2019R1 Applied Bayesian Methods (STAT6106)
Assignment 5

Yiu Chung WONG 1155017920

```
set.seed(6106);
```

1

Prior distributions for parameters are: $p(\beta_g) \sim 1$, $p(c_g) \sim 1$, $p(d_g) \sim 1$, $p(\sigma_g^2) \sim \frac{1}{\sigma^2}$

We adopt Gibbs sampling + MH algorithm to sample these parameters:

During the i-th step

1. Sample $\beta_{g(i+1)}$ from condition distribution $p(\beta_{g(i+1)}|c_{g(i)}, d_{g(i)}, \sigma^2_{g(i)}, Y)$, with acceptance ratio 1.
2. Sample $c_{g(i+1)}$ from condition distribution: $p(c_{g(i+1)}|\beta_{g(i+1)}, d_{g(i)}, \sigma^2_{g(i)}, Y)$, with acceptance ratio 1.
3. Sample $d_{g(i+1)} \sim \mathcal{TN}(1, 0.5, 0, \infty)$ with acceptance probability

$$\min\left(\frac{L(\beta_{g(i+1)}, c_{g(i+1)}, d_{g(try)}, \sigma^2_{g(i)})}{L(\beta_{g(i+1)}, c_{g(i+1)}, d_{g(i)}, \sigma^2_{g(i)})} \cdot \frac{dlnorm(d_{g(i)}, d_{g(try)}, \sigma^2)}{dlnorm(d_{g(try)}, d_{g(i)}, \sigma^2)}, 1\right)$$

4. Sample $\sigma^2_{g(i+1)}$ from condition distribution: $p(\sigma^2_{g(i+1)}|\beta_{g(i+1)}, c_{g(i+1)}, d_{g(i+1)}, Y)$, with acceptance ratio 1.

2

```
q2.Y.Means <- as.data.frame(matrix(0, nrow = nrow(Y), ncol = ncol(Y)))
q2.para.Means <- matrix(0, nrow = nrow(Y), ncol = 7)
q2.para.SDs <- matrix(0, nrow = nrow(Y), ncol = 7)

para.names <- c("ag", "bg", "fg", "hg", "cg", "dg", "sigma2g")
colnames(q2.para.Means) <- para.names
colnames(q2.para.SDs) <- para.names

#for each gene
for (gene in 1:nrow(Y)){

  #Obtain data for a gene
  Yg<-t(Y[gene,])

  #Initial values, only one chain per gene
  betg0 <- c(0.1, 0.1, 0.1, 0.1)
  cg0 <- 0.1
  dg0 <- 0.5
  sig20 <- 0.2

  #the matrix will store the parameter values of all iterations from one chain
  para<-matrix(NA, nrow=7, ncol=n)
  para[,1]<-c(betg0,cg0,dg0,sig20)

   for(i in 2:n) #iterations
   {
       #sampling each parameters iteratively
      betg0<-f.abfh(cg0, dg0, sig20)
      cg0<-f.c(betg0, dg0, sig20)
      dg0<-f.d(betg0, cg0, dg0, sig20)
      sig20<-f.sig(betg0, cg0, dg0)
       #store the parameter values of the current iteration
      para[,i]<-c(betg0,cg0,dg0,sig20)
   }

  #-----------------result analysis:  posterior summary--------------------------
  #--------------------------   mean method: get posterior   ---------------------
  # parameter estimation
  (Mean<-apply(para[,(n/2):n], 1, mean))
  (Sd<-apply(para[,(n/2):n], 1, sd))
  (b<-0.01*Mean[2])
  (A<-sqrt((Mean[3])^2+(Mean[4])^2))     #the Ag in Equation (1)
  (phi<- acos(Mean[3]/A))           #the phi_g in Equation (1)

  # posterior summary
  q2.para.Means[gene,] <- Mean
  q2.para.SDs[gene,] <- Sd

  # for fitting plot
  q2.Y.mean<-Fit(Mean[1:4], Mean[5], Mean[6])  #the fitted values
  q2.Y.Means[gene,] <- q2.Y.mean
  (Dist.mean<-t(Yg-q2.Y.mean)%*%(Yg-q2.Y.mean)) #sum square of the residue
```

```
}
```

