Bayesian Inference in Python

Brett Naul
Adapted from slides by Joey Richards

Q: What is Bayesian Statistics?

Answer: A system for statistical inference and decision making based on expressing all uncertainty in terms of conditional probability statements; e.g., P(theory | data)

- Inferential Statistics - given some data, what do we believe?
- In Bayesian Inference, probability is the fundamental measure of uncertainty
- Bayesian Statistics allows us to make probabilistic statements about a situation given the available data

- In Bayesian Statistics, we construct probability models over both observations (data) and beliefs (unknown parameters and theories)
- In Classical (Frequentist) Statistics, defining probability models over parameters and hypotheses is disallowed
  - parameters are fixed and unknown

In Bayesian statistics, we perform inferences with *posterior probability distributions* on **parameters** of interest, θ, given some data **X**

According to Bayes' Theorem:

$$\underbrace{p(\theta|\mathbf{X})}_{\text{“posterior”}} \propto \underbrace{p(\mathbf{X}|\theta)}_{\text{“likelihood”}} \underbrace{p(\theta)}_{\text{“prior”}}$$

A Brief History of Bayesian Statistics

- Thomas Bayes (1702–1761), a minister & amateur mathematician, proved a case of Bayes' Theorem in a 1763 paper.

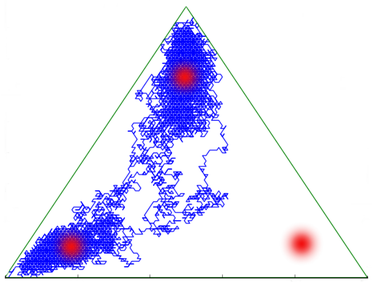
- Pierre-Simon Laplace (1749–1827) introduced a general version of the theorem and applied it to several fields, including celestial mechanics & medicine. When insufficient knowledge was available to specify an informed prior, Laplace used uniform priors, according to his "principle of insufficient reason"

- Fell out of favor in the early 20th century, where Frequentist Statistics of Fisher, Neyman, and Pearson dominated the field

- Around 1950, statisticians began to advocate Bayesian methods to overcome the limitations of frequentist approach: L.J. Savage, Bruno de Finetti, Dennis Lindley

- Bayesian Statistics did not become popular until the 1980's and 90's:

  ➡ the Bayesian approach requires evaluation of complex, multi-dimensional integrals



- Faster and cheaper computing along with efficient sampling algorithms led to the revitalization of the field and wide-spread acceptance

Objective versus Subjective Bayes

- An essential ingredient to obtaining the posterior, $p(\theta \mid \text{data})$, is the prior distribution, $p(\theta)$, symbolizing our belief in θ before collecting or observing any data

- The prior can have a large impact on the inferences and opens one up to charges of non-objectivity

  ➡ However, by the same argument, the choice of *Likelihood function* (probability model for the data, given the model parameters) used by both Bayesians and Frequentists is also subjective

- There is a lot of work attempting to minimize the effect of the prior on resulting inference. These are non-informative or reference priors

- "Subjective" Bayesians believe in the complete subjectivity of the interpretation of probability and believe that informative priors should always be used, if available

Steps in Bayesian Analysis:

1. Specify likelihood and prior (before looking at the data!)
2. Compute the posterior distribution for the parameter(s) of interest given the particular X that we observed.
   - In cases where direct derivation of the posterior is impossible, we instead draw samples from the posterior
3. Check that the model fits well (posterior predictive checks)
4. Perform statistical inferences (parameter estimation, predictions on new data, model comparison)

Example: Obama's Approval Rating

A Gallup poll taken August 20-26 asked $n = 3691$ U.S. citizens: *"Do you approve of the way Barack Obama is handling his job as President?"*

We want to estimate the true proportion, θ, of U.S. citizens who approve of Obama's work as president.

1. Specify a likelihood and a prior (before collecting data!)

First, what is an appropriate random variable expressing the data outcome?

Binomial

The likelihood is:

$$p(x|\theta) = \binom{n}{x}\theta^x(1-\theta)^{n-x}$$

So what prior distribution should we choose for θ?

For our binomial likelihood, we have that:

$$p(\theta|x) \propto \theta^x (1-\theta)^{n-x}\, p(\theta)$$

Observe: if $p(\theta)$ takes the form $\theta^a(1-\theta)^b$, then so will $p(\theta|x)$

Example: Obama's Approval Rating

A Gallup poll taken August 20-26 asked $n = 3691$ U.S. citizens: *"Do you approve of the way Barack Obama is handling his job as President?"*

We want to estimate the true proportion, θ, of U.S. citizens who approve of Obama's work as president.

