



# Clustering

Sistemi informativi per le Decisioni

Slide a cura di Prof. Claudio Sartori



# Scenario: Analisi e gestione dei mercati

## ■ Customer profiling

- Quali tipi di cliente acquistano quali prodotti?
- Clustering, classificazione

## ■ Identificare le richieste dei clienti

- Trovare il prodotto migliore per clienti diversi
- Predire quali fattori possono attrarre nuovi clienti

## ■ Produrre informazioni di sommario

- Rapporti multi-dimensionali
- Sintesi statistiche descrittive



# Clustering - argomenti

- Cosa è il clustering e quali sono i suoi metodi
- L'algoritmo k-means
  - Intuizione
  - Codifica e distorsione
  - Ottimizzazione e terminazione
- Algoritmi gerarchici
  - Dendrogramma
- Algoritmi basati sulla densità



# Clustering - descrizione del problema

- input:

- un insieme di  $N$  oggetti  $d$ -dimensionali

- output:

- determinare un partizionamento naturale dell'insieme di dato in  $k$  clusters + rumore
- proprietà desiderate nei cluster:
  - oggetti nello stesso cluster sono simili  
→ massimizzata la similarità intra-cluster
  - oggetti in cluster diversi sono differenti  
→ minimizzata la similarità inter-cluster



# Prospettiva di ricerca

- Dal passato...
  - il clustering è un problema ben noto in statistica
  - ricerche più recenti
    - machine learning
    - database
    - visualizzazione
- ... per il futuro
  - algoritmi efficaci ed efficienti per il clustering di grandi insiemi di dati (in rapido aumento), con elevato numero di dimensioni, molto rumore
  - richiede scalabilità rispetto a:
    - numero di punti dati (N)
    - numero di dimensioni (d)
    - livello di rumore
    - frequenza di aumento del numero di punti dati



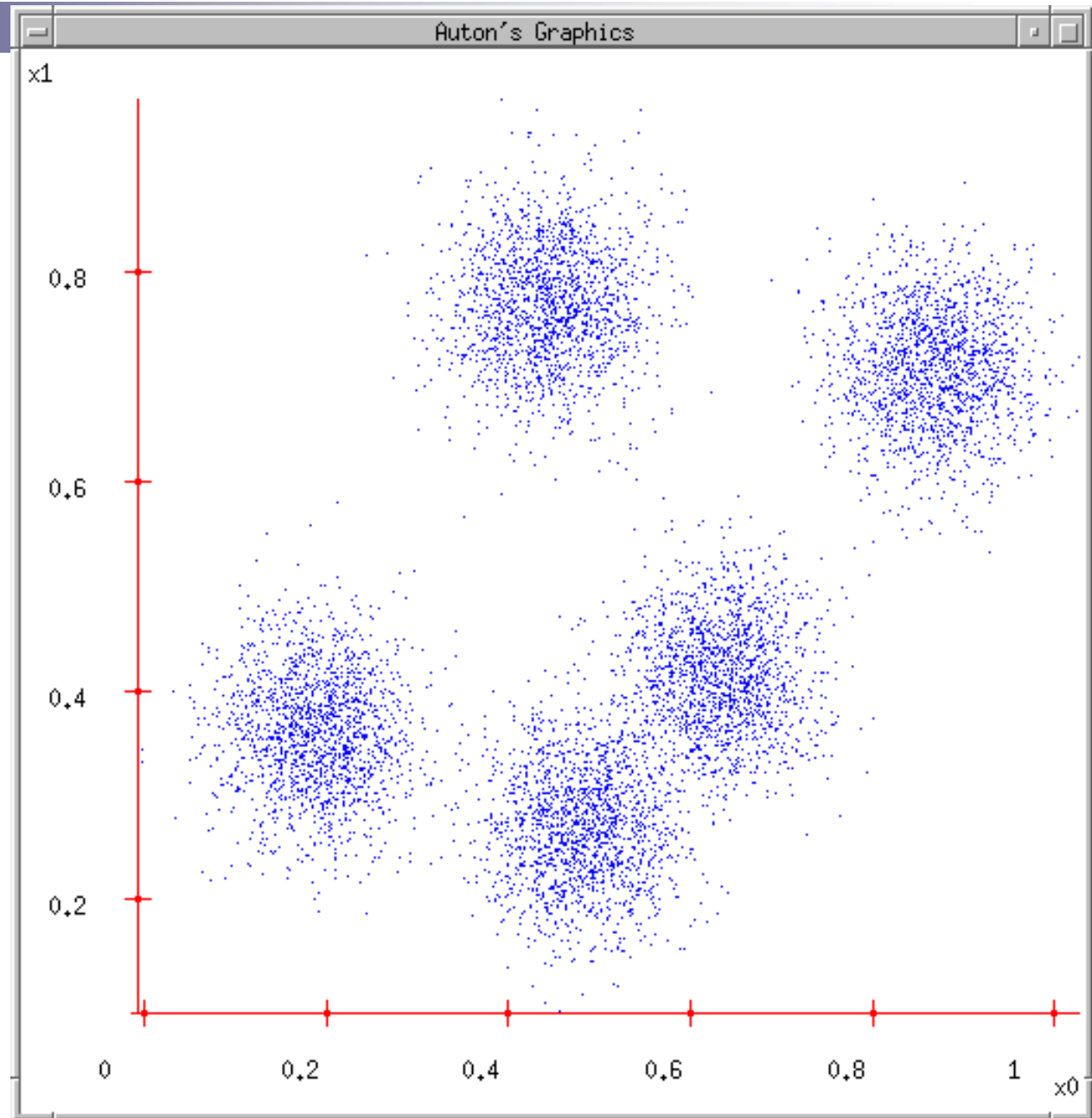
# Tassonomia dei principali metodi di clustering

- partizionanti
  - k-means (MacQueen 67)
  - expectation maximization (Lauritzen 95)
  - CLARANS (Ng and Han 94)
- gerarchici
  - agglomerativi/divisivi
  - BIRCH (Zhang et al 96)
  - CURE (Guha et al 98)
- basati sul collegamento (linkage)
- basati sulla densità
  - DBSCAN (Ester et al 96)
  - DENCLUE (Hinnenburg and Keim 98)
- statistici
  - IBM-IM demographic clustering
  - COBWEB (Fisher 87)
  - Autoclass (Cheeseman 96)

# Alcuni dati

Potrebbero  
facilmente essere  
modellati come  
una distribuzione  
gaussiana con 5  
componenti

Ma cerchiamo  
una soluzione più  
“amichevole”  
e soddisfacente...

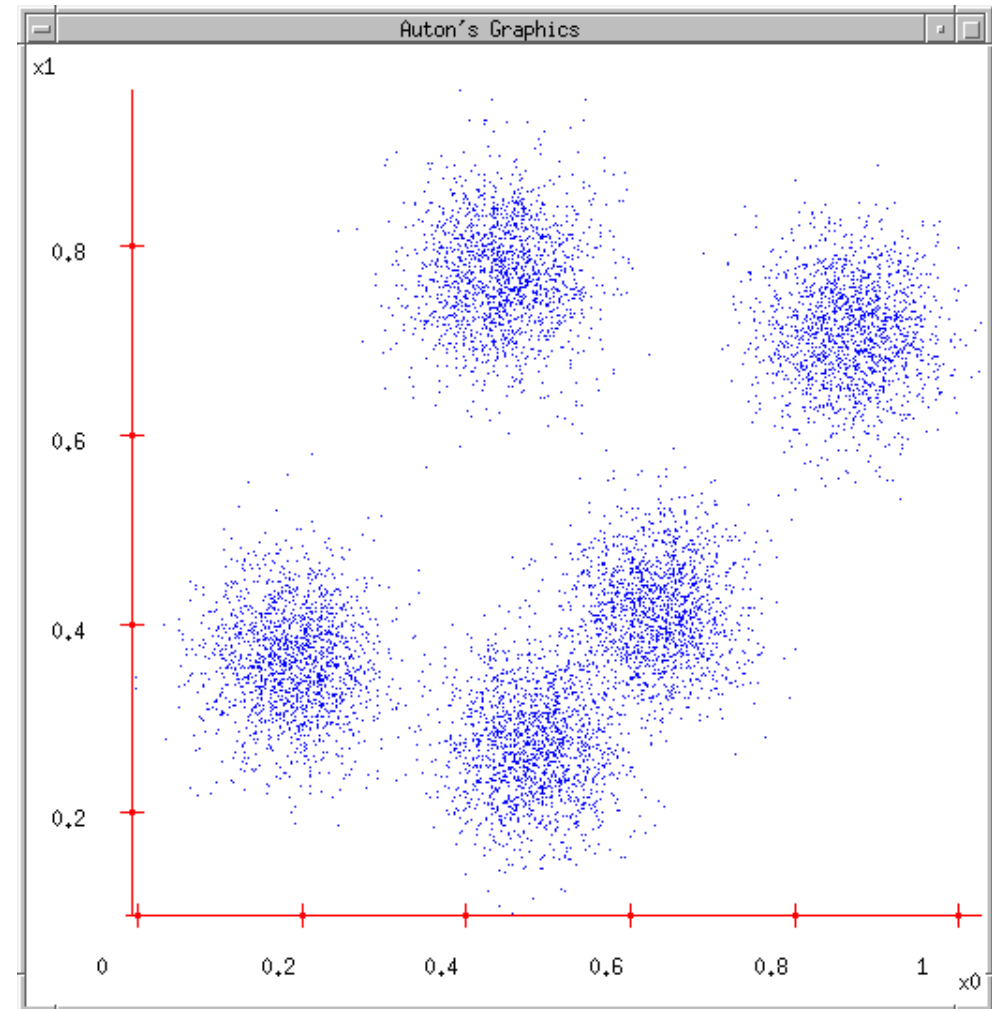


# Compressione con perdita

Supponiamo di dover trasmettere le coordinate di punti presi a caso da questo insieme: dovremo ideare un meccanismo di codifica/decodifica. Limitazione: ci è permesso di trasmettere soltanto due bit per punto. La trasmissione sarà *con perdita (lossy)*

*Perdita* = somma dei quadrati degli errori tra le coordinate decodificate e quelle originali.

Quale codifica/decodifica minimizza la perdita?



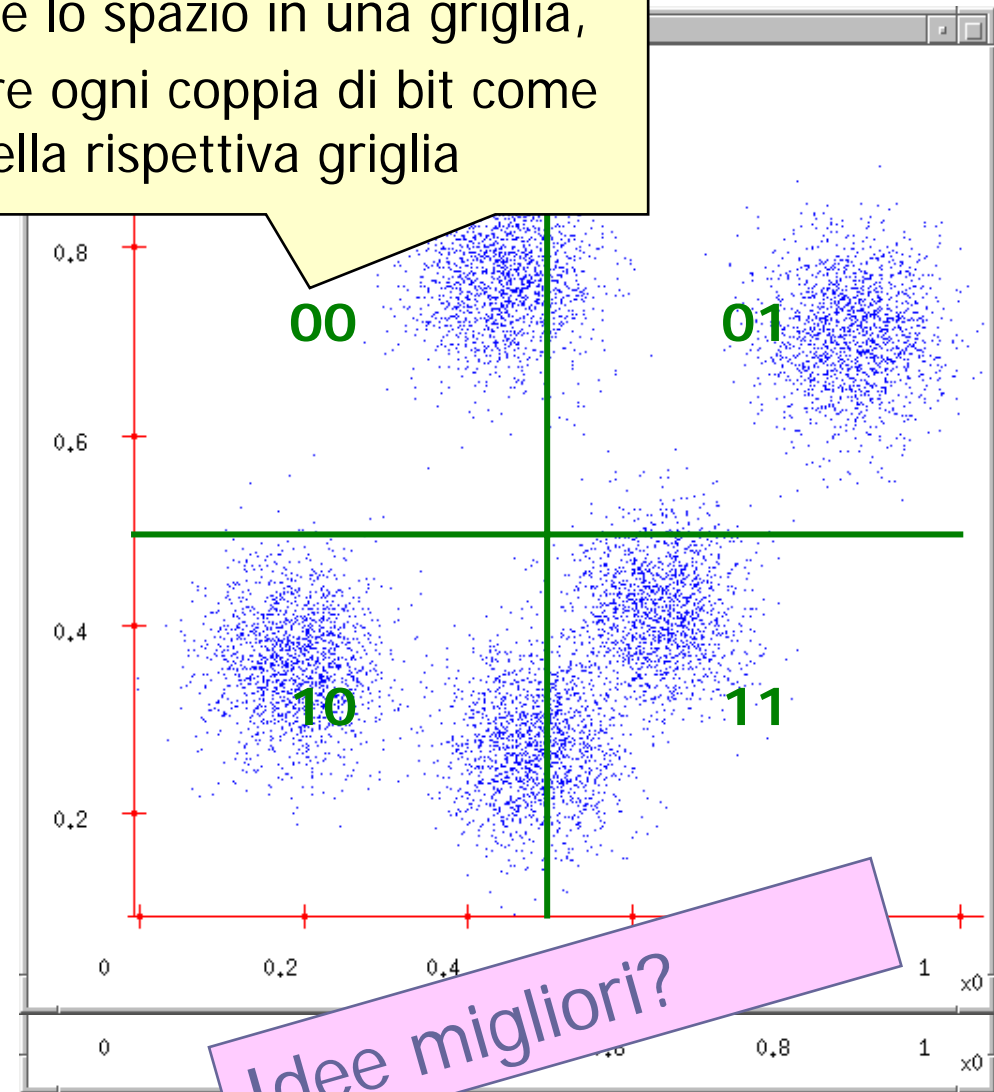


# Idea uno

Supponiamo di dover trasmettere le coordinate di punti presi a caso da questo insieme: dovremo ideare un meccanismo di codifica/decodifica. Limitazione: ci è permesso di trasmettere soltanto due bit per punto. La trasmissione sarà *con perdita (lossy)*

*Perdita* = somma dei quadrati degli errori tra le coordinate decodificate e quelle originali. Quale codifica/decodifica minimizza la perdita?

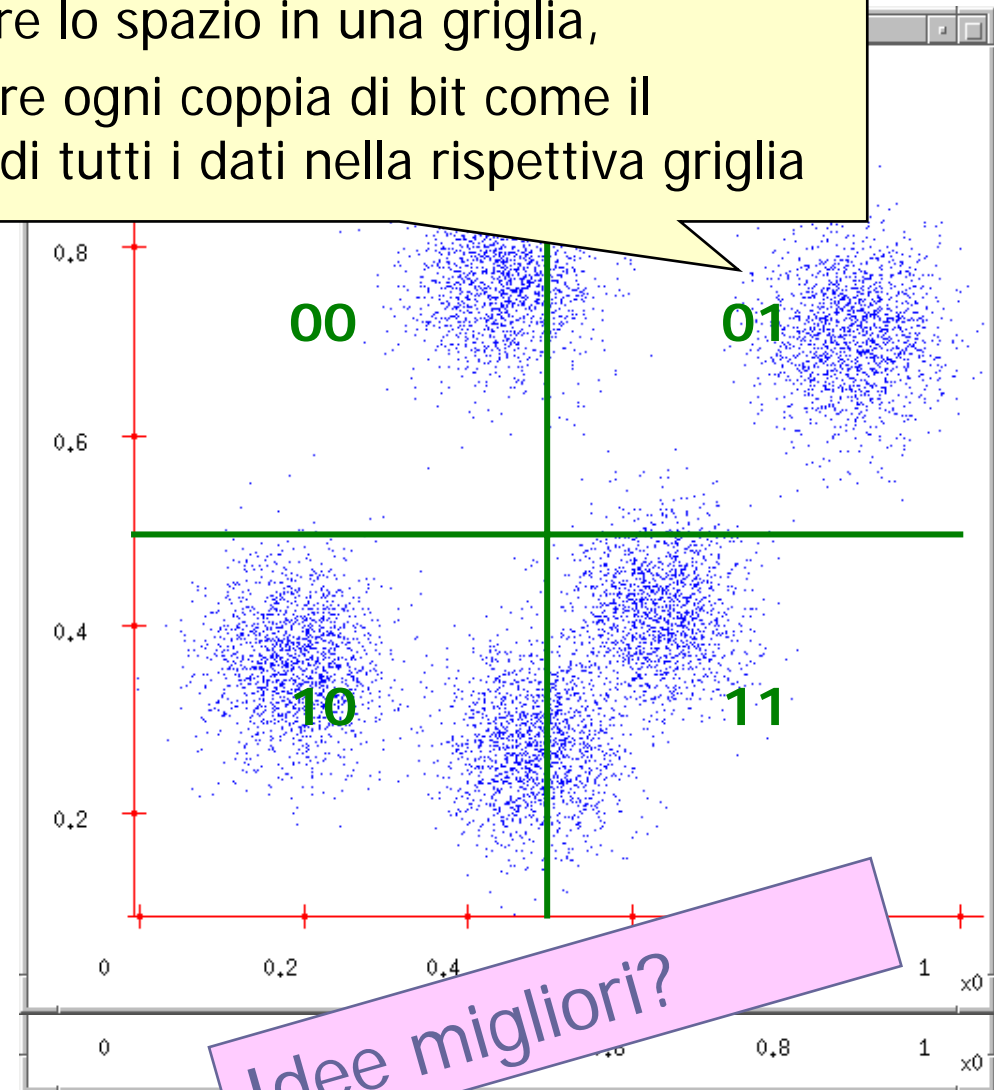
partizionare lo spazio in una griglia, decodificare ogni coppia di bit come il centro della rispettiva griglia



