Incorporating GRM

Meera

2023-04-19

Import R Libraries

```
library(rjags)
```

```
## Loading required package: coda
```

```
## Linked to JAGS 4.3.1
```

```
## Loaded modules: basemod,bugs
```

```
library(MASS)
library(kinship2)
```

```
## Loading required package: Matrix
```

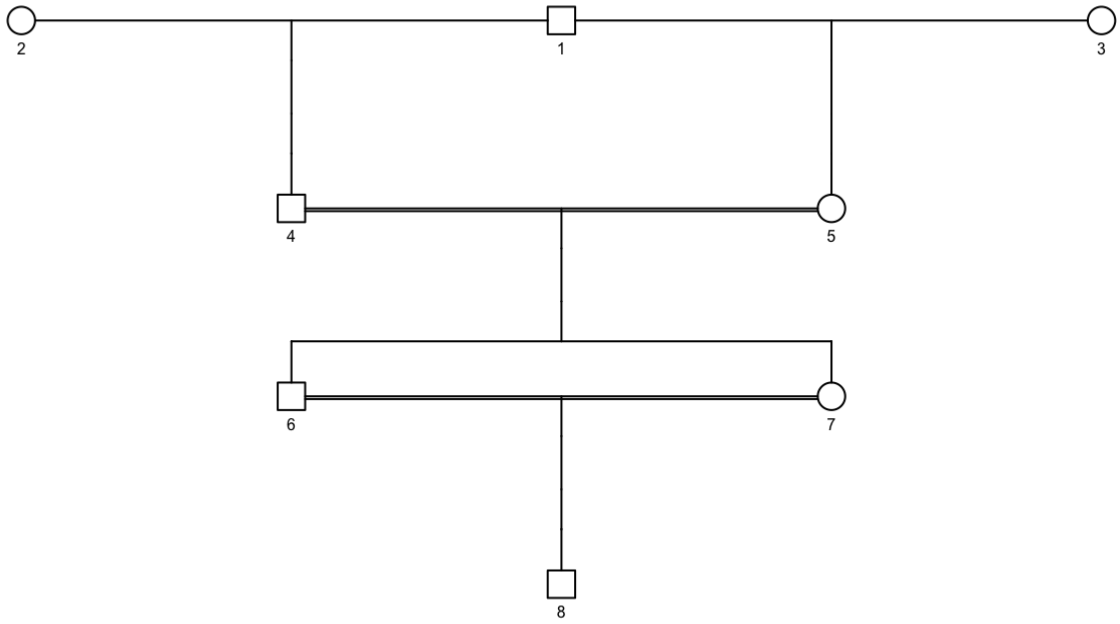
```
## Loading required package: quadprog
```

```
library(ggplot2)
```

Simulating Growth Curves

Simulating pedigree

```
sample.ped = rbind(c(1,1,0,0,1), c(1,2,0,0,2), c(1,3,0,0,2), c(1,4,1,2,1),
                   c(1,5,1,3,2), c(1,6,4,5,1), c(1,7,4,5,2), c(1,8,6,7,1))
colnames(sample.ped) = c("ped","id", "father", "mother", "sex")
sample.ped = as.data.frame(sample.ped)
# sample.ped
pedAll <- pedigree(id=sample.ped$id, dadid=sample.ped$father,
                   momid=sample.ped$mother, sex = sample.ped$sex)
plot.pedigree(pedAll, cex = 0.5)
```




```
GRM = kinship(pedAll)
print(GRM)
```

```
##      1     2     3      4      5      6      7       8
## 1 0.50 0.000 0.000 0.2500 0.2500 0.2500 0.2500 0.25000
## 2 0.00 0.500 0.000 0.2500 0.0000 0.1250 0.1250 0.12500
## 3 0.00 0.000 0.500 0.0000 0.2500 0.1250 0.1250 0.12500
## 4 0.25 0.250 0.000 0.5000 0.1250 0.3125 0.3125 0.31250
## 5 0.25 0.000 0.250 0.1250 0.5000 0.3125 0.3125 0.31250
## 6 0.25 0.125 0.125 0.3125 0.3125 0.5625 0.3125 0.43750
## 7 0.25 0.125 0.125 0.3125 0.3125 0.3125 0.5625 0.43750
## 8 0.25 0.125 0.125 0.3125 0.3125 0.4375 0.4375 0.65625
```

