Ch10.
Introduction to Bayesian computation

Jihu Lee

October 13, 2020

Seoul National University

Table of Contents

1. Introduction
2. Numerical integration
3. Distributional approximations
4. Direct simulation and rejection sampling
5. Importance sampling
6. How many simulation draws are needed?
7. Computing environments
8. Debugging Bayesian computing

Table of Contents

1. Introduction
2. Numerical integration
3. Distributional approximations
4. Direct simulation and rejection sampling
5. Importance sampling
6. How many simulation draws are needed?
7. Computing environments
8. Debugging Bayesian computing

Introduction

- Steps of Bayesian computation
  - posterior $p(\theta|y)$
  - predictive $p(\tilde{y}|y)$
- Complicated or unusual models or in high dimensions need more elaborate algorithms
- Ch10 gives a brief summary of procedures to approximately evaluate integrals

Normalized and unnormalized densities

- target distribution $p(\theta|y)$
- We call a easily computable funtion $q(\theta|y)$ unnormalized density, if $\frac{q(\theta|y)}{p(\theta|y)}$ is a constant only depends on y
- ex) in usual bayes rule, $q(\theta|y)$ can be $p(\theta)p(\theta|y)$

Log densities

- We can use log densities to avoid overflow or underflow when possible
- We can also take exponentiation only when necessary
    - It should be taken as late as possible

Table of Contents

1. Introduction
2. Numerical integration
3. Distributional approximations
4. Direct simulation and rejection sampling
5. Importance sampling
6. How many simulation draws are needed?
7. Computing environments
8. Debugging Bayesian computing

Numerical integration

- Numerical integration = Quadrature
- Methods in which integral over continous functions is evaluated by computing the value of function at finite number of points
  - Deterministic methods
    - Trapezoidal rule
    - Simpson's rule
  - Simulation methods
    - Monte Carlo methods
- Method with more points gives more accurate approximation

Posterior expectation of $h(\theta)$

- Posterior expectation of any function $h(\theta)$ is give as

$$E(h(\theta)|y) = \int h(\theta)p(\theta|y)d\theta$$

- Conversely, we can express any integral over the space of θ as $E(h(\theta)|y)$ by defining proper $h(\theta)$
- for θ^s from $p(\theta|y)$, take

$$E(h(\theta)|y) \simeq \frac{1}{S}\sum_{s=1}^{S} h(\theta^s)$$

  (in Ch 10.5)

- Hard to draw from the posterior/$h(\theta^s)$ varies too much -> needs other sampling methods

Simulation methods

- $E(h(\theta)|y) \simeq \frac{1}{S}\sum_{s=1}^{S} h(\theta^s)$
- More accuracy when more samples
- Basic Monte Carlo methods (MC) <- independent samples (Ch 10.3-4)
- Markov Chain Monte Carlo methods (MCMC) <- dependent samples (Ch 11-12)
- Combining general ideas could give more efficient computation

Deterministic methods

- Basic version

$$E(h(\theta)|y) = \int h(\theta)p(\theta|y)d\theta \sim \frac{1}{S}\sum_{s=1}^{S} w_s h(\theta^s)p(\theta^s|y)$$

- More elaborate rules use local polynomials, which gives more accuracy
- (typically) Gives lower variance than simulation methods, but hard to choose point locations

Table of Contents

1. Introduction
2. Numerical integration
3. Distributional approximations
4. Direct simulation and rejection sampling
5. Importance sampling
6. How many simulation draws are needed?
7. Computing environments
8. Debugging Bayesian computing

Distributional approximations

- Distributional approximations approximates the posterior with some simpler parameteric distribution
- ex) Normal approximation(Ch 4), Advanced approximation(Ch 13)

Crude estimation by ignoring some information

- Rough estimation of the location of the target distribution is recommended before starting the approximation
- Ex1) Hierarchical model
  - Roughly estimate the main parameters γ
  - First estimating the hyperparameters ϕ, then use the conditional posterior distribution $p(\gamma|\phi, y)$

Crude estimation by ignoring some information

- Ex2) Educational testing analysis (Ch 5.5)
  - The school effects θ_j can be crudely estimated by the data y_j
- When some data are missing, it is good to simplistically imputing the missing values based on available data
- Crude inferences are useful for comparison with later results
- If the rough estimate differs greatly from the results of the full analysis, the latter may well have errors

