05-15-2023

Hengde Ouyang

2023-05-15

Code Last Week

```r
library(survival)
library(mvtnorm)
```

```r
# Compute theta
THETA <- function(A,beta){
  n = dim(A)[1]
  Abeta <- A%*%beta
  Abeta_bar = A%*%beta- mean(Abeta)
  theta = Abeta_bar/as.numeric(t(Abeta_bar)%*%Abeta_bar)
  theta[is.nan(theta)] = 0
  return(theta)
}
```

```r
# Compute C-index type loss function
HarrellC_Wmat <- function(Y, delta, tau) {
    n <- length(Y)
    Wmat <- matrix(0, n, n)
    for(i in 1:n) {
       if(delta[i] != 0) {
          Wmat[i,] <- delta[i]*(Y[i] < Y)*(Y[i] < tau)
       }
    }
    Wmat <- Wmat/sum(Wmat)
    return(Wmat)
}


HarrellC <- function(theta, Wmat) {
  theta <- c(theta)
  Thetamat <- (outer(theta, theta, FUN="-") > 0)
  # Case that theta are all 0
  if (sum(Wmat*Thetamat) == 0){
    return (0.0001)
  }

  return(sum(Wmat*Thetamat))
}
```

Tasks this week

1. Uno's C Statistics

```
# Wight matrix for Uno's C-statistics

UnoC_Wmat <- function(Y, delta, tau) {

    # An index indicates whether the observation is censored
    censor = ifelse(delta==0,1,0)

    # Censoring Distribution Estimate using Kaplan-Meier Estimator
    KM_G = survfit(formula = Surv(Y ,censor) ~ 1)

    # Get G(Y) for each observation
    # (Since G(Y) in KM_G is ordered we want each G(Y) to match original Y)
    G_y = KM_G$surv[match(Y,KM_G$time)]


    n <- length(Y)
    Wmat <- matrix(0, n, n)
    for(i in 1:n) {
       if(delta[i] != 0) {
          Wmat[i,] <- delta[i]*(Y[i] < Y)*(Y[i] < tau)*G_y[i]^(-2)
       }
    }
    Wmat <- Wmat/sum(Wmat)
    return(Wmat)
}
```

2. Modify the MH-Sampling

Newly update:
1. we have the burn-in option (B)
2. We can choose different values of eta
3. We can choose to use Harrell C or Uno C statistics

```
library(mvtnorm)
MH_Sampling <- function(Y,delta,tau,
                        A,beta0,sigma0,var.prop,
                        m,B,eta,
                        Wmat_option){


  accept = 0
  beta = beta0

  # What we want to record
  BETA = matrix(0,m,dim(A)[2])
  ThetaRecord <- matrix(0, m, length(Y))
  C_stat = c()
```

```
# For safety m>B
if (B>m){
  B = 0
}


# 0 means we use Harrell C statistics
# 1 means we use Uno C statistics
if (Wmat_option==0){
  Wmat <- HarrellC_Wmat(Y, delta, tau)
}else if (Wmat_option==1){
  Wmat <- UnoC_Wmat(Y, delta, tau)
}else{  # Other Possible C index...
  Wmat <- HarrellC_Wmat(Y, delta, tau)
}


for (i in 1:m){

  # Sample beta from proposal distribution
  beta.p = t(rmvnorm(1,beta,var.prop))


  # Compute theta from current and last iteration
  theta.p = THETA(A,beta.p)
  theta = THETA(A,beta)

  # Record theta from last iteration
  ThetaRecord[i,] <- theta

  # Compute C-statistics from current and last iteration
  HC.p = HarrellC(theta.p, Wmat)
  HC = HarrellC(theta, Wmat)

  # Record C-statistics from last iteration
  C_stat = c(C_stat,HC)


  # Compute log of MH ratio

  lrMH = eta*log(HC.p) +
        sum(dnorm(beta.p,beta0,sigma0,log=T))-
        eta*log(HC) -
        sum(dnorm(beta,beta0,sigma0,log=T))

    if (log(runif(1))<lrMH){
      beta = beta.p
      accept = accept + 1
    }
    BETA[i,] = beta

}
```

```
  if (B == 0){
    return(list(BETA=BETA,
                accept_rate=accept/m,
                THETA = ThetaRecord,
                C_stat = C_stat))
  }else{
    return(list(BETA=BETA[-c(1:B),],
                accept_rate=accept/m,
                THETA = ThetaRecord[-c(1:B),],
                C_stat = C_stat[-c(1:B)]))
  }


}
```

