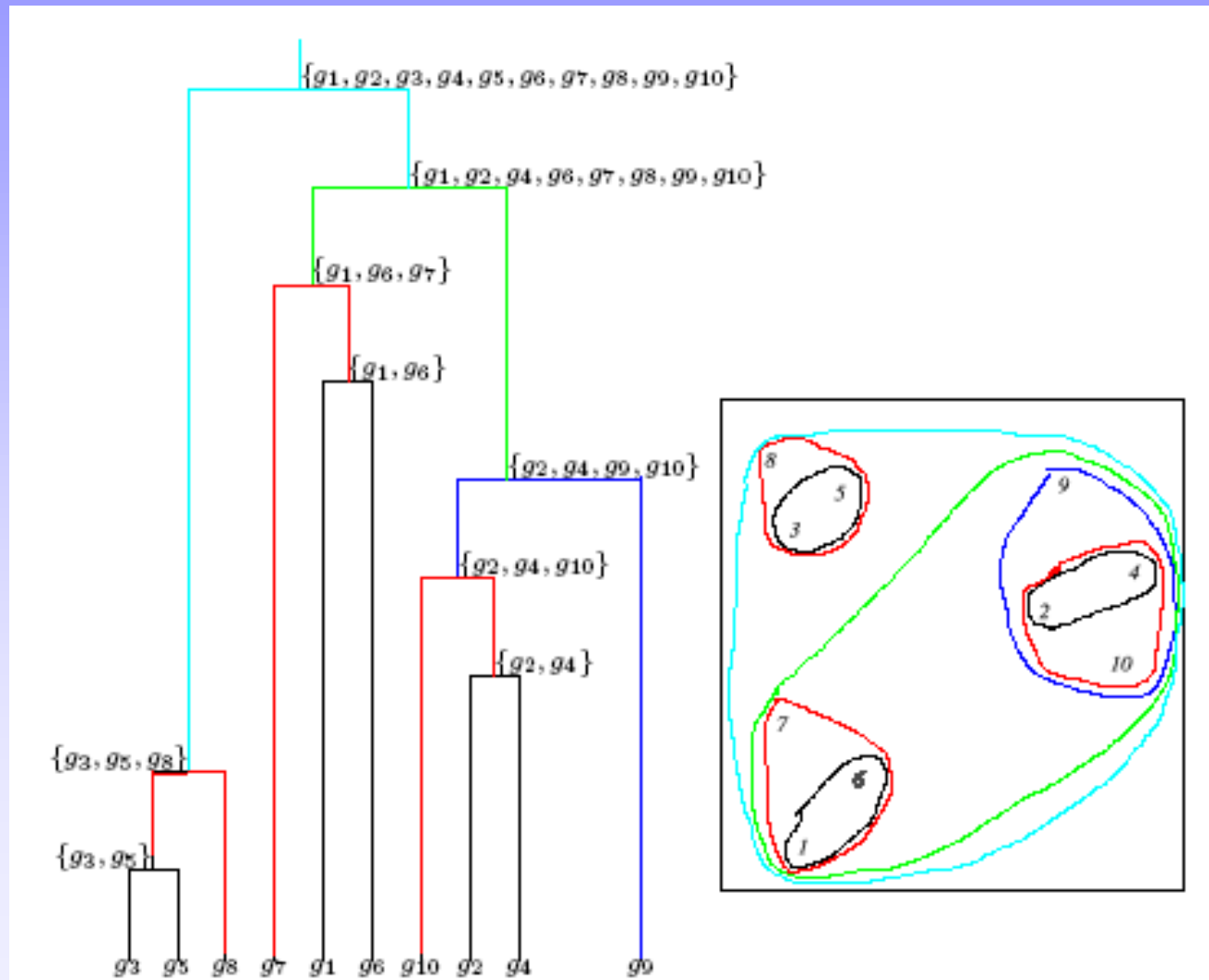
Hierarchical Clustering: Example



Hierarchical Clustering: Example

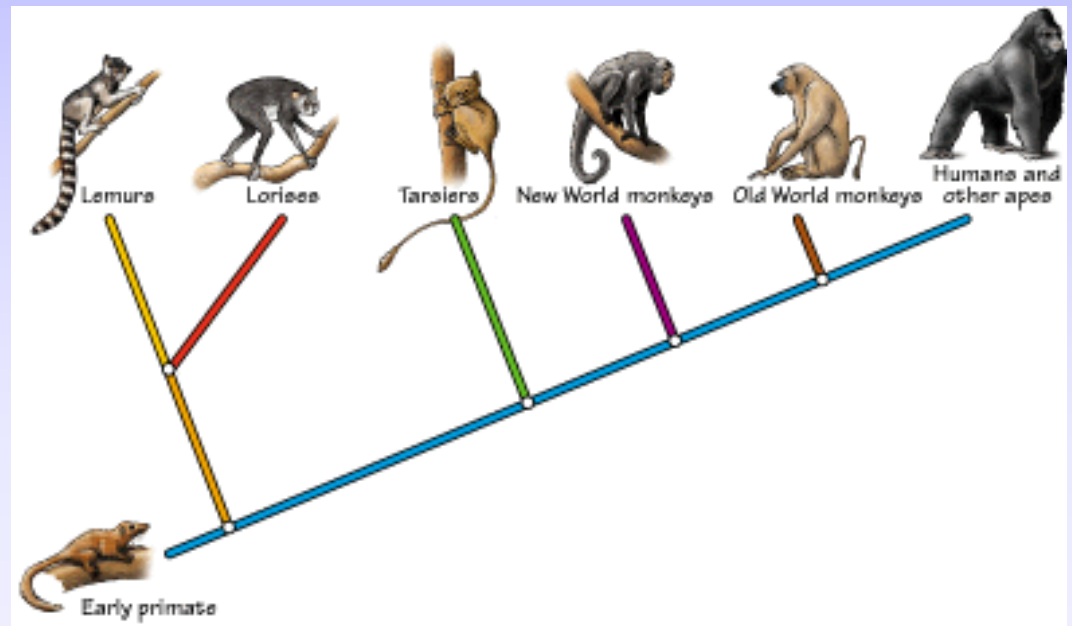


Hierarchical Clustering: Example



Hierarchical Clustering (cont'd)

- Hierarchical Clustering is often used to reveal evolutionary history
- Here is an example using the evolution of the primates



Hierarchical Clustering Algorithm

1. Hierarchical Clustering (d, n)
2. Form n clusters each with one element
3. Construct a graph T by assigning an one vertex to each cluster
4. while there is more than one cluster
5. Find the two closest clusters C_1 and C_2
6. Merge C_1 and C_2 into new cluster C with $|C_1| + |C_2|$ elements
7. Compute distance from C to all other clusters
8. Add a new vertex C to T and connect to vertices C_1 and C_2
9. Remove rows and columns of d corresponding to C_1 and C_2
10. Add a row and column for d for the new cluster C
11. return T

The algorithm takes a $n \times n$ distance matrix d of pairwise distances between points

Hierarchical Clustering: Recomputing Distances

- Different ways to define distances between points/clusters may lead to different clusterings
- $d_{\min}(C, C^*) = \min_{x \in C^*, y \in C} d(x, y)$
 - Distance between two clusters is the smallest distance between any pair of their elements
- $d_{\text{avg}}(C, C^*) = (1 / |C^*||C|) \sum_{x \in C^*, y \in C} d(x, y)$
 - distance between two clusters is the average distance between all pairs of their elements