Posterior under a Binomial(n, θ) likelihood and Beta(α, β) prior:

$$\begin{aligned} p(\theta|x) &\propto \theta^x(1-\theta)^{n-x}\,\theta^{\alpha-1}(1-\theta)^{\beta-1} \\ &\propto \underbrace{\theta^{x+\alpha-1}(1-\theta)^{n-x+\beta-1}}_{\text{Kernel of a Beta distribution}} \end{aligned}$$

Posterior is Beta(α^*, β^*) with parameters:

$$\alpha^* = x + \alpha \qquad \beta^* = n - x + \beta$$

Where x is the number of people, out of $n = 3691$, who approve.

Conjugate Priors

- With a conjugate prior, the posterior takes a closed form. In particular, the posterior is from the same type of distribution as the prior.
- Computations of the posterior with a conjugate prior are trivial. The posterior parameters are typically easy to solve.
- As we collect even more data (under the same likelihood), the posterior can be used as the next prior and it will be conjugate! Allows for easy updates.

However, with some likelihoods, conjugacy **does not exist**, especially in multi-dimensional parameter spaces.

With most *practical* problems in Bayesian inference, conjugacy does not exist. We need to use computationally intensive sampling procedures to characterize the posterior distribution!

MCMC for Bayesian Inference

- Markov Chain Monte Carlo: Stochastic methods useful for sampling from target posterior distributions.
- Can be implemented where conjugacy does not hold and grid approximations fail
- Can be used even in high dimensional examples
- They are iterative: must decide when convergence has happened
- Three popular methods: Metropolis-Hastings, Metropolis and Gibbs Sampler

MCMC for Bayesian Inference

- **Idea:** Suppose that sampling from the posterior, $p(\theta|\mathbf{x})$, is hard, but that we can somehow generate a Markov chain $\{\theta(t), t \in T\}$ with stationary distribution $p(\theta|\mathbf{x})$.
- Here, we know the stationary distribution
- We want to set up a Markov chain that will take us to the stationary distribution $\pi = p(\theta|\mathbf{x})$
- Once we find such a chain, we will start from some initial guess, θ^0, and run the chain for a large number of steps until it converges to π
- After convergence, we run a bunch more steps of the Markov chain and **use those as draws from** $p(\theta|\mathbf{x})$

- All MCMC methods are based on this idea. The differences are in how the Markov chain transitions are created
- **Gibbs sampler:** An iterative algorithm that produces Markov chains with stationary distribution $p(\theta|\mathbf{x})$ by cycling through all conditional posterior distributions, $p(\theta_i|\text{rest}, \mathbf{x})$
- **Example:** Suppose that $\theta = (\theta_1, \theta_2, \theta_3)$. Gibbs steps are:
  1. Start with an initial guess, $\theta^0 = (\theta_1^0, \theta_2^0, \theta_3^0)$
  2. Draw θ_1^1 from $p(\theta_1|\theta_2 = \theta_2^0, \theta_3 = \theta_3^0, \mathbf{x})$
  3. Draw θ_2^1 from $p(\theta_1|\theta_1 = \theta_1^1, \theta_3 = \theta_3^0, \mathbf{x})$
  4. Draw θ_3^1 from $p(\theta_1|\theta_1 = \theta_1^1, \theta_2 = \theta_2^1, \mathbf{x})$
- These steps constitute **one iteration** of the sampler
- We then replace the "initial guess" with the current value of θ and repeat the above steps n times (until convergence), and then $(\theta^{n+1}, \theta^{n+2}, ...)$ is our posterior sample

Why are samples from the posterior useful?

