Bayesian Inference part II

Loukia Meligkotsidou , Department of Mathematics, NKUA

MSc in Statistics and OR

R functions available at eclass website

Overview of *MCMC*

This part of the course will focus mostly on a class of very powerful simulation algorithms, known as Markov Chain Monte Carlo (MCMC). These algorithms allow us to tackle problems of real complexity that were impossible (or ex- tremely difficult) to handle before.

MCMC methods are used in Bayesian inference to deal with multi-parameter problems, in which the posterior distribution can not be calculated analytically. Instead, MCMC algorithms are used to simulate draws from the joint posterior distribution of the model parameters. The most important MCMC algorithms are the Gibbs sampler and the Metropolis-Hastings algorithm.

This course will be focussed on the application of MCMC methods to Bayesian inference.

Contents

1. Overview of Bayesian Inference
2. Gibbs Sampling
3. Data Augmentation
4. The Metropolis-Hastings Algorithm
5. MCMC in GLM

Overview of Bayesian Inference

Let $\mathbf{y} = (y_1, \ldots, y_n)$ be a vector of observations with sampling density (likelihood) $f(\mathbf{y}|\theta)$.

Let $\pi(\theta)$ be the prior distribution of the unknown parameter θ.

The posterior distribution is computed via Bayes' theorem:

$$\pi(\theta|\mathbf{y}) = \frac{f(\mathbf{y}|\theta)\pi(\theta)}{f(\mathbf{y})} \propto f(\mathbf{y}|\theta)\pi(\theta) \quad (posterior \propto likelihood \times prior),$$

where

$$f(\mathbf{y}) = \int f(\mathbf{y}|\theta)\pi(\theta)d\theta \quad (marginal \quad likelihood).$$

Forecasting a future (or missing) value of an observation y_f is based on the predictive distribution:

$$f(y_f|\mathbf{y}) = \int f(y_f|\theta)\pi(\theta|\mathbf{y})d\theta.$$

Quantities of Interest

Often interest lies in the posterior expectation of some function $t(\theta)$:

$$E[t(\theta)|\mathbf{y}] = \int t(\theta)\pi(\theta|\mathbf{y})d\theta.$$

- if $t(\theta) = \theta$, the posterior expectation of the parameter θ;
- if $t(\theta) = \theta^r$, for some $r > 1$, higher order posterior moments of θ;
- if $t(\theta) = I[\theta \in A]$, then $E[t(\theta)|\mathbf{y}]$ is the posterior probability that the parameter θ lies in the set A;
- if $t(\theta) = f(y_f|\theta)$, then $E[t(\theta)|\mathbf{y}]$ is the predictive density $f(y_f|\mathbf{y})$

Other features of interest might be quantiles (*e.g.* median or certain percentiles) of the posterior distribution.

If $\theta = (\theta_1, \ldots, \theta_d)$ is multi-dimensional, we are also interested in the marginal posterior distributions of the components: $\pi(\theta_j|\mathbf{y})$ for $j = 1, \ldots, d$.

Conjugacy

The posterior belongs to the same family of distributions as the prior.

Example: $y_1, \ldots, y_n$ i.i.d. $\sim \text{Exp}(\theta) = \text{Gamma}(1, \theta)$

$$\Longrightarrow f(\mathbf{y}|\theta) = \prod_{i=1}^{n} f(y_i|\theta) = \theta^n \exp\big[-\theta \sum_{i=1}^{n} y_i\big]$$

Prior: $\theta \sim \text{Gamma}(2, 1) \Longrightarrow \pi(\theta) = \theta \exp(-\theta)$
Posterior: $\pi(\theta|\mathbf{y}) \propto f(\mathbf{y}|\theta)\pi(\theta) \propto \theta^{1+n} \exp\big[-\theta(1 + \sum_i y_i)\big]$

$$\Longrightarrow \theta|\mathbf{y} \sim \text{Gamma}\big(2 + n, 1 + \sum_{i=1}^{n} y_i\big)$$

In many problems with a single parameter, it is possible to find useful conjugate priors. But, in multi-parameter models, it is much more difficult.

Conditional Conjugacy

In multi-parameter models, we can often find conditionally conjugate priors.

Example 2.1: $Y_i|\mu,\omega \sim N(\mu, 1/\omega)$, independently for $i = 1, \ldots, n$.

Prior: μ and ω independent, $\pi(\mu) \sim N(\mu_0, 1/\kappa_0)$, $\pi(\omega) \sim \text{Gamma}\,(\alpha_0, \lambda_0)$

$\Longrightarrow$ The posterior,

$$\begin{aligned}
\pi(\mu,\omega|\mathbf{y}) &\propto f(\mathbf{y} \mid \mu,\omega)\pi(\mu)\pi(\omega) \\
&\propto \omega^{n/2} \exp\left[-\frac{\omega}{2}\sum_{i=1}^{n}(y_i-\mu)^2\right] \\
&\times \exp\left[-\frac{\kappa_0}{2}(\mu-\mu_0)^2\right]\omega^{\alpha_0-1}\exp\left[-\lambda_0\omega\right] I[\omega>0]
\end{aligned}$$

is a complex bi-variate density.

However, the conditional posterior densities,

$$\begin{aligned}
\pi(\mu|\omega,\mathbf{y}) &\propto \exp\left[-\frac{\omega}{2}\sum_{i=1}^{n}(y_i-\mu)^2\right]\exp\left[-\frac{\kappa_0}{2}(\mu-\mu_0)^2\right] \\
&\propto \exp\left[-\frac{1}{2}(\kappa_0+\omega n)\left(\mu-\frac{\kappa_0\mu_0+\omega\sum_{i=1}^{n}y_i}{\kappa_0+n\omega}\right)^2\right] \\
&\Longrightarrow \mu|\omega,\mathbf{y}\sim N\left(\frac{\kappa_0\mu_0+\omega\sum_{i=1}^{n}y_i}{\kappa_0+n\omega},\frac{1}{\kappa_0+n\omega}\right),
\end{aligned}$$

$$\begin{aligned}
\pi(\omega|\mu,\mathbf{y}) &\propto \exp\left[-\frac{\omega}{2}\sum_{i=1}^{n}(y_i-\mu)^2\right]\omega^{\frac{n}{2}+\alpha_0-1}\exp[-\lambda_0\omega] \\
&\Longrightarrow \omega|\mu,\mathbf{y}\sim \text{Gamma}\left(\alpha_0+\frac{n}{2},\lambda_0+\frac{\sum_{i=1}^{n}(y_i-\mu)^2}{2}\right),
\end{aligned}$$

are tractable, and they belong to the same families of distributions as the priors (conditional conjugacy).

MCMC

In multi-dimensional Bayesian problems it is rarely possible to compute analytically summary statistics such as posterior means and variances, or even posterior probabilities.

Therefore, it is necessary to estimate the quantities of interest using a Monte Carlo approach. However, simulating from an arbitrary high dimensional distribution is usually difficult and often impossible to do directly. Instead, Markov chain Monte Carlo (MCMC) methods are used to simulate a Markov chain, whose stationary or limiting distribution is the posterior distribution of interest.

The concept of conditional conjugacy is crucial in the construction of one of the basic forms of MCMC: the Gibbs sampler.

The Gibbs Sampler

When the conditional posterior distributions are of known form, Gibbs sampling is possible.

Objective: To obtain a sample from a multivariate target distribution $\pi(\theta_1, \ldots, \theta_d)$. In Bayesian statistics, the target is the joint posterior.

The Gibbs sampling algorithm

1. Initialize with $\theta = (\theta_1^{(0)}, \ldots, \theta_d^{(0)})$.
2. Simulate $\theta_1^{(1)}$ from the conditional distribution $\pi(\theta_1 | \theta_2^{(0)}, \theta_3^{(0)} \ldots, \theta_d^{(0)})$.
3. Simulate $\theta_2^{(1)}$ from the conditional distribution $\pi(\theta_2 | \theta_1^{(1)}, \theta_3^{(0)}, \ldots, \theta_d^{(0)})$.
4. $\ldots$
5. Simulate $\theta_d^{(1)}$ from the conditional distribution $\pi(\theta_d | \theta_1^{(1)}, \theta_2^{(1)}, \ldots, \theta_{d-1}^{(1)})$.
6. Iterate this procedure.

Under mild regularity conditions, the above Markov chain converges to draws from the stationary distribution $\pi(\theta_1, \ldots, \theta_d)$. So, after a burn–in period, the subsequent draws $\theta^{(1)}, \ldots, \theta^{(J)}$ can be regarded as realizations from this distribution.

With the Gibbs sampler, we obtain a sample from the joint posterior distribution without worrying about computing its normalising constant (needed for exact inference).

Once we obtain a sample $\theta^{(1)}, \ldots, \theta^{(J)}$ from $\pi(\theta|\mathbf{y})$ we can approximate any feature of the posterior using the draws.

- Posterior mean of $t(\theta)$: $E[t(\theta)|\mathbf{y}] \approx \frac{1}{J}\sum_{j=1}^{J} t(\theta^{(j)})$.
- Posterior quantiles: Order the draws $\theta_{[1]} < \theta_{[2]} < \ldots < \theta_{[J]}$. Then,

Median of $\pi(\theta|\mathbf{y}) \approx \theta_{[J/2]}$
95% credible region $\approx \left(\theta_{[J\times 0.025]}, \theta_{[J\times 0.975]}\right)$

If $\theta = (\theta_1, \ldots, \theta_d)$, then the ith component of each of the draws $\theta^{(1)}, \ldots, \theta^{(J)}$ constitutes a sample from $\pi(\theta_i|\mathbf{y})$.

Gibbs sampling for Normal example

1. Start the chain with some values $(\mu^{(0)}, \omega^{(0)})$.
2. Simulate $\mu^{(1)}$ from the distribution $N\left(\frac{\kappa_0\mu_0+\omega^{(0)}\sum_{i=1}^n y_i}{\kappa_0+n\omega^{(0)}}, \frac{1}{\kappa_0+n\omega^{(0)}}\right)$.
3. Simulate $\omega^{(1)}$ from the distribution Gamma $\left(\alpha_0+\frac{n}{2}, \lambda_0+\frac{\sum_{i=1}^n(y_i-\mu^{(1)})^2}{2}\right)$.
4. Iterate this procedure.

The R function **`gibbs`** implements this algorithm in R. I have stored this function in an external file called "gibbs.r". To be able to use this function, make sure you store the file "gibbs.r" in the same directory where you are going to start R. Once you start an R session, simply do

```
> source("gibbs.r")
```

and you will have the function **`gibbs`** available for use.

Implementing Gibbs sampling with R

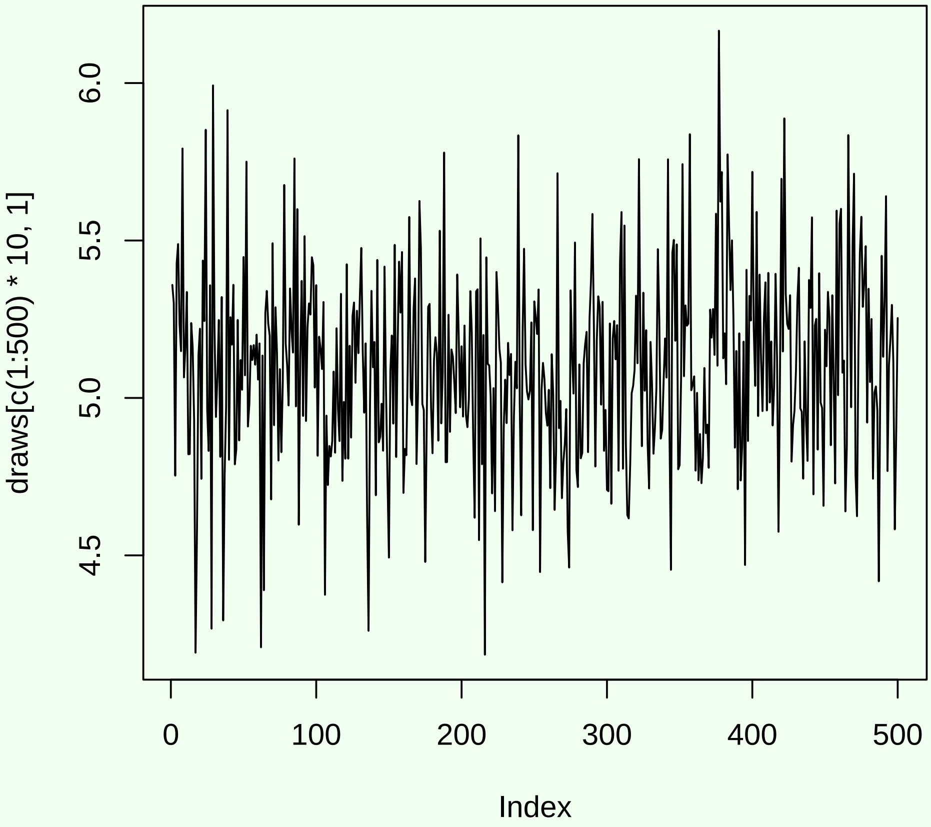
I have generated a vector of observations $\mathbf{y}$ consisting of 50 independent realizations from a Normal distribution with $\mu = 5$ and $\omega = 0.2$ (so the variance is $(1/\omega) = 5$).
To apply the function `gibbs` to the data-set $\mathbf{y}$ do:

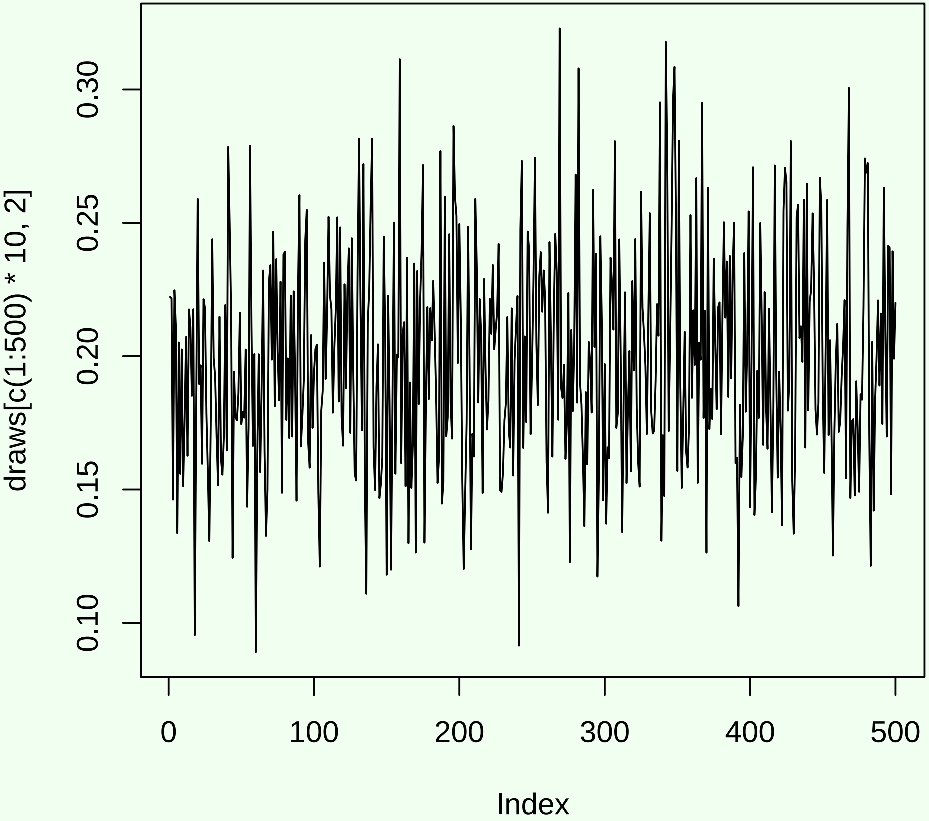
```
> draws <- gibbs(data=y,mu0=3,kappa0=1,alpha0=0.1,
                     lambda0=0.1,nburn=0,ndraw=5000)
```

