MAS8303 Modern Bayesian Inference
Part 2

M. Farrow
School of Mathematics and Statistics
Newcastle University

Semester 1, 2012-13

Chapter 4

Mixture Models

4.1 Mixtures

4.1.1 Finite mixtures as sampling distributions

In MAS3301 we looked at the use of mixture distributions as priors. We can also use mixtures as sampling distributions.

There are two reasons why we might want to do this:

1. We might believe that there really are two or more sub-populations and it makes sense to represent each by a component of the mixture. For example, the amount of a compound found in blood samples taken from animals might depend on whether or not the animal carries a particular infection. There are thus two sub-populations, one of infected animals and one of non-infected animals. We might not know which animals are infected but it might make sense to allow for these two sub-populations in a model. The distribution of the amount of the substance might be bimodal.

2. Even when there is no "physical" interpretation of the mixture components, using a mixture distribution allows more flexibility in the sampling model. We can relax the assumption that the data are "normally distributed", for example.

Consider a simple two-component mixture model. Our sampling model for observation Y_i has pdf

$$f(y_i;\ \pi, \theta_1, \theta_2) = \pi f_1(y_i; \theta_1) + (1-\pi) f_2(y_i; \theta_2).$$

Here $f_j(y; \theta_j)$ is the pdf for component j and depends on parameters θ_j. The component membership probabilities are π and $1-\pi$, with $0 \le \pi \le 1$.

Suppose that we have n independent (given the parameters) observations $y_1, \dots, y_n$. The likelihood is

$$L = \prod_{i=1}^{n} \left\{ \pi f_1(y_i; \theta_1) + (1-\pi) f_2(y_i; \theta_2) \right\}. \tag{4.1}$$

This has a rather complicated form. For example, it is a polynomial of degree n in π.

More generally we could have J components with

$$f(y_i;\ \underline{\pi}, \Theta) = \sum_{j=1}^{J} \pi_j f_j(y_i; \theta_j), \tag{4.2}$$

where $\sum_{j=1}^{J} \pi_j = 1$ and $\pi_j \ge 0$ for $j = 1, \dots, J$. In this case the likelihood is

$$L = \prod_{i=1}^{n} \left\{ \sum_{j=1}^{J} \pi_j f_j(y_i; \theta_j) \right\}. \tag{4.3}$$

This could be very complicated.

We can make things much simpler by introducing a group-membership variable which is unobserved. The values form auxiliary data so this is an example of data augmentation.

We introduce, for observation i, an auxiliary variable c_i, which can take the values $1, \dots, J$. Then, given that $c_i = j$, the conditional pdf for observation i is simply $f_j(y_i;\ \theta_j)$. The corresponding conditional likelihood is then just

$$L_c = \prod_{i=1}^{n} \pi_{c_i} f_{c_i}(y_i;\ \theta_{c_i}).$$

Now we give c_i a multinomial (or "categorical") distribution, in which $\Pr(c_i = j) = \pi_j$. We give the parameters $\underline{\pi} = (\pi_1, \dots, \pi_J)^T$ and $\Theta = \{\theta_1, \dots, \theta_J\}$ a suitable prior distribution. Then, by "integrating out", i.e. "averaging over", $c_1, \dots, c_n$, we obtain the correct posterior distribution.

The joint probability (density) that $c_i = j$ and $Y_i = y_i$ is

$$f(y_i, c_i = j;\ \underline{\pi}, \Theta) = \pi_j f_j(y_i;\ \theta_j).$$

To find the marginal probability density of y_i we sum over j and obtain (4.2) as required.

4.1.2 MCMC and label-switching

MCMC

Once we have the model set up with the auxiliary variables $c_1, \dots, c_n$ as above, we have a prior distribution with density $f_0(\Theta, \underline{\pi})$ for the parameters and we have initial values for the unknowns, Θ, $\underline{\pi}$, $c_1, \dots, c_n$, then we can proceed with MCMC as follows.

1. Sample a new value for Θ.

The fcd density is proportional to

$$f_0(\Theta, \underline{\pi}) \prod_{j=1}^{J} L_{c,j}$$

where

$$L_{c,j} = \prod_{i \in C_j} f_j(y_i;\ \theta_j)$$

and $i \in C_j$ if $c_i = j$. That is C_j is the set of observations currently assigned to component j. We might well have $f_0(\Theta, \underline{\pi}) = f_{0,\theta}(\Theta) f_{0,\pi}(\underline{\pi})$ in which case the fcd density is proportional to

$$f_0(\Theta) \prod_{j=1}^{J} L_{c,j}$$

2. Sample a new value for $\underline{\pi}$.

The fcd density is proportional to

$$f_0(\Theta, \underline{\pi}) \prod_{j=1}^{J} \pi_j^{n_j}$$

where n_j is the number of observations currently assigned to component j. If $f_0(\Theta, \underline{\pi}) = f_{0,\theta}(\Theta) f_{0,\pi}(\underline{\pi})$ then the fcd density is proportional to

$$f_{0,\pi}(\underline{\pi}) \prod_{j=1}^{J} \pi_j^{n_j}.$$

A popular choice for $f_{0,\pi}(\underline{\pi})$ would be a Dirichlet density. In this case the fcd is also a Dirichlet distribution. Sampling from a Dirichlet distribution is quite easy.

3. Sample a new value for each of $c_1, \ldots, c_n$.

The fcd is a categorical distribution with

$$\Pr(c_i = j) \propto \pi_j f_j(y_i;\ \theta_j).$$

4. Repeat.

Label-switching

Consider the likelihood (4.1).
Suppose that both component distributions are of the same family so that the likelihood is

$$L = \prod_{i=1}^{n} \{\pi f_y(y_i;\theta_1) + (1-\pi) f_y(y_i;\theta_2)\}\,.$$

Suppose that we "switch the labels" and write

$$\tilde{L} = \prod_{i=1}^{n} \left\{\tilde{\pi} f_y(y_i;\tilde{\theta}_1) + (1-\tilde{\pi}) f_y(y_i;\tilde{\theta}_2)\right\}$$

where $\tilde{\pi} = 1-\pi,\ \tilde{\theta}_1 = \theta_2$ and $\tilde{\theta}_2 = \theta_1$.

Clearly $L = \tilde{L}$. The likelihood is therefore bimodal and, in fact, the modes match each other. If the prior does not strongly favour one mode over the other then the posterior distribution will also be bimodal.

In the more general case of (4.3) we can also permute the component labels and get the same likelihood (provided that the distributions are all of the same family). This time the posterior will be multimodal unless the prior strongly favours one mode.

Unless we do something about this, it can cause difficulties in MCMC sampling using the data-augmentation method. If the posterior is multimodal then, eventually, the sampler will jump from one mode to another. The auxiliary variables c_i will suddenly change values so that observations move from one component to another and the parameters "go with them." This might only happen after thousands of iterations. Therefore we might need a very large number of iterations before the sampler has stayed in each mode the correct proportion of the time.

Clearly this behaviour is undesirable. If θ_j is a scalar parameter we can (usually) avoid the problem by imposing an order constraint on the parameters. That is by requiring that $\theta_1 < \theta_2 < \cdots < \theta_J$.

