Applied Bayesian Analysis : NCSU ST 540

Midterm2

Bruce Campbell

Test section - VAR(1) in JAGS

This section is a test section where we generate and fit a vector autoregressive model - $VAR(1) \in \mathbf{R}^6$ given by

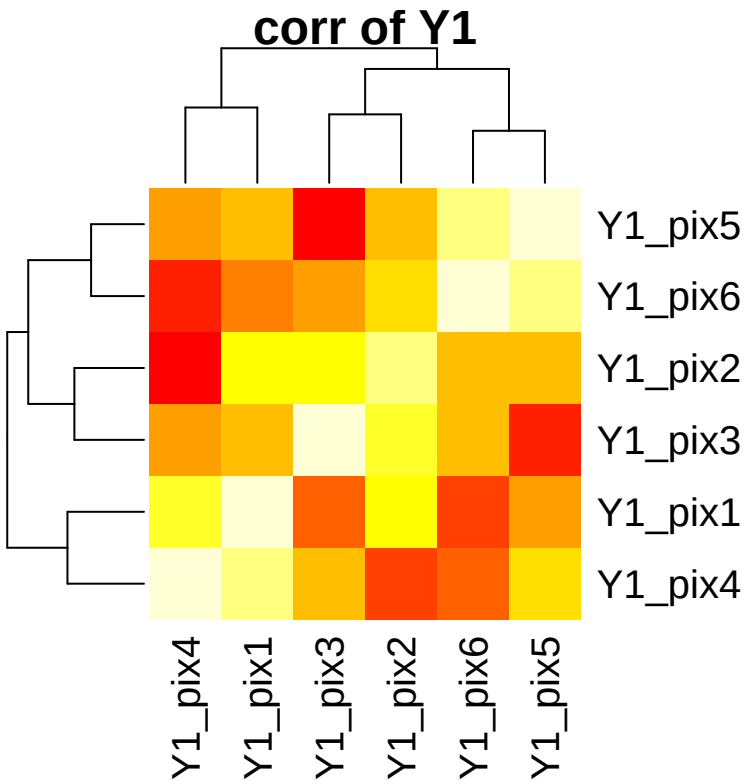
$$y_t = \nu + \rho * y_{t-1} + \epsilon$$

$$\epsilon \sim N(0, \Sigma)$$

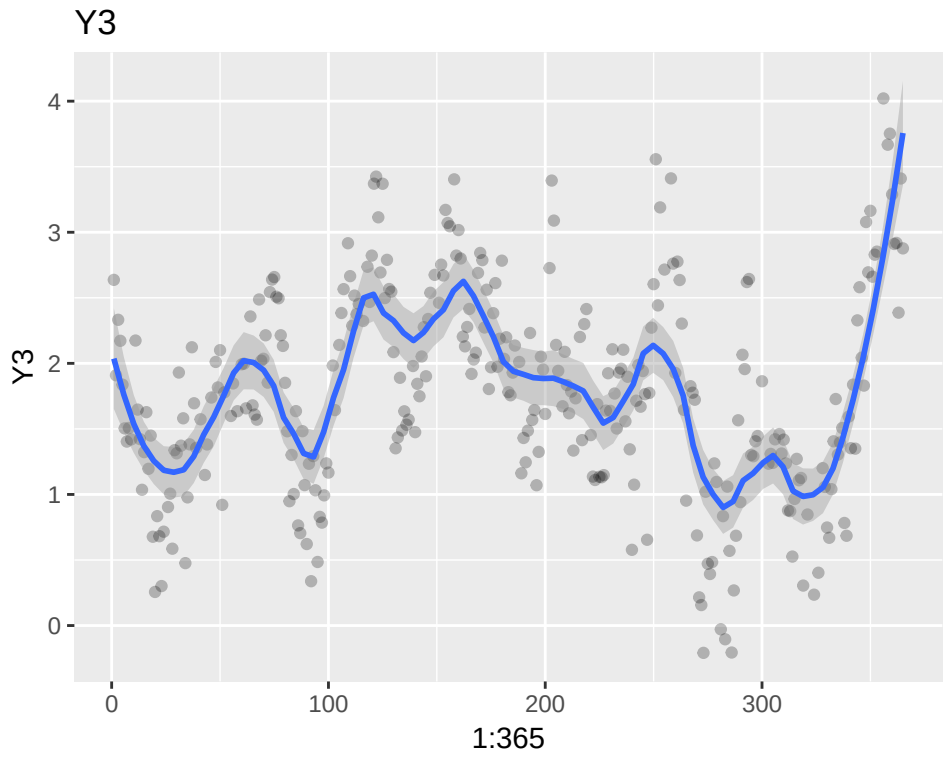
We use the $y1$ data to calculate a NaN firendly sample covariance and then we find the nearest positive semidefinite matrix to use to generate data for the model.

Notes - imputation is not working for this model - this section seeks to find the parameters that best explain the data

```
library(rjags)
library(coda)
library(modeest)
library(MASS)
load("E2.RData")

nSamples <- 10000
n.chains <- 2

cor.y1 <- cor(Y1, use = "pairwise.complete.obs")
cov.y1 <- cov(Y1, use = "pairwise.complete.obs")
heatmap(cor.y1, main = "corr of Y1")
```



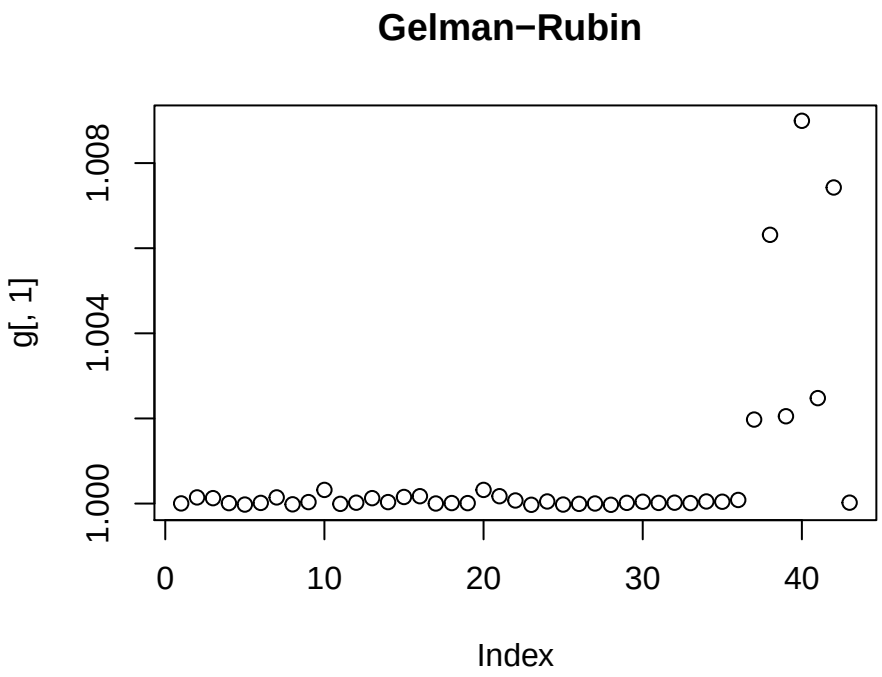

```
ggplot(data.frame(Y3 = Y3), aes(x = 1:365,
    y = Y3)) + geom_point(alpha = 0.25) +
    geom_smooth(method = "loess", span = 0.22) +
    ggtitle("Y3")
```



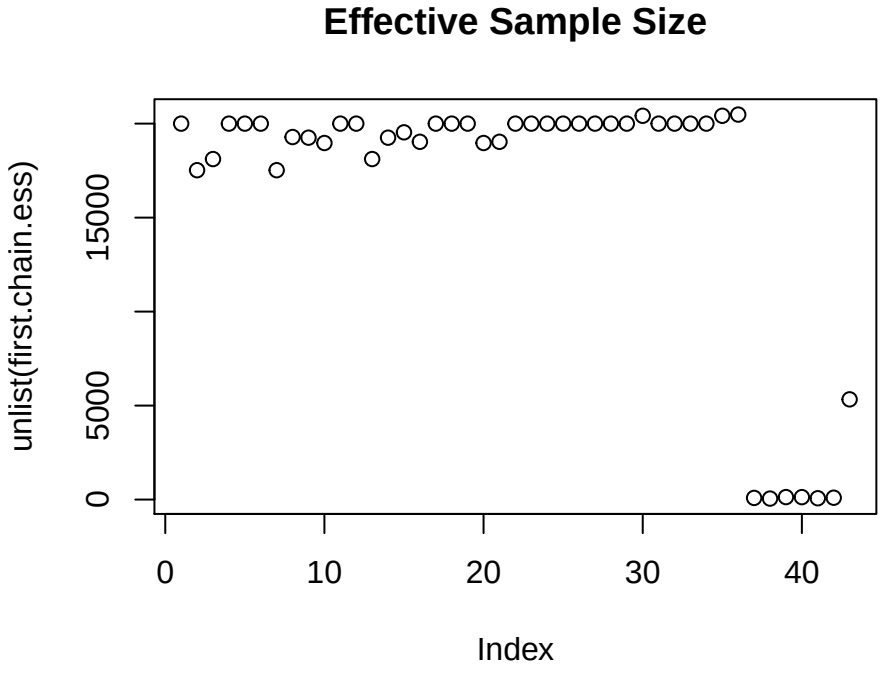
```r
N <- nrow(Y1)
p = 6
Y1.scaled <- scale(Y1)
# Try to simulate using the corr
#install.packages("Matrix")
library("Matrix")

sig <- nearPD(cov.y1)
N = 365
Sigma = sig$mat
rho = .6
nu = matrix(rep(.1,6), p, 1)
y = matrix(NA, N,p)
y[1,] = nu
for(t in 2:N)
{
  y[t,] = mvrnorm(1, nu + rho * y[t-1,], Sigma)
}

# Jags code to fit the model to the simulated data
model_code = '
model
{
  # Likelihood
  for (t in 2:N)
  {
    y[t, ] ~ dmnorm(mu[t, ], precisionAR)
    mu[t, 1:p] <- nu + rho * y[t-1,]
  }
  precisionAR ~ dwish(I, p+1)
  Sigma <- inverse(precisionAR)

  # Priors
  rho ~ dunif(-1, 1)
  for(i in 1:p)
  {
    nu[i] ~ dnorm(0, 0.01)
  }
}
'
# Set up the data
model_data = list(N = N, p = p, y = y, I = diag(p))
# Choose the parameters to watch
model_parameters =  c("Sigma","nu", "rho" )

model <- jags.model(textConnection(model_code),data = model_data,n.chains = n.chains)#Compile
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 364
##    Unobserved stochastic nodes: 8
##    Total graph size: 1153
##
## Initializing model
```

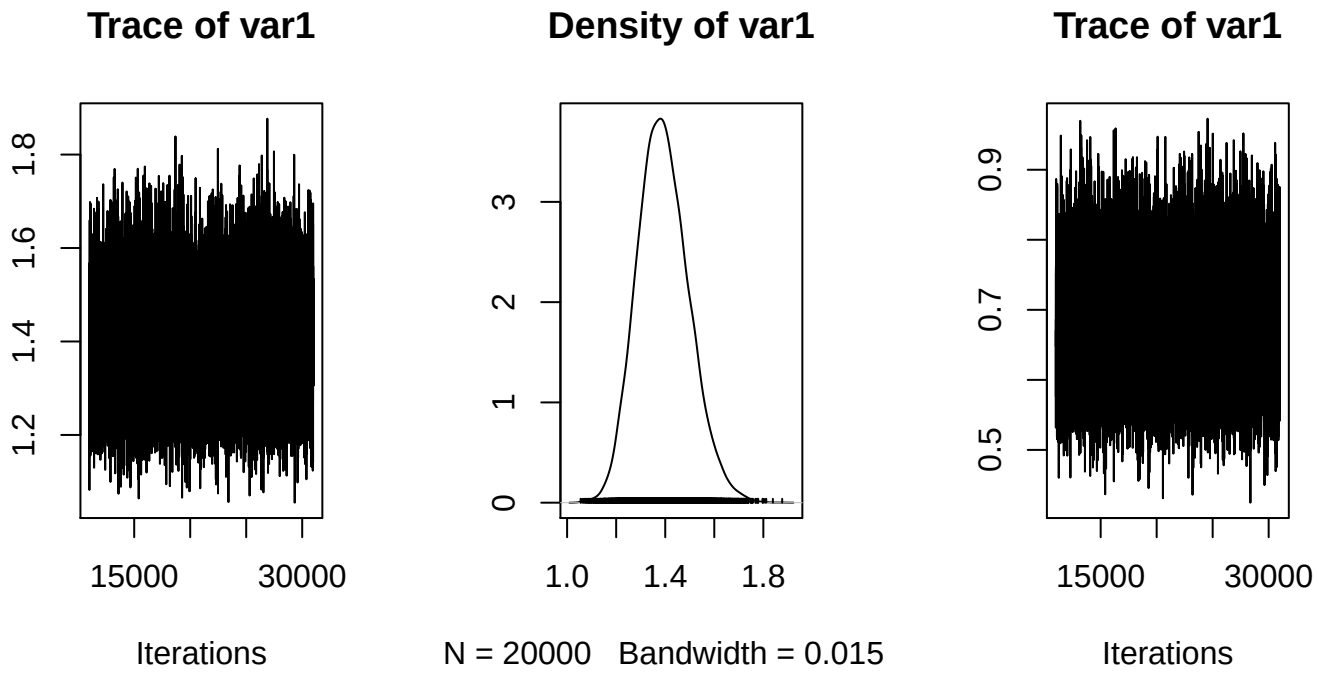
```
update(model, nSamples, progress.bar="none"); # Burnin
if(TRUE)
{
  out.coda  <- coda.samples(model, variable.names=model_parameters,n.iter=2*nSamples)
  save(out.coda,file = "out.coda_JAGS_VAR1.RData")
}else{
  load("out.coda_JAGS_VAR1.RData")
}

#plot(out.coda )

if(n.chains > 1)
{
  g <- matrix(NA, nrow=nvar(out.coda), ncol=2)
  for (v in 1:nvar(out.coda)) {
   g[v,] <- gelman.diag(out.coda[,v])$psrf
  }
  #multivariate - don't use if monitoring highly correlated variables
  #gelman.srf <-gelman.diag(out.coda)
  count.coeff.gt <- sum(g[,1]>1.1)
  count.coeff.gt
  plot(g[,1],main="Gelman-Rubin ")
}
```