Suppose we have B samples $\theta_1, ..., \theta_B$ from the posterior, $p(\theta|\mathbf{x})$

- Posterior mean: exact equation: $E[\theta|\mathbf{x}] = \int \theta\, p(\theta|\mathbf{x}) d\theta$
  Using the sample: $E[\theta|\mathbf{x}] \approx \frac{1}{B} \sum_{b=1}^{B} \theta_b$
- Marginalization: exact equation:
  $p(\theta_1|\mathbf{x}) = \int p(\theta_1, ..., \theta_p|\mathbf{x})\, d\theta_2 ... d\theta_p$
  Using the sample: $\theta_{1,1}, ..., \theta_{B,1} \sim p(\theta_1|\mathbf{x})$
- Predictions: exact equation: $p(\tilde{x}|\mathbf{x}) = \int p(\tilde{x}|\theta)\, p(\theta|\mathbf{x}) d\theta$
  Using the sample: $\tilde{x}_1|\theta_1, ..., \tilde{x}_B|\theta_B \sim p(\tilde{x}|\mathbf{x})$

Metropolis Algorithm (1953)

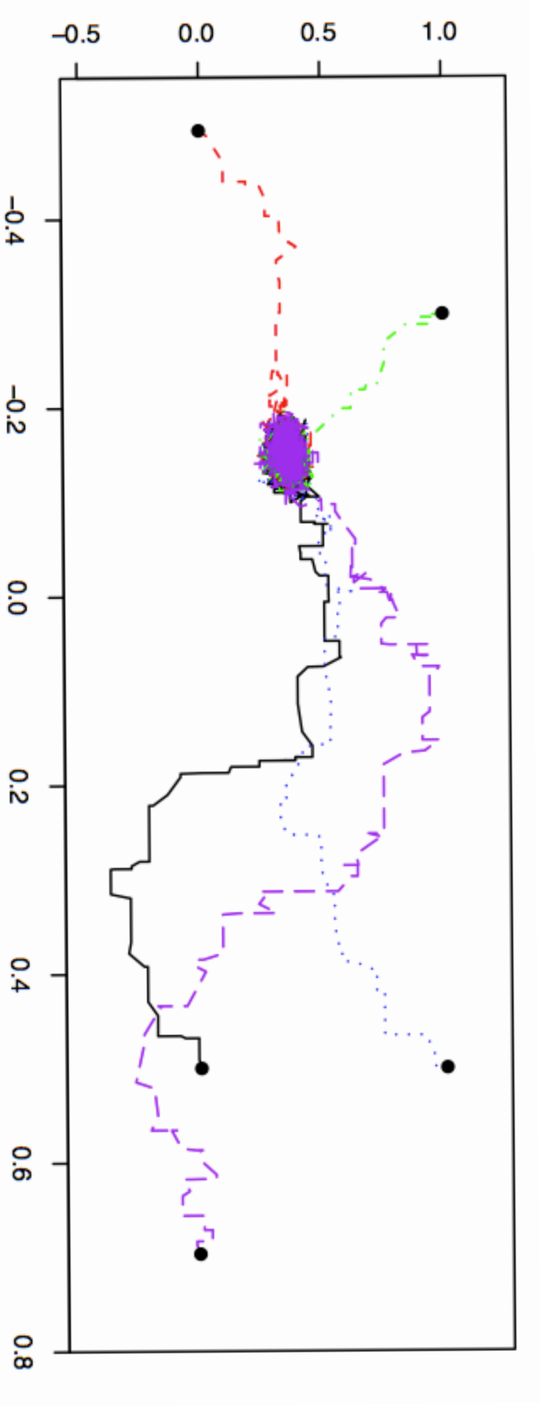
1. Choose a starting point, θ^0.
2. For $t = 1, 2, \ldots$:
   (a) Sample a proposal θ^* from a proposal (jumping) distribution, $J_t(\theta^*|\theta^{t-1})$.
   The proposal distribution must be symmetric for the Metropolis algorithm, with $J_t(\theta_a|\theta_b) = J_t(\theta_b|\theta_a) \quad \forall\ \theta_a, \theta_b, t$
   (b) Calculate the ratio of the posteriors,

$$r = \frac{p(\theta^*|\mathbf{x})}{p(\theta^{t-1}|\mathbf{x})}$$

   (c) Set the next point in the Markov chain to

$$\theta^t = \begin{cases} \theta^* & \text{with probability } \min(r, 1) \\ \theta^{t-1} & \text{otherwise} \end{cases}$$

- We always accept proposals for which $p(\theta^*|\mathbf{x}) \geq p(\theta^{t-1}|\mathbf{x})$
- We sometimes accept proposals for which $p(\theta^*|\mathbf{x}) < p(\theta^{t-1}|\mathbf{x})$
- So the MCMC does not always go uphill, and is thus does not get stuck in local minima (MCMC techniques are useful for optimization)