Means

```
q2.para.Means
```

```
##                ag           bg          fg          hg          cg        dg     sigma2g
##  [1,]  0.060419331 -0.007594882  0.50325539 -0.80310961 -0.72473219 0.4798957 0.020422921
##  [2,] -0.140282687  0.091298704  0.39388068 -1.01477915 -0.85613904 0.6340927 0.019997777
##  [3,] -0.097379774  0.028012666  0.25325629 -0.80431587 -0.76029492 0.4321895 0.016009793
##  [4,] -0.017288056  0.073362905  0.14555103 -0.38303285 -0.45669170 0.4625706 0.009887766
##  [5,] -0.500026706  0.138934308  0.89038870 -1.79643163 -0.10370004 0.5256826 0.050466383
##  [6,] -0.029682252  0.047296708  0.01597768 -0.34985011  0.43006021 0.3528368 0.006970898
##  [7,] -0.028144112 -0.020801780  0.04920586 -0.03557512  0.21917210 0.5335483 0.036717864
##  [8,] -0.062986501  0.024909828  0.03740261  0.31734891  0.98313652 0.4132764 0.005918343
##  [9,] -0.014989389  0.100905278 -0.12777092 -1.33690157 -0.65105265 0.5319810 0.028362313
## [10,] -0.154710791  0.081288906  0.51007204 -1.23515525 -0.74912452 0.7062936 0.031803667
## [11,] -0.182373812  0.088336156 -1.02876888 -0.84780595  0.60110772 0.4849611 0.017531778
## [12,] -0.142777285  0.080614127 -0.78290003 -0.71971749  0.56307950 0.6028338 0.019259536
## [13,]  0.144627483 -0.026093457  0.35188236 -0.68963452 -0.50218055 0.6631579 0.022816020
## [14,]  0.051143403 -0.052205166  0.35992658 -0.67483750  0.04223925 0.3551326 0.009643693
## [15,]  0.145611212  0.003663660  0.22690759 -0.95915258 -1.02681156 0.5805570 0.029251938
## [16,]  0.148520625  0.023428881  0.50706813 -0.57428669 -1.36742995 0.5256048 0.021500303
## [17,] -0.025651444  0.094024470  0.03291568 -1.23960228  0.11232055 0.3864950 0.026537701
## [18,] -0.080911212  0.082560820 -0.18930815 -0.51459851  0.18758799 0.4963999 0.016217658
## [19,] -0.309739230  0.041706946  0.75985552 -1.92243245 -0.55820812 0.6046460 0.043666580
## [20,]  0.006935686 -0.005128531  0.13560001 -0.61774388  0.11551434 0.3916695 0.010171792
```