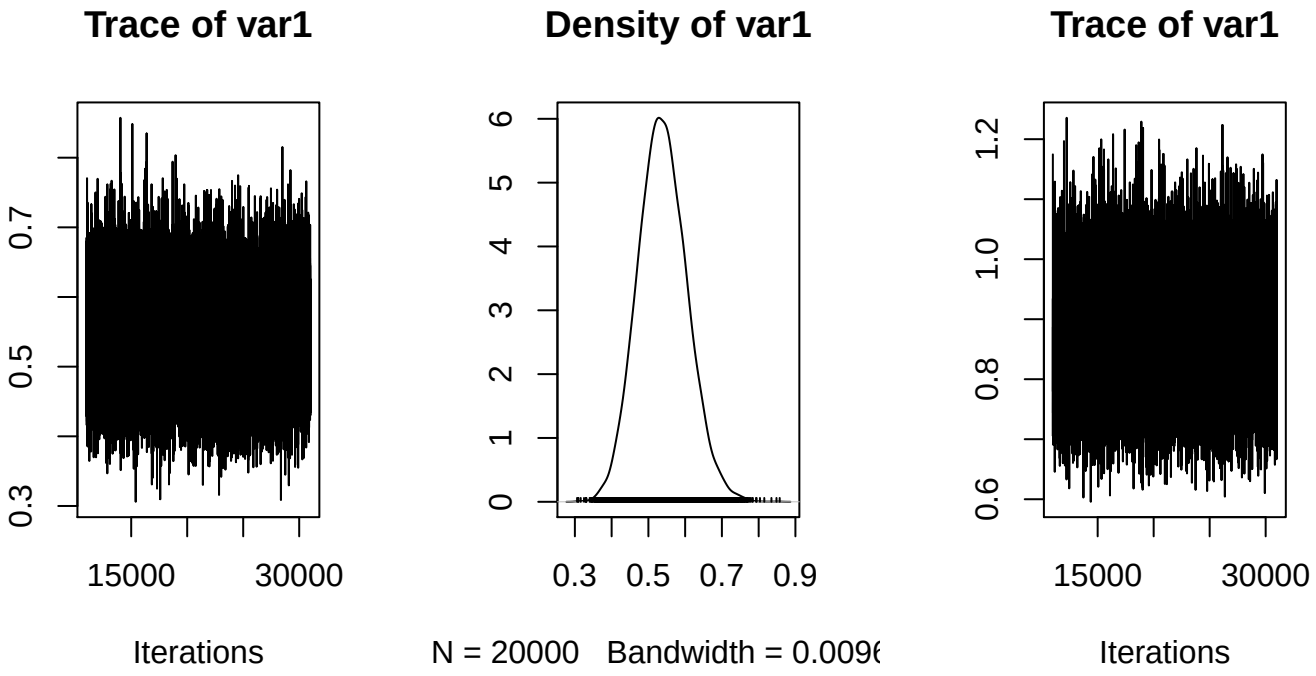


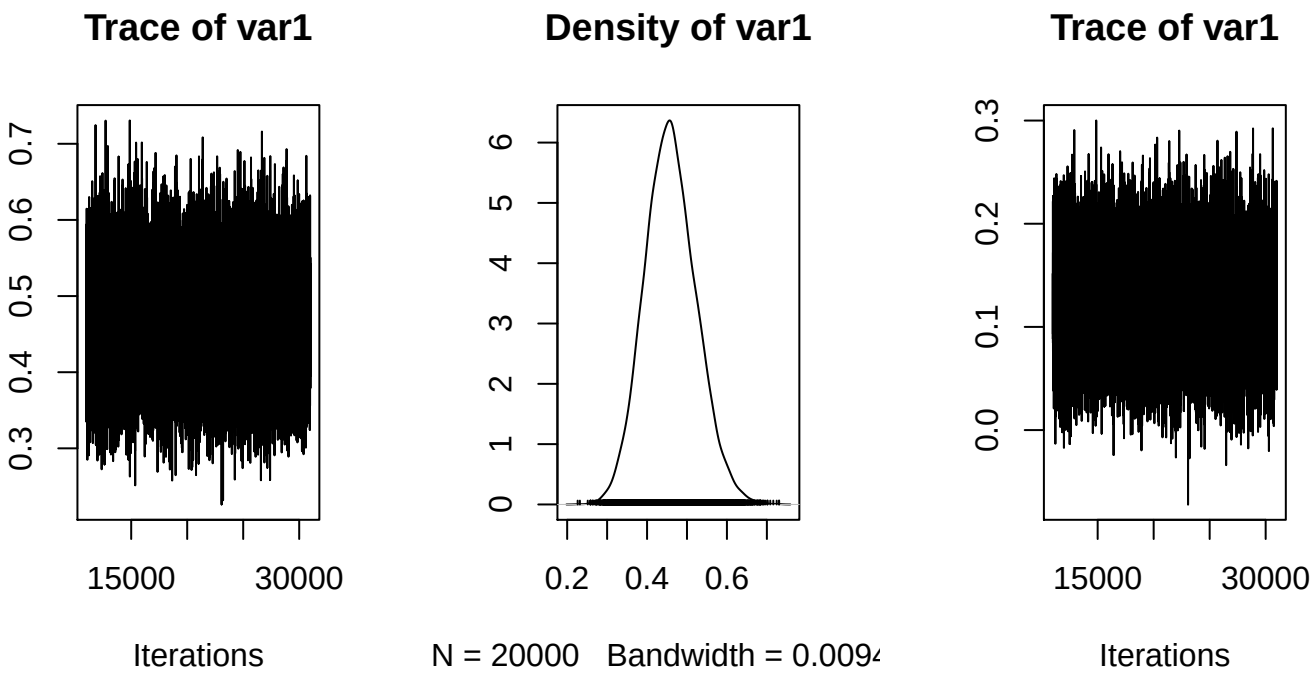

```
chains.ess <- lapply(out.coda,effectiveSize)
first.chain.ess <- chains.ess[1]
plot(unlist(first.chain.ess), main="Effective Sample Size")
```

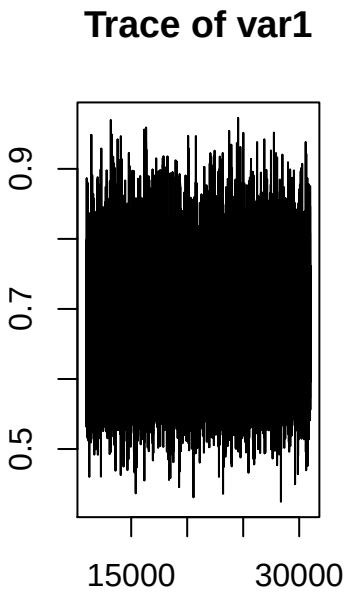


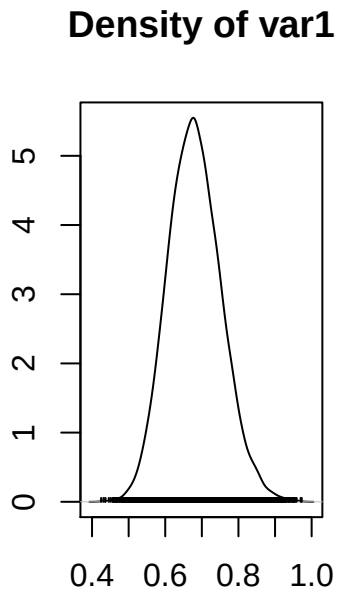
```
chain <- out.coda[[1]]
posterior.means <- list()
posterior.modes <- list()
for( i in 1:length(colnames(chain)) )
{
  colname <- colnames(chain)[i]
  plot(chain[,i])
  samples <- chain[,i]
  posterior.means[colname] <-mean(samples)
  posterior.modes[colname] <-mlv(samples)$M
}
```

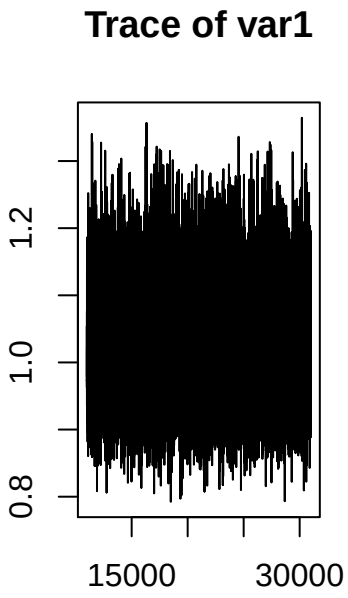


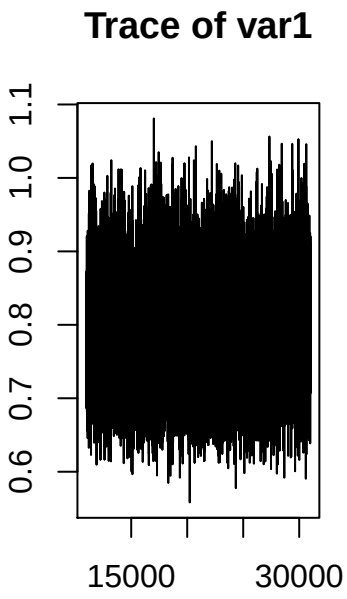

Trace of var1
Density of var1
Trace of var1
0.3
0.5
0.7
15000
30000
Iterations
0
1
2
3
4
5
6
0.3
0.5
0.7
0.9
N = 20000 Bandwidth = 0.009
0.6
0.8
1.0
1.2
15000
30000
Iterations

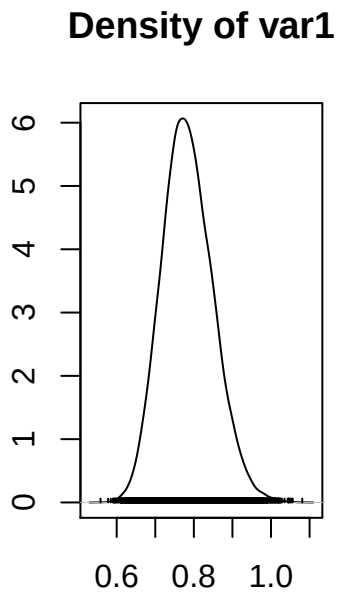

Trace of var1
Density of var1
Trace of var1
0.3
0.4
0.5
0.6
0.7
15000
30000
Iterations
0
1
2
3
4
5
6
0.2
0.4
0.6
N = 20000 Bandwidth = 0.009
0.0
0.1
0.2
0.3
15000
30000
Iterations