# Idea due

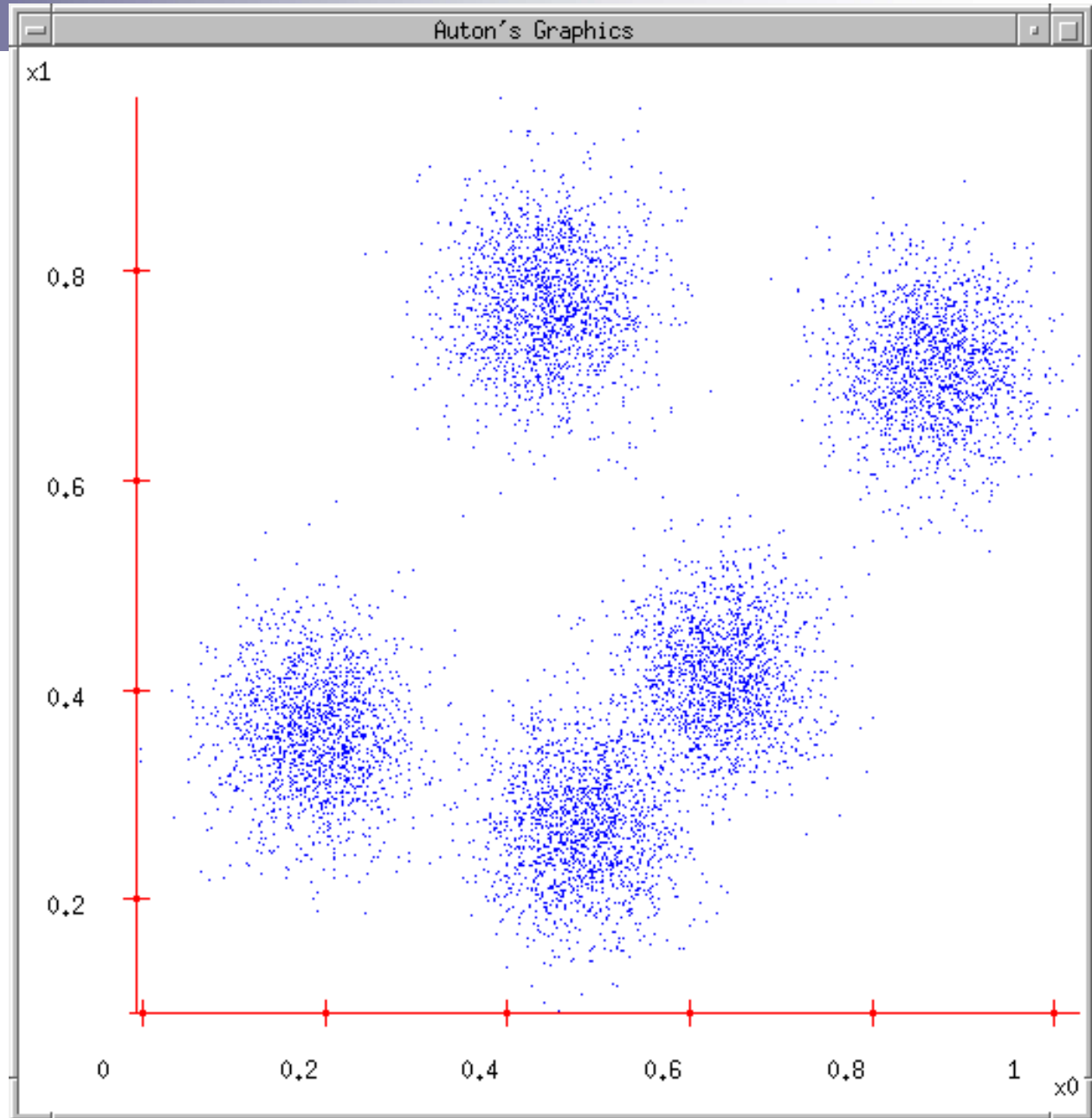
Supponiamo di dover trasmettere le coordinate di punti presi a caso da questo insieme: dovremo ideare un meccanismo di codifica/decodifica. Limitazione: ci è permesso di trasmettere soltanto due bit per punto. La trasmissione sarà *con perdita (lossy)*. *Perdita* = somma dei quadrati degli errori tra le coordinate decodificate e quelle originali. Quale codifica/decodifica minimizza la perdita?

partizionare lo spazio in una griglia, decodificare ogni coppia di bit come il centroide di tutti i dati nella rispettiva griglia



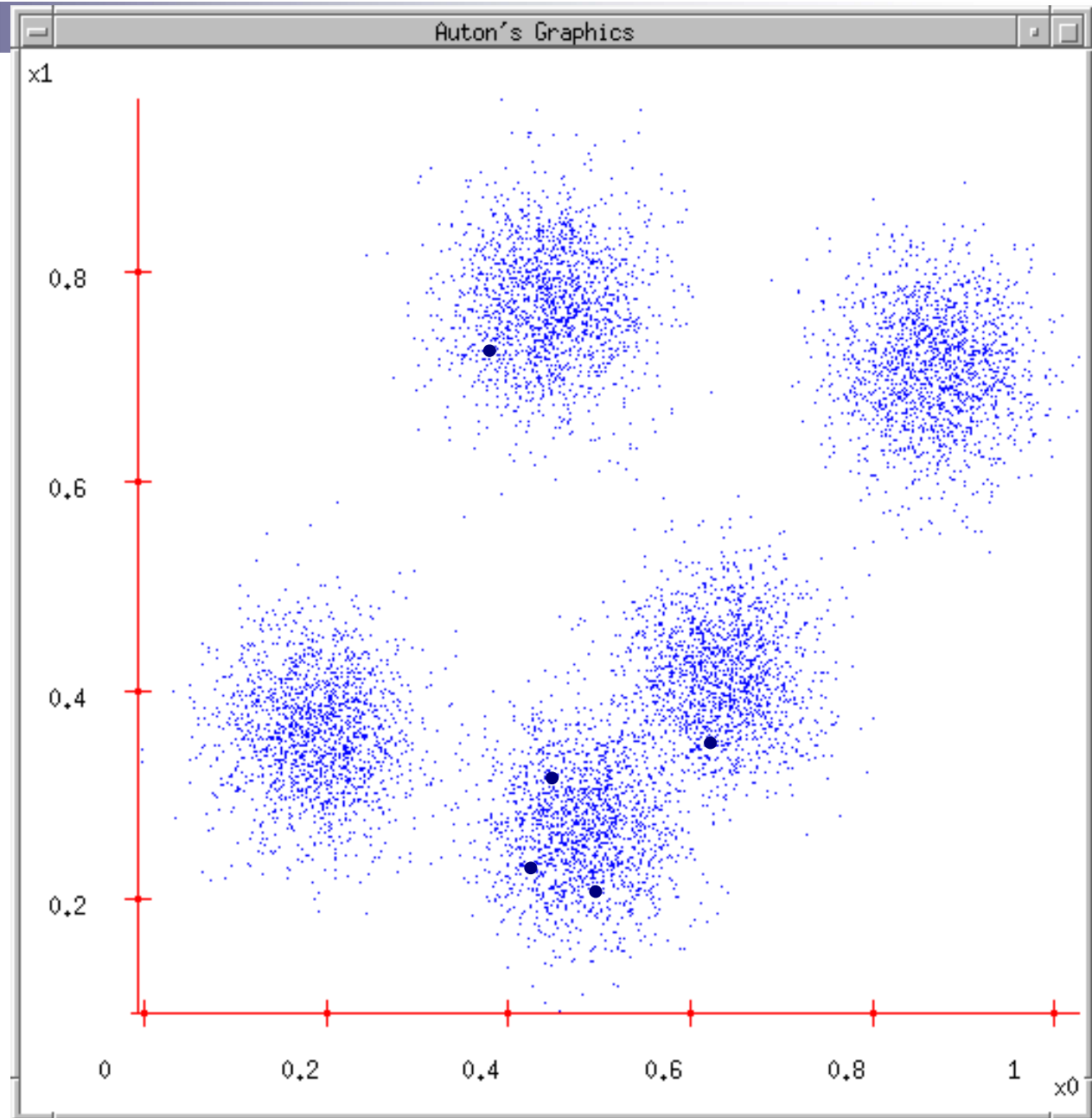
# K-means

1. chiedi all'utente quanti cluster vuole (es.  $k=5$ )



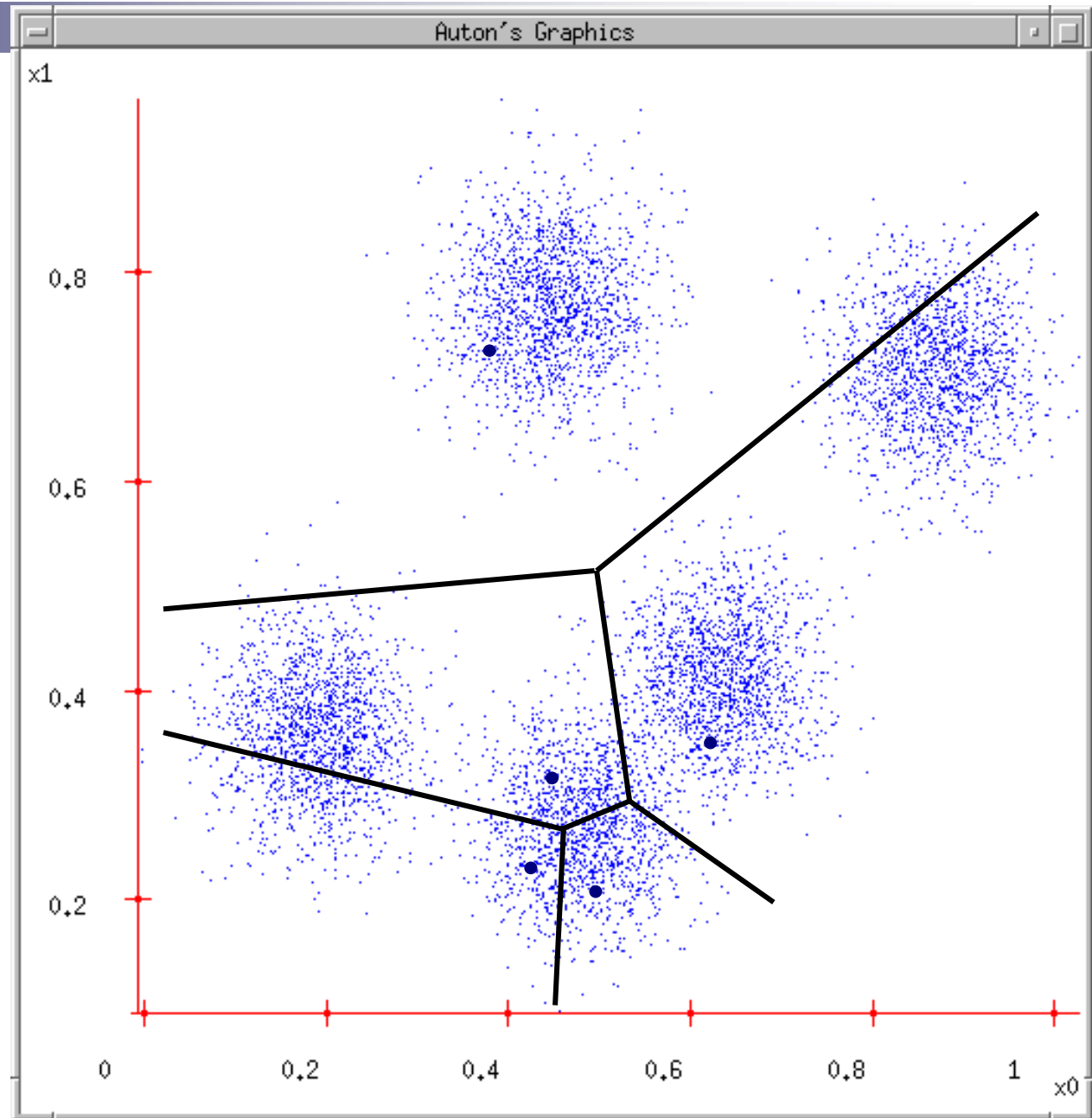
# K-means

1. chiedi all'utente quanti cluster vuole (es.  $k=5$ )
2. scegli a caso  $k$  posizioni come centri



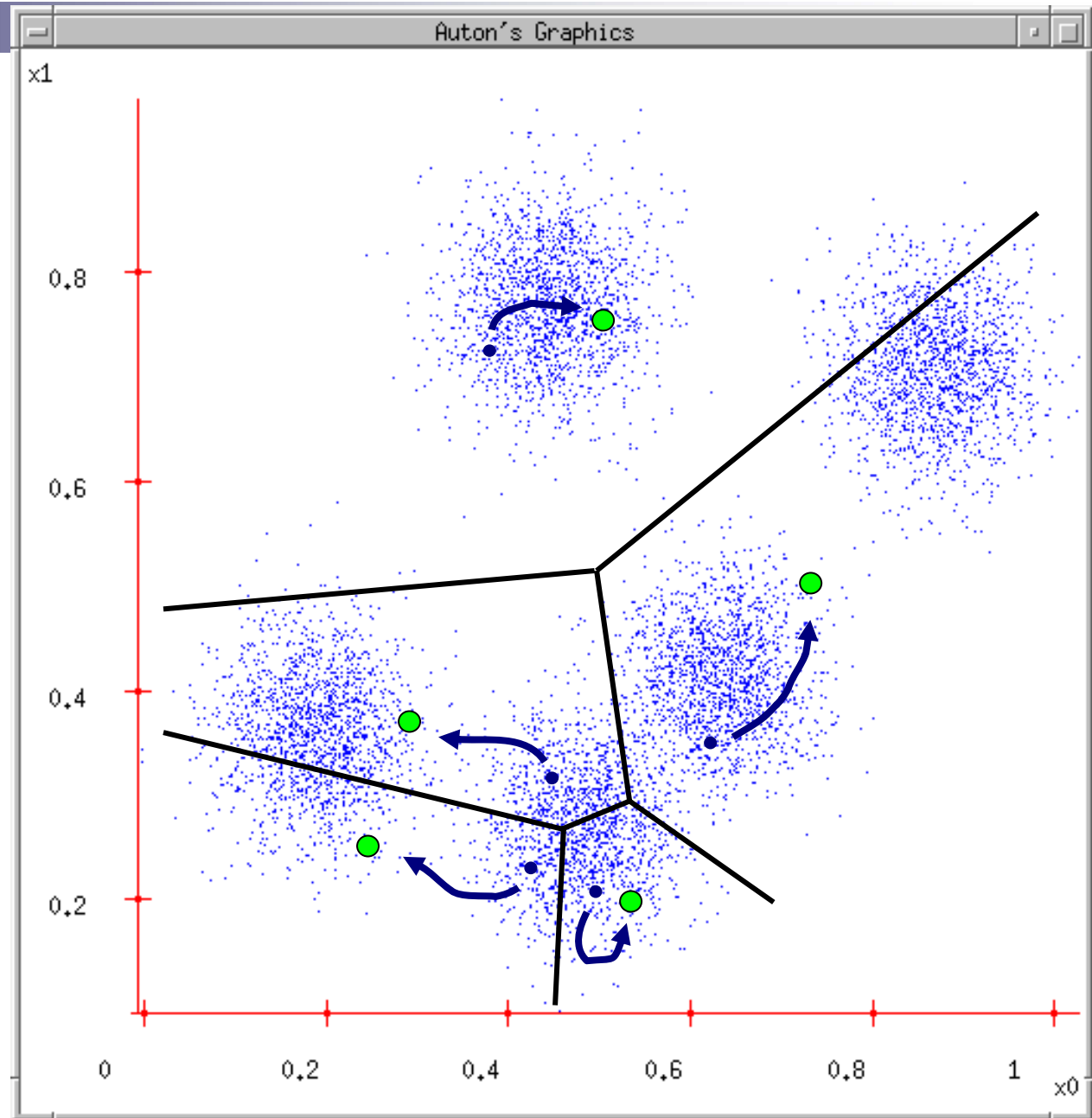
# K-means

1. chiedi all'utente quanti cluster vuole (es.  $k=5$ )
2. scegli a caso  $k$  posizioni come centri
3. ogni punto trova quale è il suo centro più vicino