The draws from $\pi(\mu, \omega|\mathbf{y})$ have been stored in a $5,000 \times 2$ matrix called `draws`. The first column contains the draws of μ and the second column the draws of ω. To visualize these draws, we can plot, for example, 500 equally spaced draws (out of the 5,000) for μ and similarly for ω.

```
> plot(draws[c(1:500)*10,1],type="l")
```

```
> plot(draws[c(1:500)*10,2],type="l")
```


draws[c(1:500) * 10, 1]
4.5
5.0
5.5
6.0
0
100
200
300
400
500
Index


draws[c(1:500) * 10, 2]
0.10
0.15
0.20
0.25
0.30
0
100
200
300
400
500
Index

Convergence of the algorithm

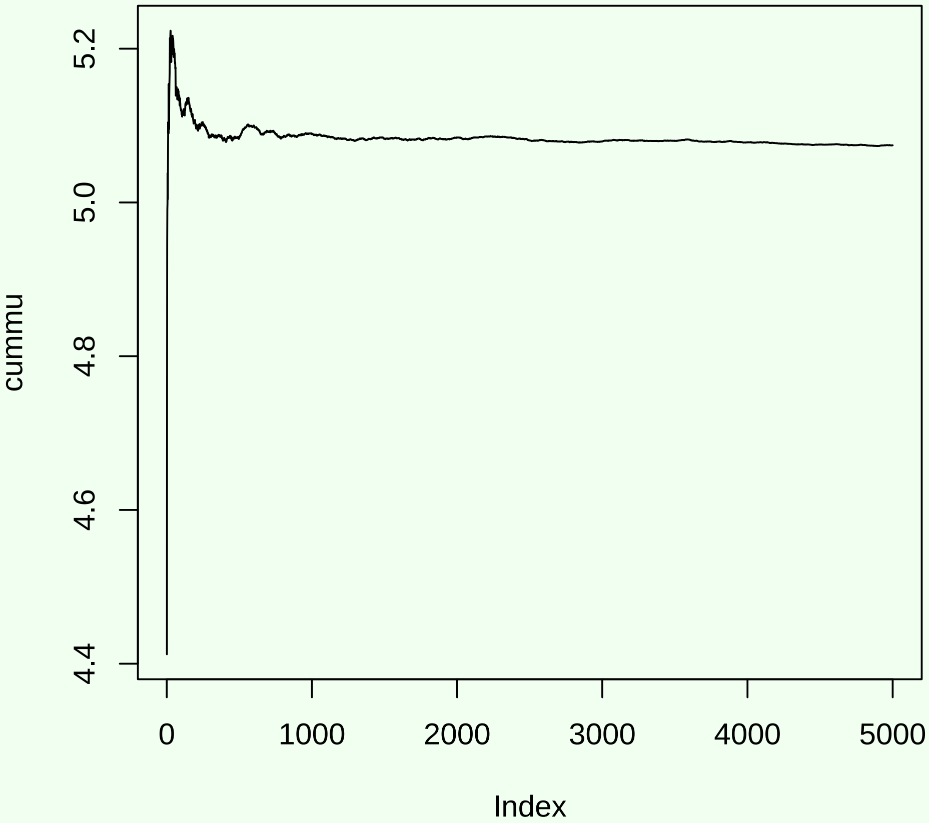
An informal way to get an idea of the convergence of the sampler is to plot the posterior mean of the parameters against the number of iterations of the sampler. If convergence is achieved, the mean should settle around a certain value. We do this for μ and ω:

```
> cum <- cumsum(draws[,1])/c(1:5000)
> plot(cum,type="l")
```

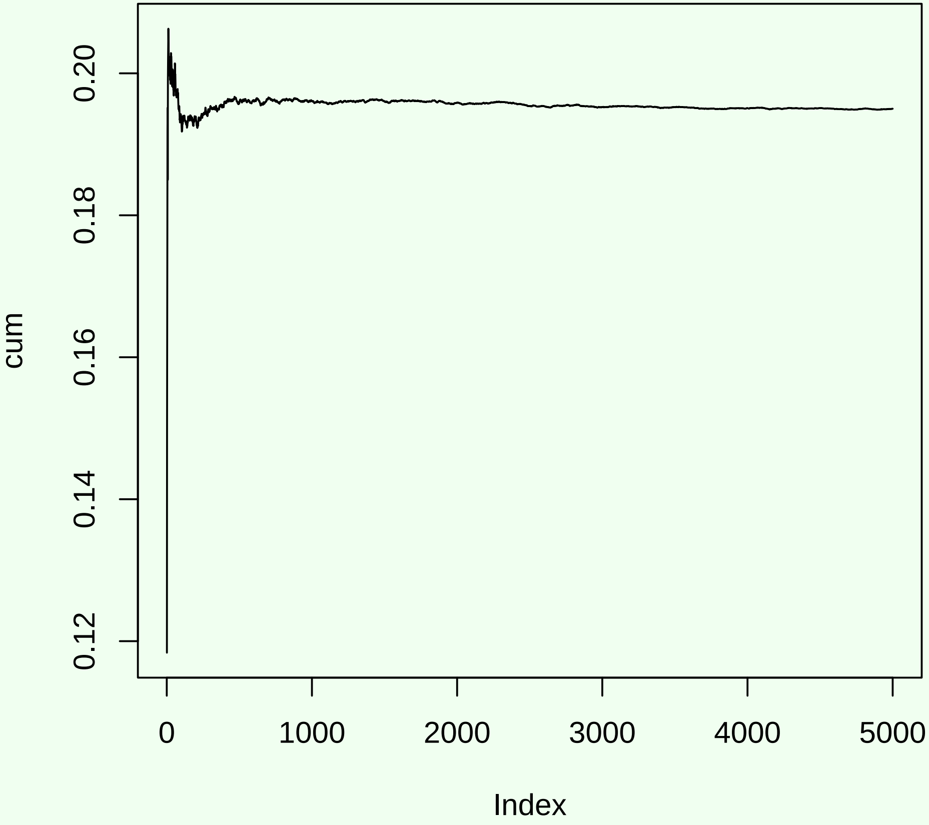
Note that the posterior mean for μ has settled at a value that is in between its prior mean $\mu_0 = 3$ and the sample mean, which is:

```
> mean(y)
[1] 5.296271
```

```
> cum <- cumsum(draws[,2])/c(1:5000)
> plot(cum,type="l")
```



cummu
5.2
5.0
4.8
4.6
4.4
0
1000
2000
3000
4000
5000
Index


cum
0.12
0.14
0.16
0.18
0.20
0
1000
2000
3000
4000
5000
Index

Inference

The plots suggest that convergence has been achieved after about 1,000 iterations of the sampler. So we remove the first 1,000 draws (the burn-in) and keep the remaining 4,000 to conduct inference and prediction.

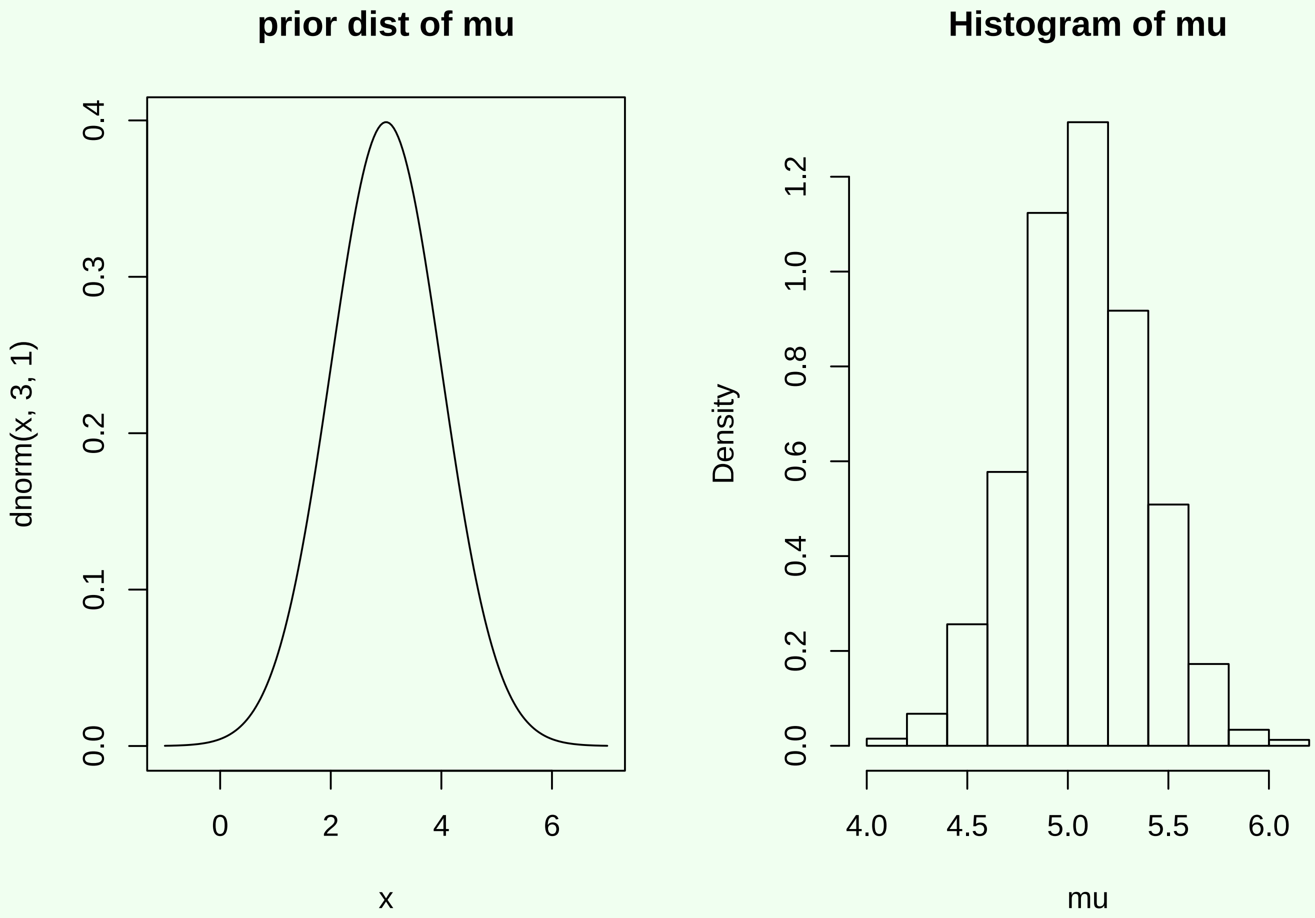
```
> mu <- draws[c(1001:5000),1]
> omega <- draws[c(1001:5000),2]
```

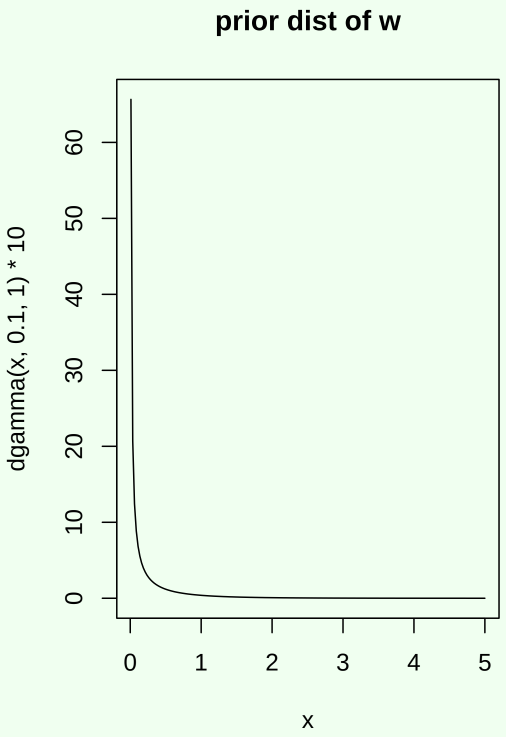
To plot the prior and posterior distributions of μ do:

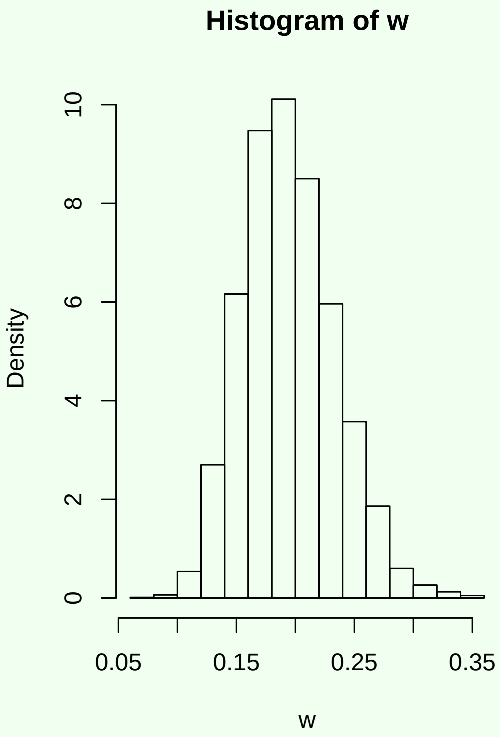
```
> par(mfrow=c(1,2))
> x <- seq(-1,7,length=200)
> plot(x,dnorm(x,3,1),type="l")
> title("prior dist of mu")
> hist(mu,probab=T)
```

