Statistics 225
Bayesian Statistical Analysis (Part 3)

Hal Stern

Department of Statistics
University of California, Irvine
sternh@uci.edu

March 28, 2019

Computation
Introduction

- Goal: Posterior inference for parameters, missing data (if any), and predictions
- Thus far:
  - analytic results or exact simulation in small problems
  - normal approximation for large samples
  - grid approximation
  - use hierarchical structure (e.g., $\tau|y$, then $\mu|\tau, y$, then $\theta|y, \mu, \tau$ in hierarchical normal-normal model)
- Now consider additional tools:
  - numerical integration
  - simulation (including MCMC)
  - approximation (including optimization strategy)
- Some of this is usually covered in a statistical computing class

Computation

- An overall computation strategy
  - initial (perhaps crude) estimates of parameters
  - numerical integration or direct simulation when possible
  - if direct simulation is not possible
    - iterative simulation via MCMC algorithms (e.g., Gibbs sampler, Metropolis algorithm)
    - approximation strategies (modes, variational Bayes, EP)

Computation

Some helpful computing ideas / strategies

- Compute posterior distn on log scale (to avoid underflows or overflows)
- Factoring the posterior distribution (e.g., $p(\theta_1, \theta_2|y) = p(\theta_1|\theta_2, y)p(\theta_2|y)$)
  - reduce to easier, lower-dimensional problems
  - isolate the parameters most influenced by prior distribution (e.g., τ in 8 schools example)
  - difficulties:
    - can't generally find marginal distn easily
    - hard to use a grid with a high-dimensional marginal distn
- Transformations
  - create more understandable parameters
  - make prior independence plausible
  - improve normal approximation (e.g., log of scale parameter)
  - speed/simplify iterative simulation

Computation
Notation/Notes

- $p(\theta|y)$ is the posterior distn
  - θ now includes all parameters (i.e., both θ and ϕ in the hierarchical model)
  - often we only know the unnormalized posterior distn $q(\theta|y)$
    - i.e., $p(\theta|y) \propto p(y|\theta)p(\theta) = q(\theta|y)$
    - more formally, $p(\theta|y) = c(y)q(\theta|y)$
  - our computation discussion will generally use $p(\theta|y)$ to refer to both normalized/unnormalized posterior distribution
  - I will point out whether it matters whether the posterior distn is normalized

Computation

Initial estimation

- Starting point for subsequent approaches
- Serves as a check for other approaches
- Problem-specific methods are required
  - use results from other statistical approaches (e.g., maximum likelihood estimates in bioassay logistic regression)
  - fix hyperparameters at crude estimates
    (e.g., consider separate and pooled estimates for the 8 schools example which are equivalent to $\tau = \infty$ and $\tau = 0$)

Computation

Numerical integration

- Many quantities of interest in a Bayesian analysis can be written as $E(h(\theta)|y) = \int h(\theta)p(\theta|y)d\theta$
  (e.g., posterior mean)
- These can be obtained by numerical integration
- In modern world, simulation is often preferred (but numerical integration still used)
- We focus here briefly on some useful tools

Computation

Numerical integration

- Traditional quadrature
  - trapezoidal rule (piecewise linear approximation)
  - Simpson's rule (piecewise quadratic)
  - algorithms for iterating
  - Gaussian quadrature

Computation
Numerical integration

- Integration via direct simulation
  - if we can generate $\theta_1, \ldots, \theta_N$ from $p(\theta|y)$ then we can estimate integral as $\sum_i h(\theta_i)/N$
  - of course, this is equivalent to direct simulation from the posterior distribution
- Importance sampling
  - can write $E(h(\theta)|y) = \int \frac{h(\theta)p(\theta|y)}{g(\theta)} g(\theta) d\theta$
  - if we can generate $\theta_1, \ldots, \theta_N$ from $g(\theta)$, then we can estimate integral as $\frac{1}{N} \sum_i \frac{h(\theta_i)p(\theta_i|y)}{g(\theta_i)}$
  - $w(\theta_i) = p(\theta_i|y)/g(\theta_i)$ is known as the importance ratio
  - improves upon simple MC if we can find g yielding low variability weights
  - works very poorly if g's tails are too short (we get some very large importance ratios)

