Clustering II

- **Agglomerative hierarchical clustering**: iteratively merging the closest pair of points/clusters
  - Given: an n-by-n matrix D of all pairwise distances (e.g., Euclidean) between n data points
  - Let d_{ij} be the minimum of D, i.e., x_i and x_j are the two closest data points
  - Merge x_i and x_j into a new cluster x', compute distances of all other points to x' (see next slide), and compute a new (n-1)-by-(n-1) distance matrix D'
  - Iterate until only a single cluster is left
  - Output a dendrogram (see Slide 3)
- Advantage: no need to choose number of clusters in advance
  - can obtain any number of clusters from dendrogram
- Disadvantage: doesn't scale well
  - time complexity $O(n^3)$ to $O(n^2 \log n)$
- Clusters only apply to given data

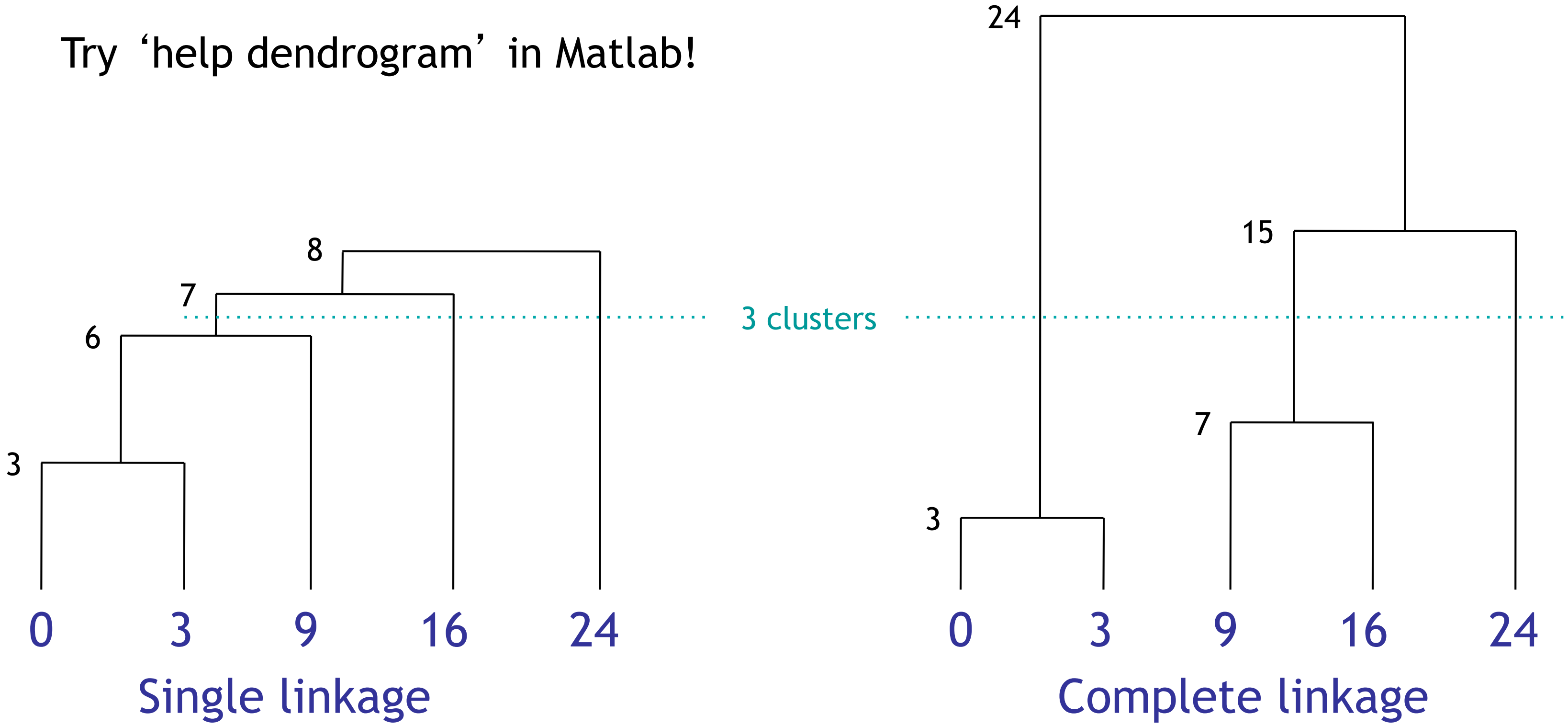
Linkage

- Distance between clusters can be calculated as:
  - the **minimum** distance between pairs from each cluster (**single linkage**)
  - the **maximum** distance between pairs from each cluster (**complete linkage**)
  - the **average** distance between pairs from each cluster (**average linkage**)
  - the distance between the centroids of each cluster (**centroid linkage**)
- Try 'help linkage' in Matlab!