To plot the prior and posterior distributions of ω do:

```
> x <- seq(0.01,5,length=200)
> plot(x,dgamma(x,0.1,rate=0.1),type="l")
> title("prior dist of w")
> hist(omega,probab=T)
```


prior dist of mu
dnorm(x, 3, 1)
0.0
0.1
0.2
0.3
0.4
0
2
4
6
x
Histogram of mu
Density
0.0
0.2
0.4
0.6
0.8
1.0
1.2
4.0
4.5
5.0
5.5
6.0
mu


prior dist of w
dgamma(x, 0.1, 1) * 10
0
10
20
30
40
50
60
0
1
2
3
4
5
x


Histogram of w
Density
0
2
4
6
8
10
0.05
0.15
0.25
0.35
w

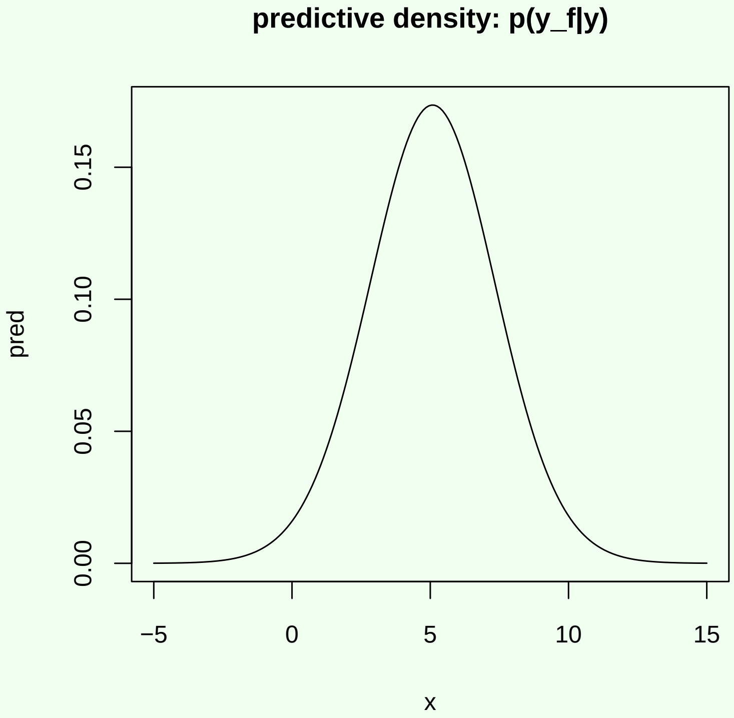
Prediction

Now suppose we want to forecast an observable y_f. This is done according to the predictive distribution, which has density

$$\begin{aligned} f(y_f|\mathbf{y}) &= \int f(y_f|\mu,\omega)\pi(\mu,\omega|\mathbf{y})d\mu d\omega \\ &\approx \frac{1}{J}\sum_{j=1}^{J}\left(\frac{\omega^{(j)}}{2\pi}\right)^{1/2}\exp\left\{-\frac{\omega^{(j)}}{2}(y_f-\mu^{(j)})^2\right\} \end{aligned}$$

where $(\mu^{(j)},\omega^{(j)})$ are the draws obtained through the Gibbs sampler.
To do this with R:

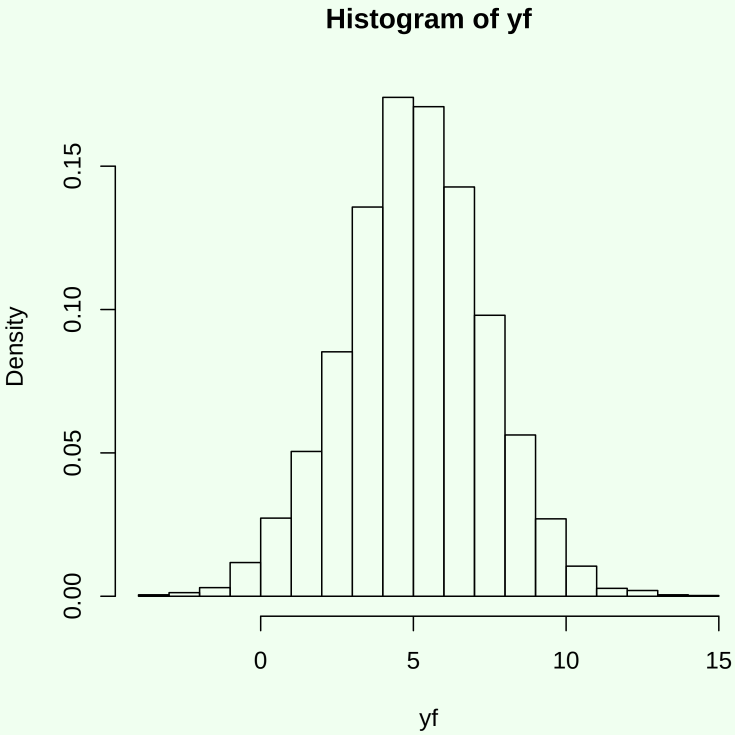
```
> x <- seq(-5,15,length=200)
> pred <- numeric(200)
> for(j in 1:200) pred[j] <- mean(dnorm(x[j],mu,1/sqrt(omega))
> plot(x,pred,type="l")
> title("predictive density: p(y_f|y)")
```


predictive density: p(y_f|y)
pred
0.00
0.05
0.10
0.15
−5
0
5
10
15
x

Prediction

If we want to compute some other characteristic of the predictive distribution, *e.g.* the probability that a future observable lies in the interval $(0, 5)$, we can take a sample of draws from the predictive distribution. So, for each value $(\mu^{(j)}, \omega^{(j)})$ drawn in the Gibbs sampler, we draw a value of y_f from a $N(\mu^{(j)}, 1/\omega^{(j)})$ distribution. In R:

```
> yf <- rnorm(4000,mu,1/sqrt(omega))
> pos <- yf[yf>0]
> length(pos[pos<5])/length(yf)
[1] 0.47275
> hist(yf,probab=T)
```


Histogram of yf
Density
0.00
0.05
0.10
0.15
0
5
10
15
yf

Poisson count change point problem

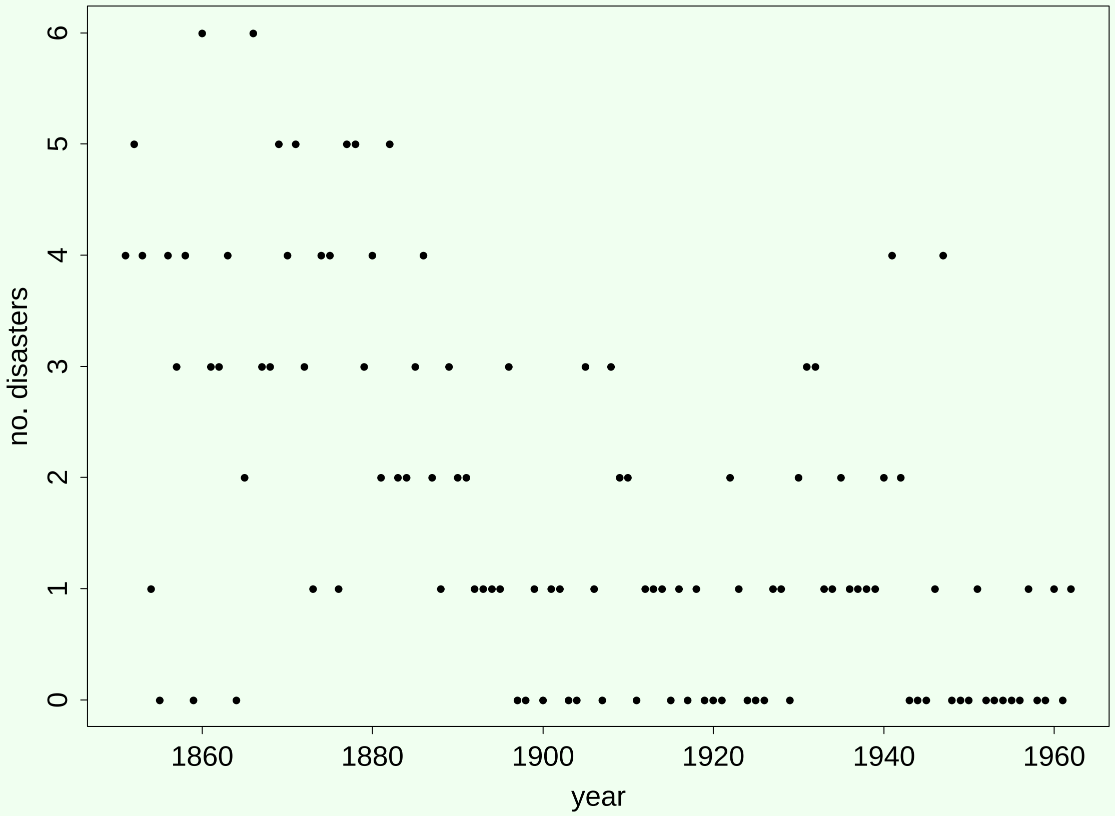
The time series stored in the file "coal.dat" gives the number of British coal mining disasters per year, over the period 1851 – 1962.

Let Y_i denote the number of disasters in year i, $i = 1, \ldots, 112$. A model that has been proposed in the literature has the form

$$Y_i \sim \text{Poisson } (\theta); \quad i = 1, \ldots k;$$

$$Y_i \sim \text{Poisson } (\lambda); \quad i = k+1, \ldots n.$$

Priors: $\theta \sim \text{Gamma } (a_1, b_1)$, $\lambda \sim \text{Gamma } (a_2, b_2)$, $k \sim$ discrete uniform over $\{1, \ldots, n\}$, each independent of one another, and then $b_1 \sim \text{Gamma } (c_1, d_1)$ and $b_2 \sim \text{Gamma } (c_2, d_2)$.



no. disasters
0
1
2
3
4
5
6
1860
1880
1900
1920
1940
1960
year

Likelihood:

$$f(\mathbf{y}_1 \mid \theta) = \prod_{i=1}^{k} \frac{e^{-\theta}\theta^{y_i}}{y_i!} \propto e^{-\theta k}\theta^{\sum_{i=1}^{k} y_i}$$

$$f(\mathbf{y}_2 \mid \lambda) = \prod_{i=k+1}^{n} \frac{e^{-\lambda}\lambda^{y_i}}{y_i!} \propto e^{-\lambda(n-k)}\lambda^{\sum_{i=k+1}^{n} y_i}$$

Priors:

$$\pi(\theta \mid a_1, b_1) \propto b_1^{a_1}\theta^{a_1-1}e^{-b_1\theta}$$

$$\pi(\lambda \mid a_2, b_2) \propto b_2^{a_2}\lambda^{a_2-1}e^{-b_2\lambda}$$

$$\pi(k) = \frac{1}{n}I[k \in \{1, 2, \ldots, n\}]$$

$$\pi(b_1 \mid c_1, d_1) \propto b_1^{c_1-1}e^{-d_1 b_1}$$

$$\pi(b_2 \mid c_2, d_2) \propto b_2^{c_2-1}e^{-d_2 b_2}$$

Joint posterior:

$$\begin{aligned}\pi(\theta, \lambda, k, b_1, b_2|\mathbf{y}) &\propto f(\mathbf{y}_1 \mid \theta) f(\mathbf{y}_2 \mid \lambda) \times \pi(\theta \mid a_1, b_1)\pi(\lambda \mid a_2, b_2)\\ &\quad \pi(b_1 \mid c_1, d_1)\pi(b_2 \mid c_2, d_2)\pi(k)\\ &\propto e^{-\theta k}\theta^{\sum_{i=1}^{k} y_i} e^{-\lambda(n-k)}\lambda^{\sum_{i=k+1}^{n} y_i} \times b_1^{a_1}\theta^{a_1-1}e^{-b_1\theta} b_2^{a_2}\lambda^{a_2-1}e^{-b_2\lambda}\\ &\quad b_1^{c_1-1}e^{-d_1 b_1} b_2^{c_2-1}e^{-d_2 b_2} I[k \in \{1, 2, \ldots, n\}]\end{aligned}$$

Conditional posteriors:

$$\pi(\theta|\mathbf{y}, \lambda, b_1, b_2, k) \propto e^{-\theta k}\theta^{\sum_{i=1}^{k} y_i}\theta^{a_1-1}e^{-b_1\theta} = e^{-\theta(b_1+k)}\theta^{\sum_{i=1}^{k} y_i + a_1 - 1}$$

$$\Longrightarrow \theta|\mathbf{y}, \lambda, b_1, b_2, k \sim \text{Gamma}\left(a_1 + \sum_{i=1}^{k} y_i, b_1 + k\right)$$

$$\pi(\lambda|\mathbf{y}, \theta, b_1, b_2, k) \propto e^{-\lambda(n-k)}\lambda^{\sum_{i=k+1}^{n} y_i}\lambda^{a_2-1}e^{-b_2\lambda} = e^{-\lambda(b_2+n-k)}\lambda^{\sum_{i=k+1}^{n} y_i + a_2 - 1}$$

$$\Longrightarrow \lambda|\mathbf{y}, \theta, b_1, b_2, k \sim \text{Gamma}\left(a_2 + \sum_{i=k+1}^{n} y_i, b_2 + n - k\right)$$

$$\pi(b_1|\mathbf{y},\theta,\lambda,b_2,k) \propto b_1^{a_1} e^{-b_1\theta} b_1^{c_1-1} e^{-d_1 b_1} = b_1^{c_1+a_1-1} e^{-b_1(d_1+\theta)}$$
$$\Longrightarrow b_1|\mathbf{y},\theta,\lambda,b_2,k \sim \text{Gamma}\,(c_1+a_1, d_1+\theta)$$

$$\pi(b_2|\mathbf{y},\theta,\lambda,b_1,k) \propto b_2^{a_2} e^{-b_2\lambda} b_2^{c_2-1} e^{-d_2 b_2} = b_2^{c_2+a_2-1} e^{-b_2(d_2+\lambda)}$$
$$\Longrightarrow b_2|\mathbf{y},\theta,\lambda,b_1,k \sim \text{Gamma}\,(c_2+a_2, d_2+\lambda)$$

$$P(k|\mathbf{y},\theta,\lambda,b_1,b_2) = \frac{e^{(\lambda-\theta)k}\ \ (\theta/\lambda)^{\sum_{i=1}^{k} y_i}}{\sum_{j=1}^{n}\left\{\ e^{(\lambda-\theta)j}\ \ \left(\frac{\theta}{\lambda}\right)^{\sum_{i=1}^{j} y_i}\right\}}\ I[k \in \{1,2,\ldots,n\}].$$

