Bayesian HW5

Kerui Cao

11/10/2019

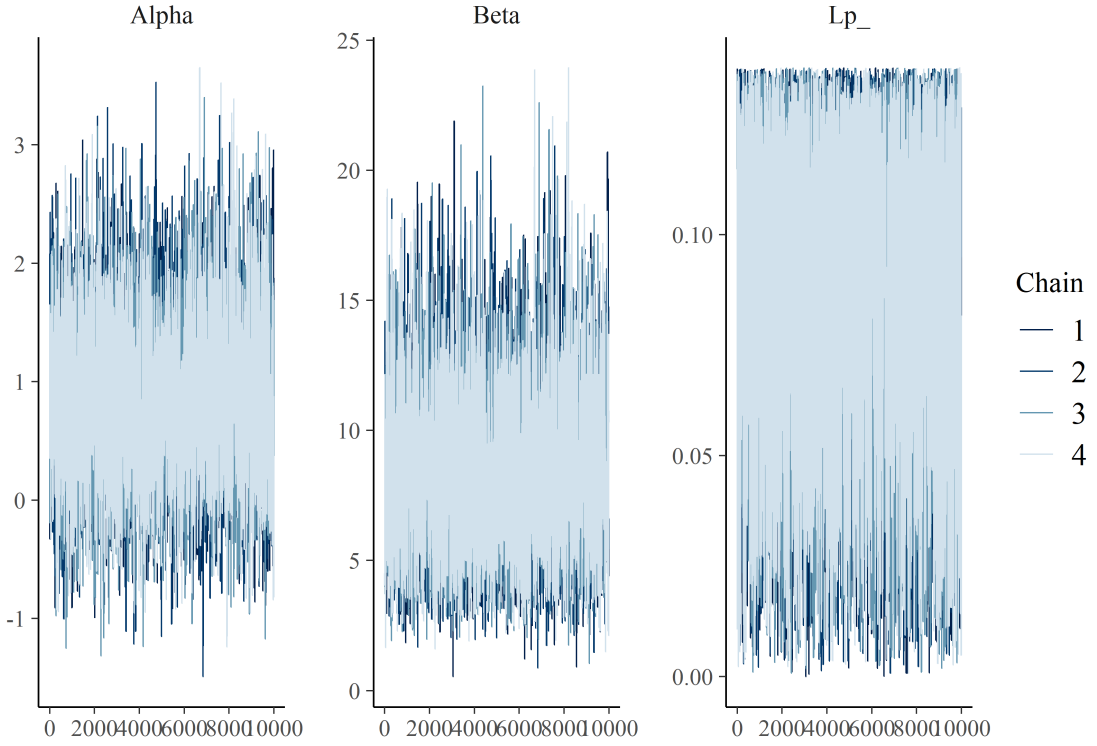
BDA Problem 11.2

Metropolis algorithm: Replicate the computations for the bioassay example of Section 3.7 using the Metropolis algorithm. Be sure to define your starting points and your jumping rule. Compute with log-densities (see page 261). Run the simulations long enough for approximate convergence.

