Computing for Bayesian Inference

(with thanks to Tom Loredo for helpful notes)

Astroinformatics Summer School, Penn State University

June 2018

Murali Haran

Department of Statistics, Penn State University

Outline

Bayesian Inference

- Data: $\mathbf{X} \in \chi$, parameter: $\theta \in \Theta$
- Probability model: $f(\mathbf{X}|\theta)$
- Prior specifies the distribution for the parameters before obtaining observations : $p(\theta)$
- Inference is based on the posterior distribution

$$\begin{aligned}\pi(\theta|\mathbf{X}) &= p(\theta)f(\mathbf{X}|\theta)/C \\ &\propto p(\theta)f(\mathbf{X}|\theta) = h(\theta|\mathbf{X}),\end{aligned}$$

where $h(\theta|\mathbf{X})$ is the unnormalized posterior and C is the normalizing constant so

$$\pi(\theta|\mathbf{X}) = h(\theta|\mathbf{X})/C, \;\; C = \int h(\theta|\mathbf{X})d\theta$$

(Toy) Example 1

Inferring Poisson rate from count data

- $X_1, \ldots, X_n \overset{iid}{\sim}$Poisson($\lambda$)
- Prior $p(\lambda) \sim$Gamma($a = 1, b = 20$), expectation=20, variance=40
- Inference based on $\pi(\lambda | X_1, \ldots, X_n)$

$$
\begin{aligned}
&\propto \prod_{i=1}^{n} \frac{\lambda^{X_i} e^{-\lambda}}{X_i!} \frac{1}{\Gamma(a) b^a} \lambda^{a-1} e^{-\lambda/b} \\
&\propto \lambda^{\sum_{i=1}^{n} X_i + a - 1} e^{-\lambda(n + 1/b)} \\
&\propto \text{Gamma}\left(\sum_{i=1}^{n} X_i + a, (n + 1/b)^{-1} \right)
\end{aligned}
$$

Example 2

Simple linear regression

- $Y_1, \ldots, Y_n$ are conditionally independent, with $Y_i \mid \beta_1, \tau \sim N(\beta_1 X_i, \tau),\ i = 1, \ldots, n$.
- "Flat" priors for β_1, that is, pdfs $p(\beta_1) \propto 1$, and $\tau \sim$Unif$(a = 2, b = 10)$.

  (This is a computing module, so we will leave prior specification details to other tutorials.)
- Inference is based on $\pi(\beta_1, \tau \mid \mathbf{Y})$
- From an algorithm that approximates $\pi(\beta_1, \tau \mid \mathbf{Y})$ obtain
  (1) Marginal and joint distributions of parameters
  (2) Point and interval estimates for parameters
  (3) Predictions at new data points X^*, while incorporating uncertainty about the parameters from (1) above

Example 2: Posterior Distribution

- Inference is based on $\pi(\beta_1, \tau \mid \mathbf{Y})$

$$\propto \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\tau}} \exp\left(-\frac{1}{2\tau}(Y_i - \beta_1 X_i)^2\right) \mathbf{1}_{(2<\tau<10)}$$

$$\propto \tau^{-n/2} \exp\left(-\frac{1}{2\tau}\sum_{i=1}^{n}(Y_i - \beta_1 X_i)^2\right) \mathbf{1}_{(2<\tau<10)}$$

- **Full conditional distribution** for a particular parameter: delete parts of function that do not contain that parameter

$$\pi(\beta_1|\tau, \mathbf{Y}) \propto \exp\left(-\frac{1}{2\tau}\sum_{i=1}^{n}(Y_i - \beta_1 X_i)^2\right)$$

$$\pi(\tau|\beta_1, \mathbf{Y}) \propto \tau^{-n/2} \exp\left(-\frac{1}{2\tau}\sum_{i=1}^{n}(Y_i - \beta_1 X_i)^2\right) \mathbf{1}_{(2<\tau<10)}$$

