COLUMBIA UNIVERSITY

IN THE CITY OF NEW YORK

STAT 4224/5224

Bayesian Statistics

Dobrin Marchev

Nonconjugate Priors

- When conjugate or semiconjugate prior distributions are used, the posterior distribution can be approximated with the Monte Carlo method or the Gibbs sampler.
- In situations where a conjugate prior distribution is unavailable or undesirable, the full conditional distributions of the parameters do not have a standard form and the Gibbs sampler cannot be easily used.
- Now we present the Metropolis-Hastings algorithm as a generic method of approximating the posterior distribution corresponding to any combination of prior distribution and sampling model.

Generalized Linear Models

A Generalized Linear Model (GLM) extends the classic linear regression model in two ways:

1. $Y \mid \boldsymbol{x} \sim$ Exponential family

2. Link between the outcome and the predictors:

$$g[E(Y|\mathbf{x})] = \boldsymbol{x}'\boldsymbol{\beta}$$

the function g() is called the *link function.*

GLM

More specifically, GLM assumes that the response variable has a *generalized linear model* pdf:

$$f(y_i;\theta_i,\phi) = e^{\frac{y_i\theta_i - b(\theta_i)}{\phi} + c(y_i,\phi)}, i = 1, \ldots, n$$

where ϕ is called a *dispersion parameter* (and could be known). This form of the distribution is aka *natural* exponential family.

Furthermore, if E($Y_i|\boldsymbol{x}_i$) = μ_i, then it is assumed that Var($Y_i|\boldsymbol{x}_i$) = ϕv(μ_i) and $g(\mu_i) = \eta_i = \boldsymbol{x}_i'\boldsymbol{\beta}$ where *g* is a monotone link function.

Classification and Logistic Regression

The linear regression model assumes the response variable *Y* is quantitative (numerical) and the error terms are normally distributed. If, instead, the response variable is *qualitative*, (or *categorical*), the task of predicting responses is aka *classification*. In such cases the error terms are *not* normally distributed.

Examples of classification problems:

- Your email service determines whether to label an incoming message as “spam” or not based on the text of the email, the subject, and your history of interaction with the sender.
- Hand-written zip codes are scanned and stored as an image file and a computer is programmed to classify each digit as a “0”, “1”, “2”, … , “9”.

Linear Regression Approach for Binary Response

- Consider $Y_i \sim$ Bernoulli(π_i), where π_i = P(Y_i = 1 | $X_i = x_i$).
- We could use a linear probability model (LPM) with the usual assumptions:

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{i,p} + \varepsilon_i$$
$$\varepsilon_i \sim N(0, \sigma^2)$$

- The expected values for the linear probability model from RHS are:

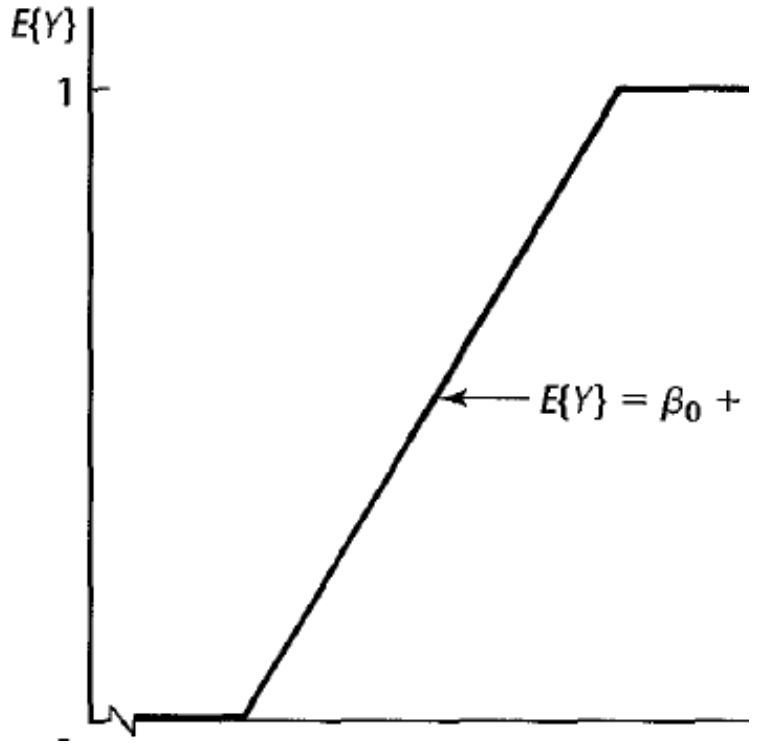
$$E(Y_i | X_i = x_i) = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{i,p}$$