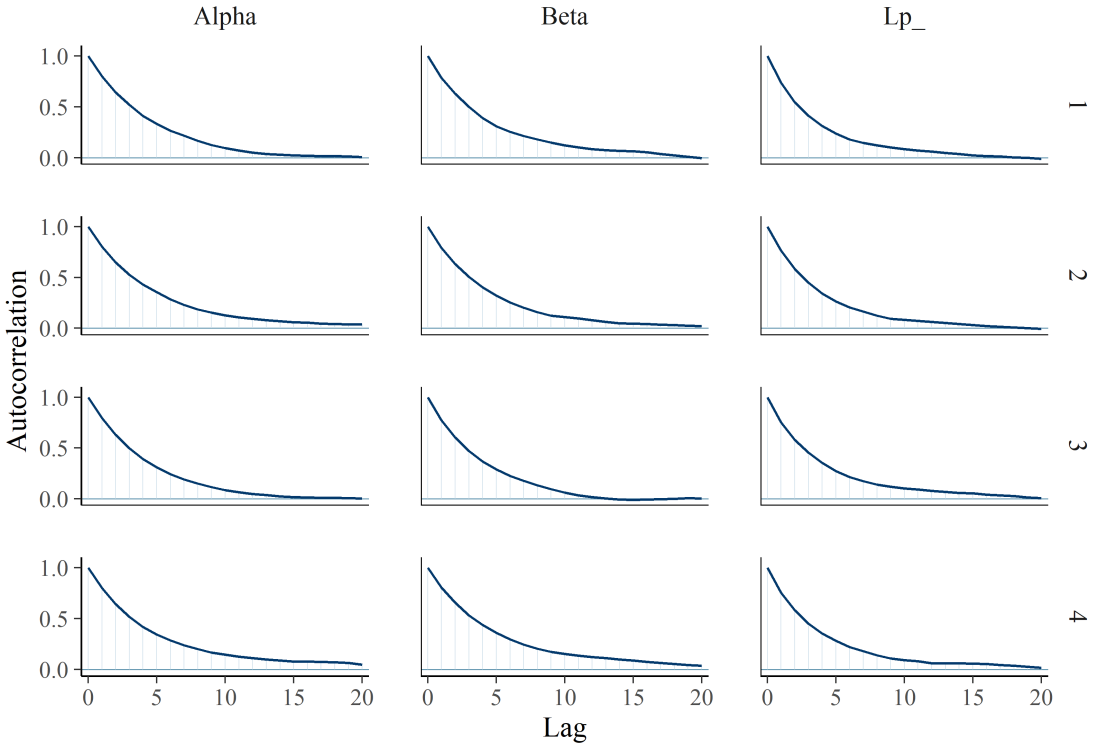
```
a.c = 0.8
b.c = 7.7
a.sd = 1
b.sd = 4.9

mcmc_array <- function (ns, nchains = 1, params) {
  nparams <- length(params)
  array(dim = c(ns, nchains, nparams),
        dimnames = list(iterations = NULL,
                        chains = paste0("chain", 1:nchains),
                        parameters = params))
}

data = data.frame("y" = c(0,1,3,5),
                  "n" = c(5,5,5,5),
                  "x" = c(-0.86,-0.3,-0.05,0.73))

in.lo = function(al,be){
  the = al + be * data$x
  the = invlogit(the)
  return(the)
}

po = function(al,be){
  the = in.lo(al=al,be=be)
  sum.cho = sum(lchoose(n = data$n,k = data$y))
  p2 = sum(data$y*log(the))
  p3 = sum(log(1-the)*(data$n-data$y))
  lopo = exp(sum.cho + p2 + p3)
  return(lopo)
}

nc = 4
ns = 10000

cs = c("Alpha","Beta","Lp_")
sims = mcmc_array(ns,nchains = nc, params = cs)
for(j in 1:nc){
  for(i in 1:ns){
  a.s = rnorm(n = 1,mean = a.c,sd = a.sd)
  b.s = rnorm(n = 1,mean = b.c,sd = b.sd)
```

```
  up = po(al = a.s,be = b.s)*dnorm(a.s,mean = 0.8,sd = 1)*dnorm(b.s,mean = 7.7,sd = 4.9)
  do = po(al = a.c,be = b.c)*dnorm(a.c,mean = 0.8,sd = 1)*dnorm(b.c,mean = 7.7,sd = 4.9)
  r = up/do
  a = runif(1)
  if(r >a){
    a.c = a.s
    b.c = b.s
  }
  post = po(al = a.c,be = b.c)
  sims[i,j,] = c(a.c,b.c,post)
  }
}
```