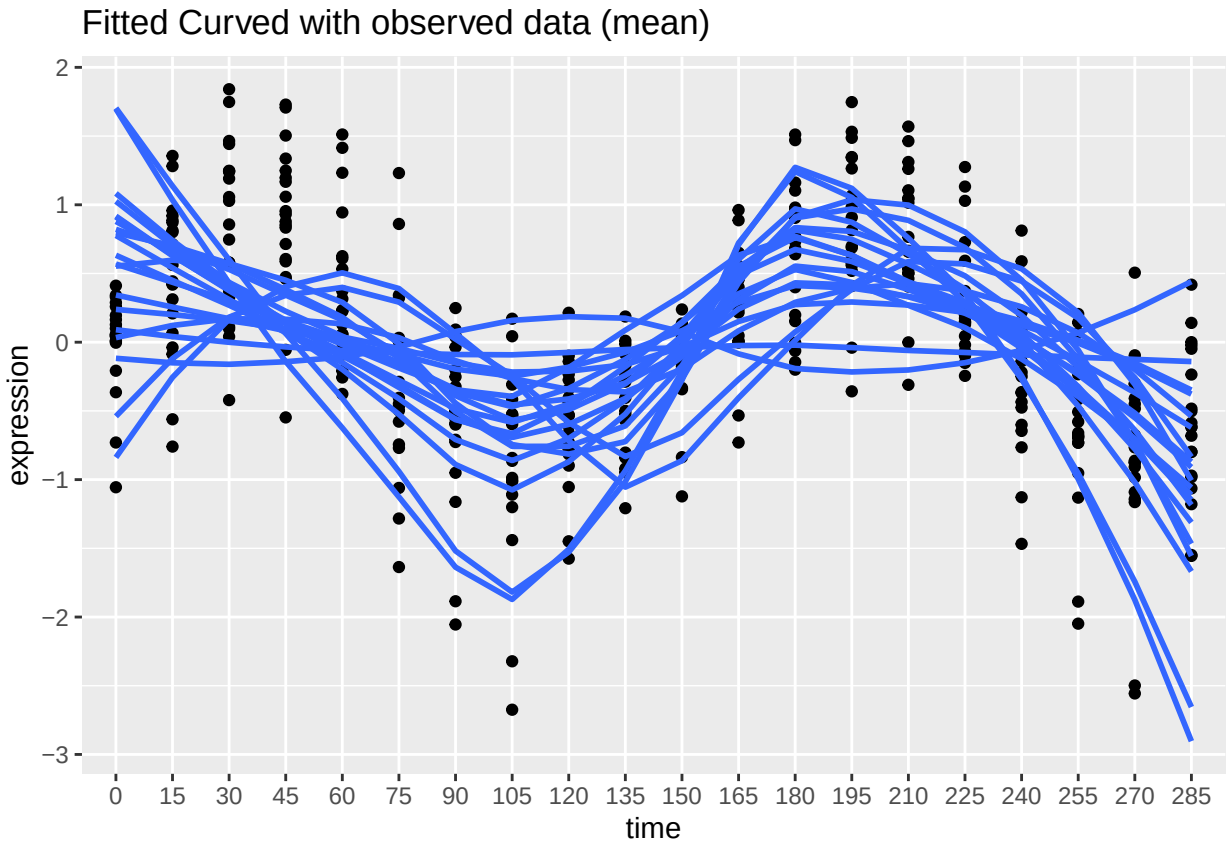
Sd

```
q2.para.SDs
```

```
##               ag         bg         fg         hg       cg        dg     sigma2g
##  [1,] 0.08064299 0.02679066 0.04815182 0.04341916 1.505059 0.4128451 0.005671988
##  [2,] 0.07926844 0.02597002 0.04940655 0.04601610 1.252772 0.4702685 0.005822084
##  [3,] 0.07088643 0.02216295 0.04046641 0.03934422 1.383486 0.3802308 0.005332900
##  [4,] 0.05660096 0.01868983 0.03387169 0.03203748 1.302687 0.4451608 0.002950137
##  [5,] 0.13101865 0.03918699 0.07427564 0.06936060 1.627667 0.5911508 0.018332400
##  [6,] 0.04433264 0.01390409 0.02746011 0.02665943 1.362495 0.3340048 0.001924363
##  [7,] 0.11393856 0.03379844 0.06185732 0.05608745 1.419309 0.4919399 0.011428791
##  [8,] 0.04376032 0.01541329 0.02660048 0.02523383 1.129848 0.2816360 0.001706712
##  [9,] 0.09854628 0.02992813 0.05836441 0.05614855 1.377879 0.4634355 0.008956403
## [10,] 0.10011977 0.03309159 0.05848877 0.05626264 1.158605 0.5608061 0.010137954
## [11,] 0.07679055 0.02348880 0.04328714 0.04189789 1.541396 0.5430222 0.005792056
## [12,] 0.08256382 0.02595447 0.04704434 0.04284246 1.345471 0.5553842 0.006842055
## [13,] 0.08290040 0.02526034 0.05296306 0.04610806 1.452963 0.7645902 0.006810036
## [14,] 0.05726281 0.01774667 0.03202185 0.03264284 1.295123 0.3319730 0.002912244
## [15,] 0.09876175 0.03058709 0.05502158 0.05585555 1.297671 0.4374596 0.009567400
## [16,] 0.08293786 0.02746098 0.04823114 0.04922924 1.160044 0.2633934 0.006194148
## [17,] 0.08978065 0.02964159 0.05487771 0.05012458 1.460810 0.4110668 0.008239239
## [18,] 0.06838035 0.02073931 0.04216928 0.04067703 1.390068 0.6198100 0.005333866
```

```
## [19,] 0.11859465 0.03813171 0.07109185 0.06505125 1.294029 0.5110104 0.012942264
## [20,] 0.05779145 0.01814152 0.03178269 0.03300214 1.409643 0.4170841 0.002878372
```

```r
#---------------result analysis:   fitting plot ---------

# Observed data
Y.Observes <- cbind(1:20, Y)
colnames(Y.Observes) <- c("gene_no", 15 * 0:19)

observe_melt <- melt(Y.Observes, id="gene_no", value.name = "expression", variable.name="time")
observe_melt$gene_no <- as.factor(observe_melt$gene_no)

# Fitted data
q2.Y.Means <- cbind(1:20, q2.Y.Means)
colnames(q2.Y.Means) <- c("gene_no", 15 * 0:19)

#ggplot likes melted data frames
q2.mean_melt <- melt(q2.Y.Means, id="gene_no", value.name = "expression", variable.name="time")
q2.mean_melt$gene_no <- as.factor(q2.mean_melt$gene_no)

# Plotting
ggplot(observe_melt, aes(x = time, y = expression, group = gene_no)) +
  geom_point() +
  geom_smooth(data = q2.mean_melt, aes(x = time, y = expression, group = gene_no), se = FALSE) +
  ggtitle("Fitted Curved with observed data (mean)")
```