	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9	g_{10}
g_1	0.0	8.1	9.2	7.7	9.3	2.3	5.1	10.2	6.1	7.0
g_2	8.1	0.0	12.0	0.9	12.0	9.5	10.1	12.8	2.0	1.0
g_3	9.2	12.0	0.0	11.2	0.7	11.1	8.1	1.1	10.5	11.5
g_4	7.7	0.9	11.2	0.0	11.2	9.2	9.5	12.0	1.6	1.1
g_5	9.3	12.0	0.7	11.2	0.0	11.2	8.5	1.0	10.6	11.6
g_6	2.3	9.5	11.1	9.2	11.2	0.0	5.6	12.1	7.7	8.5
g_7	5.1	10.1	8.1	9.5	8.5	5.6	0.0	9.1	8.3	9.3
g_8	10.2	12.8	1.1	12.0	1.0	12.1	9.1	0.0	11.4	12.4
g_9	6.4	2.0	10.5	1.6	10.6	7.7	8.3	11.4	0.0	1.1
g_{10}	7.0	1.0	11.5	1.1	11.6	8.5	9.3	12.4	1.1	0.0

g_3 g_5 g_8 g_7 g_1 g_6 g_{10} g_2 g_4 g_9

	g_1	g_2
g_1	0.0	8.1
g_2	8.1	0.0
g_4	7.7	0.9
g_6	2.3	9.5
g_7	5.1	10.1
g_8	10.2	12.8
g_9	6.4	2.0
g_{10}	7.0	1.0

g_4
7.7
0.9
0.0
9.2
9.5
12.0
1.6
1.1

g_6	g_7	g_8	g_9	g_{10}
2.3	5.1	10.2	6.1	7.0
9.5	10.1	12.8	2.0	1.0
9.2	9.5	12.0	1.6	1.1
0.0	5.6	12.1	7.7	8.5
5.6	0.0	9.1	8.3	9.3
12.1	9.1	0.0	11.4	12.4
7.7	8.3	11.4	0.0	1.1
8.5	9.3	12.4	1.1	0.0

|

g_3

|

g_5

|

g_8

|

g_7

|

g_1

|

g_6

|

g_{10}

|

g_2

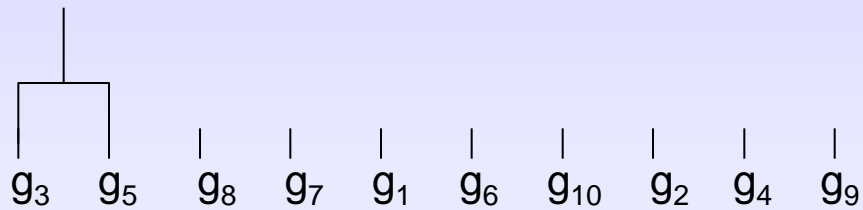
|

g_4

|

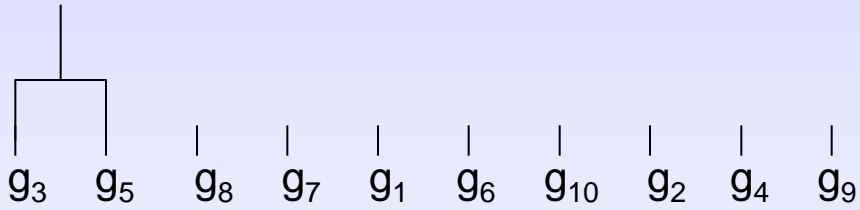
g_9

	g_1	g_2	g_4	g_6	g_7	g_8	g_9	g_{10}	$c_i(g_3g_5)$
g_1	0.0	8.1	7.7	2.3	5.1	10.2	6.1	7.0	9.2
g_2	8.1	0.0	0.9	9.5	10.1	12.8	2.0	1.0	12.0
g_4	7.7	0.9	0.0	9.2	9.5	12.0	1.6	1.1	11.2
g_6	2.3	9.5	9.2	0.0	5.6	12.1	7.7	8.5	11.1
g_7	5.1	10.1	9.5	5.6	0.0	9.1	8.3	9.3	8.1
g_8	10.2	12.8	12.0	12.1	9.1	0.0	11.4	12.4	1.0
g_9	6.4	2.0	1.6	7.7	8.3	11.4	0.0	1.1	10.5
g_{10}	7.0	1.0	1.1	8.5	9.3	12.4	1.1	0.0	11.5
$c_i(g_3g_5)$	9.2	12.0	11.2	11.1	8.1	1.0	10.5	11.5	0.0

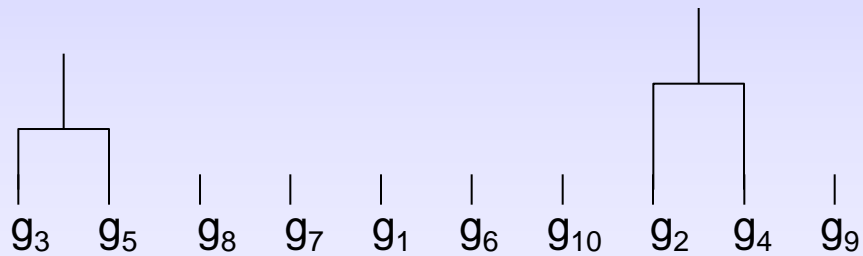


	g_1
g_1	0.0
g_6	2.3
g_7	5.1
g_8	10.2
g_9	6.4
g_{10}	7.0
$c_1(g_3g_5)$	9.2

g_6	g_7	g_8	g_9	g_{10}	$c_1(g_3g_5)$
2.3	5.1	10.2	6.1	7.0	9.2
0.0	5.6	12.1	7.7	8.5	11.1
5.6	0.0	9.1	8.3	9.3	8.1
12.1	9.1	0.0	11.4	12.4	1.0
7.7	8.3	11.4	0.0	1.1	10.5
8.5	9.3	12.4	1.1	0.0	11.5
11.1	8.1	1.0	10.5	11.5	0.0

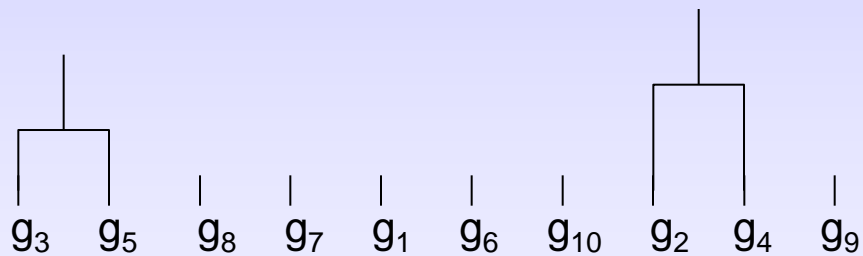


	g_1	g_6	g_7	g_8	g_9	g_{10}	$c_1(g_3g_5)$	$c_2(g_2g_4)$
g_1	0.0	2.3	5.1	10.2	6.1	7.0	9.2	7.7
g_6	2.3	0.0	5.6	12.1	7.7	8.5	11.1	9.2
g_7	5.1	5.6	0.0	9.1	8.3	9.3	8.1	9.5
g_8	10.2	12.1	9.1	0.0	11.4	12.4	1.0	12.0
g_9	6.4	7.7	8.3	11.4	0.0	1.1	10.5	1.6
g_{10}	7.0	8.5	9.3	12.4	1.1	0.0	11.5	1.0
$c_1(g_3g_5)$	9.2	11.1	8.1	1.0	10.5	11.5	0.0	11.2
$c_2(g_2g_4)$	7.7	9.2	9.5	12.0	1.6	1.0	11.2	0.0

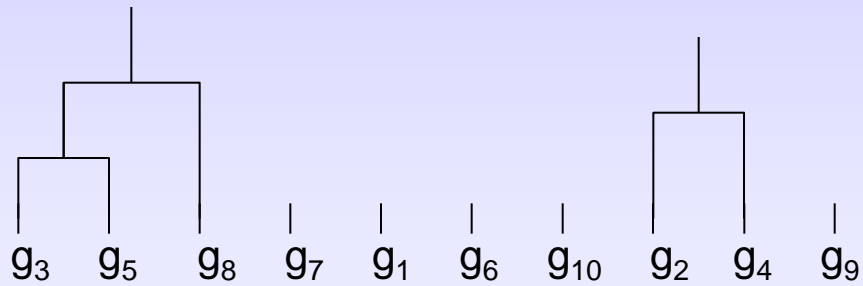


	g_1	g_6	g_7
g_1	0.0	2.3	5.1
g_6	2.3	0.0	5.6
g_7	5.1	5.6	0.0
g_9	6.4	7.7	8.3
g_{10}	7.0	8.5	9.3
$c_2(g_2g_4)$	7.7	9.2	9.5

g_9	g_{10}	$c_2(g_2g_4)$
6.1	7.0	7.7
7.7	8.5	9.2
8.3	9.3	9.5
0.0	1.1	1.6
1.1	0.0	1.0
1.6	1.0	0.0