Computation

Numerical integration

- Dealing with unnormalized distributions
  - suppose we only have $q(\theta|y)$
  - numerical integration and importance sampling approaches can work
  - write $E(h(\theta)|y) = \frac{\int h(\theta)q(\theta|y)d\theta}{\int q(\theta|y)d\theta}$
  - apply numerical integration or importance sampling separately to numerator and denominator
- There are many techniques for improving upon Monte Carlo (simulation) approaches to numerical integration (e.g., antithetic variables) ... see statistical computing texts

Computation

Numerical integration

- Analytical approximation (Laplace's method)
  - can write $E(h(\theta)|y) = \int e^{\log(h(\theta)p(\theta|y))} d\theta$
  - approximate $u(\theta) = \log(h(\theta)p(\theta|y))$ using a quadratic expansion around the mode θ_o
  - find $E(h(\theta)|y) \approx h(\theta_o)p(\theta_o|y)(2\pi)^{-d/2}|-u''(\theta_o)|^{1/2}$
  - requires large samples
  - need two approximations for unnormalized posterior distn
    $(E(h(\theta)|y) = \int h(\theta)q(\theta|y)d\theta / \int q(\theta|y)d\theta)$

Computation
Direct simulation

- We have already seen that simulation is a powerful approach for studying the posterior distn in a Bayesian analysis
- Next, briefly discuss some basic direct simulation tools
  - these are useful in simpler (low dimensional) problems
  - these same tools are useful as components for more advanced simulations
- Simulation analysis
  - report number of draws
  - report summary statistics (mean, sd, percentiles)
  - graphs
  - how many draws? depends on desired accuracy (e.g., if we have iid simulations then std error of posterior mean is equal to posterior s.d. divided by $\sqrt{n}$)

Computation

Direct simulation approaches

- Exact simulation
  - standard algorithms for drawing from standard distns (uniform, normal, Poisson, gamma, etc.)
  - available in most software including R
- Grid approximation
  - discrete (evenly spaced) grid $\theta_1, \theta_2, \ldots, \theta_N$,

$$\Pr_{grid}(\theta = \theta_j) = p(\theta_j|y)/(\sum_i p(\theta_i|y))$$

  - we have already seen this approach
  - works for normalized or unnormalized posterior distn
  - hard in 2 or more dimensions
  - choice of grid can affect the answer

Computation

Direct simulation approaches

- Probability integral transform
  - consider posterior distn $p(\theta|y)$ with corresponding cdf $F(\theta|y)$
  - recall probability result: if $U \sim \text{Unif}(0,1)$, then $\theta = F^{-1}(U)$ is a r.v. with distn $p(\theta|y)$
  - e.g., if $\theta|y \sim N(\mu, \tau^2)$, then $\theta = \mu + \tau\Phi^{-1}(U)$
  - discrete r.v.'s are possible but harder to program
  - can use this to improve grid by making a trapezoidal approximation

Computation

Direct simulation approaches

- Rejection sampling
  - suppose we find $g(\theta)$ that we can sample from with $p(\theta|y)/g(\theta) \leq M$ (with M known)
  - algorithm:
    - draw $\theta \sim g(\theta)$
    - accept θ with prob $p(\theta|y)/(Mg(\theta))$, otherwise reject and draw a new candidate
    - for log-concave densities this approach can be used with trapezoids defining rejection function (Gilks and Wild, 1992, Applied Statistics)
- Many other useful methods for direct simulation that we don't have time to discuss here

Computation

Iterative simulation

- Basic idea: to sample from $p(\theta|y)$ create a Markov chain with $p(\theta|y)$ as stationary distribution
- Algorithms:
  - Gibbs sampler (full conditionals)
  - Metropolis-Hastings algorithm (jumping distn)
  - combinations of Gibbs and M-H
  - Hamiltonian Monte Carlo
- Implementation issues (later)