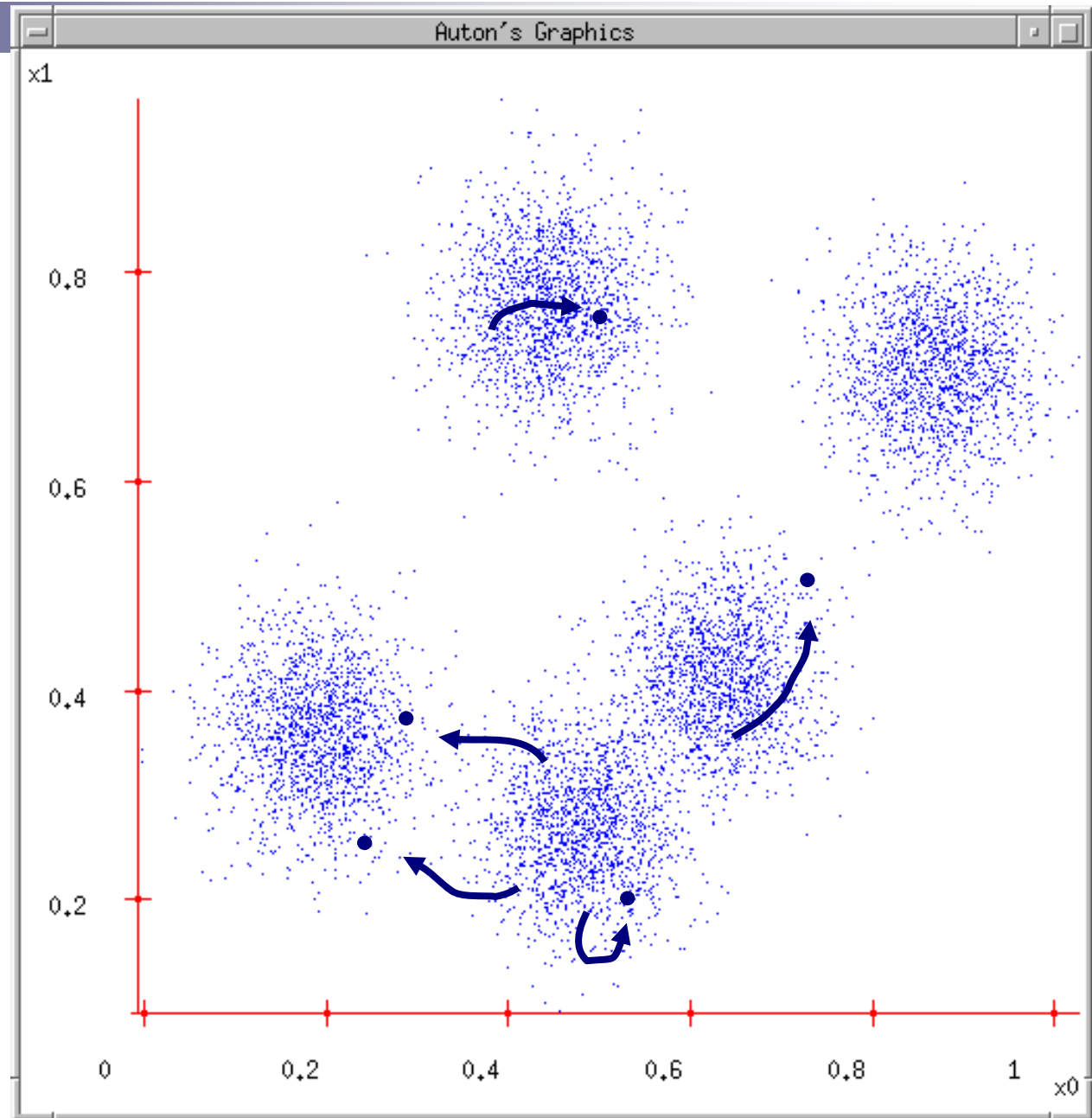
# K-means

1. chiedi all'utente quanti cluster vuole (es.  $k=5$ )
2. scegli a caso  $k$  posizioni come centri
3. ogni punto trova quale è il suo centro più vicino
4. ogni centro trova il centroide dei punti che possiede...



# K-means

1. chiedi all'utente quanti cluster vuole (es.  $k=5$ )
2. scegli a caso  $k$  posizioni come centri
3. ogni punto trova quale è il suo centro più vicino
4. ogni centro trova il centroide dei punti che possiede...
5. ... e si sposta nel centroide
6. ... ripeti fino al termine!

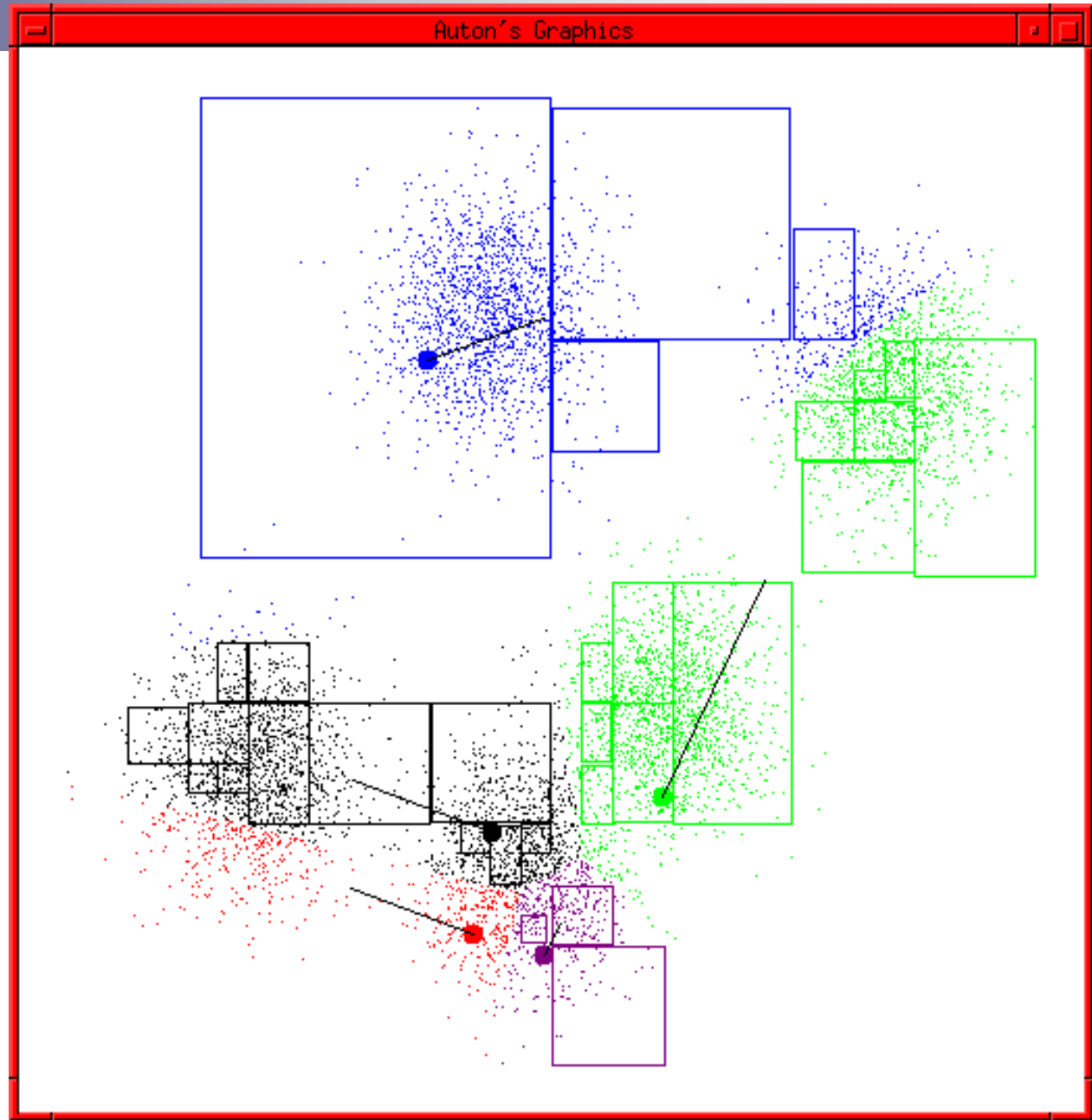




# K-means Partenza

Example generated by  
Dan Pelleg's super-duper  
fast K-means system:

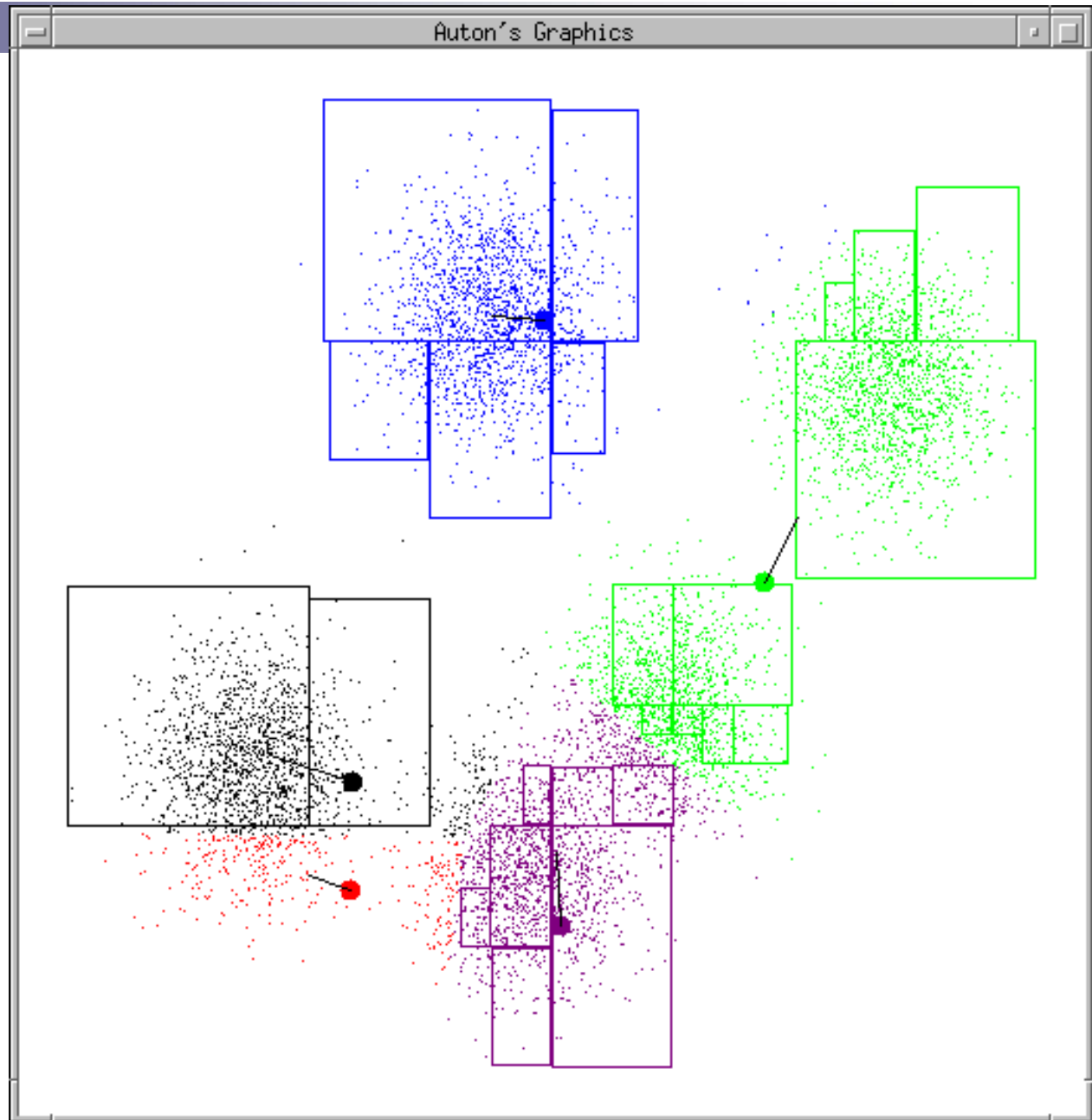
*Dan Pelleg and Andrew  
Moore. Accelerating Exact  
k-means Algorithms with  
Geometric Reasoning.  
Proc. Conference on  
Knowledge Discovery in  
Databases 1999,  
(KDD99) (available on  
[www.autonlab.org/pap.html](http://www.autonlab.org/pap.html))*





# K-means continua

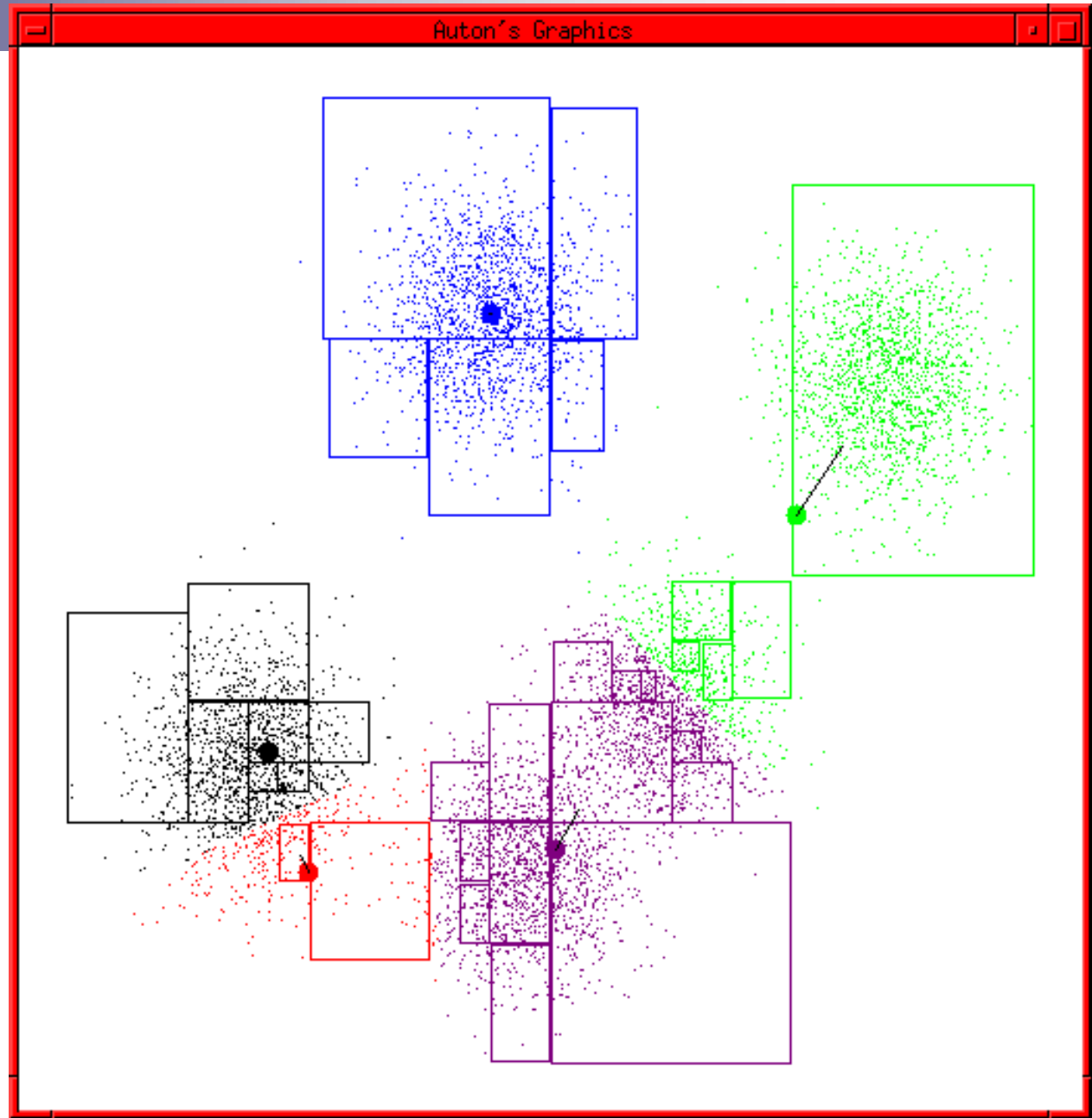
...