- But for a Bernoulli outcome from the LHS we have:

$$E(Y_i | X_i = x_i) = \pi_i = \mathrm{P}(Y_i = 1 | X_i = x_i)$$

Problems with Linear Probability Model

- Predicted values from linear regression are not constrained to the interval [0,1], even though probabilities should be.



Problems with Linear Regression (Continued)

- Predicted values from linear regression are not constrained to the interval [0,1], even though probabilities should be.
- Residuals from the linear model are not normally distributed because they are dichotomous for each $X = x$:

$$\varepsilon_i = \begin{cases} -p(X = x) \text{ if } Y_i = 0 \\ 1 - p(X = x) \text{ if } Y_i = 1 \end{cases}$$

- Also, the variance of ε_i is $p(X = x)[1 - p(X = x)]$, which means it is not constant.
- Furthermore, if the outcome has more than two categories, linear regression becomes impossible unless they are ordered, and we assume the distances between all categories are identical.
- We need a method that deals with the above-mentioned deficiencies.

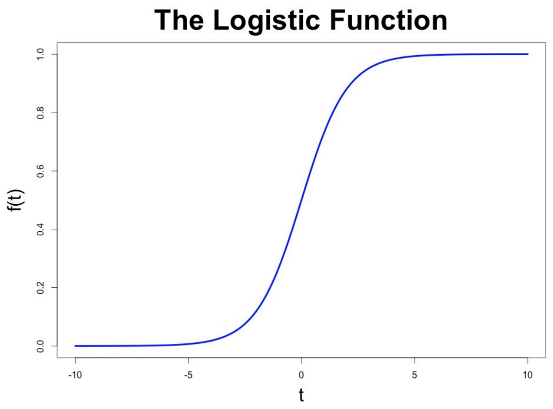
Logistic Regression: Foundation

- Assume the outcome Y is coded as 0/1. Our goal is to specify a model for
$$p(X) = \Pr(Y = 1|X = x)$$

- In linear regression we use a linear model in the covariates $X_1, \ldots, X_p$:
$$p(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

- The model above, however, has a range equal to $(-\infty, \infty)$. To restrict our range to $(0,1)$, we need a function f:$(-\infty, \infty) \rightarrow (0,1)$. Ideally, the function will be simple to write, continuous, and monotonic. The *logistic function* is a prime candidate:

$$f(t) = \frac{\exp(t)}{1 + \exp(t)}$$



Sigmoidal Response Functions

- The *logistic function* is a prime candidate:

$$f(t) = \frac{\exp(t)}{1 + \exp(t)}$$

- An alternative is the *probit function*:

$$\Phi(t)$$

where Φ is the cdf of the standard normal distribution.

- Theoretically, any cumulative distribution function can serve as a response function.

Logistic Regression: Foundation

- Applying the logistic function to the linear transformation of the predictors gives three equivalent formulations.

1. probability

$$p(X) = \frac{\exp\left(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p\right)}{1 + \exp\left(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p\right)}$$

2. odds

$$\frac{p(X)}{1 - p(X)} = \exp\left(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p\right)$$

3. log (odds) or "logit"

$$\log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

Exercise: Show algebraically that 3 implies 1.

- The logit is linear in $X_1, \ldots, X_p$.

