APTS Computer Intensive Statistics 2013/14

Computer Practical 3 Outline Solutions Markov Chains and Monte Carlo

1. A warm-up. In a simplified model of the game of Monopoly, we consider the motion of the piece around a loop of 40 spaces. We can model this as a Markov chain on the integers $0, \ldots, 39$ in which the transition kernel adds the result of two six-sided dice to the current state modulo 40 to obtain the new state.

   (a) Implement a piece of R code which simulates this Markov chain.

```
#Increment probabilities
ivals  <- c(2,3,4,5,6,7,8,9,10,11,12)
iprobs <- c(1,2,3,4,5,6,5,4,3,2,1) / 36

rmonopoly <- function(n=1,x0=0) {
#Sample n increments
is <- sample(ivals, size=n, replace=TRUE, prob=iprobs)

x <- (x0 + cumsum(is)) %% 40
}
```

   (b) Run the code for a large number of iterations, say $100,000$, and plot a histogram of the states visited.

   Taking a little care over the location of the bins, noting that X lives in *discrete* space:

```
X <- rmonopoly(1000000)
hist(X,breaks=seq(-0.5,39.5,1), prob=T)
```

   (c) Based on the output of the chain, would you conjecture that there is an invariant distribution for this Markov chain? If so, what?

   This histogram is quite strongly suggestive that the invariant distribution might be uniform and. (Indeed this seems logical: the sum of many independent variables if finite variance will ultimately look very like a normal random variable and will have variance linearly increasing with the number of terms. Taking this value, modulo 40, it will wrap around many times and the resulting density will become increasingly flat.)

   (d) Write the transition kernel down mathematically.

   There are lots of ways of doing this, letting $K_{ij} := \mathbb{P}(X_{t+1} = j | X_t = i)$, we have:

$$K_{ij} = \begin{cases} 1 & j-i \mod 40 = 2 \text{ or } j-i \mod 40 = 12 \\ 2 & j-i \mod 40 = 3 \text{ or } j-i \mod 40 = 11 \\ 3 & j-i \mod 40 = 4 \text{ or } j-i \mod 40 = 10 \\ 4 & j-i \mod 40 = 5 \text{ or } j-i \mod 40 = 9 \\ 5 & j-i \mod 40 = 6 \text{ or } j-i \mod 40 = 8 \\ 6 & j-i \mod 40 = 7 \end{cases}$$

   (e) Show that the Markov kernel you have written down is invariant with respect to any distribution conjectured in part (c).

Let $f_i = 1/40\mathbb{I}_{\{0\ldots,39\}}(i)$ be the putative invariant distribution:

$$\begin{aligned}\sum_i f_i K_{ij} =& \frac{1}{40}\sum_i K_{ij}\\ =&\frac{1}{40}\left(1+2+3+4+5+6+5+4+3+2+1\right)/36 = 1/40 = f_j.\end{aligned}$$

where the second equality follows by noticing that $j - i \mod 40 = k$ for exactly one i for every pair $j, k \in \{0, \ldots, 39\}$.

2. Gibbs Sampling: recall the Poisson changepoint model discussed in lectures, and on p26 of the supporting notes, and think about the following closely related model.

   Observations $y_1, \ldots, y_n$ comprise a sequence of M iid $\mathsf{N}(\mu_1, 1)$ random variables followed by a second sequence of $n - M$ iid $\mathsf{N}(\mu_2, 1)$ random variables. M, μ_1 and μ_2 are unknown.

   The prior distribution over M is a discrete uniform distribution on $\{1, \ldots, n-1\}$ (there is at least one observation of each component). The prior distribution over μ_i $(i = 1, 2)$ is $\mathsf{N}\left(0, 10^2\right)$. The three parameters are treated as being a priori independent.

   (a) Write down the joint density of $y_1, \ldots, y_n, \mu_1, \mu_2$ and M and obtain the posterior distribution of μ_1, μ_2 and M, up to proportionality, in as simple a form as you can.

$$\begin{aligned}&p(y_1,\ldots,y_n,\mu_1,\mu_2,M)\\ =&p(M)p(\mu_1)p(\mu_2)p(y_1,\ldots,y_n|\mu_1,\mu_2,M)\\ =&\frac{1}{n-1}\mathbb{I}_{\{1,\ldots,n-1\}}(M)\frac{1}{\sqrt{200\pi}}\exp\left(-\frac{\mu_1^2}{2\cdot 10^2}\right)\frac{1}{\sqrt{200\pi}}\exp\left(-\frac{\mu_2^2}{2\cdot 10^2}\right)\cdot\\ &\prod_{i=1}^{M}\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(y_i-\mu_1)^2}{2}\right)\prod_{i=M+1}^{n}\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(y_i-\mu_2)^2}{2}\right)\\ \propto&\mathbb{I}_{\{1,\ldots,n-1\}}(M)\exp\left(-\frac{\mu_1^2+\mu_2^2}{2\cdot 10^2}\right)\exp\left(-\sum_{i=1}^{M}\frac{(y_i-\mu_1)^2}{2}-\sum_{i=M+1}^{n}\frac{(y_i-\mu_2)^2}{2}\right)\end{aligned}$$

   (b) Find the "full conditional" distributions of μ_1, μ_2 and M. (i.e. the conditional distributions of each of these variables given all other variables).