I say "usually" because we can encounter another problem It may be that our mixture model has J components but the data, through the likelihood, suggest only $J-1$ components. Then we might encounter switching between different possibilities for which label is the absent component. There are more advanced methods, beyond the scope of this module, which can deal with this problem.

When θ_j is a vector parameter we may need more ingenuity to devise suitable constraints.

4.1.3 Multivariate mixtures

It is, of course, possible to make a mixture model where the observation $\underline{y}$ is multivariate. For example, we might make several measurements on each of a sample of birds belonging to one species with the idea that there might be two or more subspecies. In two dimensions we might expect a plot of observations y_1 against y_2 to reveal "clusters" of observations.

4.1.4 Continuous mixtures

As well as the finite mixtures described above it is possible to have a mixture model with an infinite number of components. It is also possible to have a *continuous mixture*. In a continuous mixture model, instead of (4.2), we have, for example,

$$f(y_i) = \int_\Omega f_\theta(\theta) f_y(y_i; \theta, \lambda_i)\ d\theta. \tag{4.4}$$

Here θ is a parameter with a continuous distribution specified by the *mixing density* $f_\theta(\theta)$. The range of values of θ is denoted by Ω. There may be other parameters which do not vary in this way and these are denoted by λ_i.

We saw an example of this in Section 3.3.3 where we used Student-t errors in a regression. The model was

$$\begin{array}{rcl} Y_i \mid \mu_i, X_i & \sim & N(\mu_i,\ X_i^{-1}), \\ d\sigma^2 X_i & \sim & \chi^2_d. \end{array}$$

Here μ_i corresponds to λ_i in (4.4) and X corresponds to θ in (4.4). The mixing density is that of a scaled χ^2 distribution and $f_y(y_i; \theta, \lambda_i)$ in (4.4) corresponds to $\phi(X_i^{1/2}[y_i - \mu_i])$ where $\phi()$ is the standard normal pdf.

4.2 Mixture Examples

4.2.1 “Old Faithful”

Table 4.1 shows 299 successive waiting times, in minutes, between the starts of eruptions of the “Old Faithful” geyser in the Yellowstone National Park, Wyoming, USA. The data are taken from Azzalini and Bowman (1990).

Figure 4.1 shows histograms of the data and the logs of the data. In each case we appear to see two distinct modes. However the human brain is very good at spotting patterns, even when they are not there. The evidence in the data migh not be as strong as we imagine. Each of the two-component mixture models below has five parameters, compared to two parameters for a simple normal or gamma model. The likelihood might not distinguish very strongly between all possible values of these five parameters. Therefore careful choice of a prior distribution might be important. If we really believe that there are two sub-populations then our prior may need to reflect this.

Normal mixture

Let us try using a two-component normal mixture model for the log intervals. So

$$
\begin{aligned}
\Pr(c_i = 1) &= \pi \\
\Pr(c_i = 2) &= 1 - \pi \\
\pi &\sim \text{Beta}(a_\pi, b_\pi) \\
y_i \mid \mu_j, \tau_j, c_i = j &\sim N(\mu_j,\ \tau_j^{-1}) \\
\mu_j \mid \mu_0 &\sim N(\mu_0 + \delta_j,\ \tau_\mu^{-1}) \\
\mu_0 &\sim N(M_\mu,\ V_\mu) \\
\tau_j &\sim \text{Ga}(a_\tau, b_\tau)
\end{aligned}
$$

Notice that we have given $\mu_1,\ \mu_2$ a “hierarchical prior.” Each depends on μ_0 which then has a prior of its own. In order to avoid label switching we can impose the restriction $\mu_1 < \mu_2$. We also push the conditional prior means of μ_1, μ_2 apart by making them $\mu_0 + \delta_1$ and $\mu_0 + \delta_2$ respectively, where $\delta_1 = -\delta$ and $\delta_2 = \delta$.

We could also use a hierarchical prior for τ_1 and τ_2 although this is not quite as straightforward with gamma distributions as it is with normal distributions. I have just given them independent priors here. There is no need to impose an order constraint on $\tau_1,\ \tau_2$.

The specification of the prior is completed by giving numerical values to $a_\pi, b_\pi, a_\tau, b_\tau, M_\mu, V_\mu, \tau_\mu, \delta$. We will use the following values.

$$a_\pi = 3, \quad b_\pi = 3, \quad a_\tau = 4, \quad b_\tau = 0.04,$$

$$M_\mu = 4.0 \approx \log(60), \quad V_\mu = 0.30 \approx (\log(3)/2)^2, \quad \tau_\mu = 3.3 \approx (\log(3)/2)^{-2}, \quad \delta = 0.2.$$

Figure 4.2 shows a BUGS model specification for this example.

80	71	57	80	75	77	60	86	77	56	81	50	89	54	90
73	60	83	65	82	84	54	85	58	79	57	88	68	76	78
74	85	75	65	76	58	91	50	87	48	93	54	86	53	78
52	83	60	87	49	80	60	92	43	89	60	84	69	74	71
108	50	77	57	80	61	82	48	81	73	62	79	54	80	73
81	62	81	71	79	81	74	59	81	66	87	53	80	50	87
51	82	58	81	49	92	50	88	62	93	56	89	51	79	58
82	52	88	52	78	69	75	77	53	80	55	87	53	85	61
93	54	76	80	81	59	86	78	71	77	76	94	75	50	83
82	72	77	75	65	79	72	78	77	79	75	78	64	80	49
88	54	85	51	96	50	80	78	81	72	75	78	87	69	55
83	49	82	57	84	57	84	73	78	57	79	57	90	62	87
78	52	98	48	78	79	65	84	50	83	60	80	50	88	50
84	74	76	65	89	49	88	51	78	85	65	75	77	69	92
68	87	61	81	55	93	53	84	70	73	93	50	87	77	74
72	82	74	80	49	91	53	86	49	79	89	87	76	59	80
89	45	93	72	71	54	79	74	65	78	57	87	72	84	47
84	57	87	68	86	75	73	53	82	93	77	54	96	48	89
63	84	76	62	83	50	85	78	78	81	78	76	74	81	66
84	48	93	47	87	51	78	54	87	52	85	58	88	79	

Table 4.1: Waiting times, in minutes, between eruptions of the "Old Faithful" geyser. Data are to be read along the rows.