Logistic Regression: Estimating Betas

- Just as in simple regression, the coefficients β_0 and β_1 are unknown and must be estimated using the available data. *Maximum likelihood* is the most commonly used method for this problem.
- When $Y_i|x_i \sim Bernoulli\big(p(x_i)\big)$, then the likelihood function is:

$$L(\beta_0, \beta_1) = \prod_{i=1}^{n} p(x_i)^{y_i}[1 - p(x_i)]^{1-y_i}$$

Since $\frac{p(x_i)}{1-p(x_i)} = e^{\beta_0+\beta_1 x_i} \Rightarrow 1 - p(x_i) = \frac{1}{1+e^{\beta_0+\beta_1 x_i}}$ and therefore

$$L(\beta_0, \beta_1) = \prod_{i=1}^{n} \left(e^{\beta_0+\beta_1 x_i}\right)^{y_i} \frac{1}{1 + e^{\beta_0+\beta_1 x_i}}$$

$$\Rightarrow l(\beta_0, \beta_1) = \sum_{i=1}^{n} y_i(\beta_0 + \beta_1 x_i) - \sum_{i=1}^{n} \log\left(1 + e^{\beta_0+\beta_1 x_i}\right)$$

Logistic Regression: Estimating Betas

- The logistic regression loglikelihood is:

$$l(\beta_0, \beta_1) = \sum_{i=1}^{n} y_i(\beta_0 + \beta_1 x_i) - \sum_{i=1}^{n} \log\left(1 + e^{\beta_0 + \beta_1 x_i}\right)$$

$$\Rightarrow \frac{\partial l(\beta_0, \beta_1)}{\partial \beta_0} = \sum_{i=1}^{n} \left(y_i - \frac{e^{\beta_0 + \beta_1 x_i}}{1 + e^{\beta_0 + \beta_1 x_i}} \right)$$

- Unlike the closed-form analytical solution (i.e., the normal equations) available for linear regression, the score function for logistic regression is transcendental. That is, there is no closed-form solution so we must solve with numerical optimization methods such as Newton's method (IRLS) which is done in several iterations.
- Although we have focused on simple logistic regression, this likelihood (and score) function generalize directly to the case of p predictors.

Bayesian Version

- As in the case of ordinary regression, a natural class of prior distributions for **β** is the class of multivariate normal distributions.
- However, for the logistic regression model would a prior distribution from this class result in a multivariate normal posterior distribution for **β**.
- Furthermore, standard conjugate prior distributions for generalized linear models do not exist (except for the normal regression model).
- Although independent Monte Carlo sampling from the posterior is not available for this logistic regression model, will show how to construct a Markov chain that can approximate $f(\mathbf{\beta}|\boldsymbol{X}, \boldsymbol{y})$ for any prior distribution $\pi(\boldsymbol{\beta})$.

The Metropolis Algorithm

Suppose we the model

$$X \sim f(x|\theta)$$
$$\theta \sim \pi(\theta)$$

Then the target distribution is

$$f(\theta|x) = \frac{\pi(\theta) f(x|\theta)}{\int \pi(\theta) f(x|\theta) d\theta}$$

Recall: the target posterior is difficult/impossible to calculate directly due to the denominator $m(x) = \int \pi(\theta) f(x|\theta) d\theta$.

If we can sample iid $\theta^{(1)}, \dots, \theta^{(S)} \sim f(\theta|x)$ then the problem is solved and we can obtain Monte Carlo approximations, like

