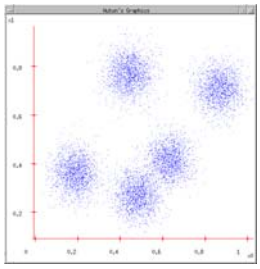
Data Structures for Fast Mixture Models

The Auton Lab

Carnegie Mellon University

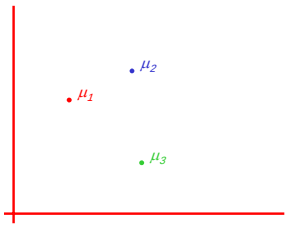
www.autonlab.org

What if we want to do density estimation with multimodal or clumpy data?



The GMM assumption

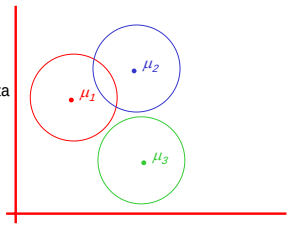
- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i



The GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 \mathbf{I}$

Assume that each datapoint is generated according to the following recipe:

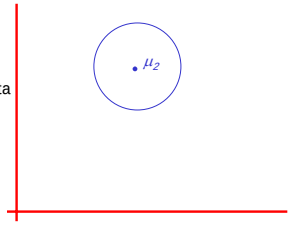


The GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 \mathbf{I}$

Assume that each datapoint is generated according to the following recipe:

1. Pick a component at random. Choose component i with probability $P(\omega_i)$.

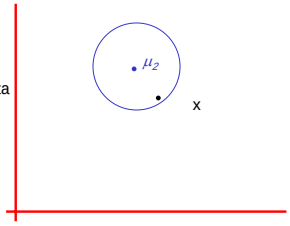


The GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 \mathbf{I}$

Assume that each datapoint is generated according to the following recipe:

1. Pick a component at random. Choose component i with probability $P(\omega_i)$.
2. Datapoint ~ $N(\mu_i, \sigma^2 \mathbf{I})$

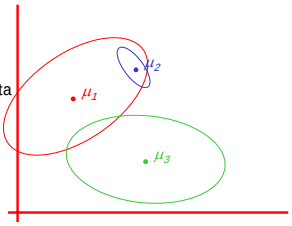


The General GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix Σ_i

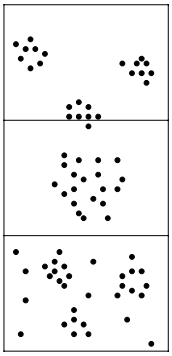
Assume that each datapoint is generated according to the following recipe:

1. Pick a component at random. Choose component i with probability $P(\omega_i)$.
2. Datapoint ~ $N(\mu_i, \Sigma_i)$



7

Unsupervised Learning: not as hard as it looks



Sometimes easy

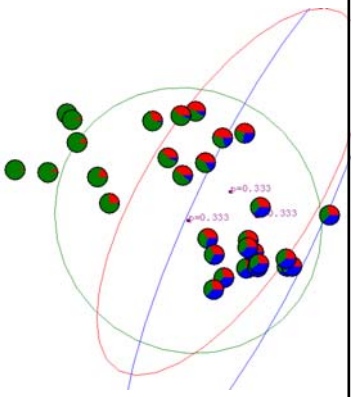
Sometimes impossible

and sometimes in between

*IN CASE YOU'RE WONDERING WHAT THESE DIAGRAMS ARE, THEY SHOW 2-d UNLABELED DATA (**X** VECTORS) DISTRIBUTED IN 2-d SPACE. THE TOP ONE HAS THREE VERY CLEAR GAUSSIAN CENTERS*