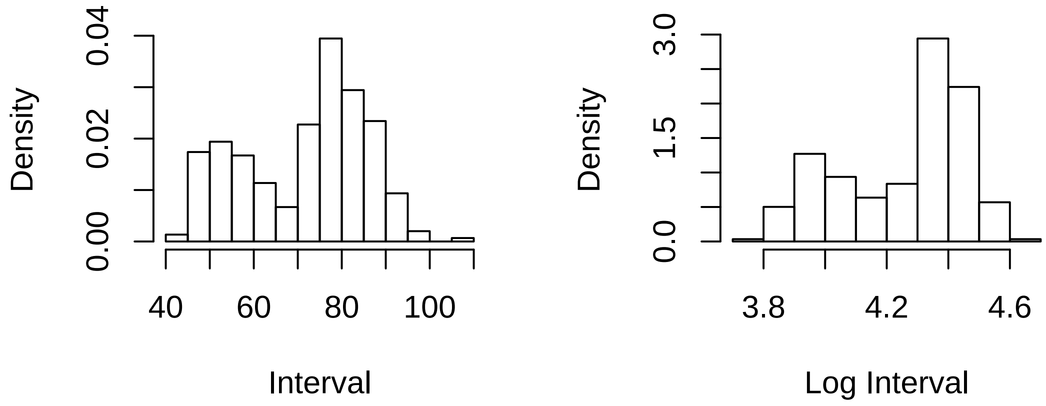

Figure 4.1: Histograms of time intervals between eruptions of "Old Faithful" and logs of the intervals.

```
model faithnorm

{for (i in 1:n)
   {c[i]~dcat(q[])
    y[i]~dnorm(mu[c[i]],tau[c[i]])
    }

 for (j in 1:2)
   {tau[j]~dgamma(4,0.04)
    }

 mumean[1]<-mu0-0.2
 mumean[2]<-mu0+0.2
 mu[1]~dnorm(mumean[1],3.3) I(,mu[2]) # This imposes the order constraint.
 mu[2]~dnorm(mumean[2],3.3) I(mu[1],)

 mu0~dnorm(4.0,p.mu)
 p.mu<-1/0.3

 pi~dbeta(3,3)
 q[1]<-pi
 q[2]<-1-pi
 }
```

Figure 4.2: BUGS model specification for "Old Faithful" normal mixture model.

Gamma mixture

As an alternative to the normal mixture for the log intervals, which is, of course, equivalent to a lognormal mixture for the intervals, we could try a gamma mixture for the intervals themselves.

$$
\begin{aligned}
\Pr(c_i = 1) &= \pi \\
\Pr(c_i = 2) &= 1 - \pi \\
\pi &\sim \text{Beta}(a_\pi, b_\pi) \\
t_i \mid \alpha_j, \beta_j, c_i = j &\sim \text{Ga}(\alpha_j,\ \beta_j) \\
\beta_j &= \alpha_j / \lambda_j \\
\lambda_j &= \exp(\mu_j) \\
\mu_j \mid \mu_0 &\sim N(\mu_0 + \delta_j,\ \tau_\mu^{-1}) \\
\mu_0 &\sim N(M_\mu,\ V_\mu) \\
\alpha_j &\sim \text{Ga}(a_\alpha, b_\alpha)
\end{aligned}
$$

Since the mean of a gamma(α_j, β_j) distribution is α_j/β_j and we set $\beta_j = \alpha_j/\lambda_j$, the mean

```
model faithgamma

{for (i in 1:n)
   {c[i]<-dcat(q[])
    t[i]~dgamma(alpha[c[i]],beta[c[i]])
    }

 for (j in 1:2)
   {alpha[j]~dgamma(3,0.1)
    beta[j]<-alpha[j]/lambda[j]
    lambda[j]<-exp(mu[j])
    }

 mumean[1]<-mu0-0.2
 mumean[2]<-mu0+0.2
 mu[1]~dnorm(mumean[1],3.3) I(,mu[2]) # This imposes the order constraint.
 mu[2]~dnorm(mumean[2],3.3) I(mu[1],)

 mu0~dnorm(4.0,p.mu)
 p.mu<-1/0.3

 pi~dbeta(3,3)
 q[1]<-pi
 q[2]<-1-pi
 }
```

Figure 4.3: BUGS model specification for "Old Faithful" gamma mixture model.

interval, in component j, is λ_j. We then treat $\mu_j = \log(\lambda_j)$ in the same way as we treated μ_j in the lognormal mixture. Of course the log of the mean is not the same as the mean of the logs but, in this case, this difference has little effect. (To avoid this slight discrepancy we would have to make λ_j the median rather than the mean but this is not convenient with a gamma distribution).

I have not used a hierarchical prior for α_1, α_2. I have just given them independent priors here. There is no need to impose an order constraint on $\alpha_1,\ \alpha_2$.

We will use the following values to complete the prior specification.

$$a_\pi = 3, \quad b_\pi = 3, \quad a_\alpha = 3, \quad b_\alpha = 0.1,$$

$$M_\mu = 4.0 \approx \log(60), \quad V_\mu = 0.30 \approx (\log(3)/2)^2, \quad \tau_\mu = 3.3 \approx (\log(3)/2)^{-2}, \quad \delta = 0.2.$$

Figure 4.3 shows a BUGS model specification for this example.

4.2.2 Road vehicle headways

Cowburn (2003) describes the use of mixture models for the time gaps, or "headways", between vehicles passing along a road. See also Cowburn and Farrow (2007). The idea is that headways fall naturally into one of two sub-populations:

1. Headways where the following vehicle is not impeded by the vehicle in front.

2. "Congested" headways where the following vehicle is impeded by the vehicle in front.

Cowburn proposed that non-congested headways would follows an exponential distribution, that is a $\text{Ga}(1, \beta_1)$ distribution, and congested headways would follow a $\text{Ga}(\alpha_2, \beta_2)$ distribution with $\alpha_2 > 1$. (A number of other mixture models have been proposed in the highway engineering literature). Successive headways are independent (given the model parameters) in this version of the model. We will see a version where this is not the case in the next lecture.

```
model headway;

{

 for (i in 1:N)
     {c[i]~dcat(q[])
      t[i]~dgamma(alpha[c[i]],beta[c[i]])
      }


 alpha[1]<-1
 alpha[2]<-1+aa
 aa~dgamma(1,0.5)
 beta[1]~dgamma(2,8) I(,bb)
 beta[2]<-alpha[2]*bb
 bb~dgamma(1,2) I(beta[1],)
 pi~dbeta(1,2)
 q[1]<-pi
 q[2]<-1-pi

 mu[1]<-1/beta[1]
 mu[2]<-1/bb

 }
```

Figure 4.4: BUGS specification for independent headways model.

A BUGS model specification is given in Figure 4.4. The constraint that $\alpha_2 > 1$ is imposed by letting $\alpha_2 = 1 + A$ where $A \sim \text{Ga}(a_A, b_A)$. In the BUGS code A is represented by `aa`. We have $a_A = 2$ and $b_A = 8$. The mean headway in component 1 is $\mu_1 = \beta_1^{-1}$. The mean headway in component 2 is $\mu_2 = \alpha_2/\beta_2$. We set $\beta_2 = \alpha_2 B$ where $B \sim \text{Ga}(a_B, b_B)$. In the BUGS code B is represented by `bb`. We have $a_B = 1$ and $b_B = 2$. Thus $\mu_2 = B^{-1}$. By imposing the constraint $B > \beta_1$ we ensure that $\mu_1 > \mu_2$. (So, in fact, B has *truncated* gamma distribution). The headways are recorded in seconds.

4.3 Hidden Markov Models

4.3.1 Introduction

In the mixture models above, the unobserved (or *latent*) component memberships c_i are independent of each other, given the model parameters. Sometimes, when the data have a natural ordering, as in a time series, we may wish to allow the component memberships to depend on each other.