$$E[g(\theta)|x] \approx \frac{1}{S} \sum_{i=1}^{S} g\left(\theta^{(i)}\right)$$

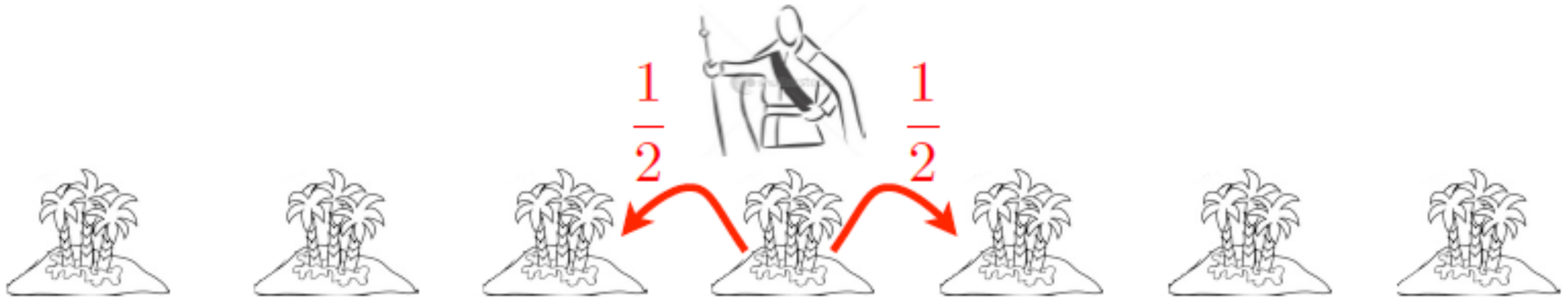
What if we cannot sample directly from $f(\theta|x)$?

Metropolis: How does it work?

The Metropolis Archipelago

Contract: King Markov must visit each island in proportion to its population size.

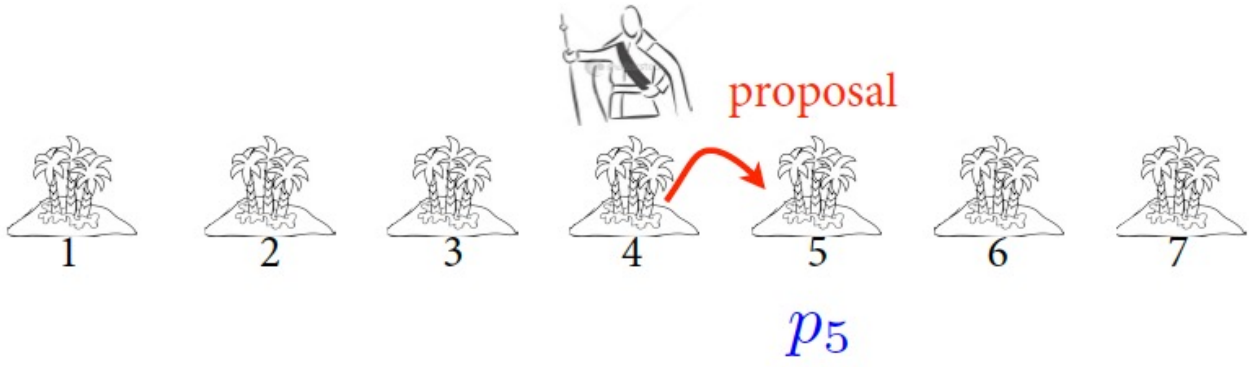
Imagine the king is too lazy to keep track of schedules and wants a simple algorithm using only the current location and nearby islands' population sizes. Here is how he does it:



(1) Flip a coin to choose island on left or right.
Call it the "proposal" island.

Propose a move

(1) Flip a coin to choose island on left or right.
Call it the "proposal" island.

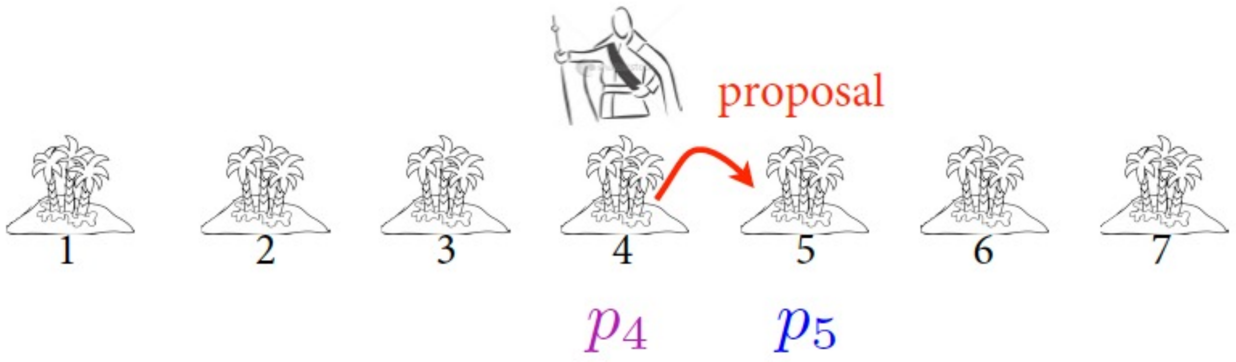


(2) Find population of proposal island.