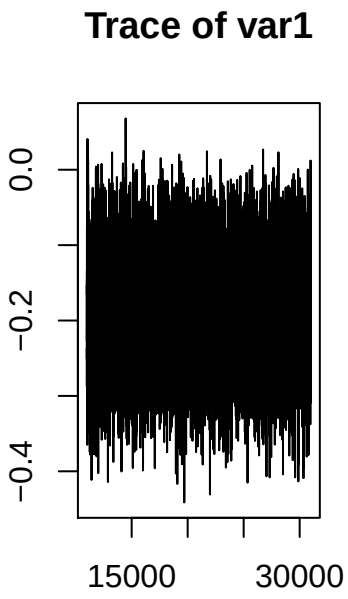

Trace of var1
0.5
0.7
0.9
15000
30000
Iterations

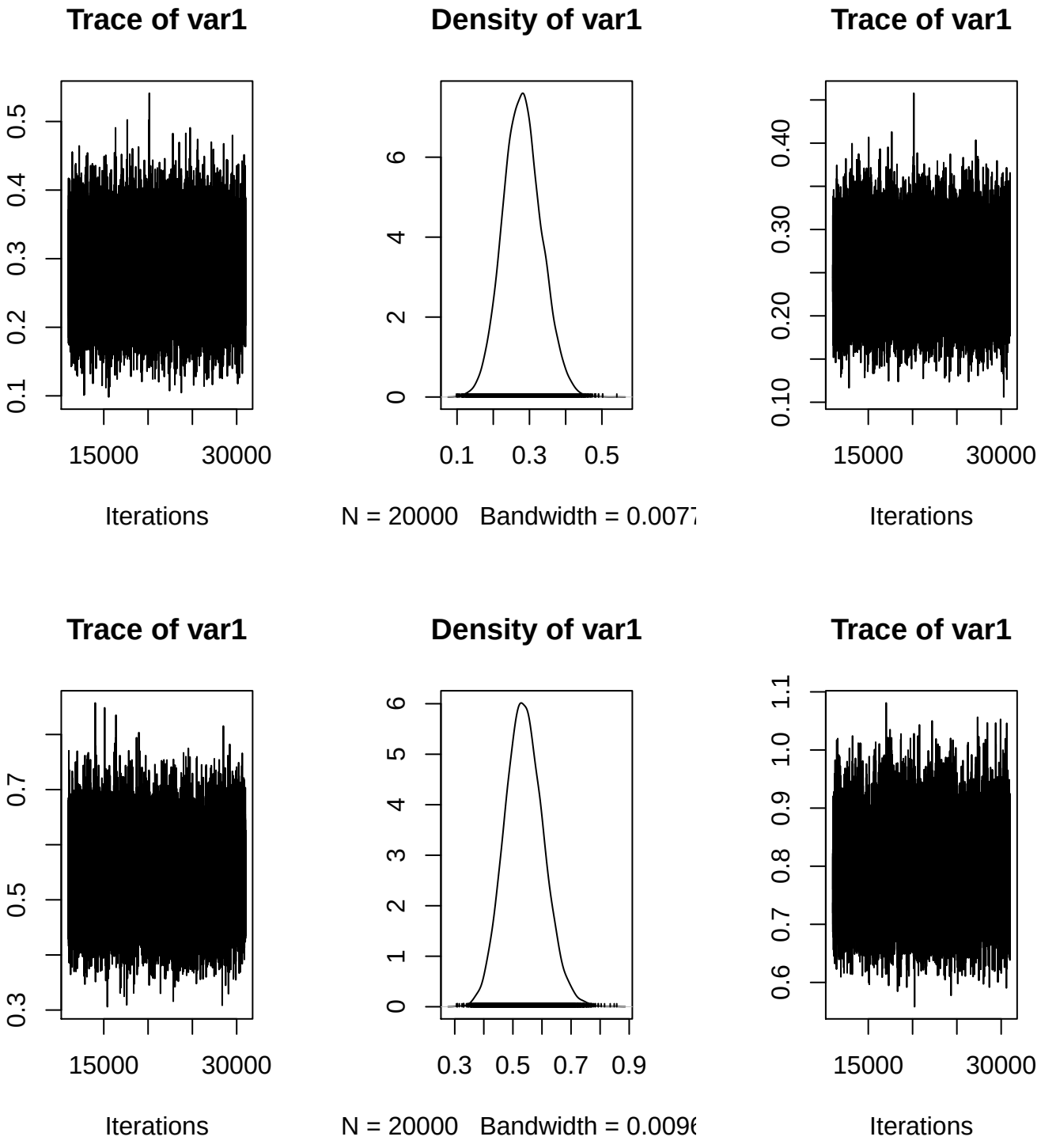

Density of var1
0
1
2
3
4
5
0.4
0.6
0.8
1.0
N = 20000 Bandwidth = 0.010

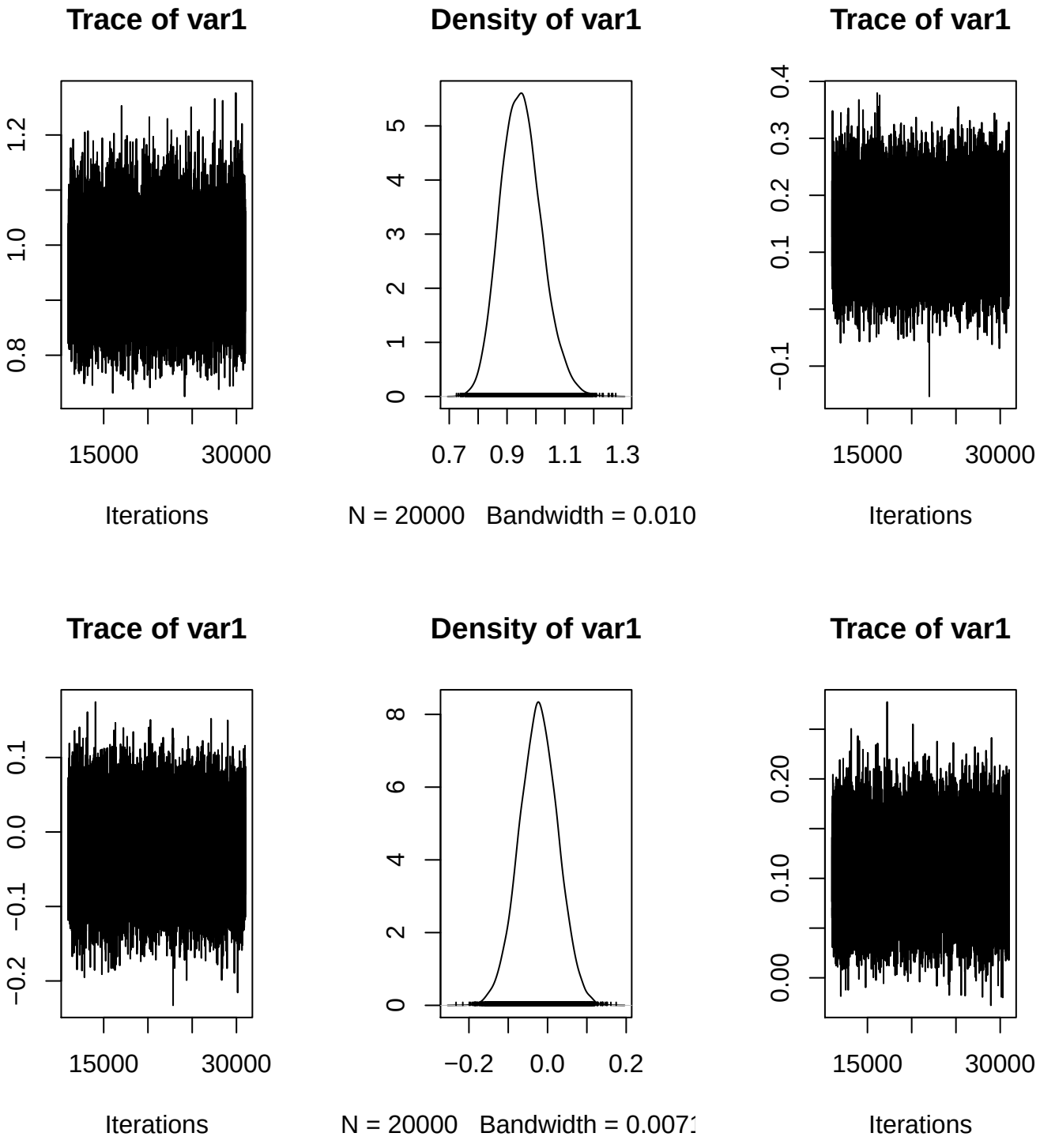

Trace of var1
0.8
1.0
1.2
15000
30000
Iterations

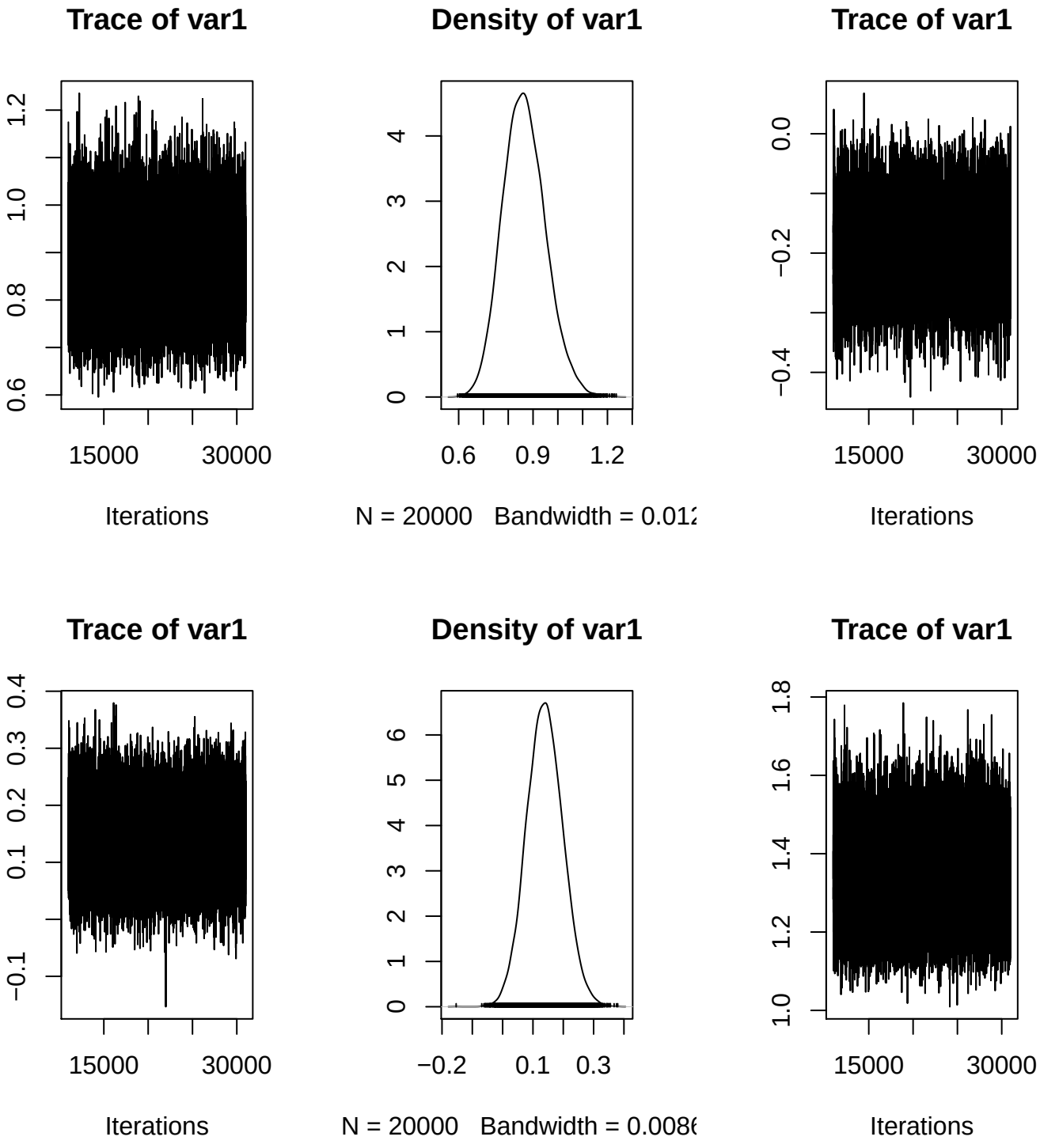

Trace of var1
0.6
0.7
0.8
0.9
1.0
1.1
15000
30000
Iterations

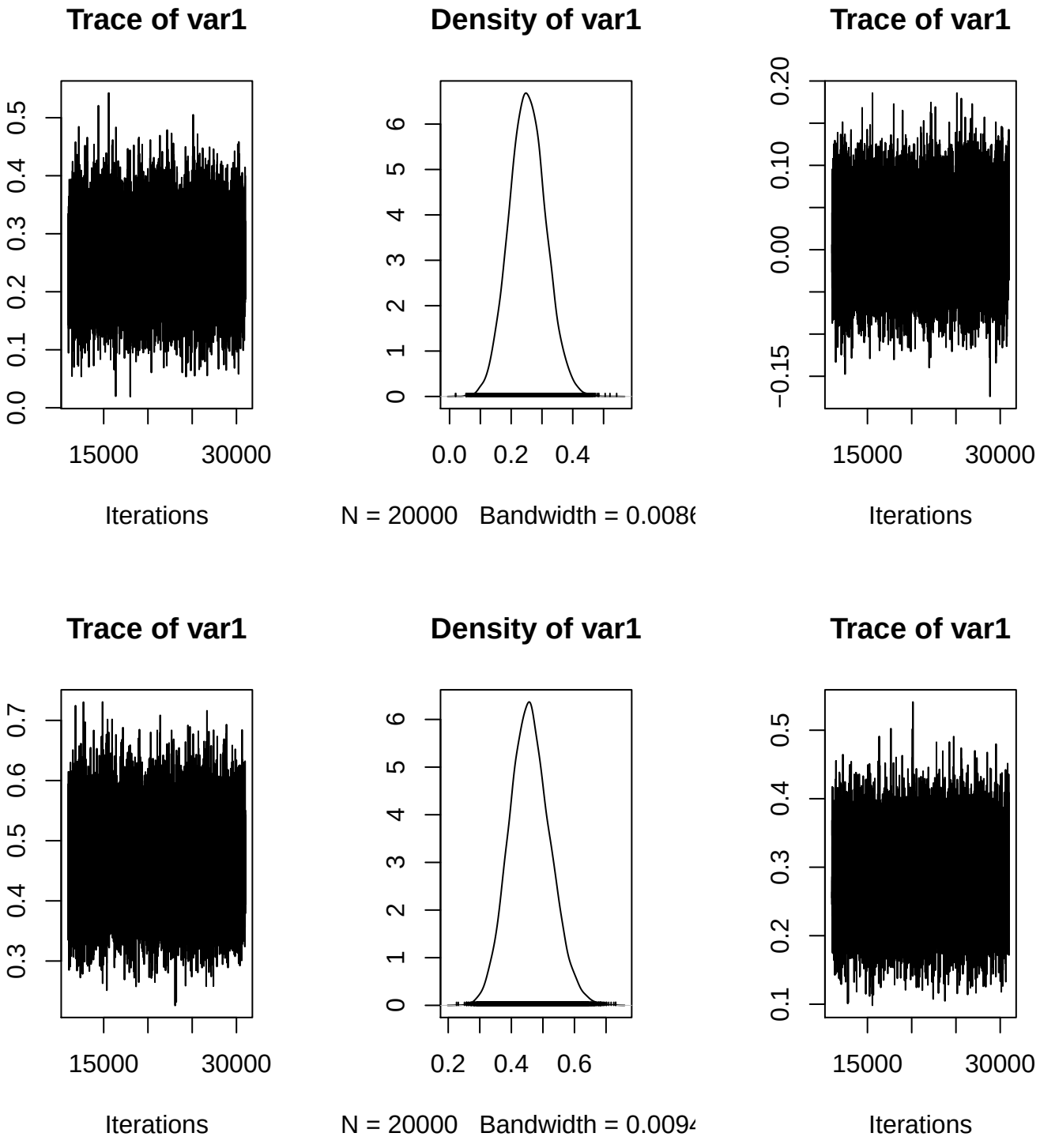

Density of var1
0
1
2
3
4
5
6
0.6
0.8
1.0
N = 20000 Bandwidth = 0.009

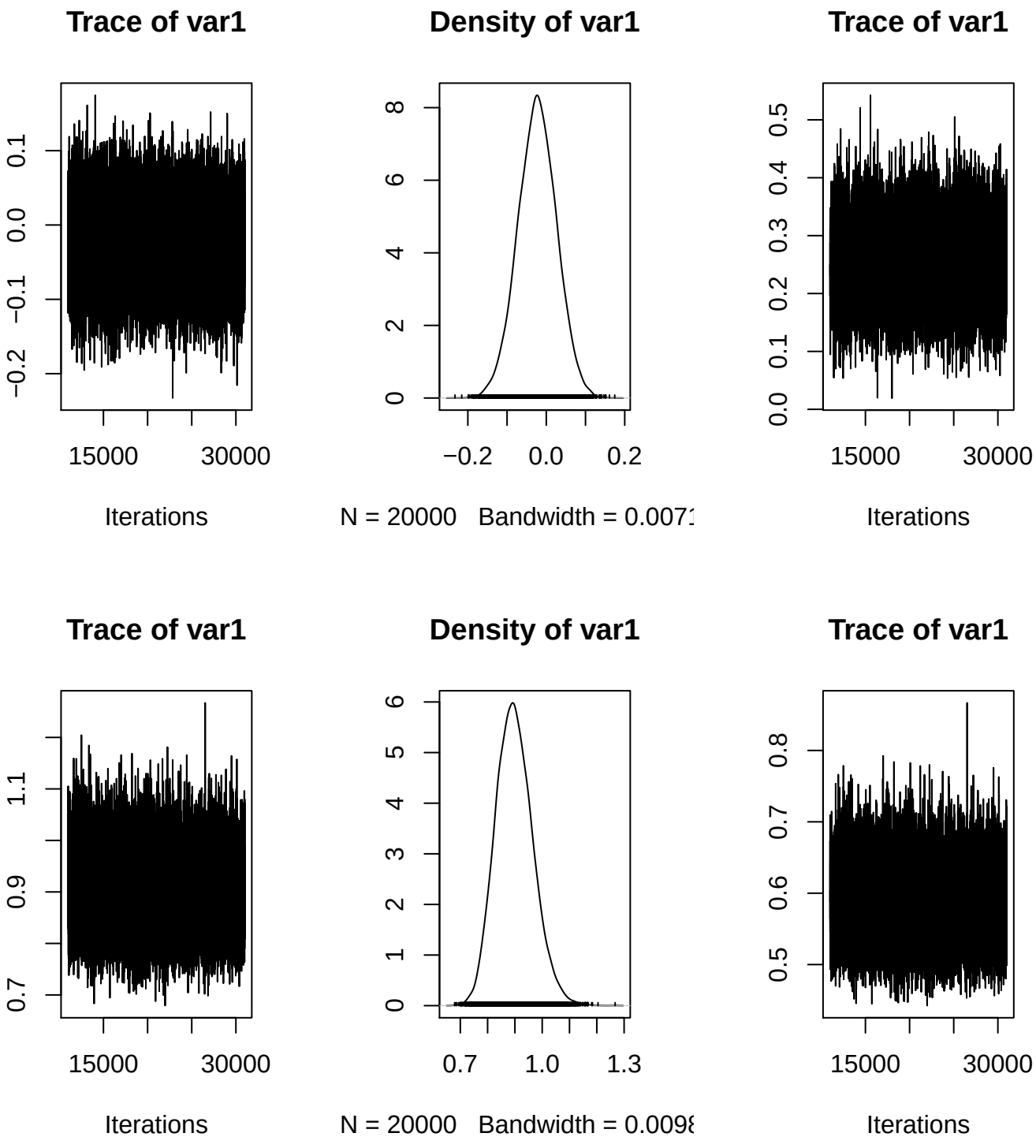

Trace of var1
−0.4
−0.2
0.0
15000
30000
Iterations

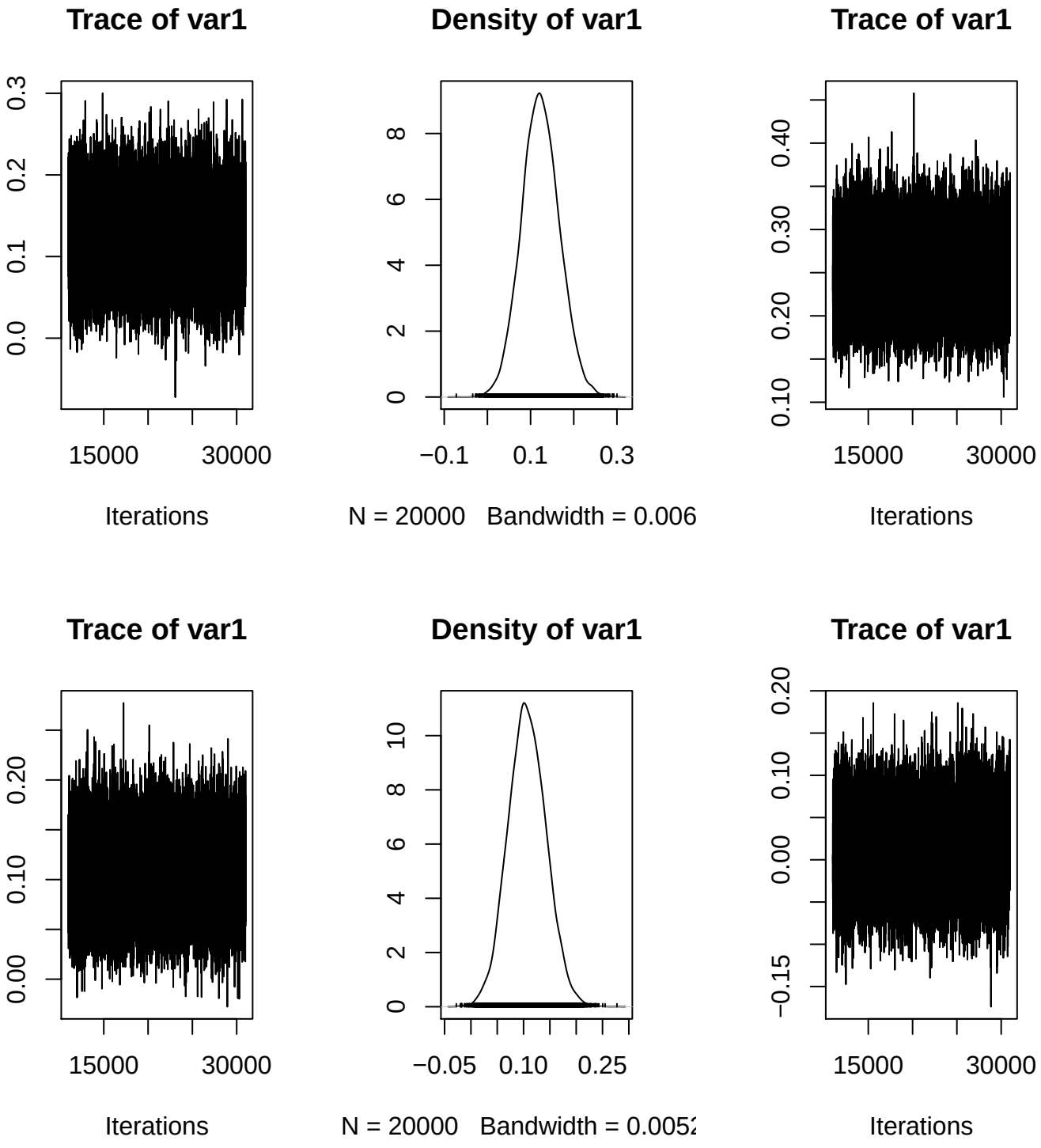

Trace of var1
Density of var1
Trace of var1
0.1
0.2
0.3
0.4
0.5
15000
30000
Iterations
0
2
4
6
0.1
0.3
0.5
N = 20000 Bandwidth = 0.007
0.10
0.20
0.30
0.40
15000
30000
Iterations
Trace of var1
Density of var1
Trace of var1
0.3
0.5
0.7
15000
30000
Iterations
0
1
2
3
4
5
6
0.3
0.5
0.7
0.9
N = 20000 Bandwidth = 0.009
0.6
0.7
0.8
0.9
1.0
1.1
15000
30000
Iterations

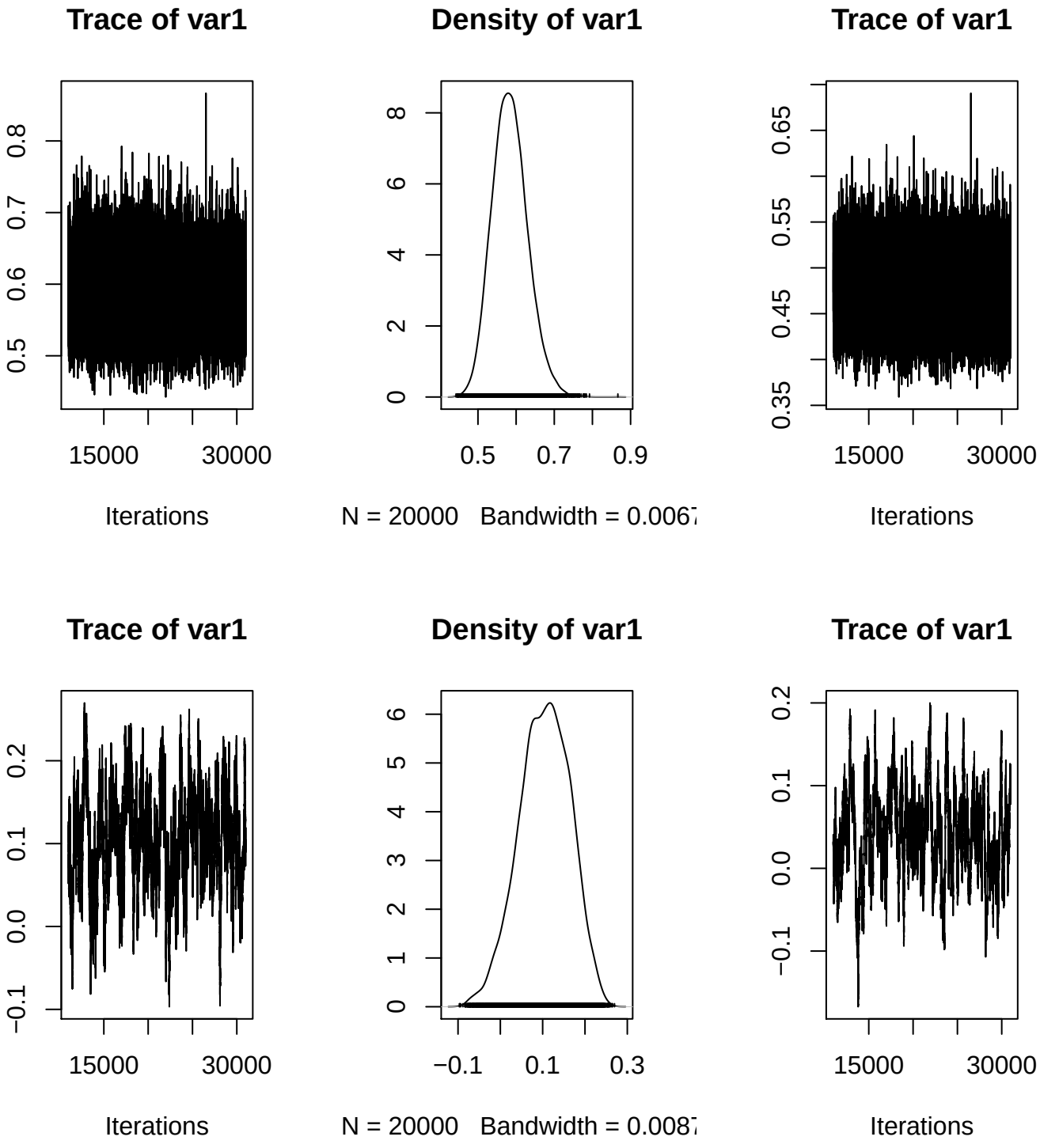

Trace of var1
Density of var1
Trace of var1
0.8
1.0
1.2
15000
30000
Iterations
0
1
2
3
4
5
0.7
0.9
1.1
1.3
N = 20000 Bandwidth = 0.010
−0.1
0.1
0.2
0.3
0.4
15000
30000
Iterations
Trace of var1
Density of var1
Trace of var1
−0.2
−0.1