Figure 4.5 shows the logarithms of the time intervals between eruptions of “Old Faithful”. The i^{th} log interval y_i is plotted against the preceding log interval y_{i-1}. Clearly successive intervals are not independent. One way to model this might be to suppose that there are “short intervals” and “long intervals” and that a short interval is always followed by a long interval but a long interval may be followed by either a short interval or a long interval. Thus we could model the sequence $c_1, \ldots, c_n$ using a two-state Markov chain with the following transition matrix, where $q_{j,k} = \Pr(c_i = j \mid c_{i-1} = k)$.

$$\begin{pmatrix} q_{1,1} & q_{1,2} \\ q_{2,1} & q_{2,2} \end{pmatrix} = \begin{pmatrix} 0 & \pi \\ 1 & 1-\pi \end{pmatrix}. \tag{4.5}$$

Of course, before we saw the data we would not know about this pattern so it could be argued that we should use a more general model in which we allow $q_{1,1} > 0$. In this case we would have

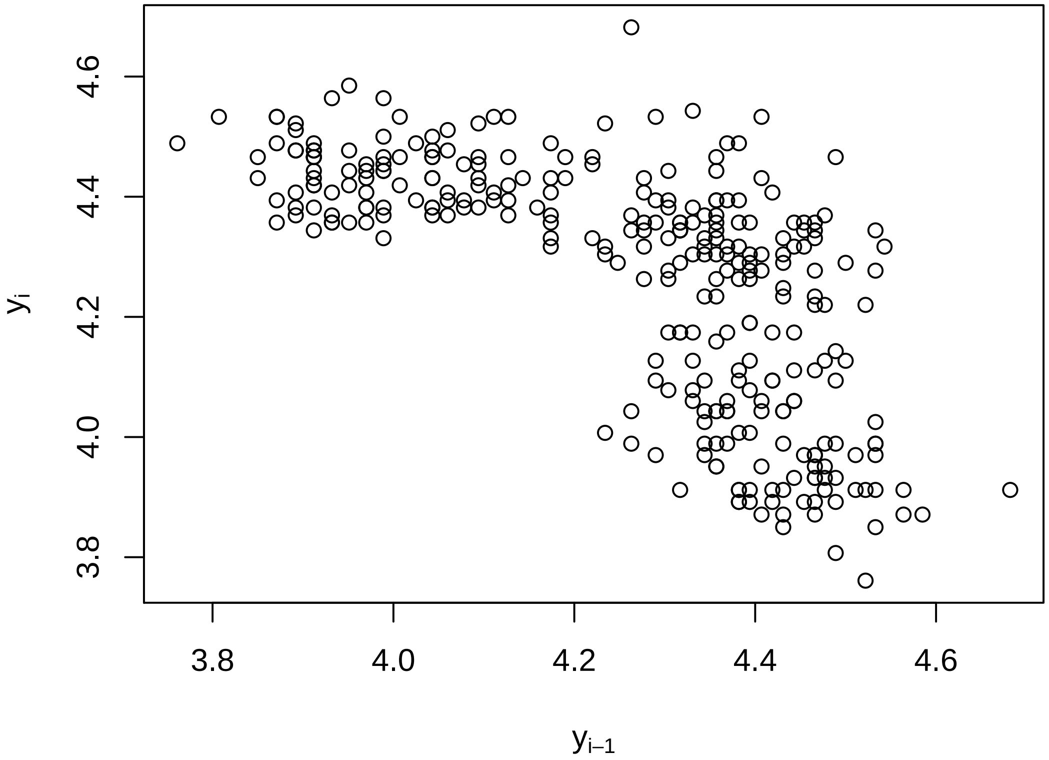

Figure 4.5: Logarithms of time intervals between eruptions of “Old Faithful”. The i^{th} log interval y_i is plotted against the preceding log interval y_{i-1}.

$$\begin{pmatrix} q_{1,1} & q_{1,2} \\ q_{2,1} & q_{2,2} \end{pmatrix} = \begin{pmatrix} 1-\pi_2 & \pi_1 \\ \pi_2 & 1-\pi_1 \end{pmatrix}. \tag{4.6}$$

This is an example of a *hidden Markov model* or HMM. In this case there are two *hidden states.* There are many different kinds of HMM and they have many applications, in such diverse areas as time series, DNA sequences and linguistics. In general, in a HMM, we have a sequence of (possibly vector) observations $\ldots y_{i-1}, y_i, y_{i+1} \ldots$ where the distribution of y_i depends on the value of an unobserved (i.e. *latent*) (possibly vector) variable x_i and the sequence $\ldots x_{i-1}, x_i, x_{i+1}, \ldots$ forms a Markov chain. There may, of course, be more than two hidden states.

Figure 4.6 shows a DAG for a typical HMM. There will typically also be unknown parameters on which the distributions depend but these have been omitted. Notice that (given the model parameters) the observations Y only depend on each other through the latent variables X. Figure 4.7 shows a DAG with the addition of the unknown parameters $\underline{\mu} = (\mu_1, \mu_2)^T$, $\underline{\tau} = (\tau_1, \tau_2)^T$ and π, for a case where the conditional distribution of Y_i when $X_i = c_i$ is $N(\mu_{c_i},\ \tau_{c_i}^{-1})$ for $c_i = 1, 2$.

4.3.2 Two-state hidden Markov model

In the Old Faithful model, the latent variable X_i is the component membership c_i and there are two components. This is an example of a two-state HMM.

The transition matrix (4.6) defines the conditional distribution of c_i given c_{i-1}. To complete the model specification we have to give a distribution to the initial state c_1 (or to c_0, the state immediately before the start of the data). Very often we regard the process as stationary. That is

