Stat 8931 Spin Glass Homework Solution, Part II

Charles J. Geyer

December 14, 2005

1 Setup

Read betas.

```
> foo <- try(scan("betas.txt"))
> if (inherits(foo, "try-error")) {
+     write(c(br, bd), file = "betas.txt")
+     foo <- scan("betas.txt")
+ }
> n <- sqrt(length(foo)/2)
> br <- matrix(foo[1:n^2], n, n)
> bd <- matrix(foo[n^2 + 1:n^2], n, n)
```

Set tau.

```
> tau <- 0.2
```

Setup.

```
> i <- matrix(seq(1, n^2), n, n)
> ir <- cbind(i[, -1], i[, 1])
> il <- cbind(i[, n], i[, -n])
> id <- rbind(i[-1, ], i[1, ])
> iu <- rbind(i[n, ], i[-n, ])
> x <- matrix(1, n, n)

> co <- (i + (1 - n%%2) * col(i))%%2
> ico0 <- i[co == 0]
> ico1 <- i[co == 1]
```

Now we want to do a parallel tempering setup. For a start we need the whole unnormalized log density.

```
> logh <- function(x, tau) sum((x * x[ir] * br + x * x[id] * bd)/tau)
```

Then we need to decide how how many "helper" chains we want. We'll start with one. And we need to decide what temperature to use for the helper.

```
> x.help <- x
> tau.help <- 0.3
```

Define update.

```
> block.gibbs <- function(x, tau) {
+     foo <- x[ir] * br + x[id] * bd + x[il] * br[il] + x[iu] *
+         br[iu]
+     foo <- foo/tau
+     p <- 1/(1 + exp(-2 * foo))
+     x[ico0] <- as.numeric(runif(n^2/2) < p[ico0]) * 2 - 1
+     foo <- x[ir] * br + x[id] * bd + x[il] * br[il] + x[iu] *
+         br[iu]
+     foo <- foo/tau
+     p <- 1/(1 + exp(-2 * foo))
+     x[ico1] <- as.numeric(runif(n^2/2) < p[ico1]) * 2 - 1
+     return(x)
+ }
```

Set random seed (if we always want the same results, otherwise omit).

```
> set.seed(42)
```