0.0
0.1
15000
30000
Iterations
0
2
4
6
8
−0.2
0.0
0.2
N = 20000 Bandwidth = 0.007
0.00
0.10
0.20
15000
30000
Iterations

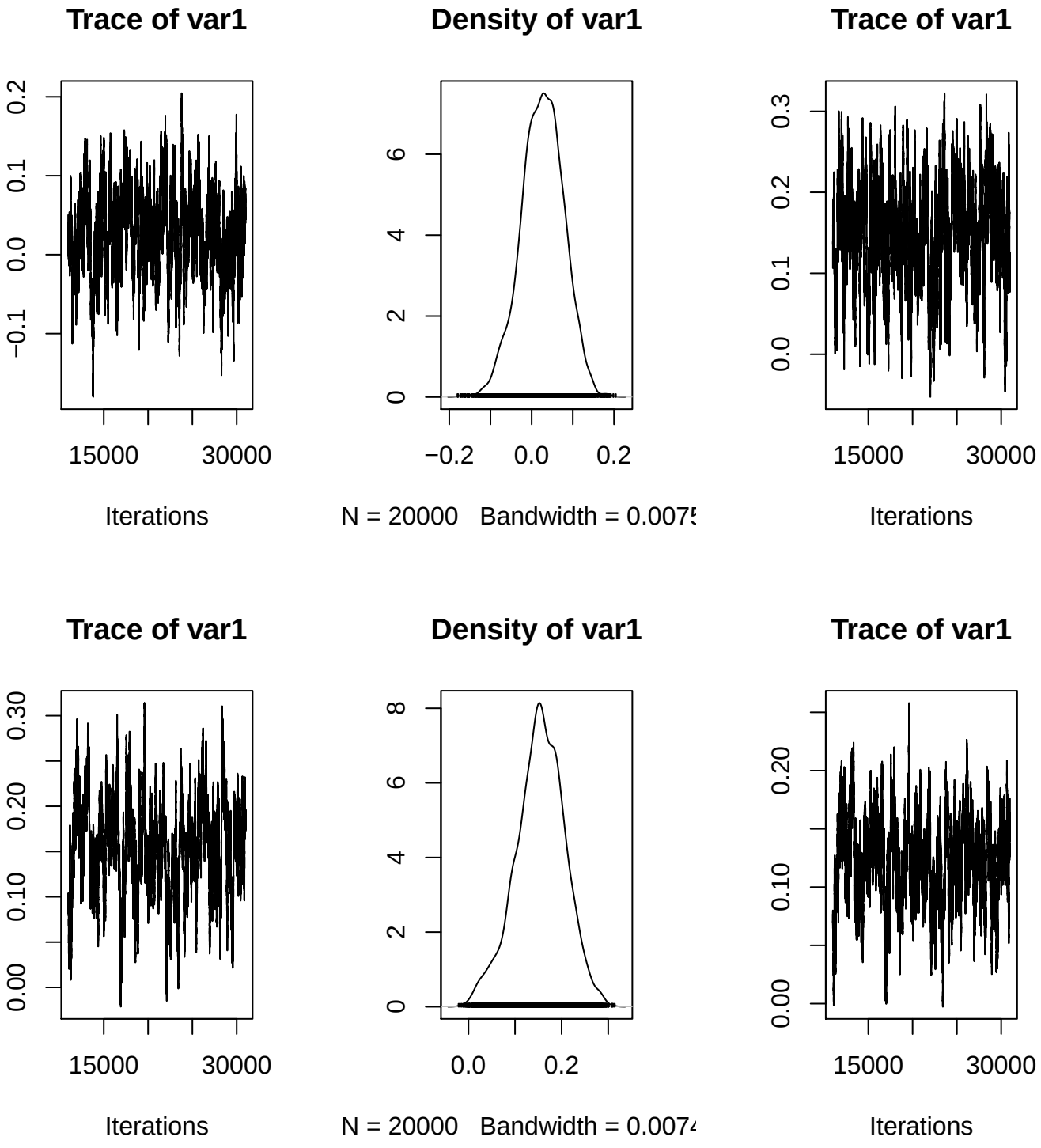

Trace of var1
Density of var1
Trace of var1
0.6
0.8
1.0
1.2
15000
30000
Iterations
0
1
2
3
4
0.6
0.9
1.2
N = 20000 Bandwidth = 0.01
−0.4
−0.2
0.0
15000
30000
Iterations
Trace of var1
Density of var1
Trace of var1
−0.1
0.1
0.2
0.3
0.4
15000
30000
Iterations
0
1
2
3
4
5
6
−0.2
0.1
0.3
N = 20000 Bandwidth = 0.008
1.0
1.2
1.4
1.6
1.8
15000
30000
Iterations

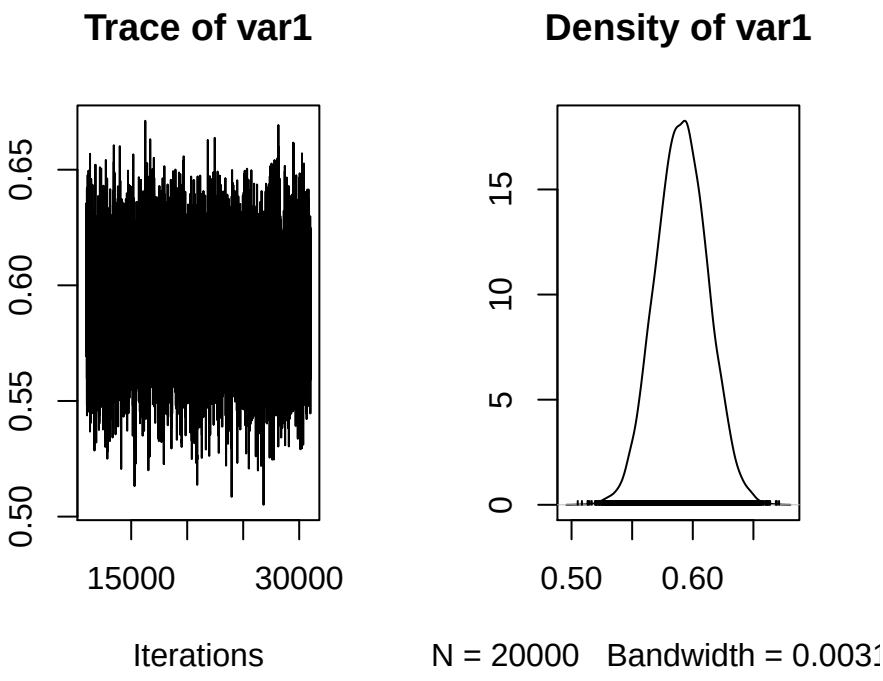

Trace of var1
Density of var1
Trace of var1
0.0 0.1 0.2 0.3 0.4 0.5
0 1 2 3 4 5 6
−0.15 0.00 0.10 0.20
15000 30000
0.0 0.2 0.4
15000 30000
Iterations
N = 20000 Bandwidth = 0.008
Iterations
Trace of var1
Density of var1
Trace of var1
0.3 0.4 0.5 0.6 0.7
0 1 2 3 4 5 6
0.1 0.2 0.3 0.4 0.5
15000 30000
0.2 0.4 0.6
15000 30000
Iterations
N = 20000 Bandwidth = 0.009
Iterations


Trace of var1
Density of var1
Trace of var1
−0.2
−0.1
0.0
0.1
15000
30000
Iterations
0
2
4
6
8
−0.2
0.0
0.2
N = 20000 Bandwidth = 0.007
0.0
0.1
0.2
0.3
0.4
0.5
15000
30000
Iterations
Trace of var1
Density of var1
Trace of var1
0.7
0.9
1.1
15000
30000
Iterations
0
1
2
3
4
5
6
0.7
1.0
1.3
N = 20000 Bandwidth = 0.009
0.5
0.6
0.7
0.8
15000
30000
Iterations