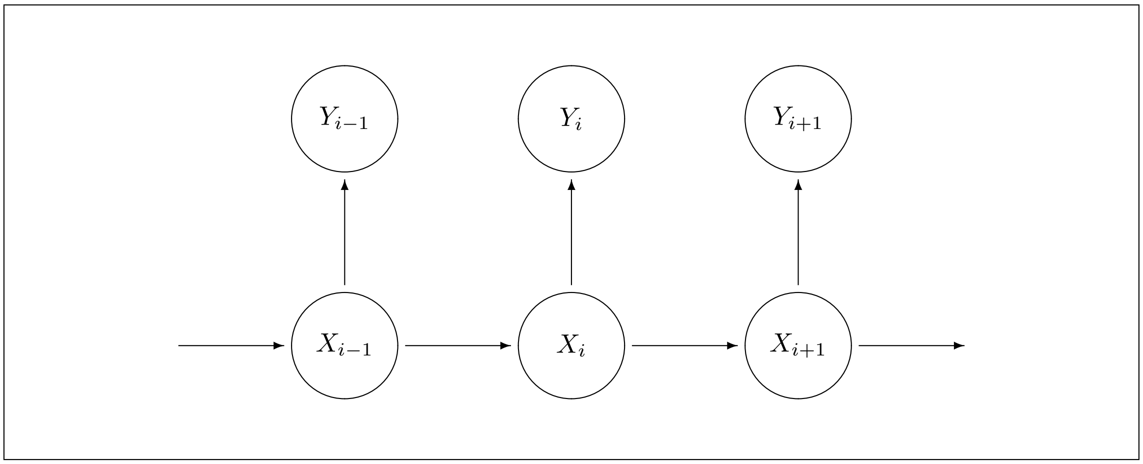

Figure 4.6: Directed acyclic graph for hidden Markov model

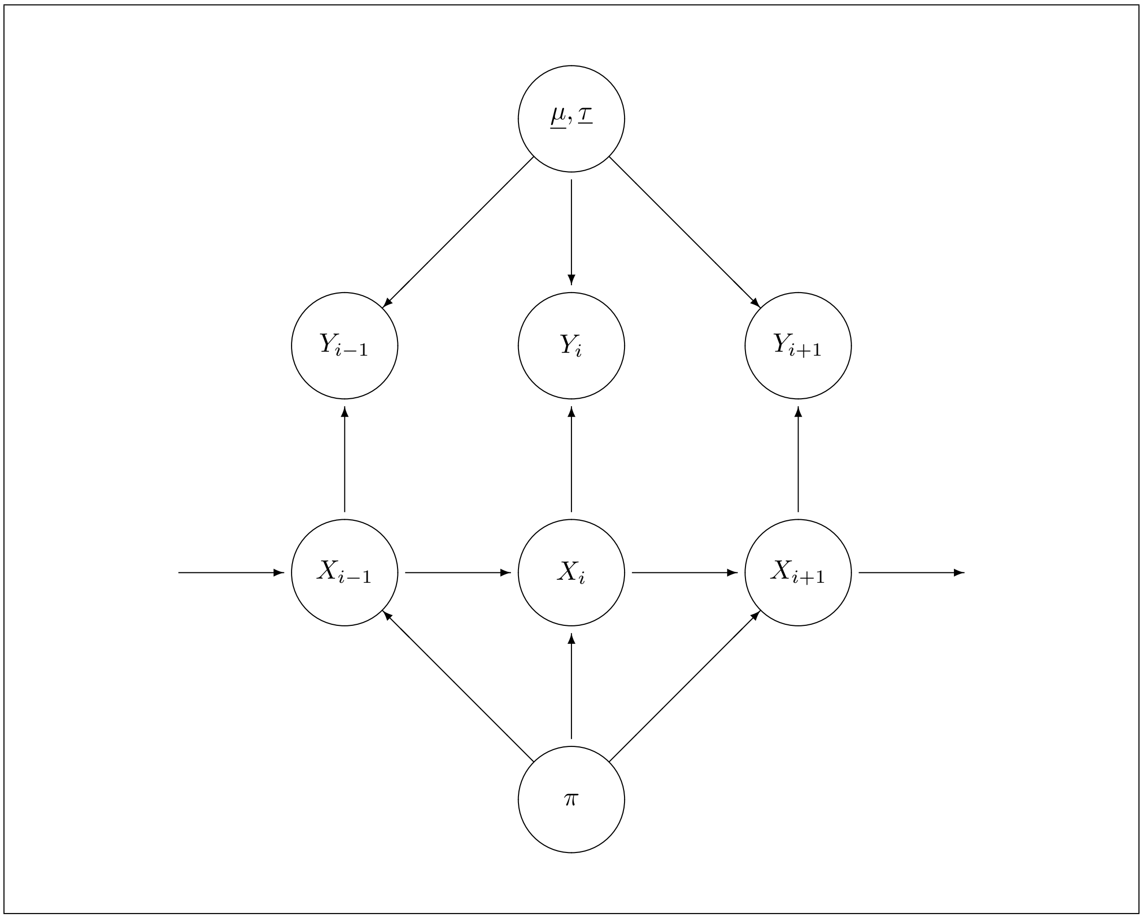

Figure 4.7: Directed acyclic graph for hidden Markov model showing unknown parameters

the properties are not changing over time. In this case the initial state should have the stationary distribution of the Markov chain which can be found by solving

$$\begin{pmatrix} 1-\pi_2 & \pi_1 \\ \pi_2 & 1-\pi_1 \end{pmatrix} \begin{pmatrix} P_1 \\ P_2 \end{pmatrix} = \begin{pmatrix} P_1 \\ P_2 \end{pmatrix}$$

for P_1 and P_2 with the constraint that $P_1 + P_2 = 1$. The solution is

$$P_1 = \Pr(c_1 = 1) = \frac{\pi_1}{\pi_1 + \pi_2}, \qquad P_2 = \Pr(c_1 = 2) = \frac{\pi_2}{\pi_1 + \pi_2}. \tag{4.7}$$

Unfortunately we have two problems with this.

1. BUGS software can not (or could not) handle the resulting likelihood with the complication of (4.7). We can, of course, write a Gibbs or Metropolis-Hastings algorithm of our own in R or some other programming language. However we could also use a trick to make BUGS work and which would give the correct result to a good approximation. The trick is to add the states $c_{-s}, \ldots, c_0$ as auxiliary variables for some reasonably large s (e.g. 30). We then give c_{-s} a convenient distribution, e.g. $\Pr(c_{-s} = 1) = \Pr(c_{-s} = 2) = 0.5$, or even just fix its value. The choice of this distribution or value has little effect on the posterior distribution. (This can be checked numerically). Figure 4.8 shows a BUGS model specification for the "Old Faithful" model, using (4.5) and normal distributions for the log interval times.

2. The BUGS model shown in Figure 4.8 worked satisfactorily in previous years. However, this year, it causes R to crash. This seems to be associated with a change in the version of R. Perhaps a change needs to be made to the BRugs package and this has not been made.

Because of these problems, especially Point 2, we will not attempt to use BRugs for hidden Markov models this year but, instead use specially-written R functions to demonstrate the use of MCMC with these models. In many cases we can sample from most of the fcds straightforwardly. The one exception is sampling values for π_1 and π_2 because of the term for the initial state. However we can use a Metropolis-Hastings sampler for this and thus have a Metropolis-within-Gibbs scheme.

For sampling π_1 and π_2 we can proceed as follows.

Suppose that the current value of the state at time 1 is c_1. Let

$$P(c_1, \pi_1, \pi_2) = \frac{\pi_{c_1}}{\pi_1 + \pi_2}.$$

Let the current numbers of state transitions, according to the currently allocated states, be $n_{1,1}, n_{1,2}, n_{2,1}, n_{2,2}$. That is, according to the allocation of observations to states at this iteration, we have $n_{j,k}$ transitions from state k to state j. Let $L^\star$ be the "likelihood" based just on these transitions. Then

$$L^\star = (1-\pi_2)^{n_{1,1}} \pi_2^{n_{2,1}} \pi_1^{n_{1,2}} (1-\pi_1)^{n_{2,2}}.$$

Suppose, for example, that we have independent beta prior distributions for π_1 and π_2, so that the joint prior density for π_1 and π_2 is

$$g_1^{(0)}(\pi_1) g_2^{(0)}(\pi_2) \propto \pi_1^{a_1-1}(1-\pi_1)^{b_1-1} \pi_2^{a_2-1}(1-\pi_2)^{b_2-1}.$$

Then, based just on $L^\star$, the joint "posterior" density is $g_1^{(1)}(\pi_1) g_2^{(1)}(\pi_2)$ where $g_1^{(1)}(\pi_1)$ is the density of a Beta$(a_1 + n_{1,2},\ b_1 + n_{2,2})$ distribution and $g_2^{(1)}(\pi_2)$ is the density of a Beta$(a_2 + n_{2,1},\ b_2 + n_{1,1})$ distribution. As a proposal, we can sample values $\pi_{1,\text{prop}}$ and $\pi_{2,\text{prop}}$ for π_1 and π_2 from this joint "posterior". That is we take independent samples from the two beta distributions.