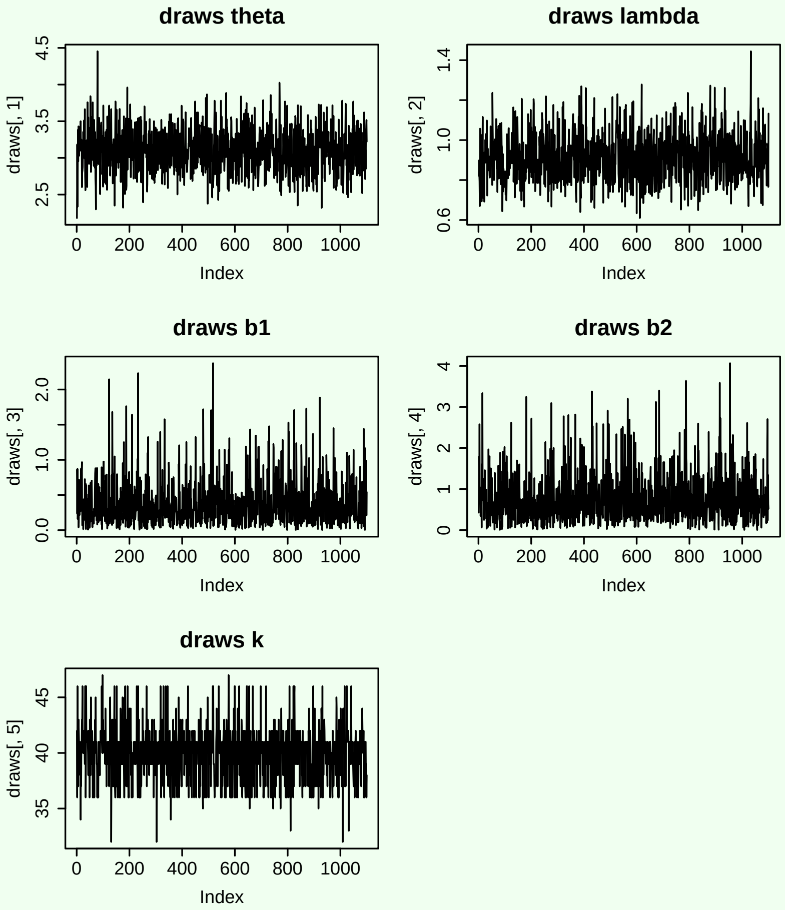
Convergence of the algorithm

The corresponding Gibbs sampler is implemented in with the code stored in the external file “gibbs2.r”. To run it with 1,100 iterations and no burn-in, and plot the output, I did:

```
> draws <- gibbs2(data=coal,ndraw=1100)
> par(mfrow=c(3,2))
> plot(draws[,1],type="l",main="draws theta")
> plot(draws[,2],type="l",main="draws lambda")
> plot(draws[,3],type="l",main="draws b1")
> plot(draws[,4],type="l",main="draws b2")
> plot(draws[,5],type="l",main="draws k")
```

Convergence of the algorithm was rapid, so I deleted the first 100 values and based the subsequent analysis on the remaining 1000 points:

```
> theta <- draws[c(101:1100),1]
> lambda <- draws[c(101:1100),2]
> k <- draws[c(101:1100),5]
```

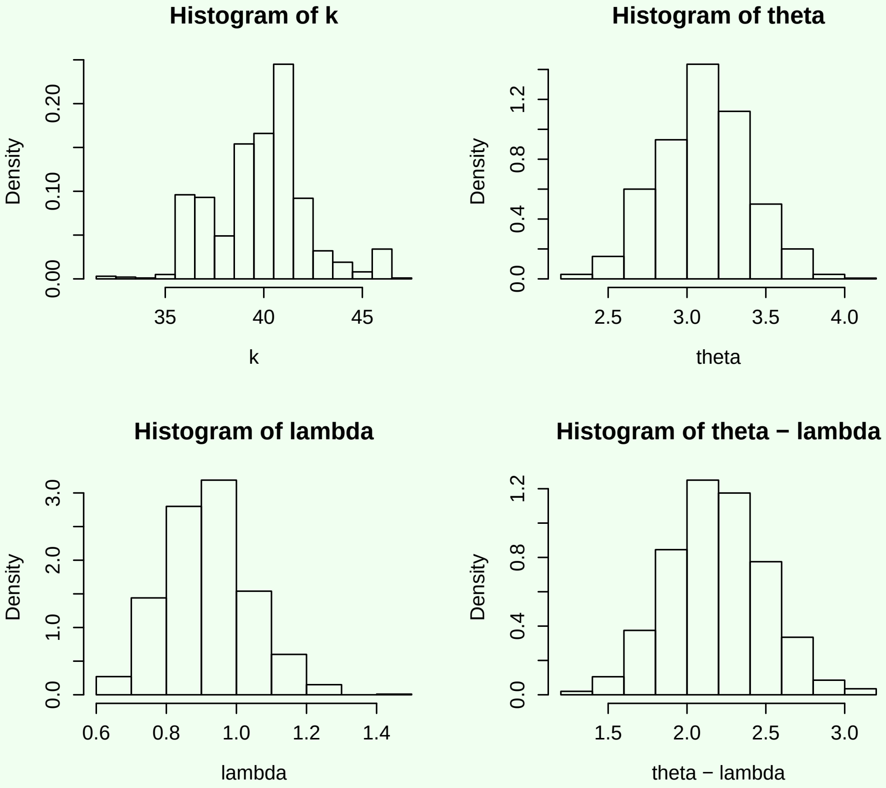

draws theta
draws[, 1]
2.5
3.5
4.5
0
200
400
600
800
1000
Index
draws lambda
draws[, 2]
0.6
1.0
1.4
0
200
400
600
800
1000
Index
draws b1
draws[, 3]
0.0
1.0
2.0
0
200
400
600
800
1000
Index
draws b2
draws[, 4]
0
1
2
3
4
0
200
400
600
800
1000
Index
draws k
draws[, 5]
35
40
45
0
200
400
600
800
1000
Index

Histograms

Histograms of the posterior distributions of the model parameters can be obtained in R as follows:

```
> par(mfrow=c(2,2))
> hist(k,breaks=seq(min(k)-0.5,max(k)+0.5,by=1),probab=T)
> hist(theta,probab=T)
> hist(lambda,probab=T)
> hist(theta-lambda,probab=T)
```

From the MCMC output, it is almost certain that a changepoint has occurred, with the posterior mode estimate being $k = 41$, corresponding to a changepoint at the year 1891.



Histogram of k

Histogram of theta

Histogram of lambda

Histogram of theta − lambda

Prediction

The predictive distribution of a future observation y_f given the current observations $\mathbf{y}$ is defined as

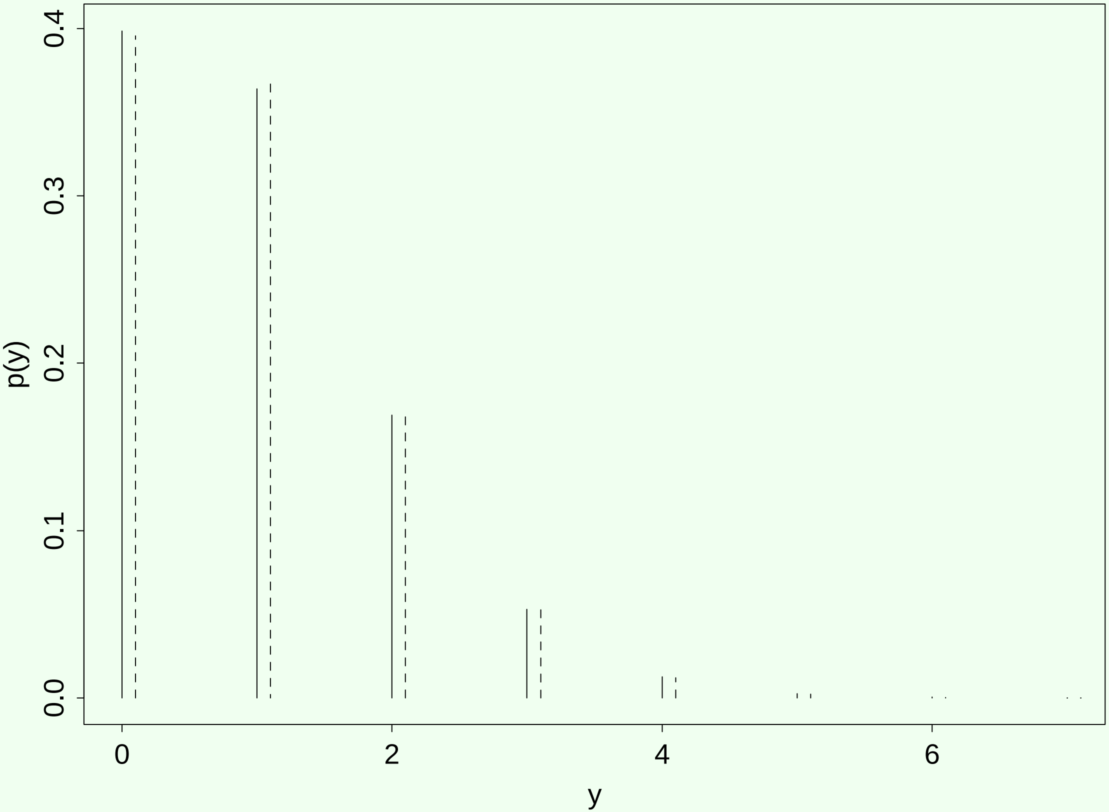
$$f(y_f|\mathbf{y}) = \int f(y_f|\theta)\pi(\theta|\mathbf{y})d\theta,$$

so that the likelihood, $f(y_f|\theta)$ is averaged across the uncertainty in θ contained in the posterior distribution $\pi(\theta|\mathbf{y})$.

Hence, given a sampled sequence of realizations $\theta^{(1)}, \ldots, \theta^{(J)}$ from this posterior, we can estimate

$$f(y_f|\mathbf{y}) \approx \frac{1}{J}\sum_{j=1}^{J} f(y_f|\theta^{(j)})$$

For the coal mining example, we can use this estimator to estimate the predictive distribution of the number of disasters in a future year for which the Poisson rate is λ.


0.4
0.3
0.2
0.1
0.0
p(y)
0
2
4
6
y

Data Augmentation

Augment the model by including additional (random) variables in a way that leads to a model that is easier to handle computationally

Data augmentation can be helpful in making problems amenable to Gibbs sampling

It can be applied equally well to the following two situations:

1. missing data
2. intractable likelihood function that becomes tractable if we condition on a collection of unobserved variables

Example 1

Suppose $Y_0, Y_1, \ldots, Y_n$ is a time series of random variables with $Y_0 = 0$ and for each $i = 1, \ldots, n$, $Y_i = Y_{i-1} + S_i$, where $S_i \sim Beta(\theta, \theta)$, $\theta > 0$. Therefore,

$$Y_i | Y_0, Y_1, \ldots Y_{i-1} \sim Y_{i-1} + S_i.$$

$$f(y_0, \ldots, y_n|\theta) = f(y_0|\theta) \prod_{i=1}^{n} f(y_i|y_0, \ldots, y_{i-1}, \theta) = \prod_{i=1}^{n} f(y_i|y_{i-1}, \theta)$$

$$= \prod_{i=1}^{n} \frac{\Gamma(2\theta)}{\{\Gamma(\theta)\}^2} (y_i - y_{i-1})^{\theta-1} \{1 - (y_i - y_{i-1})\}^{\theta-1} I[0 < y_i - y_{i-1} < 1]$$

If an observation y_{i^*} is missing $\Longrightarrow$ likelihood no longer available in closed form

Example 2

Suppose that $Y_1, \ldots Y_n$ are i.i.d. from the mixture density:

$$f(y_i|\theta) = \frac{1}{2}\left(\frac{1}{(2\pi)^{1/2}}\exp\left(-\frac{y_i^2}{2}\right) + \frac{1}{(2\pi)^{1/2}}\exp\left[-\frac{(y_i-\theta)^2}{2}\right]\right)$$

θ enters the likelihood in a non-straightforward way:

$$f(\mathbf{y}|\theta) = \prod_{i=1}^{n} f(y_i|\theta) \propto \prod_{i=1}^{n}\left(\exp\left(-\frac{y_i^2}{2}\right) + \exp\left[-\frac{(y_i-\theta)^2}{2}\right]\right) .$$

$\Longrightarrow$ no prior on θ will lead to a recognisable posterior

This mixture density can be thought of as the density of the random variable obtained by the following procedure.

Toss a fair coin (1 coin toss per observation y_i):
If head $\Longrightarrow$ sample y_i from a $N(0,1)$ distribution.
If tail $\Longrightarrow$ sample y_i from a $N(\theta,1)$ distribution.

If I knew the result of the n coin tosses, I would know whether $y_i \sim N(0,1)$ or $y_i \sim N(\theta, 1)$ for each $i = 1, \ldots, n$, and the likelihood would be

$$f(y_1, \ldots, y_n | \theta, \text{sequence of H and T}) \propto \prod_{i:\text{toss} \ = \ \text{T}} \exp\left[-\frac{(y_i - \theta)^2}{2}\right],$$

which is very easy to handle!

In these examples exact inference is not possible, both in a Bayesian and in a classical framework. Instead, we need to use Computationally intensive methods.

Data augmentation ideas, used in a Bayesian framework, are very similar to the EM algorithm for maximum likelihood inference.

The Idea

Remark: in a Bayesian setting both data and parameters are treated as random variables.

When doing data augmentation, we add further random variables (denoted by z) to the model. Then the above examples demonstrate the situation where the likelihood of the observed data, $f(\mathbf{y}|\theta)$, is not tractable, but complete data likelihood, $f(\mathbf{y}, z|\theta)$, is easy to handle. As a result, the posterior distribution of (θ, z) is proportional to

$$\pi(\theta, z|\mathbf{y}) \propto f(\mathbf{y}, z|\theta)\pi(\theta).$$

Data augmentation proceeds by carrying out Gibbs sampling to sample successively from θ and z to produce a sample from this joint distribution. The marginal distribution of θ is the posterior distribution of interest.

Example 1

In this case, $\mathbf{y}$ is the vector $(y_0, \ldots, y_n)$ excluding y_{i^*}, $z = y_{i^*}$, and hence

$$f(\mathbf{y}, y_{i^*}|\theta) = f(y_0, \ldots, y_n|\theta)$$

$$\propto \prod_{i=1}^{n} \frac{\Gamma(2\theta)}{\{\Gamma(\theta)\}^2}(y_i - y_{i-1})^{\theta-1}\{1-(y_i - y_{i-1})\}^{\theta-1} I[0 < y_i - y_{i-1} < 1]$$

Therefore, the conditional posterior $\pi(\theta|\mathbf{y}, y_{i^*}) \propto f(\mathbf{y}, y_{i^*}|\theta)\pi(\theta)$ is explicit and can be sampled easily.

To complete the Gibbs sampler, we also need to sample from

$$f(y_{i^*}|\mathbf{y}, \theta) \propto f(\mathbf{y}, y_{i^*}|\theta)$$

$$\propto \Big[(y_{i^*} - y_{i^*-1})\{1-(y_{i^*} - y_{i^*-1})\}(y_{i^*+1} - y_{i^*})\{1-(y_{i^*+1} - y_{i^*})\}\Big]^{\theta-1}$$

on the region $y_{i^*} \in (y_{i^*-1}, y_{i^*-1}+1) \cap (y_{i^*+1}-1, y_{i^*+1})$. This sampling can be carried out (among other ways) by rejection sampling.

