13: Modal And Distributional Approximations

Taylor

University of Virginia

Introduction

We mention:

1. a few ways to find the posterior mode
2. how to approximate a posterior using a mode
3. slightly more involved ways to approximate your posterior

For various reasons, we also frequently split up our parameters into two groups: $\theta = (\gamma, \phi)$.

Newton's Method aka the Newton-Raphson algorithm

Based on a first-order approximation of the first derivative of the log-likelihood.

Approximate $L'(\theta) = (\log p(\theta \mid y))'$ as

$$\mathbf{0} \overset{\text{set}}{=} L'(\theta + \delta\theta) \approx L'(\theta) + L''(\theta)(\delta\theta)$$

rearranges to

$$\delta\theta = -[L''(\theta)]^{-1} L'(\theta)$$

Newton's Method

Repeat the following iteration until convergence:

$$\theta^t = \theta^{t-1} - [L''(\theta^{t-1})]^{-1} L'(\theta^{t-1})$$

Newton's Method aka the Newton-Raphson algorithm

Newton's Method

Repeat the following iteration until convergence:

$$\theta^t = \theta^{t-1} - [L''(\theta^{t-1})]^{-1} L'(\theta^{t-1})$$

Notes:

1. easily handles unnormalized densities
2. starting value is important because it is not guaranteed to converge from everywhere
3. The derivatives can be determined analytically or numerically

Quasi-Newton and conjugate gradient methods

Notes:

1. Quasi-Newton methods are available when second derivatives are too costly or unavailable
2. "Broyden-Fletcher-Goldfarb-Shanno" is a common example of a Quasi-Newton method
3. in `R`: `optim(2.9,F,method="BFGS")`
4. conjugate-gradient methods only use gradient information, but they are for models of the form $||A\theta - b||_2$ (also handled by `optim()`)

Quasi-Newton and conjugate gradient methods

In `optim`, if you don't provide a function to calculate the gradient, then it uses a finite-difference approximation:

$$L_i'(\theta) = \frac{dL}{d\theta_i} \approx \frac{L(\theta + \delta_i e_i) - L(\theta - \delta_i e_i)}{2\delta_i}$$

and

$$\begin{aligned} L_{ij}''(\theta) &= \frac{d^2 L}{d\theta_i d\theta_j} \\ &\approx \frac{L_i'(\theta + \delta_j e_j) - L_i'(\theta - \delta_j e_j)}{2\delta_j} \end{aligned}$$

where e_j is the vector of all zeros except for a 1 in the jth spot, and δ_j is a small number (`optim`'s default is $1e-3$)

Gaussian approximations

Once the mode or modes have been found (perhaps after including a boundary-avoiding prior distribution as discussed in section 13.2, or after transforming the parameters appropriately), we can construct an approximation based on the (multivariate) normal distribution.

Let $\hat{\theta}$ be the mode, then

$$p(\theta \mid y) \approx N(\hat{\theta}, V_\theta)$$

where

$$V_\theta) = \left[- \frac{d^2 \log p(\theta \mid y)}{d\theta^2} \bigg|_{\theta=\hat{\theta}} \right]^{-1}$$

is calculated exactly or approximated using the formula from a few slides ago.

Gaussian approximations: Laplace's Method

If you want approximations to posterior *expectations* (say $E[h(\theta) \mid y]$), then you might consider Laplace's method, which is based on a Taylor approximation of the function $u(\theta) = \log[h(\theta)p(\theta \mid y)]$ centered at its maximizing value θ_0:

$$
\begin{aligned}
u(\theta) &\approx u(\theta_0) + (\theta - \theta_0)^T u'(\theta_0) + \frac{1}{2}(\theta - \theta_0)^T u''(\theta_0)(\theta - \theta_0) \\
&= u(\theta_0) + \frac{1}{2}(\theta - \theta_0)^T u''(\theta_0)(\theta - \theta_0)
\end{aligned}
$$

It assumes h is twice continuously differentiable.

Gaussian approximations: Laplace's Method

$$
\begin{aligned}
u(\theta) &\approx u(\theta_0) + (\theta - \theta_0)^T u'(\theta_0) + \frac{1}{2}(\theta - \theta_0)^T u''(\theta_0)(\theta - \theta_0) \\
&= u(\theta_0) + \frac{1}{2}(\theta - \theta_0)^T u''(\theta_0)(\theta - \theta_0)
\end{aligned}
$$

So, if d is the dimension of θ,

$$
\begin{aligned}
E[h(\theta) \mid y] &= \int \exp\left[u(\theta)\right] \mathrm{d}\theta \\
&\approx \exp\left[u(\theta_0)\right] \int \exp\left[\frac{1}{2}(\theta - \theta_0)^T u''(\theta_0)(\theta - \theta_0)\right] \mathrm{d}\theta \\
&= h(\theta_0) p(\theta_0 \mid y) \int \exp\left[-\frac{1}{2}(\theta - \theta_0)^T \left[-u''(\theta_0)\right] (\theta - \theta_0)\right] \mathrm{d}\theta \\
&= h(\theta_0) p(\theta_0 \mid y)(2\pi)^{-d/2} \det\left(\left[-u''(\theta_0)\right]^{-1}\right)^{-1/2}
\end{aligned}
$$

Gaussian approximations

The book has a few more generalizations that we don't address:

1. using only the unnormalized posterior density
2. approximating multimodal distributions with normal mixtures
3. approximating multimodal distributions with t mixtures

The EM Algorithm

The **expectation-maximization algorithm** finds the argument that maximizes the marginal posterior. It's useful in situations where there is missing data in a model (e.g. factor models, hidden markov models, state space models, etc.).