3

```
X<-matrix(0, nrow=20, ncol=4) #a = 0
X[,2]<-0 #b = 0
X[,3]<-cos(mu*Tim)
X[,4]<- -sin(mu*Tim)

f.fh0<-function(cg, dg, sig2)
{
   temp1<-cbind(Tim-dg,0)
   temp2<-(apply(temp1, 1, min))^2
   Ybetg<-Yg-cg*temp2
   Vmat<-solve(t(X)%*%X/sig2 + diag(rep(1,4))/alp1)
   Mean<-Vmat%*%t(X)%*%Ybetg/sig2
   Mean + chol(Vmat)%*%matrix(c(0,0,rnorm(2)),ncol=1) #First 2 parameters 0
}
```

```
q3.Y.Means <- as.data.frame(matrix(0, nrow = nrow(Y), ncol = ncol(Y)))
q3.para.Means <- matrix(0, nrow = nrow(Y), ncol = 7)
q3.para.SDs <- matrix(0, nrow = nrow(Y), ncol = 7)

para.names <- c("ag", "bg", "fg", "hg", "cg", "dg", "sigma2g")
colnames(q3.para.Means) <- para.names
colnames(q3.para.SDs) <- para.names

#for each gene
for (gene in 1:nrow(Y)){

  #Obtain data for a gene
  Yg<-t(Y[gene,])

  #Initial values, only one chain per gene
  betg0 <- c(0, 0, 0.1, 0.1)
  cg0 <- 0
  dg0 <- 0
  sig20 <- 0.2

  #the matrix will store the parameter values of all iterations from one chain
  para<-matrix(NA, nrow=7, ncol=n)
  para[,1]<-c(betg0,cg0,dg0,sig20)

   for(i in 2:n) #iterations
   {
       #sampling each parameters iteratively
      betg0<-f.fh0(cg0, dg0, sig20)
      cg0<-0
      dg0<-0
      sig20<-f.sig(betg0, cg0, dg0)
       #store the parameter values of the current iteration
      para[,i]<-c(betg0,cg0,dg0,sig20)
   }

  #-----------------result analysis:  posterior summary--------------------------
  #-------------------------   mean method: get posterior   ---------------------
```

```
  # parameter estimation
  (Mean<-apply(para[,(n/2):n], 1, mean))
  (Sd<-apply(para[,(n/2):n], 1, sd))
  (b<-0.01*Mean[2])
  (A<-sqrt((Mean[3])^2+(Mean[4])^2))     #the Ag in Equation (1)
  (phi<- acos(Mean[3]/A))           #the phi_g in Equation (1)

  # posterior summary
  q3.para.Means[gene,] <- Mean
  q3.para.SDs[gene,] <- Sd

  # for fitting plot
  q3.Y.mean<-Fit(Mean[1:4], Mean[5], Mean[6])  #the fitted values
  q3.Y.Means[gene,] <- q3.Y.mean
  (Dist.mean<-t(Yg-q3.Y.mean)%*%(Yg-q3.Y.mean)) #sum square of the residue
}
```

