Covarying Growth Curve Parameters

Meera

2023-04-19

Import R Libraries

```
library(rjags)
```

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

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

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

```
library(ggplot2)
library(MASS)
```

Simulating Growth Curves

Setting up variables

```
ngeno = 100 # number of genotypes
nrep = 3 # number of replicates
# for reproducibility
seed = 13
set.seed(seed)
genotype = 1:ngeno
replicate = 1:nrep
```

Simulating growth curves

$$[b_i, d_i, e_i] \sim MVN(\ [-0.5, 0.5, 20]\ ,\ \Sigma)$$

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

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

```
R = diag(c(100,10,10)) # scaled matrix for Wishart dist
df = 4 # degrees of freedom for Wishart dist
set.seed(seed)
sigma = solve(rWishart(n = 1, df = df, Sigma = R)[,,1])

set.seed(seed)
mvmean = c(-0.5,0.5,20) # means for b,d,e parameters
params = mvrnorm(ngeno, mvmean, sigma)

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

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)

#  removing outlier genotype, identified in e_outliers.R
bad_geno = c(50)
alldata = alldata[!(alldata$genotype %in% bad_geno),]
dim(alldata)
```

```
## [1] 2673    4
```

```
genotype = genotype[!(genotype %in% bad_geno)]
ngeno = length(genotype)

head(alldata)
```

```
##   genotype rep time           y
## 1        1   1    0  0.019620266
## 2        1   1    5  0.058177537
## 3        1   1   10 -0.002092789
## 4        1   1   15  0.013284432
## 5        1   1   20  0.099179687
## 6        1   1   25  0.301163781
```

Summary of simulated growth curves

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

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

```
sigma
```

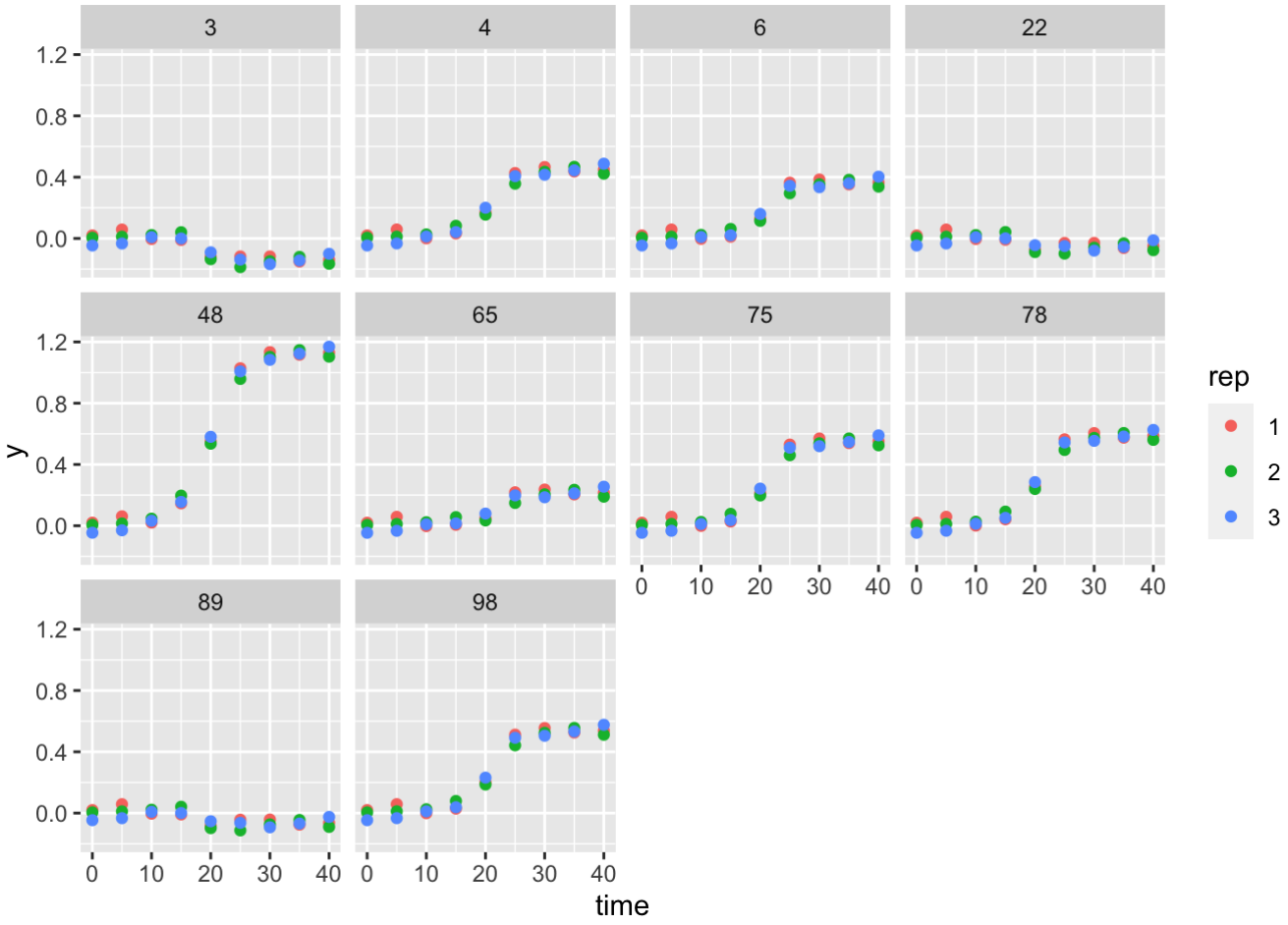
```
##              [,1]        [,2]         [,3]
## [1,]  0.010302941  0.02597693 -0.003015684
## [2,]  0.025976926  0.10803708 -0.042113755
## [3,] -0.003015684 -0.04211376  0.043887623
```

```
paramnames = c("b", "d", "e")
for (i in 1:3){
  print(paste("realized variance for",paramnames[i], ":", var(params[,i])))
}
```

```
## [1] "realized variance for b : 0.0109912839587263"
## [1] "realized variance for d : 0.099910929787099"
## [1] "realized variance for e : 0.0440119387932997"
```

