Section 2.
Monte Carlo Methods

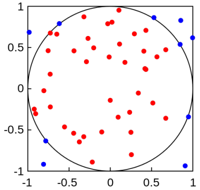
Bob Carpenter

Columbia University

Part I
Monte Carlo Integration

Monte Carlo Calculation of π

- Computing $\pi = 3.14\ldots$ via simulation is *the* textbook application of Monte Carlo methods.
- Generate points uniformly at random within the square
- Calculate proportion within circle ($x^2 + y^2 < 1$) and multiply by square's area (4) to produce the area of the circle.
- This area is π (radius is 1, so area is $\pi r^2 = \pi$)



Plot by Mysid Yoderj courtesy of Wikipedia.

Monte Carlo Calculation of π (cont.)

- R code to calcuate π with Monte Carlo simulation:

```
> x <- runif(1e6,-1,1)
> y <- runif(1e6,-1,1)

> prop_in_circle <- sum(x^2 + y^2 < 1) / 1e6

> 4 * prop_in_circle
[1] 3.144032
```

Accuracy of Monte Carlo

- Monte Carlo is *not* an approximation!
- It can be made exact to within any ϵ
- Monte Carlo draws are i.i.d. by definition
- Central limit theorem: expected error decreases at rate of

$$\frac{1}{\sqrt{N}}$$

- 3 decimal places of accuracy with sample size 1e6
- Need $100\times$ larger sample for each digit of accuracy

General Monte Carlo Integration

- MC can calculate arbitrary definite integrals,

$$\int_a^b f(x)\,dx$$

- Let d upper bound $f(x)$ in (a,b); tightness determines computational efficiency
- Then generate random points uniformly in the rectangle bounded by (a,b) and $(0,d)$
- Multiply proportion of draws (x,y) where $y < f(x)$ by area of rectangle, $d \times (b-a)$.
- Can be generalized to multiple dimensions in obvious way

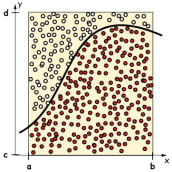


Image courtesy of Jeff Cruzan, http://www.drcruzan.com/NumericalIntegration.html

Expectations of Function of R.V.

- Suppose $f(\theta)$ is a function of random variable vector θ
- Suppose the density of θ is $p(\theta)$
  - *Warning*: θ overloaded as random and bound variable
- Then $f(\theta)$ is also random variable, with expectation

$$\mathbb{E}[f(\theta)] = \int_{\Theta} f(\theta)\, p(\theta)\, d\theta.$$

  - where Θ is support of $p(\theta)$ (i.e., $\Theta = \{\theta \mid p(\theta) > 0\}$

QoI as Expectations

- Most Bayesian quantities of interest (QoI) are expectations over the posterior $p(\theta \mid y)$ of functions $f(\theta)$
- **Bayesian parameter estimation**: $\hat{\theta}$
  - $f(\theta) = \theta$
  - $\hat{\theta} = \mathbb{E}[\theta|y]$ minimizes expected square error
- **Bayesian parameter (co)variance estimation**: $\mathrm{var}[\theta \mid y]$
  - $f(\theta) = (\theta - \hat{\theta})^2$
- **Bayesian event probability**: $\Pr[A \mid y]$
  - $f(\theta) = \mathrm{I}(\theta \in A)$

Expectations via Monte Carlo

- Generate draws $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(M)}$ drawn from $p(\theta)$
- Monte Carlo Estimator **plugs in average** for expectation:

$$\mathbb{E}[f(\theta)|y] \approx \frac{1}{M} \sum_{m=1}^{M} f(\theta^{(m)})$$

- Can be made **as accurate as desired**, because

$$\mathbb{E}[f(\theta)] = \lim_{M \to \infty} \frac{1}{M} \sum_{m=1}^{M} f(\theta^{(m)})$$