Example 2

Here z is a sequence of n heads or tails, with one element per observation. Hence, $z = (z_1, \ldots, z_n)$ and z_i equals 1 if observation i corresponds to head and 2 if it corresponds to tail. Suppose that the prior for θ is $N(0, 1)$. Then we have

$$f(y_i, z_i|\theta) = f(y_i|z_i, \theta)P(z_i),$$

where

$$f(y_i|z_i, \theta) = \begin{cases} \frac{1}{(2\pi)^{1/2}} \exp\left(-\frac{y_i^2}{2}\right) & \text{if } z_i = 1 \\ \frac{1}{(2\pi)^{1/2}} \exp\left(-\frac{(y_i-\theta)^2}{2}\right) & \text{if } z_i = 2 \end{cases}$$

and

$$P(z_i = 1) = P(z_i = 2) = \frac{1}{2}.$$

The complete data likelihood is

$$\begin{aligned} f(\mathbf{y}, \mathbf{z} \mid \theta) &= \left\{ \prod_{i=1}^{n} f(y_i | z_i, \theta) P(z_i) \right\} \\ &\propto \exp\left\{ -\frac{1}{2} \sum_{i:z_i=1} y_i^2 \right\} \exp\left\{ -\frac{1}{2} \sum_{i:z_i=2} (y_i - \theta)^2 \right\} \end{aligned}$$

Using that $\pi(\theta, \mathbf{z}|\mathbf{y}) \propto f(\mathbf{y}, \mathbf{z} \mid \theta)\pi(\theta)$, it follows that

$$\pi(\theta|\mathbf{y}, \mathbf{z}) \propto \exp\left\{ -\frac{1}{2} \sum_{i:z_i=2} (y_i - \theta)^2 \right\} \exp\left(-\frac{\theta^2}{2} \right) \equiv N\left(\frac{\sum_{i:z_i=2} y_i}{1 + n_2}, \frac{1}{1 + n_2} \right),$$

where n_2 is the number of observations for which $z_i = 2$. For each i,

$$P(z_i = 2|\theta, \mathbf{y}) = \frac{e^{-(y_i-\theta)^2/2}}{e^{-(y_i-\theta)^2/2} + e^{-y_i^2/2}}, \quad P(z_i = 1|\theta, \mathbf{y}) = \frac{e^{-y_i^2/2}}{e^{-(y_i-\theta)^2/2} + e^{-y_i^2/2}}.$$

Hence, it is easy to implement a Gibbs sampler to simulate the posterior distribution of $(\theta, z_1, \ldots, z_n)$.

Genetic Linkage Example

This example concerns genetic linkage of 197 animals. The animals are distributed into 4 categories:

$$\mathbf{y} = (y_1, y_2, y_3, y_4) = (125, 18, 20, 34)$$

with cell probabilities

$$\left(\frac{2+\theta}{4}, \frac{1}{4}(1-\theta), \frac{1}{4}(1-\theta), \frac{\theta}{4}\right), \quad 0 \leq \theta \leq 1.$$

Prior of θ: Uniform$(0, 1)$.

Posterior of θ :

$$\pi(\theta|\mathbf{y}) \propto f(\mathbf{y}|\theta)\pi(\theta) \propto (2+\theta)^{y_1}(1-\theta)^{y_2+y_3}\theta^{y_4} I[\theta \in (0, 1)].$$

Though it is possible to sample the posterior of θ directly (e.g. via rejection sampling), data augmentation brings about a substantial simplification.

Data Augmentation

Augment the observed data (y_1, y_2, y_3, y_4) by dividing the first cell into two portions, with respective probabilities proportional to θ and 2:

$$z|\mathbf{y}, \theta \sim \text{Binomial}\left(y_1, \frac{\theta}{2+\theta}\right).$$

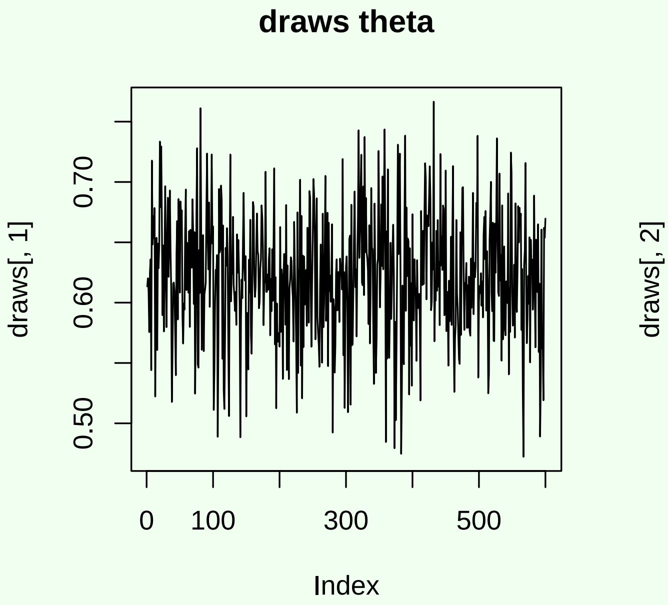
This gives the augmented data set $(\mathbf{y}, z)$, for which we have that

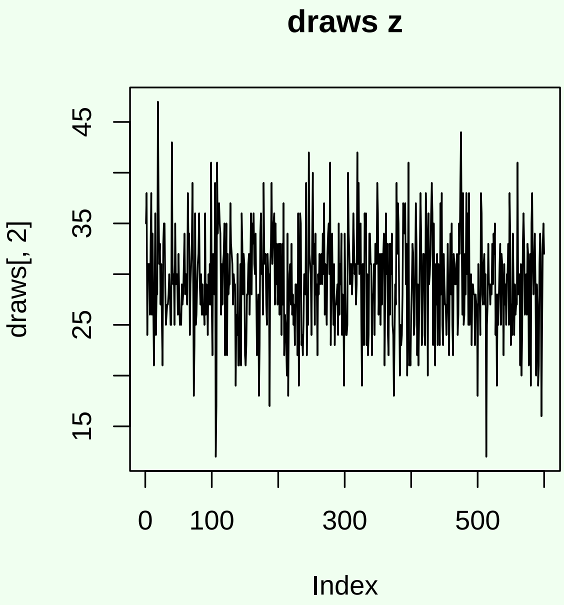
$$\begin{aligned} f(\mathbf{y}, z|\theta) &= f(\mathbf{y}|\theta)\pi(z|\mathbf{y}, \theta) \\ &\propto (2+\theta)^{y_1}(1-\theta)^{y_2+y_3}\theta^{y_4}\binom{y_1}{z}\left(\frac{\theta}{2+\theta}\right)^z\left(\frac{2}{2+\theta}\right)^{y_1-z} \end{aligned}$$

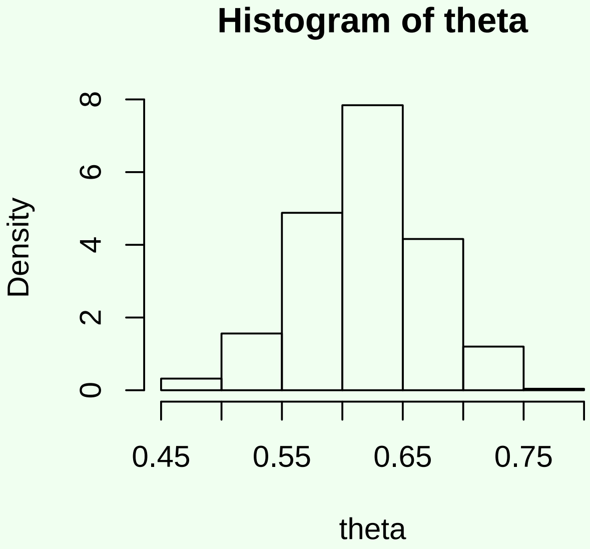
Now it is immediate that

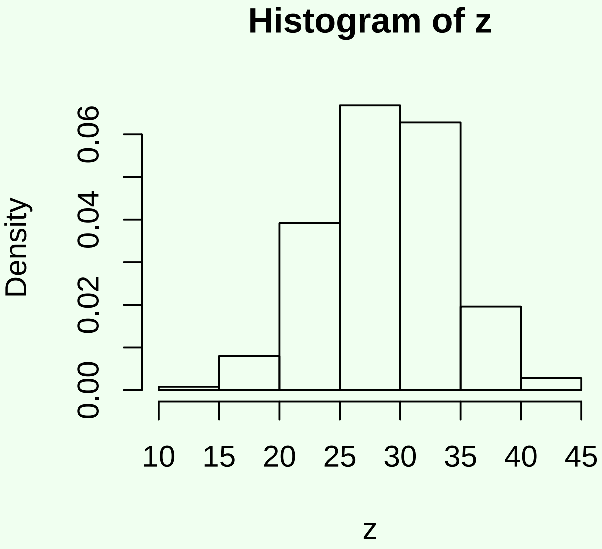
$$\pi(\theta|\mathbf{y}, z) \propto \theta^{z+y_4}(1-\theta)^{y_2+y_3}I[\theta \in (0,1)] \equiv Beta(z+y_4+1, y_2+y_3+1).$$

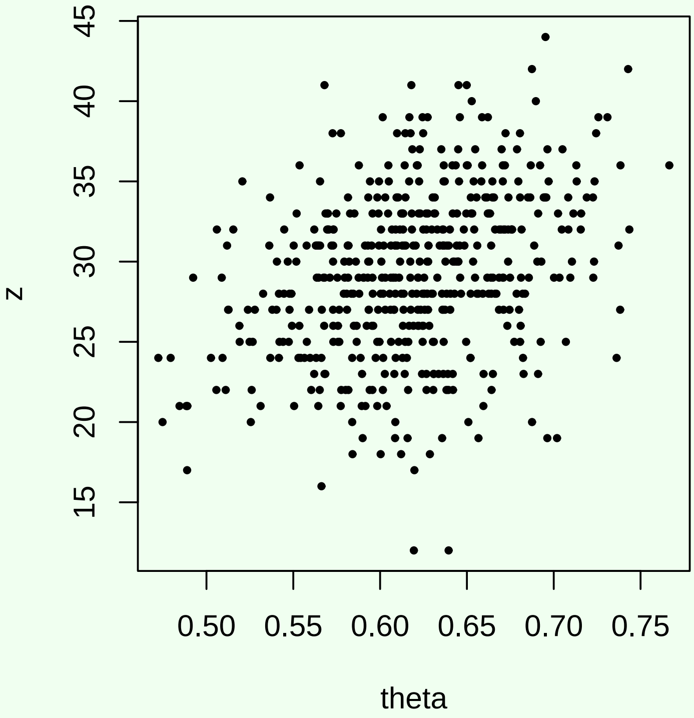
To complete the Gibbs sampler we also need to generate draws from the conditional posterior distribution of z. I have implemented this Gibbs sampler in R, with `ndraw=600` iterations and no burn-in.


draws theta
draws[, 1]
0.50
0.60
0.70
0
100
300
500
Index


draws z
draws[, 2]
15
25
35
45
0
100
300
500
Index


Histogram of theta
Density
0
2
4
6
8
0.45
0.55
0.65
0.75
theta


Histogram of z
Density
0.00
0.02
0.04
0.06
10
15
20
25
30
35
40
45
z


z
theta
15
20
25
30
35
40
45
0.50
0.55
0.60
0.65
0.70
0.75

Remark

It seems that convergence of the algorithm is very fast.

However, interpretation of the results does need care, since the sequence of realisations are not independent. There are two alternative strategies:

1. Run a single chain for sufficiently long, so that dependence between successive realisations does not diminish the precision of the sample information; and

2. Run several chains, and average across them.

I'm going to stick with the first of these approaches, though it should also be recognised that approaches to assessing convergence of such chains are often based on comparisons of within–chain and across–chain variability.

The general MCMC algorithm

The Gibbs sampler is the best publicised MCMC algorithm, but there are many others. It is not possible to use Gibbs sampling in all problems since sometimes the conditional posterior distributions can not be calculated.

Example 4.1. Suppose $Y_i|\mu, \omega \sim \text{Cauchy}(\mu, 1/\omega)$ independently for $i = 1, \ldots, n$. Hence, the sampling density is

$$f(\mathbf{y}|\mu, \omega) = \prod_{i=1}^{n} f(y_i|\mu, \omega) = \prod_{i=1}^{n} \frac{\omega^{1/2}}{\pi} \frac{1}{1 + \omega(y_i - \mu)^2}$$

Suppose also that $\mu \sim N(\mu_0, 1/\kappa_0)$ and $\omega \sim \text{Gamma}\,(\alpha_0, \lambda_0)$, where μ and ω are considered to be a priori independent, and μ_0, κ_0, α_0 and λ_0 are considered to be known hyperparameters. Then,

$$\pi(\mu, \omega|\mathbf{y}) \propto \left\{ \prod_{i=1}^{n} \frac{1}{1 + \omega(y_i - \mu)^2} \right\} e^{-\frac{\kappa_0}{2}(\mu-\mu_0)^2} \omega^{\frac{n}{2}+\alpha_0-1} e^{-\lambda_0\omega} I[\omega > 0].$$

Clearly, the posterior is a complex two-dimensional distribution.

Conditional posterior densities:

$$\pi(\mu|\omega, \mathbf{y}) \propto \left\{ \prod_{i=1}^{n} \frac{1}{1+\omega(y_i-\mu)^2} \right\} e^{-\frac{\kappa_0}{2}(\mu-\mu_0)^2},$$

and

$$\pi(\omega|\mu, \mathbf{y}) \propto \left\{ \prod_{i=1}^{n} \frac{1}{1+\omega(y_i-\mu)^2} \right\} \omega^{\frac{n}{2}+\alpha_0-1} e^{-\lambda_0\omega}\, I[\omega > 0].$$

None of these distributions has a well-known form, so Gibbs sampling is precluded. In order to compute the posterior distribution in this model, we need to use more general MCMC algorithms.

This example illustrates the need for MCMC algorithms that are more general than Gibbs sampling.

General MCMC algorithm

1. Break the components of θ into d groups $\theta_1, \ldots, \theta_d$, where each group θ_j has dimension ≥ 1.
2. Initialize with $\theta_1^{(0)}, \ldots, \theta_d^{(0)}$.
3. Update $\theta_1^{(0)}$ to $\theta_1^{(1)}$ *"according to"* the conditional distribution $\pi(\theta_1|\theta_2^{(0)}, \theta_3^{(0)} \ldots, \theta_d^{(0)})$.
4. Update $\theta_2^{(0)}$ to $\theta_2^{(1)}$ *"according to"* the conditional distribution $\pi(\theta_2|\theta_1^{(1)}, \theta_3^{(0)}, \ldots, \theta_d^{(0)})$.
5. $\ldots$
6. Update $\theta_d^{(0)}$ to $\theta_d^{(1)}$ *"according to"* the conditional distribution $\pi(\theta_d|\theta_1^{(1)}, \theta_2^{(1)}, \ldots, \theta_{d-1}^{(1)})$.
7. Iterate this updating procedure.