Visualise some growth curves

```
set.seed(seed)
plot = ggplot() + geom_point(data = alldata[alldata$genotype %in% sample(genotype, 1
0),], aes(x = time, y = y, color = rep))
plot = plot + facet_wrap(~genotype)
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 99
```

Setting up JAGS code

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

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

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

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

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

```
R = diag(c(100,100,100)) # uninformative scaled matrix
df = 4

jagsData <- list("Y"=res,"N"=ngeno,"nrT"=nrT,"time"=time, "nRep" = nrep, "R" = R, "df" =
df)

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(params[i,2] / (1 + exp(params[i,1] * (time[t] - params[i,
3]))), 1/pow(sd,2))
      }
    }
  }
  sd ~ dunif(0, 100)

  for (i in 1:N) { # loop over genotypes
    params[i, 1:3] ~ dmnorm(mu, TAU[1:3, 1:3])
  }
  TAU ~ dwish(R, df)

  for (j in 1:3) {
    mu[j] ~ dunif(0,100)
  }

  # output covariance matrix
  VCOV <- inverse(TAU)
  vars[1] <- VCOV[1,1]; vars[2] <- VCOV[2,2]; vars[3] <- VCOV[3,3]
  cov[1] <- VCOV[1,2]; cov[2] <- VCOV[1,3]; cov[3] <- VCOV[2,3]
}"
```

Running JAGS

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

parameters = c("params", "sd", "vars", "cov", "mu")
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: 2673
##    Unobserved stochastic nodes: 104
##    Total graph size: 7566
##
## Initializing model
```

```
## Warning in jags.model(textConnection(model_string), data = jagsData, n.chains =
## 1, : Adaptation incomplete
```

```
update(model, n.iter=nupdate)
samples <- coda.samples(model, variable.names=parameters, n.iter=nimplement)
```