Compare to current state

(1) Flip a coin to choose island on left or right.
Call it the "proposal" island.

(2) Find population of proposal island.



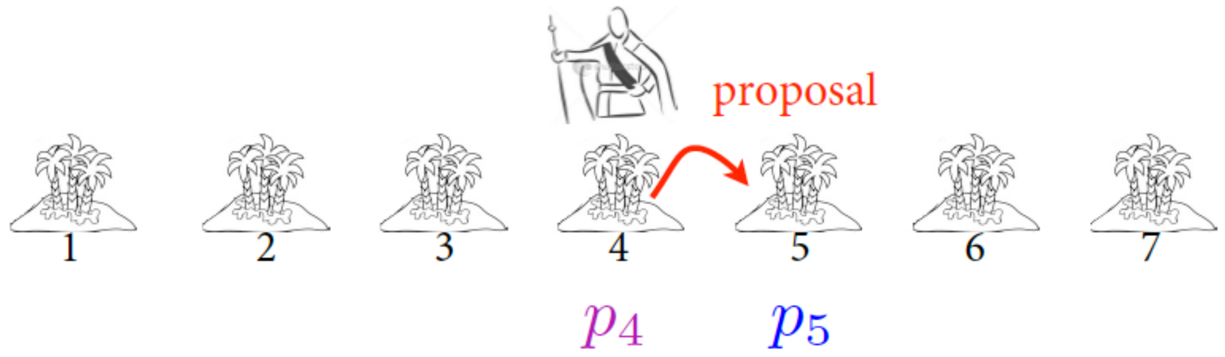
(3) Find population of current island.

Decide whether to jump

(1) Flip a coin to choose island on left or right.
Call it the "proposal" island.

(2) Find population of proposal island.

(3) Find population of current island.



(4) Move to proposal, with probability = $\frac{p_5}{p_4}$

Repeat until "convergence"

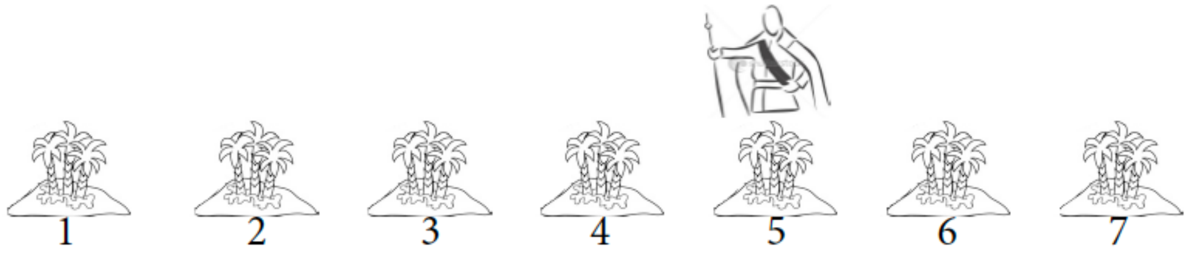
(1) Flip a coin to choose island on left or right.
Call it the "proposal" island.

(2) Find population of proposal island.

(3) Find population of current island.

(4) Move to proposal, with probability = $\frac{p_5}{p_4}$

(5) Repeat from (1)



This procedure ensures visiting each island in proportion to its population, *in the long run.*

How does it work in general?