Bayesian Computational Problems

1. **Marginal and joint distributions** of parameters, $\pi(\theta|\mathbf{X})$.
2. Predictions while accounting for parametric uncertainty. **Posterior predictive distribution**, $\int f(X^*|\theta)\pi(\theta|\mathbf{X})d\theta$
3. Comparing evidence for different models. **Bayesian model choice/hypothesis testing**. Two competing models, M_1, M_2, compare $P(M_1|\mathbf{X})$ to $P(M_2|\mathbf{X})$. Bayes factor,

$$\frac{P(M_2|\mathbf{X})}{P(M_1|\mathbf{X})} = \frac{f(\mathbf{X}|M_2)P(M_2)}{f(\mathbf{X}|M_1)P(M_1)},$$

need marginals like $f(\mathbf{X}|M_2) = \int f(\mathbf{X}|\theta)f(\theta)d\theta$, integrating out θ parameter(s) of M_2

Integration

- What do all the computational problems have in common? **Integration/summation**, often high-dimensional
- Note that there are alternatives, for example

$$\hat{\theta} = \arg\max_{\theta \in \Theta} \pi(\theta|\mathbf{x})$$

  to obtain the maximum a posterior (MAP) parameter estimate, $\hat{\theta}$
- Very fast optimization approaches exist. This is a potential advantage over integration methods
- But advantages to integration-based approaches – more flexibility, stability, and ability to tackle complicated problems, especially with high-dimensional latent variables

Useful Algorithms for Integration

- Numerical approximations
  - (1) Laplace approximations
  - (2) Quadrature methods (will skip this)
- Monte Carlo methods
  - (3) Basic (IID) Monte Carlo
  - (4) Importance sampling
  - (5) Markov chain Monte Carlo (MCMC)
- I will describe each of these algorithms at a high level first, providing details later
- These methods have nothingly inherently to do with Bayesian inference. But for ease of exposition, I will describe everything in a Bayesian context