Means

```
q3.para.Means[,c("fg", "hg", "sigma2g")]
```

```
##                 fg          hg     sigma2g
##  [1,]  0.488330003 -0.81378210 0.019599609
##  [2,]  0.350295072 -0.97773747 0.022443263
##  [3,]  0.235786334 -0.79061630 0.017062073
##  [4,]  0.110157360 -0.36016244 0.014631199
##  [5,]  0.884401283 -1.72404839 0.102144972
##  [6,]  0.004355144 -0.33128909 0.007664347
##  [7,]  0.064283883 -0.04266268 0.033528935
##  [8,]  0.045414492  0.33048299 0.005935544
##  [9,] -0.185104381 -1.29860360 0.037835561
## [10,]  0.474336722 -1.20613012 0.033877221
## [11,] -1.039502317 -0.80931838 0.018911126
## [12,] -0.784721934 -0.68088287 0.018827871
## [13,]  0.340085415 -0.70819859 0.025812062
## [14,]  0.375503989 -0.69557385 0.009341647
## [15,]  0.190228580 -0.95910712 0.039112953
## [16,]  0.459008503 -0.57926808 0.036889350
## [17,]  0.001064837 -1.20844905 0.032744672
## [18,] -0.212382371 -0.48566341 0.017104997
## [19,]  0.747695518 -1.89434809 0.074636456
## [20,]  0.140803379 -0.61863787 0.008615552
```

