Hierarchical Models and MCMC Estimation

February 21, 2013

Overview

- Extend Bayesian inference for more complex models
  - Models with multiple unknowns/priors
  - Hieriarchical approach to model fitting
  - Regularization and ‘shrinkage’
- Gibbs and Metropolis Algorithms for Markov Chain Monte Carlo (MCMC) posterior inference in R
  - Basic simulation inference
  - Bugs/Jags model language

Multiparameter Bayesian Models

- Consider a two-parameter model with $\tilde{\theta} = \{\gamma, \delta\}$
- We estimate $\tilde{\theta}$ by randomly sampling from the *joint posterior* $p(\tilde{\theta}|y)$ distribution:

$$\begin{aligned} p(\tilde{\theta}|y) &\propto p(\tilde{\theta})p(y|\tilde{\theta}) \\ p(\gamma, \delta|y) &\propto p(\gamma, \delta)p(y|\gamma, \delta) \end{aligned}$$

Multiparameter Bayesian Models

- Consider a two-parameter model with $\tilde{\theta} = \{\gamma, \delta\}$
- We estimate $\tilde{\theta}$ by randomly sampling from the *joint posterior* $p(\tilde{\theta}|y)$ distribution:

$$
\begin{aligned}
p(\tilde{\theta}|y) &\propto p(\tilde{\theta})p(y|\tilde{\theta}) \\
p(\gamma, \delta|y) &\propto p(\gamma, \delta)p(y|\gamma, \delta)
\end{aligned}
$$

- To proceed, we need to parameterize the:
  - *joint prior*: $p(\gamma, \delta)$
  - *joint likelihood*: $p(y|\gamma, \delta)$
- Use posterior inference to estimate $\hat{\gamma}$ and $\hat{\delta}$, making sure to consider any dependences that arise

Specify Joint Model

- The joint likelihood is (usually) just the likelihood, e.g.,

$$y|\mu, \sigma^2 \sim N(y|\mu, \sigma^2)$$

  where μ *and* σ^2 are the unknowns

- The main action is on parameterizing the joint prior $p(\gamma, \delta)$, e.g.,

$$p(\beta, \sigma^2) \propto \frac{1}{\sigma^2}$$

- Then perform marginal posterior inference on the resulting joint posterior

Marginal Posterior Inference

- To estimate $\hat{\gamma}$ we can (provisionally) treat δ as a 'nuisance' parameter
    - We want to sample from the marginal distribution $\hat{\gamma} = E[p(\gamma|y)]$
    - Target the conditional posterior $p(\gamma|\delta, y)$, since this embodies any dependencies between γ and δ

Marginal Posterior Inference

- To estimate $\hat{\gamma}$ we can (provisionally) treat δ as a 'nuisance' parameter
  - We want to sample from the marginal distribution $\hat{\gamma} = E[p(\gamma|y)]$
  - Target the conditional posterior $p(\gamma|\delta, y)$, since this embodies any dependencies between γ and δ
- The marginal posterior $p(\gamma|y)$ is the density of γ given the observed data when we hold δ at its average:

$$p(\gamma|y) = \int p(\gamma, \delta|y) \mathrm{d}\delta$$

- We can show this factorizes into:

$$p(\gamma|y) = \int p(\gamma|\delta, y) p(\delta|y) \mathrm{d}\delta$$

- Analytically or iteratively evaluate this integral

Recall Diffuse Linear Model

- We factored the joint posterior distribution $p(\beta, \sigma^2|y)$ into its component conditional and marginal posteriors:

$$\begin{aligned} \beta|\sigma^2, y &\sim N(\bar{y}, \frac{\sigma^2}{n}) \\ \sigma^2|y &\sim \text{Inv-}\chi^2(n-1, s^2) \end{aligned}$$

- We iteratively sampled from $\sigma^2|y$, and then from $\beta|\sigma^2, y$ (using previous σ^2) to obtain posterior estimates for β, (implicitly) integrating over σ^2

Recall Diffuse Linear Model

- We factored the joint posterior distribution $p(\beta, \sigma^2|y)$ into its component conditional and marginal posteriors:

$$\begin{aligned} \beta|\sigma^2, y &\sim N(\bar{y}, \frac{\sigma^2}{n}) \\ \sigma^2|y &\sim \text{Inv-}\chi^2(n-1, s^2) \end{aligned}$$