Setting up variables

```
ngeno = dim(GRM)[1] # number of genotypes
nrep = 3 # number of replicates
# for reproducibility
seed = 13
set.seed(seed)
genotype = 1:ngeno
replicate = 1:nrep
```

Simulating growth curves

$$[b, d, e] \sim MVN(\,[-0.5, 0.5, 20]\,,\ \Sigma \otimes \mathrm{GRM})$$

$$\Sigma = \text{inverse Wishart}\left(S = \begin{bmatrix} 100 & & \\ & 10 & \\ & & 10 \end{bmatrix}, \text{df} = 4\right)$$

$$y_{i,j} \sim MVN\left(\frac{d_i}{1 + exp(b_i(T - e_i))}, \begin{bmatrix} 0.001 & & \\ & \ddots & \\ & & 0.001 \end{bmatrix}\right)$$

```
R = diag(c(100,10,10))
df = 4

set.seed(seed)
sigma = solve(rWishart(n = 1, df = df, Sigma = R)[,,1])
sigmaGRM = sigma %x% GRM
mean = c(-0.5,0.5,20)
mvmean = c(rep(mean[1],ngeno), rep(mean[2],ngeno), rep(mean[3],ngeno))
set.seed(seed)
params = mvrnorm(1, mvmean, sigmaGRM)

p = matrix(nrow = ngeno, ncol = 3)
for (i in 1:3) {
  p[,i] = params[(((i-1) * ngeno) + 1) : (i * ngeno)]
}

paramnames = c("b", "d", "e")
colnames(p) = paramnames
data = as.data.frame(cbind(genotype, p))

alldata = c()
time = seq(0, 40,5)
var_y = 0.001
sigma_y = diag(var_y, length(time))
for (i in 1:ngeno) {
  subdata = data[data$genotype == i,] # pick the genotype-specific b,d,e
  mu_arr = (subdata$d)/(1 + exp(subdata$b*(time-subdata$e))) # growth curve as mean
  set.seed(seed)
  y = mvrnorm(nrep, mu_arr, sigma_y) # n replicates
  for (j in 1:nrep) {
    alldata = rbind(alldata, cbind(rep(i, length(time)), rep(j, length(time)),time, y
[j,]))
  }
}

alldata = as.data.frame(alldata)
colnames(alldata) = c("genotype", "rep", "time", "y")
alldata$rep = as.character(alldata$rep)
head(alldata)
```

```
##   genotype rep time           y
## 1        1   1    0 0.019685327
## 2        1   1    5 0.058829898
## 3        1   1   10 0.004296943
## 4        1   1   15 0.069317333
## 5        1   1   20 0.370015287
## 6        1   1   25 0.757903566
```

Summary of simulated growth curves

```
print("covariance matrix (sigma) for growth curve parameters")
```

```
## [1] "covariance matrix (sigma) for growth curve parameters"
```

```
sigma
```