Dendrogram

- Tree where each internal node corresponds to a pair of clusters merged in an iteration
  - the height of each node indicates distance between clusters
  - tree can be cut at desired number of clusters

Try ‘help dendrogram’ in Matlab!



Silhouettes

- $a(\mathbf{x})$ is average distance to points in own cluster
- $b(\mathbf{x})$ is average distance to points in nearest cluster
- $s(\mathbf{x}) = b(\mathbf{x})-a(\mathbf{x})/\max(a(\mathbf{x}),b(\mathbf{x}))$; should be large
- Silhouette plots $s(\mathbf{x})$ for each $\mathbf{x}$, grouped by cluster

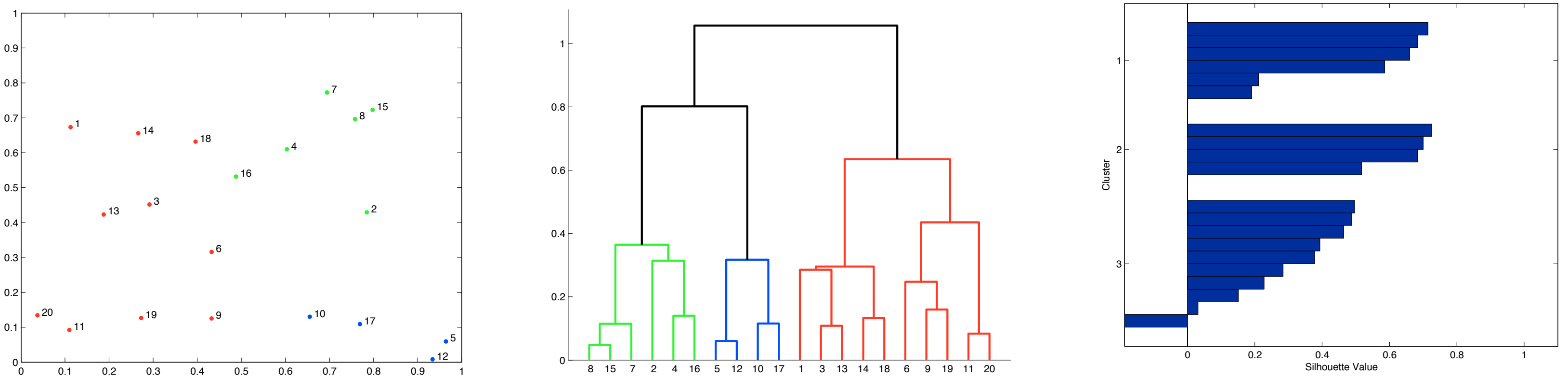


Figure 8.18. (left) 20 data points, generated by uniform random sampling. **(middle)** The dendrogram generated from complete linkage. The cluster structure suggested by the dendrogram is mostly spurious as it cannot be observed in the data. **(right)** The rapidly decreasing silhouette values in each cluster confirm the absence of a strong cluster structure. Point 18 has a negative silhouette value as it is on average closer to the green points than to the other red points.

Gaussian mixture models

- Approach to clustering where each cluster is modelled as a multivariate normal distribution with its own mean and covariance matrix
- Would be easy if we knew from which Gaussian each data point came, but then it would be a supervised classification problem
    - maximum-likelihood estimation of means and covariances
- New idea: treat cluster membership as continuous ***hidden variable***
    - *K*-means is special case: 0–1 cluster membership
    - solved by a very general algorithm called **Expectation-Maximisation (EM)** — here introduced by example only

Reminder: ML estimation

- Suppose *a* students got an A, *b* got a B, *c* got a C and *d* got a D (with *a*, *b*, *c*, *d* known). Suppose we also know that $P(\mathrm{A})=1/2$, $P(\mathrm{B})=\mu$, $P(\mathrm{C})=2\mu$, and thus $P(\mathrm{D})=1/2-3\mu$. What is μ?
  - can be solved by maximum likelihood estimation:

$P(a,b,c,d \mid \mu) \propto (1/2)^a \mu^b (2\mu)^c (1/2-3\mu)^d$, hence

$$\log P(a,b,c,d \mid \mu) = l + a\log 1/2 + b\log\mu + c\log 2\mu + d\log(1/2-3\mu)$$

Taking the derivative wrt. μ and setting to 0 yields

$\dfrac{b}{\mu} + \dfrac{2c}{2\mu} - \dfrac{3d}{1/2-3\mu} = 0$, which gives $\mu = \dfrac{b+c}{6(b+c+d)}$