$M|y_{1,\dots,n},\mu_1,\mu_2$ has the discrete distribution of $1,\dots,n-1$ with

$$\mathbb{P}(M=m) \propto \exp\left(-\sum_{i=1}^{M}\frac{(y_i-\mu_1)^2}{2} - \sum_{i=M+1}^{n}\frac{(y_i-\mu_2)^2}{2}\right)$$

which can be easily (if not cheaply) sampled from.
While the two mean parameters can be readily seen to be normally distributed (complete the square in μ_1 or μ_2, respectively, in the exponential terms which involve each parameter), and:

$$\begin{aligned}
p(\mu_1|\mu_2,M,y_1,\dots,y_n) &\propto \exp\left(-\frac{\mu_1^2}{2\cdot 10^2}-\right)\exp\left(-\sum_{i=1}^{M}\frac{(y_i-\mu_1)^2}{2}\right)\\
&\propto \exp\left(-\frac{\mu_1^2+100M\mu_1^2-200\mu_1\sum_{i=1}^{M}y_i}{2\cdot 10^2}\right)\\
&\propto \exp\left(-\frac{\mu_1^2-200\mu_1\sum_{i=1}^{M}y_i/(1+100M)}{2\cdot 100/(1+100M)}\right)\\
&\propto \mathsf{N}\left(\mu_1;\sum_{i=1}^{M}100y_i/(1+100M),100/(1+100M)\right)
\end{aligned}$$

and similarly for y_2, but with the remaining $n-M$ observations:

$$p(\mu_2|\mu_1,M,y_1,\dots,y_n) = \mathsf{N}\left(\mu_2;\sum_{i=M+1}^{n}100y_i/(1+100(n-M)),100/(1+100(n-M))\right)$$

(c) Implement a Gibbs sampler which makes use of these full conditional distributions in order to target the posterior distribution identified in part (b).

```
sample.mu <- function(obs,n.obs) {
    mean <- 100 * sum(obs) / (1+100*n.obs)
    var <- 100 / (1+100 * n.obs)
    rnorm(1,mean,sqrt(var))
}

sample.M <- function(obs,n.obs,mu1,mu2) {
    log.p <- c()
    for(i in 1:(n.obs-1)) {
        log.p[i] <- -(sum((obs[1:i]-mu1)^2) + sum((obs[(i+1):n.obs]-mu2)^2)) / 2
    }

    log.p <- log.p - max(log.p)
    p <- exp(log.p)
    p <- p / sum(p)

    sample.int(n=n.obs-1,size=1,replace=TRUE,prob=p)
}

gibbs.ncpm <- function(n.it=10000, y, M = length(y)/2, mu1 = -1, mu2 = +1) {
    n <- length(y)
    Ms <- c()
    mu1s <- c()
    mu2s <- c()

    Ms[1]   <- sample.M(y,n,mu1,mu2)
    mu1s[1] <- sample.mu(y[1:Ms[1]],Ms[1])
    mu2s[1] <- sample.mu(y[(Ms[1]+1):n],n-Ms[1])

    for (i in 2:n.it) {
        Ms[i]   <- sample.M(y,n,mu1s[i-1],mu2s[i-1])
        mu1s[i] <- sample.mu(y[1:Ms[i]],Ms[i])
        mu2s[i] <- sample.mu(y[(Ms[i]+1):n],n-Ms[i])
    }

    list(M = Ms, mu1 = mu1s, mu2=mu2s)
}
```

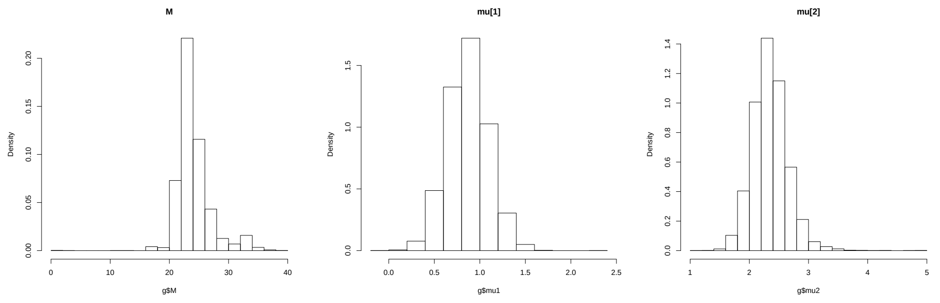
The only potentially surprising feature in the above code is perhaps the simulation of M. The use of logarithms and the subtraction of the maximum is solely for numerical reasons and is generally a good idea – combinations of very large / very small values can otherwise lead to strange behaviour in some circumstances.

(d) Simulate some data from the model for various parameter values and test your Gibbs sampler.

This works reasonably well for most reasonable parameter values. Here's an example:

```
y <- c(rnorm(23, 1,1),rnorm(17,2,1))
g <- gibbs.ncpm(1E4,y,10,2,-4)
```

Which produces the following histograms for M, μ_1 and μ_2, respectively:



(e) How might you extend this algorithm if instead of a changepoint model you had a mixture model in which every observation is drawn from a mixture, i.e.:

$$Y_1, \ldots, Y_n \overset{\text{iid}}{\sim} p\mathsf{N}(\cdot; \mu_1, 1) + (1-p)\mathsf{N}(\cdot; \mu_2, 1)$$

(so the likelihood is $\prod_{i=1}^{n}[p\mathsf{N}(y_i; \mu_1, 1) + (1-p)\mathsf{N}(y_i; \mu_2, 1)]$ in which p, μ_1 and μ_2 are unknown (and M is no longer a parameter of the model)?

Consider the following things:

i. The prior distribution over p.

For Gibbs sampling to work we require conjugacy, a $\mathsf{U}[0,1]$ prior would suffice, but is a special case of the Beta distribution, any member of which would lead to an implementable algorithm.

ii. Any other variables you may need to introduce.

As in the examples considered in lectures, we'd need a latent allocation variable for every observation indicating which of the two components it's treated as coming from in the completed model.

iii. The resulting algorithm.

A standard Gibbs Sampler should work adequately once these prior distributions and latent variables have been specified.

If you have time, implement the resulting algorithm and apply it to some simulated data.