Table of Contents

1. Introduction
2. Numerical integration
3. Distributional approximations
4. Direct simulation and rejection sampling
5. Importance sampling
6. How many simulation draws are needed?
7. Computing environments
8. Debugging Bayesian computing

Direct simulation and rejection sampling

- For simple non-hierarchical models, it is easy to draw from the posterior directly especially if conjugate prior has assumed
- If the model is more complicated, we have to simulate by parts
- Basic samplings are introduced in Appendix A

Simulating from predictive distributions

- Once we have a sample from the posterior $p(\theta|y)$, it is typically easy to draw from the predictive distribution
- For each draw of θ, just draw one $\tilde{y}$ from the predictive $p(\tilde{y}|\theta)$
- Set of $\tilde{y}$'s characterizes the posterior predictive distribution

Rejection sampling

- Rejection sampling can be used when we want to draw a single random value from $p(\theta|y)$ or $q(\theta|y)$
- First, we have to define $g(\theta)$ for all θ for which $p(\theta|y) > 0$ with following properties
  - $g(\theta)$ has a finite integral
  - $\frac{p(\theta|y)}{g(\theta)} \leq M$ for all θ, known constant M

Rejection sampling - Algorithm

1. Sample θ at random from the probability density proportional to $g(\theta)$
2. With probability $\frac{p(\theta|y)}{Mg(\theta)}$, accept θ as a draw from p.
   If the drawn θ is rejected, return to step 1

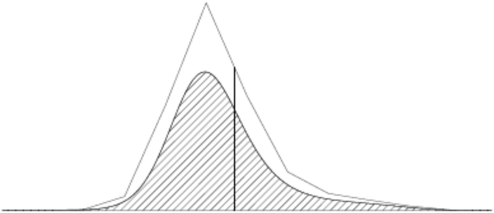

Figure 1: Rejection sampling
Top curve: $Mg(\theta)$, bottom curve: $p(\theta|y)$

Rejection sampling

- Ideal situation is that $g(\theta) \propto p(\theta|y)$ and have suitable M, which makes rejection not be occured
- If $g(\theta)$ is nearly proportional to $p(\theta|y)$, the bound M must be set so large that almost all draws will be rejected
- Self-monitoring: if the method is not working efficiently, few simulated draws will be accepted
- Usage) some fast methods for sampling from standard univariate distributions, generic truncated multivariate distributions

Table of Contents

1. Introduction
2. Numerical integration
3. Distributional approximations
4. Direct simulation and rejection sampling
5. Importance sampling
6. How many simulation draws are needed?
7. Computing environments
8. Debugging Bayesian computing

Importance sampling

- Importance sampling is a method related to rejection sampling and a precursor to the Metropolis algorithm (Ch 11)
- Let $g(\theta)$ be a approximated distribution to the target that we can generate random draw from

Importance sampling

- Suppose we are interested in $E(h(\theta)|y)$, express it as

$$E(h(\theta)|y) = \frac{\int h(\theta)q(\theta|y)d\theta}{\int q(\theta|y)d\theta} = \frac{\int [h(\theta)q(\theta|y)/g(\theta)]g(\theta)d\theta}{\int [q(\theta|y)/g(\theta)]g(\theta)d\theta}$$

- This can be estimated using S draws $\theta^1, ..., \theta^S$ from $g(\theta)$ as

$$\frac{\frac{1}{S}\sum_{s=1}^{S} h(\theta^s)w(\theta^s)}{\frac{1}{S}\sum_{s=1}^{S} w(\theta^s)}$$

  where $w(\theta^s) = \frac{q(\theta^s|y)}{g(\theta^s)}$ (importance ratios / importance weights)

Importance sampling

- If $g(\theta)$ can be chosen s.t. $\frac{hq}{g}$ is roughly constant, then fairly precise estimates can be obtained
- If the importance ratios vary substantially, then the sampling is not useful
- The worst scenario occurs when the importance ratios are small with high probability, and are huge with low probability
- It happens when hq has wide tails compared to g as a function of θ

