Bayesian Estimation using MCMC

Kevin M. Esterling (UC – Riverside)

Presentation to GradQuant

October 9, 2017

Why Bayesian inference?

There are philosophical reasons rooted in fundamental beliefs about the purpose of science. But this workshop will focus on the *practical* reasons to be Bayesian, given computational methods:

- Can approximate frequentist results (or "regularize" results, depending on your audience)
- But Bayesian methods are much more flexible and/or computationally faster to solve arbitrarily complex models
- Easier to state uncertainty about arbitrary functions of parameters
- Natural way to multiply impute missing data (additional parameters to estimate)

What is Bayesian inference?

You are probably already familiar with frequentist inference . . .

Philosophy of Frequentist Inference

Probability is an objective property of the external/natural world

- Probability is an inherent property of a coin or dice or population, etc. but cannot ever be observed
- Inferring this property requires repeated observation; e.g., 1,000 coin flips
- Reporting results is awkward; can't report probability of a heads but instead the probability that an interval will cover this property

What is Bayesian inference?

Philosophy of Bayesian Inference

Bayesian inference posits that probability is best conceived of as a *subjective belief*

The goal of research is to change **beliefs** about properties of the world; Bayesian analysis is a way to inform your audience how they rationally should change their beliefs after observing data

Bayes Rule:

$$\text{Posterior Beliefs} = \frac{\text{Prior Beliefs} \times \text{Data Likelihood}}{\text{Probabilty of the Data}}$$

Bayes rule illustration: testing for a disease

You run a test on a patient and you get an ALERT! Here is what we know about the test procedure.

- $D =$ "The patient has the disease"
- $D^C =$ "The patient does not have the disease"
- $A =$ "The test gave an alert"

- Incidence of the disease in a population, $p(D) = 0.02$ **("Prior")**
- Probability of the test giving an alert given the presence of the disease, $p(A|D) = 0.95$ **("Likelihood")**
- Probability of the test giving an alert in the absence of the disease (false positive), $p(A|D^c) = 0.03$

Bayes rule: testing for a disease (cont.)

If test is positive (Alert occurs) use Bayes' Rule:

$$p(D|A) = \frac{prior \times likelihood}{prob\ \ of\ \ alert}$$

$$= \frac{p(D)p(A|D)}{p(D)p(A|D)+p(D^c)p(A|D^c)}$$

$$= \frac{0.02\times 0.95}{0.02\times 0.95+0.98\times 0.03} = 0.38$$

$\Rightarrow$ updated subjective probability that patient has the disease

- Patient went from subjective probability of 0.02 to 0.38 of having the disease
- Physician learned a lot from only one observation (!)
- But, just because test is positive does not mean the patient certainly has the disease

Philosophical benefits of Bayesian analysis

If you agree with the subjective view of probability, cool things happen:

- A single observation can be quite meaningful
- Reporting results is intuitive; e.g., probability the patient has the disease (or, probability of a heads; probability $\beta_1 \leq 0$)
- We have beliefs/information about probabilities before we observe data, so that information can be incorporated

Doing Bayesian statistical analysis

General form of Bayes rule for statistical modeling:

$$p(\theta|y) = \frac{p(\theta)p(y|\theta)}{p(y)}$$

In words, the posterior density (beliefs after seeing the data) is proportional to the prior density (beliefs before seeing the data) times the likelihood of observing the data given those prior beliefs, divided by a normalizing constant.

We can drop the normalizing constant that makes the posterior a true probability density

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

Doing Bayesian statistical analysis, closed-form solutions

The easiest way to derive the posterior *analytically* is using "conjugate" priors

- A conjugate prior for a likelihood yields a posterior in the same form as the prior
- Example, conjugate prior for a binomial distribution is the beta distribution; or normal-normal . . .

Closed-form Example: Conjugate Normal Prior

Analytical solution for conjugate normal prior

Let $y_i \overset{iid}{\sim} N(\mu, \sigma^2), i = 1, \ldots, n$, with σ^2 known, and $\mathbf{y} = (y_1, \ldots, y_n)'$. If $\mu \sim N(\mu_0, \sigma_0^2)$ is the prior density for μ, the μ has posterior density,

