贝叶斯统计作业

朱强强

17064001

应用统计学1701

8.1.

Section 8.1.3 includes a function for using Monte Carlo integration to approximate the integral of the unnormalized posterior in the example problem with a binomial likelihood and a histogram prior. Write an R function to approximate the posterior mean of π in this example using Monte Carlo integration. Compare your results with those obtain by numeric integration.

Solution

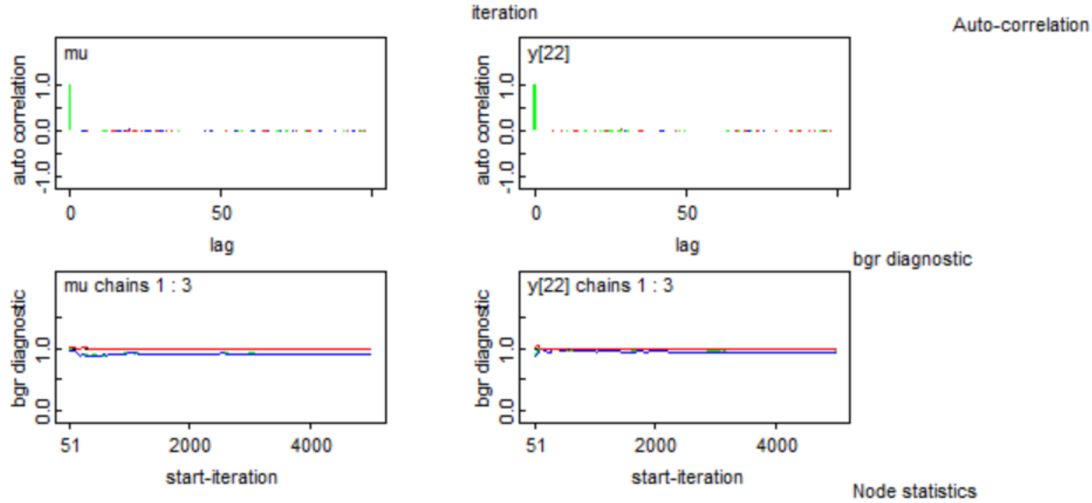
The result computed by R is 0.1493286, which is very close to value 0.1493313 obtained by numeric integration.

```r
## 8.1
unnormpost <- function(pi, pr) {
  like <- pi^7 * (1-pi)^43
  like*pr
}

mcintegral <- 0

for (i in 1:1000) {
  mcinteg <- function() {
    endpoints <- c(0, 0.1, 0.2, 0.3, 0.4)
    prior <- c(2.5, 5.0, 2.0, 0.5)
    nrand <- 100  # number of random points in each interval
    integral <- 0  # initialize variable to accumalate total integral

    for (i in 1:4) {
      mypis <- runif(nrand, endpoints[i], endpoints[i+1])
      heights <- unnormpost(mypis, prior[i])
      integral <- integral + mean(heights) * (endpoints[i+1]-endpoints[i])
    }

    integral

  }

  mcintegral <- mcintegral + mcinteg()

}
```

```
mcintegral <- mcintegral/1000

expec <- function(pi, pr) {
  like <- pi^8 * (1-pi)^43
  like*pr
}

exintegral <- 0

for (i in 1:1000) {
  expecinteg <- function() {
    endpoints <- c(0, 0.1, 0.2, 0.3, 0.4)
    prior <- c(2.5, 5.0, 2.0, 0.5)
    nrand <- 100  # number of random points in each interval
    integral <- 0  # initialize variable to accumalate total integral

    for (i in 1:4) {
      mypis <- runif(nrand, endpoints[i], endpoints[i+1])
      heights <- expec(mypis, prior[i])
      integral <- integral + mean(heights) * (endpoints[i+1]-endpoints[i])
    }

    integral/mcintegral

  }

  exintegral <- exintegral + expecinteg()
}

exintegral/1000
```

8.2. Go through the steps to use OpenBUGS for Model 1 for the fish mercury data. Note that, since y[22] is an unknown quantity in the model, it needs an initial value. The easiest approach is to leave the initial values lists as they are, and then, after loading the initial values for the third chain, to click “Gen inits” to have OpenBUGS generate its own initial values for y[22]. Compare your results to those obtained in Sect. 6.2.8.3.