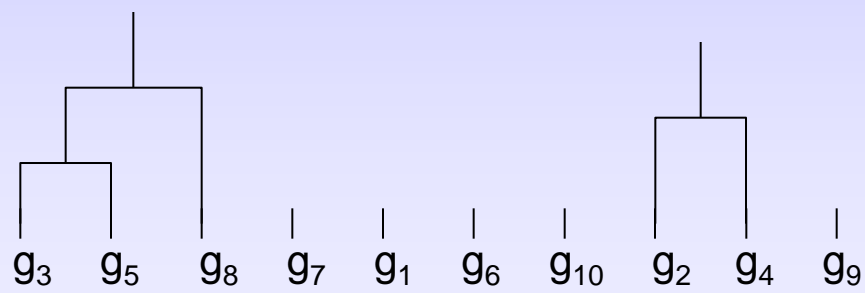


	g_1	g_6	g_7	g_9	g_{10}	$c_2(g_2g_4)$	$c_3(g_3g_5g_8)$
g_1	0.0	2.3	5.1	6.1	7.0	7.7	9.2
g_6	2.3	0.0	5.6	7.7	8.5	9.2	11.1
g_7	5.1	5.6	0.0	8.3	9.3	9.5	8.1
g_9	6.4	7.7	8.3	0.0	1.1	1.6	10.5
g_{10}	7.0	8.5	9.3	1.1	0.0	1.0	11.5
$c_2(g_2g_4)$	7.7	9.2	9.5	1.6	1.0	0.0	11.2
$c_3(g_3g_5g_8)$	9.2	11.1	8.1	10.5	11.5	11.2	0.0

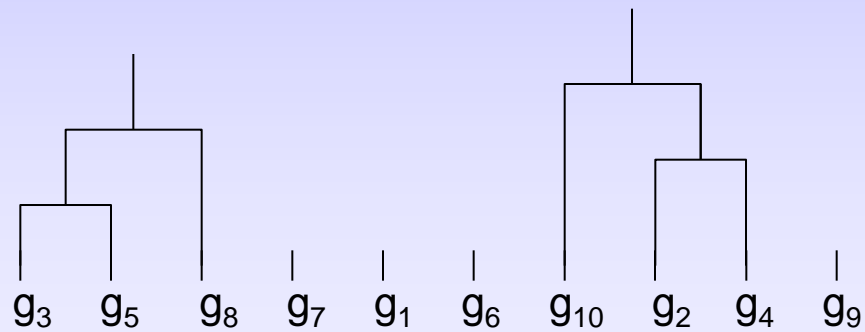


	g_1	g_6	g_7	g_9
g_1	0.0	2.3	5.1	6.1
g_6	2.3	0.0	5.6	7.7
g_7	5.1	5.6	0.0	8.3
g_9	6.4	7.7	8.3	0.0
$c_3(g_3g_5g_8)$	9.2	11.1	8.1	10.5

$c_3(g_3g_5g_8)$
9.2
11.1
8.1
10.5
0.0

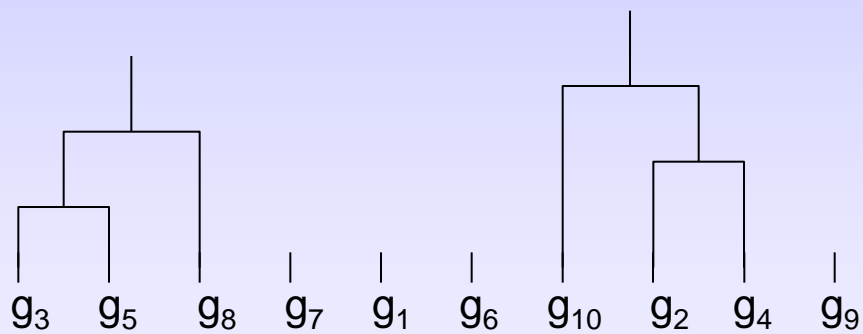


	g_1	g_6	g_7	g_9	$c_3(g_3g_5g_8)$	$c_4(g_{10}c_2)$
g_1	0.0	2.3	5.1	6.1	9.2	7.0
g_6	2.3	0.0	5.6	7.7	11.1	8.5
g_7	5.1	5.6	0.0	8.3	8.1	9.3
g_9	6.4	7.7	8.3	0.0	10.5	1.1
$c_3(g_3g_5g_8)$	9.2	11.1	8.1	10.5	0.0	11.2
$c_4(g_{10}c_2)$	7.0	8.5	9.3	1.1	11.2	0.0

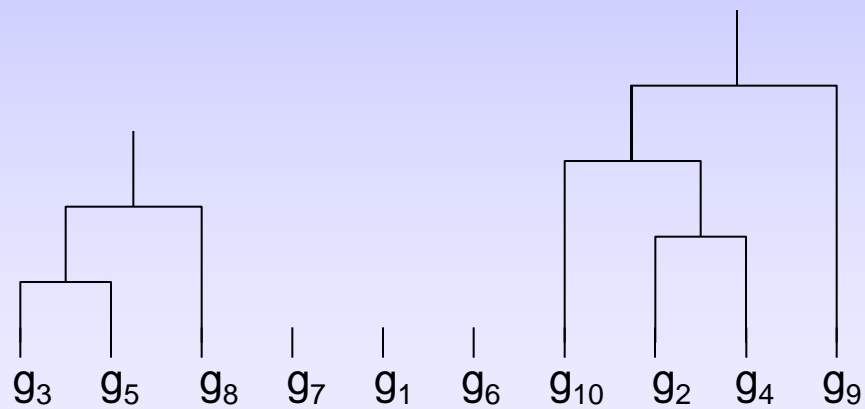


	g_1	g_6	g_7
g_1	0.0	2.3	5.1
g_6	2.3	0.0	5.6
g_7	5.1	5.6	0.0
$c_3(g_3g_5g_8)$	9.2	11.1	8.1

$c_3(g_3g_5g_8)$
9.2
11.1
8.1
0.0

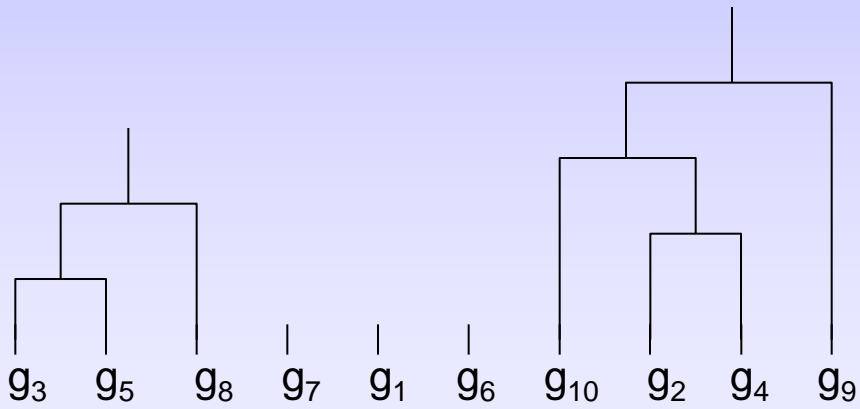


	g_1	g_6	g_7	$c_3(g_3g_5g_8)$	$c_5(g_9c_4)$
g_1	0.0	2.3	5.1	9.2	6.1
g_6	2.3	0.0	5.6	11.1	7.7
g_7	5.1	5.6	0.0	8.1	8.3
$c_3(g_3g_5g_8)$	9.2	11.1	8.1	0.0	10.5
$c_5(g_9c_4)$	6.1	7.7	8.3	10.5	0.0