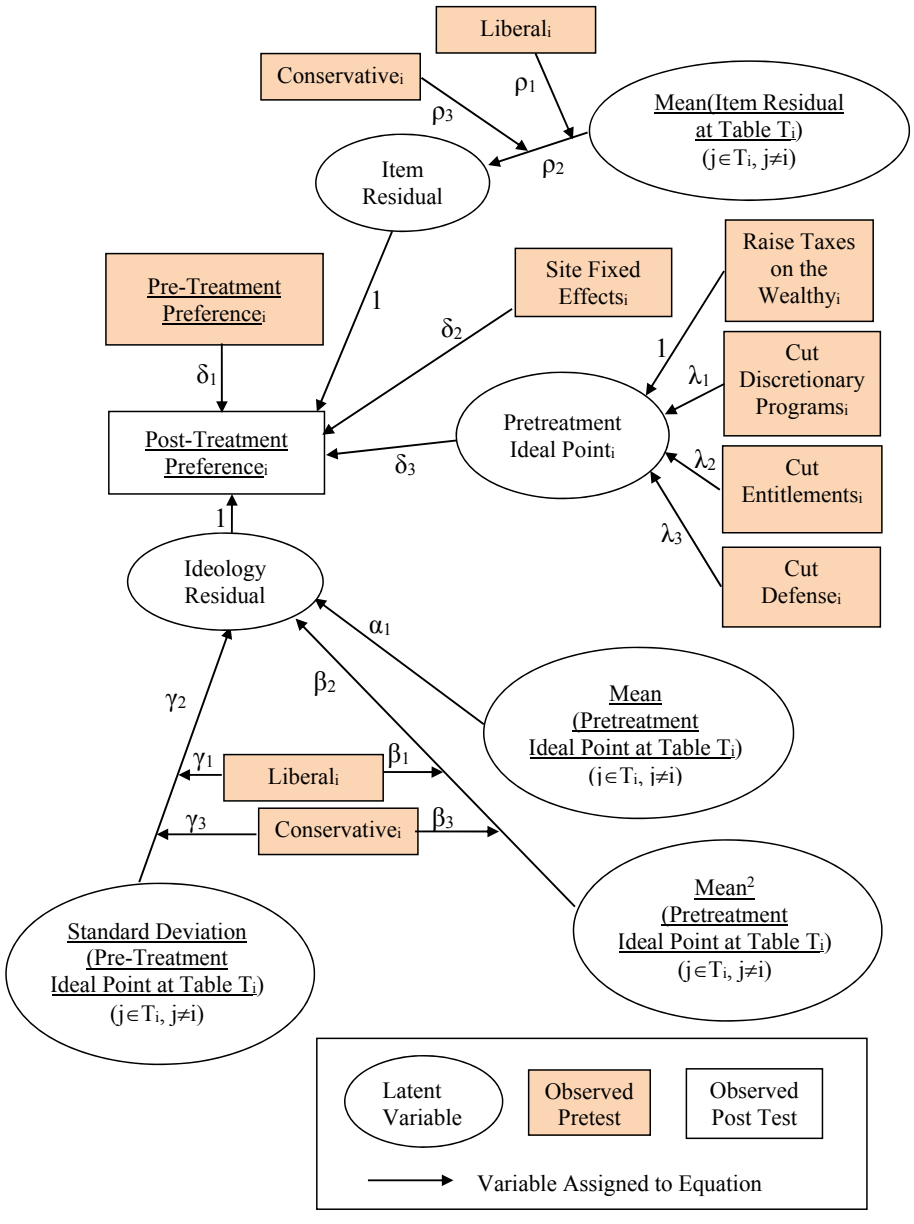
$$\mu|\mathbf{y} \sim N\left(\frac{\mu_0\sigma_0^{-2} + \overline{y}\frac{n}{\sigma^2}}{\sigma_0^{-2} + \frac{n}{\sigma^2}}, \left(\sigma_0^{-2} + \frac{n}{\sigma^2}\right)^{-1}\right)$$

Results:

- *Posterior mean* is a weighted average of the prior and data
- *Posterior variance* is the sum of the prior precision and the data precision
- Note: Bayesian and MLE converge with diffuse priors and/or lots of data

Doing Bayesian statistical analysis in the real world

Sadly, most modeling problems do not lend themselves to closed-form solutions, especially complex multilevel (random effect) models . . .