Iterative simulation

Gibbs sampler

- Key features
  - break problem into lower-dimensional pieces using conditional distributions
  - conditional posterior distributions often have simple form
- Start by drawing an initial $\theta = (\theta_1, \ldots, \theta_k)$ from an approximation to $p(\theta|y)$.
- Repeat the following steps using most recently drawn values for variables in conditioning set:
  - draw θ_1 from $p(\theta_1 \mid \theta_2, \ldots, \theta_k, y)$
  - draw θ_2 from $p(\theta_2 \mid \theta_1, \theta_3, \ldots, \theta_k, y)$
  - $\ldots$
  - draw θ_k from $p(\theta_k \mid \theta_1, \ldots, \theta_{k-1}, y)$
- Can update parameters one at a time (as above) or in blocks

Iterative simulation

Gibbs sampling

- Efficiency considerations
  - partitioning parameters into groups/blocks is often a good idea
  - works best if we can create independent or nearly independent blocks of parameters
  - transform distributions/parameters (e.g., t as a scale mixture of normals, centering random effects)
- Example of Gibbs sampling (normal-normal model)

Iterative simulation

Non-standard distributions

- It may happen that one or more of the Gibbs sampling distns is not a known distn
- What then?
  - can go back to previous direct simulation discussion (i.e., use grid approximation, rejection sampling, etc.) but this is not ideal
  - Metropolis (or (Metropolis-Hastings) algorithm

Iterative simulation

Metropolis-Hastings (M-H) algorithm

- Replaces "conditional draws" of Gibbs sampler with "jumps" around the parameter space
- Algorithm:
  - given current draw θ (scalar or vector)
  - sample a candidate point θ^* from jumping distribution $J(\theta^*|\theta)$
  - accept candidate or stay in place with probabilities determined by importance ratio

$$r = \frac{p(\theta^*|y)/J(\theta^*|\theta)}{p(\theta|y)/J(\theta|\theta^*)}$$

- Simplifies if J is symmetric (Metropolis algorithm)
- Combining M-H and Gibbs: M-H steps can be used in place of one conditional distn in a Gibbs sampler, or a single M-H step can replace several (or even all) of the conditional distns

Iterative simulation

Efficiency considerations - M-H

- How do we choose the jumping distribution $J(\theta|\theta^{(t-1)})$?
- Optimal J is $p(\theta|y)$ independent of current value $\theta^{(t-1)}$
  - this always accepts ($r = 1$)
  - but if we could do this we wouldn't need M-H
- Goals in choosing J:
  - J should be easy to sample from
  - it should be easy to compute r
  - jumps should go far (so we move around the parameter space) but not too far (so they are not always rejected)

Iterative simulation

Efficiency considerations - M-H

- Three common approaches
  - independence M-H
  - random walk M-H (used most often)
  - approximation M-H
- Independence M-H
  - find a distribution $g(\theta)$ independent of current $\theta^{(t-1)}$ and keep generating candidates from $g(\theta)$
  - requires g be a reasonably good approximation
  - hard to do for M-H within Gibbs

Iterative simulation

Efficiency considerations - M-H

- Random Walk M-H
  - generate candidate using random walk (often normal) centered at current value
  - $J(\theta|\theta^{(t-1)}) = N(\theta|\theta^{(t-1)}, cV)$
  - note this is symmetric so M-H acceptance calculation simplifies
  - works well if V is chosen to be posterior variance (don't know this but can use a pilot run to get some idea)
  - c is a constant chosen to optimize efficiency
  - theory results indicate optimal acceptance rate for this kind of jumping distn is between .2 and .5 (decreases with dimension)

Iterative simulation

Efficiency considerations - M-H

- Approximation M-H
  - generate candidate using an approximation to target distn (varying from iteration to iteration)
  - e.g., $J(\theta|\theta^{(t-1)}) = N(\theta|\theta^{(t-1)}, V_{\theta^{(t-1)}})$
  - now variance matrix depends on current value so this is no longer symmetric
  - idea is to make this a good approximation (high acceptance rate)