```
## NOTE: Stopping adaptation
```

```
s = as.data.frame(as.matrix(samples))
head(s[,1:5])
```

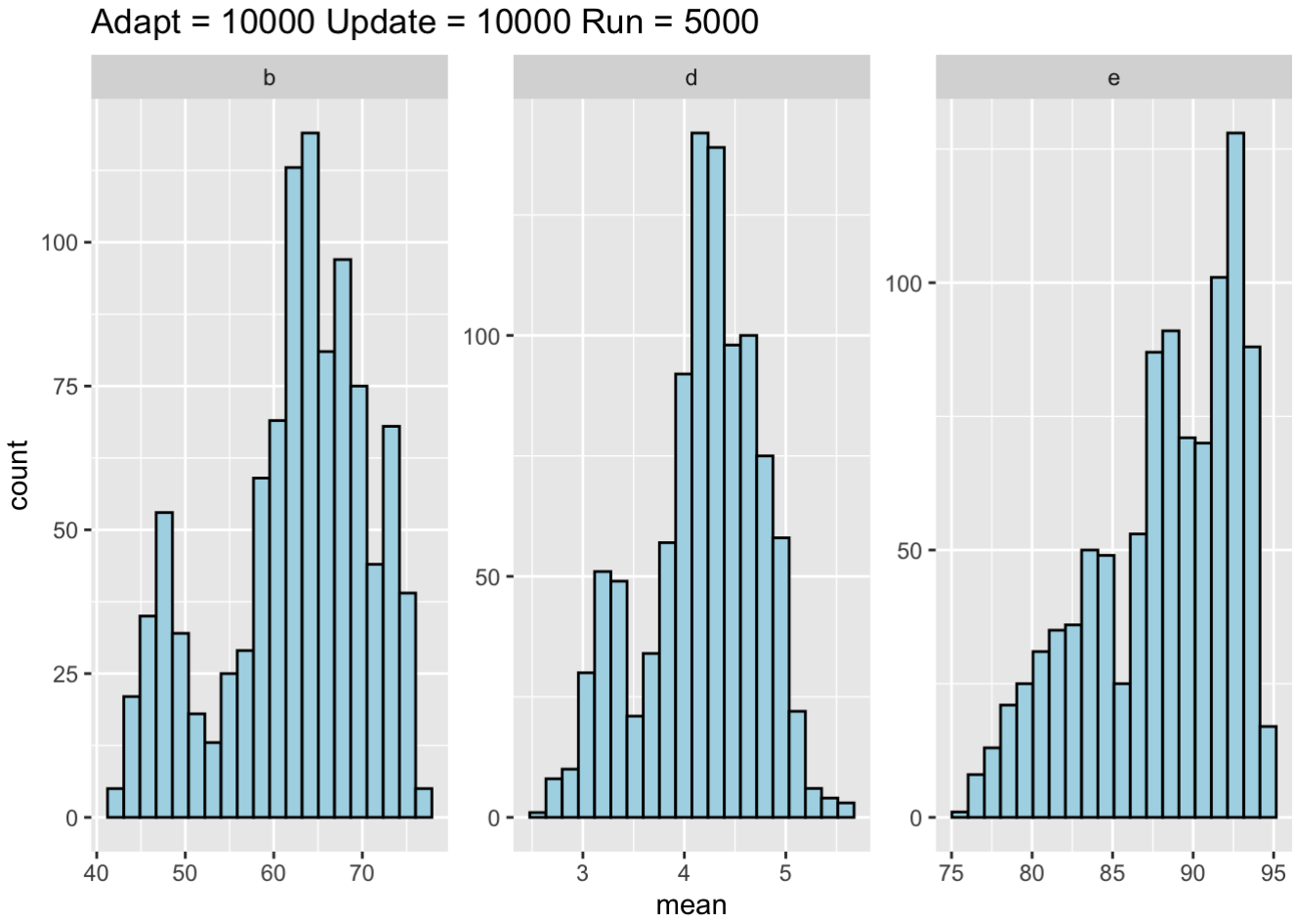
```
##       cov[1]     cov[2]       cov[3]    mu[1]     mu[2]
## 1  0.2241184  -9.072780  0.008785224 70.59267 0.1824007
## 2  0.6881172  -6.911033  0.195732291 70.36942 0.1884730
## 3 -1.2746557  -7.452264 -1.204721458 70.03694 0.2307052
## 4 -1.2745960 -10.331567  0.390435666 69.63325 0.4760297
## 5  0.3223769  -8.086990  1.402252250 71.27365 0.2388657
## 6  0.5849843  -8.921648 -0.395539278 70.00420 0.1838828
```

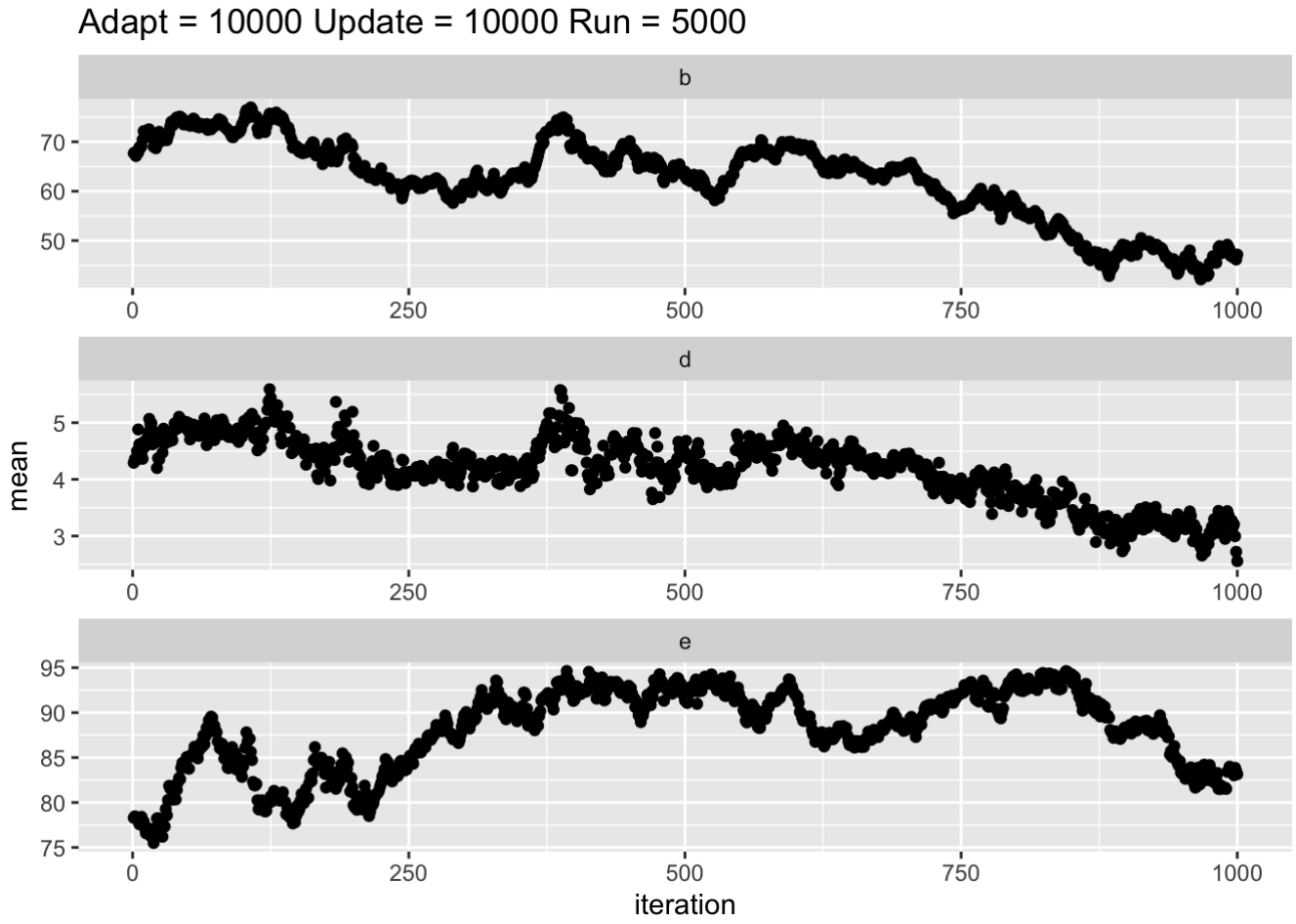
Plotting estimates

Plotting b,d,e estimates

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