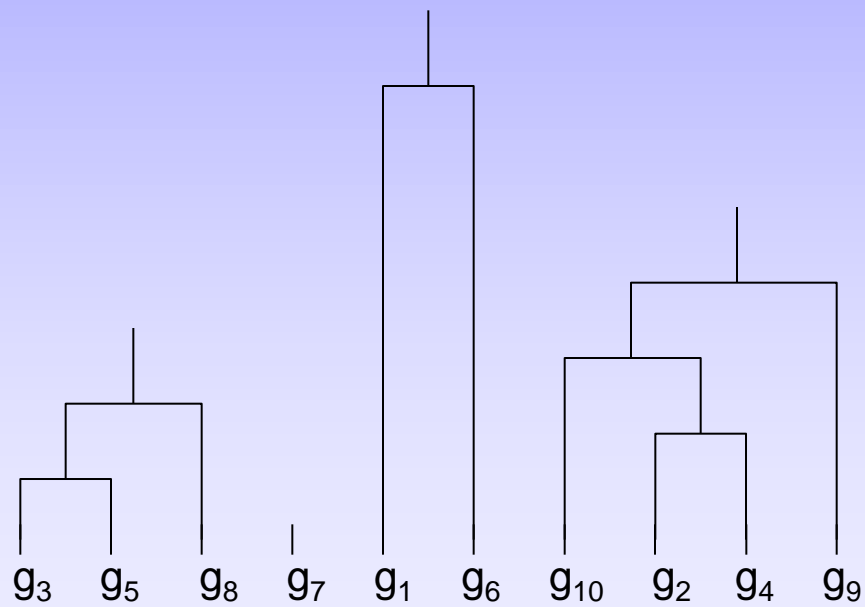


g_7
$c_3(g_3g_5g_8)$
$c_5(g_9c_4)$

g_7	$c_3(g_3g_5g_8)$	$c_5(g_9c_4)$
0.0	8.1	8.3
8.1	0.0	10.5
8.3	10.5	0.0

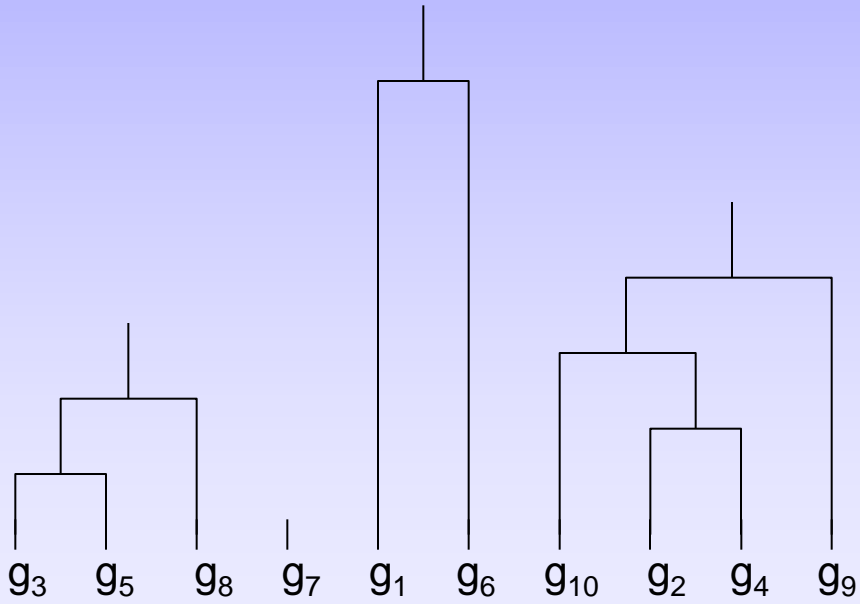


	g_7	$c_3(g_3g_5g_8)$	$c_5(g_9c_4)$	$c_6(g_1g_6)$
g_7	0.0	8.1	8.3	5.1
$c_3(g_3g_5g_8)$	8.1	0.0	10.5	9.2
$c_5(g_9c_4)$	8.3	10.5	0.0	6.1
$c_6(g_1g_6)$	5.1	9.2	6.1	0.0

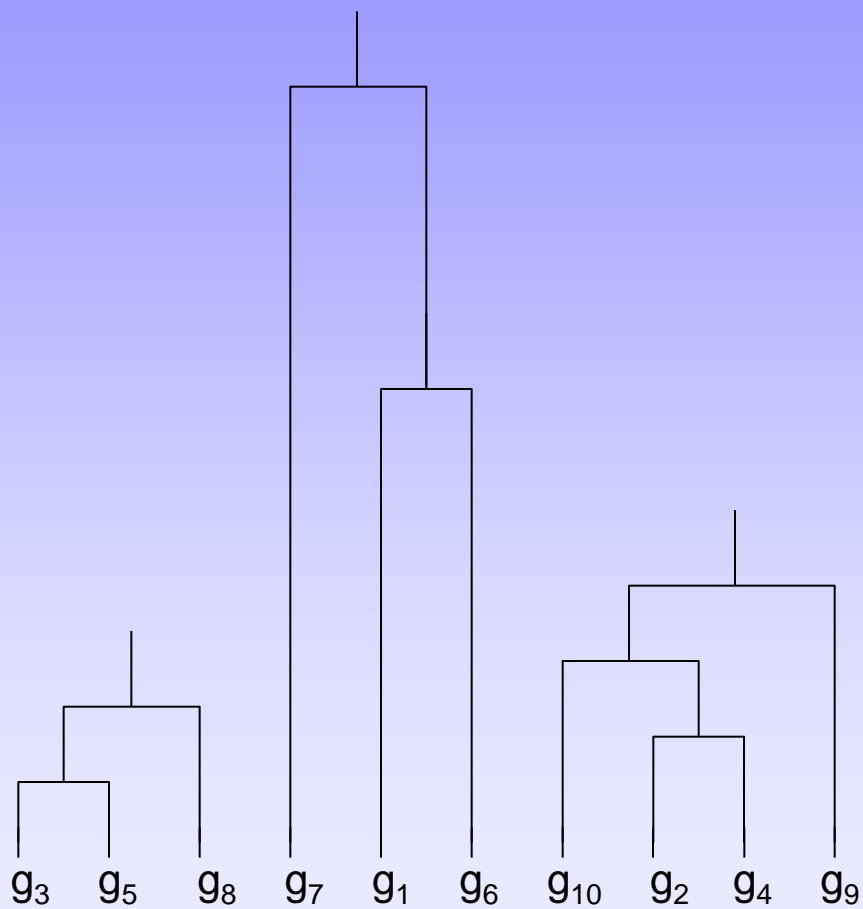


$c_3(g_3g_5g_8)$
$c_5(g_9c_4)$

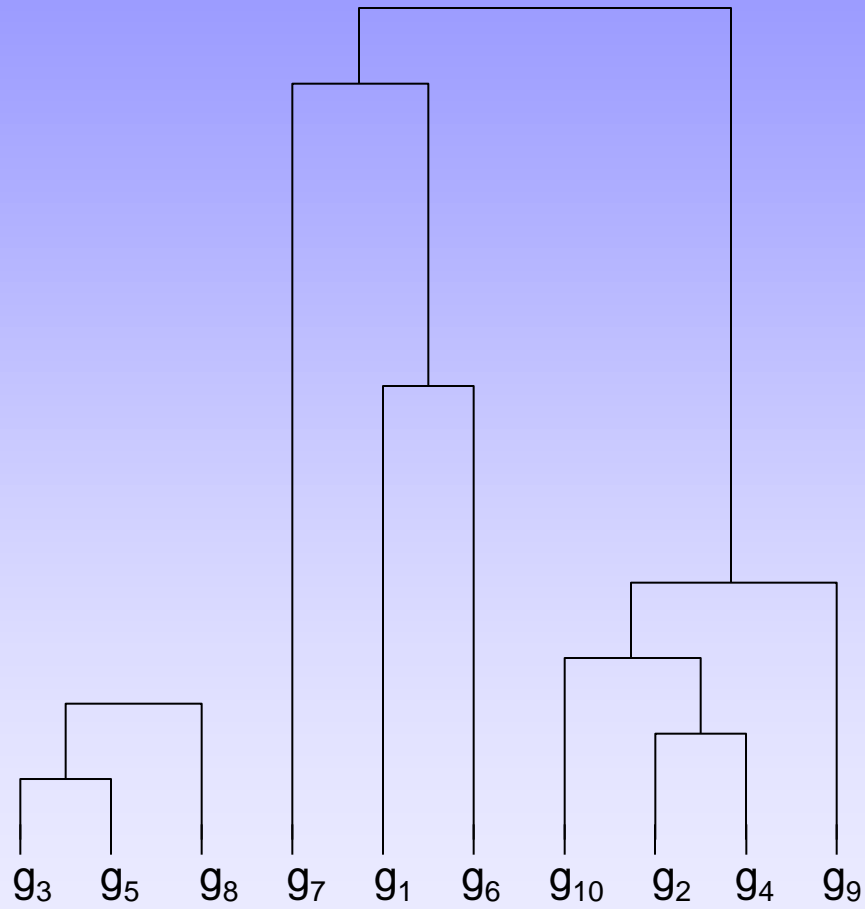
$c_3(g_3g_5g_8)$	$c_5(g_9c_4)$
0.0	10.5
10.5	0.0



	$c_3(g_3g_5g_8)$	$c_5(g_9c_4)$	$c_7(g_7c_6)$
$c_3(g_3g_5g_8)$	0.0	10.5	8.1
$c_5(g_9c_4)$	10.5	0.0	6.1
$c_7(g_7c_6)$	8.1	6.1	0.0