8

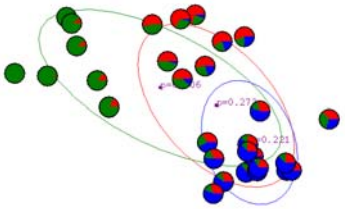
Gaussian Mixture Example: Start



Advance apologies: in Black and White this example will be incomprehensible

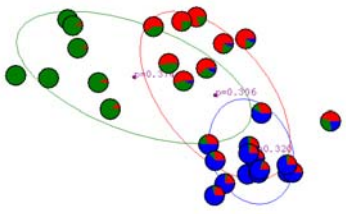
9

After first iteration



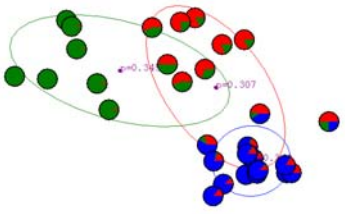
10

After 2nd iteration



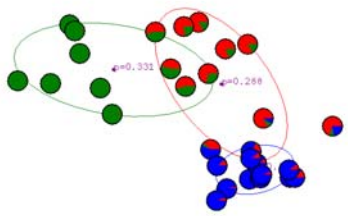
11

After 3rd iteration

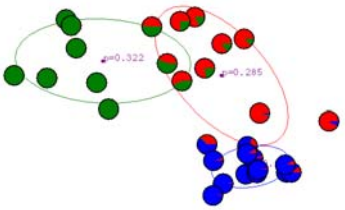


12

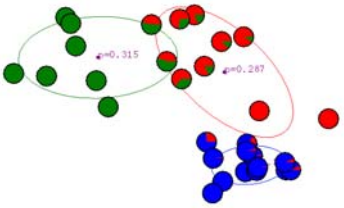
After 4th iteration



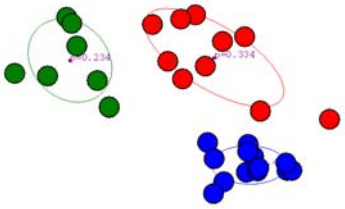
After 5th iteration



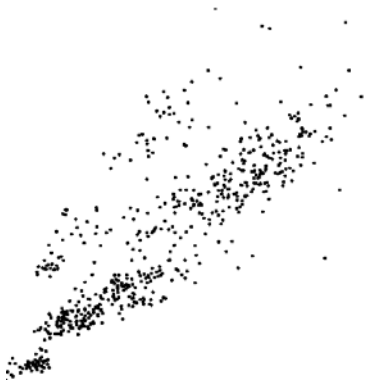
After 6th iteration



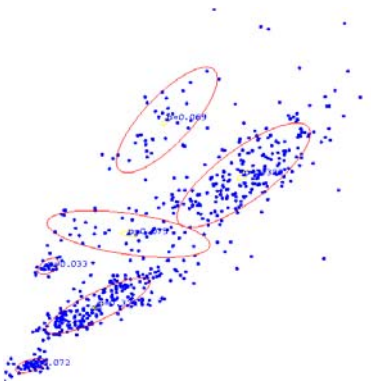
After 20th iteration



Some Bio Assay data



GMM clustering of the assay data



Resulting Density Estimator

19

The MRKD-tree approach

Kdtrees are a multidimensional data structure for representing sets of points in k-dimensional space. Invented in the early 70's by Jon Bentley.

MRKD-trees are Kd-trees containining extra information (cached sufficient statistics) including covariance matrices at every node in order to speed up statistical queries. [Deng and Moore, 1995].

With MRKD-trees we have been able to speed up a wide range of multidimensional statistics and machine learning algorithms [Deng and Moore 95, Moore Schneider and Deng 97, Moore 1998, Pelleg and Moore 1999]

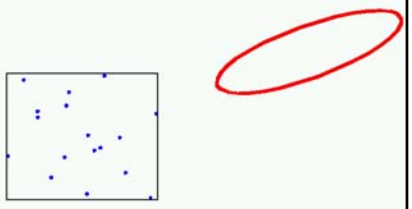
Turns out to be very useful for clustering. Typically, instead of visiting millions of datapoints at each iteration, simply visit a few thousand tree nodes. Some quadratic programming and algebraic bounds on Bayes Rule guarantees correct performance.