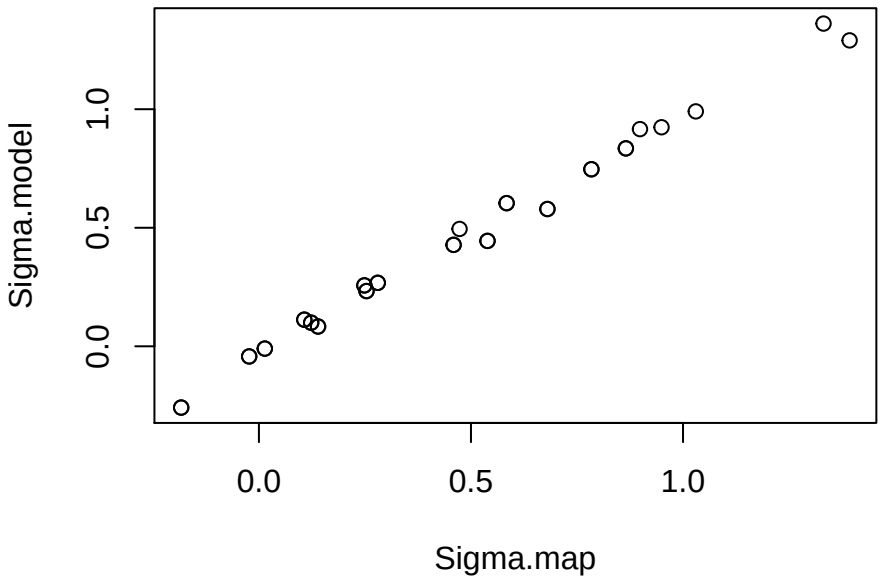

Trace of var1
Density of var1
Trace of var1
0.0
0.1
0.2
0.3
15000
30000
Iterations
0
2
4
6
8
−0.1
0.1
0.3
N = 20000 Bandwidth = 0.006
0.10
0.20
0.30
0.40
15000
30000
Iterations
Trace of var1
Density of var1
Trace of var1
0.00
0.10
0.20
15000
30000
Iterations
0
2
4
6
8
10
−0.05
0.10
0.25
N = 20000 Bandwidth = 0.005
−0.15
0.00
0.10
0.20
15000
30000
Iterations


Trace of var1
Density of var1
Trace of var1
0.5
0.6
0.7
0.8
15000
30000
Iterations
0
2
4
6
8
0.5
0.7
0.9
N = 20000 Bandwidth = 0.006
0.35
0.45
0.55
0.65
15000
30000
Iterations
Trace of var1
Density of var1
Trace of var1
−0.1
0.0
0.1
0.2
15000
30000
Iterations
0
1
2
3
4
5
6
−0.1
0.1
0.3
N = 20000 Bandwidth = 0.008
−0.1
0.0
0.1
0.2
15000
30000
Iterations


Trace of var1
Density of var1
Trace of var1
−0.1
0.0
0.1
0.2
15000
30000
Iterations
0
2
4
6
−0.2
0.0
0.2
N = 20000 Bandwidth = 0.007
0.0
0.1
0.2
0.3
15000
30000
Iterations
Trace of var1
Density of var1
Trace of var1
0.00
0.10
0.20
0.30
15000
30000
Iterations
0
2
4
6
8
0.0
0.2
N = 20000 Bandwidth = 0.007
0.00
0.10
0.20
15000
30000
Iterations




```
Sigma.map <- unlist(posterior.means)[1:36]
Sigma.model<- unlist(sig$mat)
plot(Sigma.map,Sigma.model)
```



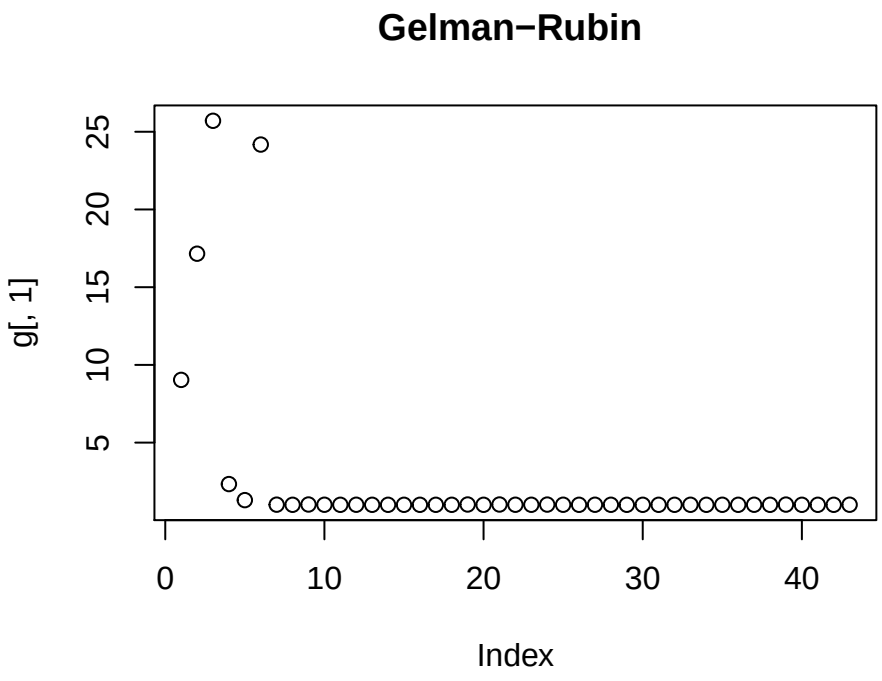
```
rho.map <- unlist(posterior.means)[43]

nu.map <-  unlist(posterior.means)[37:42]
```

Replicate JAGS in OpenBugs

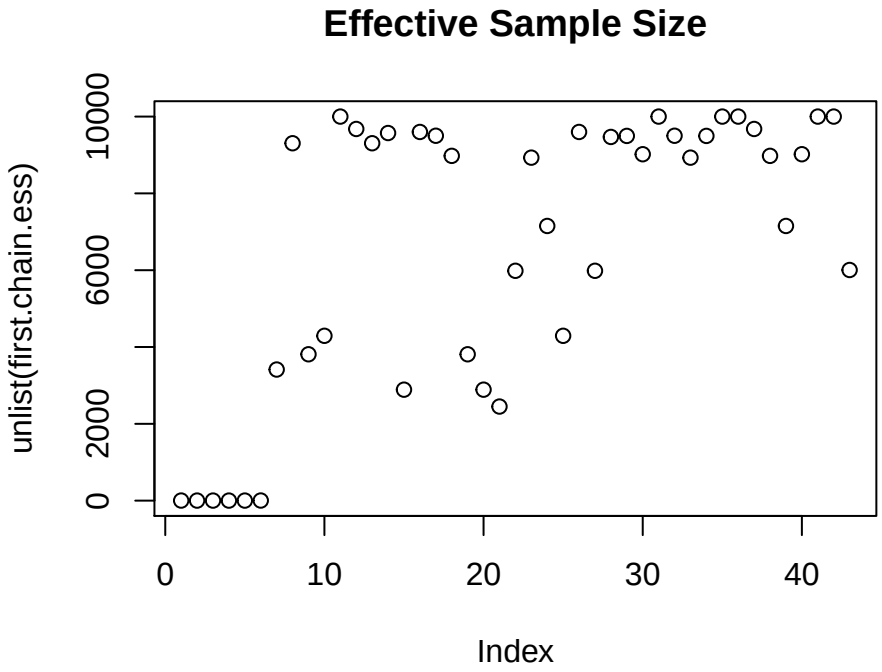
VAR(1) on Y1 with starting values, Y2 and Y3 incorporated imputation in OpenBugs

```
library(R2OpenBUGS)

openbugs_model <- function() {
    for (i in 2:N) {
        Y1[i, 1:p] ~ dmnorm(theta[i, 1:p],
            precision[, ])
        for (j in 1:p) {
            theta[i, j] <- nu[j] + rho *
                Y1[i - 1, j]
        }
    }

    # Priors
    rho ~ dunif(-1, 1)

    for (j in 1:p) {
        nu[j] ~ dnorm(0, 0.01)
    }

    # Sigma[1:p,1:p] <-
    # inverse(precision[1:p,1:p])

    precision[1:p, 1:p] ~ dwish(R[, ],
        k)


    k <- p + 1
    for (j1 in 1:p) {
        for (j2 in 1:p) {
            R[j1, j2] <- equals(j1, j2)
        }
    }

}
```

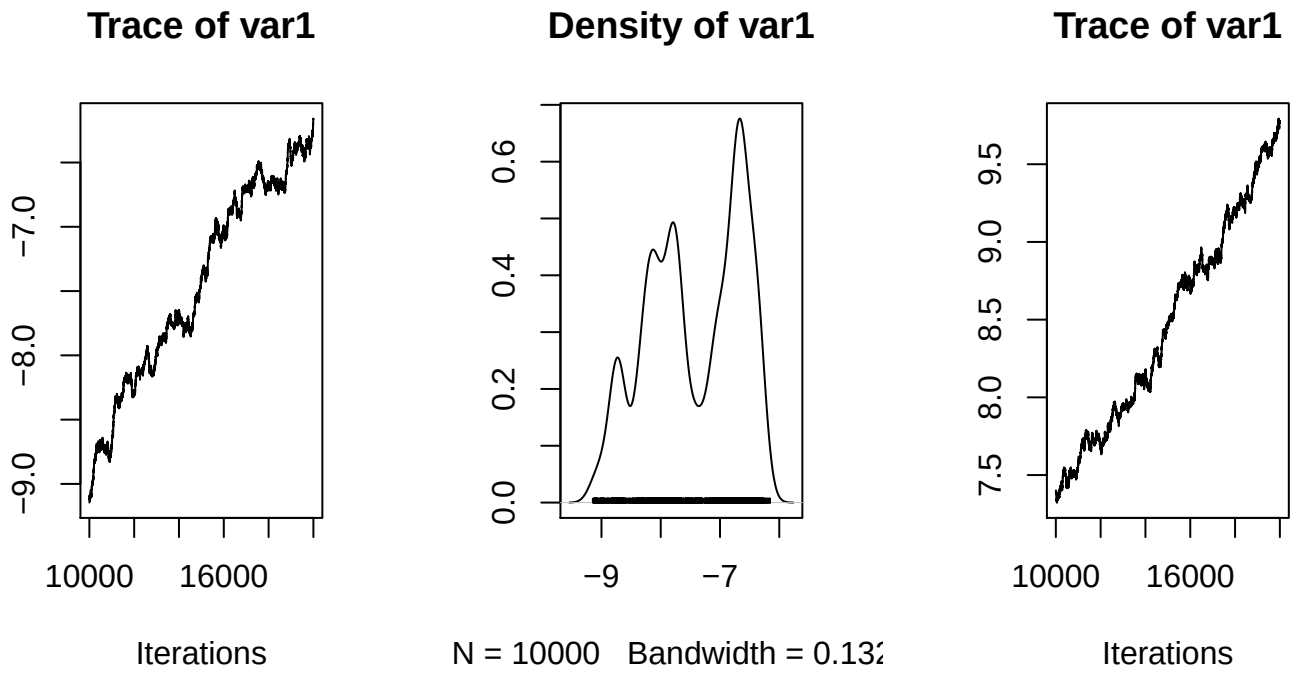
```
stacks_dat <- list(Y1 = scale(y), p = 6,
    N = 365)
mlr_inits <- function() {
    list(rho = 0, precision = diag(p))
}
parameters.to.save = c("precision", "nu",
    "rho")

samps <- bugs(data = stacks_dat, inits = mlr_inits,
    parameters.to.save = parameters.to.save,
    model.file = openbugs_model, codaPkg = TRUE,
    n.chains = n.chains, n.burnin = 10000,
    n.iter = 2 * nSamples, DIC = F)  #, n.thin=10
out.coda <- read.bugs(samps)
save(out.coda, file = "out.coda_BUGS_VAR.RData")

if (n.chains > 1) {
    g <- matrix(NA, nrow = nvar(out.coda),
        ncol = 2)
    for (v in 1:nvar(out.coda)) {
        g[v, ] <- gelman.diag(out.coda[,
            v])$psrf
    }
    # multivariate - don't use if
    # monitoring highly correlated
    # variables gelman.srf
    # <-gelman.diag(out.coda)
    count.coeff.gt <- sum(g[, 1] > 1.1)
    count.coeff.gt
    plot(g[, 1], main = "Gelman-Rubin ")
}
```