```
## [1] 99
```

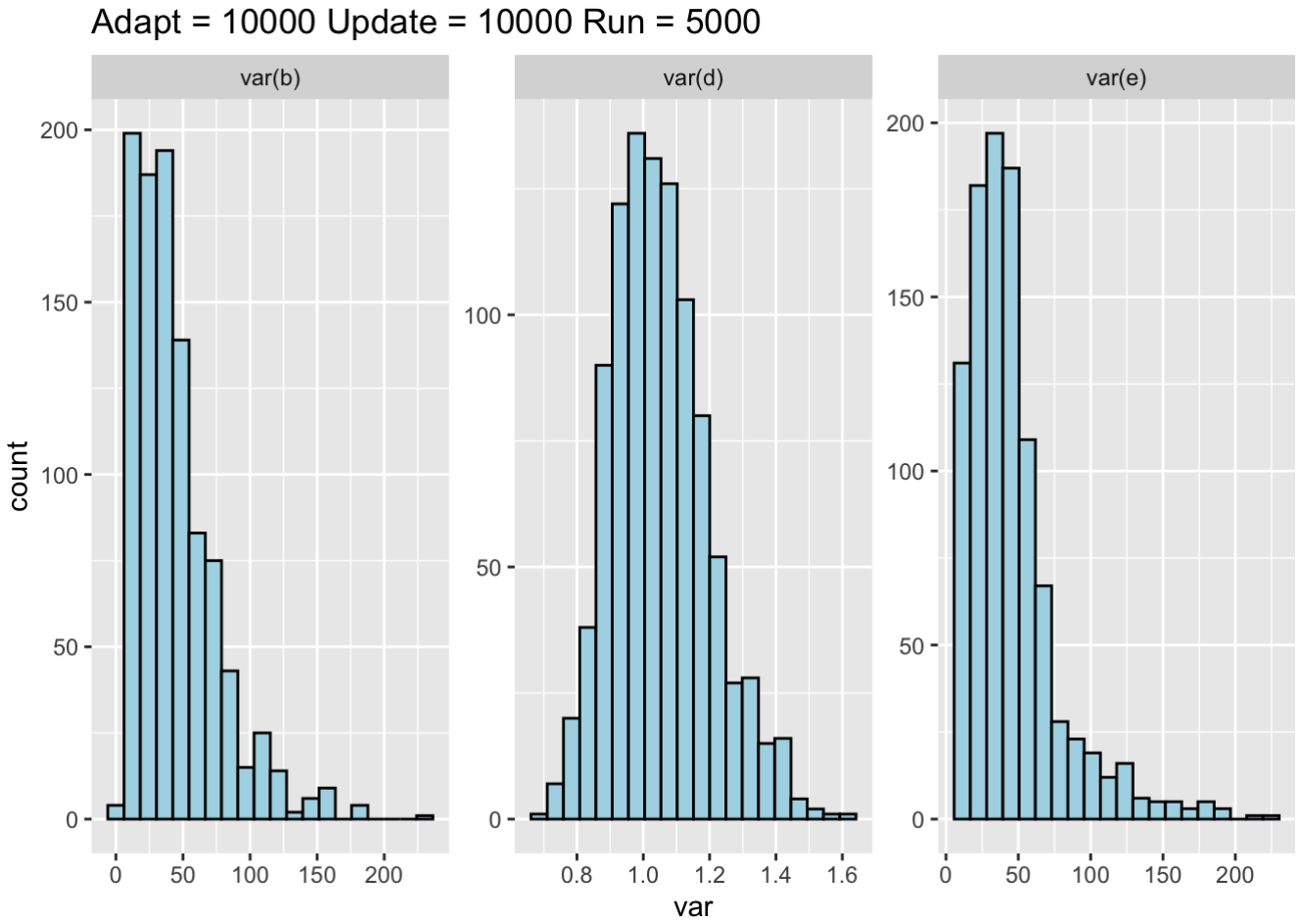


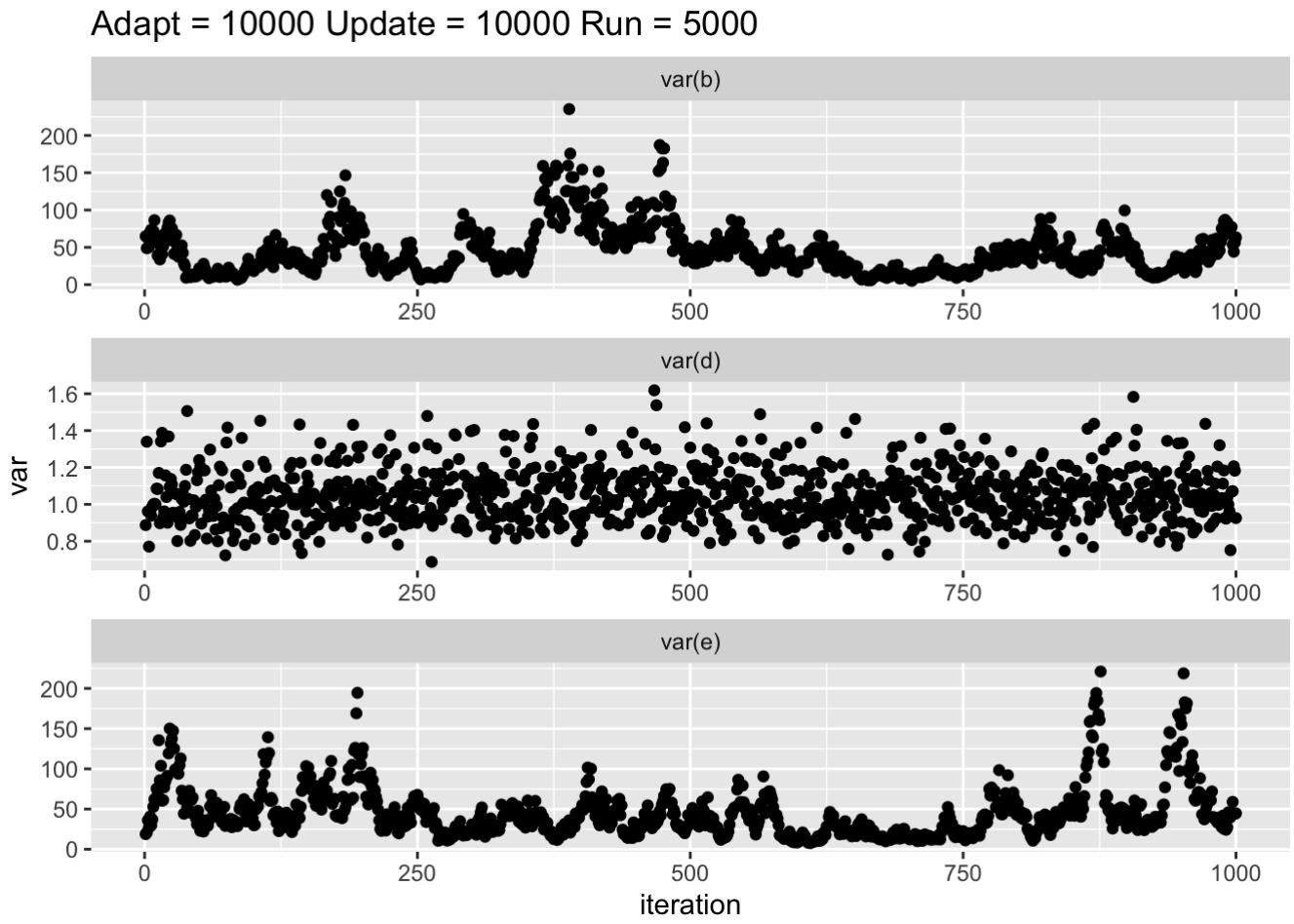


###

Plotting variance of b,d,e