Remarks

Running any general MCMC algorithm, after discarding an initial number of draws (the burn-in), the remaining draws can be regarded as a sample from the target distribution $\pi(\theta)$ (under mild regularity conditions, see Appendix 1 in lecture notes).

In the updating steps of the general MCMC we *"update according to"* the appropriate conditional distribution. For Gibbs sampling, this means *simulating from* the conditional distribution, but for other MCMC algorithms *"update according to"* means something else.

The most general algorithm in this context is the Metropolis–Hastings algorithm.

The Metropolis-Hastings algorithm

In a general MCMC algorithm, suppose that the current value of the chain is $\theta_1^{(j)}, \ldots, \theta_d^{(j)}$ and that we now want to simulate $\theta_1^{(j+1)}$.

- Propose a candidate value θ_1^{can}, which is a draw from an arbitrary distribution with density $q(\theta_1^{can}|\theta_1^{(j)}, \theta_2^{(j)}, \ldots, \theta_d^{(j)})$.

- Take as the next value of θ_1 in the chain

$$\theta_1^{(j+1)} = \begin{cases} \theta_1^{can} & \text{with probability } p \\ \theta_1^{(j)} & \text{with probability } 1-p \end{cases}$$

where

$$p = \min\left\{1,\ \frac{\pi(\theta_1^{can}|\theta_2^{(j)}, \ldots, \theta_d^{(j)})}{\pi(\theta_1^{(j)}|\theta_2^{(j)}, \ldots, \theta_d^{(j)})}\ \frac{q(\theta_1^{(j)}|\theta_1^{can}, \theta_2^{(j)}, \ldots, \theta_d^{(j)})}{q(\theta_1^{can}|\theta_1^{(j)}, \theta_2^{(j)}, \ldots, \theta_d^{(j)})}\right\},$$

with $\pi(\theta_1^{can}|\theta_2^{(j)}, \ldots, \theta_d^{(j)})$ denoting the density corresponding to the conditional posterior of θ_1 at $\theta_1 = \theta_1^{can}$ and similarly for $\pi(\theta_1^{(j)}|\theta_2^{(j)}, \ldots, \theta_d^{(j)})$.

Some comments

- To implement the second part of MH: draw a value u from a Uniform$(0,1)$ distribution, and take $\theta_1^{(j+1)} = \theta_1^{can}$ if $u < p$ and $\theta_1^{(j+1)} = \theta_1^{(j)}$ otherwise.

- The *candidate generator* $q(\theta_1^{can}|\theta_1^{(j)}, \theta_2^{(j)}, \ldots, \theta_d^{(j)})$ is arbitrary so, in principle, any choice should work. In practice, the choice matters for *mixing* properties of the algorithm. Note that the candidate generator can depend on the *current* value of the chain, although this is not a requirement.

- MH algorithm has the major advantage over Gibbs that it is not necessary to know all the conditional posterior distributions. We only need to know the conditionals up to proportionality.

- Gibbs sampling is a special case of the MH where the candidate generator is $q(\theta_1^{can}|\theta_1^{(j)}, \theta_2^{(j)}, \ldots, \theta_d^{(j)}) = \pi(\theta_1^{can}|\theta_2^{(j)}, \ldots, \theta_d^{(j)})$. In this case, $p = 1$.

- *Random walk Metropolis algorithm with Normal increments*:

$q(\theta_1^{can}|\theta_1^{(j)}, \theta_2^{(j)}, \ldots, \theta_d^{(j)})$ is the density of a $\mathrm{N}(\theta_1^{(j)}, v)$ distribution. The symmetry of the candidate generator means that the terms involving $q(\cdot)$ cancel in the formula for the acceptance probability:

$$p = \min\left\{1, \ \frac{\pi(\theta_1^{can}|\theta_2^{(j)}, \ldots, \theta_d^{(j)})}{\pi(\theta_1^{(j)}|\theta_2^{(j)}, \ldots, \theta_d^{(j)})}\right\}.$$

The variance of the candidate generator v plays an important role in the mixing properties of the algorithm:
If v too large $\Longrightarrow$ moves proposed too bold $\Longrightarrow$ low acceptance probabilities $\Longrightarrow$ slow mixing
If v too small $\Longrightarrow$ acceptance probability very high but moves in little steps $\Longrightarrow$ slow mixing
v is typically chosen by trial and error, aiming at an acceptance probability roughly around 30%.

- To obtain $\theta_2^{(j+1)}$ propose a candidate θ_2^{can} from an arbitrary distribution $q(\theta_2^{can}|\theta_1^{(j+1)}, \theta_2^{(j)}, \ldots, \theta_d^{(j)})$ and accept this candidate as $\theta_2^{(j+1)}$ with probability

$$p = \min\left\{1,\ \frac{\pi(\theta_2^{can}|\theta_1^{(j+1)}, \theta_3^{(j)}, \ldots, \theta_d^{(j)})}{\pi(\theta_2^{(j)}|\theta_1^{(j+1)}, \theta_3^{(j)}, \ldots, \theta_d^{(j)})}\ \frac{q(\theta_2^{(j)}|\theta_1^{(j+1)}, \theta_2^{can}, \ldots, \theta_d^{(j)})}{q(\theta_2^{can}|\theta_1^{(j+1)}, \theta_2^{(j)}, \ldots, \theta_d^{(j)})}\right\}.$$

If θ_2^{can} is rejected, then $\theta_2^{(j+1)} = \theta_2^{(j)}$.

- The most commonly used MCMC algorithms in practice consist of *hybrid chains*. We use the general MCMC algorithm, where the various simulation steps are conducted using Gibbs sampling where possible, and MH when Gibbs is not possible.

The genetic linkage example revisited

The target distribution is

$$\pi(\theta|\mathbf{y}) \propto (2+\theta)^{y_1}(1-\theta)^{y_2+y_3}\theta^{y_4} I[0 \le \theta \le 1].$$

As candidate generator, we can take a Uniform$[0,1]$ distribution. Thus, given that the chain is currently at a certain value θ, we propose θ^{can} ∼Uniform$[0,1]$, and the acceptance probability p is

$$\begin{aligned} p &= \min\left\{1,\ \frac{\pi(\theta^{can}|\mathbf{y})}{\pi(\theta|\mathbf{y})}\right\} \\ &= \min\left\{1,\ \left(\frac{2+\theta^{can}}{2+\theta}\right)^{y_1}\left(\frac{1-\theta^{can}}{1-\theta}\right)^{y_2+y_3}\left(\frac{\theta^{can}}{\theta}\right)^{y_4}\right\}. \end{aligned}$$

The MH algorithm

1. Start the chain at some value $\theta^{(0)}$
2. Propose a *candidate* value θ^{can} $\sim$Uniform$[0,1]$. Take as the new value of the chain

$$\theta^{(1)} = \begin{cases} \theta^{can} & \text{with probability } p \\ \theta^{(0)} & \text{with probability } 1-p \end{cases}$$

   where

$$p = \min\left\{1,\ \left(\frac{2+\theta^{can}}{2+\theta^{(0)}}\right)^{y_1} \left(\frac{1-\theta^{can}}{1-\theta^{(0)}}\right)^{y_2+y_3} \left(\frac{\theta^{can}}{\theta^{(0)}}\right)^{y_4}\right\}$$

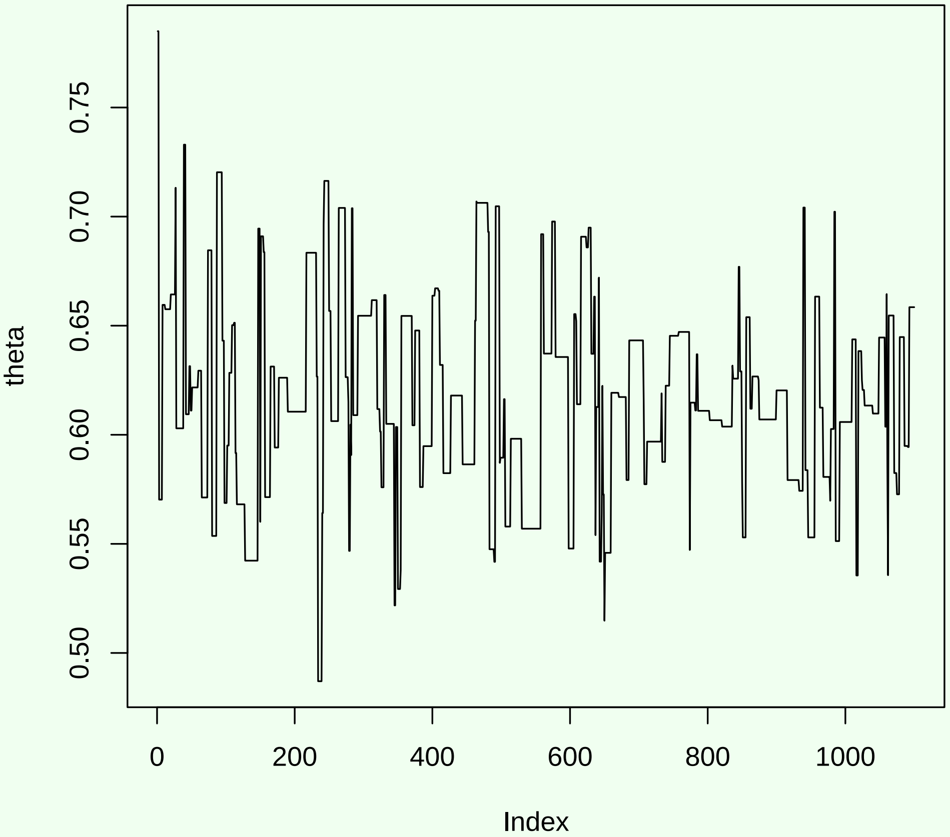
   (the latter is carried out by sampling u $\sim$Uniform$(0,1)$, and taking $\theta^{(1)} = \theta^{can}$ if and only if $u < p$).
3. Iterate this procedure

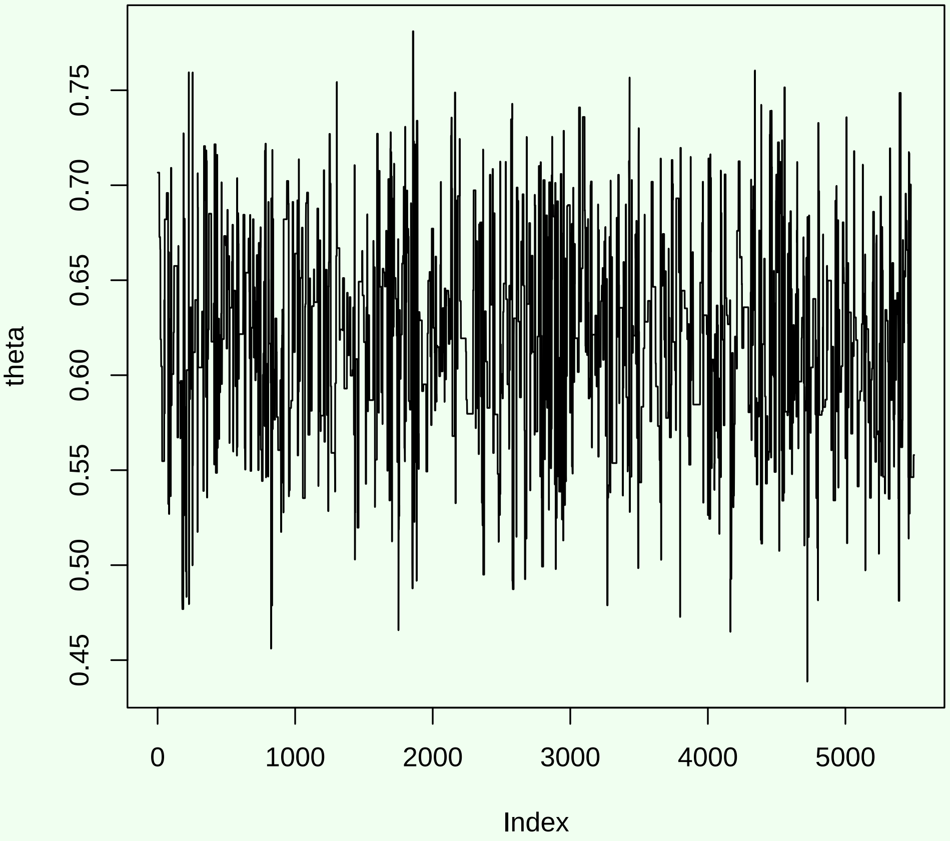
Convergence of the algorithm

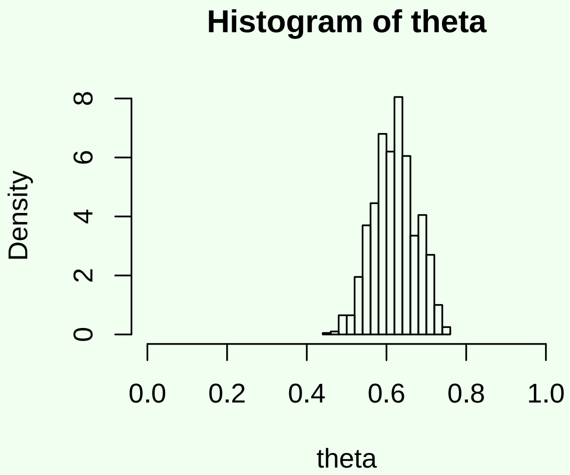
As with the Gibbs sampler, it is necessary to monitor the output to ensure convergence.

Running the MH algorithm in R with 1,100 iterations and no burn-in I obtained acceptance rate of 16%, a bit low for good mixing. Hence, I decided to run a longer chain of 5,500 iterations. Convergence seemed ok after the first few iterations, so I deleted the first 500 draws. In addition, to counteract for the slow mixing, I kept only every 5th draw.