- We iteratively sampled from $\sigma^2|y$, and then from $\beta|\sigma^2, y$ (using previous σ^2) to obtain posterior estimates for β, (implicitly) integrating over σ^2
- Analytically evaluating the integral $p(\beta|y)$ yields:

$$\frac{\beta - \bar{y}}{s/\sqrt{n}}|y \sim t_{n-1}$$

Specify More Joint Structure

- Specify a joint prior $p(\tilde{\theta})$ where γ and δ each depends on additional parameters $\tilde{\theta}_0 = \{\gamma_0, \delta_0\}$
  - The joint prior takes the general form $p(\gamma, \delta, \gamma_0, \delta_0)$
  - Note that the joint likelihood is still $p(y|\gamma, \delta)$
  - So the posterior is:

$$\begin{aligned} p(\gamma, \delta, \gamma_0, \delta_0|y) &\propto p(\gamma, \delta, \gamma_0, \delta_0)p(y|\gamma, \delta) \\ p(\tilde{\theta}, \tilde{\theta}_0|y) &\propto p(\tilde{\theta}, \tilde{\theta}_0)p(y|\tilde{\theta}) \end{aligned}$$

Specify More Joint Structure

- Specify a joint prior $p(\tilde{\theta})$ where γ and δ each depends on additional parameters $\tilde{\theta}_0 = \{\gamma_0, \delta_0\}$
  - The joint prior takes the general form $p(\gamma, \delta, \gamma_0, \delta_0)$
  - Note that the joint likelihood is still $p(y|\gamma, \delta)$
  - So the posterior is:

$$
\begin{aligned}
p(\gamma, \delta, \gamma_0, \delta_0|y) &\propto p(\gamma, \delta, \gamma_0, \delta_0)p(y|\gamma, \delta) \\
p(\tilde{\theta}, \tilde{\theta}_0|y) &\propto p(\tilde{\theta}, \tilde{\theta}_0)p(y|\tilde{\theta})
\end{aligned}
$$

- Parameterize the joint prior in terms of the 'hyperparameters' $\tilde{\theta}_0$ and model parameters $\tilde{\theta}$

Specify More Joint Structure

- Specify a joint prior $p(\tilde{\theta})$ where γ and δ each depends on additional parameters $\tilde{\theta}_0 = \{\gamma_0, \delta_0\}$
    - The joint prior takes the general form $p(\gamma, \delta, \gamma_0, \delta_0)$
    - Note that the joint likelihood is still $p(y|\gamma, \delta)$
    - So the posterior is:

$$\begin{aligned} p(\gamma, \delta, \gamma_0, \delta_0 | y) &\propto p(\gamma, \delta, \gamma_0, \delta_0) p(y|\gamma, \delta) \\ p(\tilde{\theta}, \tilde{\theta}_0 | y) &\propto p(\tilde{\theta}, \tilde{\theta}_0) p(y|\tilde{\theta}) \end{aligned}$$

- Parameterize the joint prior in terms of the 'hyperparameters' $\tilde{\theta}_0$ and model parameters $\tilde{\theta}$
- When $\tilde{\theta}_0$ are unknown, we add additional structure through 'hyperprior' distributions, or prior distributions on the hyperparameters $p(\tilde{\theta}_0)$

Specify More Joint Structure

- Add hyperprior distributions in terms of (known) $\tilde{\theta}_{-1}$ hyperparameters:
    - The joint prior takes the general form $p(\tilde{\theta}, \tilde{\theta}_0, \tilde{\theta}_{-1})$
    - This factors: $p(\tilde{\theta}, \tilde{\theta}_0, \tilde{\theta}_{-1}) = p(\tilde{\theta}_0, \tilde{\theta}_1)p(\tilde{\theta}|\tilde{\theta}_0, \tilde{\theta}_1)$
    - So the posterior is:

$$p(\tilde{\theta}, \tilde{\theta}_0, \tilde{\theta}_{-1}|y) \propto p(\tilde{\theta}_0, \tilde{\theta}_1)p(\tilde{\theta}|\tilde{\theta}_0, \tilde{\theta}_1)p(y|\tilde{\theta})$$