Let's think intuitively about how we might construct a collection of (not necessarily iid) $\theta^{(1)}, \dots, \theta^{(S)} \sim f(\theta|x)$. Suppose we have a working collection $\{\theta^{(1)}, \dots, \theta^{(s)}\}$ to which we would like to add a new/proposed value $\theta^{(s+1)}$. Let's consider adding a value θ^* which is "nearby" $\theta^{(s)}$. Should we include θ^* in the set or not? If $f(\theta^*|\boldsymbol{x}) > f(\theta^{(s)}|\boldsymbol{x})$ then we want more θ^*'s in the set than $\theta^{(s)}$'s. Since $\theta^{(s)}$ is already in the set, then it seems we should include θ^* as well. On the other hand, if $f(\theta^*|\boldsymbol{x}) < f(\theta^{(s)}|\boldsymbol{x})$ then it seems we should not necessarily include θ^*. Compute:

$$r = \frac{f(\theta^*|x)}{f(\theta^{(s)}|x)} = \frac{\pi(\theta^*)f(x|\theta^*)}{m(x)} \frac{m(x)}{\pi(\theta^{(s)})f\left(x\middle|\theta^{(s)}\right)} = \frac{\pi(\theta^*)f(x|\theta^*)}{\pi(\theta^{(s)})f\left(x\middle|\theta^{(s)}\right)}$$

- If $r \geq 1$: accept θ^* into the set, i.e. $\theta^{(s+1)} = \theta^*$
- If $r < 1$: set $\theta^{(s+1)} = \theta^*$ or $\theta^{(s)}$ with probability r and $1 - r$.

This is the basic intuition behind the famous *Metropolis algorithm*.

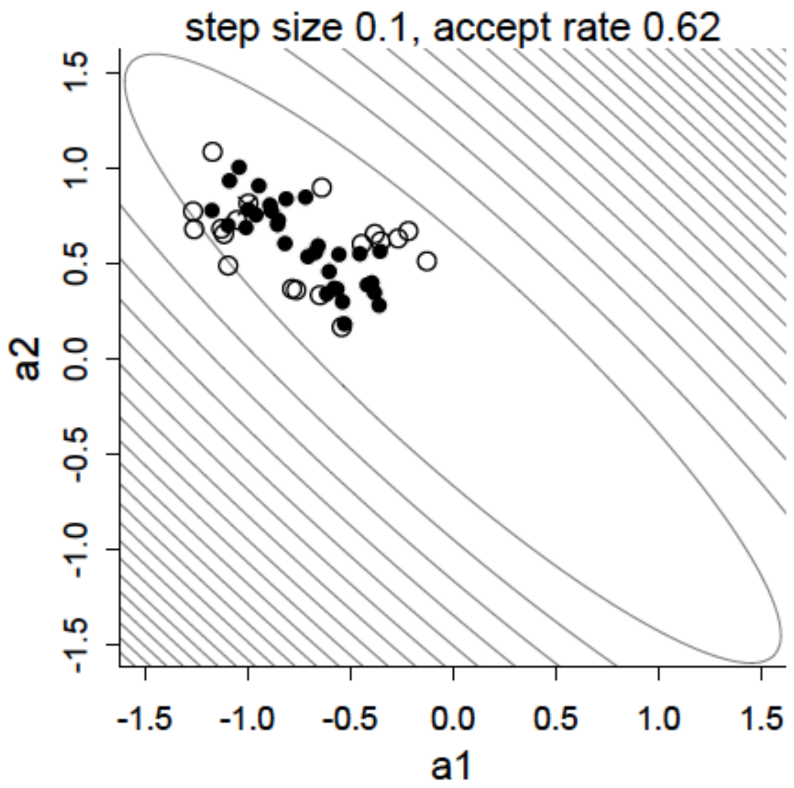
Metropolis Algorithm

The Metropolis algorithm proceeds by sampling a *proposal* value θ^* nearby the current value $\theta^{(s)}$ by using a *symmetric proposal distribution* $J(\theta^*|\theta^{(s)})$. Symmetric means that $J(\theta_a|\theta_b) = J(\theta_b|\theta_a)$. Usually, the proposal distribution $J(\theta^*|\theta^{(s)})$ is very simple, like $U(\theta^{(s)} - \delta, \theta^{(s)} + \delta)$ or $N(\theta^{(s)}, \delta^2)$. The value of the parameter δ is generally chosen to make the approximation algorithm run efficiently.

