Homework 5: Gibbs Sampling

Harry Bendekgey
Math 153: Bayesian Statistics

April 12, 2018

Consider observing $x_t \sim \text{Poisson}(\lambda)$ from times $t = 1, 2, \ldots, N$. Assume that each x_t is independent. At some point in this time span, say $t = n$, the value of λ switches from some value λ_1 to λ_2. Our goal is to estimate both values of λ_i as well as the time point n at which the switch occurs. We will use a Gibbs sampler to obtain estimates for the joint posterior density of $n, \lambda_1, and \lambda_2$
Bayes rule tells us:

$$f(\lambda_1, \lambda_2, n|x) \propto L(\lambda_1, \lambda_2, n|x) f(\lambda_1) f(\lambda_2) f(n)$$

Let λ_1, λ_2 have gamma priors. We recognize that these priors to be $\lambda_1^{\alpha_1 - 1} e^{-\beta_1 \lambda_1}$ and $\lambda_2^{\alpha_2 - 1} e^{-\beta_2 \lambda_2}$. If we let n's prior be a discrete uniform distribution, it has no kernel. The likelihood is given by:

$$L(\lambda_1, \lambda_2, n|x) = \prod_{t<n} \frac{\lambda_1^{x_t} e^{-\lambda_1}}{x_t!} \cdot \prod_{t \geq n} \frac{\lambda_2^{x_t} e^{-\lambda_2}}{x_t!}$$

Note here that I am assuming that at time interval n, the data point is distribution according to λ_2. Thus n can be interpreted as the first data point from the new distribution. We see that we are taking the factorial of every data point, before or after the switch, and thus can be removed from the kernel. This gives us the posterior's kernel:

$$f(\lambda_1, \lambda_2, n|x) \propto \lambda_1^{\alpha_1 - 1} e^{-\beta_1 \lambda_1} \lambda_2^{\alpha_2 - 1} e^{-\beta_2 \lambda_2} \prod_{t<n} \lambda_1^{x_t} e^{-\lambda_1} \cdot \prod_{t \geq n} \lambda_2^{x_t} e^{-\lambda_2}$$

If we take the marginal with respect to λ_1, we see:

$$f(\lambda_1|\lambda_2, n, x) \propto \lambda_1^{\alpha_1 - 1} e^{-\beta_1 \lambda_1} \prod_{t<n} \lambda_1^{x_t} e^{-\lambda_1} = \lambda_1^{\alpha_1 + \sum_{t<n} x_t - 1} e^{-(\beta_1 + n - 1)\lambda_1}$$

We recognize this as the kernel of a gamma distribution with parameters $\alpha_1 + \sum_{t<n} x_t, \beta_1 + n - 1$. Similarly, we see that λ_2 is distributed according to a gamma distribution with parameters $\alpha_1 + \sum_{t \geq n} x_t, \beta_1 + N - n + 1$. When we look at the marginal with respect to n, we don't see anything we recognize. However, because it is a discrete distribution we can just enumerate the value of the kernel at each value in the support, normalize them and sample from that:

```
gen.n <- function(lambda1, lambda2, x) {
  probs <- c()
  sum.x <- sum(x)
  N <- length(x)
  sum.x.below <- 0
  for (n in 1:N) {
    probs[n] <- exp(-1 * lambda1)^(n-1) * exp(-1 * lambda2)^(N-n+1) *
                lambda1^sum.x.below * lambda2^(sum.x-sum.x.below)
```

```
    sum.x.below <- sum.x.below + x[n]
  }
  return(which.max(rmultinom(1,1,probs)[,1]))
}
```

We also write the following functions to generate values of λ_1 and λ_2:

```
gen.lam1 <- function(n, x) {
  a <- lam1a + sum(x[1:(n-1)])
  b <- lam1b + n - 1
  return(rgamma(1, shape=a, rate=b))
}

gen.lam2 <- function(n, x) {
  a <- lam2a + sum(x[n:(length(x))])
  b <- lam2b + length(x) - n + 1
  return(rgamma(1, shape=a, rate=b))
}
```

Finally, we implement Gibbs sampling, throwing out the first 10% of data points to account for burn-in, and return point estimates for n, λ_1, λ_2 as well as 90% credible intervals for all 3.

```
# start = c(n, lambda1, lambda2)
Gibbs <- function(data, start=c(1,1,1), t=10000) {
  samples <- matrix(start, ncol=3)
  for (i in 1:t) {
    x <- samples[nrow(samples),]
    n <- gen.n(x[2], x[3], data)
    l1 <- gen.lam1(n, data)
    l2 <- gen.lam2(n, data)
    samples <- rbind(samples,c(n, l1, l2))
  }
  samples <- samples[(t/10):t,] #burn-in
  return(list(n.mean=mean(samples[,1]),
              n.credible.interval=quantile(samples[,1],c(0.05,0.95)),
              lambda1.mean=mean(samples[,2]),
              lambda1.credible.interval=quantile(samples[,2],c(0.05,0.95)),
              lambda2.mean=mean(samples[,3]),
              lambda2.credible.interval=quantile(samples[,3],c(0.05,0.95)) ))
}
```

We know this algorithm works better at smaller sample sizes when we have a good candidate distribution. To fascilitate that, let's assume we're using good priors, and generate our lambda values from there. Let's say we have a sample of size 25.

```
# set hyperparameters
lam1a <- 8; lam1b <- 3; lam2a <- 16; lam2b <- 2; N <- 25

# genereate true parameter values
lambda1 <- rgamma(1, shape=lam1a, rate=lam1b)
lambda2 <- rgamma(1, shape=lam2a, rate=lam2b)
```

```
n <- sample(c(1:N),1)

# generate data
data <- rpois(n-1, lambda1)
data <- c(data, rpois(N - n + 1, lambda2))
```

Let's see what the results look like:

```
n; lambda1; lambda2

## [1] 15
## [1] 4.652
## [1] 9.213

data

##  [1]  6  4  4  5  8  2  6  3  5  5  8  4  1  2  9  6  9  9  6 11  8  4  4
## [24]  9 12
```

Finally, let's run the sampler!

```
Gibbs(data)

## $n.mean
## [1] 15.14
##
## $n.credible.interval
##  5% 95%
##  13  17
##
## $lambda1.mean
## [1] 4.288
##
## $lambda1.credible.interval
##    5%   95%
## 3.416 5.222
##
## $lambda2.mean
## [1] 7.839
##
## $lambda2.credible.interval
##    5%   95%
## 6.478 9.282
```

These are pretty good!