Solution



Node statistics

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
mu	-2.586	0.1287	7.334E-4	-2.839	-2.586	-2.336	1	30000
y[22]	-2.589	0.6492	0.003652	-3.852	-2.59	-1.318	1	30000

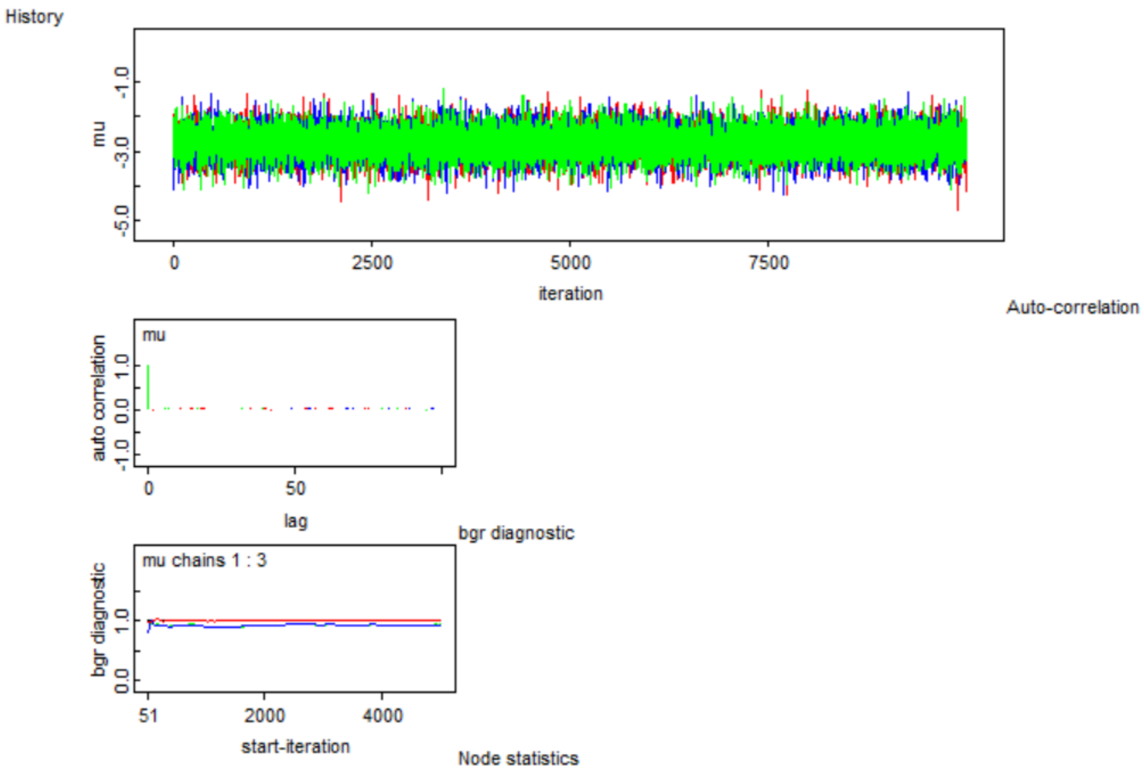
A 95% equal-tail posterior credible set for μ calculated in Sect. 6.2.8.3 is [-2.839030, -2.332970], and the value of μ computed by Monte Carlo method in this problem is -2.596, which means that this result can be accepted.

```
# Model 1
# Assuming data are draws from a normal population with known precision
# Population mean mu is unknown parameter
# We can also estimate the posterior predictive distribution
# by monitoring y[22]


model
{
# likelihood
for (i in 1:N) {
y[i] ~ dnorm(mu, tausq)
}
# priors
mu ~ dnorm(-2.75, 7.5)
}

# data
list(y=c(-2.526, -1.715, -1.427, -2.12, -2.659,
-2.408, -3.219, -1.966,
-2.526, -1.833, -2.813, -1.772, -2.813, -2.526,
-3.219, -2.526,
-2.813, -2.526, -3.507, -2.996, -3.912, NA), N=22,
tausq= 2.5)


# inits for model 1
list(mu = -5)
list(mu = -2.5)
```

```
list(mu = 0)
```

8.3. Go through the steps to use OpenBUGS for Model 3 for the fish mercury data. Compare your results to those obtained in Sect. 7.1.1.