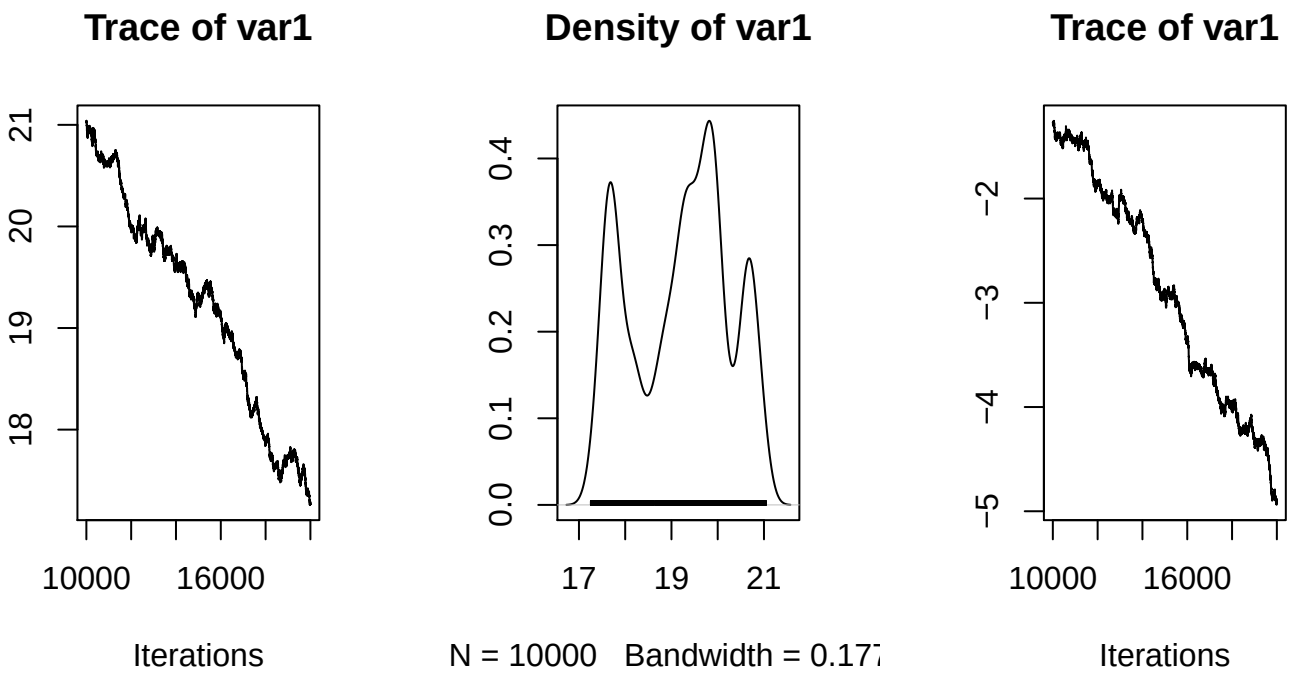


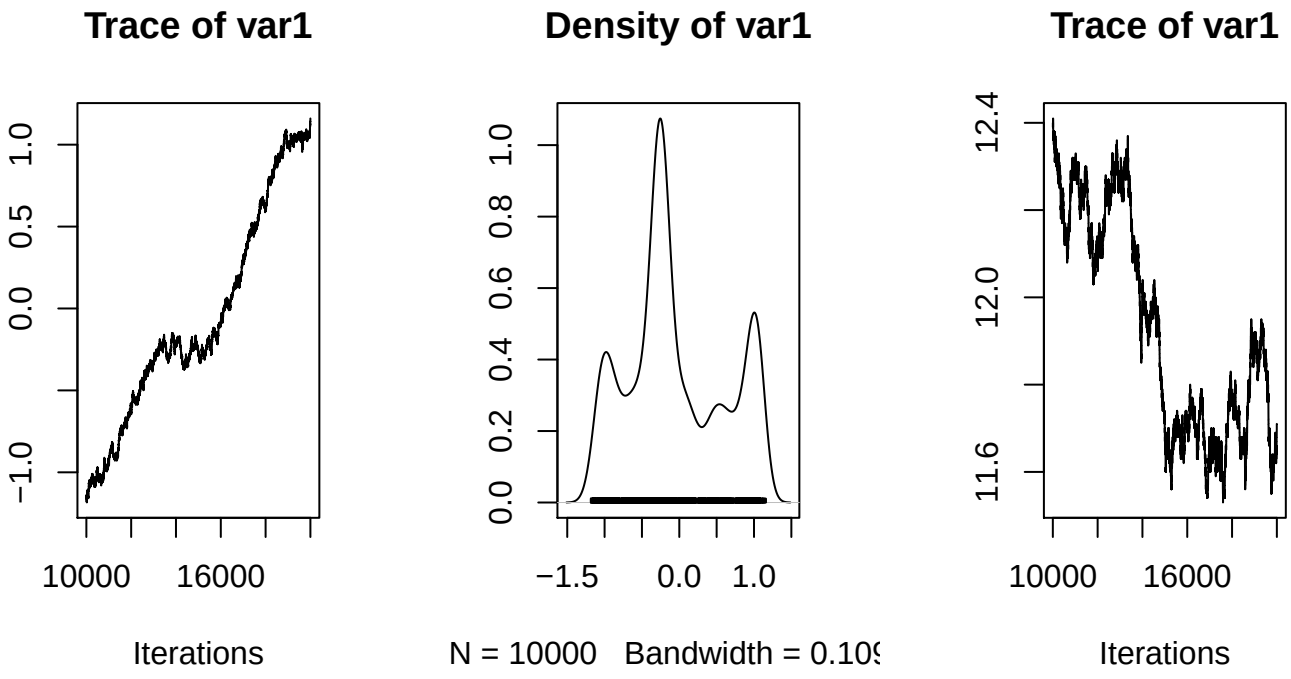

```
chains.ess <- lapply(out.coda, effectiveSize)
first.chain.ess <- chains.ess[1]
plot(unlist(first.chain.ess), main = "Effective Sample Size")
```

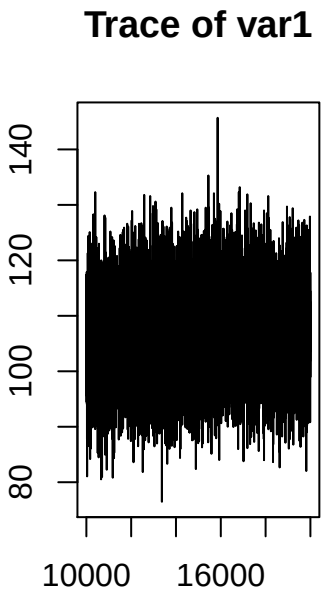


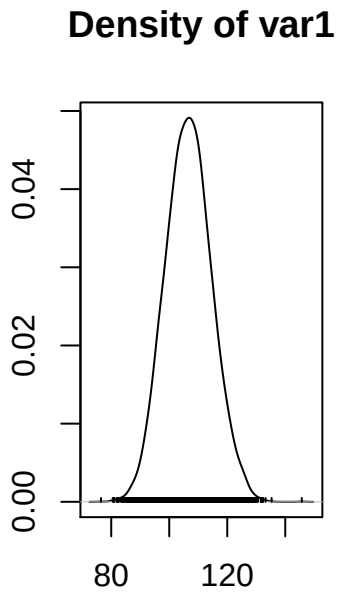
```r
chain <- out.coda[[1]]
posterior.means <- list()
posterior.modes <- list()
for (i in 1:length(colnames(chain))) {
    colname <- colnames(chain)[i]
    plot(chain[, i])
    samples <- chain[, i]
    posterior.means[colname] <- mean(samples)
    posterior.modes[colname] <- mlv(samples)$M
}
```

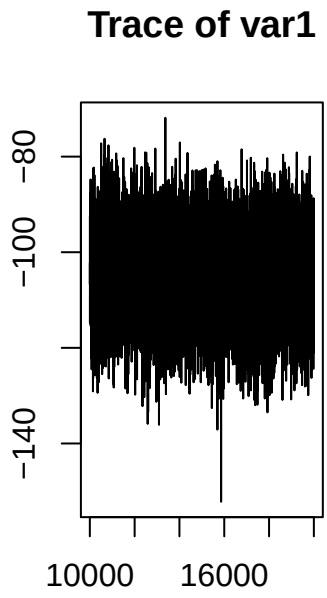


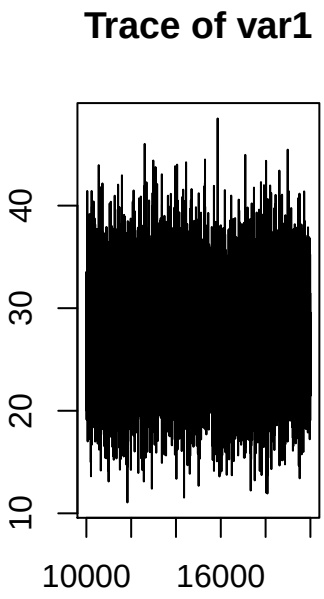

Trace of var1
Density of var1
Trace of var1
18
19
20
21
10000
16000
Iterations
0.0
0.1
0.2
0.3
0.4
17
19
21
N = 10000 Bandwidth = 0.17
−5
−4
−3
−2
10000
16000
Iterations

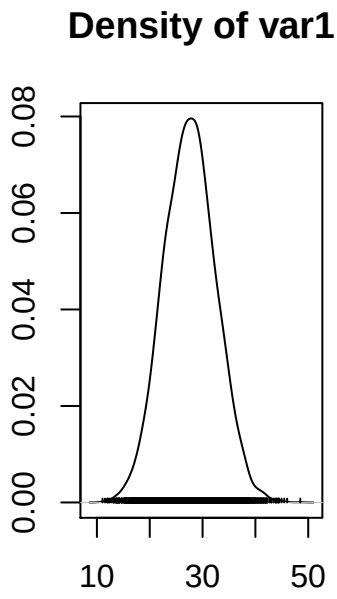

Trace of var1
Density of var1
Trace of var1
−1.0
0.0
0.5
1.0
10000
16000
Iterations
0.0
0.2
0.4
0.6
0.8
1.0
−1.5
0.0
1.0
N = 10000 Bandwidth = 0.10
11.6
12.0
12.4
10000
16000
Iterations