3. C-statistics for Cox Proportional Hazard Model

```
coxmodel <- coxph(Surv(rfstime, status)~age + meno + size + grade + er + hormon,x=TRUE, data = gbsg)
summary(coxmodel)
```

```
## Call:
## coxph(formula = Surv(rfstime, status) ~ age + meno + size + grade + 
##     er + hormon, data = gbsg, x = TRUE)
## 
##   n= 686, number of events= 299 
## 
##             coef  exp(coef)   se(coef)      z Pr(>|z|)    
## age    -0.0095790  0.9904667  0.0092688 -1.033 0.301384    
## meno    0.3534726  1.4240040  0.1808870  1.954 0.050689 .  
## size    0.0141638  1.0142646  0.0036669  3.863 0.000112 ***
## grade   0.3801394  1.4624885  0.1013234  3.752 0.000176 ***
## er     -0.0005981  0.9994021  0.0004709 -1.270 0.204097    
## hormon -0.3554848  0.7008336  0.1287210 -2.762 0.005751 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
##        exp(coef) exp(-coef) lower .95 upper .95
## age       0.9905     1.0096    0.9726    1.0086
## meno      1.4240     0.7022    0.9989    2.0299
## size      1.0143     0.9859    1.0070    1.0216
## grade     1.4625     0.6838    1.1991    1.7838
## er        0.9994     1.0006    0.9985    1.0003
## hormon    0.7008     1.4269    0.5446    0.9019
## 
## Concordance= 0.627  (se = 0.017 )
## Likelihood ratio test= 46.34  on 6 df,   p=3e-08
## Wald test            = 47.52  on 6 df,   p=1e-08
## Score (logrank) test = 47.94  on 6 df,   p=1e-08
```

```
theta_cox = THETA(coxmodel$x,coxmodel$coefficients)
Wmat_cox = HarrellC_Wmat(gbsg$rfstime,gbsg$status,2500)
C_cox = HarrellC(theta_cox,Wmat_cox)
print(C_cox)
```

```
## [1] 0.6273096
```

Both the summary result and our function show that using Cox-PH model can have a c-statistics of 0.627. And we will use this value as a comparison.

Simulation

Summary of the Result

1. Prior variance for beta does not change simulation of C statistics that much. But it changes the acceptance rate of our algorithm a lot.

2. The choice of η will significantly affect our C statistics. It also changes the acceptance rate

3. It seems that we a lower value of C statistics using Uno's Method, comparing to Harrell's Method. (This needs further investigation)

Function for plotting beta iteration (gbsg data only)

```
beta_iter <- function(result,m){
  par(mfrow=c(2,3))
  plot(1:m,result$BETA[,2],xlab = "Iteration",
     ylab = "Beta Age",type = "l")
  plot(1:m,result$BETA[,3],xlab = "Iteration",
     ylab = "Beta Meno",type = "l")
  plot(1:m,result$BETA[,4],xlab = "Iteration",
     ylab = "Beta Size",type = "l")
  plot(1:m,result$BETA[,5],xlab = "Iteration",
     ylab = "Beta Grade",type = "l")
  plot(1:m,result$BETA[,6],xlab = "Iteration",
     ylab = "Beta Er",type = "l")
  plot(1:m,result$BETA[,7],xlab = "Iteration",
     ylab = "Beta Hormon",type = "l")

}

beta_hist <- function(result){
  par(mfrow=c(2,3))
  hist(result$BETA[,2],xlab = "Beta Age",main = "")
  hist(result$BETA[,3],xlab = "Beta Meno",main = "")
  hist(result$BETA[,4],xlab = "Beta Size",main = "")
  hist(result$BETA[,5],xlab = "Beta Grade",main = "")
  hist(result$BETA[,6],xlab = "Beta Er",main = "")
  hist(result$BETA[,7],xlab = "Beta Hormon",main = "")
}
```

Result (1-5) for changing prior variance

```
Y = gbsg$rfstime
delta=gbsg$status
tau = 2500
A <- model.matrix(rfstime ~ age + meno + size + grade + er + hormon,data=gbsg)
beta0 = rep(0,dim(A)[2])


var.prop = var(Y)*solve(t(A)%*%A)
m = 2000
B = 0
eta = length(Y)
Wmat_option = 0

system.time({
  sigma0 = rep(10,dim(A)[2])
  result1 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)

  sigma0 = rep(100,dim(A)[2])
  result2 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)
  sigma0 = rep(300,dim(A)[2])
  result3 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)

  sigma0 = rep(500,dim(A)[2])
  result4 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)
  sigma0 = rep(1000,dim(A)[2])
  result5 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)
})
```

