STAT40380/STAT40390/STAT40850 Bayesian Analysis

Dr Niamh Russell

School of Mathematics and Statistics
University College Dublin

niamh.russell@ucd.ie

March 2016

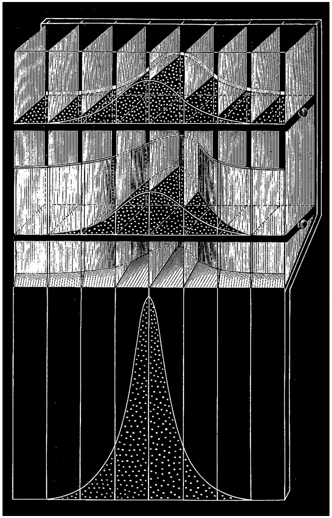
The EM algorithm in a nutshell

- The *Expectation-Maximisation* algorithm is a method for finding the posterior mode of a parameter
- It works iteratively to produce increasing values of $p(\theta^{(t)}|\mathbf{x})$
- The key part of the EM algorithm is that the data are *augmented* with extra hypothetical observations which make our posterior distributions simpler to work with
- The two steps involved in updating the parameters are:
    1. The E-step where we calculate the expected value of the posterior distribution with respect to the augmented data
    2. The M-step where we maximise this value to produce a new estimate of the parameter
- The EM algorithm is very popular for certain types of problem and there are many extensions of it

Other simple methods to estimate posterior distributions and posterior modes

- Before moving on to advanced Monte-Carlo methods, we will look at some simple techniques to estimate the posterior distribution when we do not know the normalising constant
- We will look at more methods to estimate the posterior mode, including:
  - Conditional maximisation
  - Newton-Raphson
  - Numerical derivatives
- and some simple methods to estimate the whole posterior distribution:
  - Parametric grids
  - Rejection sampling
- They are all applicable to situations where we can only observe the posterior distribution up to the constant of proportionality

Galton’s Bayesian Machine



Setup

- We will write $q(\theta|mathbfx)$ as the posterior density known up to a constant so that $p(\theta|\mathbf{x}) = q(\theta|\mathbf{x})/g(\mathbf{x})$
- Our task is to produce samples from the *joint posterior distribution*
- Some of these techniques we cover today require us to be able to simulate from known distributions such as the normal, uniform, Poisson etc
- This is very easy to do in R, Matlab, etc. WinBUGS simulates from distributions automatically without us needing to give specific commands
- Bizarrely, these techniques tend to be most useful on the *simplest* (ie not particularly interesting) or *most complex* (ie very hard to calculate) problems!

Posterior modes 1: conditional maximisation

- Write $q(\boldsymbol{\theta}|\mathbf{x}) = q(\theta_1, \ldots, \theta_K|\mathbf{x})$, ie we have K parameters
- Steps
  1. Guess at initial values of parameters
  2. Loop through each parameter k in turn, keeping the others fixed
  3. Find $\hat{theta}_k$ such that $q(\theta_k|\theta_1, \ldots, \theta_{k-1}, \theta_{k+1}, \ldots, \theta_K, \mathbf{x})$ is maximised
  4. Keep iterating until $q()$ reaches a local maximum
  5. Report values of $\hat{\theta}_1, \ldots, \hat{\theta}_K$
- Some remarks:
  - The optimisation could be performed analytically by computing the derivative of $q()$ with respect to θ_k, or by using numerical optimisation (such as `nlminb` in R)
  - To avoid getting stuck in local maxima, we could run the routine many times using different starting values. It should be possible to estimate reasonable bounds for the parameters to come up with good variable starting values

Posterior modes 2: Newton-Raphson

- A maximisation method that estimates posterior mode from $L = \log q(\theta|\mathbf{x})$
- Method comes from a Taylor expansion and required first and second derivatives of log posterior. This is a vector and a matrix respectively when there are multiple parameters
- Steps
  1. Choose starting value(s), $\boldsymbol{\theta}^0$
  2. For $t = 1, 2, 3, \ldots$
     1. Compute first and second order derivatives $L'(\boldsymbol{\theta}^{t-1})$ and $L''(\boldsymbol{\theta}^{t-1})$
     2. Set
$$\boldsymbol{\theta}^t = \boldsymbol{\theta}^{t-1} - \left[L''(\boldsymbol{\theta}^{t-1})\right]^{-1} L'(\boldsymbol{\theta}^{t-1})$$
- Extremely fast convergence if the starting values are close to the solution 'item Starting values very important: $L''(\boldsymbol{\theta}^{t-1})$ may not be positive definite at certain points

Posterior modes 3: numerical computation of derivatives

- Useful when numerci computation of log q is difficult
- Set
$$L_i'(\boldsymbol{\theta}) = \frac{\partial L}{\partial \theta_i} \simeq \frac{L(\boldsymbol{\theta} + \delta_i \boldsymbol{\epsilon}_i) - L(\boldsymbol{\theta} - \delta_i \boldsymbol{\epsilon}_i)}{2\delta_i}$$
  where δ_i is a small value,$\boldsymbol{\epsilon}_i$ is a unit vector corresponding to the ith component of $\boldsymbol{\theta}$
- The value(s) of δ needs to be chosen depending on the scale of the parameter
- The numerical second derivative can be calculated similarly to the above to produce $\frac{\partial^2 L}{\partial \theta_i \partial \theta_j}$
- Need to be careful to avoid numerical instability

Full posterior methods 1: grid based methods

- If the number of parameters is small (approx < 6), then a grid-based method may be an efficient way to approximate the full posterior distribution
- Steps
  1. Produce a grid of G possible values, $\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, \ldots, \boldsymbol{\theta}^G$. These should cover the vast majority of possible parameter values
  2. Calculate
  $$p(\boldsymbol{\theta}|\mathbf{x}) \simeq \frac{q(\boldsymbol{\theta}|\mathbf{x})}{\sum_{i=1}^{G} q(\boldsymbol{\theta}^i|\mathbf{x})}$$
- relies on a distrete approximation to the posterior, so the density of the grid is important in the precision of the approximation
- Breaks down when there are many parameters because G gets prohibitively large

Full posterior methods 2: rejection sampling

- Idea: propose lots of different values, then reject those that do not 'match' the posterior
- To perform rejection sampling we need a probability distribution $h(\theta)$ with the following properties:
  1. We are able to draw random samples from $h()$
  2. The *importance ratio* $R = q(\theta|\mathbf{y})/h(\theta)$ must have an upper bound, ie $q(\theta|\mathbf{y})/h(\theta) \leq M$ for all θ
- Rejection sampling steps:
  1. Sample a set of proposal values θ^{prop} from the distribution $h(\theta)$
  2. With probability $q(\theta)|\mathbf{y})/Mh(\theta) = R/M$, accept θ^{prop} as a draw from q. If the draw is rejected, return to step (1)
- The second step can be performed by comparing a $U(0,1)$ draw with the importance ratio. If $U < R/M$ accept, otherwise reject

Rejection sampling: Notes

- The key to a good rejection sampling algorithm is that h is roughly proportioal to q
- If h is far from q we will reject many samples, and the algorithm will be vary inefficient
- If we can obtain $h \propto q$ with M very close to 1, we can accept nearly every single draw
- In many cases we will use the data $\mathbf{x}$ to pick a good h
- A good rejection sampling algorithm will often make use of the posterior modes

Example

Example

Suppose $x_i \sim Bin(10, \theta)$ and $\theta \sim Be(2, 2)$. We observe 7 observations with $\sum x_i = 25$

1. Use a grid-based method to produce the estimated posterior density
2. Use rejection sampling to obtain samples from the posterior distribution