Liberal$_i$
Conservative$_i$
ρ_1
ρ_3
ρ_2
Mean(Item Residual at Table T$_i$) (j∈T$_i$, j≠i)
Item Residual
Pre-Treatment Preference$_i$
1
Site Fixed Effects$_i$
δ_2
Raise Taxes on the Wealthy$_i$
1
λ_1
Cut Discretionary Programs$_i$
δ_1
Pretreatment Ideal Point$_i$
Post-Treatment Preference$_i$
δ_3
λ_2
Cut Entitlements$_i$
1
λ_3
Cut Defense$_i$
Ideology Residual
α_1
β_2
γ_2
Mean (Pretreatment Ideal Point at Table T$_i$) (j∈T$_i$, j≠i)
γ_1
Liberal$_i$
β_1
γ_3
Conservative$_i$
β_3
Standard Deviation (Pre-Treatment Ideal Point at Table T$_i$) (j∈T$_i$, j≠i)
Mean^2 (Pretreatment Ideal Point at Table T$_i$) (j∈T$_i$, j≠i)
Latent Variable
Observed Pretest
Observed Post Test
Variable Assigned to Equation

Doing Bayesian statistical analysis in the real world

Bayesian computational methods allow you to solve arbitrary, complex models much more flexibly and/or computationally faster

Computational Bayesian statistics: MCMC

"Bayesian estimation using Gibbs sampling" (`WinBUGS`, `OpenBUGS`, `JAGS`, `Stan`, etc.)

- Gibbs sampling: sample an estimate from a candidate posterior distribution for each parameter, conditional on the current estimate of all other parameters
- MCMC = "Markov Chain Monte Carlo" = run the Gibbs sampler repeatedly until the parameters estimates converge to the posterior distribution
  - Start at an arbitrary set of initial values
  - Discard "burn-in period" draws
  - Save and analyze "stationary period" draws

Computational Bayesian statistics: MCMC

Table: Simulated Posterior Distribution

	β_0	β_1	$Y[2]$
Burn-in Period			
t_0	20	-200	12
t_1	17	-105	65
t_3	2	-2	99
t_4	0.9	1.5	86
...			
t_{9997}	0.7	1.7	87
t_{9998}	0.8	1.8	89
t_{9999}	0.6	1.7	95
t_{10000}	0.6	2.0	91

Computational Bayesian statistics: MCMC

Table: Simulated Posterior Distribution

	β_0	β_1	$Y[2]$
Stationary Period			
t_{10001}	0.7	1.9	89
t_{10002}	0.6	1.5	87
t_{10003}	0.8	1.6	89
t_{10004}	0.5	1.8	83
t_{10005}	0.7	2.1	99
...			
t_{10997}	0.4	2.2	87
t_{10998}	0.7	1.7	97
t_{10999}	0.6	1.9	99
t_{11000}	0.9	1.5	102

Computational Bayesian statistics: MCMC (cont.)

Result is a simulated posterior distribution: computational approximation of the posterior

- The vector of draws post-convergence for each parameter is the marginal posterior distribution
- Summarize (mean, SD, 95% intervals) and plot densities
- Trivial to create sampling distributions of functions of parameters ($\widehat{\frac{ln(\beta_0)}{1+sin(\beta_1)}}$)
- Natural way to impute missing data for correct standard errors

Computational Bayesian statistics: MCMC (even still cont.)

MCMC procedure

1. Specify model (likelihood and priors) with `WinBUGS` code
2. Load data and compile model
3. Provide initial values for parameters, latent variables, missing data
4. Run model for an initial "burn-in" period until MCMC converges on the posterior distribution
5. Save a sample of draws for parameters of interest
6. Summarize marginal distributions, plots, statistical tests

Model of the mean

Likelihood:

$$\text{mass}_i \sim \phi(\mu, \tau) \Big\} 1 \leq i \leq \text{n.obs} \qquad \text{IID assumption}$$

Priors:

$$\mu \sim \text{dunif}(0, 5000) \qquad \text{Flat postive prior ("informative" prior!)}$$
$$\tau = \frac{1}{\sigma^2} \qquad \text{Precision is inverse of variance}$$
$$\sigma^2 = \sigma \times \sigma \qquad \text{Define variance in model}$$
$$\sigma \sim \text{dunif}(0, 100) \qquad \text{Flat positive prior}$$

Let's run this model in `OpenBUGS` ...

Model of the mean

Likelihood:

$$\text{mass}_i \sim \phi(\mu, \tau) \Big\} 1 \leq i \leq \text{n.obs} \qquad \text{IID assumption}$$

Priors:

$$\mu \sim \text{dunif}(0, 5000) \qquad \text{Flat postive prior ("informative" prior!)}$$
$$\tau = \frac{1}{\sigma^2} \qquad \text{Precision is inverse of variance}$$
$$\sigma^2 = \sigma \times \sigma \qquad \text{Define variance in model}$$
$$\sigma \sim \text{dunif}(0, 100) \qquad \text{Flat positive prior}$$

Let's run this model in `OpenBUGS` ...