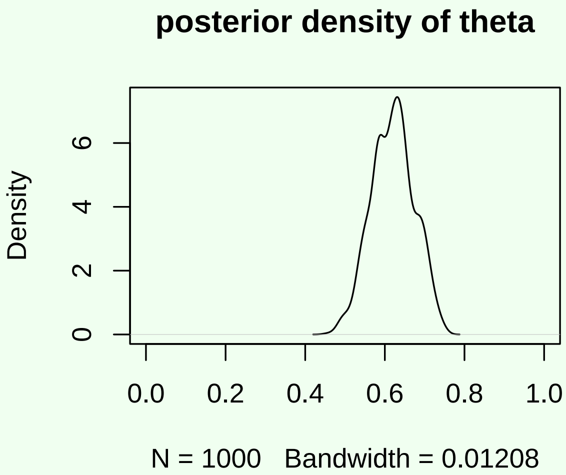
```
> theta <- mh(data=c(125,18,20,34),ndraw=1100)
[1] "percentage accepted draws:" "15.6363636363636"
> plot(theta,type="l")
> theta <- mh(data=c(125,18,20,34),ndraw=5500)
> theta <- theta[-c(1:500)]
> theta <- theta[c(1:1000)*5]
```


theta
0.50
0.55
0.60
0.65
0.70
0.75
0
200
400
600
800
1000
Index


theta
0.45
0.50
0.55
0.60
0.65
0.70
0.75
0
1000
2000
3000
4000
5000
Index



Histogram of theta



posterior density of theta

Example 4.1 revisited: Cauchy distribution

Conditional posterior densities:

$$\pi(\mu|\omega, \mathbf{y}) \propto \left\{ \prod_{i=1}^{n} \frac{1}{1+\omega(y_i-\mu)^2} \right\} e^{-\frac{\kappa_0}{2}(\mu-\mu_0)^2},$$

$$\pi(\omega|\mu, \mathbf{y}) \propto \left\{ \prod_{i=1}^{n} \frac{1}{1+\omega(y_i-\mu)^2} \right\} \omega^{\frac{n}{2}+\alpha_0-1} e^{-\lambda_0\omega}\, I[\omega > 0].$$

Both for drawing μ and for drawing ω we are going to use Random Walk Metropolis with Normal increments.

MH algorithm

1. Choose initial values $(\mu^{(0)}, \omega^{(0)})$.
2. Given that the chain is currently at $(\mu^{(j)}, \omega^{(j)})$:
   - Draw $\mu^{can} \sim N(\mu^{(j)}, v_\mu)$ and take

$$\mu^{(j+1)} = \begin{cases} \mu^{can} & \text{with probability } p \\ \mu^{(j)} & \text{with probability } 1-p \end{cases}$$

   where

$$\begin{aligned} p &= \min\left(1, \frac{\pi(\mu^{can} \mid \omega^{(j)}, \mathbf{y})}{\pi(\mu^{(j)} \mid \omega^{(j)}, \mathbf{y})} \frac{q(\mu^{(j)} \mid \mu^{can})}{q(\mu^{can} \mid \mu^{(j)})}\right) \\ &= \min\left(1, e^{\frac{\kappa_0}{2}\left\{(\mu^{(j)}-\mu_0)^2-(\mu^{can}-\mu_0)^2\right\}} \prod_{i=1}^{n} \left\{\frac{1+\omega^{(j)}(y_i-\mu^{(j)})^2}{1+\omega^{(j)}(y_i-\mu^{can})^2}\right\}\right) \end{aligned}$$

(this is implemented by drawing $u \sim$Uniform$(0,1)$ and taking $\mu^{(j+1)} = \mu^{can}$ if and only if $u < p$).

2. • Draw $\omega^{can} \sim N(\omega^{(j)}, v_\omega)$ and take

$$\omega^{(j+1)} = \begin{cases} \omega^{can} & \text{with probability } p \\ \omega^{(j)} & \text{with probability } 1-p \end{cases}$$

   where

$$p = \min\left(1, \left(\frac{\omega^{can}}{\omega^{(j)}}\right)^{\frac{n}{2}+\alpha_0-1} e^{\lambda_0(\omega^{(j)}-\omega^{can})} \prod_{i=1}^{n} \left\{\frac{1+\omega^{(j)}(y_i-\mu^{(j+1)})^2}{1+\omega^{can}(y_i-\mu^{(j+1)})^2}\right\}\right)$$

   (this is implemented by drawing $u \sim$Uniform$(0,1)$ and taking $\omega^{(j+1)} = \omega^{can}$ if and only if $u < p$).

3. Iterate Step 2 a large number of times. Discard an initial number of draws (burn-in) and base inference on subsequent draws.

Important remark: The acceptance probability p for ω^{can} in *Step 2* can only be positive if the drawn value $\omega^{can} > 0$. If we draw a value $\omega^{can} < 0$, then $p = 0$ and $\omega^{(j+1)} = \omega^{(j)}$.

Further recommendations about MCMC

- Assessing convergence of the algorithm is extremely important, but can be problematic in high dimensional situations.

Always run the chain several times using different starting values and check that the output from the various chains is very similar.

Run long chains and have long burn-in periods.

- Be extremely cautious if you use improper prior distributions.

If you use improper priors always check that the joint posterior distribution is proper, otherwise you can not have any faith in the results obtained via computation.

This problem does not arise if you use proper prior distributions.

MCMC in GLMs

We will discuss various approaches to implementing MCMC algorithms in GLMs. In particular, we will consider logistic and probit regression models for binary data and log-linear Poisson regression models for count data.

These models have in common that there are no conjugate priors available for regression parameters and, therefore, conditional posterior distributions have non-standard forms. We will review different algorithms proposed in the literature and discuss their implementation. Those will mostly involve MH proposals.

We will first discuss models which are essentially Bayesian versions of GLMs. Here we can perform a comparison with the ML approach.

We will then extend this class of models by introducing additional random effects (generalised linear mixed models).

Logistic regression

Response variable: the occurrence or non-occurrence of infection following birth by Caesarian section.

Covariates:
$x_1 = 1$ if Caesarian section was not planned and $x_1 = 0$ otherwise;
$x_2 = 1$ if there is presence of risk factor(s) and $x_2 = 0$ otherwise;
$x_3 = 1$ if antibiotics were given as prophylaxis and $x_3 = 0$ otherwise.

There is a total of 251 births:

Covariates			Infection	
x_1	x_2	x_3	yes	no
0	0	0	8	32
0	0	1	0	2
0	1	0	28	30
0	1	1	1	17
1	0	0	0	9
1	0	1	0	0
1	1	0	23	3
1	1	1	11	87

We have aggregated all binary responses in each covariate category obtaining binomial responses. Thus,

$$y_i \sim \text{Binomial}(n_i, p_i)$$

where i is an index for each covariate combination, y_i is the number of infections, n_i the total number of observations, and p_i the probability of infection for each individual with this covariate combination.

Suppose we assume the binary logistic regression model,

$$\text{logit } p_i = \log\left(\frac{p_i}{1-p_i}\right) = \eta_i = x_i^T\beta$$

where $x_i^T = (1, x_{1i}, x_{2i}, x_{3i})$ denotes the vector of covariates. Then,

$$\frac{p_i}{1-p_i} = \exp(x_i^T\beta) \Rightarrow p_i = \frac{\exp(x_i^T\beta)}{1+\exp(x_i^T\beta)}.$$

Therefore, the likelihood is given by

$$\begin{aligned}
f(y|\beta) &= \prod_{i=1}^{n} f(y_i|\beta) \propto \prod_{i=1}^{n} p_i^{y_i}(1-p_i)^{n_i-y_i} \\
&= \prod_{i=1}^{n} \left[\frac{\exp({x_i}^T\beta \cdot y_i)}{1+\exp({x_i}^T\beta)}\right]^{y_i} \left[\frac{1}{1+\exp({x_i}^T\beta)}\right]^{n_i-y_i} \\
&= \prod_{i=1}^{n} \frac{\exp({x_i}^T\beta \cdot y_i)}{[1+\exp({x_i}^T\beta)]^{n_i}}.
\end{aligned}$$

A Maximum likelihood analysis in R gives the following results.

```
Deviance Residuals:
[1]   1.21563  -0.15231  -0.78520   0.26470  -2.56229   0.0000
[8]  -0.07162
```

```
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.8926     0.4124  -4.590 4.44e-06 ***
noplan        1.0720     0.4253   2.520   0.0117 *
factor        2.0299     0.4552   4.459 8.23e-06 ***
antib        -3.2544     0.4813  -6.762 1.37e-11 ***
---
Sig. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

    Null deviance: 83.491  on 6  degrees of freedom
Residual deviance: 10.997  on 3  degrees of freedom
AIC: 36.178

Number of Fisher Scoring iterations: 4
```

The 10.997 value for the residual deviance is not obviously inconsistent with a χ^2_3 distribution, since we can compute $P(\chi^2_3 > 10.997)$ as

```
> 1-pchisq(10.997,3)
[1] 0.01174211
```

so there is no strong warning sign that the model may not fit adequately.

Bayesian analysis

A Bayesian analysis places a prior on $\beta = (\beta_0, \beta_1, \beta_2, \beta_3)^T$, usually a multivariate normal prior $\beta \sim N_4(\mu_0, C_0)$. The posterior distribution for β is

$$\begin{aligned}\pi(\beta|y) &\propto f(y|\beta)\pi(\beta) \\ &\propto \left\{\prod_{i=1}^{n} \frac{\exp({x_i}^T\beta \cdot y_i)}{(1+\exp({x_i}^T\beta))^{n_i}}\right\} \exp\left(-\frac{(\beta-\mu_0)^T C_0^{-1}(\beta-\mu_0)}{2}\right).\end{aligned}$$

The posterior density is a complicated, non-linear function of β. How can we construct an MCMC algorithm to sample from this distribution?

1. **Update β component-wise using Gibbs**: Generate draws from the conditional posterior of each component of β. **Problems:** • Tedious to sample from conditionals (non-standard); • slow convergence due to strong correlations between parameters.

2. **Update β component-wise via MH**: Simple random walk proposal.
   Problems: • Needs tuning; • problems with collinearity.

3. **Update β jointly**: Multivariate Normal candidate generator with mean equal to posterior mode and covariance matrix equal to inverse curvature at mode.
   Problems: • Newton-Raphson for $\pi(\beta|y)$; • relies on asymptotic normality of posterior distribution; • may rarely propose values in the tails of the posterior.

4. **Update β jointly**: Multivariate Gaussian random walk candidate generator, with covariance matrix equal to the inverse curvature at posterior mode times a (scalar) tuning factor, or some other pre-estimated covariance matrix
   Problems: • Needs tuning and pre-calculation of covariance estimate.

The code to implement algorithms 2, 3 and 4 is stored in the file "logistic.r". The output from running these algorithms is plotted in the files "logistic2.pdf", "logistic3.pdf" and "logistic4.pdf".

Estimates (posterior mean, standard deviation and posterior probability of exceeding 0) using the third algorithm (5,000 samples after burn-in of 500) are given in the following table:

```
Coefficients:     mean    Std probability
(Intercept)    -1.9544 0.4228           0
noplan          1.1071 0.4229      0.9968
factor          2.0955  0.467           1
antib          -3.3322 0.4867           0
```

The overall acceptance rate of this run was 87.6%.

Generally, we prefer an algorithm which is automatic (no tuning required) and updates β in one block, (to avoid strong autocorrelations in the samples).

Weighted Least Squares

The ML estimator in a GLM and its asymptotic covariance matrix are obtained by iterative use of WLS on transformed observations. The algorithm takes a simple form in the case of canonical link functions $g(\mu_i) = \eta_i = x_i^T \beta$, where $\mu_i = E(y_i)$, such as the logit link for binary regression.

For logistic regression with binomial response, the canonical link is

$$g(\mu_i) = \log\left(\frac{p_i}{1-p_i}\right) = \log\left(\frac{n_i p_i}{n_i - n_i p_i}\right) = \log\left(\frac{\mu_i}{n_i - \mu_i}\right).$$

Define a vector of transformed observations $\tilde{y}(\beta)$ and a diagonal matrix of weights $W(\beta)$ as

$$\begin{aligned} \tilde{y}_i(\beta) &= \eta_i + (y_i - \mu_i) g'(\mu_i) \\ W_i(\beta) &= 1/g'(\mu_i), \end{aligned}$$

where $g'(\mu_i)$ is the first derivative of the link function. For the canonical link for logistic regression $g'(\mu_i) = \frac{n_i}{\mu_i(n_i - \mu_i)}$.

Iterative Weighted Least Squares

The Iterative Weighted Least Squares (IWLS) algorithm starts with some arbitrary value $\beta^{(0)}$, and iteratively obtains $\beta^{(t)}$, $t = 1, \ldots$ as the least squares estimator in the general (with weights) linear model

$$\tilde{y}(\beta^{(t-1)}) \sim N(X\beta, W^{-1}(\beta^{(t-1)}))$$

i.e.

$$\beta^{(t)} = (X^T W(\beta^{(t-1)})X)^{-1} X^T W(\beta^{(t-1)})\tilde{y}(\beta^{(t-1)}).$$

After convergence, the final estimate $\hat{\beta}$ is the ML estimate and the matrix $(X^T W(\hat{\beta})X)^{-1}$ is the associated asymptotic covariance matrix.

Bayesian context

The latter estimates also have a Bayesian interpretation: they can be shown to correspond to the mode and inverse curvature matrix at the mode of the posterior $\pi(\beta|y)$ under a flat prior on β, i.e. under $\pi(\beta) \propto 1$.

With a proper normal prior on β, $\beta \sim N(\mu_0, C_0)$, the IWLS algorithm can be modified in order to find the mode and curvature at the mode of $\pi(\beta|y)$.

We will combine

$$\tilde{y}(\beta^{(t-1)}) \sim N(X\beta, W^{-1}(\beta^{(t-1)})),$$

i.e.

$$f(\tilde{y}(\beta^{(t-1)}) \mid \beta) \propto \exp\left\{-\frac{1}{2}\left(\tilde{y}(\beta^{(t-1)}) - X\beta\right)^T W\left(\beta^{(t-1)}\right)\left(\tilde{y}(\beta^{(t-1)}) - X\beta\right)\right\}$$

with the prior:

$$\pi(\beta) \propto \exp\left\{-\frac{1}{2}(\beta - \mu_0)^T C_0^{-1}(\beta - \mu_0)\right\}.$$

Posterior: $\pi(\beta \mid \tilde{y}(\beta^{(t-1)}))$

$$\begin{aligned}
&\propto \; \pi(\beta) f(\tilde{y}(\beta^{(t-1)}) \mid \beta) \\
&\propto \; \exp\left\{-\frac{1}{2}(\beta - \mu_0)^T C_0^{-1}(\beta - \mu_0)\right\} \\
&\times \; \exp\left\{-\frac{1}{2}\left(\tilde{y}(\beta^{(t-1)}) - X\beta\right)^T W\left(\beta^{(t-1)}\right)\left(\tilde{y}(\beta^{(t-1)}) - X\beta\right)\right\} \\
&\propto \; \exp\left\{-\frac{1}{2}[\beta^T C_0^{-1}\beta - 2\beta^T C_0^{-1}\mu_0]\right\} \\
&\times \; \exp\left\{-\frac{1}{2}[\beta^T X^T W\left(\beta^{(t-1)}\right) X\beta - 2\beta^T X^T W\left(\beta^{(t-1)}\right)\tilde{y}(\beta^{(t-1)})]\right\}
\end{aligned}$$

$$= \exp[-\frac{1}{2}\beta^T(C_0^{-1} + X^T W\left(\beta^{(t-1)}\right) X)\beta + \beta^T(C_0^{-1}\mu_0 + X^T W\left(\beta^{(t-1)}\right)\tilde{y}(\beta^{(t-1)}))],$$

that is

$$\beta|\tilde{y}(\beta^{(t-1)}) \sim N(\beta^{(t)}, (C_0^{-1} + X^T W(\beta^{(t-1)})X)^{-1}).$$

To find the posterior mode we iterate until convergence.