20

Computing likelihood of datapoints...

Suppose you want to compute the sum of log-likelihoods of all the blue dots given they'd been generated by the big red Gaussian.

Computing likelihood of datapoints...



Suppose we happen to know their bounding box

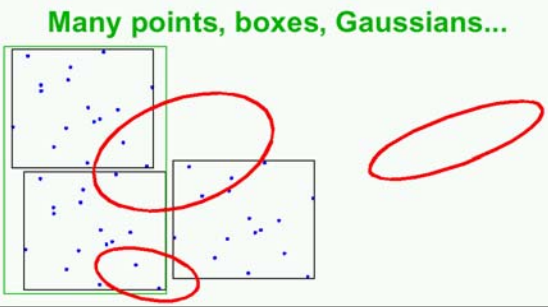
Computing likelihood of datapoints...

Greatest log-likelihood if they're all here

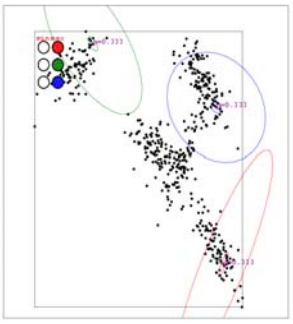
Least log-likelihood if they're all here

Without visiting the points individually, we can put bounds on their contributions to the Gaussian. Sometimes those bounds'll be tight enough...

Many points, boxes, Gaussians...

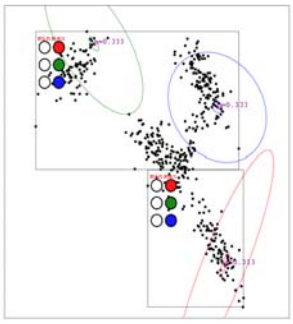


MRKD at the top node



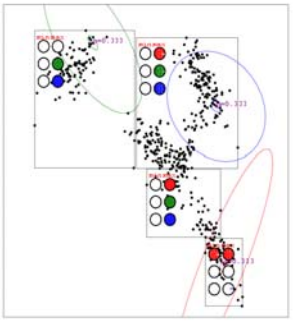
25

MRKD at level one nodes



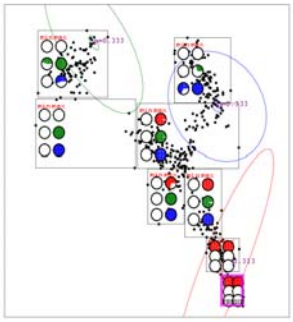
26

MRKD at level two nodes



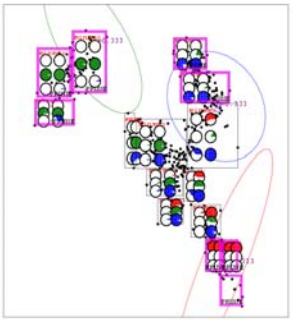
27

MRKD at level three nodes



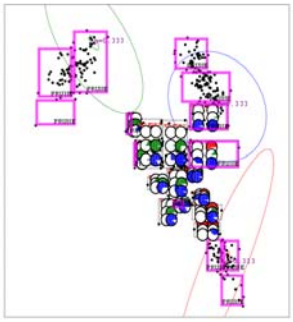
28

MRKD at level four nodes



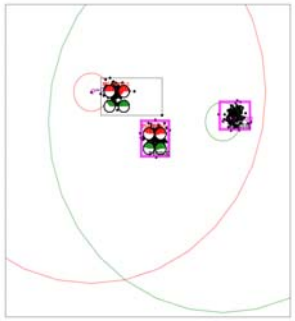
29

MRKD at level five nodes



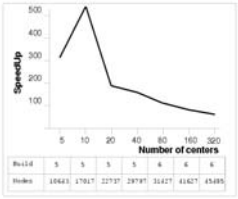
30

Another MRKD prune example



31

Effect of Number of Centers



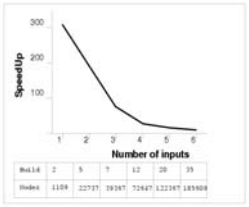
Build	5	5	5	5	6	6	6
Nodes	10643	17017	22737	29787	31427	41627	45495