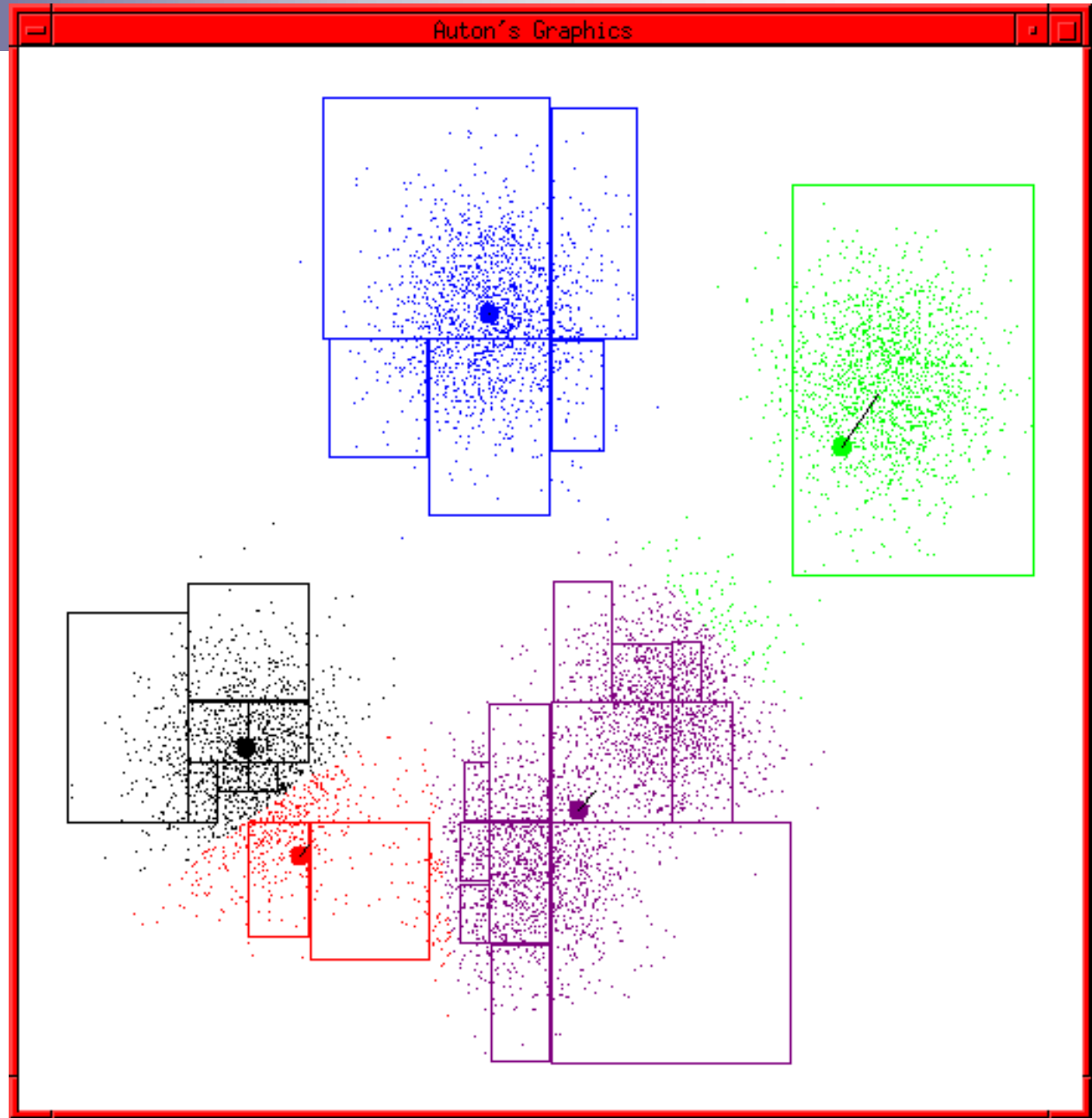
# K-means continua

...



# K-means continua

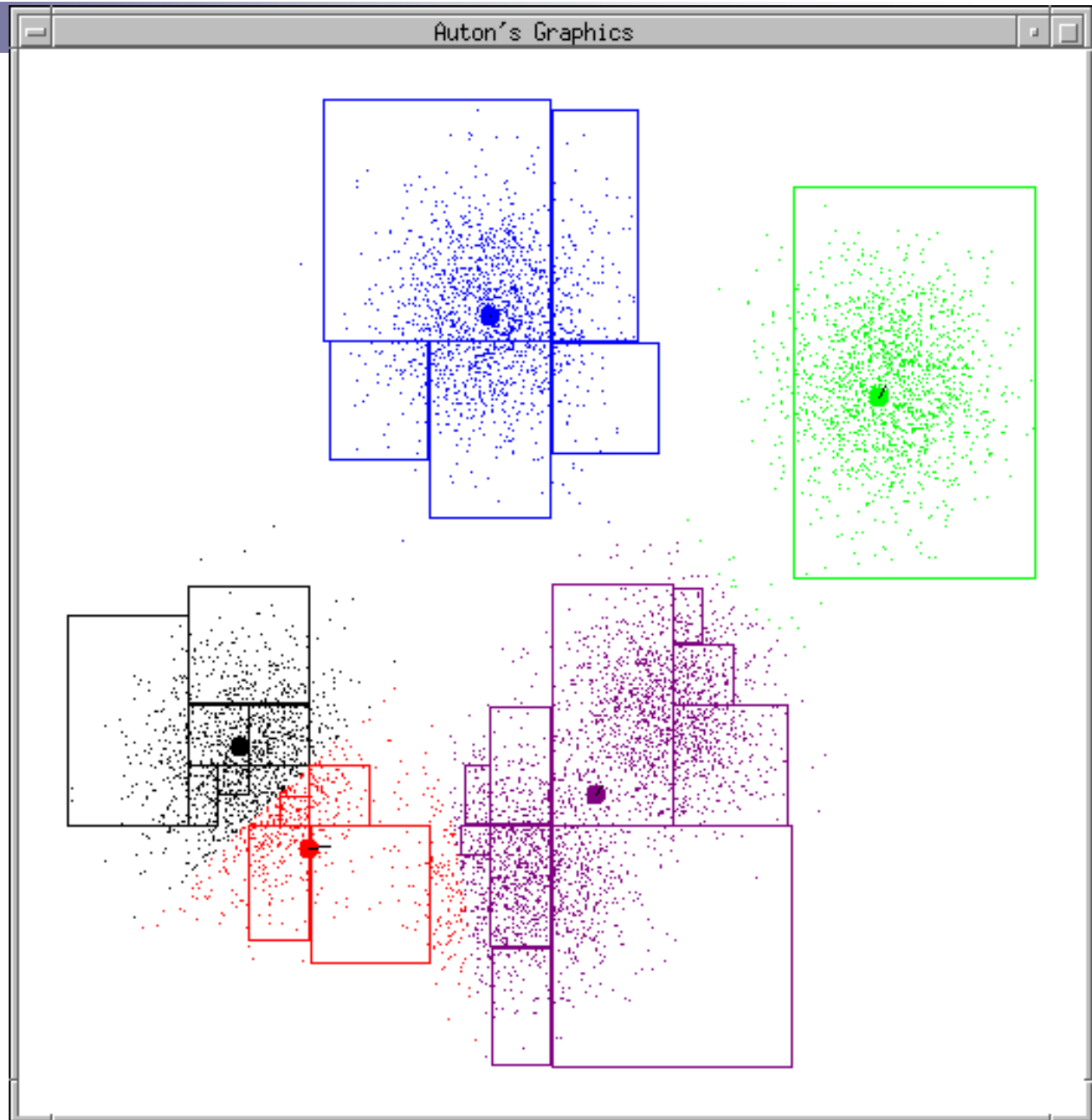
...





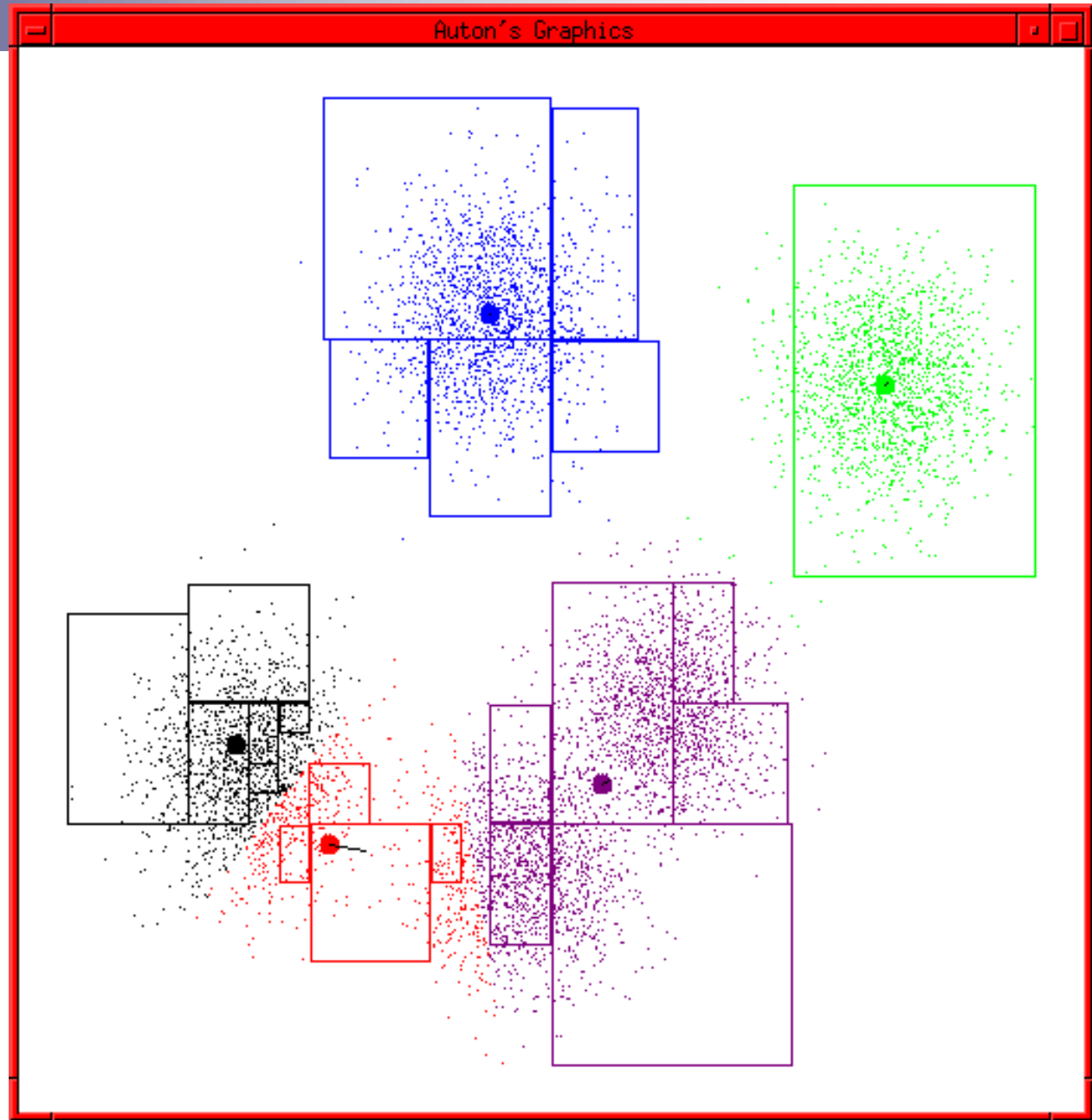
# K-means continua

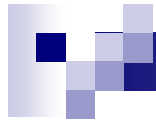
...



# K-means continua

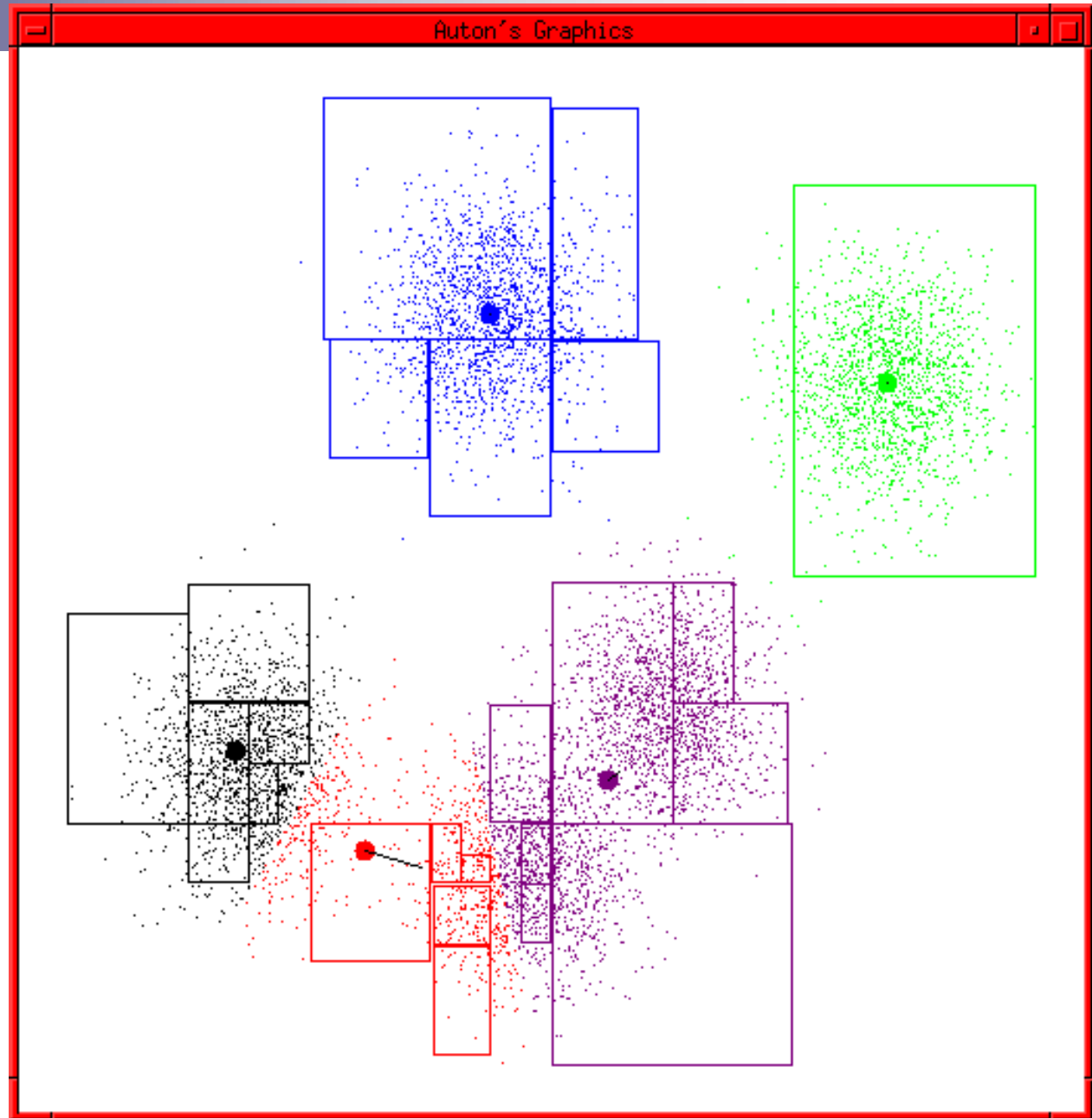
...