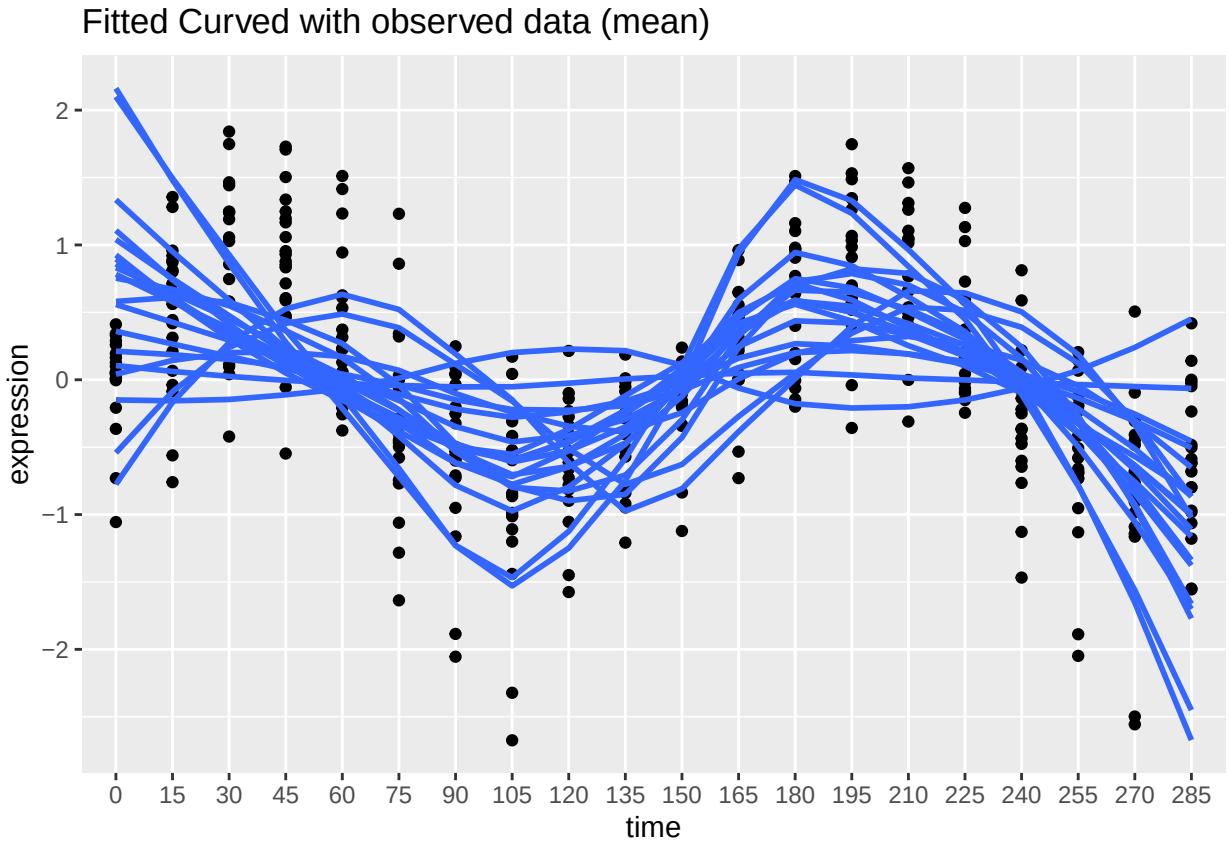
Sd

```
q3.para.SDs[,c("fg", "hg", "sigma2g")]
```

```
##               fg         hg     sigma2g
##  [1,] 0.04506425 0.04155891 0.004622475
##  [2,] 0.04724193 0.04683331 0.005428516
##  [3,] 0.04228888 0.04081332 0.004208593
```

```
##  [4,] 0.03743888 0.03932646 0.003641191
##  [5,] 0.09660569 0.09841908 0.024176960
##  [6,] 0.02736237 0.02698194 0.001902940
##  [7,] 0.06111717 0.05948312 0.008547588
##  [8,] 0.02506858 0.02372375 0.001490171
##  [9,] 0.06490629 0.05914783 0.009333289
## [10,] 0.05991677 0.05686292 0.008331221
## [11,] 0.04588205 0.04343068 0.004836137
## [12,] 0.04423927 0.04070006 0.004457133
## [13,] 0.05292409 0.05074466 0.006394207
## [14,] 0.03165599 0.03011728 0.002319498
## [15,] 0.05933119 0.06502161 0.008956244
## [16,] 0.06264689 0.05887524 0.008605466
## [17,] 0.05666959 0.05685903 0.008032392
## [18,] 0.04247198 0.04136752 0.004229169
## [19,] 0.08646407 0.08436225 0.018828034
## [20,] 0.02924413 0.02874400 0.001904453
```

```r
#---------------result analysis:   fitting plot ---------

#Fitted data
q3.Y.Means <- cbind(1:20, q3.Y.Means)
colnames(q3.Y.Means) <- c("gene_no", 15 * 0:19)

#ggplot likes melted data frames
q3.mean_melt <- melt(q3.Y.Means, id="gene_no", value.name = "expression", variable.name="time")
q3.mean_melt$gene_no <- as.factor(q3.mean_melt$gene_no)

# Plotting
ggplot(observe_melt, aes(x = time, y = expression, group = gene_no)) +
  geom_point() +
  geom_smooth(data = q3.mean_melt, aes(x = time, y = expression, group = gene_no), se = FALSE) +
  ggtitle("Fitted Curved with observed data (mean)")
```


Fitted Curved with observed data (mean)
expression
2
1
0
−1
−2
0
15
30
45
60
75
90
105
120
135
150
165
180
195
210
225
240
255
270
285
time