Solution



Node statistics

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
mu	-2.752	0.3783	0.002106	-3.5	-2.751	-2.001	1	30000

A 95% equal-tail posterior credible set for μ calculated in Sect. 7.1.1 is [-2.085963, 2.085963], and the value of μ computed by Monte Carlo method in this problem is -2.752, which means that this result can be accepted.

```
model
{
# likelihood
for (i in 1:N) {
y[i] ~ dnorm( mu, tausq )
}
# priors
tausq0 <- 3 * tausq
mu ~ dnorm( -2.75, tausq0)
tausq ~ dgamma( 13.3, 5.35)
sigmasq <- 1/tausq
}


# Here is a different way to give data to WinBUGS.
# data
```

```

list(N = 22 )
additional data
y[]
-2.526
-1.715
-1.427
-2.12
-2.659
-2.408
-3.219
-1.966
-2.526
-1.833
-2.813
-1.772
-2.813
-2.526
-3.219
-2.526
-2.813
-2.526
-3.507
-2.996
-3.912
NA
END

# inits for model 2
list(mu = 0, tausq = 1)
list(mu = 20, tausq = 100)
list(mu = 40, tausq = 1000)
```

8.4. This problem is a continuation of Problem 6.2. Now use OpenBUGS or WinBUGS to carry out the analysis. You will have to specify the model in terms of the precision.

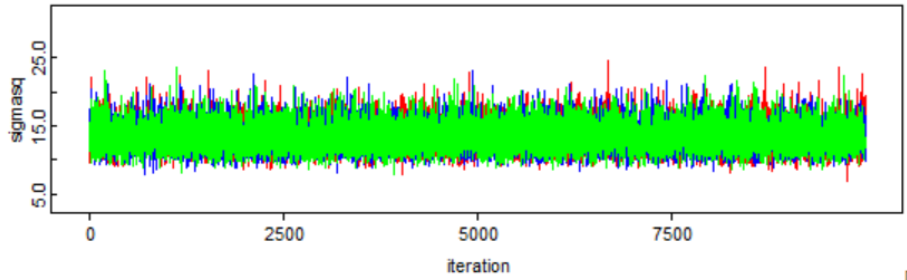
1. What is the conjugate family of prior distributions for a normal precision when the mean is known? (You will then use the same parameters in this prior as you used for the prior on the variance in Problem 6.3.)

   Solution

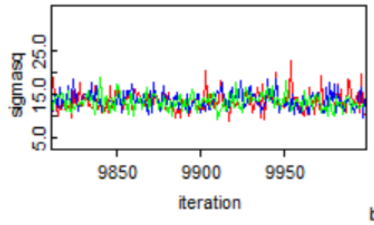
   The conjugate family is inverse gamma.

2. Include the computation of the variance in your OpenBUGS/WinBUGS program.

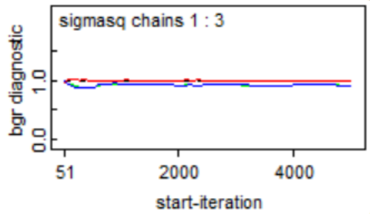
   Solution



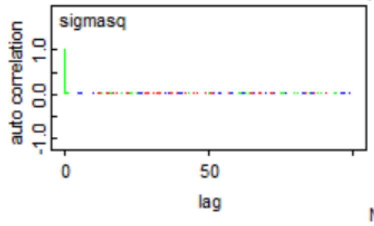
Dynamic trace



bgr diagnostic



Auto-correlation



Node statistics

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
sigmasq	13.37	1.975	0.01074	10.04	13.19	17.73	1	30000

```
# model
# mu is konwn and tausq is unknown

model
{
# likelihood
for (i in 1:N) {
y[i] ~ dnorm(mu, tausq)
}
# priors
tausq ~ dgamma(38, 444)
sigmasq <- 1/tausq
}

# data
list(y=c(46, 58, 40, 47, 47, 53, 43, 48, 50, 55, 49,
50, 52, 56, 49, 54, 51, 50, 52, 50), N=20, mu=51)

# inits for model
list(tausq=1)
list(tausq=50)
list(tausq=100)
```

3. Compare the posterior mean and variance obtained by OpenBUGS/WinBUGS for the variance with what you obtained analytically.

   Solution

   The posterior mean is 13.37, which is almost equal to the value 13.362 calculated in the problem 6.2. And the posterior variance computed by MCMC method is $1.975^2 = 3.9$, which is also significantly close to the result 3.88 calculated before. Thus we can consider that the results obtained by MCMC method can be accepted.