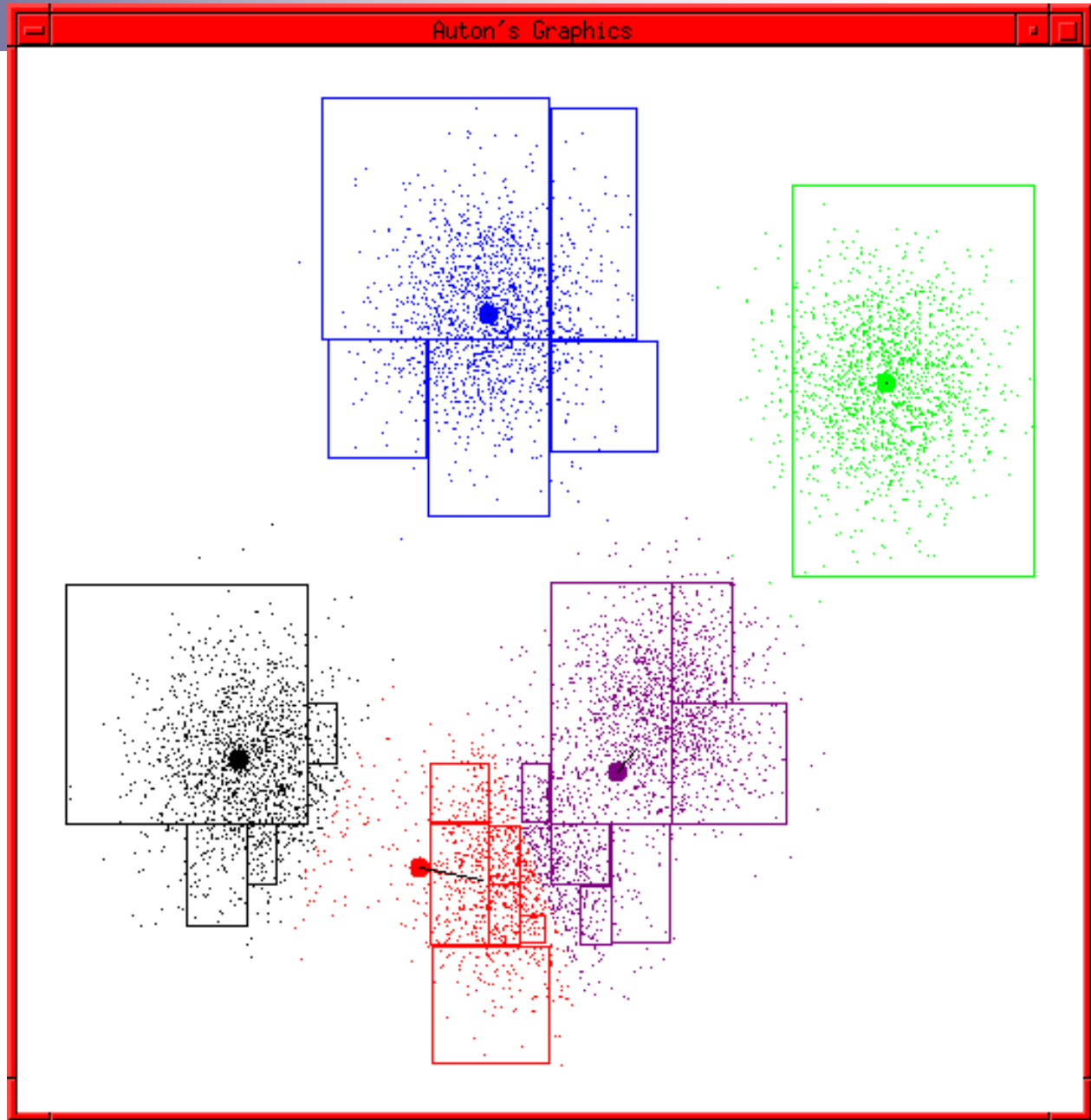
# K-means continua

...



# K-means continua

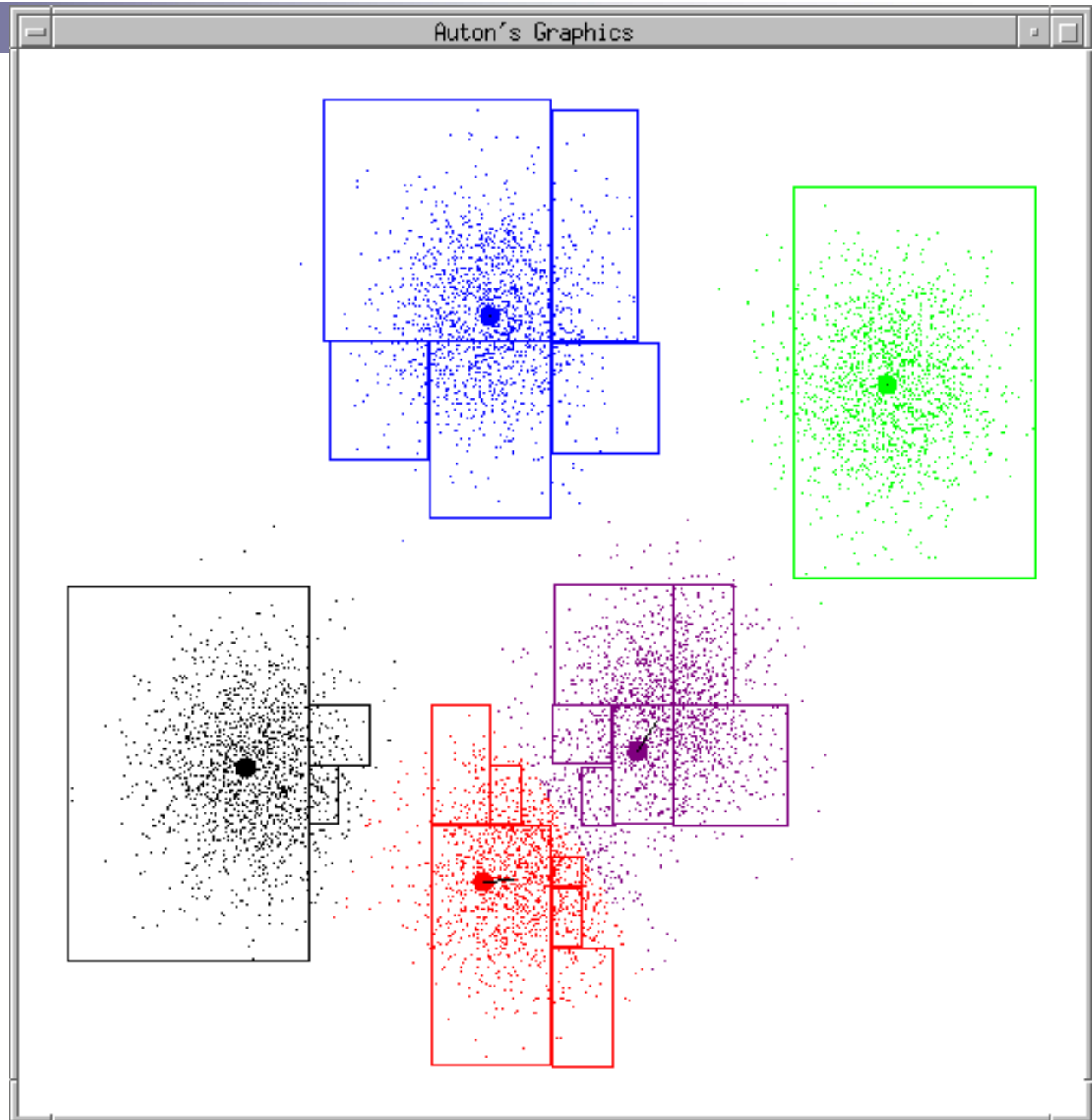
...





# K-means continua

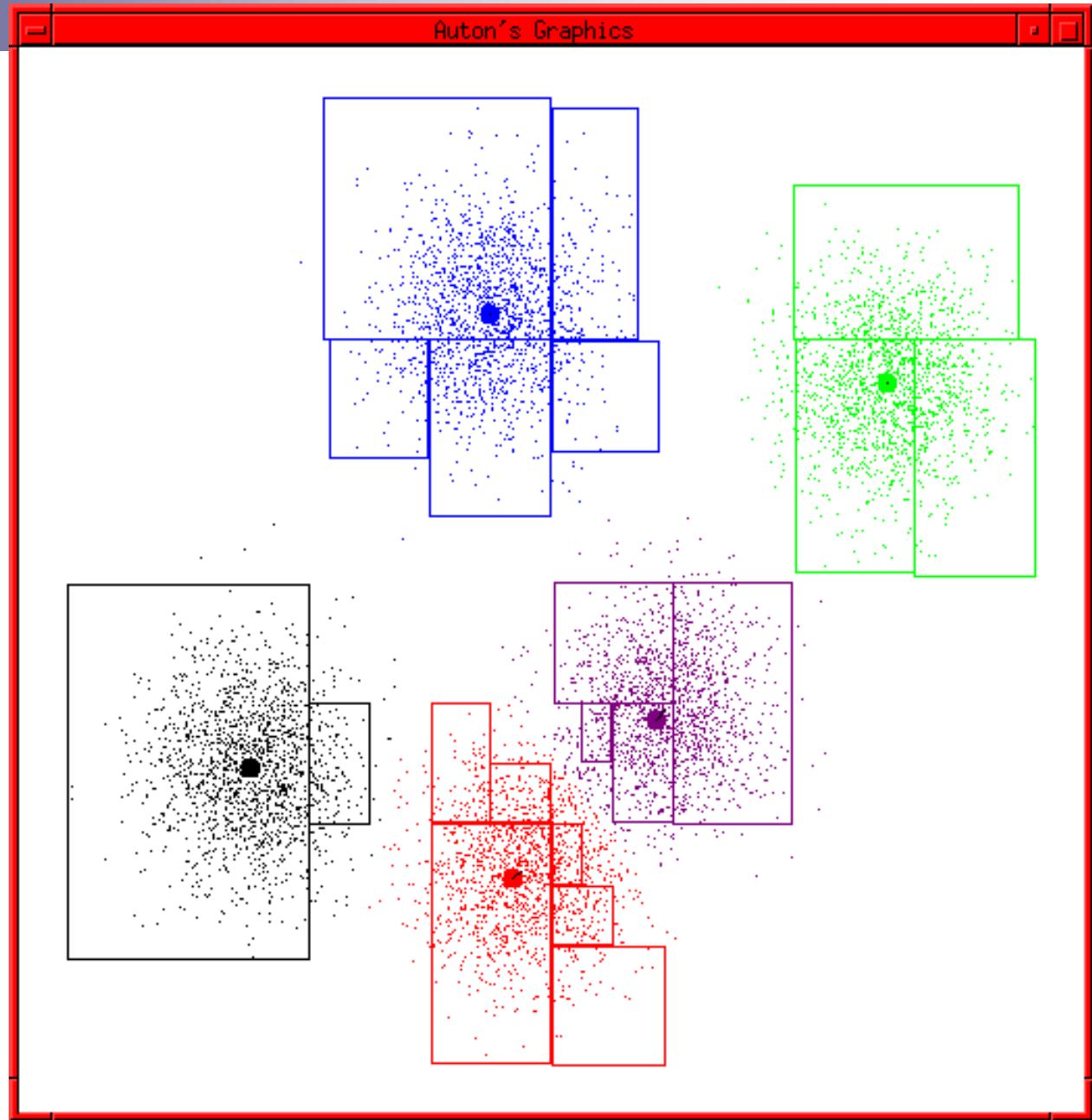
...







# K-means termina





# K-means: domande

- cosa si cerca di ottimizzare?
- è garantita la terminazione?
- siamo certi che verrà trovato il miglior clustering?
- come sarebbe opportuno partire?
- come si determina il numero di centri?
- ....qualche risposta dal pubblico...

# Distorsione

Date..

- una funzione di codifica:  $\text{ENCODE} : \mathfrak{R}^m \rightarrow [1, k]$
- una funzione di decodifica:  $\text{DECODE} : [1, k] \rightarrow \mathfrak{R}^m$

Definiamo

$$\text{Distortion} = \sum_{i=1}^R (\mathbf{x}_i - \text{DECODE}[\text{ENCODE}(\mathbf{x}_i)])^2$$

possiamo anche scrivere

$$\text{DECODE}[j] = \mathbf{c}_j$$

quindi

$$\text{Distortion} = \sum_{i=1}^R (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$



# Distorsione minimale

$$\text{Distortion} = \sum_{i=1}^R (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

Quali proprietà devono avere i centri  $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_k$  affinché la distorsione sia minimizzata?

# Distorsione minimale

$$\text{Distortion} = \sum_{i=1}^R (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

Quali proprietà devono avere i centri  $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_k$  affinché la distorsione sia minimizzata?

- $x_i$  deve essere codificato con il suo centro più vicino

$$\mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)} = \arg \min_{\mathbf{c}_j \in \{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_k\}} (\mathbf{x}_i - \mathbf{c}_j)^2$$

- perché?

# Distorsione minimale

$$\text{Distortion} = \sum_{i=1}^R (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

Quali proprietà devono avere i centri  $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_k$  affinché la distorsione sia minimizzata?

- $x_i$  deve essere codificato con il suo centro più vicino

diversamente la distorsione potrebbe essere ridotta sostituendo  $\text{ENCODE}[\mathbf{x}_i]$  con il centro più vicino

- perché?



# Distorsione minimale

$$\text{Distortion} = \sum_{i=1}^R (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

Quali proprietà devono avere i centri  $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_k$  affinché la distorsione sia minimizzata?

- La derivata parziale della distorsione rispetto alla posizione di ogni centro deve essere = 0

La derivata parziale della distorsione rispetto alla posizione di ogni centro deve essere = 0

$$\text{Distortion} = \sum_{i=1}^R (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

$$= \sum_{j=1}^k \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)^2$$

OwnedBy( $\mathbf{c}_j$ ) = insieme di record associati al centro  $\mathbf{c}_j$

$$\frac{\partial \text{Distortion}}{\partial \mathbf{c}_j} = \frac{\partial}{\partial \mathbf{c}_j} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)^2$$

$$= -2 \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j) = 0$$

perciò, al minimo:  $\mathbf{c}_j = \frac{1}{|\text{OwnedBy}(\mathbf{c}_j)|} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} \mathbf{x}_i$





# Distorsione minimale

$$\text{Distortion} = \sum_{i=1}^R (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

Quali proprietà devono avere i centri  $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_k$  affinché la distorsione sia minimizzata?

- $x_i$  deve essere codificato con il suo centro più vicino
- ogni centro deve essere il centroide dei punti a lui associati



## Migliorare una configurazione sub-ottimale

- Cambiare la codifica affinché  $x_i$  sia codificato dal suo centro più vicino
- Posizionare ogni centro al centroide dei punti posseduti
- Inutile applicare una operazione due volte
- Basta alternare i due passi
- ... and that's K-means!
  - *È possibile provare che la procedura terminerà in uno stato in cui nessuno dei due passi cambierà la configurazione?*
  - *Perché?*

# Migliorare una configurazione sub-ottimale

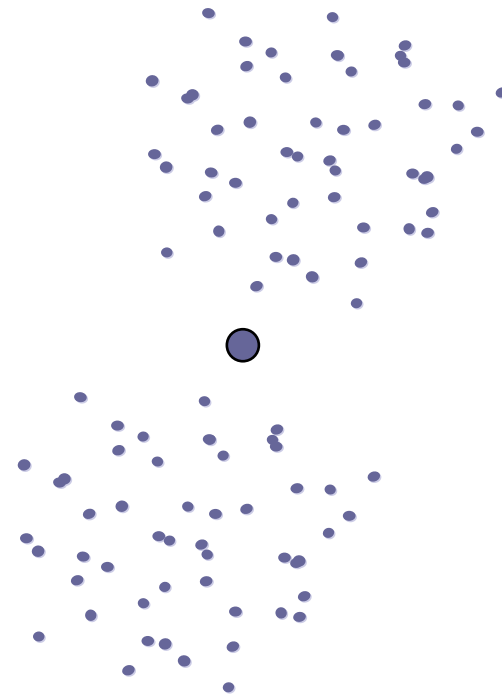
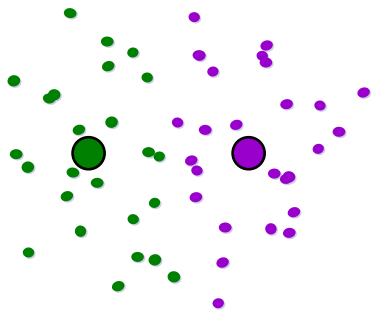
- Cambiare la configurazione dal suo stato attuale
- Posizioni dei centri non possedute
- Inutile
- Basta
- ... anche

- È possibile migliorare la configurazione?
- Perché?

- C'è soltanto un numero finito di modi di partizionare  $R$  record in  $k$  gruppi
- Quindi c'è soltanto un numero finito di possibili configurazioni in cui tutti i centri sono centroidi dei punti a loro associati
- Se la configurazione cambia in una iterazione, deve avere migliorato la distorsione
- Quindi ogni volta che la configurazione cambia, deve portare in uno stato mai visitato prima
- Quindi l'algoritmo deve arrestarsi per non disponibilità di ulteriori configurazioni da visitare

# La configurazione trovata sarà ottima?

- Non necessariamente...
- Ideare una configurazione che si arresta ma non ha distorsione minima...





# Cercare un buon ottimo

- Idea 1: Attenti al punto di partenza
- Idea 2: Eseguire l'algoritmo molte volte, ognuna con una diversa configurazione di partenza a caso
- ...

# Cercare un buon ottimo

- Idea 1: Attenti al punto di partenza

- Idea 2: ...