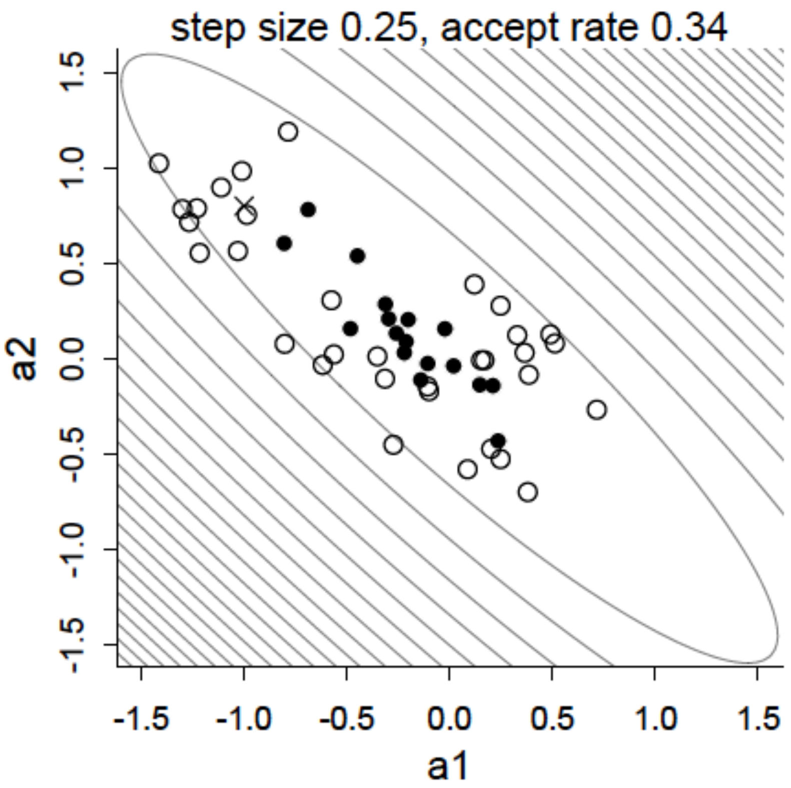
Algorithm: Given $\theta^{(s)}$, the Metropolis algorithm generates a value $\theta^{(s+1)}$ as follows

1. Sample $\theta^* \sim J(\theta|\theta^{(s)})$.
2. Compute $r = \dfrac{\pi(\theta^*)f(x|\theta^*)}{\pi(\theta^{(s)})f(x|\theta^{(s)})}$
3. Let $\theta^{(s+1)} = \theta^*$ with probability $\min\{r, 1\}$, otherwise $\theta^{(s+1)} = \theta^{(s)}$

Issues with Metropolis: not efficient



small steps, slow walk



bigs steps, low accept rate

Metropolis-Hastings Algorithm

- Very general method for constructing a Markov chain with a specific invariant density $f(x)$, which is the target for the simulation.
- Suppose that $q(y|x)$ is any conditional density, called proposal density.
- Algorithm:
  - Step 0: Start at any $X = x_0$
  - Step 1: Generate $Y_i \sim q(y|x_{i\text{-}1})$
  - Step 2: Set $X_i = \begin{cases} Y_i \text{ with probability } \alpha(x_{i-1}, y_i) \\ x_{i-1}, \quad \text{o/w} \end{cases}$

  where $\alpha(x, y) = \min\left\{\frac{f(y)}{f(x)} \frac{q(x|y)}{q(y|x)}, 1\right\}$ is called the acceptance probability
  - Go to Step 1 and repeat

  Notes:
  - When $q(y|x)$ is symmetric, the algorithm is known as Metropolis.
  - When $q(y|x)$ does not depend on x, the algorithm is known as Independent Metropolis-Hastings.
  - When $q(y|x) = \mathrm{g}(y - x)$, the algorithm is known as Random Walk Metropolis-Hastings (see next slide)