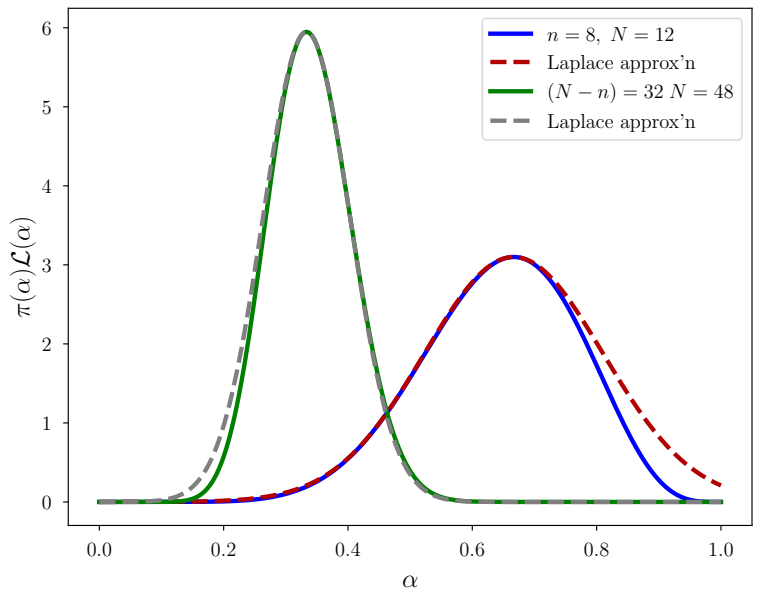
Outline

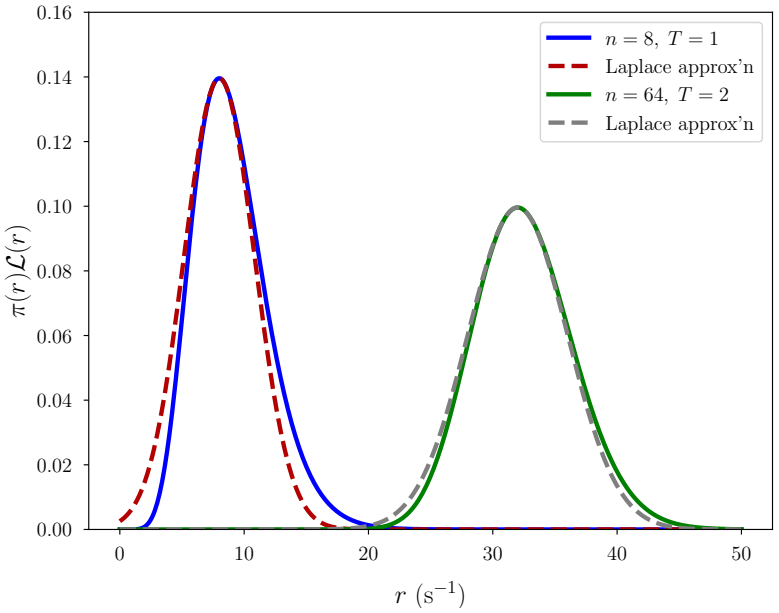
(1) Laplace Approximation Idea

(Tierney and Kadane, 1986)

- Idea: use a Taylor series approximation so that the posterior $\pi(\theta|\mathbf{X})$ is approximated by a Normal distribution
- Normal is specified with mean = mode of the posterior, covariance=curvature at the mode
- Normal distributions are easy to work with, hence easy to approximate various quantities of interest

(1) Laplace Approximation Example





Gamma and Beta examples (courtesy T. Loredo; MacKay: Information Theory, Inference, and Learning Algorithms: The Book)

(1) Laplace Approximation Algorithm

- Optimization: find $\hat{\theta} = \arg\max_{\theta \in \Theta} \pi(\theta|\mathbf{x})$
- Compute $\hat{\mathbf{I}}$, which approximates the curvature of the posterior surface at $\hat{\theta}$

$$\hat{\mathbf{I}} = -\frac{\partial^2 \log(\pi(\theta|\mathbf{x}))}{\partial^2 \theta}\Big|_{\theta=\hat{\theta}}$$

  Note that the normalizing constant vanishes in the above calculations of $\hat{\theta}, \hat{\mathbf{I}}$
- $\hat{\mathbf{I}}$ is Observed Fisher Information matrix for a flat prior $\approx$ inverse covariance matrix
- $\pi(\theta|\mathbf{x}) \approx N(\hat{\theta}, \mathbf{I}^{-1})$

Laplace Approximation Calculations

- Multivariate normal distributions: expectations, normalizing constants, probabilities etc. are easily obtained
- Posterior expectation, $\mu = E_\pi(g(\theta)) = \int g(\theta)\pi(\theta|\mathbf{X})d\theta$ approximated by $g(\tilde{\theta})\pi(\tilde{\theta})(2\pi)^{n/2}|\tilde{\mathbf{I}}|^{-1/2}$, where $\tilde{\theta}$ maximizes $g(\theta)\pi(\theta|\mathbf{X})$
- Marginal likelihood, $\int \pi(\theta|\mathbf{X})d\theta \approx \pi(\hat{\theta}|\mathbf{X})(2\pi)^{n/2}|\hat{\mathbf{I}}|^{-1/2}$, where $\hat{\theta}$ maximizes $\pi(\theta|\mathbf{X})$
- Marginal posterior for $\theta = (\phi, \eta)$, $\pi(\phi|\mathbf{X}) \approx \mathbb{P}(\phi, \hat{\eta}(\phi))\mathcal{L}_p(\phi)|\mathbf{I}(\phi)|^{-1/2}$, where $\mathcal{L}_p(\phi)$ is profile likelihood, $\max_\eta \mathcal{L}(\phi, \eta) = \mathcal{L}(\phi, \hat{\eta}(\phi))$, and $\mathbf{I}(\phi) = \partial_\eta \partial_\eta \log(\pi(\theta|\mathbf{X}))$

Comments on Laplace Approximation

- Of course depends crucially on whether posterior is smooth and unimodal. In the case of multiple modes, can try to use mixtures of Gaussians to approximate
- Due to taking ratios, the approximation can have nice theoretical properties (approximation error is of order 1/n, where n is sample size)
- The approximation is based on asymptotic arguments. Diagnostics are not simple. Unlike Monte Carlo methods, cannot get arbitrarily accurate approximations