Gamerman's MH IWLS algorithm

The idea of Gamerman's IWLS proposal is to perform only one step of the iteration, starting at the current value β, and use the resulting multivariate Gaussian as the proposal distribution in a MH setting:

$$q(\beta^{can}|\beta, y) \sim N(f(\beta), (C_0^{-1} + X^T W(\beta) X)^{-1}),$$

where

$$f(\beta) = (C_0^{-1} + X^T W(\beta) X)^{-1} (C_0^{-1} \mu_0 + X^T W(\beta) \tilde{y}(\beta)).$$

The resulting MCMC algorithm has several advantages:
- It does not require any tuning constants;
- The proposal obtained is reasonably close to the posterior, and therefore the algorithm leads to high acceptance probabilities;
- It avoids problems with independence proposals at the posterior mode;
- A similar approach can also be used in GLMMs.

Acceptance probability:

$$p = \min\left\{1, \frac{\pi(\beta^{can}|y)}{\pi(\beta|y)} \frac{q(\beta|\beta^{can}, y)}{q(\beta^{can}|\beta, y)}\right\}$$

Note that, to evaluate the proposal ratio, we also have to perform the inverse IWLS step, starting at the proposed value β^{can}.

The results using Gamerman's algorithm (in the file "logistic.r") are as follows (using 5,000 samples after burn-in of 500):

```
Coefficients:     mean     Std  probability
(Intercept)    -1.9717  0.4328            0
noplan           1.092  0.4206        0.995
factor          2.1148  0.4823            1
antib          -3.3148  0.4922            0
```

See also the file "logisticgamer.pdf".

In general, there is good agreement with the results obtained earlier. The overall acceptance rate of this run was 74.1%.

Data augmentation in binary probit regression

Consider the Bayesian probit regression model,

$$\begin{aligned} y_i &\sim \text{Bernoulli}\,\big(\Phi(\eta_i)\big) \\ \eta_i &= x_i^T\beta \\ \beta &\sim N(\mu_0, C_0) \end{aligned}$$

where $y_i \in \{0,1\}, i = 1,\ldots,n$ is a binary response variable for a collection of n objects with associated covariate measurement x_i,
$\Phi(\cdot)$ is the standard normal distribution function,
η_i is the linear predictor and β represents a $(p\times 1)$ column vector of regression coefficients.

In the probit model, the mean is given by $\mu_i = \Phi(\eta_i)$, hence, it corresponds to the probit link function: $g(\mu_i) = \Phi^{-1}(\mu_i)$.

Equivalent representation

Using auxiliary variables z_i (for $i = 1, \ldots, n$):

$$y_i = \begin{cases} 1 & \text{if } z_i > 0 \\ 0 & \text{otherwise} \end{cases} \qquad z_i \sim N(x_i^T \beta, 1),$$

where y_i is now deterministic conditional on the sign of z_i. Then,

$$P(y_i = 1) = P(z_i > 0) = P(N(x_i^T \beta, 1) > 0) = P(N(0,1) > -x_i^T \beta) = \Phi(x_i^T \beta)$$

This representation lends itself to efficient simulation using Gibbs.
The joint posterior of $(\beta, z_1, \ldots, z_n)$ is given by

$$\begin{aligned} \pi(\beta, z_1, \ldots, z_n | y) \;\propto\; & \prod_{i:y_i=1} \exp\Big[-\frac{1}{2}(z_i - {x_i}^T\beta)^2\Big] I[z_i > 0] \\ \times\; & \prod_{i:y_i=0} \exp\Big[-\frac{1}{2}(z_i - {x_i}^T\beta)^2\Big] I[z_i < 0] \\ \times\; & \exp\Big[-\frac{1}{2}(\beta - \mu_0)^T C_0^{-1}(\beta - \mu_0)\Big]. \end{aligned}$$

It is now immediate that the conditional posterior distribution of β is normal:

$$\beta|z, y \sim N\Big((C_0^{-1} + X^T X)^{-1}(C_0^{-1}\mu_0 + X^T z),\ (C_0^{-1} + X^T X)^{-1}\Big),$$

where $z = (z_1, \ldots, z_n)^T$, whereas the conditional posterior distribution for each z_i is *truncated normal*,

$$z_i|\beta, y \ \propto \ \begin{cases} N(x_i^T\beta, 1)\ I[z_i > 0] & \text{if } y_i = 1 \\ N(x_i^T\beta, 1)\ I[z_i \leq 0] & \text{otherwise,} \end{cases} \tag{1}$$

which is simple to sample from.

A slight disadvantage of the auxiliary variables approach, however, is that we can not aggregate to binomial responses (as we have done when considering logistic regression). Instead, we have to consider the individual $n = 251$ binary responses, which makes the algorithm slower.

The results obtained using the probit approach can be found in the following table (see file “probit.pdf” for additional details):

```
Coefficients:     mean    Std probability
(Intercept)     -1.115 0.2211           0
noplan          0.6092 0.2501      0.9954
factor          1.2204 0.2608           1
antib          -1.9115 0.2634           0
```

Those are quite different from the logistic model because of the different link. However, the relationship between the Probit and the Logit links is mainly a change in scale. So if we adjust our estimates of posterior mean and standard deviation by a factor of $\pi/\sqrt{3} \cdot 15/16 \approx 1.7$, the results look very similar:

```
Coefficients:     mean    Std probability
(Intercept)    -1.8955 0.3759           0
noplan          1.0356 0.4251      0.9954
factor          2.0747 0.4434           1
antib          -3.2496 0.4478           0
```

Poisson regression

In log-linear Poisson regression, it is assumed that

$$y_i \sim \text{Poisson}(\mu_i), \quad \text{where } g(\mu_i) = \log(\mu_i) = \eta_i = x_i^T \beta,$$

independently for $i = 1, \ldots, n$. We consider a normal prior: $\beta \sim N(\mu_0, C_0)$. For these models, there exists no data augmentation approach. However, we can implement methods similar to the ones discussed for logistic regression.

Often *offsets* are used in Poisson regression:

$$y_i \sim \text{Poisson}(\mu_i = e_i \lambda_i)$$

where e_i are known expected cases. Assume now we want to investigate the effect of covariates x_i in this model. Assuming $\lambda_i = \exp(x_i^T \beta)$, we obtain that

$$g(\mu_i) = \log(\mu_i) = \eta_i = \log(e_i) + x_i^T \beta.$$

Poisson regression

The posterior distribution of β is given by

$$\begin{aligned} f(\beta|y) &\propto \prod_{i=1}^{n} (e_i\lambda_i)^{y_i} \exp\{-e_i\lambda_i\} \times \exp\left\{-\frac{1}{2}(\beta-\mu_0)^T C_0^{-1}(\beta-\mu_0)\right\} \\ &\propto \exp\left\{\sum_{i=1}^{n}[y_i x_i^T\beta - e_i\exp(x_i^T\beta)]\right\} \\ &\times \exp\left\{-\frac{1}{2}(\beta-\mu_0)^T C_0^{-1}(\beta-\mu_0)\right\} \end{aligned}$$

Gamerman's MH IWLS algorithm can be used to update β. Note that $\log(e_i)$ will now appear as an *offset* and have to be incorporated in the IWLS proposal. The transformed variables and weghts are

$$\begin{aligned} \tilde{y}_i(\beta) &= \eta_i + (y_i - \mu_i)g'(\mu_i) - \log(e_i) \\ W_i(\beta) &= 1/g'(\mu_i), \end{aligned}$$

where $g'(\mu_i) = 1/\mu_i$.

IWLS MH Algorithm

IWLS proposal:

$$q(\beta^{can}|\beta, y) \sim N(f(\beta), (C_0^{-1} + X^T W(\beta) X)^{-1}),$$

where

$$f(\beta) = (C_0^{-1} + X^T W(\beta) X)^{-1} (C_0^{-1} \mu_0 + X^T W(\beta) \tilde{y}(\beta)).$$

Logistic regression with random effects

Models with random effects are often used to adjust for *overdispersion*. The most common model is to assume that the random effects are independent realizations from a Gaussian distribution with mean zero and unknown variance.

Example. Plate i $(i = 1, \ldots, n)$ contains n_i seeds of which y_i germinated. Relevant covariates in x_i are seed type (2 types), root extract (2 types) and an interaction term. A standard logistic regression model would assume

$$y_i \sim \text{Binomial}(n_i, p_i), \quad \text{where} \quad \text{logit}\, p_i = \log\left(\frac{p_i}{1 - p_i}\right) = x_i^T \beta.$$

If we want to adjust for overdispersion, we could extend this model by including additional random effects b_i $(i = 1, \ldots, n)$, as follows:

$$\text{logit}\, p_i = \eta_i = x_i^T \beta + b_i, \quad \text{where} \quad b_i \sim N(0, \omega^{-1})$$

are assumed to be independent. The precision ω is treated as unknown and is typically assigned a Gamma prior, say $\omega \sim \text{Gamma}(c, d)$.

The joint posterior distribution of the model parameters is given as

$$\pi(\beta, b, \omega|y) \propto f(y|\beta, b)\pi(\beta)\pi(b|\omega)\pi(\omega),$$

where $f(y|\beta, b)$ is the binomial likelihood and $\pi(b|\omega)$ is the normal random effects prior. As usual, we will take a Normal prior for β: $\beta \sim N(\mu_0, C_0)$. Therefore,

$$\begin{aligned} \pi(\beta, b, \omega|y) \;&\propto\; \left\{ \prod_{i=1}^{n} \frac{\exp({x_i}^T\beta \cdot y_i + b_i y_i)}{(1+\exp({x_i}^T\beta + b_i))^{n_i}} \exp\{-\frac{\omega}{2} b_i^2\} \right\} \omega^{n/2} \\ &\times\; \exp\left(-\frac{(\beta-\mu_0)^T C_0^{-1}(\beta-\mu_0)}{2}\right) \omega^{c-1} \exp\{-\omega d\}. \end{aligned}$$

How can we construct an efficient MCMC algorithm to sample from this distribution, preferably without any need for tuning? We will discuss two approaches, the first is based on Gamerman's IWLS proposals, the other one is based on a clever *reparametrization* of the model.

The full conditionals

$$
\begin{aligned}
\pi(\omega|\beta, b, y) &\propto \exp\{-\omega[d + \frac{1}{2}\sum_{i=1}^{n} b_i^2]\}\omega^{n/2+c-1} \\
&\equiv \text{Gamma}(n/2 + c, d + \frac{1}{2}\sum_{i=1}^{n} b_i^2)
\end{aligned}
$$

$$
\begin{aligned}
\pi(\beta|b, \omega, y) &\propto \left\{\prod_{i=1}^{n} \frac{\exp({x_i}^T\beta \cdot y_i + b_i y_i)}{(1+\exp({x_i}^T\beta + b_i))^{n_i}}\right\} \\
&\times \exp\left(-\frac{(\beta-\mu_0)^T C_0^{-1}(\beta-\mu_0)}{2}\right)
\end{aligned}
$$

$$
\pi(b_i|\beta, \omega, y) \propto \frac{\exp({x_i}^T\beta \cdot y_i + b_i y_i)}{(1+\exp({x_i}^T\beta + b_i))^{n_i}} \exp\{-\frac{\omega}{2} b_i^2\}
$$

An algorithm based on IWLS proposals

Note that, for updating β given b, the b_i's serve as an offset. Updating can be done similarly to how it was done in the model without random effects:

$$\begin{aligned}\tilde{y}_i(\beta) &= \eta_i + (y_i - \mu_i)g'(\mu_i) - b_i = x_i^T\beta + (y_i - \mu_i)g'(\mu_i)\\ W_i(\beta) &= 1/g'(\mu_i).\end{aligned}$$

IWLS proposal:

$$q(\beta^{can}|\beta, y) \sim N(f(\beta), (C_0^{-1} + X^T W(\beta) X)^{-1}),$$

where

$$f(\beta) = (C_0^{-1} + X^T W(\beta) X)^{-1}(C_0^{-1}\mu_0 + X^T W(\beta)\tilde{y}(\beta)).$$

Similarly, for updating b_i, the term $x_i^T\beta$ will serve as an offset:

$$\begin{aligned}\tilde{y}_i(b_i) &= \eta_i + (y_i - \mu_i)g'(\mu_i) - x_i^T\beta = b_i + (y_i - \mu_i)g'(\mu_i)\\ W_i(b_i) &= 1/g'(\mu_i).\end{aligned}$$

Note that only one likelihood term is relevant for each random effect. For updating b_i, $i = 1, \ldots, n$, the IWLS proposal distribution takes the form

$$b_i^{can} \sim N\left(\frac{W_i(b_i)\tilde{y}_i(b_i)}{\omega + W_i(b_i)}, \frac{1}{\omega + W_i(b_i)}\right).$$

Reparametrising the model

Instead of considering the parameters $b_1, \ldots, b_n$, where $b_i \sim N(0, \omega^{-1})$, we can consider $\eta_1, \ldots, \eta_n$, where $\eta_i | \beta \sim N(x_i^T \beta, \omega^{-1})$. With this parametrisation, the model can be written as

$$
\begin{aligned}
y_i &\sim \text{Binomial}(n_i, p_i), \quad \text{where logit } p_i = \eta_i \\
\eta_i &\sim N(x_i^T \beta, \omega^{-1}) \\
\beta &\sim N(\mu_0, C_0) \\
\omega &\sim \text{Gamma}(c, d)
\end{aligned}
$$

This procedure is termed *hierarchical centering* and has the advantage that the full conditional for β is now multivariate normal:

$$\beta | \eta \sim N((C_0^{-1} + \omega X^T X)^{-1}(C_0^{-1}\mu_0 + \omega X^T \eta), (C_0^{-1} + \omega X^T X)^{-1}).$$

The updating of η_i can be done along similar lines as that of b_i in the algorithm discussed earlier. If fact, the only two differences are that (a) we have to set the offset equal to zero and (b) we have to replace the prior mean (which was $\mu_i = 0$ for b_i) by $\mu_i = x_i{}^T\beta$. The proposal is then

$$\eta_i^{can} \sim N\left(\frac{\omega x_i^T\beta + W_i(\eta_i)\tilde{y}_i(\eta_i)}{\omega + W_i(\eta_i)}, \frac{1}{\omega + W_i(\eta_i)}\right),$$

where

$$\begin{aligned} \tilde{y}_i(\eta_i) &= \eta_i + (y_i - \mu_i)g'(\mu_i) \\ W_i(\eta_i) &= 1/g'(\mu_i). \end{aligned}$$

Finally, the full conditional for ω is Gamma with parameters $c + n/2$ and $d + \sum_i(\eta_i - x_i^T\beta)^2$.

Note: Both algorithms can be found in the file “logisticrandom.r”, whereas the files “crowderrand1.pdf” and “crowderrand2.pdf” display results from running them.