Model of the mean

Likelihood:

$$\text{mass}_i \sim \phi(\mu, \tau) \Big\} 1 \leq i \leq \text{n.obs} \qquad \text{IID assumption}$$

Priors:

$$\mu \sim \text{dunif}(0, 5000) \qquad \text{Flat postive prior ("informative" prior!)}$$
$$\tau = \frac{1}{\sigma^2} \qquad \text{Precision is inverse of variance}$$
$$\sigma^2 = \sigma \times \sigma \qquad \text{Define variance in model}$$
$$\sigma \sim \text{dunif}(0, 100) \qquad \text{Flat positive prior}$$

Let's run this model in `OpenBUGS` ...

Model of the mean

Likelihood:

$$\text{mass}_i \sim \phi(\mu, \tau) \Big\} 1 \leq i \leq \text{n.obs} \qquad \text{IID assumption}$$

Priors:

$$\mu \sim \text{dunif}(0, 5000) \qquad \text{Flat postive prior ("informative" prior!)}$$
$$\tau = \frac{1}{\sigma^2} \qquad \text{Precision is inverse of variance}$$
$$\sigma^2 = \sigma \times \sigma \qquad \text{Define variance in model}$$
$$\sigma \sim \text{dunif}(0, 100) \qquad \text{Flat positive prior}$$

Let's run this model in `OpenBUGS` ...

Model of the mean

Likelihood:

$$\text{mass}_i \sim \phi(\mu, \tau) \Big\} 1 \leq i \leq \text{n.obs} \qquad \text{IID assumption}$$

Priors:

$$\mu \sim \text{dunif}(0, 5000) \qquad \text{Flat postive prior ("informative" prior!)}$$
$$\tau = \frac{1}{\sigma^2} \qquad \text{Precision is inverse of variance}$$
$$\sigma^2 = \sigma \times \sigma \qquad \text{Define variance in model}$$
$$\sigma \sim \text{dunif}(0, 100) \qquad \text{Flat positive prior}$$

Let's run this model in `OpenBUGS` ...

Model of the mean

Likelihood:

$$\text{mass}_i \sim \phi(\mu, \tau) \Big\} 1 \leq i \leq \text{n.obs} \qquad \text{IID assumption}$$

Priors:

$$\mu \sim \text{dunif}(0, 5000) \qquad \text{Flat postive prior ("informative" prior!)}$$
$$\tau = \frac{1}{\sigma^2} \qquad \text{Precision is inverse of variance}$$
$$\sigma^2 = \sigma \times \sigma \qquad \text{Define variance in model}$$
$$\sigma \sim \text{dunif}(0, 100) \qquad \text{Flat positive prior}$$

Let's run this model in `OpenBUGS` ...

Model of the mean

Likelihood:

$$\text{mass}_i \sim \phi(\mu, \tau) \Big\} 1 \leq i \leq \text{n.obs} \qquad \text{IID assumption}$$

Priors:

$$\begin{aligned}
\mu &\sim \text{dunif}(0, 5000) && \text{Flat postive prior ("informative" prior!)} \\
\tau &= \frac{1}{\sigma^2} && \text{Precision is inverse of variance} \\
\sigma^2 &= \sigma \times \sigma && \text{Define variance in model} \\
\sigma &\sim \text{dunif}(0, 100) && \text{Flat positive prior}
\end{aligned}$$

Let's run this model in `OpenBUGS` ...

Common problems and some advice

- Always run multiple chains (usually three) in order to test convergence using BGR diagnostic
  - BGR diagnostic assesses within-to-between chain variance (assumes overdispersed/random initial values)
  - Consider both mathematical and empirical identification (just because you can write it down does not mean you should estimate it)
  - Best to start with simple model and build up complexity
- Assess burnin period, mixing carefully
- Be sure there are no missing data on RHS
- Read the manual; and Gelman and Hill (2006) is a great resource for multilevel modeling
- Learn scripting language

Using OpenBUGS with R

In practice, you want to store your data and analyze/graph results within R (or Stata or SAS etc.)

- Once you know how to use OpenBUGS you can read documentation to these R packages:
  - R2OpenBUGS, BRugs = Interact with OpenBUGS within R
  - CODA = Suite of tools to assess convergence and describe results
  - BRugs installs/loads all three
- Prepare data and inits text files to read directly into OpenBUGS
- Read OpenBUGS output as MCMC object for use in CODA and presenting results
- Call OpenBUGS from R for automating Bayesian analysis