Iterative simulation

Proof of convergence - using Metropolis

- Show resulting Markov chain has a unique staionary distribution (i.e., is irreducible, aperiodic, non-transient)
- Show stationary distribution is $p(\theta|y)$
  - Start algorithm at $\theta^{(t-1)} \sim p(\theta|y)$
  - We can show that $p(\theta^{(t-1)}, \theta^{(t)})$ is symmetric which means that $\theta^{(t)} \sim p(\theta|y)$ (hence p is stationary distn)
    - Let θ_a, θ_b be two points in parameter space with $p(\theta_b|y) \geq p(\theta_a|y)$
    - $p(\theta^{(t-1)} = \theta_a, \theta^{(t)} = \theta_b) = p(\theta_a|y)J_t(\theta_b|\theta_a)$ (since we accept jumps to θ_b
    - $p(\theta^{(t-1)} = \theta_b, \theta^{(t)} = \theta_a) = p(\theta_b|y)J_t(\theta_a|\theta_b)\frac{p(\theta_a|y)}{p(\theta_b|y)}$
    - Jumping distribution is symmetric so these two expressions are equal and the joint distribution is symmetric

Iterative simulation

Hamiltonian Monte Carlo

- Gibbs sampling and Metropolis-Hastings are random walk approaches
- They can perform poorly in high dimensional spaces
- Hamiltonian Monte Carlo (HMC) uses ideas from deterministic simulation of physical systems
- Newtonian mechanics works in terms of forces, masses, velocities in a fixed co-ordinate systems
- Hamiltonian and Lagrangean mechanics arise as an alterantive mathematical formalism that reproduces Newtonian results but enables modeling more complex systems
- HMC is derived from this formalism; it introduces a momentum variable ϕ_j corresponding to each model parameter θ_j

Iterative simulation

Hamiltonian Monte Carlo

- Target is now $p(\theta, \phi|y) = p(\theta|y)p(\phi)$
- Note that ϕ is independent of y
- Common choice for $p(\phi)$ is $N(\phi|0, M)$ with M (mass matrix) diagonal
- Algorithm (iterates over time, here assume we have $\theta^{(t-1)}$)
  - Generate $\phi^{(t-1)} \sim p(\phi)$
  - update θ, ϕ via L leapfrog steps (scaled by a factor ϵ)
  - repeat L times
  - $\phi \leftarrow \phi + 0.5\ \epsilon\ d(log\ p(\theta|y)/d\theta$
  - $\theta \leftarrow \theta + \epsilon M^{-1}\phi$
  - $\phi \leftarrow \phi + 0.5\ \epsilon\ d(log\ p(\theta|y)/d\theta$
  - at the end of the L steps call the result θ^*, ϕ^*
  - Accept the proposed pair with probability $r = \frac{p(\theta^*|y)p(\phi^*)}{p(\theta^{(t-1)}|y)p(\phi^{(t-1)})}$
- ϵ, L, M are turning parameters
  (often $\epsilon = 0.1, L = 10$ and M approx $Var(\theta|y)^{-1}$)

Iterative simulation

Logistics

- We have glossed over some details
  - starting values
  - monitoring convergence
  - inference from iterative simulation
  - software availability

Iterative simulation

Starting values

- Markov chain will converge to stationary distribution from **any** starting value assuming
  - chain has a nonzero probability of eventually getting from any point to any other point (i.e., parameter space is not divided into separate regions)
  - chain does not drift off to infinity (can happen if the posterior distribution is improper – which means the model is wrong!)
- Assessing when this convergence has occurred is best done using multiple chains with overdispersed starting points

Iterative simulation
Starting values

- An algorithm for choosing starting values:
  - find posterior mode (or modes)
    (marginal distn usually better than joint distn)
  - create overdispersed approximation to posterior
    (e.g., t_4 instead of normal)
  - sample 1000 points from approximation
  - resample 5 or 10 starting values
    (using importance ratios as described later)

Iterative simulation

Monitoring convergence

- Run several sequences in parallel
- Can use graphical displays to monitor convergence or semi-formal approach of Gelman and Rubin (described now)
- Two estimates of $\text{sd}(\theta|y)$
  - underestimate from sd within each sequence
  - overestimate from sd of mixture of sequences
- Potential scale reduction factor:

$$\sqrt{\widehat{R}} = \frac{\text{mixture-of-sequences estimate of sd}(\theta|y)}{\text{within-sequence estimate of sd}(\theta|y)}$$