MCMC with PyMC

https://pymc-devs.github.io/pymc3/

```
pip install pymc3
```

PyMC3 is the most widely used Markov chain Monte Carlo module in Python

- It allows straightforward coding of probability models and posterior sampling of those models with standard (optimized) MCMC algorithms
- Large and complicated (hierarchical) models can be easily coded in PyMC
- Convergence diagnostics and automatic tuning are provided
- Users can input custom probability distributions and fitting algorithms
- Great documentation

To the Notebook

Example: Mining Disasters

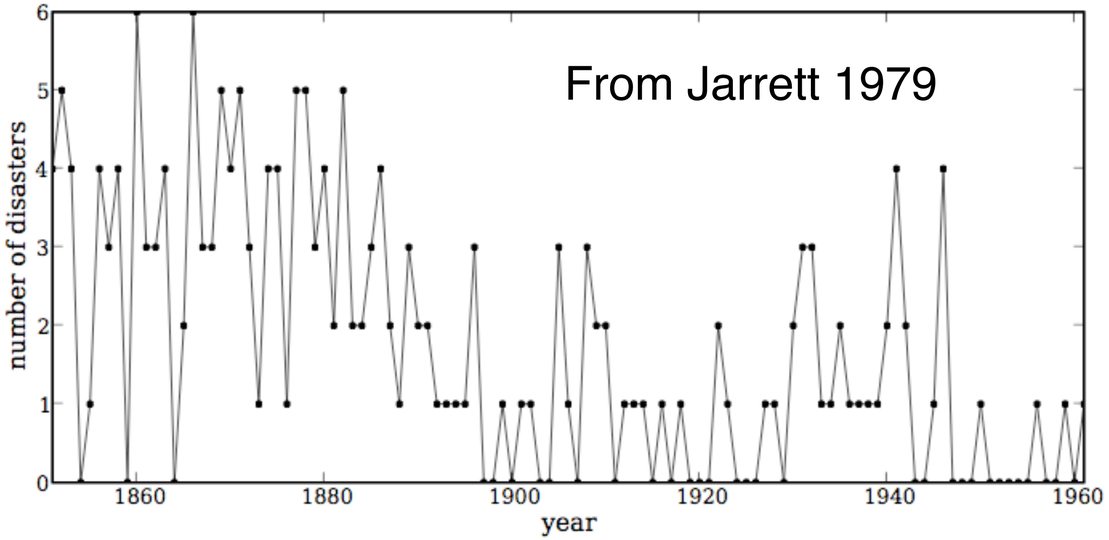


Figure 1: Recorded coal mining disasters in the UK.

Goal - Use data from coal mining disasters to estimate change point in rate of disasters (time and magnitude)

$$(D_t|s,e,l) \sim \text{Poisson}\,(r_t)\,, \qquad r_t = \begin{cases} e & \text{if} \quad t < s \\ l & \text{if} \quad t \geq s \end{cases}, \quad t \in [t_l, t_h]$$

$$s \sim \text{Discrete Uniform}(t_l, t_h)$$
$$e \sim \text{Exponential}(r_e)$$
$$l \sim \text{Exponential}(r_l)$$

D_t = # of disasters in year t
r_t = disaster rate in year t
s = year of changepoint
e, l = old, new rate