Specify More Joint Structure

- Add hyperprior distributions in terms of (known) $\tilde{\theta}_{-1}$ hyperparameters:
  - The joint prior takes the general form $p(\tilde{\theta}, \tilde{\theta}_0, \tilde{\theta}_{-1})$
  - This factors: $p(\tilde{\theta}, \tilde{\theta}_0, \tilde{\theta}_{-1}) = p(\tilde{\theta}_0, \tilde{\theta}_1)p(\tilde{\theta}|\tilde{\theta}_0, \tilde{\theta}_1)$
  - So the posterior is:

$$p(\tilde{\theta}, \tilde{\theta}_0, \tilde{\theta}_{-1}|y) \propto p(\tilde{\theta}_0, \tilde{\theta}_1)p(\tilde{\theta}|\tilde{\theta}_0, \tilde{\theta}_1)p(y|\tilde{\theta})$$

- Hyperprior distributions may be defined recursively in terms of $\tilde{\theta}_{-k}$, for $k = 1, 2, \ldots K$

Specify More Joint Structure

- Add hyperprior distributions in terms of (known) $\tilde{\theta}_{-1}$ hyperparameters:
  - The joint prior takes the general form $p(\tilde{\theta}, \tilde{\theta}_0, \tilde{\theta}_{-1})$
  - This factors: $p(\tilde{\theta}, \tilde{\theta}_0, \tilde{\theta}_{-1}) = p(\tilde{\theta}_0, \tilde{\theta}_1)p(\tilde{\theta}|\tilde{\theta}_0, \tilde{\theta}_1)$
  - So the posterior is:

$$p(\tilde{\theta}, \tilde{\theta}_0, \tilde{\theta}_{-1}|y) \propto p(\tilde{\theta}_0, \tilde{\theta}_1)p(\tilde{\theta}|\tilde{\theta}_0, \tilde{\theta}_1)p(y|\tilde{\theta})$$

- Hyperprior distributions may be defined recursively in terms of $\tilde{\theta}_{-k}$, for $k = 1, 2, \ldots K$
- Repeatedly factorize the joint prior:

$$\begin{aligned} p(\tilde{\theta}, \tilde{\theta}_0, \tilde{\theta}_{-1}|y) &\propto p(\tilde{\theta}_0, \tilde{\theta}_1)p(\tilde{\theta}|\tilde{\theta}_0, \tilde{\theta}_1)p(y|\tilde{\theta}) \\ &\propto p(\tilde{\theta}_1)p(\tilde{\theta}_0|\tilde{\theta}_1)p(\tilde{\theta}|\tilde{\theta}_0, \tilde{\theta}_1)p(y|\tilde{\theta}) \end{aligned}$$

Hierarchical Linear Model

- The joint likelihood, $p(y|\mu, \sigma^2)$:

$$y|\mu, \sigma^2 \sim N(y|\mu, \sigma^2)$$

Hierarchical Linear Model

- The joint likelihood, $p(y|\mu, \sigma^2)$:

$$y|\mu, \sigma^2 \sim N(y|\mu, \sigma^2)$$

- The joint prior, $p(\mu, \sigma^2, \mu_0, \sigma_0^2, \tau_0^2)$:

$$\begin{aligned} \mu|\sigma^2 &\sim N(\mu_0, \sigma_0^2) \\ \sigma^2 &\sim \text{Inv-}\chi^2(n-1, \tau_0^2) \end{aligned}$$

Hierarchical Linear Model

- The joint likelihood, $p(y|\mu, \sigma^2)$:

$$y|\mu, \sigma^2 \sim N(y|\mu, \sigma^2)$$

- The joint prior, $p(\mu, \sigma^2, \mu_0, \sigma_0^2, \tau_0^2)$:

$$\begin{aligned} \mu|\sigma^2 &\sim N(\mu_0, \sigma_0^2) \\ \sigma^2 &\sim \text{Inv-}\chi^2(n-1, \tau_0^2) \end{aligned}$$

- The hyperpriors (with known α, β, μ_{-1} and σ_{-1}^2):

$$\begin{aligned} \mu_0|\sigma_0^2 &\sim N(\mu_{-1}, \sigma_{-1}^2) \\ \sigma_0^2 &\sim \text{Inv-Gamma}(\alpha, \beta) \\ \tau_0^2 &\sim \text{Inv-Gamma}(\alpha, \beta) \end{aligned}$$

Hierarchical Linear Model

- The joint likelihood, $p(y|\mu, \sigma^2)$:

$$y|\mu, \sigma^2 \sim N(y|\mu, \sigma^2)$$

- The joint prior, $p(\mu, \sigma^2, \mu_0, \sigma_0^2, \tau_0^2)$:

$$\begin{aligned} \mu|\sigma^2 &\sim N(\mu_0, \sigma_0^2) \\ \sigma^2 &\sim \text{Inv-}\chi^2(n-1, \tau_0^2) \end{aligned}$$