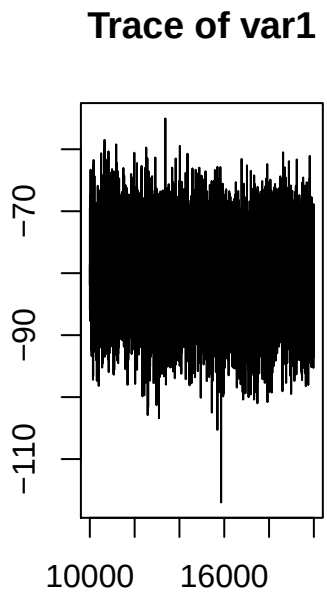

Trace of var1
80
100
120
140
10000
16000
Iterations

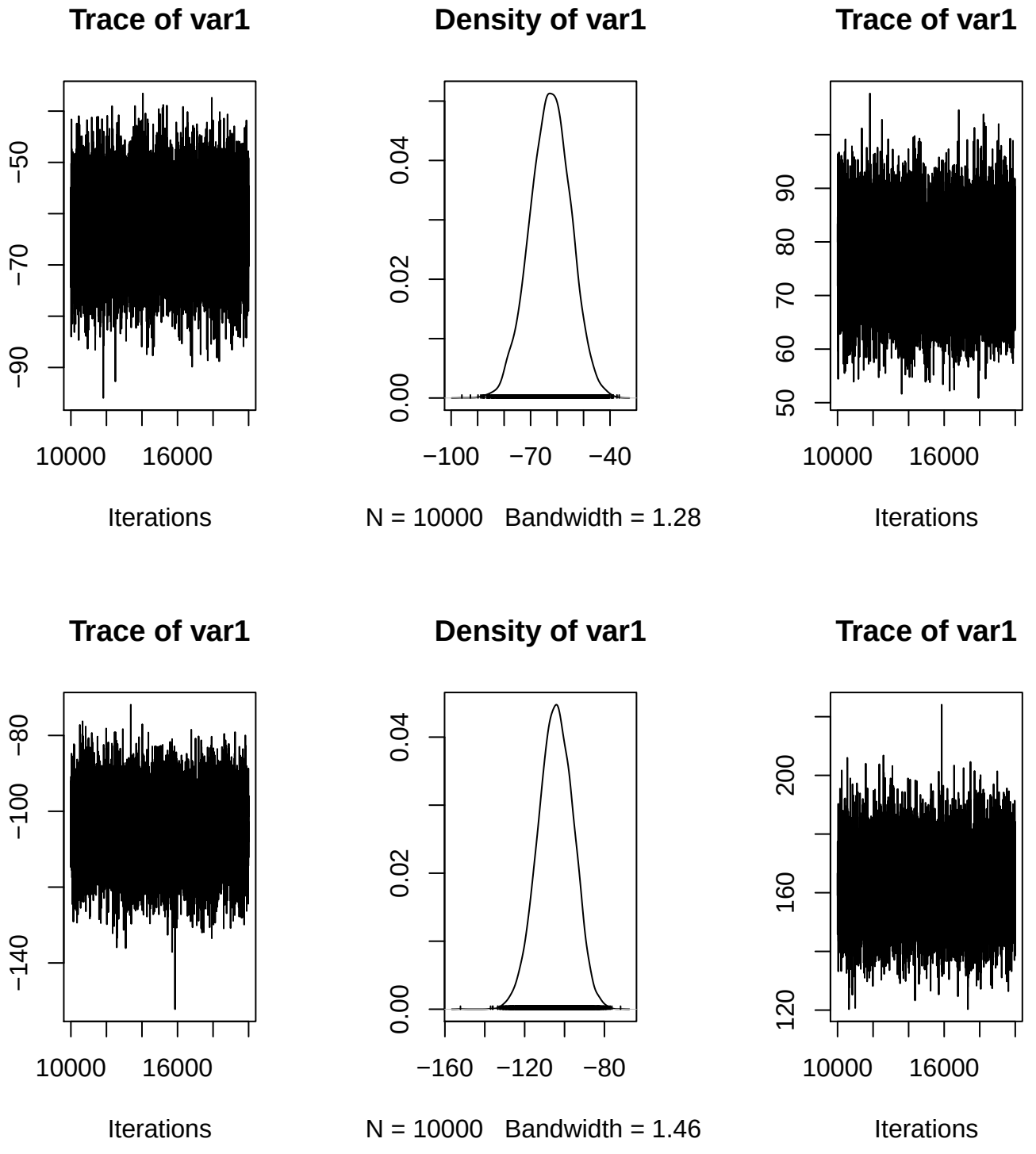

Density of var1
0.00
0.02
0.04
80
120
N = 10000 Bandwidth = 1.33

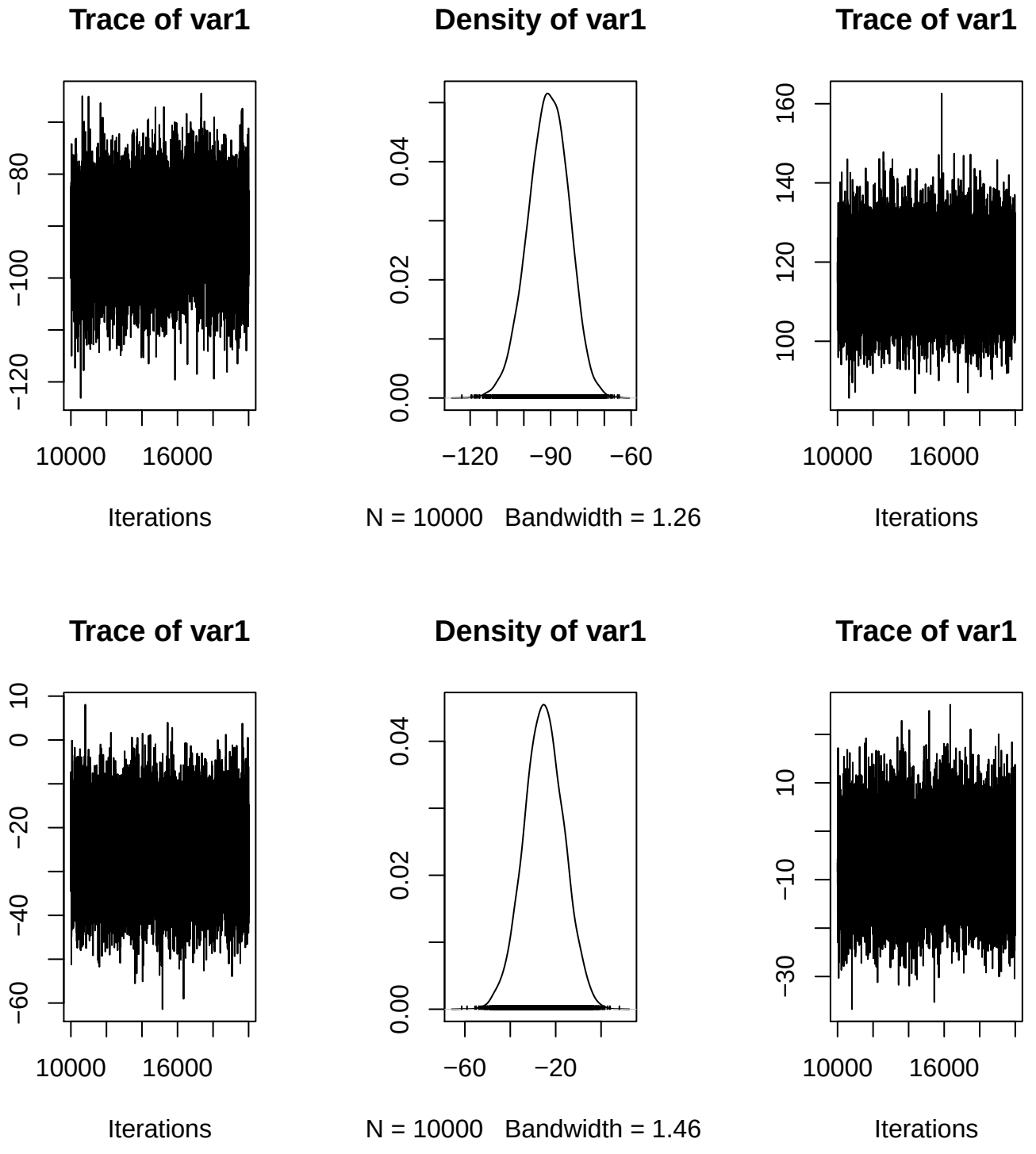

Trace of var1
−140
−100
−80
10000
16000
Iterations

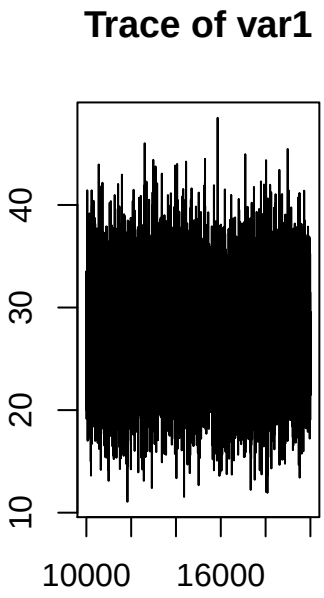

Trace of var1
10
20
30
40
10000
16000
Iterations

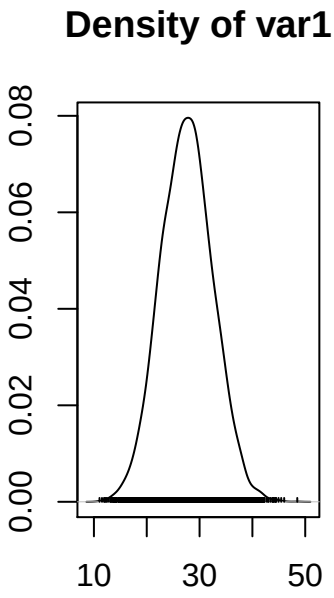

Density of var1
0.00
0.02
0.04
0.06
0.08
10
30
50
N = 10000 Bandwidth = 0.82

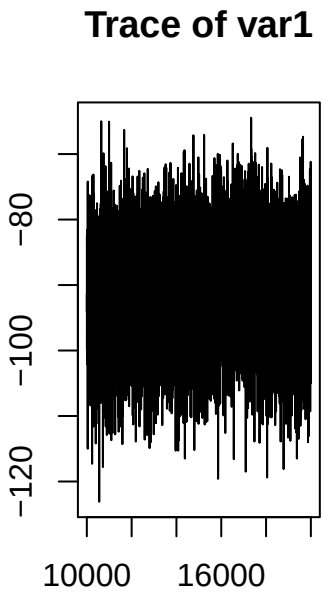

Trace of var1
−110
−90
−70
10000
16000
Iterations

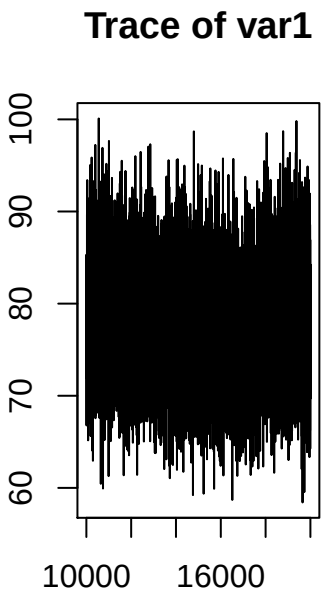

Trace of var1
Density of var1
Trace of var1
−50
−70
−90
10000
16000
Iterations
0.04
0.02
0.00
−100
−70
−40
N = 10000 Bandwidth = 1.28
90
80
70
60
50
10000
16000
Iterations
Trace of var1
Density of var1
Trace of var1
−80
−100
−140
10000
16000
Iterations
0.04
0.02
0.00
−160
−120
−80
N = 10000 Bandwidth = 1.46
200
160
120
10000
16000
Iterations

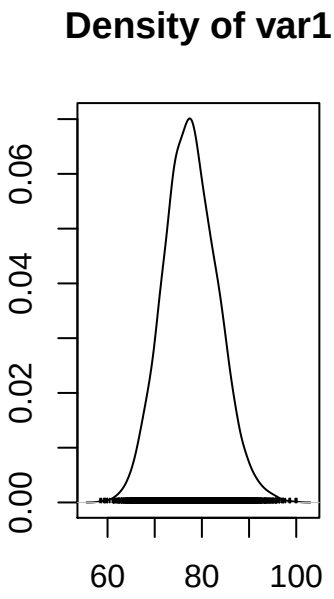

Trace of var1
Density of var1
Trace of var1
−120
−100
−80
10000
16000
Iterations
0.00
0.02
0.04
−120
−90
−60
N = 10000 Bandwidth = 1.26
100
120
140
160
10000
16000
Iterations
Trace of var1
Density of var1
Trace of var1
−60
−40
−20
0
10
10000
16000
Iterations
0.00
0.02
0.04
−60
−20
N = 10000 Bandwidth = 1.46
−30
−10
10
10000
16000
Iterations

