Bayesian Data Analysis in R

Joshua French

University of Colorado Denver

Denver R Users Group

April 17, 2012

Bayesian Statistics

Bayesian statistics is a branch of statistics in which Bayes' rule is used to update beliefs about our parameter(s) of interest using our prior beliefs and the observed data.

Practically, it is different from "traditional" statistics in that:

- Parameters are random quantities.
- We assign distributions to our parameters.
- Our ultimate goal (essentially) is to determine the **posterior distribution** of our parameter(s) of interest conditional on the observed data.

Bayesian statistics is becoming increasingly popular because it provides a unified framework for parameter estimation and prediction (and many other reasons).

Bayesian statistics involves a ton of computing related to statistical distributions, so R is a fantastic tool for performing Bayesian inference.

Quick Notation

- $\boldsymbol{\theta}$ denotes unobservable random quantities or parameters of interest.
- $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$ will denote the observed data.
- $p(\boldsymbol{y}|\boldsymbol{\theta})$ denotes the sampling or data distribution.
- $p(\boldsymbol{\theta})$ denotes our prior distribution (a distribution summarizing our beliefs about $\boldsymbol{\theta}$ before we look at the data).

Finding the Posterior Distribution

Bayes' rule states that the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{y})$ may be found as

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

where $p(\boldsymbol{y})$ is the marginal distribution of $\boldsymbol{y}$ given by

$$p(\boldsymbol{y}) = \int_{\boldsymbol{\theta}} p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})\, d\boldsymbol{\theta}.$$

In practice, we only need to work with $p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})$ to do Bayesian inference.

Simple Example

Placenta previa is a condition in which the placenta of an unborn child is implanted very low in the uterus, obstructing the child from a normal vaginal delivery. An early study concerning the sex of placenta previa births in Germany found that of a total of 980 births, 437 were female. How strong is the evidence that the proportion of female births in the population of placenta previa births is less than 0.485 (the proportion in the general population)?

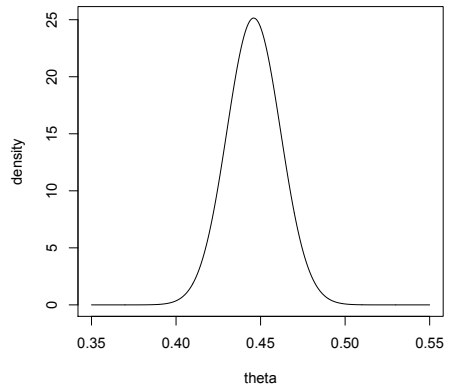
We assume:

- The data are from a Binomial($n = 980$, prob $= \theta$) sampling distribution. Thus, $p(y|\theta) = \binom{980}{437}\theta^{437}(1-\theta)^{980-437}$
- The prior distribution for θ is Uniform(0, 1). Thus, $p(\theta) = 1$.

Thus, the posterior distribution is:

$$\begin{aligned} p(\theta|y) &\propto p(y|\theta)p(\theta) \\ &\propto \theta^{437}(1-\theta)^{543} \\ &\propto \text{Beta}(438,544). \end{aligned}$$

Posterior density for θ:



MCMC Methods

It is often (typically?) not possible to determine the posterior distribution analytically, so we can use simulation techniques, such as Markov chain Monte Carlo methods, to approximate the posterior distribution.

We draw values of θ from approximate distributions and then correct those draws to approximate the posterior distribution $p(\theta|y)$.

The Gibbs Sampler

The Gibbs sampler, originating with Geman and Geman (1984), is the most widely used MCMC method.

Assume our parameter vector $\boldsymbol{\theta}$ has d parts, i.e., $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d)$.

Each iteration of the Gibbs sampler cycles through the d components of the parameter vector θ, drawing each part conditional on the value of the others.

- Going through each component of the d components is a **cycle**.