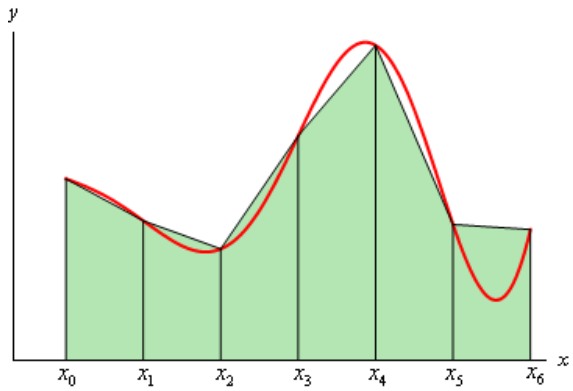
Outline

(2) Quadrature

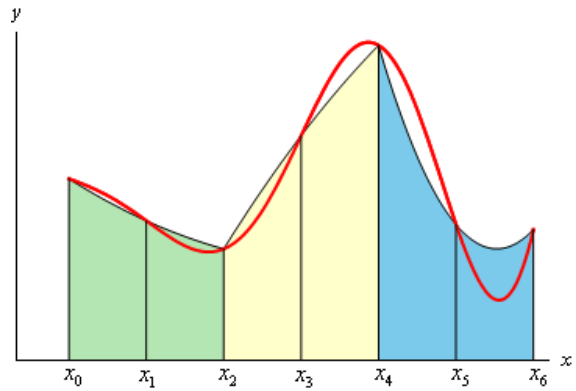
Recall old ideas from calculus

Trapezoid and Simpson rules

Trapezoid rule
Piecewise-linear approximation



Simpson's rule
Piecewise-parabolic approximation



From http://tutorial.math.lamar.edu

(2) Quadrature Idea

- Idea: approximate $\mu = \int_a^b g(\theta)\pi(\theta|\mathbf{X})d\theta$ by taking a grid of points, $\theta^{(i)}, i = 1, \ldots, n$, and then calculating $\hat{\mu} = \sum_{i=1}^{n} w_i g(\theta^{(i)})\pi(\theta^{(i)}|\mathbf{X})$
- The art is in the choice of the grid and the weights, w_i, leading to simple methods like the trapezoid rule and Simpson's method, to advanced methods like Gauss-Hermite or adaptive quadrature
- This can be useful for very low-dimensional (< 5) problems, maybe can push it for 10-20 dimensions in some cases. Also useful in *combination with other methods*, that is, can use quadrature to do lots of small integrals within a more complicated algorithm

Quadrature and Related Methods

- Related: Quasi Monte Carlo is an active area of research (look up recent SAMSI program to get a sense of the state of the art) where they are trying to push the envelope in terms of problem dimensions
- Also, see R. Joseph "Bayesian Computation Using Design of Experiments-Based Interpolation Technique" (Technometrics, 2012)
- Best to use existing implementations to the extent possible, combining it with your other code
- Adaptive quadrature for 1-d problems is implemented in base `R` command `integrate`. In python: `ADAPT`, `CUHRE`, `BAYESPACK`, `CUBA`

Outline

(3) IID Monte Carlo

- Goal: approximate expectations with respect to $\pi(\theta|\mathbf{X})$.
  - $\mu = E_\pi(g(\theta))$, for some real-valued function $g(\cdot)$.
    E.g. if $g(x) = x$ this is the expected value of θ under π:
    $\mu = E_\pi(\theta) = \sum g(\theta_i)\pi(\theta_i|\mathbf{X})$ or $\int g(\theta_i)\pi(\theta_i|\mathbf{X})d\theta$
- Simple idea: use pseudo-random values from $\pi(\theta|\mathbf{X})$
  (1) Simulate $\theta^{(1)}, \theta^{(2)}, \ldots \overset{iid}{\sim} \pi(\theta|\mathbf{X})$
  (2) Approximate μ by the sample mean

$$\hat{\mu} = \sum_{i=1}^{n} \frac{g(\theta^{(i)})}{n} \rightarrow \mu$$

the convergence results on the right is simply the Law of Large Numbers (LLN). The LLN holds as long as $E_\pi(g(\theta)) < \infty$