- *Reminder*: By CLT, error goes down as $1 \,/\, \sqrt{M}$

Part II

Markov Chain Monte Carlo

Markov Chain Monte Carlo

- Standard Monte Carlo draws i.i.d. draws

$$\theta^{(1)}, \ldots, \theta^{(M)}$$

  according to a probability function $p(\theta)$

- Drawing an i.i.d. sample is often impossible when dealing with complex densities like Bayesian posteriors $p(\theta|y)$

- So we use Markov chain Monte Carlo (MCMC) in these cases and draw $\theta^{(1)}, \ldots, \theta^{(M)}$ from a Markov chain

Markov Chains

- A Markov Chain is a sequence of random variables

  $$\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(M)}$$

  such that $\theta^{(m)}$ only depends on $\theta^{(m-1)}$, i.e.,

  $$p(\theta^{(m)}|y, \theta^{(1)}, \ldots, \theta^{(m-1)}) \;=\; p(\theta^{(m)}|y, \theta^{(m-1)})$$

- Drawing $\theta^{(1)}, \ldots, \theta^{(M)}$ from a Markov chain according to $p(\theta^{(m)} \mid \theta^{(m-1)}, y)$ is more tractable

- Require marginal of each draw, $p(\theta^{(m)}|y)$, to be equal to true posterior

Applying MCMC

- Plug in just like ordinary (non-Markov chain) Monte Carlo
- Adjust standard errors for dependence in Markov chain

MCMC for Posterior Mean

- Standard Bayesian estimator is posterior mean

$$\hat{\theta} \;=\; \int_{\Theta} \theta \, p(\theta|y) \, d\theta$$

  - Posterior mean minimizes expected square error

- Estimate is a conditional expectation

$$\hat{\theta} = \mathbb{E}[\theta|y]$$

- Compute by averaging

$$\hat{\theta} \;\approx\; \frac{1}{M} \sum_{m=1}^{M} \theta$$

MCMC for Posterior Variance

- Posterior variance works the same way,

$$\begin{aligned}\mathbb{E}[(\theta - \mathbb{E}[\theta \mid y])^2 \mid y] &= \mathbb{E}[(\theta - \hat{\theta})^2] \\ &\approx \frac{1}{M} \sum_{m=1}^{M} (\theta^{(m)} - \hat{\theta})^2\end{aligned}$$

MCMC for Event Probability

- Event probabilities are also expectations, e.g.,

$$\text{Pr}[\theta_1 > \theta_2] = \mathbb{E}[\text{I}[\theta_1 > \theta_2]] = \int_\Theta \text{I}[\theta_1 > \theta_2]\, p(\theta|y) d\theta.$$

- Estimation via MCMC just another plug-in:

$$\text{Pr}[\theta_1 > \theta_2] \approx \frac{1}{M} \sum_{m=1}^{M} \text{I}[\theta_1^{(m)} > \theta_2^{(m)}]$$

- Again, can be made as accurate as necessary

MCMC for Quantiles (incl. median)

- These are not expectations, but still plug in
- Alternative Bayesian estimator is posterior median
  - Posterior median minimizes expected absolute error
- Estimate as median draw of $\theta^{(1)}, \ldots, \theta^{(M)}$
  - just sort and take halfway value
  - e.g., Stan shows 50% point (or other quantiles)
- Other quantiles including interval bounds similar
  - estimate with quantile of draws
  - estimation error goes up in tail (based on fewer draws)

Part III
MCMC Algorithms

Random-Walk Metropolis

- Draw random initial parameter vector $\theta^{(1)}$ (in support)
- For $m \in 2{:}M$
  - Sample proposal from a (symmetric) jumping distribution, e.g.,
    $$\theta^* \sim \text{MultiNormal}(\theta^{(m-1)},\ \sigma\mathbf{I})$$
    where $\mathbf{I}$ is the identity matrix
  - Draw $u^{(m)} \sim \text{Uniform}(0, 1)$ and set
    $$\theta^{(m)} = \begin{cases} \theta^* & \text{if } u^{(m)} < \dfrac{p(\theta^*|y)}{p(\theta^{(m-1)}|y)} \\ \theta^{(m-1)} & \text{otherwise} \end{cases}$$