At each iteration t, an ordering of the d subvectors of θ is chosen and, in turn, each $\theta_j^{(t)}$ is sampled from the conditional distribution given all the other components:

$$p\left(\theta_j \middle| \theta_{-j}^{(t-1)}, y\right),$$

where $\theta_{-j}^{(t-1)}$ represents all d components of θ EXCEPT θ_j, at their current values

$$\theta_{-j}^{(t-1)} = \left(\theta_1^{(t)}, \ldots, \theta_{j-1}^{(t)}, \theta_{j+1}^{(t-1)}, \ldots, \theta_d^{(t-1)}\right).$$

The distribution $p\left(\theta_j \middle| \theta_{-j}^{(t-1)}, y\right)$ is known as the **full conditional distribution** of θ_j.

Each subvector θ_j is updated conditional on the LATEST values of the other components.

- This will be iteration t for already updated parameters, and iteration $t-1$ for the others.

Gibbs sampling can be used to approximate the posterior distribution in many settings.

Gibbs sampler algorithm:

1. Choose starting values: $\theta^{(0)} = \left(\theta_1^{(0)}, \theta_2^{(0)}, \ldots, \theta_d^{(0)}\right)$.
2. Starting at $j = 1$, complete a single cycle for the d subvectors by generating observations according to $p\left(\theta_j \middle| \theta_{-j}^{(t-1)}, y\right)$.
3. Increment t, and repeat until convergence.

Doing Bayesian Data Analysis in R

Options for determining the posterior distribution:

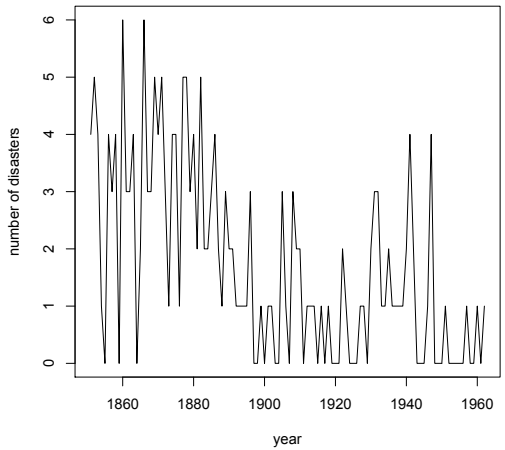
1. Code it up yourself
2. Find an R package to do the heavy lifting
3. Use R to call JAGS (or some other BUGS variant)

Assessing convergence:

- After running an MCMC algorithm, one needs to analyze the results to assess whether the results are giving you what you want (or even expect).

Coal Mining Example

Coal mining is a notoriously dangerous occupation. Consider the tally of coal mine disasters over a 112-year period (1851 to 1962) in the United Kingdom. The data have relatively high disaster counts in the early era, and relatively low counts in the later era. When did technology improvements and safety practices have an actual effect on the rate of serious coal mining disasters in the United Kingdom?



We assume that our data can be described as having a Poisson(λ) distribution for the first k years, and a Poisson(ϕ) distribution in years $k+1, \ldots, 112$. We believe that in year k, technology and/or safety improvements changed the pattern of coal mine disasters. Assume a Gamma(a, b) prior distribution for λ, a Gamma(c, d) prior distribution for ϕ, and a discrete uniform prior distribution on [1, 2, … , 112] for k.

We'll later assume that $a = 4, b = 1, c = 1, d = 2$.

We can derive that the full conditional distributions are:

$$p(\lambda|\phi, k, \boldsymbol{y}) = \text{Gamma}(a + \sum_{i=1}^{k} y_i, b + k)$$

$$p(\phi|\lambda, k, y) = \text{Gamma}(c + \sum_{i=k+1}^{112} y_i, 112 - k + d)$$

$$p(k|\lambda, \phi, \boldsymbol{y}) = \frac{e^{k(\phi-\lambda)}\left(\frac{\lambda}{\phi}\right)^{(\sum_{i=1}^{k} y_i)}}{\sum_{k=1}^{112} e^{k(\phi-\lambda)}\left(\frac{\lambda}{\phi}\right)^{(\sum_{i=1}^{k} y_i)}}.$$

The First Attempt to Code this in R

1. Not worried about speed or efficiency.
2. Not initially programmed up as a function.
3. Coded specifically for this problem.