A	B	C	D
15	5	10	10

→ $\mu=1/10$

Example with missing information

- Suppose ***h*****=*****a*****+*****b* students got an A or a B**, *c* got a C and *d* got a D (with *h*, *c*, *d* known). Suppose we also know that *P*(A)=1/2, *P*(B)=μ, *P*(C)=2μ, and thus *P*(D)=1/2–3μ. What is μ?
  - if we knew μ (which we do not), we could compute the **expected** value of *a* and *b*:

$$\left.\begin{array}{r}\frac{a}{b}=\frac{1/2}{\mu}\\ a+b=h\end{array}\right\} a=\frac{1/2}{1/2+\mu}h \text{ and } b=\frac{\mu}{1/2+\mu}h$$

  - if we knew the expected value of *a* and *b* (which we do not), we could compute the **maximum** likelihood estimate of μ (see previous slide)
- So: let's iterate **Expectation** and **Maximisation** -> EM algorithm

EM calculations

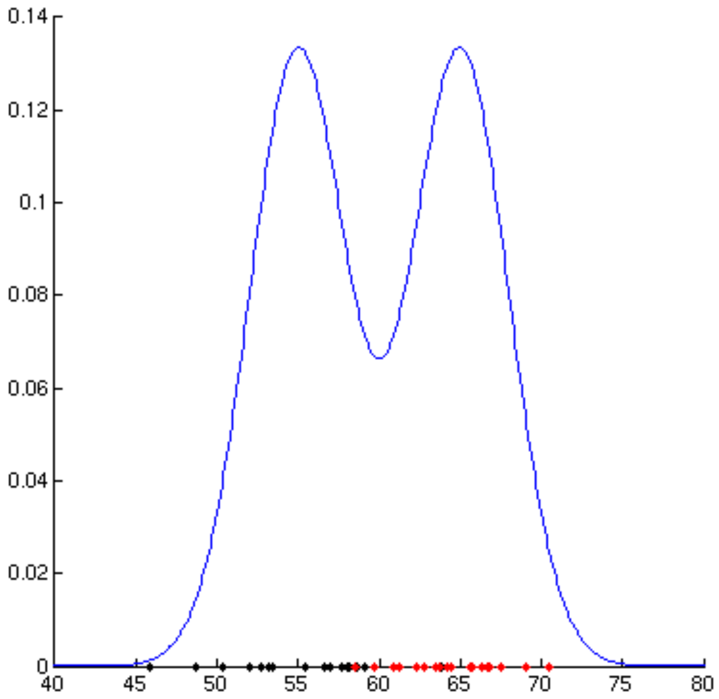
- Define
  - $\mu(t)$: estimate of μ after the tth iteration
  - $b(t)$: estimate of b after the tth iteration
- E-step: $b(t) = E[b \mid \mu(t)] = \dfrac{\mu(t)}{1/2 + \mu(t)} h$
- M-step: $\mu(t+1) = \arg\max_{\mu} P(a(t), b(t), c, d \mid \mu) = \dfrac{b(t) + c}{6(b(t) + c + d)}$

Two example calculations with h=20, c=d=10 and different initial guesses for $\mu(0)$:

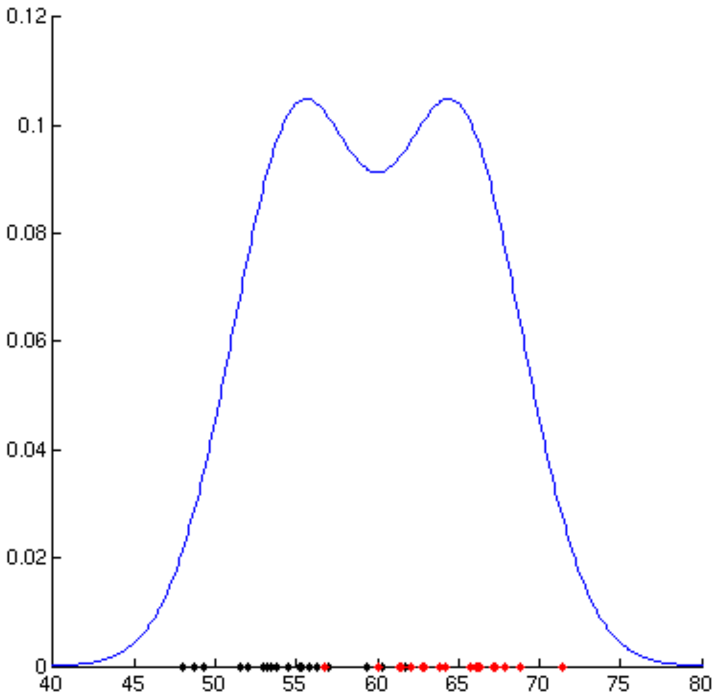
t	$\mu(t)$	$b(t)$
0	0	0
1	0.0833	2.857
2	0.0937	3.158
3	0.0947	3.185
4	0.0948	3.187
5	0.0948	3.187
6	0.0948	3.187

t	$\mu(t)$	$b(t)$
0	1/6	5
1	0.1	3.333
2	0.0952	3.2
3	0.0944	3.177
4	0.0948	3.187
5	0.0948	3.187
6	0.0948	3.187