However the fcd has density $k(n_{1,1}, n_{1,2}, n_{2,1}, n_{2,2}) P(c_1, \pi_1, \pi_2) g_1^{(1)}(\pi_1) g_2^{(1)}(\pi_2)$ where the constant $k(n_{1,1}, n_{1,2}, n_{2,1}, n_{2,2})$ does not depend on π_1 or π_2. Therefore, if $\pi_{1,\text{old}}$ and $\pi_{2,\text{old}}$ are the current values, from the preceding iteration, the acceptance ratio is

```
model faithnormhmm

{p0[1]<-0.5
 p0[2]<-0.5
 cc[1]~dcat(p0[])

 for (i in 2:30)
   {cc[i]~dcat(q[,cc[i-1]])   # This is the "burn-in"section.
    }

 c[1]~dcat(q[,cc[30]])     # This is for the initial state.

 for (i in 2:n)
   {c[i]~dcat(q[,c[i-1]])
    }

 for (i in 1:n)
   {y[i]~dnorm(mu[c[i]],tau[c[i]])
    }

 for (j in 1:2)
   {tau[j]~dgamma(4,0.04)
    }

 mumean[1]<-mu0-0.2
 mumean[2]<-mu0+0.2
 mu[1]~dnorm(mumean[1],3.3) I(,mu[2]) # This imposes the order constraint.
 mu[2]~dnorm(mumean[2],3.3) I(mu[1],)

 mu0~dnorm(4.0,p.mu)
 p.mu<-1/0.3

 q[1,2]<-pi
 pi~dbeta(1,1)
 q[2,2]<-1-q[1,2]
 q[1,1]<-0.0
 q[2,1]<-1-q[1,1]

 }
```

Figure 4.8: BUGS model specification for “Old Faithful” normal hidden Markov model.

$$\begin{aligned} A &= \frac{k(n_{1,1},n_{1,2},n_{2,1},n_{2,2})P(c_1,\pi_{1,\text{prop}},\pi_{2,\text{prop}})g_1^{(1)}(\pi_{1,\text{prop}})g_2^{(1)}(\pi_{2,\text{prop}})}{k(n_{1,1},n_{1,2},n_{2,1},n_{2,2})P(c_1,\pi_{1,\text{old}},\pi_{2,\text{old}})g_1^{(1)}(\pi_{1,\text{old}})g_2^{(1)}(\pi_{2,\text{old}})} \\ &\quad \times \frac{g_1^{(1)}(\pi_{1,\text{old}})g_2^{(1)}(\pi_{2,\text{old}})}{g_1^{(1)}(\pi_{1,\text{prop}})g_2^{(1)}(\pi_{2,\text{prop}})} \\ &= \frac{P(c_1,\pi_{1,\text{prop}},\pi_{2,\text{prop}})}{P(c_1,\pi_{1,\text{old}},\pi_{2,\text{old}})}. \end{aligned}$$

When sampling the component memberships c_i, the fcd depends on the transition probability from the preceding state, the transition probability to the succeeding state and the conditional density of the observation. So, for example, suppose that the conditional distributions are normal with means μ_1 and μ_2 and precisions τ_1 and τ_2 and the observation is y_i. Then write the conditional densities as $f(y_i;\mu_1,\tau_1)$ and $f(y_i;\mu_2,\tau_2)$. Given that the preceding state is c_{i-1} and the succeeding state is c_{i+1} and the transition matrix is as given in (4.6), then the "prior probability" that $c_i = j$ is proportional to

$$\Pr(c_i = j \mid c_{i-1})\Pr(c_{i+1} \mid c_i = j) = q_{j,c_{i-1}}q_{c_{i+1},j}.$$

Multiplying "prior" by "likelihood" we find that the fcd probability that $c_i = j$ is proportional to $q_{j,c_{i-1}}q_{c_{i+1},j}f(y_i;\mu_j,\tau_j)$. Therefore the fcd probability that $c_i = j$ is

$$\frac{q_{j,c_{i-1}}q_{c_{i+1},j}f(y_i;\mu_j,\tau_j)}{\sum_{c=1}^{2} q_{c,c_{i-1}}q_{c_{i+1},c}f(y_i;\mu_c,\tau_c)}.$$

Figures 4.9 and 4.10 show a R function for a two-state HMM, as developed here, with normal conditional distributions. The conjugate normal-gamma form is used for the prior for each normal distribution.Note that the R command `table` produces the transpose of the table of counts used in these notes. Therefore, in the R function `hmmnorm`, the variables `ns[2,1]` and `ns[1,2]` correspond to $n_{1,2}$ and $n_{2,1}$ respectively.

4.3.3 Forward-backward algorithm

Mixing can be poor when using a Gibbs sampler with a HMM if we sample the hidden state at each time point separately. This is because there can be strong correlation in the posterior distribution between the states at neighbouring time points. We can overcome this problem by sampling the whole collection of hidden states as a block. This can be done using a procedure called the *forward-backward algorithm*. We do not have time to cover this in this course. See, for example, Scott (2002).

4.4 Practical 4

4.4.1 Simulated normal mixture data

Mixture models can sometimes be tricky to fit so we will start with an artificial example which is deliberately made so that it will work well.

The data `mixturedata.txt` and the BUGS model file `mixturenormbug.txt` can both be obtained from the web page.

1. It is often necessary to help the software by providing initial values for the Gibbs sampler. In the case of mixture models it is also particularly important to check convergence. One way to do this is to run the sampler more than once, starting with different initial values. Therefore, use Notepad (or whatever you prefer) to create two different initial value files. The first should be called