Metropolis and Normalization

- Metropolis only uses posterior in a ratio:

$$\frac{p(\theta^* \mid y)}{p(\theta^{(m)} \mid y)}$$

- This **allows** the use of **unnormalized densities**

- Recall Bayes's rule:

$$p(\theta|y) \propto p(y|\theta)\, p(\theta)$$

- Thus we only need to evaluate sampling (likelihood) and prior
  - i.e., no need to compute normalizing integral for $p(y)$,

$$\int_{\Theta} p(y|\theta)\, p(\theta) d\theta$$

Metropolis-Hastings

- Generalizes Metropolis to asymmetric proposals
- Acceptance ratio is

$$\frac{J(\theta^{(m)}|\theta^{*}) \times p(\theta^{*}|y)}{J(\theta^{*}|\theta^{(m-1)}) \times p(\theta^{(m)}|y)}$$

  where J is the (potentially asymmetric) proposal density

- i.e.,

$$\frac{\text{probabilty of being at } \theta^{*} \text{ and jumping to } \theta^{(m-1)}}{\text{probability of being at } \theta^{(m-1)} \text{ and jumping to } \theta^{*}}$$

Metropolis-Hastings (cont.)

- General form ensures equilibrium by maintaining *detailed balance*
- Like Metropolis, only requires ratios
- Many algorithms involve a Metropolis-Hastings “correction”
  - Including vanilla HMC and RHMC and ensemble samplers

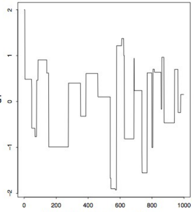
Detailed Balance & Reversibility

- Definition is measure theoretic, but applies to densities
  - just like Bayes's rule
- Assume Markov chain has stationary density $p(a)$
- Suppose $\pi(a|b)$ is density of transitioning from b to a
  - use of π to indicates different measure on Θ than p
- Detailed balance is a reversibility equilibrium condition

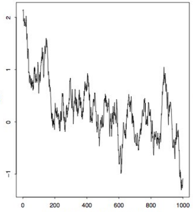
$$p(a)\,\pi(b|a) \;=\; p(b)\,\pi(a|b)$$

Optimal Proposal Scale?

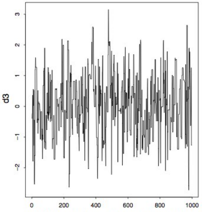
- Proposal scale σ is a free; too low or high is inefficient



(a) Proposal variance too large



(b) Proposal variance too small



(c) Proposal variance approximately optimised

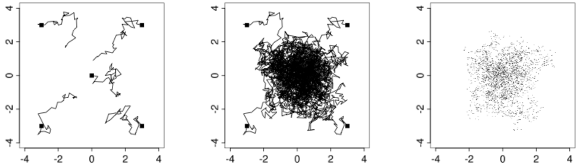
- *Traceplots* show parameter value on y axis, iterations on x
- Empirical tuning problem; theoretical optima exist for some cases

Roberts and Rosenthal (2001) Optimal Scaling for Various Metropolis-Hastings Algorithms. *Statistical Science*.

Convergence

- Imagine releasing a hive of bees in a sealed house
  - they disperse, but eventually reach equilibrium where the same number of bees leave a room as enter it (on average)
- May take many iterations for Markov chain to reach equilibrium

Convergence: Example



- Four chains with different starting points
  - *Left*: 50 iterations
  - *Center*: 1000 iterations
  - *Right*: Draws from second half of each chain

Gelman et al., *Bayesian Data Analysis*

Potential Scale Reduction ($\hat{R}$)