Simplicity of Monte Carlo

- Approximation of expectation works for any real-valued function $g(\cdot)$
- Hence, only need to change the function $g(\cdot)$ applied to the samples, and simply take the sample mean to obtain approximation
- For instance, suppose θ is 3-dimensional. To approximate $E_\pi(\theta_1/\theta_2 \log(\theta_3))$, simply use $g(\mathbf{x}) = x_1/x_2 \log(x_3)$. Other ideas remain similar, including (next slide) assessing accuracy

IID Monte Carlo Sample Size

- How do we determine the number of Monte Carlo samples?
- Use basic statistics idea: compute Monte Carlo ("simulation") standard error (MCse) of $\hat{\mu}$. We rely on Central Limit Theorem which applies here as long as second moment is also finite
  - MCse of $\hat{\mu}$ = sd $(\hat{\mu})/\sqrt{n}$
  - 95% confidence interval is $\hat{\mu} \pm 1.96 \times$MCse
  - Choose n large enough so interval is narrow enough
- The basic principle of wanting to control standard errors stays the same for other Monte Carlo approaches but things get more complicated with importance sampling, and even more complicated with Markov chain Monte Carlo

The Power of Monte Carlo

- Ever since Ulam's 1943 paper, Monte Carlo has been extremely influential and important
- Why: remarkably simple and powerful. If we can simulate efficiently from $\pi(\theta|\mathbf{X})$, we can approximate most quantities of interest for Bayesian inference
- Approximation converges fast, and does not depend on the dimension of the distribution
- **Big** challenge: simulating from $\pi(\theta|\mathbf{X})$
  - Solved by MCMC, but this complicates standard error calculation, assessment of accuracy of approximation

Outline

(4) Importance Sampling Idea

- Goal: approximate expectations with respect to $\pi(\theta|\mathbf{X})$.
  - $\mu = E_\pi(g(\theta))$, for some real-valued function $g(\cdot)$
- Simple idea: draw samples from an *importance function*, $q(\theta)$, and take weighted mean to approximate μ
- Requirement for q
  - Whenever $\pi(\theta|\mathbf{X})g(\theta) > 0$, must have $q(\theta) > 0$
- Rewrite μ as expectation with respect to q

$$
\begin{aligned}
E_\pi(g(\theta)) &= \int g(\theta)\pi(\theta|\mathbf{X})d\theta \\
&= \int \frac{g(\theta)\pi(\theta|\mathbf{X})}{q(\theta)} q(\theta)d\theta \\
&= E_q\left(\frac{g(\theta)\pi(\theta|\mathbf{X})}{q(\theta)}\right)
\end{aligned}
$$

Importance Sampling Algorithm

- Observation: If μ is written as an expectation with respect to q, can use IID Monte Carlo to approximate μ
  (1) Simulate $\theta^{(1)}, \theta^{(2)}, \ldots \overset{iid}{\sim} q(\theta)$
  (2) Approximate μ by the sample mean

$$\hat{\mu} = \sum_{i=1}^{n} \left(\frac{g(\theta^{(i)})\pi(\theta^{(i)}|\mathbf{X})}{q(\theta^{(i)})} \right) \rightarrow \mu$$

- Recall: do not usually know normalizing constant for $\pi(\theta|\mathbf{X})$
- Hence, need an approach that does not require the normalizing constant

Ratio Importance Sampling Idea

Simple: normalizing constant is an expected value, approximate it by Monte Carlo too

- Requirement for q (stronger than before)
  - Whenever $\pi(\theta|\mathbf{X}) > 0$, must have $q(\theta) > 0$
- Then, $C = \int h(\theta|\mathbf{X})d\theta = \int \frac{h(\theta|\mathbf{X})}{q(\theta)} q(\theta)d\theta = E_q\left(\frac{h(\theta|\mathbf{X})}{q(\theta)}\right)$

$$
\begin{aligned}
E_\pi(g(\theta)) &= E_q\left(\frac{g(\theta)h(\theta|\mathbf{X})}{q(\theta)}\right)/C \\
&= E_q\left(\frac{g(\theta)h(\theta|\mathbf{X})}{q(\theta)}\right)/E_q\left(\frac{h(\theta|\mathbf{X})}{q(\theta)}\right)
\end{aligned}
$$