   ```
   mixturenorminits1.txt
   ```

```
hmmnorm<-function(niter,prior,y)
{n<-length(y)
 cv<-ifelse(y<mean(y),1,2)                    # Initialise component memberships.
 m<-  matrix(nrow=2,ncol=2)
 mu<- matrix(nrow=niter,ncol=2)
 tau<-matrix(nrow=niter,ncol=2)
 pi<- matrix(nrow=niter,ncol=2)
 proportion<-numeric(niter)
 piprop<-numeric(2)
 piold <-c(0.5,0.5)
 for (iter in 1:niter)
     {ns<-table(cv[1:(n-1)],cv[2:n])                                  # Count the transitions.
      piprop[1]<-rbeta(1,prior$a[1]+ns[2,1],prior$b[1]+ns[2,2])        # Proposal for pi_1.
      piprop[2]<-rbeta(1,prior$a[2]+ns[1,2],prior$b[2]+ns[1,1])        # Proposal for pi_2.
      Pprop<-piprop[cv[1]]/sum(piprop)                                 # Stationary probability.
      Pold <-piold[cv[1]]/sum(piold)                                   # Stationary probability.
      A<-min(1,Pprop/Pold)                                             # Acceptance probability.
      U<-runif(1)
      if (U<A)                                                         # M-H sampling for pi.
         {pi[iter,]<-piprop
          piold<-piprop
          }
      else
         {pi[iter,]<-piold
          }
      m[1,1]<-1-pi[iter,2]                                             # Transition matrix.
      m[1,2]<-pi[iter,1]
      m[2,1]<-pi[iter,2]
      m[2,2]<-1-pi[iter,1]
      for (comp in 1:2)                                                # Sample other parameters.
          {nc<-sum(cv==comp)
           if (nc>0)
              {ycomp<-y[cv==comp]
               ybar<-mean(ycomp)
               s2n<-(sum(ycomp*ycomp)-nc*ybar*ybar)/nc
               ycd<-ycomp-prior$m[comp]
               r2<-sum(ycd*ycd)/nc
               k1<-prior$c[comp]+nc
               d1<-prior$d[comp]+nc
               m1<-(prior$c[comp]*prior$m[comp]+nc*ybar)/k1
               vd<-(prior$c[comp]*r2+nc*s2n)/k1
               v1<-(prior$d[comp]*prior$v[comp]+nc*vd)/d1
               tau[iter,comp]<-rgamma(1,(d1/2),(d1*v1/2))
               sd<-sqrt(1/(k1*tau[iter,comp]))
               mu[iter,comp]<-rnorm(1,m1,sd)
               }
            else
              {tau[iter,comp]<-rgamma(1,(prior$d[comp]/2),(prior$d[comp]*prior$v[comp]/2))
               sd<-sqrt(1/(prior$c[comp]*tau[iter,comp]))
               mu[iter,comp]<-rnorm(1,prior$m[comp],sd)
               }
           }
```

Figure 4.9: R function for a two-state HMM with normal conditional distributions (Part 1).

```
      P<-pi[iter,]/sum(pi[iter,])                                       # Stationary probabilities.
      pc<-P*m[cv[2],]                                                   # "Prior probs" for cv_1.
      sd<-numeric(2)
      for (comp in 1:2)
          {sd[comp]<-sqrt(1/tau[iter,comp])
           pc[comp]<-pc[comp]*dnorm(y[1],mu[iter,comp],sd[comp])        # "Prior times likelihood".
           }
      pc<-pc/sum(pc)                                                    # Normalise.
      cv[1]<-2-rbinom(1,1,pc[1])                                        # Sample cv_1.
      for (t in 2:(n-1))
          {pc<-m[,cv[t-1]]*m[cv[t+1],]                                  # "Prior probs" for cv_t.
           for (comp in 1:2)
               {pc[comp]<-pc[comp]*dnorm(y[t],mu[iter,comp],sd[comp])   # "Prior times likelihood".
                }
           pc<-pc/sum(pc)                                               # Normalise.
           cv[t]<-2-rbinom(1,1,pc[1])                                   # Sample cv_t.
           }
       pc<-m[,cv[n-1]]                                                  # "Prior probs" for cv_n.
       for (comp in 1:2)
           {pc[comp]<-pc[comp]*dnorm(y[n],mu[iter,comp],sd[comp])       # "Prior times likelihood".
            }
       pc<-pc/sum(pc)                                                   # Normalise.
       cv[n]<-2-rbinom(1,1,pc[1])                                       # Sample cv_n.
       proportion[iter]<-sum(cv==1)/n                                   # Proportion in component 1.
       }
out<-list(mu=mu,tau=tau,pi=pi,proportion=proportion)
out
 }
```

Figure 4.10: R function for a two-state HMM with normal conditional distributions (Part 2).

and should contain the following.

```
list(mu=c(1,7),pi=0.3)
```

The second should be called

```
mixturenorminits2.txt
```

and should contain the following.

```
list(mu=c(1,7),pi=0.7)
```

So, we will start with very different probabilities of an observation being in component 1.

2. Check convergence of `pi` , the probability for component 1. We will set the two different intial values in two separate chains and run them without burn-in periods.

```
modelCheck("mixturenormbug.txt")
modelData("mixturedata.txt")
modelCompile(2)
modelInits("mixturenorminits1.txt")
modelInits("mixturenorminits2.txt")
modelGenInits()
samplesSet("pi")
modelUpdate(1000)
samplesHistory("pi")
```

   Look at the resulting graph. You should see that "convergence" has been quite quick.

3. Compute the posterior distribution. This time we will use a burn-in.

```
modelCheck("mixturenormbug.txt")
modelData("mixturedata.txt")
modelCompile(2)
modelInits("mixturenorminits1.txt")
modelInits("mixturenorminits2.txt")
modelGenInits()
modelUpdate(1000)
samplesSet(c("pi","mu","tau"))
modelUpdate(2000)
```

4. Look at the results. For example:

```
samplesStats(c("pi","mu","tau"))
samplesDensity("pi")
```

4.4.2 Old Faithful log-normal mixture

We will fit a two-component normal mixture to the logs of the intervals between eruptions of "Old Faithful."

The data `geyserlogdata.txt` and the BUGS model file `faithnormbug.txt` can both be obtained from the web page.

Use Notepad (or whatever you prefer) to create two different initial value files. The first should be called

```
faithnorminits1.txt
```

and should contain the following.

```
list(mu=c(4.0,4.4),pi=0.3)
```

The second should be called

```
faithnorminits2.txt
```

and should contain the following.

```
list(mu=c(4.0,4.4),pi=0.7)
```

So, we will start with very different probabilities of an observation being in component 1.

1. Check convergence of `pi` , the probability for component 1. We will set the two different intial values in two separate chains and run them without burn-in periods.

   ```
   modelCheck("faithnormbug.txt")
   modelData("geyserlogdata.txt")
   modelCompile(2)
   modelInits("faithnorminits1.txt")
   modelInits("faithnorminits2.txt")
   modelGenInits()
   samplesSet("pi")
   modelUpdate(1000)
   samplesHistory("pi")
   ```

   Look at the resulting graph. Has "convergence" been quick?

2. Compute the posterior distribution. This time we will use a burn-in.

   ```
   modelCheck("faithnormbug.txt")
   modelData("geyserlogdata.txt")
   modelCompile(2)
   modelInits("faithnorminits1.txt")
   modelInits("faithnorminits2.txt")
   modelGenInits()
   modelUpdate(1000)
   samplesSet(c("pi","mu","tau"))
   modelUpdate(2000)
   ```

3. Look at the results. For example:

   ```
   samplesStats(c("pi","mu","tau"))
   samplesDensity("pi")
   ```

4.4.3 Old Faithful gamma mixture

We will fit a two-component gamma mixture to the intervals between eruptions of "Old Faithful."

The data `geyserdata.txt` and the BUGS model file `faithgammabug.txt` can both be obtained from the web page.

We can use the same initial value files as we used for the normal mixture.

