MCMC Examples with R

Christophe Pouzat

November 27, 2014

1 Packages

We are going to use the R mcmc package of Charles Geyer. So if the package is not already installed, you have to do it with, for instance:

```
install.pacakges("mcmc")
```

Once the package is installed, we can load it in our working environment with:

```
library(mcmc)
```

2 A simple example

We define a one dimensional target density:

```
mydens <- function(x) 0.5*dnorm(x)+0.2*dnorm(x,-8,2)+0.3*dnorm(x,15,3)
```

We can plot it with:

```
xx <- seq(-18,25,0.01)
plot(xx,mydens(xx),type="l",xlab="x",ylab="Density",main="Our target density")
```

To use the `metrop` function of the `mcmc` package, we must define a function returning the log of the (not necessarily normalized) posterior density, that's easily done here with:

```
lupost <- function(x) log(0.5*dnorm(x)+0.2*dnorm(x,-8,2)+0.3*dnorm(x,15,3))
```

2.1 Basic MH

Let's assume that we start from x=2 and use a *random walk Metropolis-Hastings* MCMC using, as the name says a random walk proposal (with a standard deviation of 1 per default). The proposal and transition kernels compared to the target looks like (log scale for the ordinate):

```
plot(xx,pmin(1,mydens(xx)/mydens(2)),col=4,
     type="l",xlab = "x", ylab = "density",
     lwd=2,log="y",main="Starting from x=2")
abline(v=2,col="grey30",lwd=2)
lines(xx,dnorm(xx,2),lwd=2,col=2)
lines(xx,mydens(xx),lwd=2)
legend(10,1e-4,c("Target","Proposal","Acceptance","Starting point"),
       col=c("black","red","blue","grey30"),lwd = 2,bty="n")
```

The transition kernel compared to the target looks like:

```
plot(xx,dnorm(xx,2)*pmin(1,mydens(xx)/mydens(2)),col=2,
     type="l",xlab = "x", ylab = "density",lwd=2)
abline(v=2,col="grey30",lwd=2)
lines(xx,mydens(xx),lwd=2)
legend(10,0.35,c("Target","Kernel","Starting point"),
       col=c("black","red","grey30"),lwd=2,lty = c(1,1,1),bty = "n")
```

We next run a realization starting from 0 and using 10^5 steps:

```
set.seed(20110928)
out <- metrop(lupost,0,1e5)
```

We can see the evolution over time of the chain's state with:

```
plot(out$batch[,1],1:1e5,type="l",xlab="x",ylab="idx",
     main="Simulated chain trajectory")
```

Make a histogram and compare to truth:

```
hist(out$batch[,1],breaks=250,probability = TRUE,xlab="x",
     ylab="Density",main="Estimated density compared to target")
lines(xx,mydens(xx),col=2,lwd=2)
```

Keep going for 9×10^5 more iterations:

```
out <- metrop(out,,9e5)
```

Look at the results:

```
hist(out$batch[,1],breaks=500,probability = TRUE,xlab="x",ylab="Density",
     main="Estimated density compared to target")
lines(xx,mydens(xx),col=2,lwd=2)
```

2.2 Parallel tempering / Replica Exchange Method

We re-run the simulations using this time *parallel tempering* with 5 temperatures: 1, 0.7, 0.5, 0.3, 0.1. We can look at the target densities at each temperature with:

```
witch.which <- c(0.1, 0.3, 0.5, 0.7, 1.0)
ncomp <- length(witch.which)
my_cols <- c(4,1,1,1,2)
plot(xx,mydens(xx),type="l",lwd=2,col=2, xlab="x",ylab="Density",
     main="The five target densities")
sapply((ncomp-1):1,
       function(beta_idx) {
           beta <- witch.which[beta_idx]
           yy <- mydens(xx)^beta/integrate(function(x) mydens(x)^beta,
                                           -50,50,subdivisions = 1000)$value
           lines(xx,yy,lwd=2,col=my_cols[beta_idx])
       })
```

We then run the chains:

```r
neighbors <- matrix(FALSE, ncomp, ncomp)
neighbors[row(neighbors) == col(neighbors) + 1] <- TRUE
neighbors[row(neighbors) == col(neighbors) - 1] <- TRUE
ludfun <- function(state) {
    stopifnot(is.numeric(state))
    stopifnot(length(state) == 2)
    icomp <- state[1]
    stopifnot(icomp == as.integer(icomp))
    stopifnot(1 <= icomp && icomp <= ncomp)
    x <- state[-1]
    bnd <- witch.which[icomp]
    return(bnd*(log(0.5*dnorm(x)+0.2*dnorm(x,-8,2)+0.3*dnorm(x,15,3))))
}
thetas <- matrix(0, ncomp, 1)
outREM <- temper(ludfun, initial = thetas, neighbors = neighbors, nbatch = 20000,
         blen = 1, scale = 5, parallel = TRUE, debug = TRUE)
```

Look at the results and compare it to what we get with 100000 steps (that is, the same computational cost) without tempering:

```r
hist(outREM$batch[,5,1],breaks=100,probability = TRUE,xlab="x",
     ylab="Density",main="Estimated densities compared to target")
out_hist <- hist(out$batch[1:100000,1],breaks=100,plot=FALSE)
lines(xx,mydens(xx),col=2,lwd=2)
lines(out_hist$mids,out_hist$density,col=4,lwd=2)
legend(10,0.15,c("Target","Est. Tempering","Est. No Tempering"),
       col=c(2,1,4),lwd=c(2,1,2),bty="n")
```

We can compare the trajectory of the chain's realization without tempering, with the one of the lowest temperature replica with tempering:

```r
layout(matrix(1:2,nc=2))
plot(out$batch[1:20000],1:20000,xlim=c(-18,25),
     xlab="x",ylab="idx",type="l",main="No tempering")
plot(outREM$batch[1:20000,5,1],1:20000,xlim=c(-18,25),
     xlab="x",ylab="idx",type="l",main="Tempering")
```