Ratio Importance Sampling Algorithm

Approximate ratio of two expectations with respect to q with an approximation to the numerator and one for the denominator. Theory works out (Law of Large Numbers + Slutsky's Theorem)

(1) Simulate $\theta^{(1)}, \theta^{(2)}, \ldots \overset{iid}{\sim} q(\theta)$

(2) Approximate μ by

$$\hat{\mu} = \sum_{i=1}^{n} \left(\frac{g(\theta^{(i)}) h(\theta^{(i)}|\mathbf{X})}{q(\theta^{(i)})} \right) / \sum_{i=1}^{n} \left(\frac{h(\theta^{(i)}|\mathbf{X})}{q(\theta^{(i)})} \right) \rightarrow \mu$$

Importance Sampling Uses

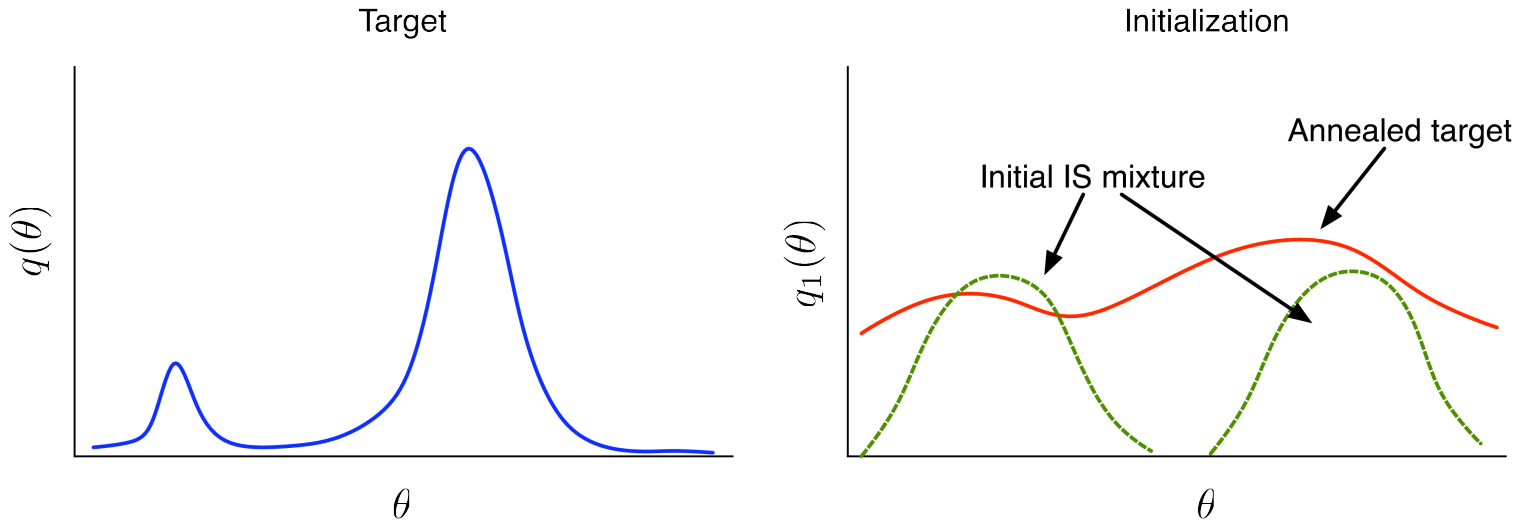
- Widely applicable/useful for approximating expectations in general, particularly marginal expectations
- Particularly adept at approximating rare event probabilities: this is where it got its name – place q mass where $g(\theta)\pi(\theta|\mathbf{X})$ is large. Can be more efficient than even sampling directly from q (IID Monte Carlo)
- Very convenient for computing expectations with respect to multiple distributions at once with the same set of samples, for instance, approximating expectations as a function of parameters of a distribution: draw single set of samples from q, reweight with respect to multiple target distributions. Also useful for prior sensitivity analysis