See drug_first.R.

The Second Attempt to Code this in R

1. Generalize the code to allow different prior parameters, starting values.
2. Program it as a function so that we can reuse the code.
3. Break up certain parts of code into separate functions called from main function.

See drug_loop.R.

The Third Attempt to Code this in R

Now it's time to optimize the code (important if we'll be doing this again, running longer chains, etc.)

Some tips:

- Avoid loops when possible.
  - Vectorize calculations instead.
- Keep as many calculations outside the loop as possible.
- Create "storage" objects in one go.
  - Don't use c(), cbind(), or add rows to matrices, etc.

This results in much faster code.

See drug_vec.R.

The Need for Speed (Rcpp)

- Sometimes we just can't avoid loops
- Sometimes we need to make our code as fast as possible.
- The C/C++ language doesn't suffer from slow loops and is much faster than R at everything (in general).
  - It is typically much slower to program in C/C++ than R.
  - C/C++ are compiled languages instead of scripting languages.
- The Rcpp package "seamlessly integrates C++ and R".
  - Allows us to call C++ from R with relative ease.
  - Maintained (mainly) by Dirk Eddelbuettel
  - Actively developed, improved.
  - Active developer mailing list.

- Rcpp is optimally used when creating packages.
- The "inline" package can be used to provide the capabilities or Rcpp in a script file without creating a package.
- Makes it much easier to debug your C++ code.
- ALWAYS prototype in R first, then port to C++.
- Sometimes you'll have to get your hands dirty and reimplement an algorithm in C++.
  - E.g., the sample() function is not available in Rcpp, so we have to reimplement it.
- Not necessary in all situations, but can results in dramatic speed increases in the right situations.
  - Running the simulation for a long time. A 5x speedup might results in weeks worth of saved computing time.

See drug_rcpp.R.

Timing Results

The Rcpp version of the Gibbs sampler is almost 100 times faster than the looped R version.

```
> time1/time3 #about 100 times faster!
    user   system  elapsed
94.40411      Inf 98.82192
```

The Rcpp version of the Gibbs sampler is almost 5 times faster than the vectorized R version.

```
> time2/time3 #about 10 times faster!
    user   system  elapsed
4.527397      Inf 4.678082
```

"Off the Shelf" Analysis

There are many R packages devoted specifically to Bayesian Analysis.

Some are general:

- bayesm: provides R functions for Bayesian inference for various models widely used in marketing and micro-econometrics.
- MCMCpack: provides model-specific Markov chain Monte Carlo (MCMC) algorithms for a wide range of models commonly used in the social and behavioral sciences.

Some are more specific:

- bcp: Bayesian changepoint analysis
- spBayes: Bayesian inference for geostatistical data
- BayesTree: Implements Bayesian Additive Regression Trees (BART)
- Many others!

Advantages of using packages:

- Saves you the time of coding it up yourself.
- The package authors (presumably) know a lot about how to fit these models and will do it correctly (while you might make an error).

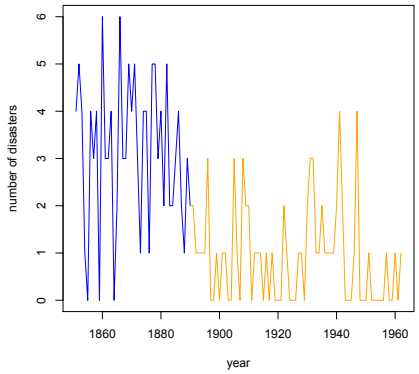
Disadvantages:

- Model might be formulated differently than you want.
- You might not really understand what they're doing.

Don't run with scissors!

See drug_mcmcpack.R for changepoint analysis using MCMCpack package.