1. Check convergence of `pi` , the probability for component 1. We will set the two different intial values in two separate chains and run them without burn-in periods.

```
modelCheck("faithgammabug.txt")
modelData("geyserdata.txt")
modelCompile(2)
modelInits("faithnorminits1.txt")
modelInits("faithnorminits2.txt")
modelGenInits()
samplesSet("pi")
modelUpdate(1000)
samplesHistory("pi")
```

Look at the resulting graph. Has "convergence" been quick?

2. Compute the posterior distribution. This time we will use a burn-in.

   ```
   modelCheck("faithgammabug.txt")
   modelData("geyserdata.txt")
   modelCompile(2)
   modelInits("faithnorminits1.txt")
   modelInits("faithnorminits2.txt")
   modelGenInits()
   modelUpdate(1000)
   samplesSet(c("pi","alpha","beta"))
   modelUpdate(3000)
   ```

3. Look at the results. For example:

   ```
   samplesStats(c("pi","alpha","beta"))
   samplesDensity("pi")
   samplesDensity("alpha")
   samplesDensity("beta")
   ```

4. The marginal posterior distributions for β_1 and β_2 are quite similar. Perhaps values of β_1 and β_2 which are close to each other would represent the data well. Let us look at the posterior distribution of β_1/β_2.

   ```
   beta1<-samplesSample("beta[1]")
   beta2<-samplesSample("beta[2]")
   betaratio<-beta1/beta2
   plot(density(betaratio))
   ```

4.4.4 Road traffic headways (independent)

We will fit an exponential/gamma mixture model to some road traffic headway data. Two files of data are avilable on the web page. They are:

```
dd01data.txt
feb28peledata.txt
```

The first was collected by my research student, Ged Cowburn. I collected the second. You can use either one. They have slightly different characteristics.

The BUGS model file is also available on the Web page as `headway0bug.txt`.

We will need some initial value files. Create two files as follows.

```
headwayinits1.txt
```

containing

```
list(aa=2, bb=0.5, pi=0.1)
```

and

```
headwayinits2.txt
```

containing

```
list(aa=2, bb=0.5, pi=0.7)
```

1. Check convergence of `pi` , the probability for component 1. We will set the two different intial values in two separate chains and run them without burn-in periods. I will use `dd01data.txt` but you can use `feb28peledata.txt` if you wish.

   ```
   modelCheck("headway0bug.txt")
   modelData("dd01data.txt")
   modelCompile(2)
   modelInits("headwayinits1.txt")
   modelInits("headwayinits2.txt")
   modelGenInits()
   samplesSet("pi")
   modelUpdate(1000)
   samplesHistory("pi")
   ```

   Look at the resulting graph. You will probably see that the samplers have “converged” but that “mixing” is not very good. Therefore we need to take a large number of samples.

2. Compute the posterior distribution. This time we will use a burn-in.

   ```
   modelCheck("headway0bug.txt")
   modelData("dd01data.txt")
   modelCompile(2)
   modelInits("headwayinits1.txt")
   modelInits("headwayinits2.txt")
   modelGenInits()
   modelUpdate(2000)
   samplesSet(c("pi","alpha","beta"))
   modelUpdate(10000)
   ```

3. Look at the results. For example:

   ```
   samplesStats(c("pi","alpha","beta"))
   samplesDensity("pi")
   samplesDensity("alpha")
   samplesDensity("beta")
   ```

4.4.5 Old Faithful (log-normal hidden Markov model)

Try fitting the log-normal hidden Markov model to the Old Faithful data. The R function shown in Figures 4.9 and 4.10 is available on the Web page as `hmmR.txt`. It seems to work well despite the fact that I have not used any defence against label-switching, other than giving the two components different prior means and initialising the component memberships to favour the correct allocation. The logs of the intervals, in a suitable form, are in a file `geyserlogdatab.txt` on the Web page.

1. Load the data:

   ```
      y<-scan("geyserlogdatab.txt")
   ```

2. Set up the prior:

```
a<-c(1,1)
b<-c(1,1)
m<-c(3.5,4.5)
d<-c(8,8)
v<-c(0.01,0.01)
c<-c(0.01,0.01)
prior<-list(a=a,b=b,m=m,d=d,v=v,c=c)
```

3. Install the function:

   ```
   source("hmmR.txt")
   ```

4. Try, for example, 1000 iterations:

   ```
   test<-hmmnorm(1000,prior,y)
   ```

5. Have a look at the results. For example:

   ```
   mu<-test$mu
   Iteration<-1:1000
   plot(Iteration,mu[,1],type="l",col=2,ylim=c(3.5,5))
   lines(Iteration,mu[,2],col=3)
   pi<-test$pi
   plot(Iteration,pi[,1],type="l",col=2,ylim=c(0.0,1.0))
   lines(Iteration,pi[,2],col=3)
   plot(density(mu[,1]))
   ```

   In particular, notice that, as expected π_2 turns out to be close to 1.

6. Try anything else you like.

Note that the function gives a single chain. If you want to try multiple chains, you have to run the function multiple times.

4.4.6 Headways (hidden Markov model)

Cowburn and Farrow (2007) discussed fitting hidden Markov models to series of road-vehicle headways. The two component distributions were as in section 4.2.2. The transition matrix was as in (4.6).

This model is complicated by the fact that both parameters are unknown in one of the conditional gamma distributions and sampling the fcd for the "shape" or "index" parameter (*ie* the first parameter) is not straightforward. To avoid this complication we will fix its value at 4.0.

You can download a suitable R function from the file `hmmheadwayR.txt` on the Web page. I have marked with `######` the places where changes have been made from `hmmR`.The first set of headway data is available in the file `dd01datab.txt` on the Web page. (You could also easily edit the other set to make it usable this way if you so wished). The function seems to work well even though, again, I have not really defended against label switching.

1. Load the data:

   ```
    t<-scan("dd01datab.txt")
   ```

2. Set up the prior:

   ```
   a<-c(1,1)
   b<-c(2,2)
   abeta<-c(2,1)
   bbeta<-c(8,0.5)
   priorh<-list(a=a,b=b,abeta=abeta,bbeta=bbeta)
   ```

3. Install the function:

   ```
   source("hmmheadwayR.txt")
   ```

4. Try, for example, 1000 iterations:

   ```
   test<-hmmheadway(1000,priorh,t)
   ```

5. Have a look at the results. For example:

   ```
   mean<-test$mean
   Iteration<-1:1000
   plot(Iteration,mean[,1],type="l",col=2,ylim=c(0,20),ylab="Mean headway")
   lines(Iteration,mean[,2],col=3)
   lrr<-log(test$pi[,2]/(1-test$pi[,1]))
   plot(density(lrr))
   ```

   I expect that you will see that a short burn-in might help. You can easily delete the first few iterations from the output.

   The quantity `lrr` is the log relative risk for being in component 2 next time comparing being in component 1 now with being in component 2 now. As you can see, there is little evidence that this differs much from zero. Thus there is little evidence that the component memberships are not independent and that we need a hidden Markov model at all. At least, this is what is suggested by this model!

6. Try anything else you like.