Accuracy and efficiency of importance sampling estimates

- Large importance ratios have more influence to the approximation than the small ones
- If the variance of the weights are finite, the effective sample size can be estimated as follows

$$S_{eff} = \frac{1}{\sum_{s=1}^{S}(\tilde{w}(\theta^s))^2}$$

  where $\tilde{w}(\theta^s) = \frac{w(\theta^s)S}{\sum_{s'=1}^{S} w(\theta^{s'})}$ are normalized weights
- Few huge weights -> small S_{eff}, occasional huge weights -> estimate is not good

Importance resampling (SIR)

- Importance resampling is used to obtain independent samples with equal weights
- Once $\theta^1, ..., \theta^S$ draws from the approximate distribution g have been sampled, a sample of k draws can be simulated as follows

1. Sample a value θ from the set $\theta^1, ..., \theta^S$, where the probability of sampling each θ^s is proportional to the weight $w(\theta^s)$
2. Sample a next k values as same, but excluding the already sampled ones

Importance resampling

- Reason for exclusion
  - If weights are moderate, then inclusion/exclusion doesn't matter
  - If few weights are huge, then few values can be sampled repeatedly if exclusion has not implemented

Uses of importance sampling in Bayesian computation

1. It can be used to improve analytic posterior approximation (Ch 13)
   - If importance sampling doesn't yield an accurate approximation, then SIR can be helpful to obtain starting points for an iterative simulation of posterior distribution
2. It is useful when considering mild changes in the posterior distribution

Table of Contents

1. Introduction
2. Numerical integration
3. Distributional approximations
4. Direct simulation and rejection sampling
5. Importance sampling
6. How many simulation draws are needed?
7. Computing environments
8. Debugging Bayesian computing

How many simulation draws are needed?

- Bayesian inferences are usually most conveniently summarized by random draws from the posterior distributions

1. Percentiles of the posterior distribution of univariate estimand
    - Reporting the 2.5%, 25%, 50%, 75%, 97.5% points of the sampled distribution provides a 50%, 95% posterior interval
2. Make inferences about predictive quantities
    - Given each θ^s, we can sample predictive $\tilde{y}^s \sim p(\tilde{y}|\theta^s)$
3. Given each simulation θ^s, we can simulate a replicated dataset $y^{rep\ s}$ to check the model by comparing the data to these posterior predictive replications

How many simulation draws are needed?

- Our goal in Bayesian computation is obtaining a set of independent draws θ^s from the posterior distribution, with enough draws S
- In general,
    - posterior median, probability near 0.5, low-dimensional summaries need less simulations
    - posterior means, probability of rare events, high-dimensional summaries need more simulations

Table of Contents

The bugs family of programs

- Bayesian inference using Gibbs sampling -> bugs
- A combination of Gibbs sampling, Metropolis algorithm, and slice sampling can provide inference for variety of models when run for a sufficiently long time

Other environments

- Stan : uses Hamiltonian Monte Carlo mehtod (Ch 12.4)
- mcsim : C program that implements Gibbs and Metropolis for differential equation systems
- PyMC : a suite of routines in Python
- HBC : for discrete-parameter models

Table of Contents

1. Introduction
2. Numerical integration
3. Distributional approximations
4. Direct simulation and rejection sampling
5. Importance sampling
6. How many simulation draws are needed?
7. Computing environments
8. Debugging Bayesian computing

Debugging using fake data

- Use when a model is particularly complicated, or its inferences are unexpected enough to be not necessarily believable

1. Pick a reasonable value for the true parameter vector θ, which shouold be a random draw from the prior distribution
2. If the model is hierarchical, then perform Step1 for hyperparameters, then draw the others from the prior distribution conditional on the specified hyperparameters
3. Simulate a large fake dataset y^{fake} from the data distribution $p(y|\theta)$
4. Perform posterior inference about θ from $p(\theta|y^{fake})$
5. Compare the posterior inferences to the true θ

Debugging using fake data

- To check that inferences are correct on average, a residual plot is helpful
- For each scalar θ_j, define predicted value as the average of the posterior simulations of θ_j, and the error as the true θ_j minus the predicted value
- If correct, the errors would approximately have zero mean
- If a model has only few parameters, one can get the same effect by performing many fake-data simulations

Model checking and convergence checking as debugging

- In practice, when a model grossly misfits the data, it is often because of a computing error
- Similarly, poor convergence of anm iterative simulation algorithm can sometimes occur from programming errors
- A useful strategy is simplifying
    - remove parameters / fix parameter values
    - use highly informative prior
    - unlink hierarchy