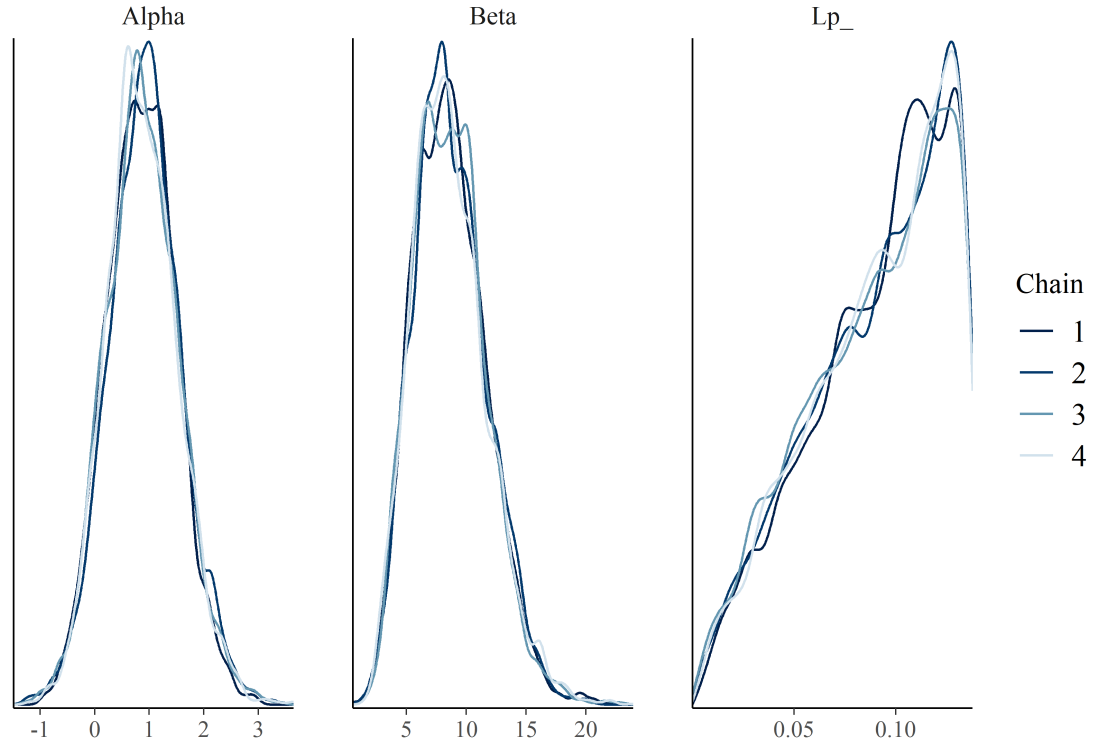
```
mcmc_trace(sims, cs)
```



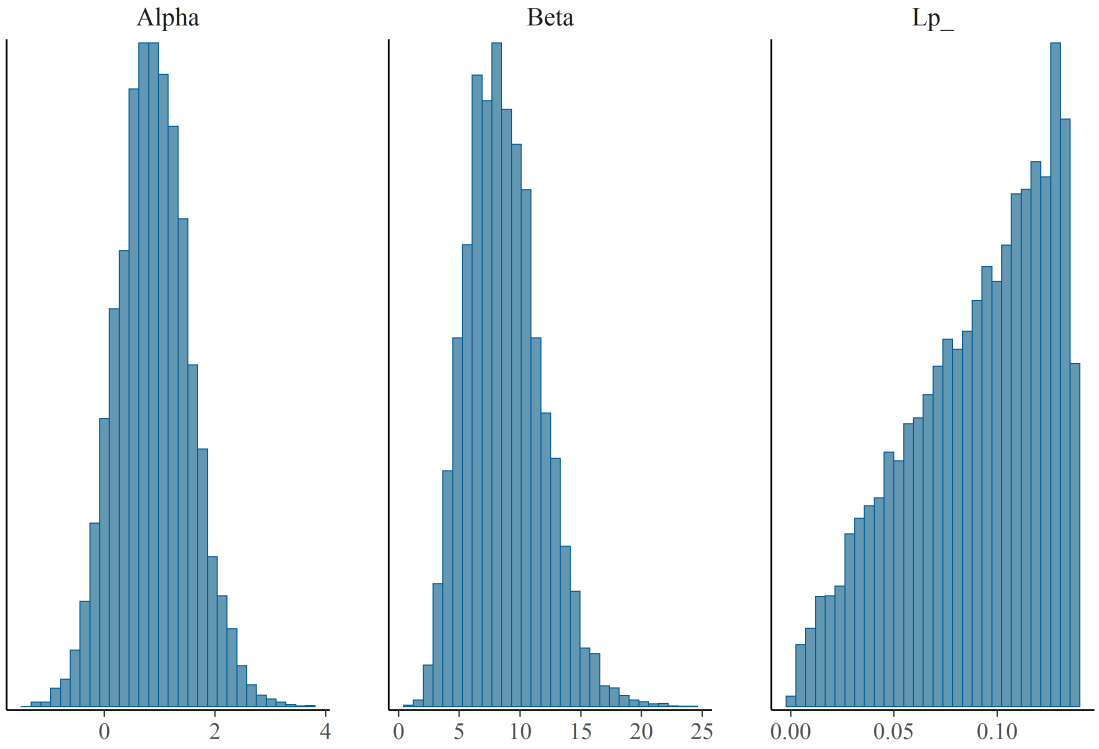

```
mcmc_acf(sims, cs)
```




```
mcmc_dens_overlay(sims, cs)
```



```
mcmc_hist(sims, cs)
```