Example: normal model

Recall the normal model with known variance:

$$\theta | x_1, \ldots, x_n, \sigma^2 \sim N(\mu_n, \tau_n^2)$$

where

- Posterior variance :

$$\tau_n^2 = \frac{1}{\frac{n}{\sigma^2} + \frac{1}{\tau_0^2}} \Rightarrow \frac{1}{\tau_n^2} = \frac{1}{\tau_0^2} + \frac{n}{\sigma^2}$$

- Posterior mean:

$$\mu_n = \tau_n^2 \left(\frac{n}{\sigma^2} \bar{x} + \frac{1}{\tau_0^2} \mu_0 \right) = \frac{\frac{1}{\tau_0^2}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma^2}} \mu_0 + \frac{\frac{n}{\sigma^2}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma^2}} x$$

We will redo this with Metropolis. Suppose that

$\sigma^2 = 1$, $\tau^2 = 10$, $\mu = 5$, $n = 5$ and $x = (9.37, 10.18, 9.16, 11.60, 10.33)$.

Example: continued

Suppose that

$\sigma^2 = 1$, $\tau^2 = 10$, $\mu = 5$, $n = 5$ and $x = (9.37, 10.18, 9.16, 11.60, 10.33)$.

Then $\theta|x \sim N(10.03, 0.44)$

For the proposal distribution we will use $N\left(\theta^{(s)}, 2\right)$, so $\delta = \sqrt{2}$

The starting value is $\theta^{(0)} = 0$.

Note: Often the computation of the ratio r is unstable, so use the trick that

$$\log r = \log[\pi(\theta^*) f(x|\theta^*)] - \log\left[\pi\left(\theta^{(s)}\right) f\left(x\middle|\theta^{(s)}\right)\right]$$

In our example,

$$f(x|\theta) = \prod_{i=1}^{n} dnorm(x_i|\theta, \sigma^2)$$

Burn-in period

Just as with the Gibbs sampler, we can approximate posterior means, quantiles and other posterior quantities of interest using the empirical distribution of $\{\theta^{(1)}, \ldots, \theta^{(S)}\}$. However, our approximation to these quantities will depend on how well our simulated sequence actually approximates $f(\theta|x)$. Results from probability theory say that, in the limit as $S \to \infty$, the approximation will be exact, but in practice we cannot run the Markov chain forever. Instead, the standard practice in MCMC approximation, using either the Metropolis algorithm or the Gibbs sampler, is as follows:

Discard the first $\{\theta^{(1)}, \ldots, \theta^{(B)}\}$ iterations and use the empirical distribution of $\{\theta^{(B+1)}, \ldots, \theta^{(B+S)}\}$ to approximate $f(\theta|x)$.

The iterations up to and including B are called the “burn-in” period, in which the Markov chain moves from its initial value to a region of the parameter space that has high posterior probability.

How large should be δ

- When δ is too small, then $r \approx 1$ for most proposed values. As a result, θ^* is accepted as the value of $\theta^{(s+1)}$ very often. Although this high acceptance rate keeps the chain moving, the moves are never very large and so the Markov chain is highly correlated. One consequence of this is that it takes a large number of iterations for the Markov chain to move from the starting value of zero to the posterior mode.
- At the other extreme, when δ is too large, the chain moves quickly to the posterior mode but once there it gets "stuck" for long periods. This is because the variance of the proposal distribution is so large that $\theta*$ is frequently very far away from the posterior mode.
- In order to construct a Markov chain with a low correlation we need a proposal variance that is large enough so that the Markov chain can quickly move around the parameter space, but not so large that the proposals end up getting rejected most of the time.

Example 2

Data Default from ISLR package

A simulated data set containing information on ten thousand customers. The aim here is to predict which customers will default on their credit card debt.

default: A factor with levels No and Yes indicating whether the customer defaulted on their debt

student: A factor with levels No and Yes indicating whether the customer is a student

balance: The average balance that the customer has remaining on their credit card after making their monthly payment

income: Income of customer

We will fit logistic regression with $N(0, 10^2)$ priors on each β