- The hyperpriors (with known α, β, μ_{-1} and σ^2_{-1}):

$$\begin{aligned} \mu_0|\sigma_0^2 &\sim N(\mu_{-1}, \sigma^2_{-1}) \\ \sigma_0^2 &\sim \text{Inv-Gamma}(\alpha, \beta) \\ \tau_0^2 &\sim \text{Inv-Gamma}(\alpha, \beta) \end{aligned}$$

- The joint posterior, $p(\mu, \sigma^2, \mu_0, \sigma_0^2, \tau_0^2|y) \propto$:

$$p(\sigma_0^2)p(\tau_0^2)p(\mu_0|\sigma_0^2)p(\mu|\mu_0, \sigma_0^2)p(\sigma^2|\tau_0^2)p(\mu|\sigma^2)p(y|\mu, \sigma^2)$$

Complexity and Model Fit

- Hierarchical models quickly become complex and analytically intractible – so what's is the upside?
- Highly flexible approach to model fitting

Complexity and Model Fit

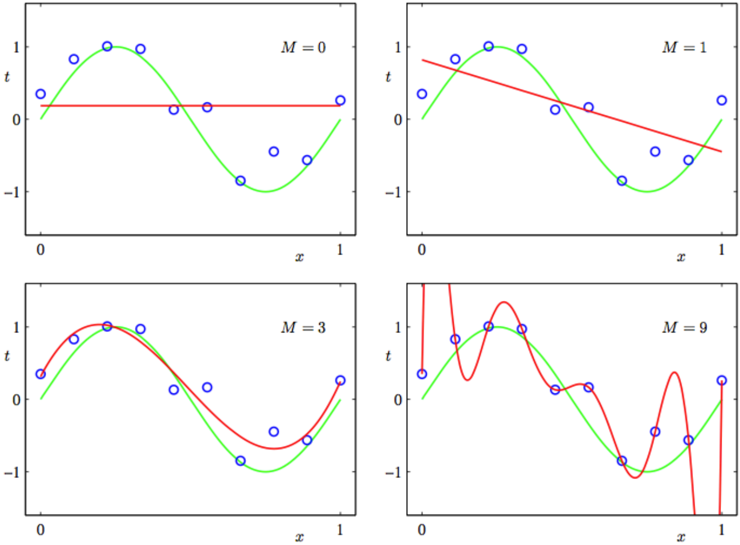
- Hierarchical models quickly become complex and analytically intractible – so what's is the upside?
- Highly flexible approach to model fitting
  - Sufficiently hierarchical models can capture any amount of complexity in a data process
  - Can check for and avoid 'overfitting' data – predicting very well in-sample well, but very poorly out-of-sample

Complexity and Model Fit

- Hierarchical models quickly become complex and analytically intractible – so what's is the upside?
- Highly flexible approach to model fitting
  - Sufficiently hierarchical models can capture any amount of complexity in a data process
  - Can check for and avoid 'overfitting' data – predicting very well in-sample well, but very poorly out-of-sample
- E.g., Fitting a polynomial function $\sin(2\pi x)$ for x on the domain of -1 to 0; (Bishop 2006)
- Use ridge regression to model an Mth-order polynomial with a penalty proportional to λ

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

Complexity and Model Fit



Shrinkage

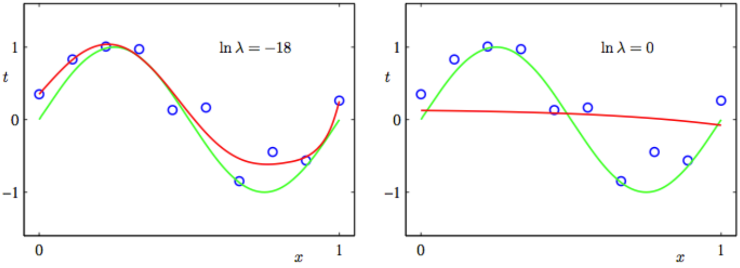


Table 1.2 Table of the coefficients $\mathbf{w}^\star$ for $M = 9$ polynomials with various values for the regularization parameter λ. Note that $\ln \lambda = -\infty$ corresponds to a model with no regularization, i.e., to the graph at the bottom right in Figure 1.4. We see that, as the value of λ increases, the typical magnitude of the coefficients gets smaller.