Accorgimenti:

- Scegliere come primo centro un punto a caso
- Scegliere come secondo centro un punto più lontano possibile dal primo

- ...

:

- Scegliere come  $j$ -esimo centro un punto più lontano possibile dal più vicino dei precedenti

:



# Scegliere il numero di cluster

- difficile...
- sperimentare diversi valori
- il valore migliore mostra
  - le minime distanze intra-cluster, e
  - le massime distanze inter-cluster



# Usi comuni di K-means

- Spesso usato come strumento per l'analisi esplorativa dei dati
- Nel caso mono-dimensionale, è un buon modo per discretizzare variabili reali in bucket non uniformi
- Usato nella comprensione del parlato, per convertire forme d'onda in una di  $k$  categorie (noto anche come Vector Quantization)
- Usato anche per scegliere le color palettes nei vecchi display grafici!





# Commenti su K-means

## ■ Punti di forza

- relativamente efficiente:  $O(tkn)$ 
  - $n$  è il numero di oggetti
  - $k$  il numero di cluster
  - $t$  il numero di iterazioni; normalmente  $k, t \ll n$
- spesso termina con un ottimo locale
- l'ottimo globale può essere trovato con tecniche come deterministic annealing e algoritmi genetici

## ■ Punti di debolezza

- applicabile soltanto in spazi in cui è definibile la media; che fare ad esempio con dati categorici?
- occorre fissare a priori  $k$
- non tratta adeguatamente dati con rumore e outlier
- inadatto a scoprire cluster con forme non convesse

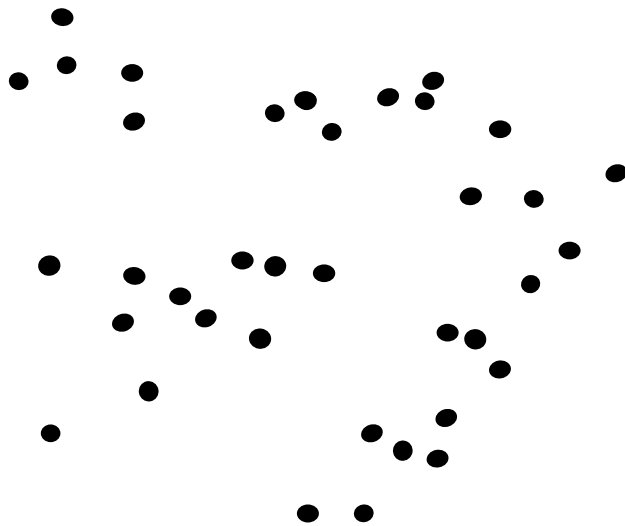


# Varianti a K-means

- Alcune varianti differiscono per
  - modo di selezione dei k centri iniziali
  - calcolo delle dissimilarità
  - strategie per calcolare i centroidi
- Trattamento di dati categorici: K-modes (Huang '98)
  - i centroidi sono sostituiti da modes
  - per gli oggetti categorici sono definite nuove misure di dissimilarità
  - i modes dei cluster sono aggiornati con metodi basati sulla frequenza

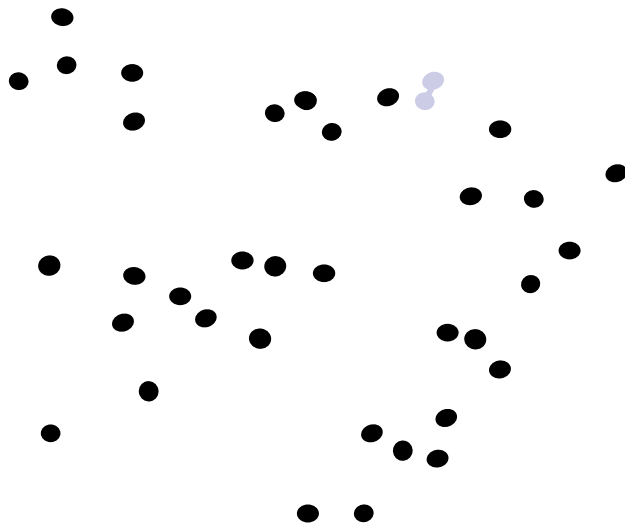


# Single Linkage Hierarchical Clustering



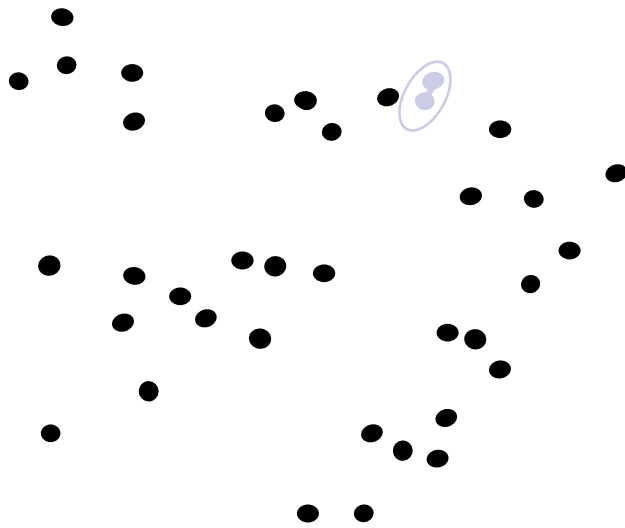
1. supponiamo che “ogni punto sia il proprio cluster”

# Single Linkage Hierarchical Clustering



1. supponiamo che “ogni punto sia il proprio cluster”
2. trovare la coppia “più simile”

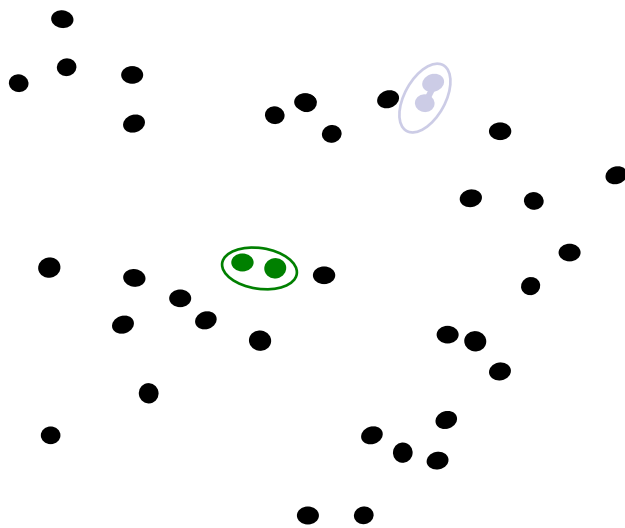
# Single Linkage Hierarchical Clustering



1. supponiamo che “ogni punto sia il proprio cluster”
2. trovare la coppia “più simile”
3. fonderla in un unico cluster



# Single Linkage Hierarchical Clustering



1. supponiamo che “ogni punto sia il proprio cluster”
2. trovare la coppia “più simile”
3. fonderla in un unico cluster
4. ripetere



# Single Linkage Hierarchical Clustering

Come definiamo la similarità?

- minima distanza tra i punti nei cluster (Euclidian Minimum Spanning Trees)
- massima distanza tra i punti nei cluster
- distanza media tra i punti nei cluster

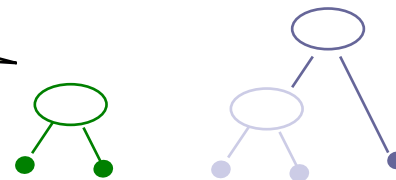
1. supponiamo che “ogni punto sia il proprio cluster”

2. scegli la coppia “più simile”

3. fonderla in un unico cluster

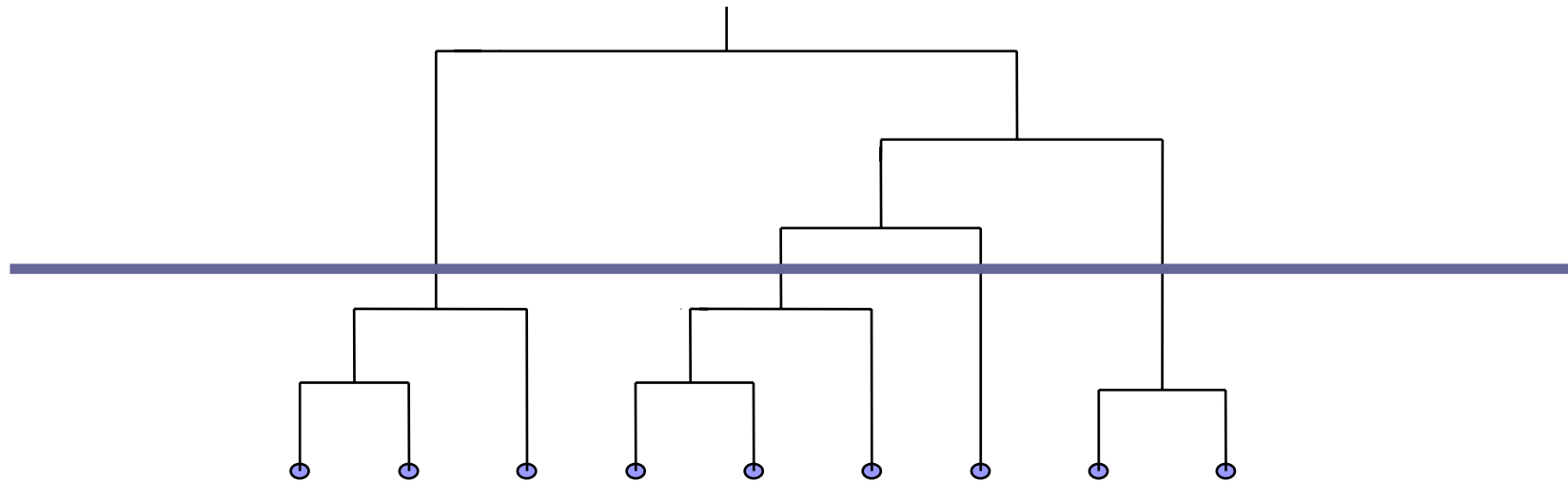
4. ripetere

Risulta un “dendrogramma”, o tassonomia, o gerarchia di punti



# Dendrogrammi

- decompongono gli oggetti in diversi livelli di partizionamento innestati
- si ottiene un *clustering* tagliando il dendrogramma al livello desiderato
  - ogni componente connesso forma un cluster



Clustering



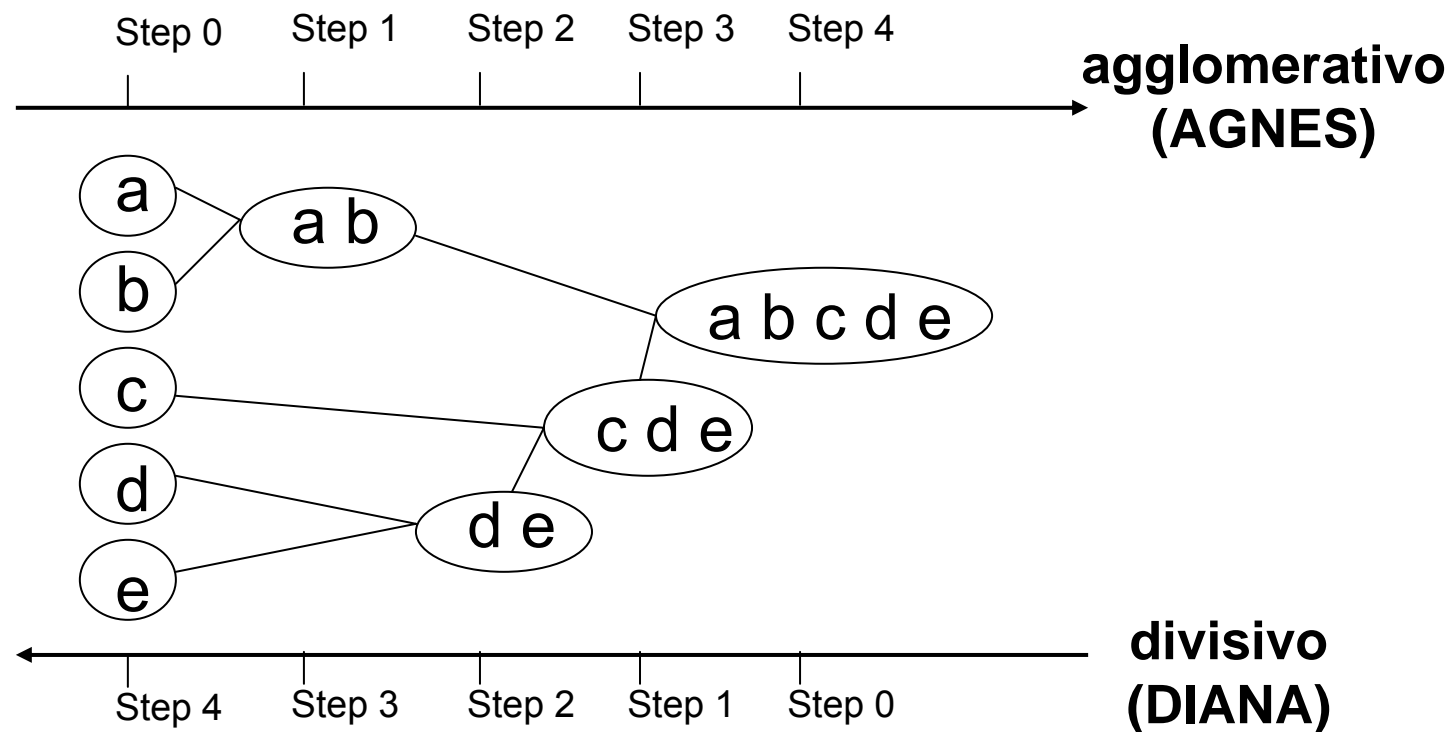


# Commenti a Single Linkage

- Noto anche come Hierarchical Agglomerative Clustering (HAC)
- È rassicurante avere una ben organizzata gerarchia anziché una collezione amorfa di gruppi
- Non si fonda su una reale e ben fondata teoria statistica o dell'informazione :-((

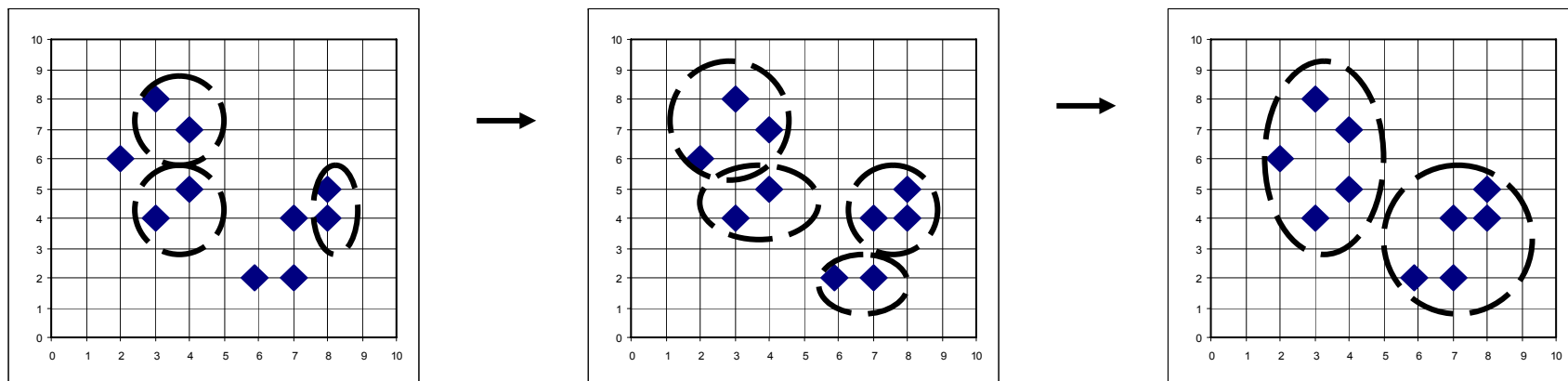
# Metodi gerarchici

- Matrice di distanze
- Non richiedono di conoscere a priori il numero di cluster
- Occorre una condizione di terminazione