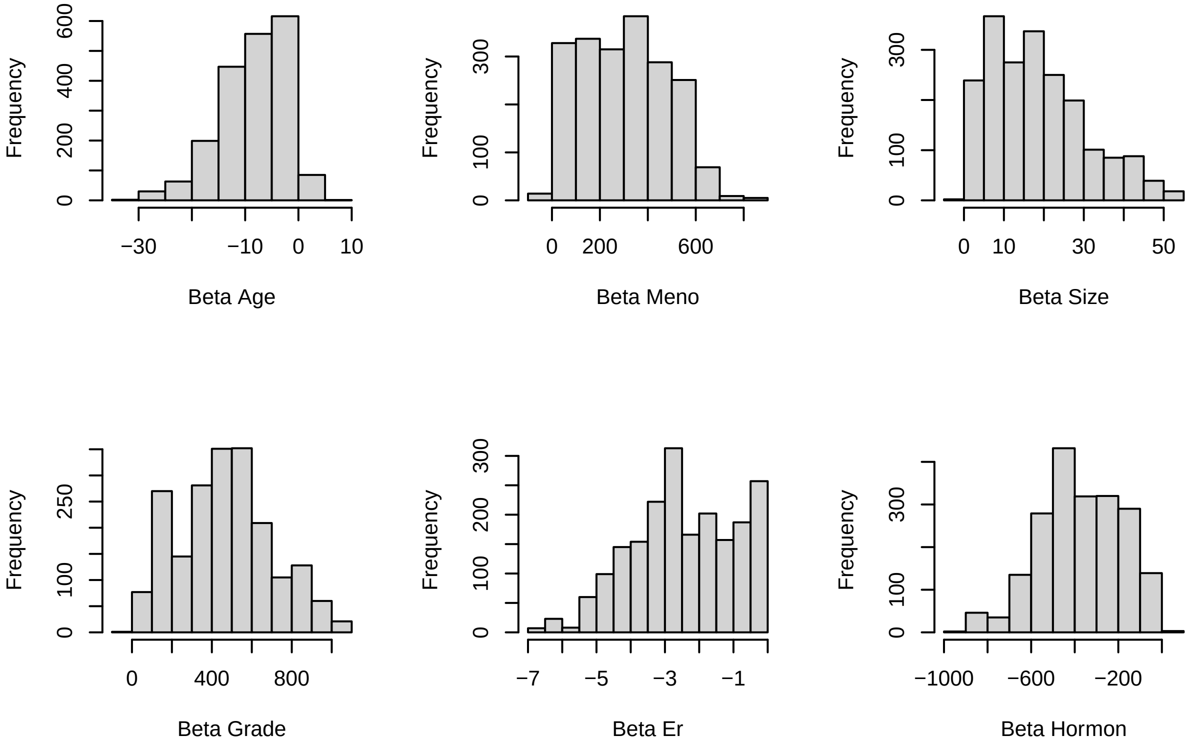
```
##    user  system elapsed
##  292.44   96.14  400.17
```

To plot the iteration for C statistics, we use result 4 as an example.

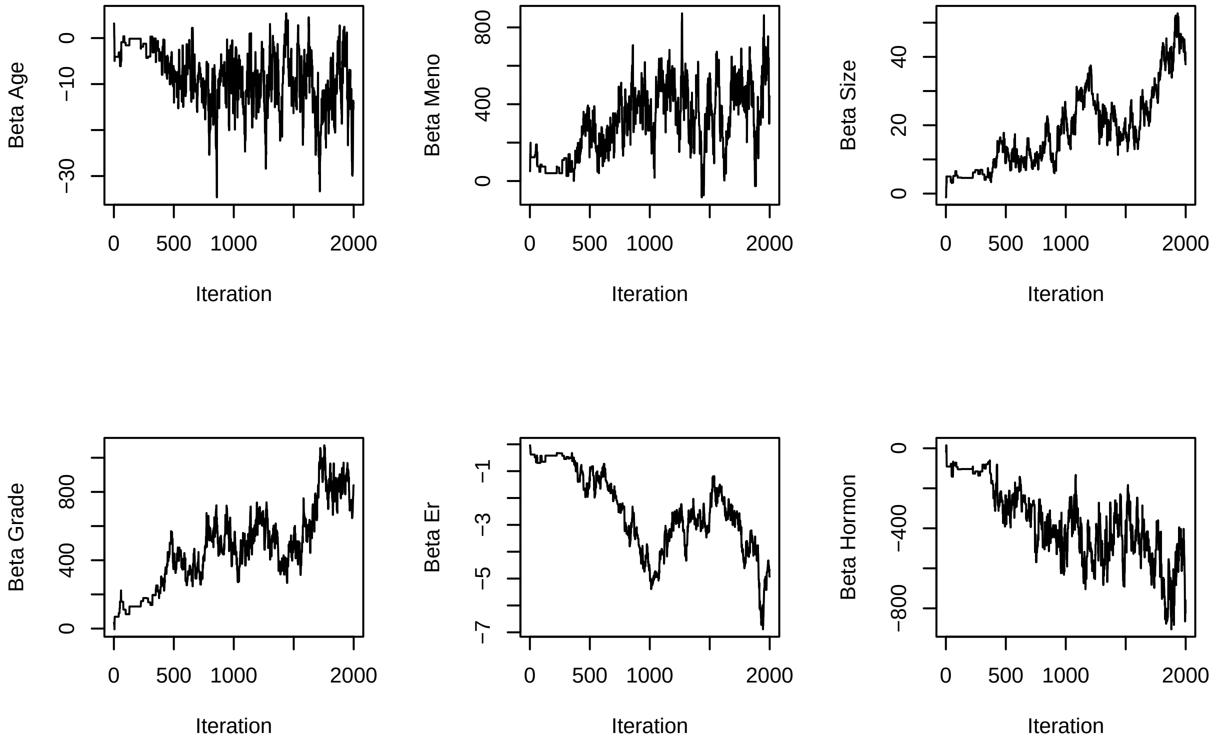
```
result4$accept_rate
```

```
## [1] 0.475
```