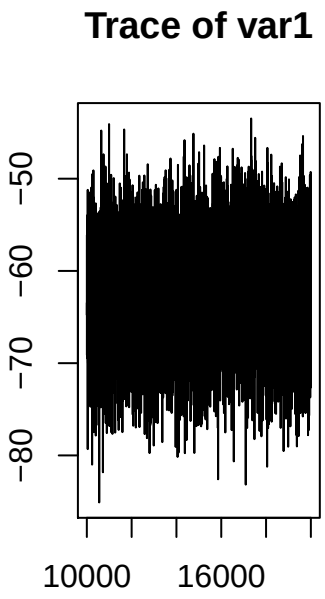

Trace of var1
10
20
30
40
10000
16000
Iterations

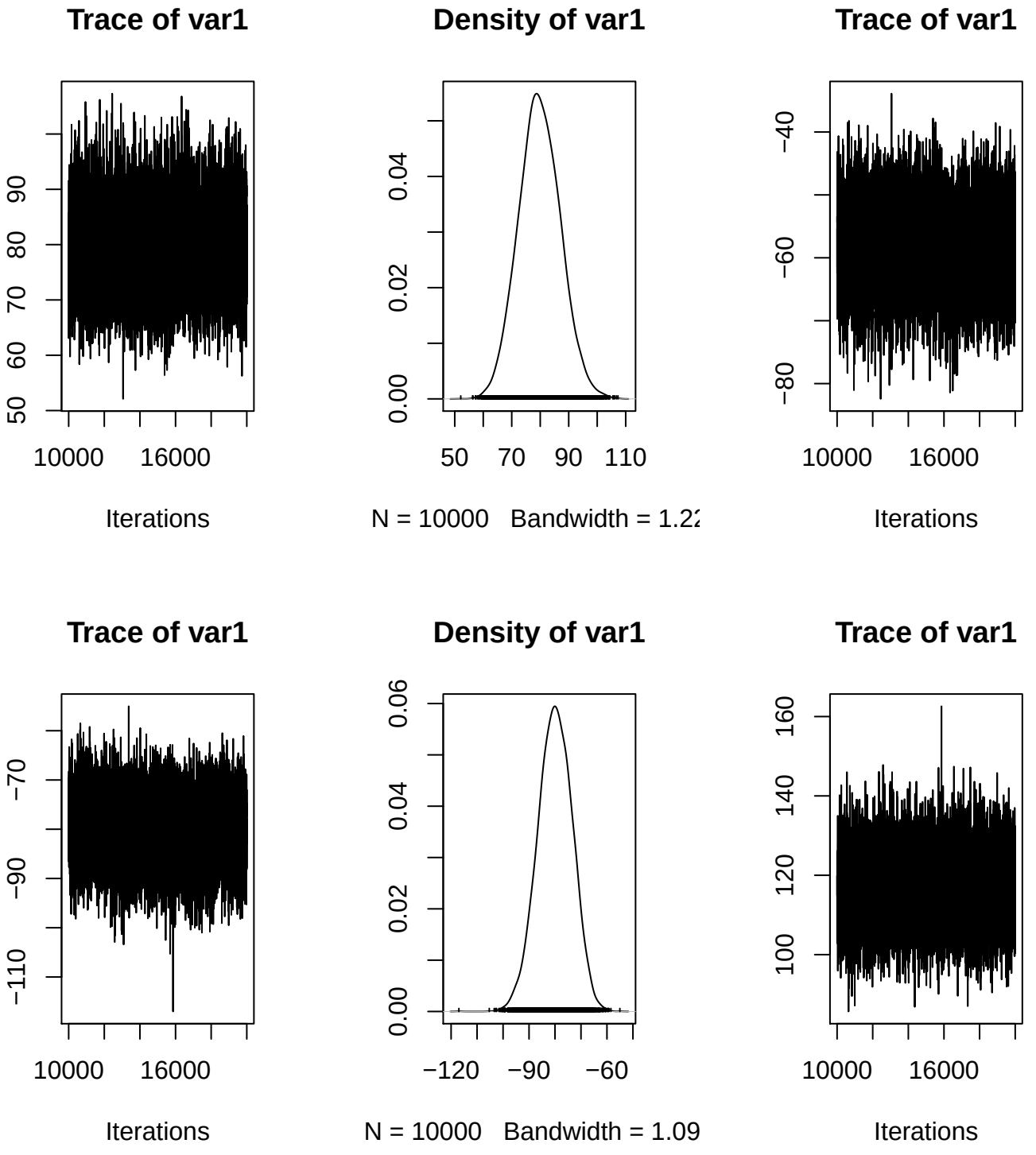

Density of var1
0.00
0.02
0.04
0.06
0.08
10
30
50
N = 10000 Bandwidth = 0.82

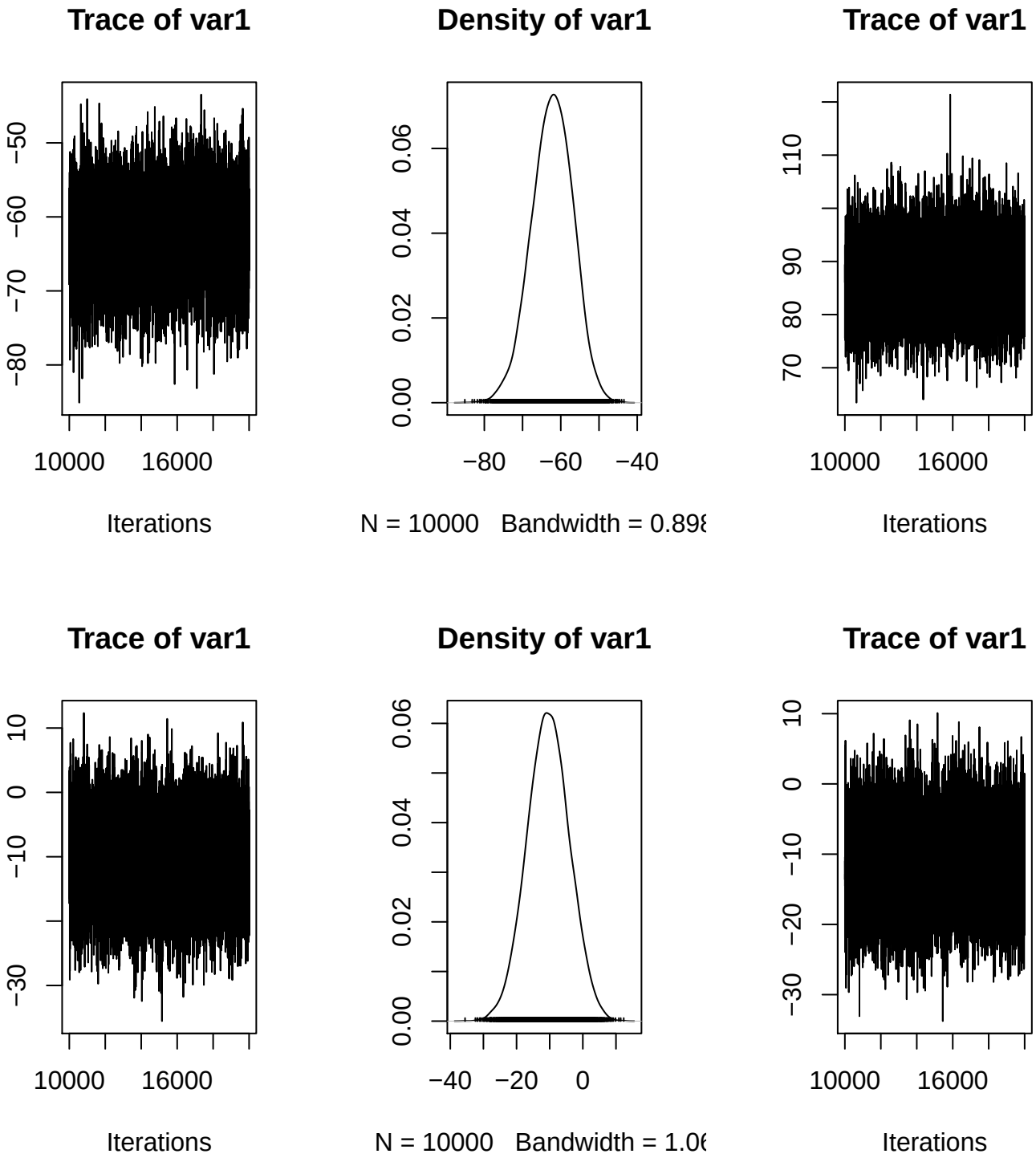

Trace of var1
−120
−100
−80
10000
16000
Iterations

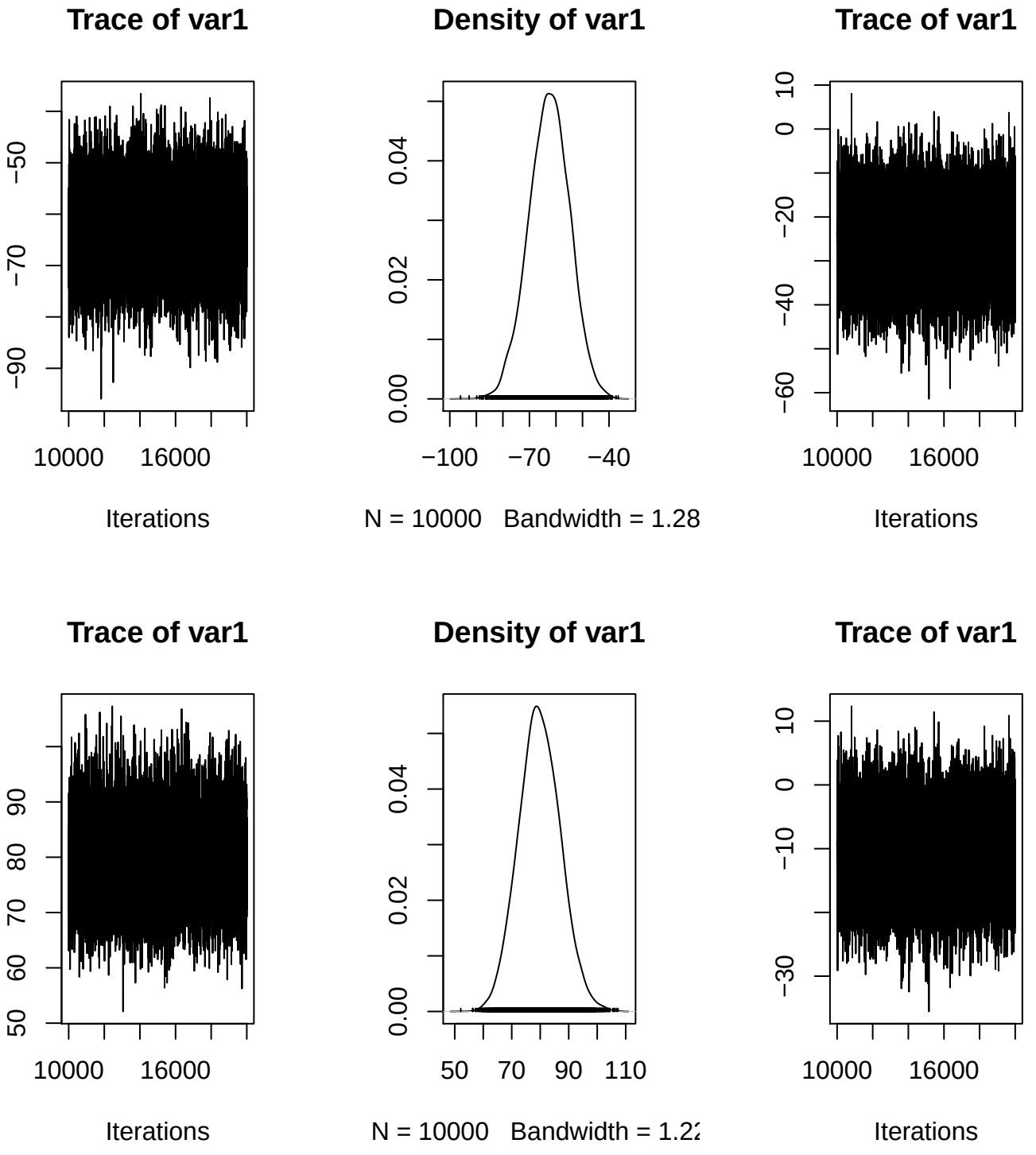

Trace of var1
60
70
80
90
100
10000
16000
Iterations

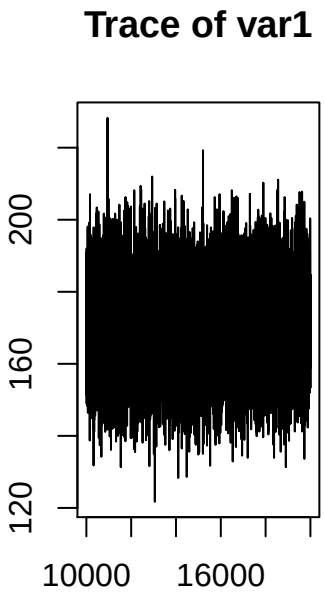

Density of var1
0.00
0.02
0.04
0.06
60
80
100
N = 10000 Bandwidth = 0.967

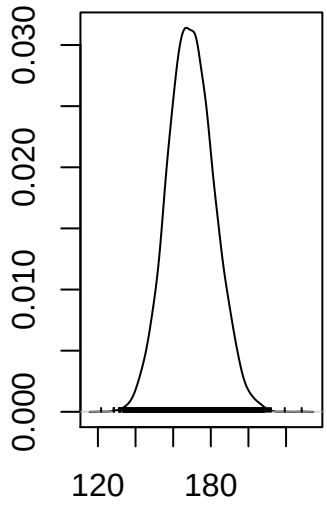

Trace of var1
−80
−70
−60
−50
10000
16000
Iterations

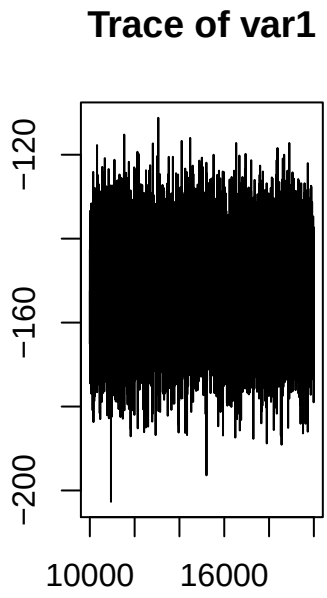

Trace of var1
Density of var1
Trace of var1
50
60
70
80
90
10000
16000
Iterations
0.00
0.02
0.04
50
70
90
110
N = 10000 Bandwidth = 1.22
−80
−60
−40
10000
16000
Iterations
Trace of var1
Density of var1
Trace of var1
−110
−90
−70
10000
16000
Iterations
0.00
0.02
0.04
0.06
−120
−90
−60
N = 10000 Bandwidth = 1.09
100
120
140
160
10000
16000
Iterations

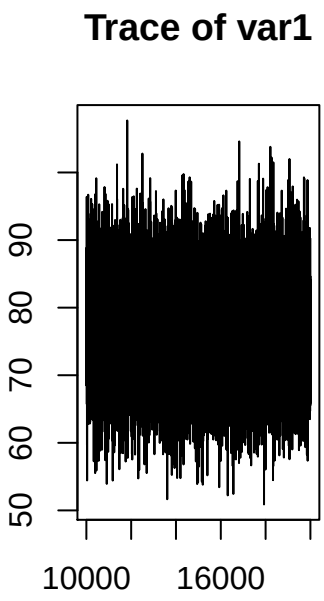

Trace of var1
Density of var1
Trace of var1
−50
−60
−70
−80
10000
16000
Iterations
0.06
0.04
0.02
0.00
−80
−60
−40
N = 10000 Bandwidth = 0.89
110
90
80
70
10000
16000
Iterations
Trace of var1
Density of var1
Trace of var1
10
0
−10
−30
10000
16000
Iterations
0.06
0.04
0.02
0.00
−40
−20
0
N = 10000 Bandwidth = 1.0
10
0
−10
−20
−30
10000
16000
Iterations

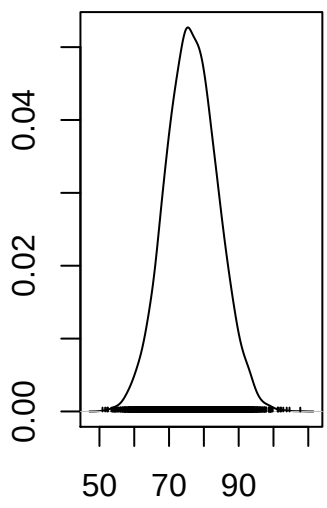

Trace of var1
Density of var1
Trace of var1
−50
−70
−90
10000
16000
Iterations
0.04
0.02
0.00
−100
−70
−40
N = 10000 Bandwidth = 1.28
10
0
−20
−40
−60
10000
16000
Iterations
Trace of var1
Density of var1
Trace of var1
90
80
70
60
50
10000
16000
Iterations
0.04
0.02
0.00
50
70
90
110
N = 10000 Bandwidth = 1.22
10
0
−10
−30
10000
16000
Iterations

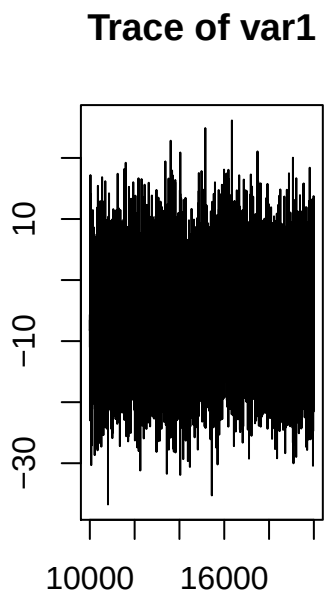

Trace of var1
120
160
200
10000
16000
Iterations

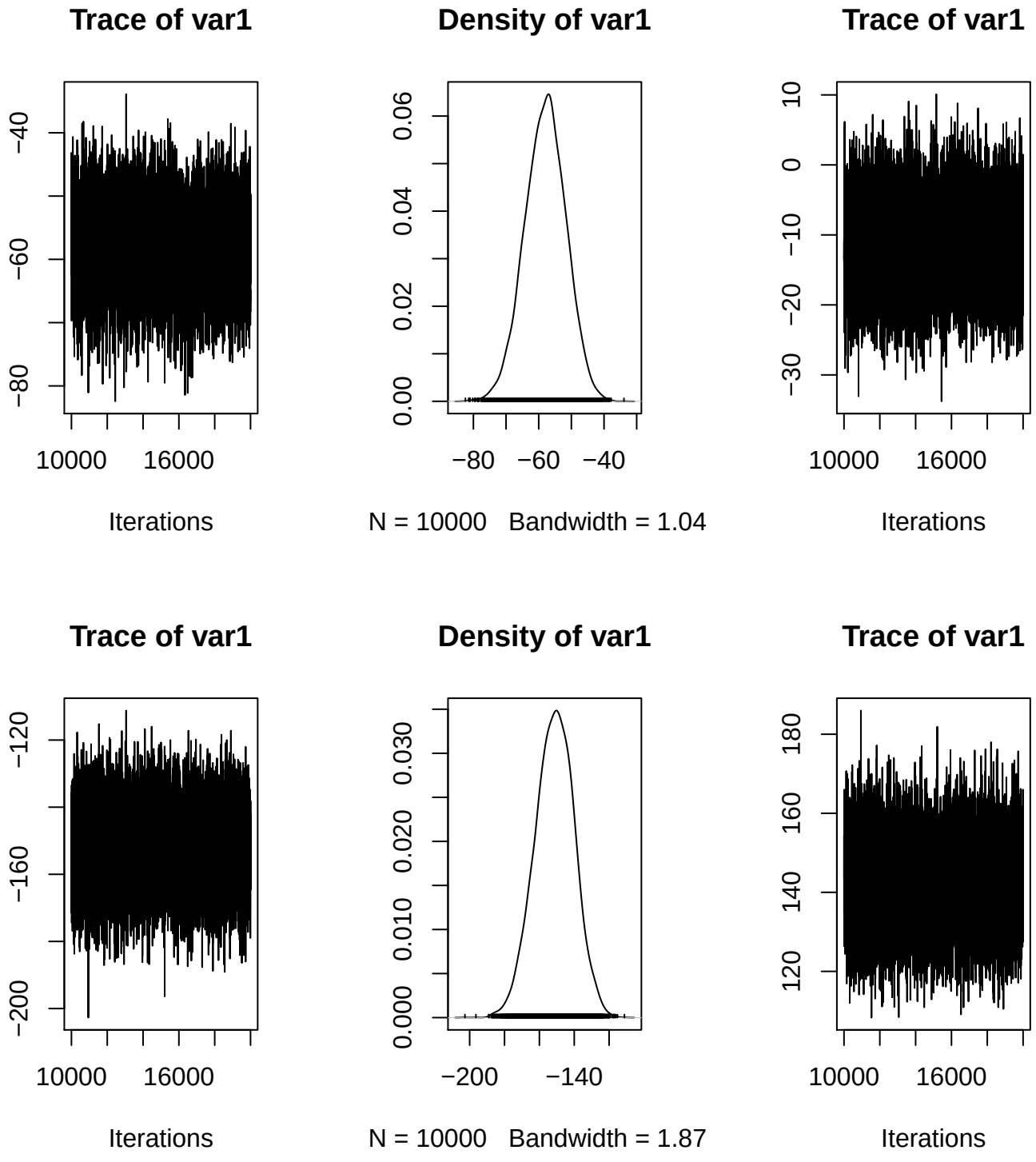

Density of var1
0.000
0.010
0.020
0.030
120
180
N = 10000 Bandwidth = 2.07

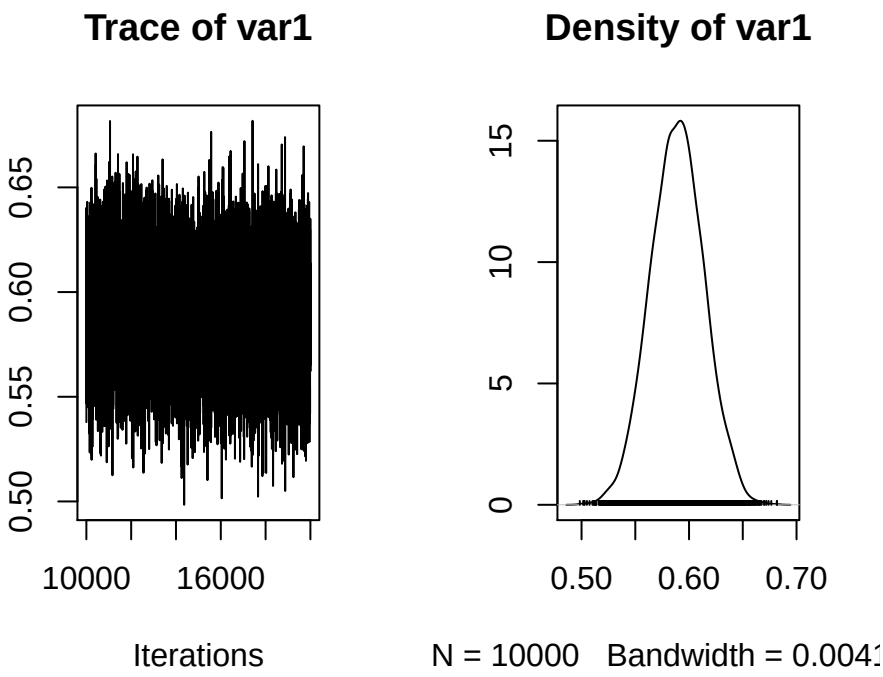

Trace of var1
−200
−160
−120
10000
16000
Iterations


Trace of var1
50
60
70
80
90
10000
16000
Iterations

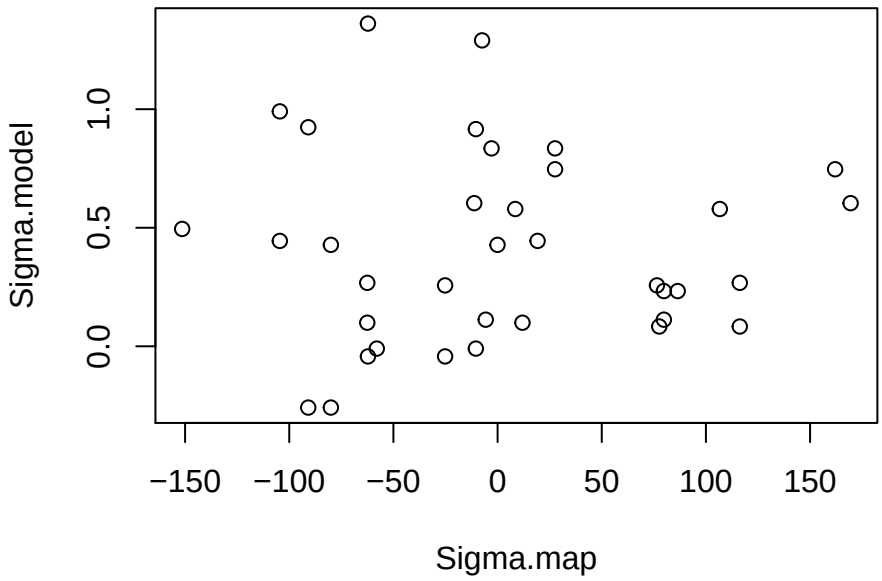

Density of var1
0.00
0.02
0.04
50
70
90
N = 10000 Bandwidth = 1.25


Trace of var1
−30
−10
10
10000
16000
Iterations


Trace of var1
Density of var1
Trace of var1
−40
−60
−80
10000
16000
Iterations
0.06
0.04
0.02
0.00
−80
−60
−40
N = 10000 Bandwidth = 1.04
10
0
−10
−20
−30
10000
16000
Iterations
Trace of var1
Density of var1
Trace of var1
−120
−160
−200
10000
16000
Iterations
0.030
0.020
0.010
0.000
−200
−140
N = 10000 Bandwidth = 1.87
180
160
140
120
10000
16000
Iterations




```
Sigma.map <- unlist(posterior.means)[1:36]
Sigma.model <- unlist(sig$mat)
plot(Sigma.map, Sigma.model)
```



```r
rho.map <- unlist(posterior.means)[43]

nu.map <- unlist(posterior.means)[37:42]
```