Constructing Importance Functions

- Important: try to make the importance function $q(\cdot)$ be heavier-tailed than target $\pi(\theta|\mathbf{X})$. This helps keep variance of approximations under control (can otherwise be infinite!)
- Art and science to this
- Nice strategies include mixture distributions, e.g. mixture of t-distributions to help match the target as well as keep the importance function reasonably heavy-tailed
- Iterative/adaptive approaches may be useful. Many papers on adaptive importance sampling and mixtures

Adaptive/iterative Methods

Initialization



(from Tom Loredo/MacKay: Information Theory, Inference, and Learning Algorithms: The Book)

Sampling Importance Resampling (SIR)

- May seem like it is the same algorithm as importance sampling, but it is not!
- Idea: Draw samples from q, then resample them with weights chosen so that the resampled values are approximately like draws from $\pi(\theta|\mathbf{X})$ (Note: in regular importance sampling, there is no resampling)
- SIR is the basis for "particle filtering": useful when $\pi(\theta|\mathbf{X})$ keeps changing, e.g. in dynamical systems
- Each time $\pi(\theta|\mathbf{X})$ changes, use a resampling step to approximate the new $\pi(\theta|\mathbf{X})$. The set of particles (samples) at each step approximately represents the current version of $\pi(\theta|\mathbf{X})$

Outline

(5) Markov chain Monte Carlo

- Monte Carlo is easy when we have i.i.d. samples from π
- Difficult to sample from π, especially high-dimensional
- More general approach: **Metropolis-Hastings (M-H) algorithm**. Use M-H to construct Markov chain $\theta^{(1)}, \theta^{(2)}, \ldots$ with stationary distribution π
- Use $\theta_i,\ i = 1, \ldots, n$ as before to approximate $E_\pi g$
- If proposals are chosen sensibly (can get everywhere in state space, no predictable "periodic" tour through space), Markov chain from M-H algorithm is *Harris-ergodic* (well behaved) $\rightarrow$ Law of Large Numbers holds if $E_\pi(g) < \infty$

$$\hat{\mu} = \sum_{i=1}^{N} g(\theta^{(i)})/N \rightarrow E_\pi g \text{ as } N \rightarrow \infty$$

A Reduced-Pain Approach to MCMC

- Use software to specify the model and let it construct the MCMC algorithm. E.g. `stan, jags, WINBUGS`
  - Fewer headaches, less time spent on coding
  - Fewer errors
- Figure out how to be careful with MCMC output (diagnostics), e.g. Eric Ford's tutorial, MCMC lab from Week 1

Writing Your Own MCMC Code

- Unfortunately, for many complicated problems you may have to write your own code. For speed, specialized functions, connecting code to other code, maybe canned software is not working well...
- Good to have some working principles/practices for doing this well
- (Unusually) I will start with MCMC diagnostics then discuss MCMC algorithm construction briefly

Being Careful With MCMC

- Approximate MCMC standard error. Can do this using `R` package `mcmcse` and function `mcse`
- Compute 95% confidence interval. Narrow enough?
- How narrow is narrow enough? This is situation dependent. For an automated approach, use effective sample size, ESS (also available in same`mcmcse` package) which counts the number of independent samples equivalent to the number of dependent Markov chain samples. Rule of thumb: make sure ESS$> 4,000$
- **Caveat:** Central Limit Theorem is not guaranteed to hold, unless the Markov chain is "fast mixing" = explores state space quickly. But ESS/MCMCse seems like a useful heuristic even when CLT doesn't hold (Flegal et al., 2008)

Being Careful With MCMC - II

- Unlike IID Monte Carlo, MCMC s.errors are not enough
  - Worry about mixing of chain (if poor, *nothing* is reliable)
  - Worry about whether the starting value is a good one. This can lead to bias
  - Stuck in a local mode? Not exploring the state space well? Especially problematic if there are well separated modes
- Root cause: Markov chain samples $\theta^{(1)}, \theta^{(2)}, \ldots$ are neither independent nor identically distributed (unless we start in stationary distribution; very rare!)
- How to address initial value issues?
  - Find initial values that are in high probability regions
  - Run really long Markov chains (as long as you have time/resources) from multiple initial values, compare results
  - Plot how approximations change with sample size. Hopefully they stabilize