- Gelman & Rubin recommend M chains of N draws with **diffuse initializations**
- Measure that each chain has same posterior mean and variance
- If not, may be stuck in multiple modes or just not converged yet
- Define statistic $\hat{R}$ of chains s.t. **at convergence**, $\hat{R} \rightarrow 1$
  - $\hat{R} >> 1$ implies non-convergence
  - $\hat{R} \approx 1$ **does not guarantee convergence**
  - Only measures marginals

Split $\hat{R}$

- Vanilla $\hat{R}$ may not diagnose non-stationarity
  - e.g., a sequence of chains with an increasing parameter
- **Split $\hat{R}$**: Stan splits each chain into first and second half
  - start with M Markov chains of N draws each
  - split each in half to creates $2M$ chains of $N/2$ draws
  - then apply $\hat{R}$ to the $2M$ chains

Calculating $\hat{R}$ Statistic: Between

- M chains of N draws each
- **Between-sample variance** estimate

$$B = \frac{N}{M-1} \sum_{m=1}^{M} (\bar{\theta}_m^{(\bullet)} - \bar{\theta}_\bullet^{(\bullet)})^2,$$

where

$$\bar{\theta}_m^{(\bullet)} = \frac{1}{N} \sum_{n=1}^{N} \theta_m^{(n)} \quad \text{and} \quad \bar{\theta}_\bullet^{(\bullet)} = \frac{1}{M} \sum_{m=1}^{M} \bar{\theta}_m^{(\bullet)}.$$

Calculating $\hat{R}$ (cont.)

- M chains of N draws each
- **Within-sample variance** estimate:

$$W = \frac{1}{M} \sum_{m=1}^{M} s_m^2,$$

where

$$s_m^2 = \frac{1}{N-1} \sum_{n=1}^{N} (\theta_m^{(n)} - \bar{\theta}_m^{(\bullet)})^2.$$

Calculating $\hat{R}$ Statistic (cont.)

- **Variance** estimate:

$$\widehat{\text{var}}^{+}(\theta|y) = \frac{N-1}{N} W + \frac{1}{N} B.$$

  recall that W is within-chain variance and B between-chain

- **Potential scale reduction** statistic (“R hat”)

$$\hat{R} = \sqrt{\frac{\widehat{\text{var}}^{+}(\theta|y)}{W}}.$$

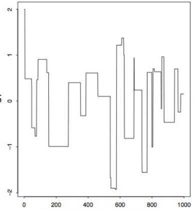
Correlations in Posterior Draws

- Markov chains typically display autocorrelation in the series of draws $\theta^{(1)}, \ldots, \theta^{(m)}$
- Without i.i.d. draws, central limit theorem *does not apply*
- Effective sample size N_{eff} divides out autocorrelation
- N_{eff} must be estimated from sample
  - Fast Fourier transform computes correlations at all lags
- Estimation accuracy proportional to

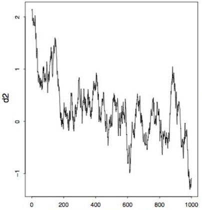
$$\frac{1}{\sqrt{N_{\text{eff}}}}$$

Reducing Posterior Correlation

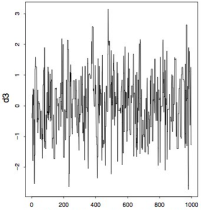
- Tuning algorithm parameters to ensure good mixing
- Recall Metropolis traceplots of Roberts and Rosenthal:



(a) Proposal variance too large



(b) Proposal variance too small



(c) Proposal variance approximately optimised

- Good jump scale σ produces good mixing and high N_{eff}

Effective Sample Size

- Autocorrelation at lag t is correlation between subseqs
  - $(\theta^{(1)}, \ldots, \theta^{(N-t)})$ and $(\theta^{(1+t)}, \ldots, \theta^{(N)})$
- Suppose chain has density $p(\theta)$ with
  - $\mathbb{E}[\theta] = \mu$ and $\mathrm{Var}[\theta] = \sigma^2$
- Autocorrelation ρ_t at lag $t \geq 0$:

$$\rho_t = \frac{1}{\sigma^2} \int_\Theta (\theta^{(n)} - \mu)(\theta^{(n+t)} - \mu)\, p(\theta)\, d\theta$$