When k=320: SlowSecs = 9278, FastSecs = 143.3, LLDiff = 0.004

Conventional EM slows down linearly with the number of classes. Fast EM is clearly sublinear, with a 70-fold speedup even with 320 classes. Note how the tree size grows. This is because more classes mean a more uniform data distribution and fewer datapoints "sharing" tree leaves.

32

Effect of Dimensionality



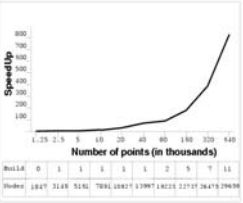
Build	2	5	7	12	20	35
Nodes	1104	22737	39167	72647	122367	185808

When M=6: SlowSecs = 2742, FastSecs = 310.25, LLDiff = 0.08

As with many kd-tree algorithms, the benefits decline as dimensionality increases, yet even in 6 dimensions, there is an 8-fold advantage.

33

Effect of Number of Datapoints



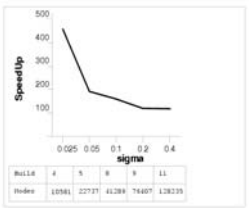
Build	0	1	1	1	1	2	5	7	11
Nodes	1847	3148	5182	7894	10827	13997	18215	22737	28475	29696

When R=640,000: SlowSecs = 2385, FastSecs = 3, LLDiff = 0.009

As R increases so does the computational advantage, essentially linearly. The tree-build time (11 seconds at worst) is a tiny cost compared with even just one iteration of Regular EM (2385 seconds, on the big dataset.)

34

Effect of Initial Gaussian Widths



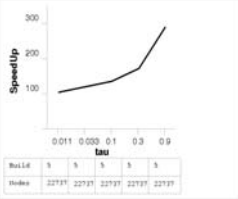
Build	4	5	8	9	11
Nodes	10581	22737	41289	76407	128235

When SigmaStart = 0.4: SlowSecs = 583, FastSecs = 4.75, LLDiff = 0.003

Even with very wide Gaussians, with wide support, we still get large savings. The nodes that are pruned in these cases are rarely nodes with one class owning all the probability, but instead are nodes where all classes have non-zero, but little varying, probability.

35

Effect of TAU (the willingness to prune)



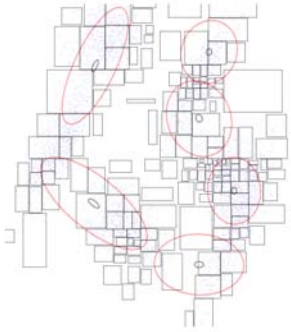
Build	5	5	5	5	5
Nodes	22737	22737	22737	22737	22737

When Tau=0.9: SlowSecs = 584.5, FastSecs = 2, LLDiff = 0.06

The larger tau, the more willing we are to prune during the tree search, and thus the faster we search, but the less accurately we mirror EM's statistical behavior.

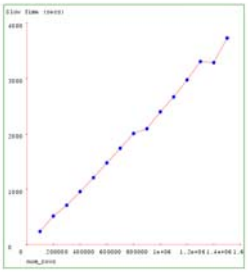
36

Nodes visited during an EM pass



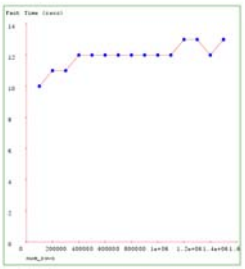
37

Sky Survey Data: Time Taken by the Slow method.



38

Sky Survey Data: Time Taken by the Fast method



39