It folows the following steps

1. replace missing values by their expectations given the guessed parameters,
2. estimate parameters assuming the missing data are equal to their estimated values,
3. re-estimate the missing values assuming the new parameter estimates are correct,
4. re-estimate parameters,

and so forth, iterating until convergence.

The EM Algorithm

Call $\theta = (\gamma, \phi)$. You're interested in the mode of $p(\phi \mid y)$.

$$\log p(\phi \mid y) = \log \frac{p(\gamma, \phi \mid y)}{p(\gamma \mid \phi, y)} = \log \underbrace{p(\gamma, \phi \mid y)}_{\text{joint posterior}} - \log \underbrace{p(\gamma \mid \phi, y)}_{\text{conditional posterior}}$$

The EM Algorithm

Call $\theta = (\gamma, \phi)$. You're interested in the mode of $p(\phi \mid y)$.

$$\log p(\phi \mid y) = \log \frac{p(\gamma, \phi \mid y)}{p(\gamma \mid \phi, y)} = \log \underbrace{p(\gamma, \phi \mid y)}_{\text{joint posterior}} - \log \underbrace{p(\gamma \mid \phi, y)}_{\text{conditional posterior}}$$

taking expectations on both sides with respect to $p(\gamma \mid \phi^{\text{old}}, y)$ yields:

$$\log p(\phi \mid y) = E\left[\log p(\gamma, \phi \mid y) \mid \phi^{\text{old}}, y\right] - E\left[\log p(\gamma \mid \phi, y) \mid \phi^{\text{old}}, y\right]$$

The EM Algorithm

We iteratively use the middle term in
$\log p(\phi \mid y) = E\left[\log p(\gamma, \phi \mid y) \mid \phi^{\text{old}}, y\right] - E\left[\log p(\gamma \mid \phi, y) \mid \phi^{\text{old}}, y\right]$.

The Q quantity in the "E" step

$$Q(\phi \mid \phi^{\text{old}}) = E\left[\log p(\gamma, \phi \mid y) \mid \phi^{\text{old}}, y\right]$$

The EM Algorithm

We iteratively use the middle term in
$\log p(\phi \mid y) = E\left[\log p(\gamma, \phi \mid y) \mid \phi^{\text{old}}, y\right] - E\left[\log p(\gamma \mid \phi, y) \mid \phi^{\text{old}}, y\right]$.

The Q quantity in the "E" step

$$Q(\phi \mid \phi^{\text{old}}) = E\left[\log p(\gamma, \phi \mid y) \mid \phi^{\text{old}}, y\right]$$

The EM algorithm

Repeat the following until convergence:

1. E-step: calculate $Q(\phi \mid \phi^{\text{old}})$
2. M-step: replace ϕ^{old} with $\arg\max Q(\phi \mid \phi^{\text{old}})$

The EM Algorithm

Clearly $\log p(\phi \mid y)$ increases at every iteration:

$$\begin{aligned}
\log p(\phi \mid y) &= E\left[\log p(\gamma, \phi \mid y) \mid \phi^{\text{old}}, y\right] - E\left[\log p(\gamma \mid \phi, y) \mid \phi^{\text{old}}, y\right] \\
&= Q(\phi \mid \phi^{\text{old}}) - \underbrace{E\left[\log p(\gamma \mid \phi, y) \mid \phi^{\text{old}}, y\right]}_{\text{don't change } \phi} \\
&= Q(\phi \mid \phi^{\text{old}}) + \text{constant}
\end{aligned}$$

The EM Algorithm

Notes:

1. The EM algo isn't inherently Bayesian. It can also be used to accomplish maximum likelihood estimation.
2. The expectation of $\log p(\gamma, \phi \mid y)$ is usually easy to compute because it is a sum, and might only depend on sufficient statistics
3. The EM algorithm implicitly deals with parameter constraints in the M-step
4. The EM algorithm is parmeterization independent
5. The *Generalized* EM (GEM) just increases Q instead of maximizing it.
6. The book describes many generalizations in addition to this one
7. You can find multiple modes if you start from multiple starting points (using mixture approximations afterwards)
8. Debug by printing $\log p(\phi^i \mid y)$ at every iteration and make sure it increases monotonically