- Because $p(\theta^{(n)}) = p(\theta^{(n+t)}) = p(\theta)$ at convergence,

$$\rho_t = \frac{1}{\sigma^2} \int_\Theta \theta^{(n)}\, \theta^{(n+t)}\, p(\theta)\, d\theta$$

Estimating Autocorrelations

- Effective sample size (N draws in chain) is defined by

$$N_{\text{eff}} \; = \; \frac{N}{\sum_{t=-\infty}^{\infty} \rho_t} \; = \; \frac{N}{1 + 2 \sum_{t=1}^{\infty} \rho_t}$$

- Estimate in terms of variograms (M chains) at lag t
  - Calculate with fast Fourier transform (FFT)

$$V_t = \frac{1}{M} \sum_{m=1}^{M} \left(\frac{1}{N_m - t} \sum_{n=t+1}^{N_m} \left(\theta_m^{(n)} - \theta_m^{(n-t)} \right)^2 \right)$$

- Adjust autocorrelation at lag t using cross-chain variance as

$$\hat{\rho}_t = 1 - \frac{V_t}{2 \, \widehat{\text{var}}^{+}}$$

- If not converged, $\widehat{\text{var}}^{+}$ overestimates variance

Estimating N_{eff}

- Let T' be first lag s.t. $\rho_{T'+1} < 0$,
- Estimate autocorrelation by

$$\hat{N}_{\text{eff}} = \frac{MN}{1 + \sum_{t=1}^{T'} \hat{\rho}_t}.$$

- NUTS avoids negative autocorrelations, so first negative autocorrelation estimate is reasonable
- For basics (not our estimates), see

  Charles Geyer (2013) Introduction to MCMC. In *Handbook of MCMC*.

  (free online at http://www.mcmchandbook.net/index.html)

Gibbs Sampling

- Draw random initial parameter vector $\theta^{(1)}$ (in support)
- For $m \in 2{:}M$
    - For $n \in 1{:}N$:
        - draw $\theta_n^{(m)}$ according to conditional

          $$p(\theta_n | \theta_1^{(m)}, \ldots, \theta_{n-1}^{(m)}, \theta_{n+1}^{(m-1)}, \ldots, \theta_N^{(m-1)}, y).$$
- e.g, with $\theta = (\theta_1, \theta_2, \theta_3)$:
    - draw $\theta_1^{(m)}$ according to $p(\theta_1 | \theta_2^{(m-1)}, \theta_3^{(m-1)}, y)$
    - draw $\theta_2^{(m)}$ according to $p(\theta_2 | \theta_1^{(m)}, \theta_3^{(m-1)}, y)$
    - draw $\theta_3^{(m)}$ according to $p(\theta_3 | \theta_1^{(m)}, \theta_2^{(m)}, y)$

Generalized Gibbs

- “Proper” Gibbs requires conditional Monte Carlo draws
  - typically works only for conjugate priors
- In general case, may need to use less efficient conditional draws
  - Slice sampling is a popular general technique that works for discrete or continuous θ_n (JAGS)
  - Adaptive rejection sampling is another alternative (BUGS)
  - Very difficult in more than one or two dimensions

Sampling Efficiency

- We care only about N_{eff} per second
- Decompose into
  1. Iterations per second
  2. Effective sample size per iteration
- Gibbs and Metropolis have high iterations per second (especially Metropolis)
- But they have low effective sample size per iteration (especially Metropolis)
- Both are particular weak when there is high correlation among the parameters in the posterior

Hamiltonian Monte Carlo & NUTS

- Slower iterations per second than Gibbs or Metropolis
- Much higher effective sample size per iteration for complex posteriors (i.e., high curvature and correlation)
- Overall, much higher N_{eff} per second

- Details in the next talk . . .
- Along with details of how Stan implements HMC and NUTS