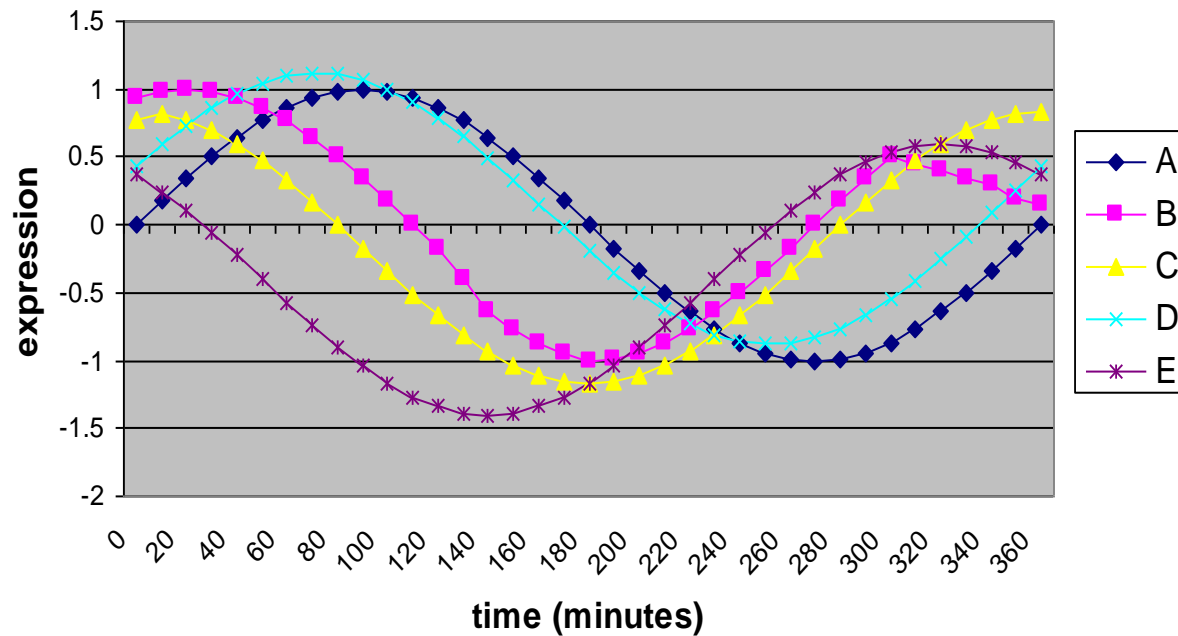


	$c_3(g_3g_5g_8)$	$c_8(c_5c_7)$
$c_3(g_3g_5g_8)$	0.0	8.1
$c_8(c_5c_7)$	8.1	0.0



Clustering: Example 1, Step 1

Algorithm: Hierarchical, Distance Metric: Correlation

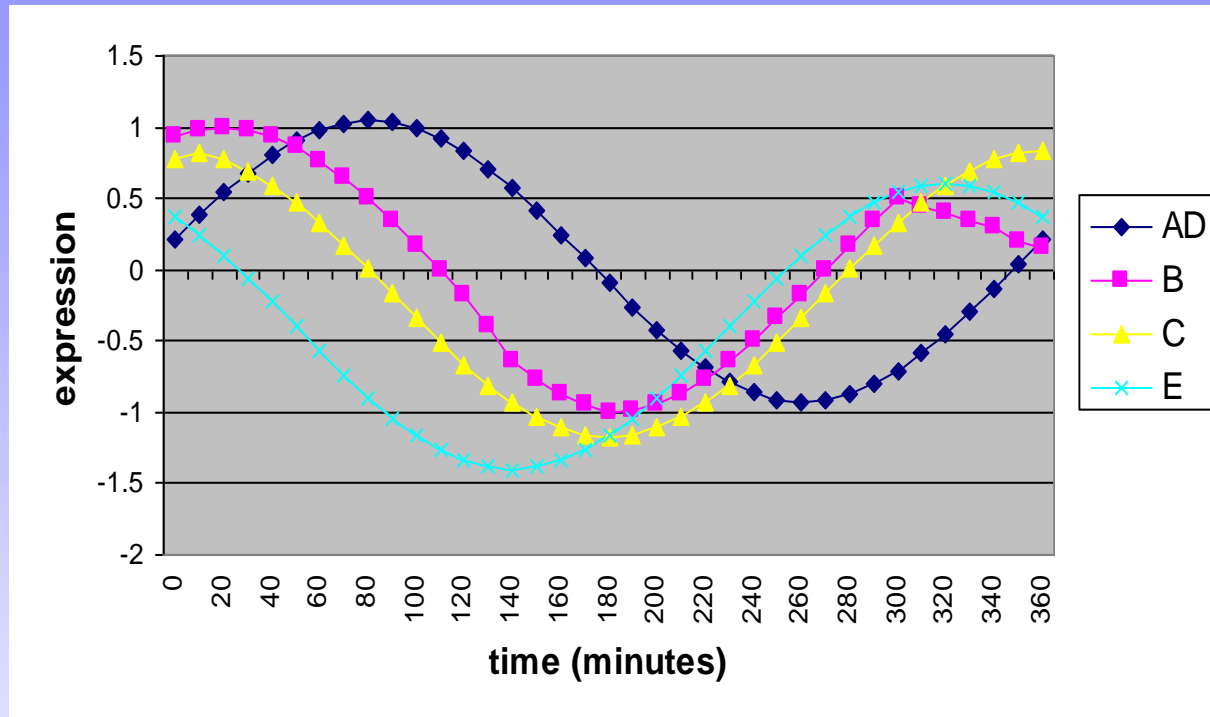


	A	B	C	D	E
A	-	0.23	0.00	0.95	-0.63
B	-	-	0.91	0.56	0.56
C	-	-	-	0.32	0.77
D	-	-	-	-	-0.36
E	-	-	-	-	-

[A
D

Clustering: Example 1, Step 2

Algorithm: Hierarchical, Distance Metric: Correlation



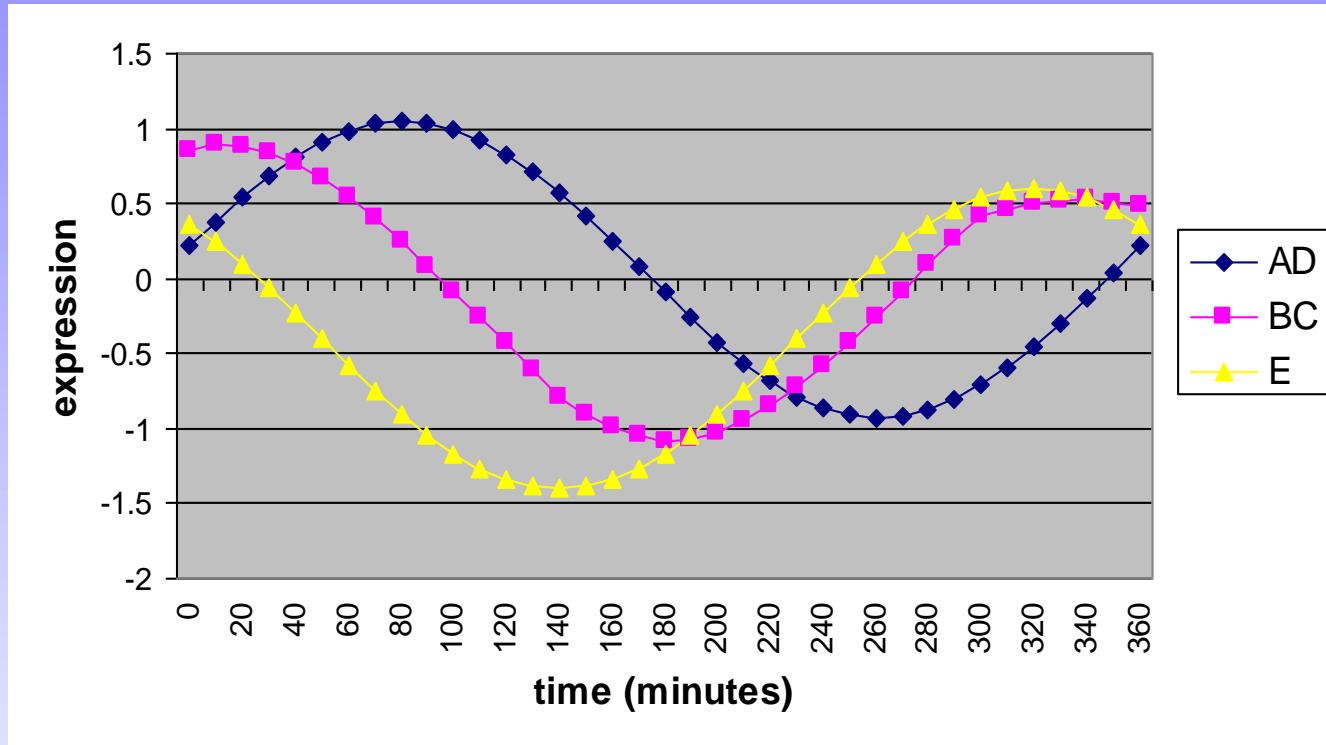
	AD	B	C	E
AD	-	0.37	0.16	-0.52
B	-	-	0.91	0.56
C	-	-	-	0.77
E	-	-	-	-