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

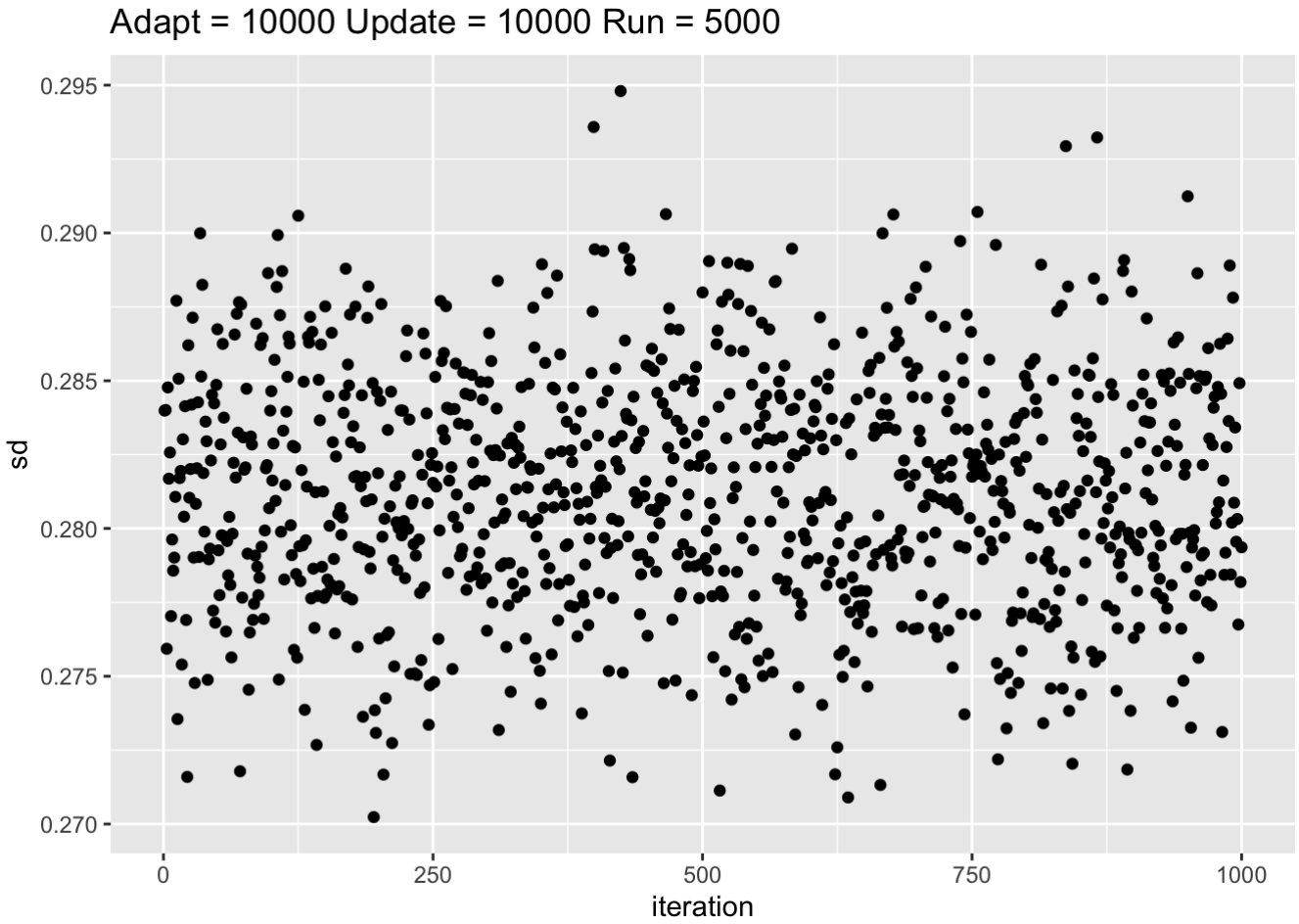


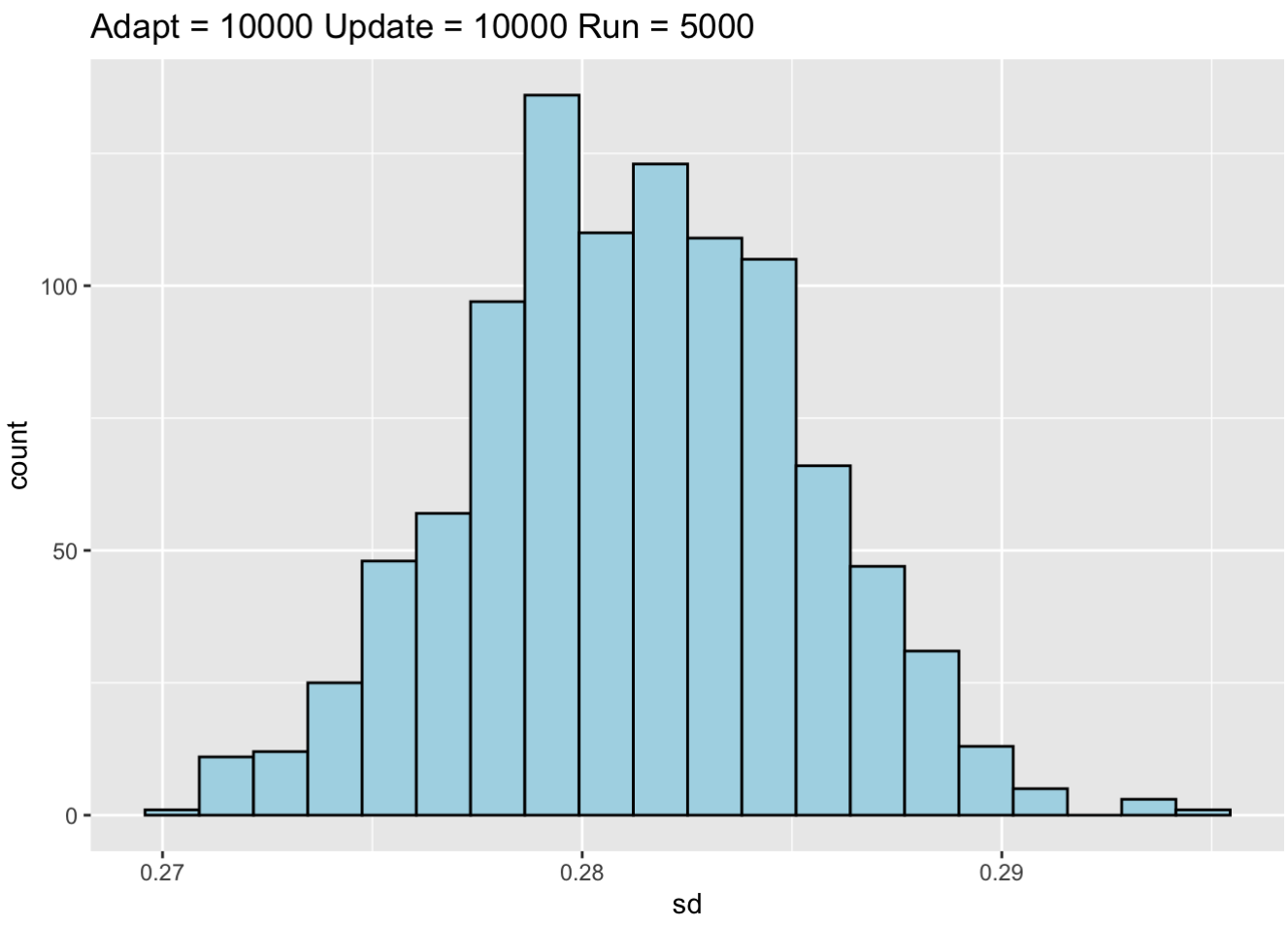


###

Plotting sd estimates

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


Adapt = 10000 Update = 10000 Run = 5000
0.295
0.290
0.285
0.280
0.275
0.270
sd
0
250
500
750
1000
iteration


Adapt = 10000 Update = 10000 Run = 5000
100
50
0
count
0.27
0.28
0.29
sd