2 Run One

```
> nbatch <- 100
> blen <- 1000
> accept <- 0
```

```
> batch <- matrix(NA, nbatch, 3 * n^2)
> for (ibatch in 1:nbatch) {
+     xbatch <- rep(0, 3 * n^2)
+     for (iiter in 1:blen) {
+         x <- block.gibbs(x, tau)
+         x.help <- block.gibbs(x.help, tau.help)
+         r <- logh(x.help, tau) + logh(x, tau.help) - logh(x,
+             tau) - logh(x.help, tau.help)
+         if (r > 0 || runif(1) < exp(r)) {
+             foo <- x.help
+             x.help <- x
+             x <- foo
+             accept <- accept + 1
+         }
+         xbatch <- xbatch + as.numeric(c(x, x * x[ir], x * x[id]))
+     }
+     batch[ibatch, ] <- xbatch/blen
+ }
> accept <- accept/(nbatch * blen)
> mu <- apply(batch, 2, mean)
> mcse <- apply(batch, 2, sd)/sqrt(nbatch)
> xmu <- matrix(mu[1:n^2], n, n)
> xmcse <- matrix(mcse[1:n^2], n, n)
> xxrmu <- matrix(mu[n^2 + 1:n^2], n, n)
> xxrmcse <- matrix(mcse[n^2 + 1:n^2], n, n)
> xxdmu <- matrix(mu[2 * n^2 + 1:n^2], n, n)
> xxdmcse <- matrix(mcse[2 * n^2 + 1:n^2], n, n)

> accept

[1] 0.29255

> round(xmu, 3)

       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,]  0.080  0.080  0.079 -0.079  0.079  0.080
[2,]  0.085  0.078 -0.079  0.079  0.079  0.093
[3,]  0.081 -0.068 -0.069  0.080 -0.092  0.093
[4,] -0.082 -0.069 -0.004  0.093 -0.093  0.093
[5,]  0.093  0.083  0.081  0.079  0.093  0.103
[6,]  0.103  0.079  0.079 -0.079  0.102 -0.103

> round(xmcse, 3)

      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
[1,] 0.078 0.078 0.078 0.078 0.078 0.077
[2,] 0.072 0.077 0.078 0.078 0.078 0.072
[3,] 0.069 0.067 0.068 0.077 0.071 0.072
[4,] 0.067 0.068 0.017 0.072 0.072 0.072
```

```
[5,] 0.071 0.075 0.076 0.078 0.072 0.073
[6,] 0.073 0.078 0.078 0.078 0.073 0.073
```

```
> round(xxrmu, 3)
```

```
       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,]  0.999  0.997 -0.998 -1.000  0.995  0.993
[2,]  0.853 -0.983 -0.998  1.000  0.755  0.864
[3,] -0.793  0.950 -0.860 -0.820 -0.940  0.842
[4,]  0.863 -0.077  0.174 -0.999 -1.000 -0.823
[5,]  0.895  0.968  0.973  0.743  0.950  0.902
[6,]  0.778  1.000 -1.000 -0.778 -0.997 -1.000
```

```
> round(xxrmcse, 3)
```

```
      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
[1,] 0.000 0.001 0.000 0.000 0.001 0.001
[2,] 0.014 0.001 0.000 0.000 0.030 0.015
[3,] 0.010 0.001 0.005 0.022 0.007 0.014
[4,] 0.013 0.009 0.017 0.000 0.000 0.010
[5,] 0.012 0.004 0.003 0.031 0.009 0.010
[6,] 0.029 0.000 0.000 0.029 0.001 0.000
```

```
> round(xxdmu, 3)
```

```
       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,]  0.869  0.985 -1.000 -1.000  1.000  0.760
[2,]  0.695 -0.854  0.856  0.987 -0.807  0.990
[3,] -0.743  0.950 -0.076  0.758  0.939  0.996
[4,] -0.800 -0.813 -0.065  0.743 -0.999  0.951
[5,]  0.901  0.942  0.974 -0.999  0.954 -0.999
[6,]  0.778  1.000  0.996  0.998  0.780 -0.786
```

```
> round(xxdmcse, 3)
```

```
      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
[1,] 0.014 0.001 0.000 0.000 0.000 0.029
[2,] 0.029 0.005 0.005 0.002 0.024 0.001
[3,] 0.019 0.001 0.009 0.029 0.007 0.000
[4,] 0.011 0.009 0.013 0.031 0.000 0.009
[5,] 0.011 0.007 0.003 0.000 0.009 0.000
[6,] 0.029 0.000 0.001 0.001 0.028 0.028
```

We happened to luck out with the helper temperature, our first guess giving an acceptance rate in the right range (at least the range specified by the assignment, no theory is available that says what acceptance rate is optimal).

The maximum of all the MCSE is 0.0778, so (assuming the MCSE were correct, which they are not) we only need to run about 77.8^2 times longer, that is `nbatch * blen` equal to

```
> ntodo <- nbatch * blen * (max(xmcse, xxrmcse, xxdmcse)/0.001)^2
> ntodo

[1] 605659560

> ntodo.exp <- floor(log10(ntodo))
> ntodo.frac <- round(ntodo/10^ntodo.exp, 1)
```

`nbatch * blen` equal to 6.1×10^8, divided in any way so the batches are long enough.

Examination of the autocorrelation plots for these shows that `blen` should be at least 20 times longer.