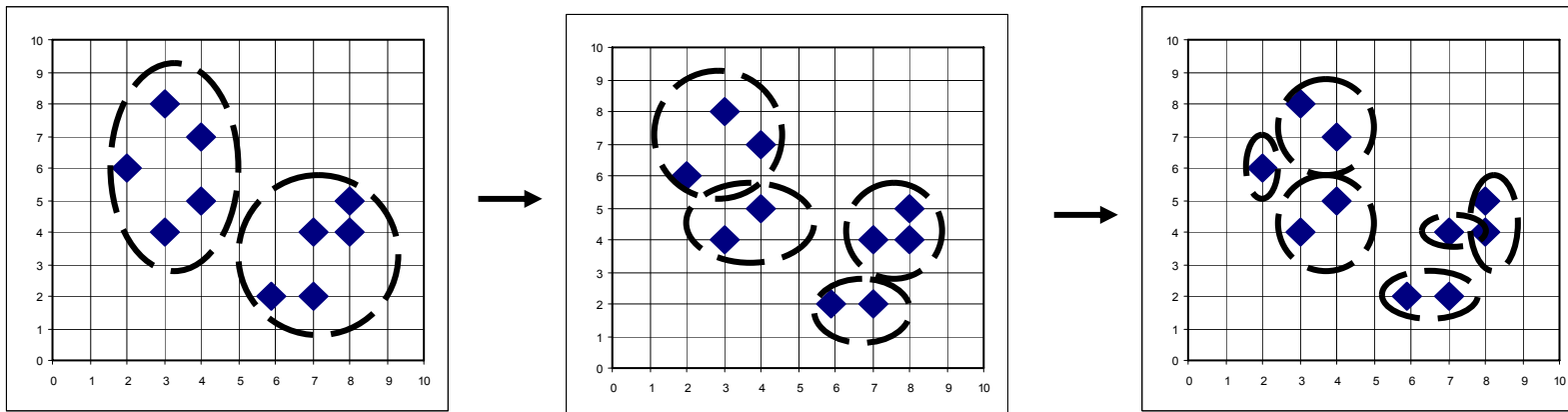
# AGNES (Agglomerative Nesting)

- Introdotto in Kaufmann e Rousseeuw (1990)
- Implementato in pacchetti di analisi statistica, es.: Splus
- Usa il metodo Single-Link e una matrice di dissimilarità
- Fonde nodi con la minima dissimilarità
- Può terminare raccogliendo tutti i nodi nello stesso cluster
- Approccio single-link
  - un cluster è rappresentato da tutti i suoi membri
  - al contrario di k-means, dove un cluster è rappresentato dal centroide



# DIANA (Divisive Analysis)

- Introdotto in Kaufmann e Rousseeuw (1990)
- Implementato in pacchetti statistici
- Ordine inverso di AGNES
- Può terminare con cluster di singoli oggetti



# Misure di distanza tra cluster

## ■ definizioni

- $|p-p'|$  = distanza tra due oggetti
- $m_i$  = media sul cluster  $C_i$
- $n_i$  = numero di oggetti in  $C_i$

## ■ distanze

- minima  $d_{\min}(C_i, C_j) = \min_{p \in C_i, p' \in C_j} |p-p'|$
- massima  $d_{\max}(C_i, C_j) = \max_{p \in C_i, p' \in C_j} |p-p'|$
- mean  $d_{\text{mean}}(C_i, C_j) = |m_i - m_j|$
- average  $d_{\text{avg}}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{p' \in C_j} |p - p'|$



# Commenti sul clustering gerarchico

- Principali punti di debolezza
  - non scala bene: complessità temporale almeno  $O(n^2)$  nel numero di oggetti
  - non può tornare sui propri passi, quindi eventuali decisioni di fusione/partizione infelici portano a un degrado di qualità
- Integrazione di clustering gerarchico con altri:  
clustering multifase
  - BIRCH (1996): usa un albero di clustering features (CF-tree) e aggiusta incrementalmente la qualità
  - CURE (1998): rappresenta ogni cluster con un numero prefissato di oggetti rappresentativi, quindi collassa gli oggetti verso il centro del cluster di un fattore predefinito
  - ROCK (1999): considera l'“interconnettività”
  - CHAMELEON (1999): usa la “modellazione dinamica”



# Birch (Zhang, Ramakrishnan, Livny, SIGMOD '96)

- *Balanced Iterative Reducing and Clustering using Hierarchies*
- Utilizza informazioni locali per decidere la divisione in cluster
- Minimizza
  - il costo di I/O
  - l'occupazione di memoria
- Gestisce gli outlier

# Birch (i)

- Clustering feature CF
- Informazioni per sintetizzare la rappresentazione dei cluster e migliorare scalabilità e efficienza
  - CF = (N, LS, SS) = momenti di ordine:

- zero - numero di oggetti  $= n_j$
- uno - somma lineare  $= \sum_{i=1}^{n_j} x_i$
- due - somma dei quadrati  $= \sum_{i=1}^{n_j} x_i^2$



# Birch (ii)

- Dalla CF è possibile calcolare:

□ centroide  $c_j = \frac{\sum_{i=1}^{n_j} x_i}{n_j}$

□ raggio del cluster  $R_j = \frac{\sum_{i=1}^{n_j} d(x_i, c_j)}{n_j}$

□ diametro del cluster  $D_j = \frac{\sum_{i=1}^{n_j} \sum_{l=1}^i d(x_i, x_l)}{n_j(n_j - 1)}$

# Birch (iii)

■ Date  $CF_j$  e  $CF_l$  è possibile calcolare:

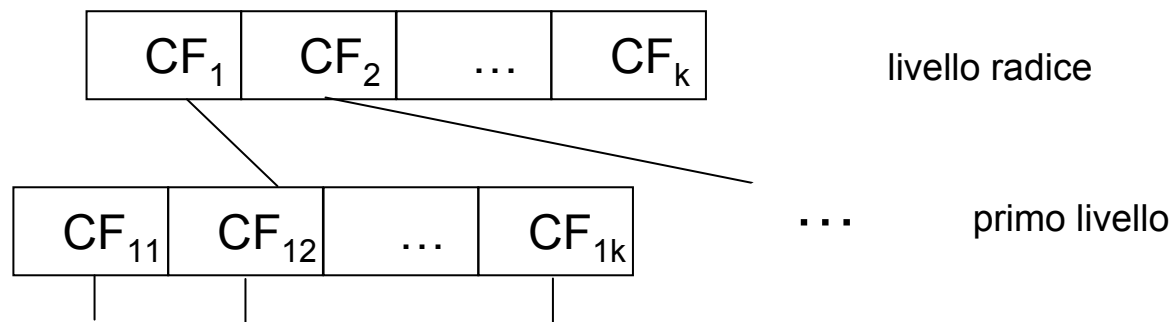
□ distanza fra i centroidi  $d(c_j, c_l)$

□ distanza inter-cluster  $D_{j,l} = \frac{\sum_{i \in \text{cluster } j} \sum_{i' \in \text{cluster } l} d(x_i, x_{i'})}{n_j n_l}$

□ distanza intra-cluster  $D'_{j,l} = \frac{\sum_{i \in \text{cluster } j,l} \sum_{i' \in \text{cluster } j,l} d(x_i, x_{i'})}{(n_j + n_l)^2}$

# Birch (iv)

- CF Tree = albero bilanciato che memorizza le CF dei cluster a ogni livello gerarchico
  - i nodi intermedi sommarizzano le informazioni dei nodi sottostanti
- Parametri:
  - branching factor = numero massimo di figli di un nodo intermedio
  - diametro soglia = massimo diametro dei sottocluster rappresentati da ciascuna foglia





# Birch (v)

- Fase 1 - accede al DB per costruire un CF-tree iniziale in memoria;
  - il CF-tree può essere visto come una compressione multilivello dei dati che tenta di preservare la struttura inerente dei cluster
  - un nuovo oggetto viene inserito dinamicamente nella foglia a cui risulta più vicino (usando una delle distanze precedenti)
  - l'aggiornamento delle CF risulta additivo
  - quando un nuovo oggetto fa superare il diametro soglia di una foglia → split
  - quando uno split fa superare il fattore di branching di un nodo intermedio → split
- Fase 2 - si applica un algoritmo di clustering ai nodi foglia



# Birch - discussione

- scala linearmente rispetto al numero di punti
- buona qualità dei cluster
- inadatto per forme non sferiche
  - dipende dal diametro dei sub-cluster
- inadatto a dati categorici



# Cure (Clustering Using REpresentatives)

- la maggior parte degli algoritmi di clustering
  - favoriscono forme sferiche e dimensioni uniformi
  - sono sensibili alla presenza di rumore
- Cure segue un approccio gerarchico agglomerativo e modifica l'approccio a “centroidi” con il concetto di “oggetti rappresentativi”
  - in ogni cluster vengono individuati i punti rappresentativi (in numero prefissato) selezionandoli in modo “ben sparso” sul cluster
  - i punti vengono poi “condensati verso il centro” di un fattore prefissato
  - ad ogni passo dell'algoritmo la fusione di due cluster viene decisa in base alla distanza dei punti rappresentativi più vicini



# Cure - osservazioni

- la scelta di più punti rappresentativi favorisce la trattazione corretta di cluster non sferici e di dimensioni variabili
- la condensazione verso il centro riduce l'effetto del rumore (outliers)
- $O(n)$  → scala facilmente all'aumentare del numero di punti, senza perdere in qualità
- richiede una sola scansione dell'intero DB
- è critica la scelta dei parametri
  - fattore di compattamento
  - numero di punti rappresentativi



# Rock

- agglomerativo gerarchico
- adatto a dati categorici
- compara l'*interconnettività aggregata* di due cluster
  - la similarità tra due cluster è valutata contando il numero di punti nei due cluster che hanno *vicini in comune (shared neighbors)*
  - costruisce un grafo a partire da una matrice di similarità, usando una soglia per l'individuazione dei vicini

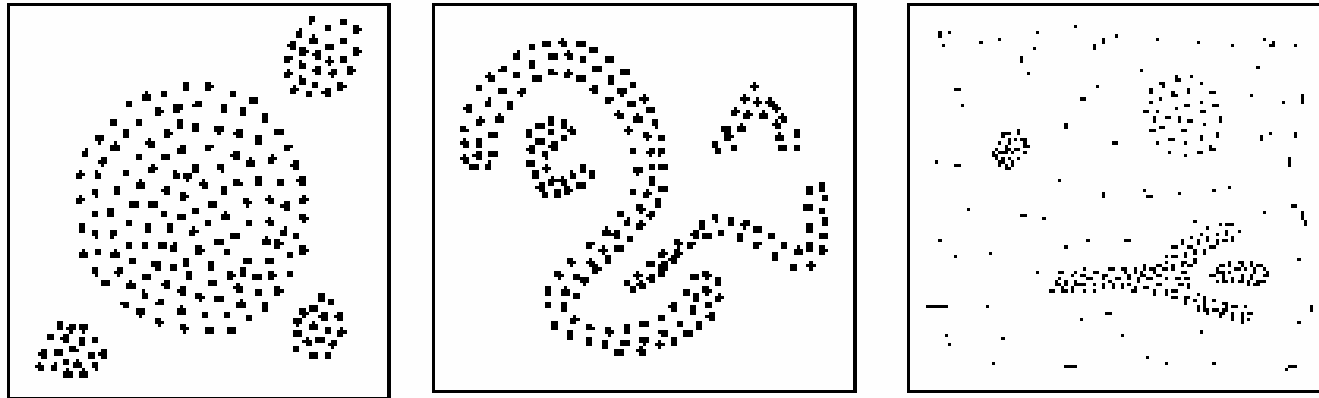




# Chameleon

- intende superare le debolezze di Cure e Rock
  - Cure trascura l'interconnettività tra oggetti di cluster diversi
  - Rock trascura la vicinanza tra cluster, enfatizzando l'interconnettività dei punti
- Chameleon
  - prima costruisce un elevato numero di piccoli cluster, con un algoritmo gerarchico partitivo
  - poi determina i cluster definitivi con un algoritmo agglomerativo
  - la fusione tiene conto sia dell'interconnettività dei punti che della distanza tra cluster
  - non dipende da un modello statico determinato dai parametri dell'utente, ma si adatta alle caratteristiche dei dati
  - è più efficace di Cure e DBScan
  - la complessità è  $O(n^2)$

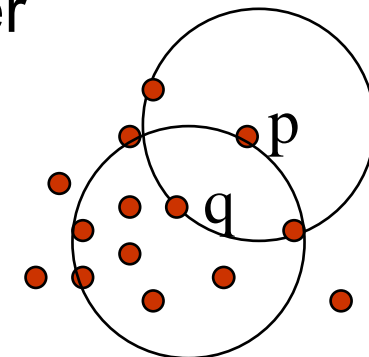
# Clustering basato sulla densità



- I cluster sono regioni ad alta densità separati da regioni a bassa densità

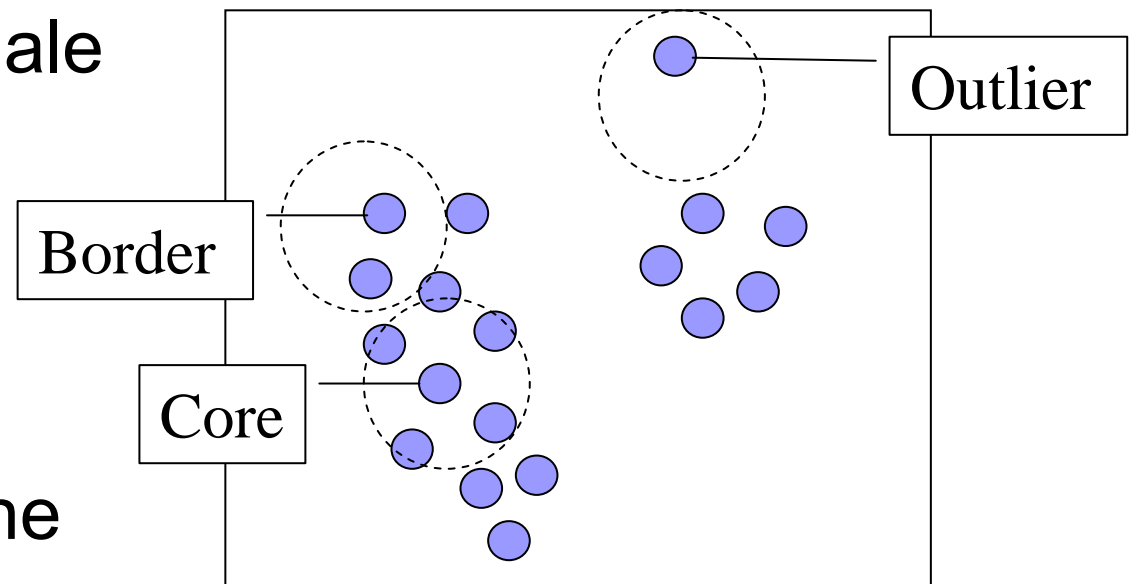
# Concetti relativi alla densità

- per ogni punto di un cluster, entro un dato “vicinato” deve esistere almeno un numero minimo di altri punti
  - densità soglia
  - la forma del “vicinato” dipende dalla funzione di distanza adottata
    - ad esempio, la “Manhattan distance” in genera un vicinato rettangolare
    - i ragionamenti successivi sono indipendenti dalla funzione distanza adottata
    - a titolo di esempio si userà la distanza Euclidea
- $\varepsilon$ -vicinato di  $p$  = insieme di punti a distanza minore di  $\varepsilon$  da  $p$
- *coreObject* = oggetto “all'interno” di un cluster
- *borderObject* = oggetto di confine, avrà meno punti nel suo  $\varepsilon$ -vicinato ma deve essere vicino ad un *coreObject*



# DBSCAN: Density Based Spatial Clustering of Applications with Noise

- si definisce cluster un insieme massimale di punti connessi dalla densità
- scopre cluster di forma arbitraria in DB spaziali, anche in presenza di rumore

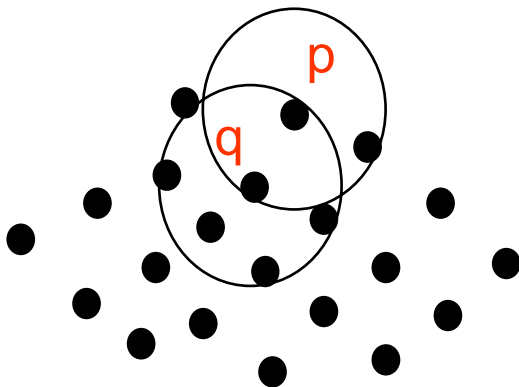


Eps = 1cm

MinPts = 5

# DBSCAN (ii)

- un punto  $p$  è *directly density-reachable* da un punto  $q$  rispetto a  $\varepsilon$  e a  $MinPts$  se:
  - $p \in N_\varepsilon(q)$
  - $|N_\varepsilon(q)| \geq MinPts$  (coreObject)
- la relazione è simmetrica per due coreObject, ma non per un core e un border