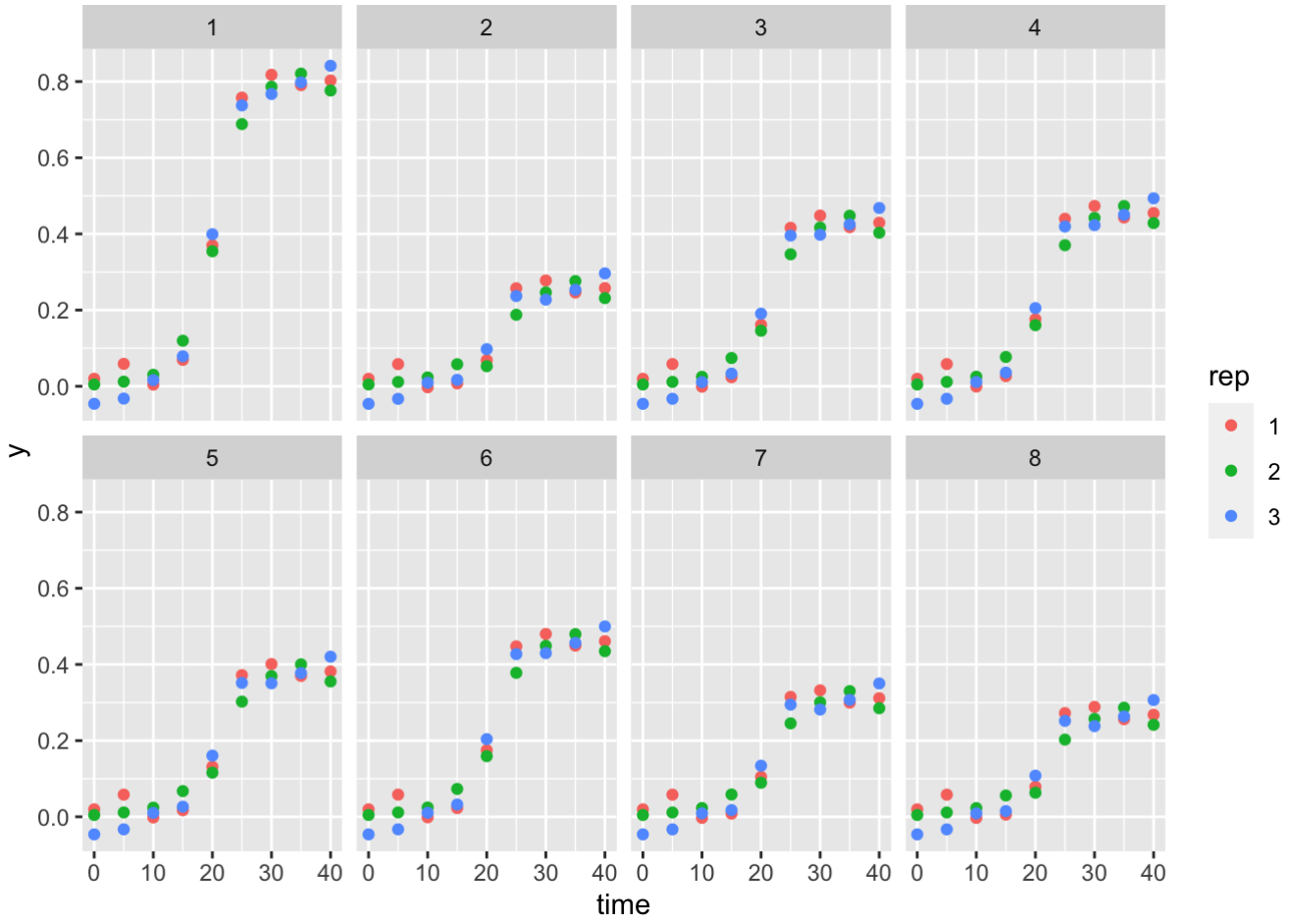
```
##              [,1]        [,2]         [,3]
## [1,]  0.010302941  0.02597693 -0.003015684
## [2,]  0.025976926  0.10803708 -0.042113755
## [3,] -0.003015684 -0.04211376  0.043887623
```

```
for (i in 1:3){
  print(paste("realized variance for",paramnames[i], ":", var(p[,i])))
}
```

```
## [1] "realized variance for b : 0.00187614623686034"
## [1] "realized variance for d : 0.0303302358452298"
## [1] "realized variance for e : 0.0210605455277084"
```

Visualise growth curves

```
plot = ggplot() + geom_point(data = alldata, aes(x = time, y = y, color = rep))
plot = plot + facet_wrap(~genotype, nrow = 2)
plot
```



JAGS model

Reformat as JAGS input

```
nrT = length(time)

for (geno in genotype) {
  ally = c()
  for (rep in replicate) {
    ally = c(ally, list(alldata[alldata$genotype == geno & alldata$rep == rep,]$y))
  }
  ally = do.call(rbind, ally)
  if(geno == 1) {
    res = array(ally, dim = c(nrep, length(time), 1))
  } else {
    res = array(c(res, ally), dim = c(nrep, length(time), dim(res)[3] + 1))
  }
}
dim(res) # should be n (rows) x nrT (columns) x ngeno (arrays)
```

```
## [1] 3 9 8
```