3 Run Two

```
> blen <- 20 * blen
> accept <- 0
> batch <- matrix(NA, nbatch, 3 * n^2)
> for (ibatch in 1:nbatch) {
+     xbatch <- rep(0, 3 * n^2)
+     for (iiter in 1:blen) {
+         x <- block.gibbs(x, tau)
+         x.help <- block.gibbs(x.help, tau.help)
+         r <- logh(x.help, tau) + logh(x, tau.help) - logh(x,
+             tau) - logh(x.help, tau.help)
+         if (r > 0 || runif(1) < exp(r)) {
+             foo <- x.help
+             x.help <- x
+             x <- foo
+             accept <- accept + 1
+         }
+         xbatch <- xbatch + as.numeric(c(x, x * x[ir], x * x[id]))
+     }
+     batch[ibatch, ] <- xbatch/blen
+ }
> accept <- accept/(nbatch * blen)
> mu <- apply(batch, 2, mean)
> mcse <- apply(batch, 2, sd)/sqrt(nbatch)
> xmu <- matrix(mu[1:n^2], n, n)
> xmcse <- matrix(mcse[1:n^2], n, n)
> xxrmu <- matrix(mu[n^2 + 1:n^2], n, n)
> xxrmcse <- matrix(mcse[n^2 + 1:n^2], n, n)
> xxdmu <- matrix(mu[2 * n^2 + 1:n^2], n, n)
> xxdmcse <- matrix(mcse[2 * n^2 + 1:n^2], n, n)
> accept

[1] 0.3260555

> round(xmu, 3)

       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,] -0.028 -0.028 -0.028  0.028 -0.028 -0.028
[2,] -0.025 -0.028  0.028 -0.028 -0.028 -0.021
[3,] -0.023  0.024  0.025 -0.028  0.023 -0.021
[4,]  0.023  0.025 -0.001 -0.021  0.021 -0.021
[5,] -0.024 -0.027 -0.027 -0.028 -0.021 -0.022
[6,] -0.022 -0.028 -0.028  0.028 -0.022  0.022

> round(xmcse, 3)

      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
[1,] 0.057 0.057 0.057 0.057 0.057 0.057
[2,] 0.055 0.056 0.057 0.057 0.057 0.054
[3,] 0.052 0.048 0.049 0.057 0.055 0.054
[4,] 0.050 0.049 0.007 0.054 0.054 0.054
[5,] 0.054 0.056 0.056 0.057 0.054 0.055
[6,] 0.055 0.057 0.057 0.057 0.055 0.055

> round(xxrmu, 3)

       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,]  0.999  0.997 -0.998 -1.000  0.996  0.995
[2,]  0.883 -0.983 -0.998  1.000  0.817  0.896
```

```
[3,] -0.810  0.953 -0.856 -0.866 -0.954  0.867
[4,]  0.884 -0.088  0.129 -0.999 -1.000 -0.843
[5,]  0.915  0.973  0.979  0.808  0.956  0.922
[6,]  0.832  1.000 -1.000 -0.832 -0.996 -1.000
```

```
> round(xxrmcse, 3)
```

```
      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
[1,] 0.000 0.000 0.000 0.000 0.000 0.000
[2,] 0.007 0.000 0.000 0.000 0.015 0.007
[3,] 0.005 0.001 0.002 0.011 0.004 0.007
[4,] 0.006 0.004 0.008 0.000 0.000 0.005
[5,] 0.006 0.002 0.001 0.015 0.002 0.005
[6,] 0.015 0.000 0.000 0.015 0.000 0.000
```

```
> round(xxdmu, 3)
```

```
       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,]  0.898  0.985 -1.000 -1.000  1.000  0.821
[2,]  0.756 -0.852  0.853  0.991 -0.857  0.993
[3,] -0.773  0.952 -0.088  0.819  0.954  0.997
[4,] -0.815 -0.815 -0.052  0.809 -0.998  0.956
[5,]  0.921  0.952  0.979 -0.999  0.960 -0.999
[6,]  0.832  1.000  0.997  0.999  0.834 -0.837
```

```
> round(xxdmcse, 3)
```

```
      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
[1,] 0.007 0.000 0.000 0.000 0.000 0.014
[2,] 0.015 0.002 0.002 0.001 0.012 0.001
[3,] 0.009 0.001 0.004 0.015 0.004 0.000
[4,] 0.005 0.004 0.006 0.015 0.000 0.002
[5,] 0.005 0.003 0.001 0.000 0.002 0.000
[6,] 0.015 0.000 0.000 0.000 0.015 0.015
```

The maximum of all the MCSE is 0.0571, so we only need to run about 57.1^2 times longer, that is `nbatch * blen` equal to

```
> ntodo <- nbatch * blen * (max(xmcse, xxrmcse, xxdmcse)/0.001)^2
> ntodo
```

```
[1] 6525625010
```

```
> ntodo.exp <- floor(log10(ntodo))
> ntodo.frac <- round(ntodo/10^ntodo.exp, 1)
```

`nbatch * blen` equal to 6.5×10^9, divided in any way so the batches are long enough.