- Initially $\sqrt{\widehat{R}}$ is large (because we use overdispersed starting points)
- At convergence, $\sqrt{\widehat{R}} = 1$ (each sequence has made a complete tour)
- Monitor $\sqrt{\widehat{R}}$ for all parameters and quantities of interest; stop simulations when they are all near 1 (e.g., below 1.2)

Iterative simulation

Inference from posterior simulations

- At approximate convergence we have many draws from the posterior distribution
- The draws are **not** independent
- This means that obtaining standard errors to assess simulation noise is difficult (can use between-chain info, batching, ...)
- Note there is a distinction here between posterior uncertainty about θ and Monte Carlo uncertainty about some summary of the posterior distn (e.g., std error of $E(\theta|y)$)
- Good news: Simulation noise is generally minor compared to posterior uncertainty about θ

Iterative simulation

Improving MC simulation

- Earlier discussed some ideas for improving efficience of Gibbs / Metropolis
- Those ideas are based on the algorithms
- Can also improve MCMC performance by modifying the model
- Some ideas follow

Iterative simulation
Transformations

- Gibbs/Metropolis work best for independent components
- Can sometimes transform parameters of a distribution
- Example: Beta distribution is usually parameterized in terms of α, β with $p(\theta|\alpha,\beta) \propto \theta^{\alpha-1}(1-\theta)^{\beta-1}$
- Can reparameters in terms of mean $\alpha/(\alpha+\beta)$ and (rough) variance parameter $1/(\alpha+\beta)$
- Can further reparameterize as logit of mean ($\log(\alpha/\beta)$) and log of variance ($\log(\alpha+\beta)$)

Iterative simulation

Auxiliary variables

- Some distributions can be expressed as mixtures of simpler distributions
- Example: consider the t distribution with ν degrees of freedom and suppose we wish to model $Y_i \sim t_\nu(\mu, \sigma^2)$
- Simulating from posterior distribution of μ, σ^2 from t-density is challenging
- Note that we can introduce V_i with $Y_i|\mu_i, V_i \sim N(\mu, V_i)$ and $V_i|\sigma^2 \sim Inv - \chi^2(\nu, \sigma^2)$
- Marginal distribution of Y_i is t-distribution
- Gibbs sampling is straightforward if we think of μ, σ^2 as parameters and V as "missing" data (another unknown to include in Gibbs sampling)

Iterative simulation

Parameter expansion

- Previous example obtained improved performance by adding "missing" variable V
- It is counterintuitive but sometimes adding an additional parameter improves efficiency
- Example: consider the t distribution with ν degrees of freedom example from previous slide
- Rewrite our model with added parameter α as $Y_i|\mu, \alpha, U_i \sim N(\mu, \alpha^2 U_i)$ and $U_i|\tau^2 \sim Inv - \chi^2(\nu, \tau^2)$
- Note that α is not identified $(\alpha^2 U_i = V_i, \alpha^2\tau^2 = \sigma^2)$
- But ....
- Gibbs sampling in this model will work if we monitor convergence in terms of $\mu, \sigma^2 = \alpha^2\tau^2, V_i = \alpha^2 U_i$
- Not only does it work, but it tends to be more reliable
- Why? Increasing the size of the parameter space can help getting trapped in uninteresting areas (e.g., σ near zero in the original formulation)

Iterative simulation

Many other extensions / expansions

- Reversible jump MCMC to explore inference across multiple models
- Simulatied tempering and other approaches to explore multiple modes

Iterative simulation
Software availability

- Variety of packages
  - R – write your own MCMC
  - WINBUGS (BUGS = Bayesian analysis Using Gibbs Sampling)
  - JAGS
  - STAN
  - JAGS and STAN can be run from within R (runjags, rstan packages)

Computation

Debugging iterative simulation methods

- Checking that programs are correct is crucial (especially if you write your own)
- Can be difficult to check because
  - output is a distribution not a point estimate
  - incorrect output may look reasonable
- Some useful debugging ideas:
  - build up from simple (debugged) models
  - when adding a new parameter, start by setting it to a fixed value, then let it vary
  - simulate fake data (repeat the following steps)
    - draw "true parameters" from prior distn (must be proper)
    - simulate data from the model
    - obtain draws from posterior distn
    - compare distns of posterior draws and "true parameters"