Setting up JAGS code

$$y_{i,j} \sim MVN\left(\frac{d_i}{1 + exp(b_i(T - e_i))}, sd \cdot I\right)$$

$$sd \sim \text{Unif}(0, 100)$$

$$[b_i, d_i, e_i] \sim MVN(\vec{\mu},\ \Sigma)$$

$$\text{prior for } \tau\ (\Sigma^{-1}) \sim \text{Wishart}\left(R = \begin{bmatrix} 100 & & \\ & 100 & \\ & & 100 \end{bmatrix}, \text{df} = 4\right)$$

$$z_{i,j} \sim N(0, 1) \text{ for j} \in (b, d, e) \text{ and i} \in 1 \ldots \text{nRep}$$

$$\text{GRM } = CC^T$$

$$\vec{b} = C\overrightarrow{z_b} + \mu_b \sim N(\mu_b, \text{GRM})$$

$$\vec{d} = C\overrightarrow{z_d} + \mu_d \sim N(\mu_d, \text{GRM})$$

$$\vec{e} = C\overrightarrow{z_e} + \mu_e \sim N(\mu_e, \text{GRM})$$

$$\mu_j \sim N(0, 100) \text{ for j} \in (b, d, e)$$

```
C = t(chol(GRM))
jagsData <- list("Y"=res,"N"=ngeno,"nrT"=nrT,"time"=time, "nRep" = nrep, "C" = C)

model_string <- "model {
  # dimensions of Y matrix = nRep x nrT x N
  for (i in 1:N) { # loop over genotypes
    for (t in 1:nrT) { # loop over time points
      for (j in 1:nRep) {
        Y[j, t, i] ~ dnorm(d[i] / (1 + exp(b[i] * (time[t] - e[i]))), 1/pow(sd,2))
      }
    }
  }
  sd ~ dunif(0, 100)

  for (i in 1:N) { z_b[i] ~ dnorm(0,1) }
  b[1:N] <- (C[,] %*% z_b[1:N]) + mu_b
  for (i in 1:N) { z_d[i] ~ dnorm(0,1) }
  d[1:N] <- (C[,] %*% z_d[1:N]) + mu_d
  for (i in 1:N) { z_e[i] ~ dnorm(0,1) }
  e[1:N] <- (C[,] %*% z_e[1:N]) + mu_e

  mu_b ~ dnorm(0, 0.01)
  mu_d ~ dnorm(0, 0.01)
  mu_e ~ dnorm(0, 0.01)
}"
```

Running JAGS

```
# nadapt = 100000
# nupdate = 100000
# nimplement = 50000
nadapt = 10000
nupdate = 10000
nimplement = 5000

parameters = c("b", "d", "e", "sd", "mu_b", "mu_d", "mu_e")
model <- jags.model(textConnection(model_string),
                    data=jagsData, n.chains=1, n.adapt = nadapt, inits = list(.RNG.name
= "base::Wichmann-Hill",.RNG.seed = seed))
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 216
##    Unobserved stochastic nodes: 28
##    Total graph size: 721
##
## Initializing model
```

```
update(model, n.iter=nupdate)
samples <- coda.samples(model, variable.names=parameters, n.iter=nimplement)
s = as.data.frame(as.matrix(samples))
head(s[,1:5])
```