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

MCMC Estimation

- Hierarchical and multiparameter models can **often** be estimated using numerical sampling approaches

MCMC Estimation

- Hierarchical and multiparameter models can **often** be estimated using numerical sampling approaches
- Markov Chain Monte Carlo (MCMC) simulation
  - Standard practice for Bayesian inference
  - Draw θ from an approximating distribution, then improve until the draws converge to the true posterior $p(\theta|y)$
  - The draws form a Markov chain – made using a probability distribution depending only on the previous draw θ_{t-1}
  - Convergence is due to the stepwise improvement in the approximating distribution

Gibbs Sampler

- Divide θ into d components or subvectors, then draw $\theta_1, \theta_2, \ldots, \theta_d$ parameters sequentially, holding the remaining subvectors at their previous values
- Bivariate normal distribution:

$$\left(\begin{array}{c}\theta_1\\ \theta_2\end{array}\right) \Big| \, y \sim N\left\{\left(\begin{array}{c}y_1\\ y_2\end{array}\right), \left(\begin{array}{cc}1 & \rho\\ \rho & 1\end{array}\right)\right\}$$

```
theta.gibbs.1 <- function(y,theta,p){
   rnorm(y[1]+p*(theta[2]-y[2]),1-p^2,n=1)}

theta.gibbs.2 <- function(y,theta,p){
   rnorm(y[2]+p*(theta[1]-y[1]),1-p^2,n=1)}

# four starting chains
y1=c(2.5,2.5); y2=c(-2.5,-2.5)
y3=c(2.5,-2.5); y4=c(-2.5,2.5)
```

Gibbs Sampler

```
chains <- matrix(NA, 1000, 8); p <- .65
chains[1,] <- c(y1, y2, y3, y4); y <- c(0,0)

for(i in 2:1000){
 chains[i,c(1,3,5,7)] <-
 c(theta.gibbs.1(y,theta=chains[i-1,1:2],p),
   theta.gibbs.1(y,theta=chains[i-1,3:4],p),
   theta.gibbs.1(y,theta=chains[i-1,5:6],p),
   theta.gibbs.1(y,theta=chains[i-1,7:8],p))

 chains[i,c(2,4,6,8)] <-
 c(theta.gibbs.2(y,theta=chains[i,1:2],p),
   theta.gibbs.2(y,theta=chains[i,3:4],p),
   theta.gibbs.2(y,theta=chains[i,5:6],p),
   theta.gibbs.2(y,theta=chains[i,7:8],p))
}
burnin <- 500
```

Gibbs Sampler

```
plot(chains[,1:2],col='white',
     ylim=c(-3,3),xlim=c(-3,3))

for(i in 2:100){lines(lty=2,col='red',
   x=c(chains[i-1,1],chains[i,1]),
   y=c(chains[i-1,2],chains[i,2]))}

for(i in 2:100){lines(lty=2,col='blue',
   x=c(chains[i-1,3],chains[i,3]),
   y=c(chains[i-1,4],chains[i,4]))}

for(i in 2:100){lines(lty=2,col='darkgreen',
   x=c(chains[i-1,5],chains[i,5]),
   y=c(chains[i-1,6],chains[i,6]))}

for(i in 2:100){lines(lty=2,col='darkorange',
   x=c(chains[i-1,7],chains[i,7]),
   y=c(chains[i-1,8],chains[i,8]))}
```

The Metropolis Algorithm

- Gibbs generally requires tractible marginals – how do we sample from arbitrary posteriors?

The Metropolis Algorithm

- Gibbs generally requires tractible marginals – how do we sample from arbitrary posteriors?
- The Metropolis Algorithm:
  1. Draw θ^0 from a symmetric starting distribution $p(\theta^0|y) > 0$

The Metropolis Algorithm

- Gibbs generally requires tractible marginals – how do we sample from arbitrary posteriors?
- The Metropolis Algorithm:
  1. Draw θ^0 from a symmetric starting distribution $p(\theta^0|y) > 0$
  2. For subsequent draws $t = 1, 2, \dots T$

The Metropolis Algorithm

- Gibbs generally requires tractible marginals – how do we sample from arbitrary posteriors?
- The Metropolis Algorithm:
    1. Draw θ^0 from a symmetric starting distribution $p(\theta^0|y) > 0$
    2. For subsequent draws $t = 1, 2, \ldots T$
        a. Sample a proposal θ^* from a jumping distribution $J^t(\theta^*|\theta^{t-1})$; symmetric is $J^t(\theta^*|\theta^{t-1}) = J^t(\theta^{t-1}|\theta^*)$