Computation

Debugging iterative simulation methods

- Common problems
  - conceptual flaw in part of model
  - prior is too vague
    - this may give improper posterior distn
    - detect by looking for values that don't make substantive sense

Computation

Approximation

- Recall results of Chapter 4 ... for large samples $p(\theta|y)$ is approx $N(\theta|\hat{\theta}, J(\hat{\theta})^{-1})$ where $\hat{\theta}$ is the posterior mode
- Often use inverse curvature matrix of log posterior density, $V_\theta = \left[-\frac{d^2}{d\theta^2} \log p(\theta|y)|_{\theta=\hat{\theta}}\right]^{-1}$, as variance matrix
- Transformations are often used to improve quality of normal approx
- May use t distn with few degrees of freedom in place of normal distn (to protect against long tails)
- Multiple modes can be a problem: $\mathrm{N}(\hat{\theta}, V_\theta)$ or $t_4(\hat{\theta}, V_\theta)$ approx at each mode (i.e., a mixture)
- Reasons **not** to approximate based on modes:
  - misleading in some problems (e.g., in 8 schools example, mode is $\tau = 0$ which is at edge of parameter space)
  - advances in algorithms have made inference from exact posterior distn possible

Computation

Approximation - mode finding

- To apply normal approximation, need posterior mode
- Review traditional stat computing topic of mode finding (optimization)
- Iterative conditional modes (ICM)
  - start at $\theta^{(0)} = (\theta_1^{(0)}, \ldots, \theta_d^{(0)})$
  - for $i = 1, \ldots$
  - for $j = 1, \ldots, d$
    - choose $\theta_j^{(i)}$ as the value that maximizes (or even just increases) $p(\theta_1^{(i)}, .., \theta_{j-1}^{(i)}, \theta, \theta_{j+1}^{(i-1)}, .., \theta_d^{(i-1)})$
- ICM leads to a local maximum

Computation

Approximation - mode finding

- Newton's method ($L = \log p(\theta|y)$)
  - start at $\theta^{(0)}$
  - iterate with $\theta^{(t)} = \theta^{(t-1)} - [L''(\theta^{(t-1)})^{-1} L'(\theta^{(t-1)}$
  - converges fast but is sensitive to starting value
  - can use numerical derivatives
- Other optimization methods
  - steepest ascent $\theta^{(t)} = \theta^{(t-1)} + \alpha L'(\theta^{(t-1)})$
  - quasi-Newton methods
  - simplex/polytope (no derivative methods)

Computation
Approximation

- For many problems, especially hierarchical models, the joint mode is not very useful
- Instead may focus on factorization
  $p(\theta, \phi|y) = p(\phi|y)p(\theta|\phi, y)$
- Often $p(\theta|\phi, y)$ is easy (e.g., conjugate family)
- Normal approximation for marginal posterior distn $p(\phi|y)$
- But need mode of $p(\phi|y)$
  - sometimes this function can be identified and maximized analytically
  - for other situations EM algorithm is helpful

Computation

Approximation - The EM algorithm

- EM is an iterative algorithm for maximizing functions (likelihoods or posterior distns) when there is missing data
- Applied here in maximizing $p(\phi|y)$ treating θ as missing data
- Idea:
  - start with initial guess for ϕ
  - given ϕ we can estimate "missing data" θ
  - given estimated θ it may be easy to now maximize for improved ϕ
  - repeat last two steps

Computation

Approximation - The EM algorithm

- Iterative algorithm with two steps
- Suppose current value of ϕ is $\phi^{(t)}$
  - E-step
    - compute $Q(\phi) = E(\log(p(\theta, \phi|y)|\phi = \phi^{(t)}) = \int \log(p(\theta, \phi|y))p(\theta|\phi^{(t)}, y)d\theta$
    - essentially computes expected value of needed functions of θ rather than estimating the "missing" θ
  - M-step
    - choose $\phi^{(t+1)}$ as the value of ϕ that maximizes $Q(\phi)$
- Can show that $p(\phi|y)$ increases after each E-M pair of steps