1-D Gaussian mixtures



- μ_1=55, μ_2=65, σ_1=σ_2=3
- some overlap between clusters



- μ_1=55, μ_2=65, σ_1=σ_2=4
- more overlap between clusters

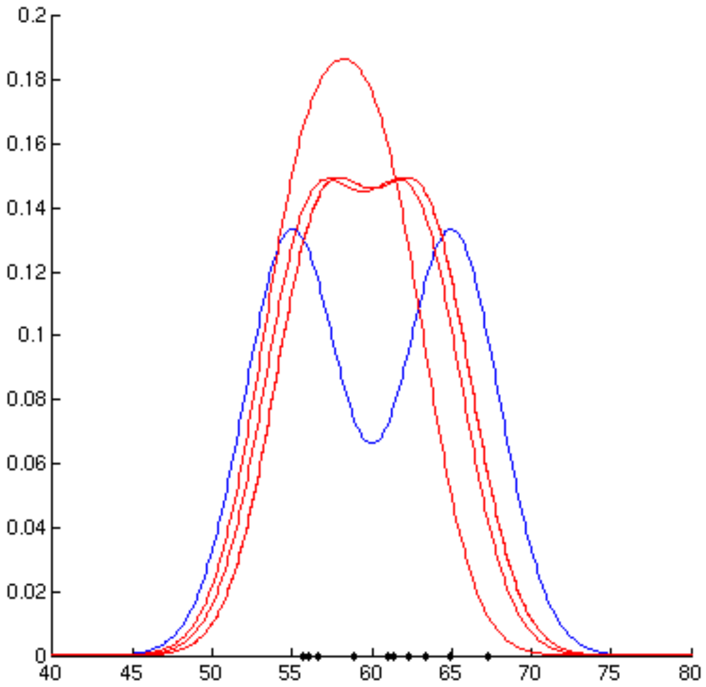
EM for 1-D Gaussian mixtures

- Given: data x_i ($1 \le i \le n$) drawn from K normal distributions with unknown means and equal variance
  - means that variance doesn't influence the outcome and can be set to 1
- Obtain: estimates of the means $\mu_1 \; \dots \; \mu_K$
- Approach: introduce **hidden variables z_{ij}** indicating the likelihood that x_i came from the j-th Gaussian
- Algorithm: Expectation-Maximisation!
  - E-step: for each data point x_i and each j

    $$z_{ij}(t) = E[z_{ij} \mid x_i, \mu_j(t)] \propto e^{-(x_i - \mu_j(t))^2/2}, \text{ normalised such that } \sum_{j=1}^{K} z_{ij}(t) = 1$$
  - M-step: for each j, estimate mean as weighted average

    $$\mu_j(t+1) = \arg\max_{\mu} p(x_1 \dots x_n, z_{1j} \dots z_{nj} \mid \mu) = \dots = \sum_{i=1}^{n} z_{ij}(t) x_i \Big/ \sum_{i=1}^{n} z_{ij}(t)$$

1-D Gmm example



x_i	55.6951	56.0631	56.5929	58.8639	61.0000	61.4035	62.2644	63.3310	64.9595	67.2668		
z_{i1}	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	μ_1	40
z_{i2}	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	μ_2	70
z_{i1}	1.0000	1.0000	0.9997	0.0372	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	μ_1	55.6951
z_{i2}	0.0000	0.0000	0.0003	0.9628	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	μ_2	60.7440
z_{i1}	1.0000	1.0000	1.0000	0.9794	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	μ_1	56.1507
z_{i2}	0.0000	0.0000	0.0000	0.0206	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	μ_2	62.7474
z_{i1}	1.0000	1.0000	1.0000	0.9996	0.0023	0.0002	0.0000	0.0000	0.0000	0.0000	μ_1	56.7931
z_{i2}	0.0000	0.0000	0.0000	0.0004	0.9977	0.9998	1.0000	1.0000	1.0000	1.0000	μ_2	63.3554
z_{i1}	1.0000	1.0000	1.0000	0.9997	0.0025	0.0002	0.0000	0.0000	0.0000	0.0000	μ_1	56.8062
z_{i2}	0.0000	0.0000	0.0000	0.0003	0.9975	0.9998	1.0000	1.0000	1.0000	1.0000	μ_2	63.3716

Discussion

- If the Gaussians have equal variance, Gaussian mixture models are very similar to *K*-means
  - main difference: ‘soft’ cluster membership
- But GMMs are more general: can also estimate covariances
- The usual caveats apply: local maxima, dependence on initial configuration, etc.
- Expectation-Maximisation is a very general technique for estimating hidden variables