MCMC Checklist

- Multiple initial values, initialize from high probability regions
- Run long chains in each case
- Check MCMC s.errors and/or ESS
- Monitor approximations and see if they stabilize. Strong check: compare density plots based on first half of chain to plots based on entire chain
- Construct fast mixing Markov chains if there are issues above
- Construct multiple MCMC algorithms separately if possible
- Run your MCMC algorithm on simulated example: good for many reasons

Constructing An MCMC Algorithm

This is a high-level sketch. Will discuss details as needed (covered elsewhere, including during the lab today)

1. Derive $h(\theta|\mathbf{X})$, unnormalized posterior. (θ may be a vector)
2. Identify blocks or components of θ that would be relatively easy to simulate jointly
3. Derive full conditional distributions for each block
4. Identify any blocks with full conditionals that have a known form. Can construct a Gibbs update; if not, construct a Metropolis-Hastings (M-H) update
5. For an M-H update, start with symmetric Normal proposal. If that does not work, explore other options
6. Run the M-H algorithm for short trial/debugging runs
7. Go through MCMC checklist and repeat until done

Outline

Choosing Appropriate Tools - I

We have a variety of methods to choose amongst. What is each algorithm good for? A rough guide, focusing on the dimension of the posterior distribution $\pi(\theta|\mathbf{X})$ (that is, dimension of θ)

- Quadrature: generally for low (1-3 dimensional) problems, though it applies to a few higher dimensional (< 20) problems
- Laplace approximation: for unimodal, roughly symmetric posterior distributions. Typically ≤ 20 dimensions. Higher-dimensional problems if normal approximation is appropriate (e.g. large sample asymptotics "kicks in")

(adapted from Tom Loredo's slides)

Choosing Appropriate Tools - II

- Importance sampling: good for ≤ 20 dimensions
  - easy to parallelize simulation/evaluation from importance function – also useful if posterior is expensive to evaluate
  - can be numerically unstable for high-dimensions
  - extensions/variants (sequential Monte Carlo): easy to parallelize some parts of these algorithms. But typically not suited for problems beyond 20 dimensions. Useful for Bayesian model choice, and some rare event problems
- MCMC: broadly applicable to high-dimensional problems, possibly hundreds to thousands of dimensions
  - not feasible if computing costs per iteration is high, e.g. expensive $\pi(\cdot|\mathbf{X})$ evaluations
  - Inherently sequential, though there are some exceptions, e.g. multi-try MCMC (Liu, Liang, and Wong, 2000) which allows for parallelly generated samples at each iteration

Other Computational Tools

Often need to go beyond the approaches discussed here

1. $f(\mathbf{X}|\theta)$ is very expensive to evaluate because
   - Based on complex model, e.g. numerical solution to system of differential equations is model mean
   - Expensive error (stats) model due to matrix computations

   This makes previous computational methods impractical
   - Alternative: approximate ("emulate"/interpolate) $f(\mathbf{X}|\theta)$ based on relatively few evaluations
   - Methods using Gaussian processes (tutorial later)
   - Only works for relatively low-dimensional (1-10) parameters
2. $f(\mathbf{X}|\theta)$ is intractable (cannot evaluate it), but can simulate realizations from it easily
   - Approximate Bayesian computing (tutorial later)

References

- Handbook of Markov chain Monte Carlo, eds. Brooks et al. (2011): both on the algorithms and lots of applications
- Markov chain Monte Carlo: Stochatic Simulation for Bayesian Inference by Gamerman and Lopes (2006)
- Art Owen notes on Importance Sampling `https://statweb.stanford.edu/~owen/mc/Ch-var-is.pdf`
- R packages for diagnostics: `coda` and `mcmcse`. Easy to read papers on Monte Carlo/MCMC standard errors Koehler et al. (2009), *American Statistician*, and Flegal et al. (2008) *Statistical Science*