[A
D

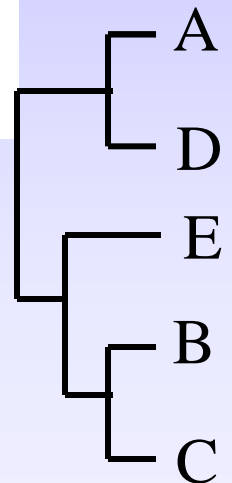
[B
C

Clustering: Example 1, Step 3

Algorithm: Hierarchical, Distance Metric: Correlation

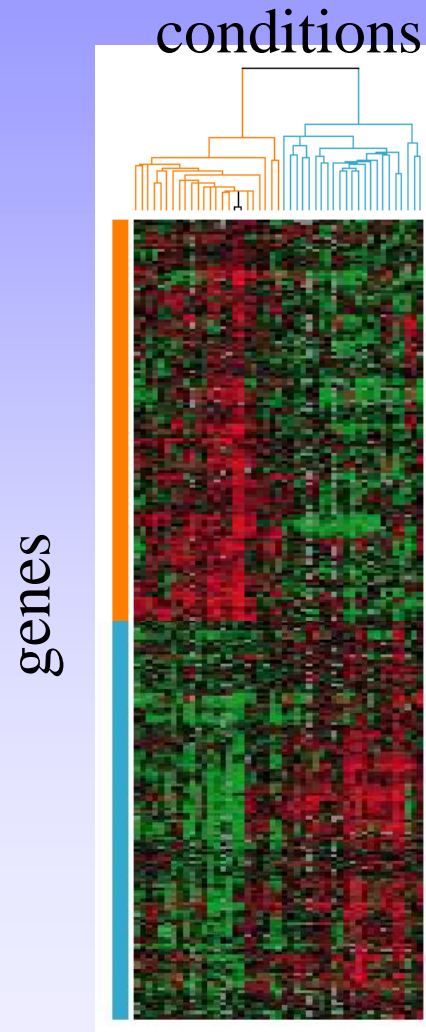


	AD	BC	E
AD	-	0.27	-0.52
BC	-	-	0.68
E	-	-	-



Tree View

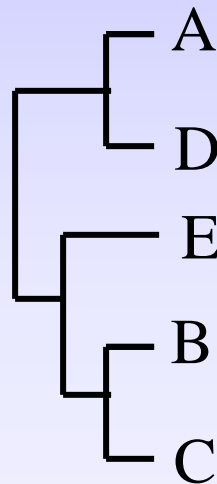
Eisen et al. (1998) PNAS 95: 14863-14868



Hierarchical Clustering Summary

Advantages

- Easy
- Very Visual
- Flexible (mean, median, etc.)



Disadvantages

- Unrelated Genes Are Eventually Joined
- Hard To Define Clusters
- Manual Interpretation Often Required

K-Means Clustering

- Configure **K** clusters to enclose all the data points, which minimizes the mean squared distance from each data point to its cluster center, or $d(\mathbf{V}, \mathbf{X})$:

Squared Distortion Error

$$d(\mathbf{V}, \mathbf{X}) = \sum d(v_i, \mathbf{X})^2 / n \quad 1 \leq i \leq n$$

- Note $d(v_i, \mathbf{X})$ refers to Euclidean Distance between the data point v_i and the center of gravity of the corresponding cluster

K-Means Clustering Problem: Formulation

- **Input:** A set, V , consisting of n points and a parameter k
- **Output:** A set X consisting of k points (cluster centers) that minimizes $d(V, X)$ over all possible choices of X

This problem is NP-complete.

An efficient heuristic method for K-Means clustering is the Lloyd algorithm

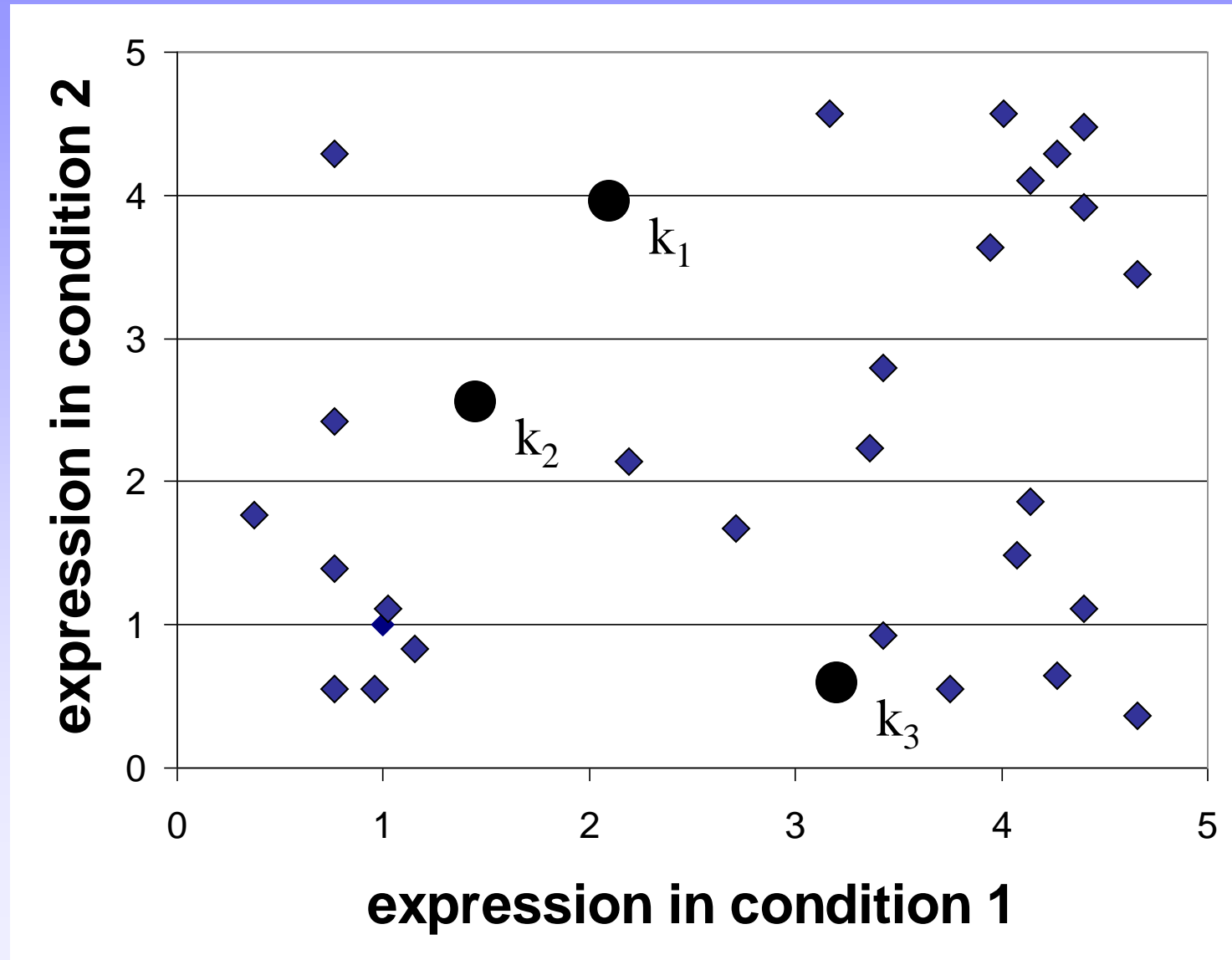
K-Means Clustering: Lloyd Algorithm

1. Lloyd Algorithm
2. Arbitrarily assign the k cluster centers
3. **while** the cluster centers keep changing
4. Assign each data point to the cluster C_i corresponding to the closest representative (center) x_i ($1 \leq i \leq k$)
5. After the assignment of all n data points, compute new cluster representatives according to the center of gravity of each existing cluster, that is, the new cluster representative is $\frac{\sum v \in C}{|C|}$ for all v in C for every cluster C

*This may lead to merely a locally optimal clustering.