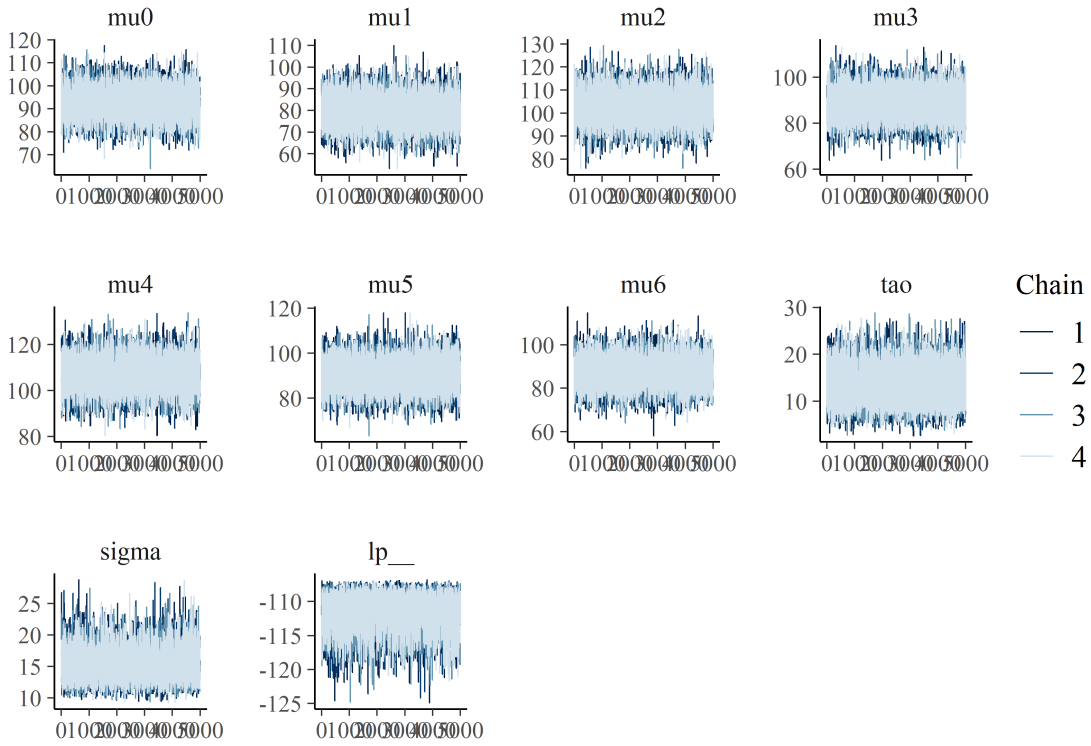
BDA Problem 11.3

```
mo1 = stan_model("machine.stan")
```

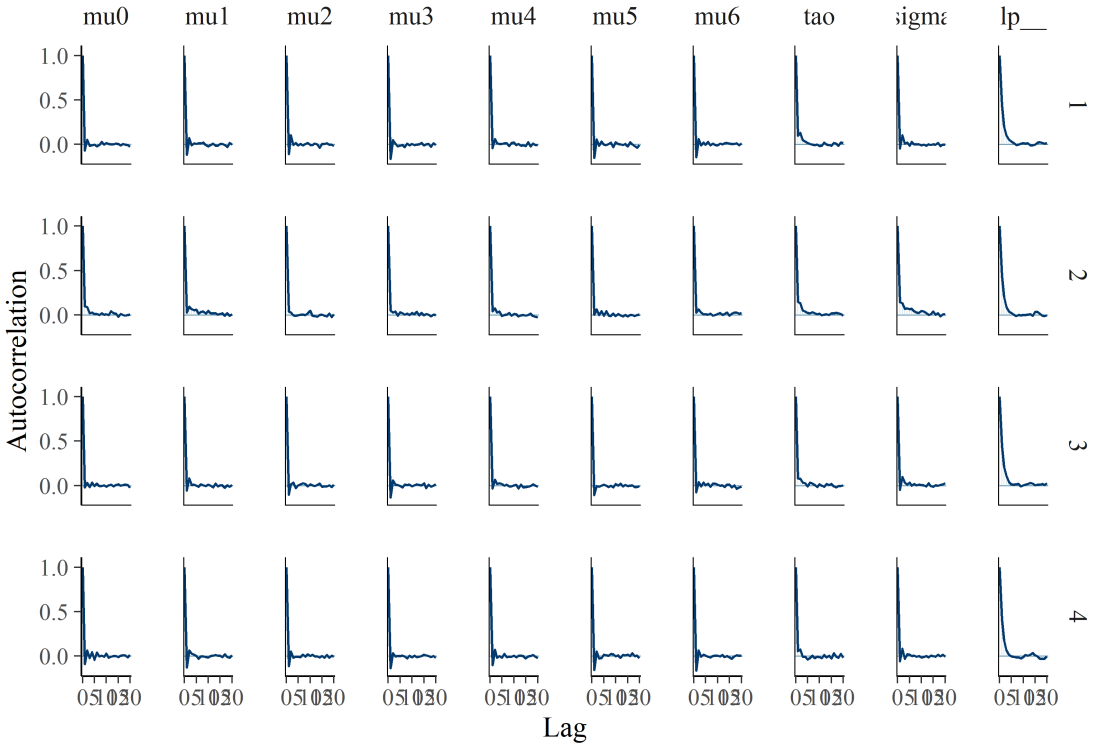
```
monitor(sf)
```

```
## Inference for the input samples (4 chains: each with iter = 10000; warmup = 0):
##
##           Q5    Q50    Q95   Mean  SD  Rhat Bulk_ESS Tail_ESS
## mu0     83.4   93.0  102.5   93.0 5.8     1    17057    14873
## mu1     69.5   79.9   90.7   80.0 6.5     1    16549    10034
## mu2     92.5  103.2  113.3  103.1 6.3     1    21104    13797
## mu3     79.1   89.0   99.1   89.0 6.1     1    21704    13084
## mu4     96.4  107.3  117.9  107.2 6.5     1    17792    12815
## mu5     80.5   90.6  100.7   90.7 6.2     1    22011    13809
## mu6     77.5   87.6   97.6   87.6 6.1     1    19752    13250
## tao      7.6   13.2   19.9   13.4 3.8     1    12405     8199
## sigma   11.9   14.9   19.4   15.2 2.3     1    14983    10809
## lp__  -115.7 -111.0 -108.2 -111.4 2.3     1     7857    11462
##
## For each parameter, Bulk_ESS and Tail_ESS are crude measures of
## effective sample size for bulk and tail quantities respectively (an ESS > 100
## per chain is considered good), and Rhat is the potential scale reduction
## factor on rank normalized split chains (at convergence, Rhat <= 1.05).
```