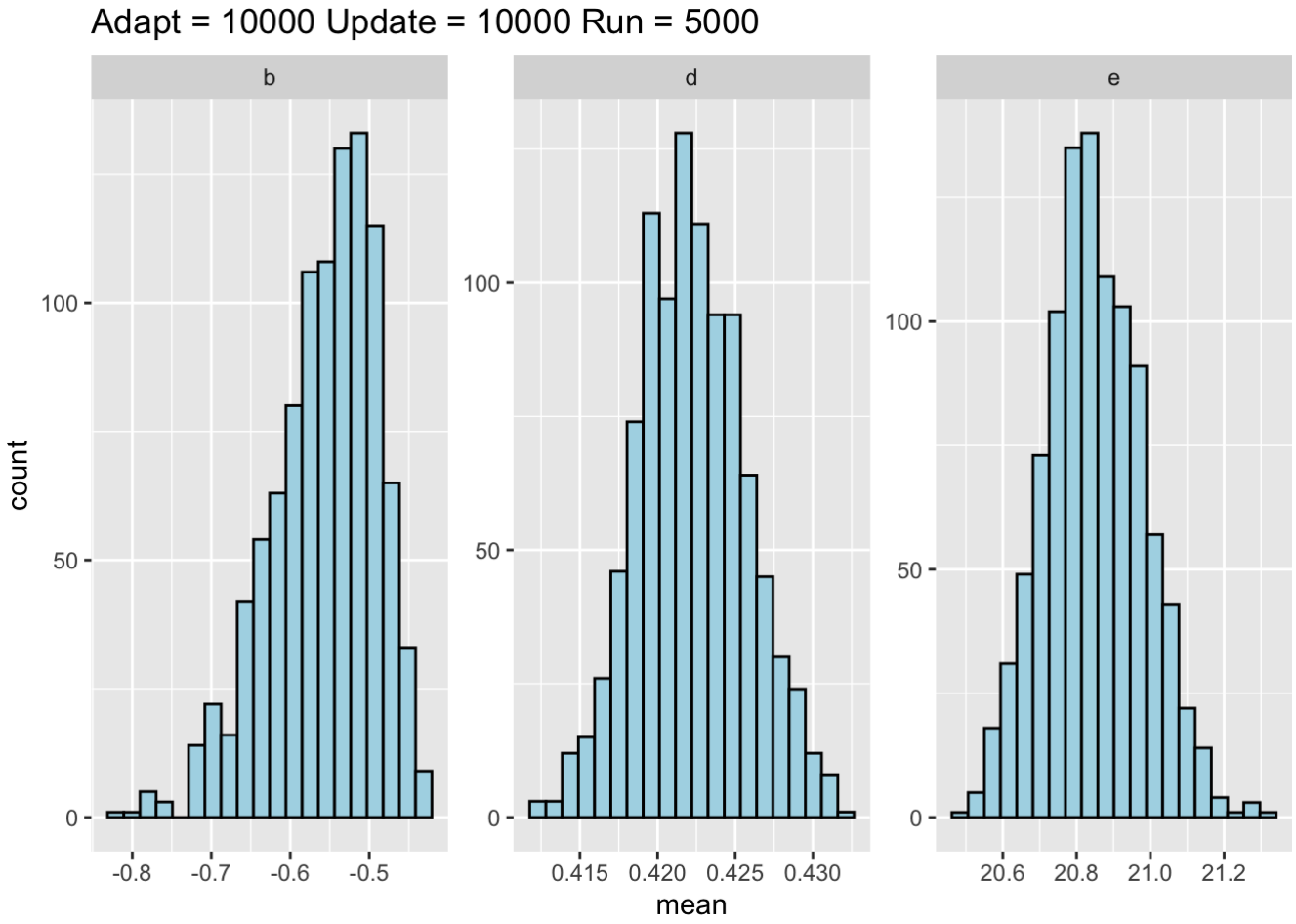
```
##         b[1]       b[2]       b[3]       b[4]       b[5]
## 1 -0.4496516 -0.5521799 -0.4044290 -0.4772189 -0.6108649
## 2 -0.4693135 -0.5385171 -0.4246355 -0.4915796 -0.5605271
## 3 -0.4413231 -0.5113017 -0.4363391 -0.4355764 -0.5000431
## 4 -0.4297605 -0.5284057 -0.4100643 -0.3729678 -0.5083649
## 5 -0.4191142 -0.5516555 -0.4586225 -0.4161126 -0.6608451
## 6 -0.4504968 -0.5557136 -0.4011187 -0.4947949 -0.5741408
```