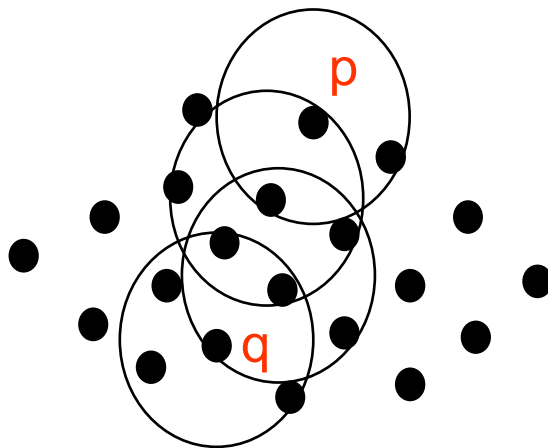


$p$  è direttamente raggiungibile da  $q$

$q$  non è direttamente raggiungibile da  $p$

# DBSCAN (iii)

- un punto  $p$  è *density-reachable* da un punto  $q$  rispetto a  $\varepsilon$  e a  $MinPts$  se:
  - c'è una catena di punti  $p_1, \dots, p_n$  con  $p_1=q, p_n=p$ ,  $p_{i+1}$  *directly density-reachable* da  $p_i$
  - due punti border possono non essere density-reachable, ma deve esserci un punto core rispetto al quale entrambi devono esserlo
- relazione transitiva e simmetrica per due coreObject

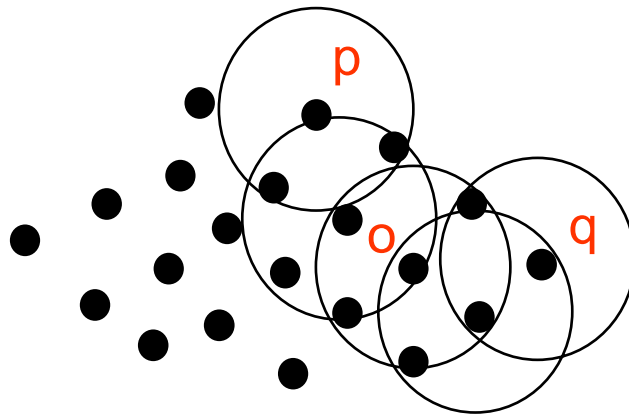


$p$  è raggiungibile da  $q$

$q$  non è raggiungibile da  $p$

## DBSCAN (iv)

- un punto  $p$  è *density-connected* a un punto  $q$  rispetto a  $\varepsilon$  e a  $MinPts$  se c'è un punto  $o$  tale che entrambi siano *density reachable* da  $o$
- la relazione è simmetrica e riflessiva per punti raggiungibili




$q$  e  $p$  sono connessi tramite  $o$

# DBSCAN (v)

- Un cluster è
  - un insieme di punti che siano density connected
  - massimale rispetto alla relazione di density reachability
- ovvero :
  - $\forall p, q$ , se  $p \in C$  e  $q$  è raggiungibile da  $p$ , allora  $q \in C$  (massimalità)
  - $\forall p, q \in C$ ,  $p$  è connesso a  $q$  (connettività)
- il rumore è costituito dai punti che non appartengono ad alcun cluster
  
- Sia  $p$  tale che  $||N_\varepsilon(p)|| \geq N_{\min}$ , allora l'insieme  $\{o \mid o \text{ è raggiungibile da } p\}$  è un cluster
- Sia  $C$  un cluster e sia  $p \in C$  tale che  $||N_\varepsilon(p)|| \geq N_{\min}$ , allora  $C$  coincide con l'insieme  $\{o \mid o \text{ è raggiungibile da } p\}$





# DBSCAN: algoritmo

```
DBSCAN(D,  $\epsilon$ ,  $N_{\min}$ ) {  
    Cld=next(Id);  
    for(i=0; i<|D|; i++) {  
        p=D[i];  
        if(p.C== $\emptyset$ )  
            if(expand(D,p,Cld, $\epsilon$ , $N_{\min}$ ))  
                Cld=next(Id);  
    }  
}
```

- p.C indica il cluster cui è stato assegnato il punto p
  - $\emptyset$  se non ancora assegnato
  - N se rumore

# DBSCAN: espansione di un cluster

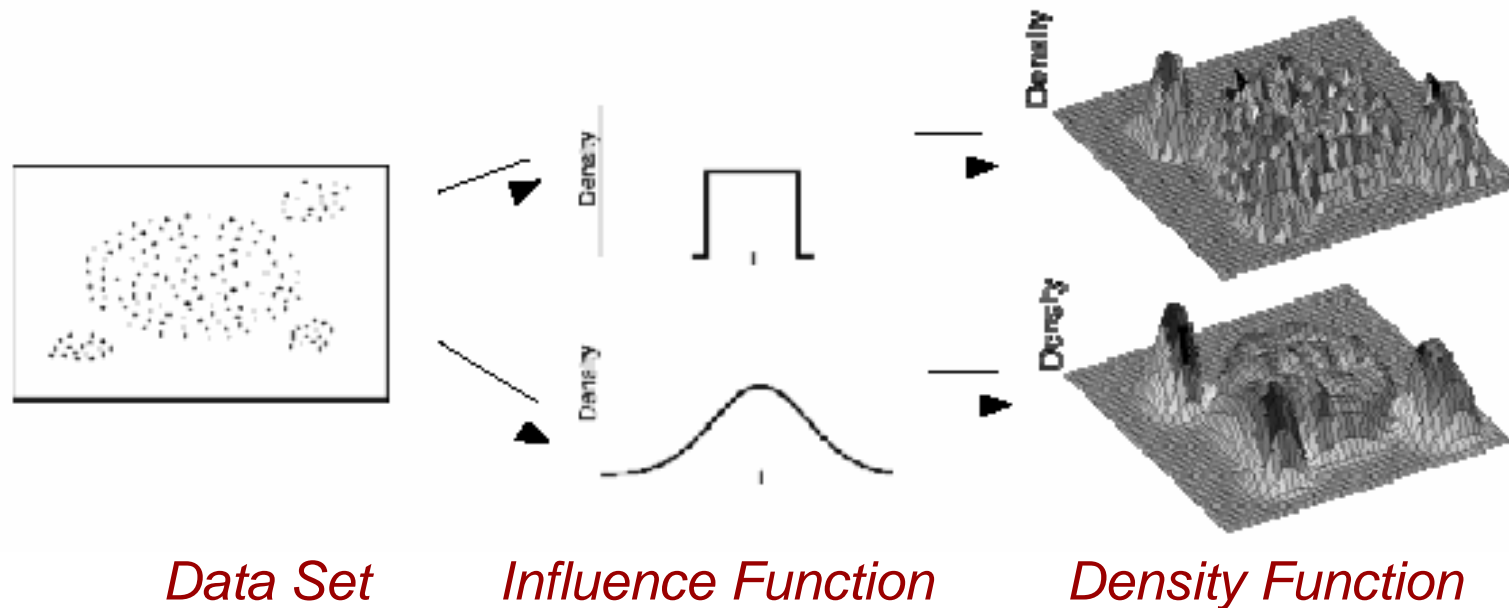
```
expand(D,p,CId, $\epsilon$ ,Nmin) {
    seeds=D.query(p, $\epsilon$ );
    if(|seeds|<Nmin) {
        p.C=N;
        return false; }
    for each q in seeds q.C=CId;
    seeds.Remove(p);
    while(|seeds|>0) {
        p'=seeds.RemoveHead();
        R=D.query(p', $\epsilon$ );
        if(|R| $\geq$ Nmin)
            while(|R|>0) {
                q'=R.RemoveHead();
                if(q'.C== $\emptyset$ ) {
                    seeds.Append(q');
                    q'.C=CId; }
                if(q'.C==N) q'.C=CId; }}
    return true; }
```



# DBSCAN - commenti

- a partire dagli oggetti core,  
l'algoritmo costruisce iterativamente i cluster
- se si dispone di indici spaziali,  
che limitano l'accesso ai soli indici  
del vicinato di un punto,  
la complessità di DBSCAN è  $O(n \log n)$
- in assenza di indici spaziali la complessità  
è  $O(n^2)$
- è sensibile ai parametri  $\varepsilon$  e MinPts

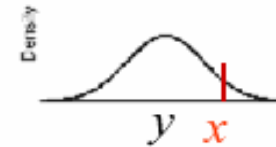
# Kernel Density Estimation



- *Influence Function:* influenza di un punto nel suo vicinato
- *Density Function:* somma delle influenze di tutti i punti
- *Density Attractor:* massimi locali della *density function*

# Kernel Density Estimation

e.g.,  $f_{Gauss}^y(x) = e^{-\frac{d(x,y)^2}{2\sigma^2}}$ .



## ■ Influence Function

- L'influenza è modellata da una funzione Kernel Density Estimation

## ■ *Density Function*

- La densità in un punto  $x$  è definita come la somma delle influenze di tutti i punti dati

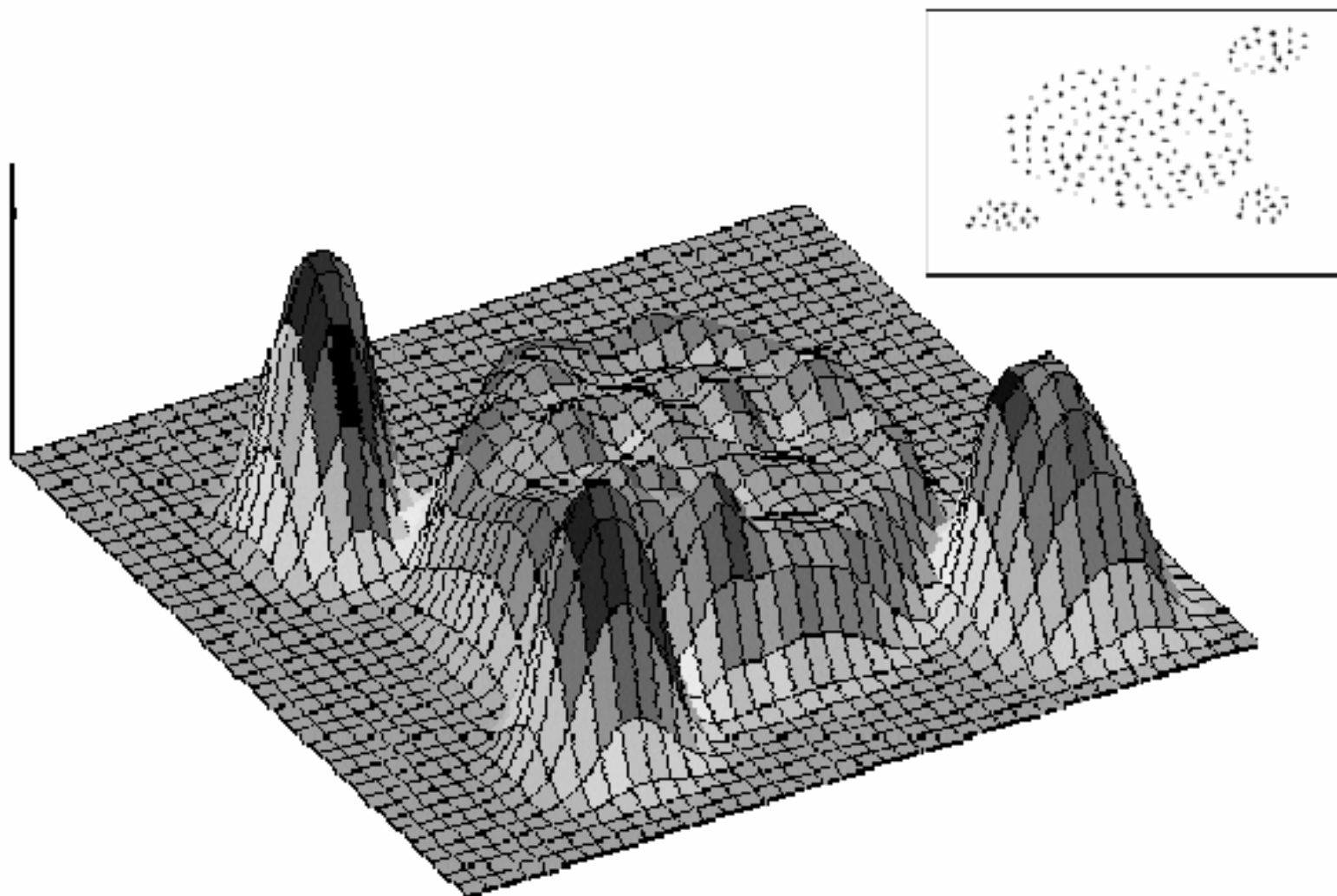
$$f_B^D(x) = \sum_{i=1}^N f_B^{x_i}(x)$$



# KDE

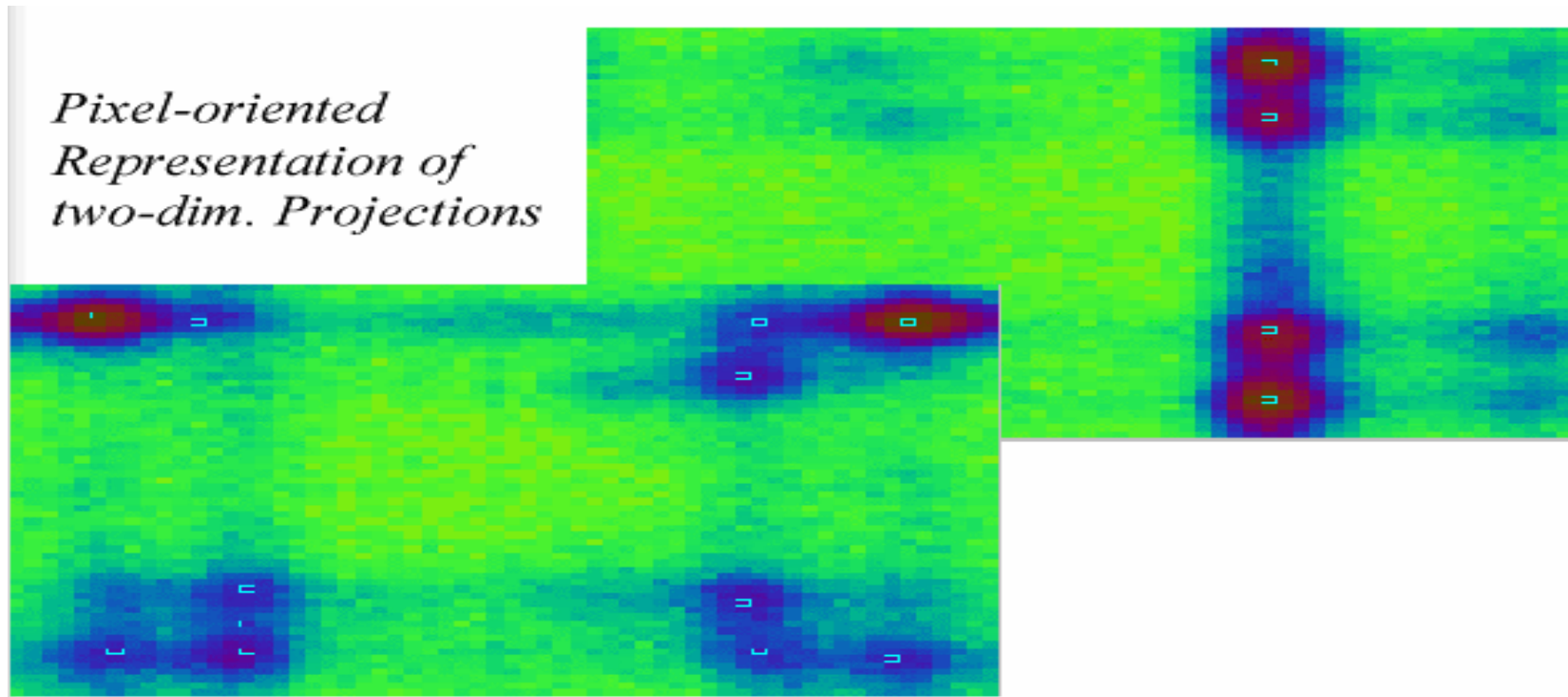
- cluster center-defined
  - dato un density attractor  $x$   
si individua un sottoinsieme di punti  
che è density-extracted intorno a lui,  
tale che la densità non sia inferiore a un dato  $\xi$
- cluster di forma arbitraria
  - è un insieme di cluster center-defined,  
tale che esista un percorso da una regione all'altra  
in cui la funzione densità non scende mai sotto  
il valore  $\xi$
- si comporta molto bene in presenza di rumore
- è sensibile alla scelta dei parametri

# KDE e clustering visuale



Clustering

# KDE e clustering visuale (ii)







# Scalabilità dei metodi di clustering

- L'efficacia degenera
  - con la dimensionalità  $d$
  - con il livello di rumore
  
- L'efficienza degenera
  - (almeno) linearmente con il numero di punti
  - esponenzialmente con la dimensionalità  $d$