Results:



Bayesian Inference Using rjags

The BUGS (**B**ayesian inference **U**sing **G**ibbs **S**ampling) project is concerned with flexible software for the Bayesian analysis of complex statistical models using Markov chain Monte Carlo (MCMC) methods.

Designed to automatically utilize MCMC methods to determine the posterior distribution of Bayesian analysis.

Began with "Classic BUGS", which morphed into WinBUGS, and now to OpenBUGS and JAGS.

JAGS and OpenBUGS are easily run from R.

BUGS is a declarative programming language.

- You tell BUGS (JAGS) what to do, but not how to do it!
- You don't need to know how to program a Gibbs sampler or other MCMC algorithms to solve a problem.
- You tell BUGS what the model is and it should take care of the rest, assuming you programmed the model correctly.
- Very flexible. Can run lots of different Bayesian models.

What does BUGS need?

1. A model file specifying the sampling distribution of the data, any prior distributions for the relevant parameters, and any additional relationships that may be needed.
2. A list containing the data.
3. Possibly, a list specifying initial values for the MCMC chain (though you can have JAGS automatically choose this for you).
4. Several additional arguments (like how much burnin, how long you want the chain to be, number of chains, etc.), and calling a few functions in R.

The model file will typically be the most difficult part of using JAGS to do analysis. The other parts will be similar in pretty much every model.

Example: Soft drink delivery times

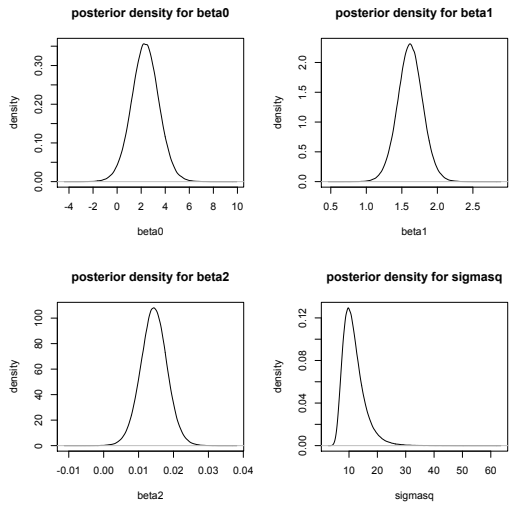
We are interested in estimation of the required time needed by each employee in a delivery system network to refill an automatic vending machine. For this reason, a small quality assurance study was set up by an industrial engineer of the company. As the response variable, the engineer considered the total service time (measured in minutes) of each machine, including its stocking with beverage products and any required maintenance or housekeeping. After examining the problem, the industrial engineer recommended two important variables that affect delivery time: the number of cases of stocked products and the distance walked by the employee (measured in feet).

We assume that sampling distribution for the data is $y_i \sim N(\beta_0 + \beta_1 \text{cases} + \beta_2 \text{distance}, 1/\tau)$ (where $\sigma^2 = 1/\tau$ is the error variance of the responses).

We assume low information prior distributions for the regression coefficients, $\beta_j \sim N(0, 10^4)$ and for the precision, $\tau \sim \text{Gamma}(0.01, 0.01)$.

The R code to run this model in JAGS is found in drug_jags.R.

Approximate posterior densities for the parameters of interest are shown below.



Assessing Convergence

Anytime we need to use MCMC methods to approximate a posterior distribution, we should use diagnostic tools to assess the quality of our approximation.

The **coda** and **boa** packages implement many of these diagnostic tools in R.

These packages allow you to determine the correlation between the parameter values of each cycle, assess whether the chain(s) have converged, run long enough etc.

See drug_coda.R for examples.

Additional Bayesian Packages in R

http://cran.r-project.org/web/views/Bayesian.html

Books Using R or R/JAGS for Bayesian Analysis

- A First Course in Bayesian Statistical Methods by Peter Hoff
- Bayesian Computation with R by Jim Albert
- Doing Bayesian Data Analysis: A Tutorial Introduction with R by John Kruschke
- Bayesian Modeling Using WinBUGS by Ioannis Ngzoufras