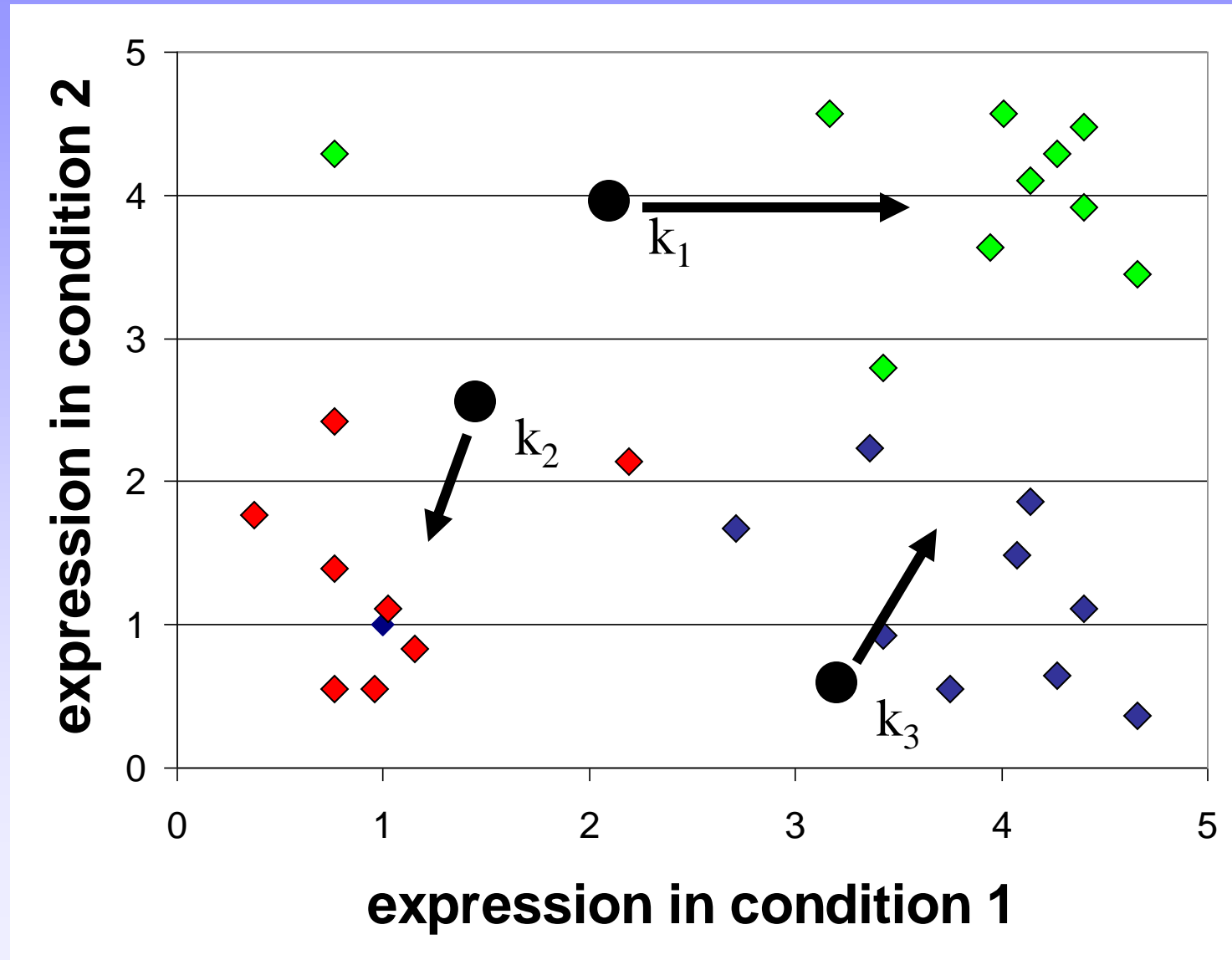
Clustering: Example 2, Step 1

Algorithm: k-means, Distance Metric: **Euclidean Distance**



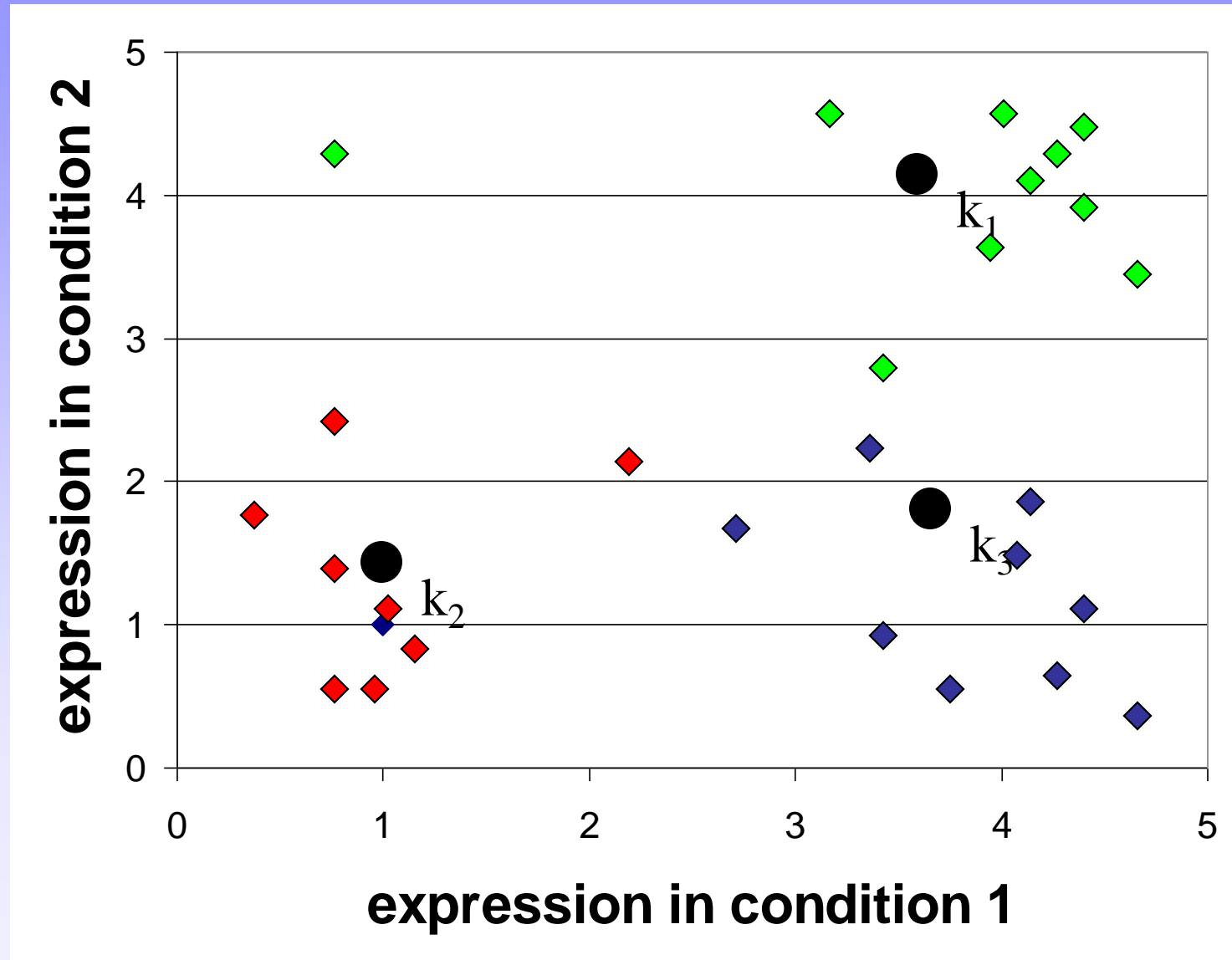
Clustering: Example 2, Step 2

Algorithm: k-means, Distance Metric: Euclidean Distance



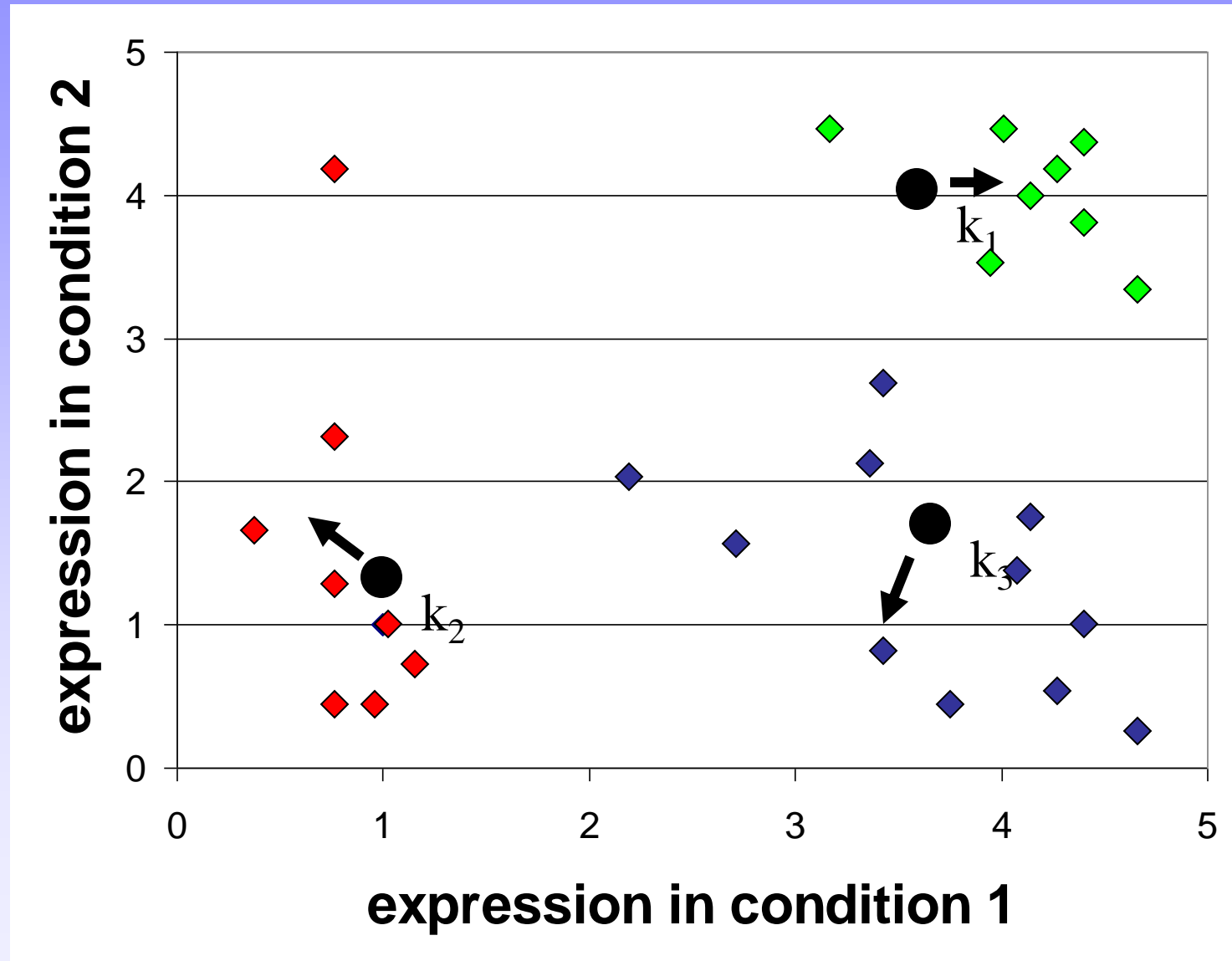
Clustering: Example 2, Step 3

Algorithm: k-means, Distance Metric: Euclidean Distance



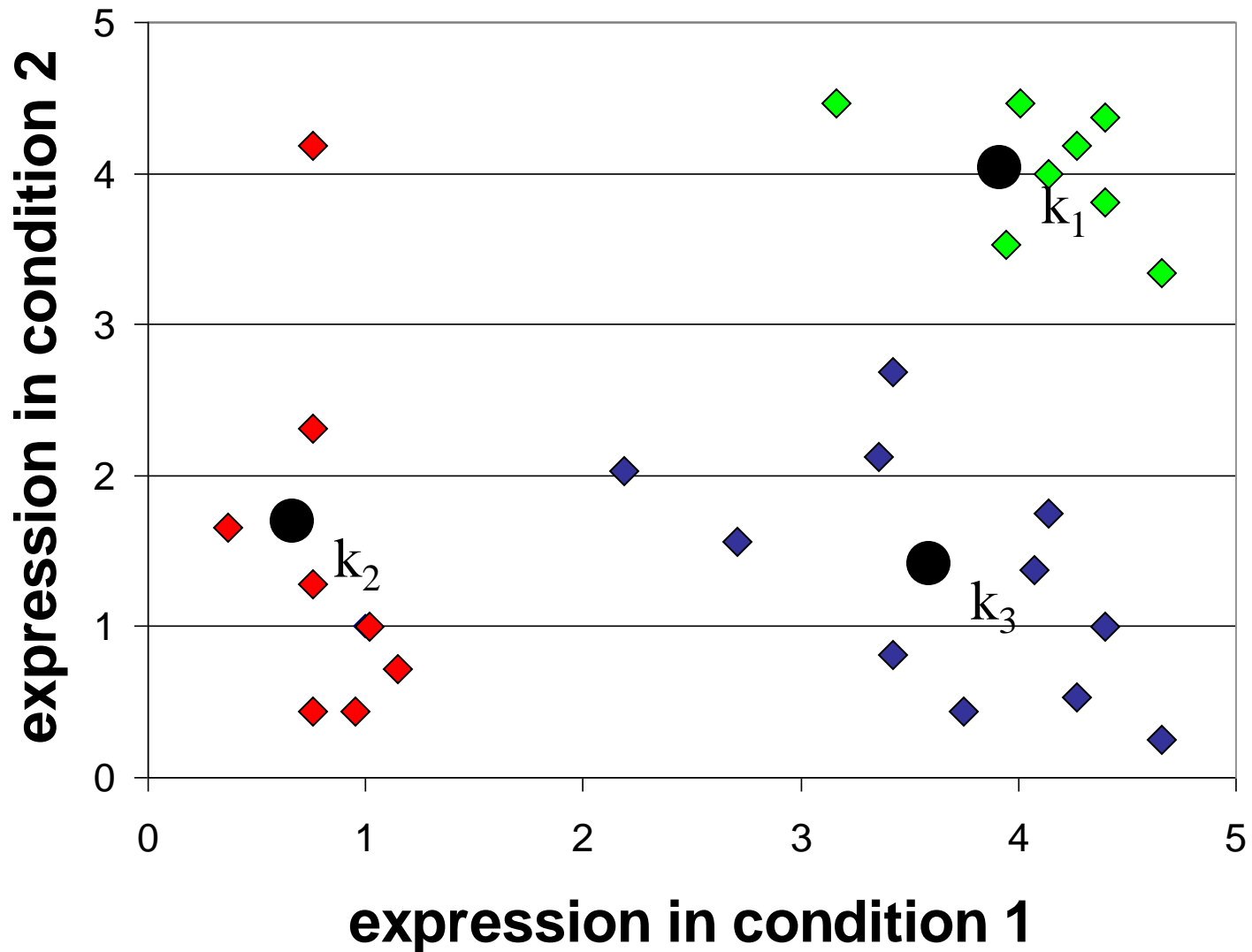
Clustering: Example 2, Step 4

Algorithm: k-means, Distance Metric: Euclidean Distance



Clustering: Example 2, Step 5

Algorithm: k-means, Distance Metric: Euclidean Distance



Conservative K-Means Algorithm – cont.

- Assume that every partition P of the n -element set into k clusters has an associated clustering $cost(P)$ – measures the partition's quality.
- Example for $cost(P)$: the squared error distortion

$$d(\mathbf{V}, \mathbf{X}) = \sum d(v_i, \mathbf{X})^2 / n \quad 1 \leq i \leq n$$

Conservative K-Means Algorithm – cont.

- A partition P is given, a cluster C within P , and an element i outside C .
- $i \rightarrow C$ - the partition obtained by moving i to C
- Improve $\text{cost}(P)$ only if

$$\Delta(i \rightarrow C) = \text{cost}(P) - \text{cost}(P_{i \rightarrow C}) > 0$$

- (The smaller the cost, the better the clustering is)
- The following ProgressiveGreedyK-Means searches for the best move in each step, for every C and for all i outside C

K-Means“Greedy” Algorithm

1. ProgressiveGreedyK-Means(k)
2. Select an arbitrary partition P into k clusters
3. **while** forever
4. $bestChange \leftarrow 0$
5. **for** every cluster C
6. **for** element i not in C
7. **if** moving i to cluster C reduces its clustering cost
8. **if** $(\text{cost}(P) - \text{cost}(P_{i \rightarrow C}) > bestChange$
9. $bestChange \leftarrow (\text{cost}(P) - \text{cost}(P_{i \rightarrow C})$
10. $i^* \leftarrow i$
11. $C^* \leftarrow C$
12. **if** $bestChange > 0$
13. Change partition P by moving i^* to C^*
14. **else**
15. **return** P