Example: Mining Disasters

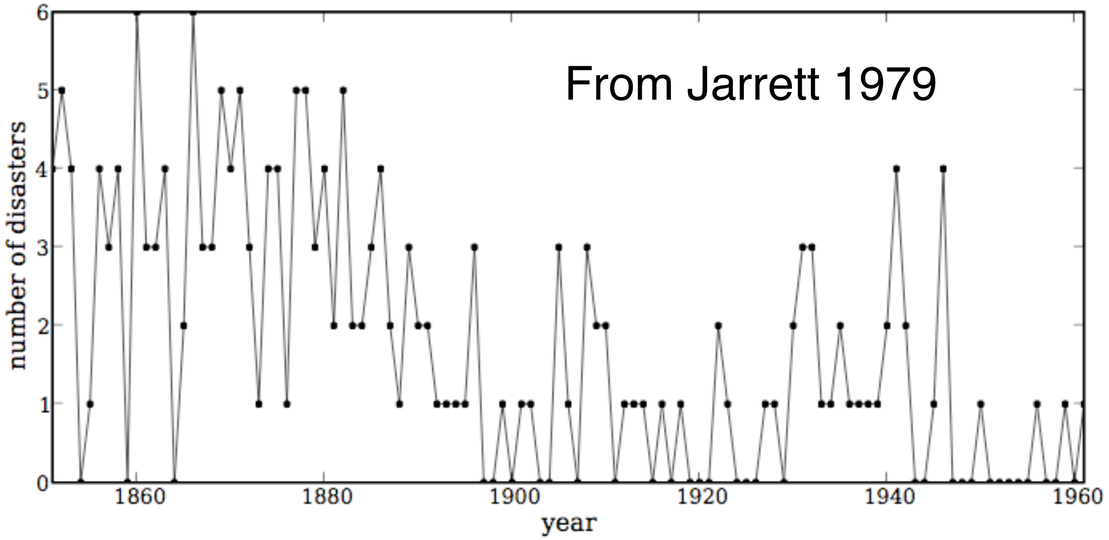


Figure 1: Recorded coal mining disasters in the UK.

Goal - Use data from coal mining disasters to estimate change point in rate of disasters (time and magnitude)

$$(D_t|s,e,l) \sim \text{Poisson}\,(r_t)\,, \qquad r_t = \begin{cases} e & \text{if} \quad t < s \\ l & \text{if} \quad t \geq s \end{cases}, \quad t \in [t_l, t_h]$$

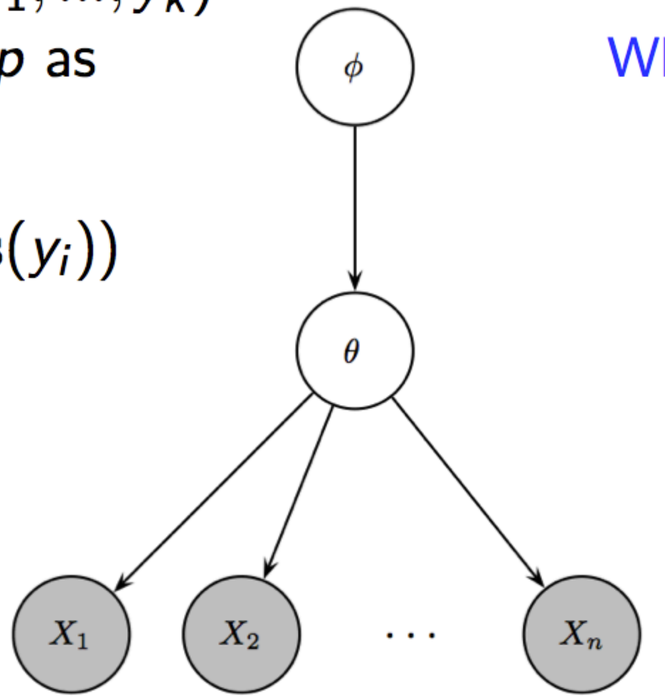
$$s \sim \text{Discrete Uniform}(t_l, t_h)$$
$$e \sim \text{Exponential}(r_e)$$
$$l \sim \text{Exponential}(r_l)$$

D_t = # of disasters in year t
r_t = disaster rate in year t
s = year of changepoint
e, l = old, new rate

Representing Models with Directed Graphs

A graph, $\mathcal{G}$, represents a joint density, $p(y_1, ..., y_k)$ if we can factor p as

$$\prod_{i=1}^{k} p(y_i|\text{parents}(y_i))$$



Why is the joint distribution important?