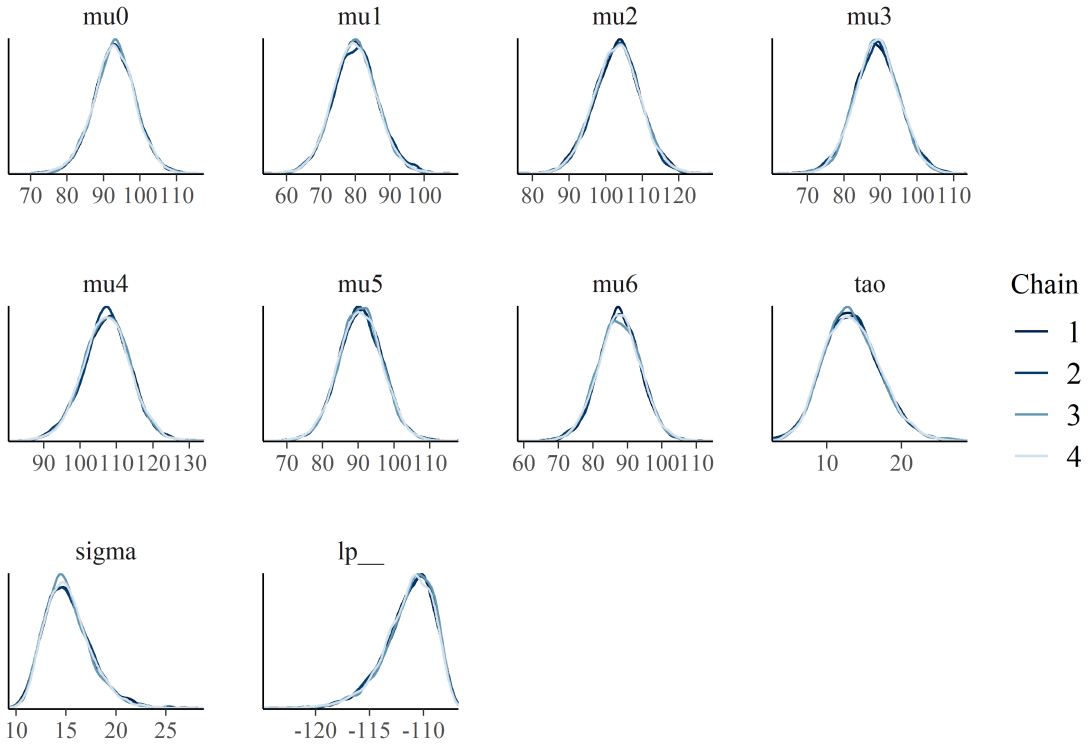
```
sims <- as.array(sf)
params <- c(paste0("mu", 0:6), "tao", "sigma", "lp__")
mcmc_trace(sims, params)
```




```
mcmc_acf(sims, params)
```




```
mcmc_dens_overlay(sims, params)
```



```
mcmc_hist(sims, params)
```