4 Run Three

```
> blen <- 2 * blen
> accept <- 0
> batch <- matrix(NA, nbatch, 3 * n^2)
> for (ibatch in 1:nbatch) {
+     xbatch <- rep(0, 3 * n^2)
+     for (iiter in 1:blen) {
+         x <- block.gibbs(x, tau)
+         x.help <- block.gibbs(x.help, tau.help)
+         r <- logh(x.help, tau) + logh(x, tau.help) - logh(x,
+             tau) - logh(x.help, tau.help)
+         if (r > 0 || runif(1) < exp(r)) {
+             foo <- x.help
+             x.help <- x
+             x <- foo
+             accept <- accept + 1
+         }
+         xbatch <- xbatch + as.numeric(c(x, x * x[ir], x * x[id]))
+     }
+     batch[ibatch, ] <- xbatch/blen
+ }
> accept <- accept/(nbatch * blen)
> mu <- apply(batch, 2, mean)
> mcse <- apply(batch, 2, sd)/sqrt(nbatch)
> xmu <- matrix(mu[1:n^2], n, n)
> xmcse <- matrix(mcse[1:n^2], n, n)
> xxrmu <- matrix(mu[n^2 + 1:n^2], n, n)
> xxrmcse <- matrix(mcse[n^2 + 1:n^2], n, n)
> xxdmu <- matrix(mu[2 * n^2 + 1:n^2], n, n)
> xxdmcse <- matrix(mcse[2 * n^2 + 1:n^2], n, n)
> accept
```

```
[1] 0.3320148
```

```
> round(xmu, 3)

       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,]  0.050  0.050  0.050 -0.050  0.050  0.050
[2,]  0.049  0.049 -0.050  0.050  0.050  0.049
[3,]  0.047 -0.042 -0.043  0.050 -0.049  0.049
[4,] -0.045 -0.043  0.000  0.049 -0.049  0.049
[5,]  0.047  0.049  0.049  0.050  0.049  0.047
[6,]  0.047  0.050  0.050 -0.050  0.047 -0.047

> round(xmcse, 3)

      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
[1,] 0.044 0.044 0.044 0.044 0.044 0.044
[2,] 0.042 0.043 0.044 0.044 0.044 0.042
[3,] 0.040 0.037 0.038 0.043 0.042 0.042
[4,] 0.039 0.038 0.005 0.042 0.042 0.042
[5,] 0.042 0.043 0.043 0.044 0.042 0.043
[6,] 0.043 0.044 0.044 0.044 0.043 0.043

> round(xxrmu, 3)

       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,]  0.999  0.997 -0.998 -1.000  0.996  0.995
[2,]  0.879 -0.983 -0.998  1.000  0.809  0.892
[3,] -0.809  0.953 -0.855 -0.860 -0.952  0.864
[4,]  0.880 -0.084  0.131 -0.999 -1.000 -0.842
[5,]  0.912  0.971  0.978  0.799  0.956  0.919
[6,]  0.825  1.000 -1.000 -0.825 -0.996 -1.000

> round(xxrmcse, 3)

      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
[1,] 0.000 0.000 0.000 0.000 0.000 0.000
[2,] 0.006 0.000 0.000 0.000 0.011 0.005
[3,] 0.003 0.000 0.002 0.008 0.003 0.005
[4,] 0.005 0.003 0.006 0.000 0.000 0.004
[5,] 0.005 0.002 0.001 0.012 0.002 0.004
[6,] 0.012 0.000 0.000 0.012 0.000 0.000

> round(xxdmu, 3)
```

```
       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
[1,]  0.894  0.985 -1.000 -1.000  1.000  0.813
[2,]  0.748 -0.851  0.851  0.991 -0.851  0.992
[3,] -0.767  0.953 -0.084  0.811  0.952  0.997
[4,] -0.813 -0.813 -0.058  0.800 -0.998  0.956
[5,]  0.918  0.950  0.979 -0.999  0.960 -0.999
[6,]  0.825  0.999  0.997  0.998  0.827 -0.830
```

```
> round(xxdmcse, 3)
```

```
      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
[1,] 0.006 0.000 0.000 0.000 0.000 0.011
[2,] 0.011 0.001 0.002 0.001 0.009 0.000
[3,] 0.007 0.000 0.003 0.011 0.003 0.000
[4,] 0.004 0.003 0.005 0.012 0.000 0.002
[5,] 0.004 0.003 0.001 0.000 0.001 0.000
[6,] 0.012 0.000 0.000 0.000 0.012 0.011
```

The maximum of all the MCSE is 0.0437, so we only need to run about 43.7^2 times longer, that is `nbatch * blen` equal to

```
> ntodo <- nbatch * blen * (max(xmcse, xxrmcse, xxdmcse)/0.001)^2
> ntodo
```

```
[1] 7627360641
```

```
> ntodo.exp <- floor(log10(ntodo))
> ntodo.frac <- round(ntodo/10^ntodo.exp, 1)
```

`nbatch * blen` equal to 7.6×10^9, divided in any way so the batches are long enough.