STAT 535

THE EXPECTATION-MAXIMIZATION (EM) ALGORITHM

Basic Framework

Let D be the observable data.

Want to maximize log-likelihood: $\log f(D|\theta)$.

Let C be the complete data.

What does this mean? There is actually a probability model for $(C|\theta)$ [which induces the model for $(D|\theta)$]. But we don't get to see C, only observe $D = g(C)$, where $g()$ is 'many-to-one'.

Very commonly, $g(C_1, C_2) = C_1$ (i.e, C_2 is 'missing').

Also very commonly, $\log f(C|\theta)$ easy to maximize, $\log f(D|\theta)$ hard.

Take advantage of this structure?

Basic EM algorithm

Iterative sequence of estimates $\theta^{(1)}, \theta^{(2)}, \ldots$

$$\theta^{(k+1)} = \text{argmax}_\theta \; h(\theta; \theta^{(k)})$$

where

$$h(\theta; \theta^*) = E_{\theta^*} \{\log f(C|\theta)|D\}$$

Note: one iteration involves an E-step (evaluate expectation) followed by an M-step (maximize).

Intuition?

Example: Normal Mixture

$X_1, \ldots, X_n$ *iid* with density

$$p\phi(x|\mu_1, \sigma_1) + (1-p)\phi(x|\mu_2, \sigma_2),$$

a simple mixture model.

Digression: Two distinct rationales/scenarios for using mixture models.

Have $\theta = (\mu_1, \sigma_1, \mu_2, \sigma_2, p)$. No closed form ML estimator (log of sum complicates things).

What would make data 'complete'?

Knowledge of which components generated which observations!

Have $C = (X_i, J_i)_{i=1}^n$, but $D = (X_i)_{i=1}^n$.

Model for $C = (X, J)$ given θ?

$J \in \{1, 2\}$ (probs p and $1 - p$)

$X|J \sim N(\mu_J, \sigma_J^2)$

Clear that complete data model $(X, J|\theta)$ implies the desired model for observable data $(X|\theta)$.

E-Step

Boils down to computing

$$\begin{aligned} w_i &= E_{\theta^{(k)}}\left(I\{J_i = 1\}|X_i = x_i\right) \\ &= Pr_{\theta^{(k)}}(J_i = 1|X_i = x_i) \\ &= \frac{p^{(k)}\phi(x_i|\mu_1^{(k)}, \sigma_1^{(k)})}{p^{(k)}\phi(x_i|\mu_1^{(k)}, \sigma_1^{(k)}) + (1 - p^{(k)})\phi(x_i|\mu_2^{(k)}, \sigma_2^{(k)})} \end{aligned}$$

the 'weight' or 'responsibility' of component 1 for the i-th datapoint, assuming $\theta = \theta^{(k)}$.

Just Bayes theorem!

M-step

Standard (weighted) normal-linear 'stuff':

$$\begin{aligned} \mu_1^{(j+1)} &\leftarrow \frac{\sum_i w_i x_i}{\sum_i w_i} \\ \mu_2^{(j+1)} &\leftarrow \frac{\sum_i (1 - w_i) x_i}{\sum_i (1 - w_i)} \\ \sigma_1^{(j+1)} &\leftarrow \left\{\frac{\sum_i w_i (x_i - \mu_1)^2}{\sum_i w_i}\right\}^{1/2} \\ \sigma_2^{(j+1)} &\leftarrow \left\{\frac{\sum_i (1 - w_i)(x_i - \mu_2)^2}{\sum_i (1 - w_i)}\right\}^{1/2} \end{aligned}$$

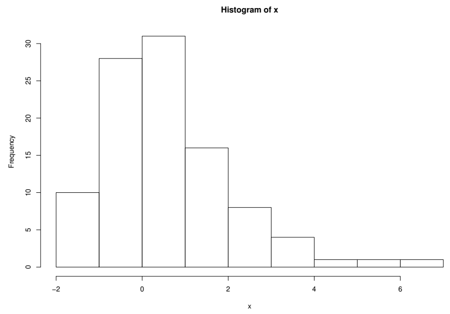
Easy to code!

```
normmix_function(x, numits=50) {
### starting vals
mu1 <- mean(x)-sqrt(var(x)); mu2 <- mean(x)+sqrt(var(x))
sig1 <- sig2 <- sqrt(var(x)/2); p <- 0.5

for (mnlp in 1:numits) {
  ### E-step
  wht <- p*dnorm(x, mu1, sig1) /
      (p*dnorm(x, mu1, sig1)+(1-p)*dnorm(x, mu2, sig2))
  ### M-step
  mu1 <- sum(wht*x)/sum(wht); mu2 <- sum((1-wht)*x)/sum(1-wht)
  sig1 <- sqrt( sum(wht*((x-mu1)^2))/sum(wht) )
  sig2 <- sqrt( sum((1-wht)*((x-mu2)^2))/sum((1-wht)) )
  p <- mean(wht) }
}
```

Try it out!

```
> x <- c( rnorm(75), rnorm(25, 2, 2) )
> hist(x)
```



First 10 iterations

mu1	sig1	mu2	sig2	p	loglik
-0.87	1.05	2.10	1.05	0.50	-197.07
-0.30	0.73	1.75	1.38	0.55	-173.88
-0.23	0.74	1.67	1.49	0.56	-172.83
-0.18	0.76	1.63	1.54	0.56	-172.40
-0.14	0.78	1.62	1.58	0.57	-172.17
-0.11	0.79	1.62	1.61	0.58	-172.03
-0.09	0.80	1.62	1.62	0.59	-171.93
-0.08	0.81	1.64	1.64	0.60	-171.86
-0.06	0.82	1.66	1.65	0.61	-171.79
-0.05	0.83	1.67	1.65	0.62	-171.74

Iterations 41-50, and 100

0.05	0.89	2.06	1.68	0.72	-171.39
0.05	0.89	2.06	1.68	0.72	-171.39
0.05	0.89	2.07	1.68	0.72	-171.39
0.05	0.89	2.07	1.68	0.72	-171.39
0.06	0.90	2.07	1.68	0.72	-171.39
0.06	0.90	2.08	1.68	0.72	-171.39
0.06	0.90	2.08	1.68	0.73	-171.39
0.06	0.90	2.08	1.68	0.73	-171.38
0.06	0.90	2.09	1.68	0.73	-171.38
0.06	0.90	2.09	1.68	0.73	-171.38

0.07	0.90	2.13	1.68	0.74	-171.38

What Else to Discuss?

Why does this work?

What other applications does it have?