The Metropolis Algorithm

- Gibbs generally requires tractible marginals – how do we sample from arbitrary posteriors?
- The Metropolis Algorithm:
  1. Draw θ^0 from a symmetric starting distribution $p(\theta^0|y) > 0$
  2. For subsequent draws $t = 1, 2, \ldots T$
     a. Sample a proposal θ^* from a jumping distribution $J^t(\theta^*|\theta^{t-1})$; symmetric is $J^t(\theta^*|\theta^{t-1}) = J^t(\theta^{t-1}|\theta^*)$
     b. Calculate the ratio of the densities:

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

The Metropolis Algorithm

- Gibbs generally requires tractible marginals – how do we sample from arbitrary posteriors?
- The Metropolis Algorithm:
  1. Draw θ^0 from a symmetric starting distribution $p(\theta^0|y) > 0$
  2. For subsequent draws $t = 1, 2, \ldots T$
     a. Sample a proposal θ^* from a jumping distribution $J^t(\theta^*|\theta^{t-1})$; symmetric is $J^t(\theta^*|\theta^{t-1}) = J^t(\theta^{t-1}|\theta^*)$
     b. Calculate the ratio of the densities:
     $$r = \frac{p(\theta^*|y)}{p(\theta^{t-1|y})}$$
     c. Set
     $$\theta^t = \begin{cases} \theta^* & \text{with probability} \min(r, 1) \\ \theta^{t-1} & \text{otherwise} \end{cases}$$

The Metropolis Algorithm

- Gibbs generally requires tractible marginals – how do we sample from arbitrary posteriors?
- The Metropolis Algorithm:
  1. Draw θ^0 from a symmetric starting distribution $p(\theta^0|y) > 0$
  2. For subsequent draws $t = 1, 2, \ldots T$
     a. Sample a proposal θ^* from a jumping distribution $J^t(\theta^*|\theta^{t-1})$; symmetric is $J^t(\theta^*|\theta^{t-1}) = J^t(\theta^{t-1}|\theta^*)$
     b. Calculate the ratio of the densities:
     $$r = \frac{p(\theta^*|y)}{p(\theta^{t-1|y})}$$
     c. Set
     $$\theta^t = \begin{cases} \theta^* & \text{with probability} \min(r, 1) \\ \theta^{t-1} & \text{otherwise} \end{cases}$$
- The transition distribution $T^t(\theta^t|\theta^{t-1})$ is a mixture of a weighting given to J^t and a point mass at $\theta^t = \theta^{t-1}$

Metropolis Bivariate Regression

```
set.seed(1005); x <- rnorm(n=40);
y <- .1 + 3.5*x + rnorm(n=40,sd=3)

loglike <- function(param){
  a = param[1]; b = param[2]; sds= param[3];
  sll <- dnorm(y, a + b*x, sd = sds, log = T);
  sum_sll <- sum(sll); return(sum_sll)}

prior <- function(param){
  a <- param[1]; b = param[2]; sds = param[3];
  apr <- dnorm(a, sd=6, log = T)
  bpr <- dnorm(b, sd=6, log = T)
  sdpr <- dunif(sds, min=0, max=30, log = T)
  return(apr+bpr+sdpr)}

poster <- function(param){
  return(loglike (param) + prior(param))}
```

Metropolis Bivariate Regression

```
propose <- function(param){
  return(rnorm(3,mean=param,sd=c(1,1,0.5)))}

runMCMC <- function(starts, iters){
  chain = matrix(NA,iters+1,3)
  probs = matrix(NA,iters+1,2)
  chain[1,] = starts
  for (i in 1:iters){
   prop = propose(chain[i,])
   probab = exp(poster(prop) - poster(chain[i,]))
   r=min(probab,1)
   ifs=sample(c(1,0),replace=F,
       prob=c(r,1-r),size=1)
   if(ifs==1){
     chain[i+1,] = prop
   } else if(ifs==0){
     chain[i+1,] = chain[i,]}}
  return(chain)}
```

Bayesian Inference in R

- R has many packages to conduct and evaluate Bayesian inference
  - `www.cran.r-project.org/web/views/Bayesian.html`
  - 'MCMCPack'
  - 'mcmc'