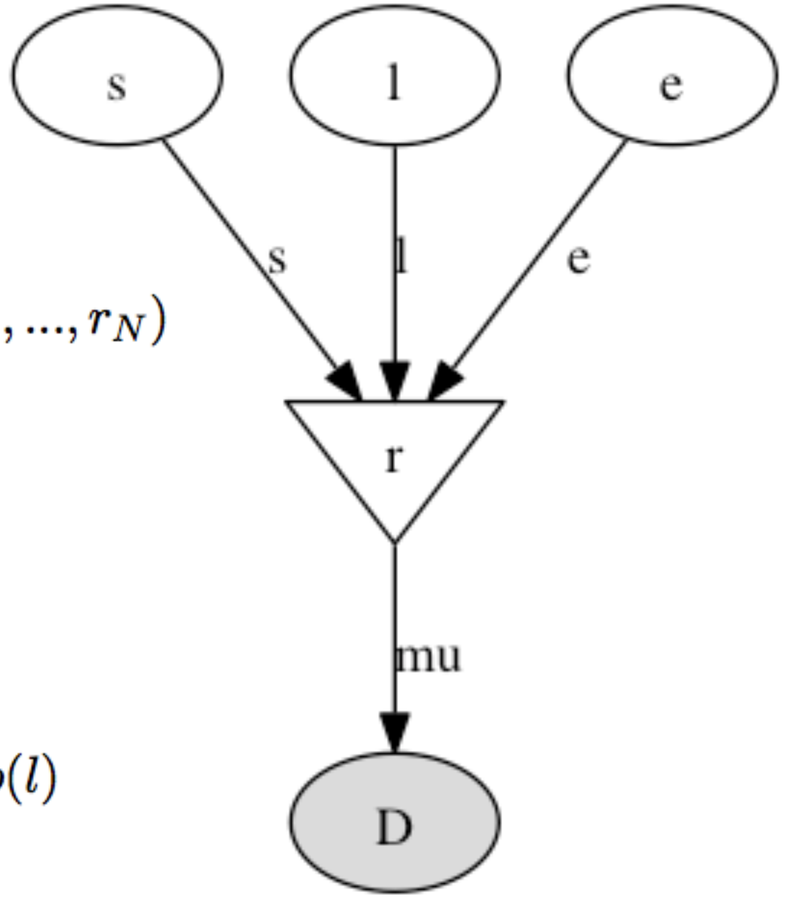
$$\begin{aligned} p(\theta, \phi|x_1, ..., x_n) &= \frac{p(x_1, ..., x_n, \theta, \phi)}{p(x_1, ..., x_n)} \\ &\propto p(x_1, ..., x_n, \theta, \phi) \\ &= \left[\prod_{i=1}^{n} p(x_i|\theta)\right] p(\theta|\phi)p(\phi) \end{aligned}$$

$$p(x_1, ..., x_n, \theta, \phi) = \left[\prod_{i=1}^{n} p(x_i|\theta)\right] p(\theta|\phi)p(\phi)$$

Example: Mining Disasters

$$(D_t|s,e,l) \sim \text{Poisson}\,(r_t)\,, \qquad r_t = \begin{cases} e & \text{if} \quad t < s \\ l & \text{if} \quad t \geq s \end{cases}\,, \quad t \in [t_l, t_h]$$

$$s \sim \text{Discrete Uniform}(t_l, t_h)$$
$$e \sim \text{Exponential}(r_e)$$
$$l \sim \text{Exponential}(r_l)$$



$$\begin{aligned} p(s,e,l,r_1,...,r_N|D_1,...,D_N) &\propto p(D_1,...,D_N|s,e,l,r_t)\,p(s,e,l,r_1,...,r_N) \\ &= \prod_{t=1}^{N} p(D_t|s,e,l,r_t)\,p(s,e,l,r_t) \\ &= \prod_{t=1}^{N} p(D_t|r_t)\,p(r_t|s,e,l)\,p(s,e,l) \\ &= \prod_{t=1}^{N} p(D_t|r_t)\,p(r_t|s,e,l)\,p(s)\,p(e)\,p(l) \end{aligned}$$

Checking Convergence

Determining whether an MCMC has converged can be difficult, especially in high-dimensional parameter spaces

A number of **diagnostics** (both formal and informal) exist, and many of these are available in PyMC

First check - start multiple chains from different starting values and see that they converge to the same place
More formal methods - Raftery-Lewis, Geweke, autocorrelation, etc.
Goodness of fit - Posterior Predictive Checks which simulate data from your fitted model and compare to the observed data (checks convergence AND the suitability of the chosen model)

Other MCMC Code on the Market

WinBUGS, OpenBUGS - Bayesian inference Using Gibbs Sampling
JAGS - Just Another Gibbs Sampler (C++)

in R -
mcmc, rbugs, BRugs, MCMCpack, adaptMCMC, rjags, etc.
See rpy2 (http://rpy.sourceforge.net/rpy2.html)

in Python -
bayesian-inference (http://code.google.com/p/bayesian-inference/)
emcee (http://danfm.ca/emcee/)

What about Bayesian Approaches to Machine Learning?

- Many of the “Bayesian” ML approaches (e.g., Naive Bayes, Bayes Nets) are usually applied in a Frequentist way
    - “prior” probabilities are estimated with MLE instead of assigning probability distributions (e.g., Dirichlet)

- There are lots of great non-parametric Bayes approaches (many of which are being developed here at UCB!)
    - Gaussian processes (priors over smooth functions)
    - Dirichlet processes (priors over allocation of objects to classes)