Computation

Approximating the Conditional Distribution

- EM-based approximation works when we know the conditional distribution $p(\theta|\phi, y)$
- If not, an alternative is to first approximate this conditional distribution on a grid of ϕ values, e.g., $p_{approx}(\theta|\phi, y) = N(\theta|\hat{\theta}(\phi), V_\theta(\phi))$
- Then can derive approximation to the marginal distribution

$$p_{approx}(\phi|y) = \frac{p(\phi, \theta|y)}{p_{approx}(\theta|\phi, y)}$$

Computation

Approximation - Variational Inference

- In very large or complex problems it can be prohibitively expensive to carry out MCMC calculations
- Variational inference is an alternative approach that builds an approximation to the joint posterior distribution from simpler functions
- Note MCMC is simulating from the correct distribution (but has MC error)
- Variational inference is simulating from a different (nearby) distribution (and has MC error)

Computation

Approximation - Variational Inference

- Most common approach is to choose to approximate $p(\theta|y)$ with $g(\theta|\phi) = \prod_{j=1}^{J} g_j(\theta_j|\phi_j)$ where J is the number of parameters
  - Note that ϕ here is not a hyperparameter, it is a parameterization of our approximating distribution
  - Goal is to estiimate ϕ (i.e., it will depend on the data) and then use simulations from $g(\theta|\hat{\phi})$ as our (approximate) draws from the posterior distribution
- Which g? Construct g to minimize the K-L divergence $KL(g||p) = -E_g \log(p(\theta|y)/g(\theta))$

Computation

Approximation - Variational Inference

- How does this work in practice
- Can find best functional form for $g_j(\theta_j|\phi_j)$ by examining $E_{g_{-j}}(\log p(\theta|y))$
- This quantity is viewed as a function of θ_j with expectation taken over all other parts of θ (for which we assume we already have approximating g's)
- Then the algorithm proceeds as follows
  - initial guesses for all the ϕ_j's
  - iterate $j = 1, \ldots, J$, update ϕ_j such that $\log g_j(\theta_j|\phi_j) = E_{g_{-j}}(\log p(\theta|y))$
- A common alternative to the above is to just choose convenient forms for g_j and numerically minimize K-L divergence

Computation

Approximation - Expectation Propagation

- Variational inference approximates posterior by considering each dimensions separately
- Expectation propagation is an alternative strategy that focuses on approximating each data contribution to the posterior separately
- The target is $p(\theta|y) = p(\theta)\prod_i p((y_i|\theta) = p(\theta)\prod_i p_i(\theta)$
- Our approximation is $g(\theta)$ (often multivariate normal)
- Turns out this approach is equivalent to minimizing the alternative K-L divergence, $KL(p||g) = -E_p \log(g(\theta)/p(\theta|y))$
- No details here (some in book and other references)

Computation

Approximation - Approximate Bayes Computation (ABC)

- In some problems we don't have the likelihood in closed form (e.g., have only a simulation model for $y|\theta$)
- ABC is an apporach that can work in this case
- Algorithm - repeat as often as desired
  - draw θ from $p(\theta)$ (requires proper prior distribution)
  - simulate $y^{(rep)}$ from $p(y|\theta)$
  - compute $d(y^{(rep)}, y)$ for suitable distance function d (so that y and $y^{(rep)}$ agree on relevant features)
  - accept θ if $d(y^{(rep}, y) < \epsilon$
- How does it work? We are simulating from $p(\theta, y)$ and then conditioning on observed y which yields the posterior
- Challenges - Need to define d, ϵ. Doesn't work well if prior distribution is too broad

Computation
Summary

- Goal: posterior inference concerning the vector of parameters (and any missing data)
- Simulation is an extremely powerful tool, especially in complex models
- Basic approach
  - initial estimates
  - direct simulation (if possible)
  - if direct simulation is not possible:
    - normal or t approximation about posterior mode
    - iterative simulation (Gibbs, Metropolis-Hastings)
- For iterative simulation
  - inference is conditional on the starting points
  - use multiple sequences and run until they mix