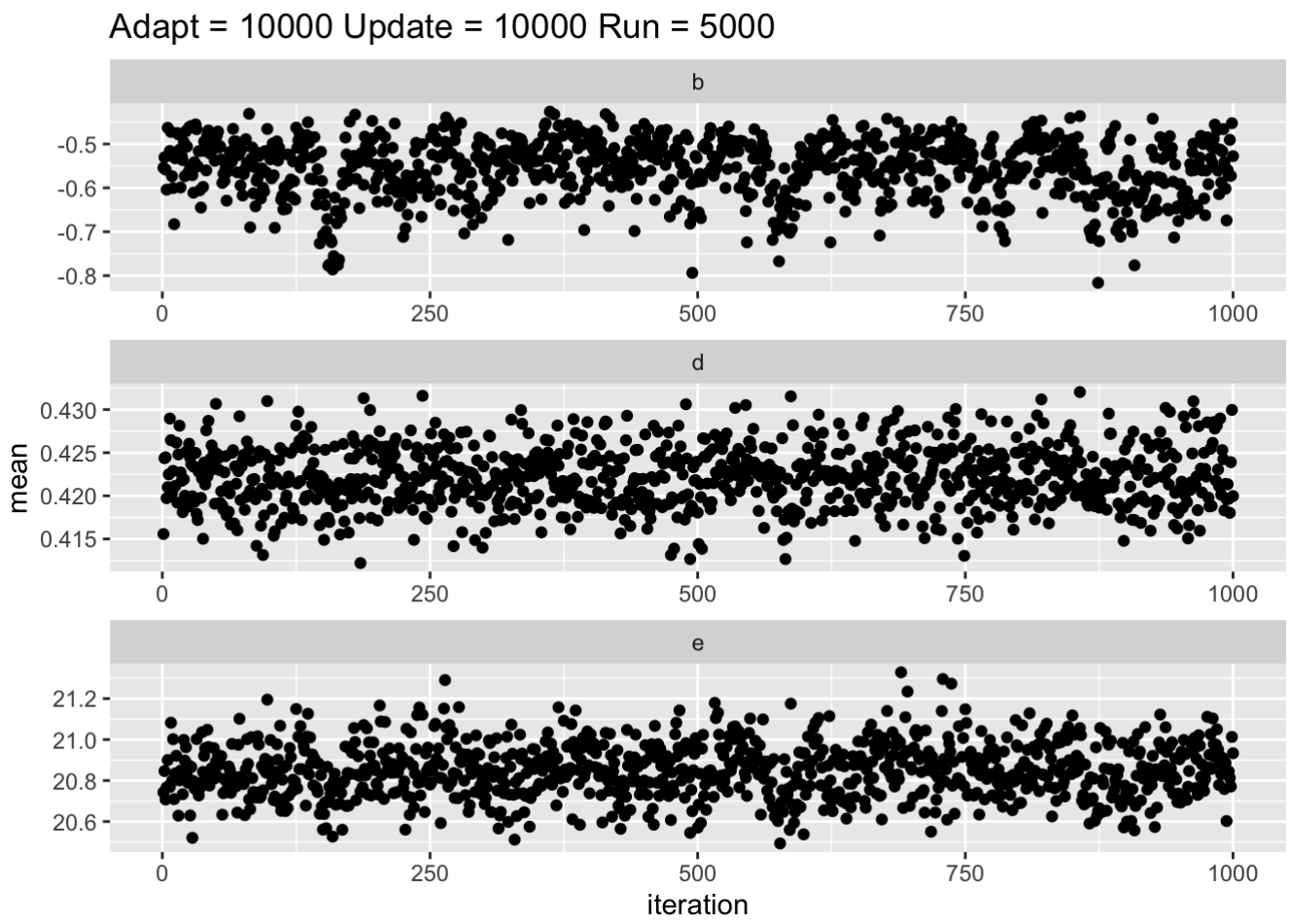
Plotting estimates

Plotting b,d,e estimates

```
## [1] "thinning to use every 5"
```

```
## [1] 8
```

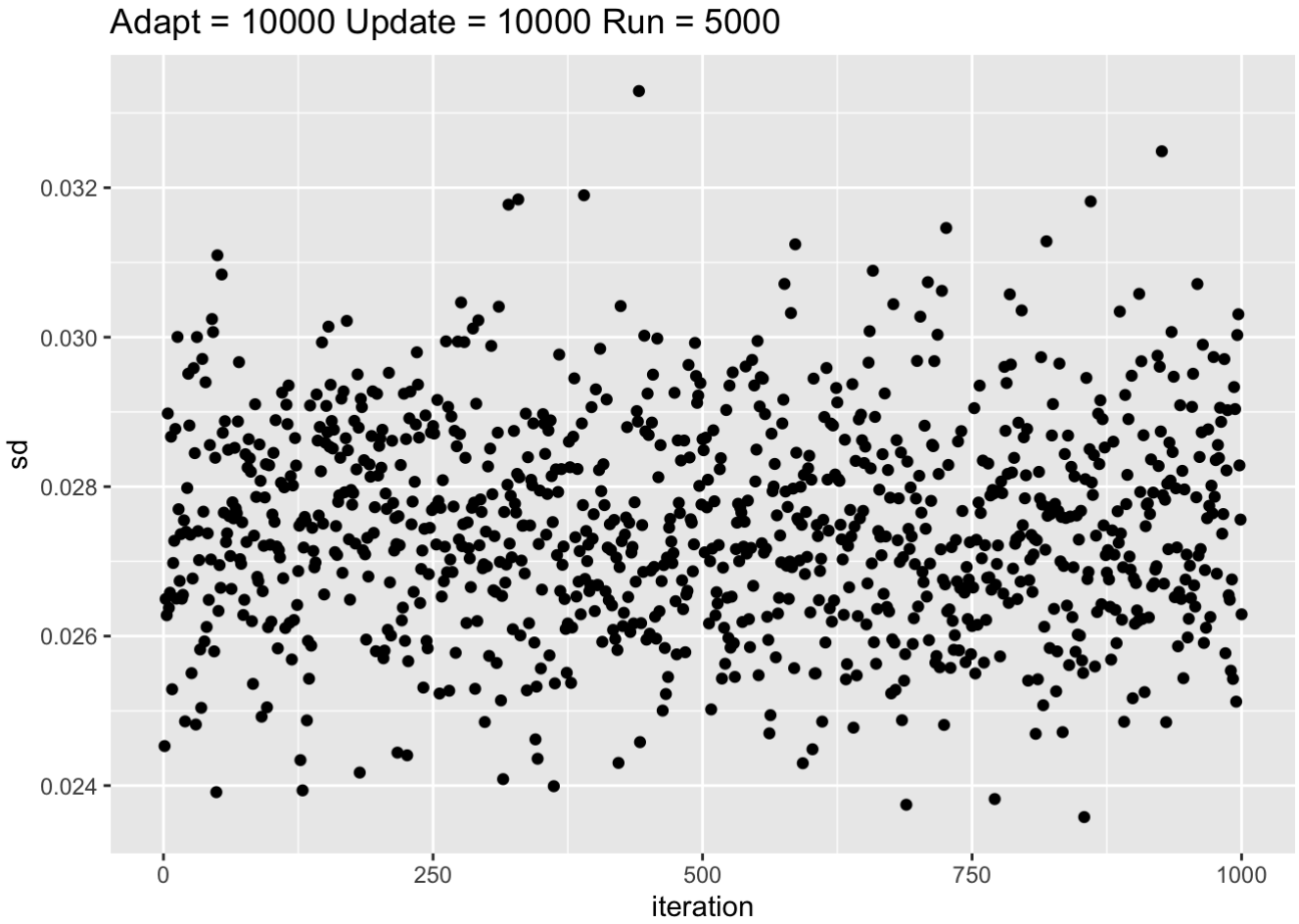


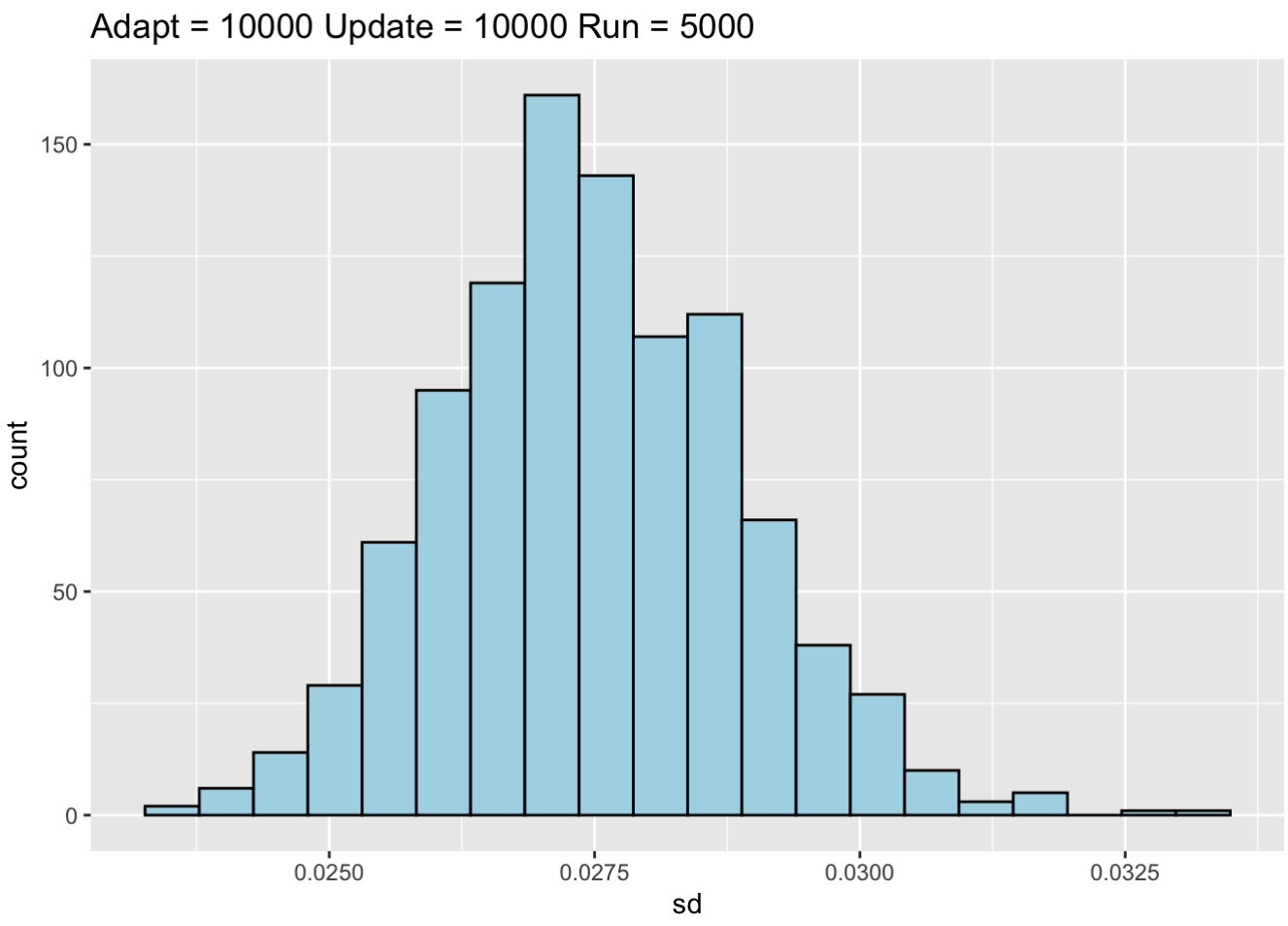


###

Plotting sd estimates

```
## [1] "thinning to use every 5"
```


Adapt = 10000 Update = 10000 Run = 5000
sd
0.032
0.030
0.028
0.026
0.024
0
250
500
750
1000
iteration


Adapt = 10000 Update = 10000 Run = 5000
count
150
100
50
0
0.0250
0.0275
0.0300
0.0325
sd