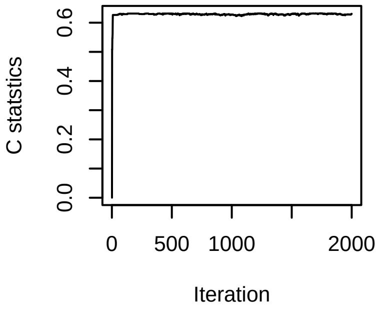
```
beta_hist(result4)
```



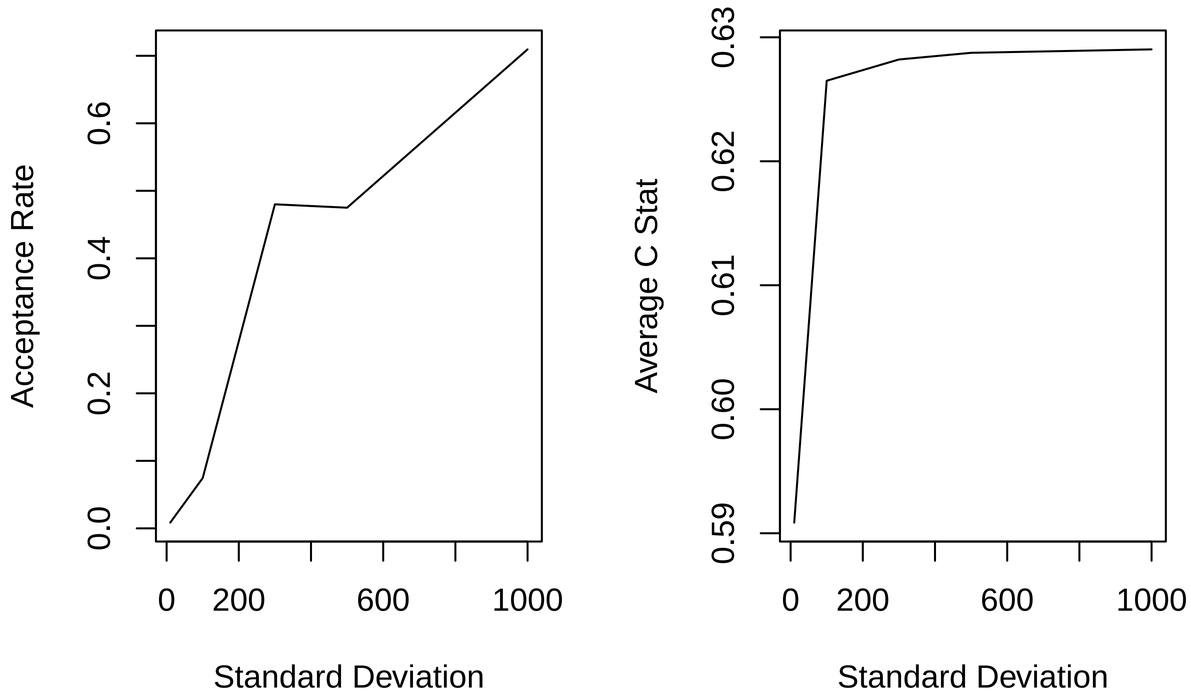
```
beta_iter(result4,m)
```



```r
plot(1:m,result4$C_stat,xlab = "Iteration",
     ylab = "C statstics",type = "l")
```



```
par(mfrow=c(1,2))
plot(c(10,100,300,500,1000),c(result1$accept_rate,result2$accept_rate,
      result3$accept_rate,result4$accept_rate,result5$accept_rate),
     xlab = "Standard Deviation",ylab = "Acceptance Rate",type = "l")

plot(c(10,100,300,500,1000),c(mean(result1$C_stat),mean(result2$C_stat),                  mean(result3$C
      xlab = "Standard Deviation",ylab = "Average C Stat",type = "l")
```



Result 6 is to test burn-in option (Compared to Result 4)

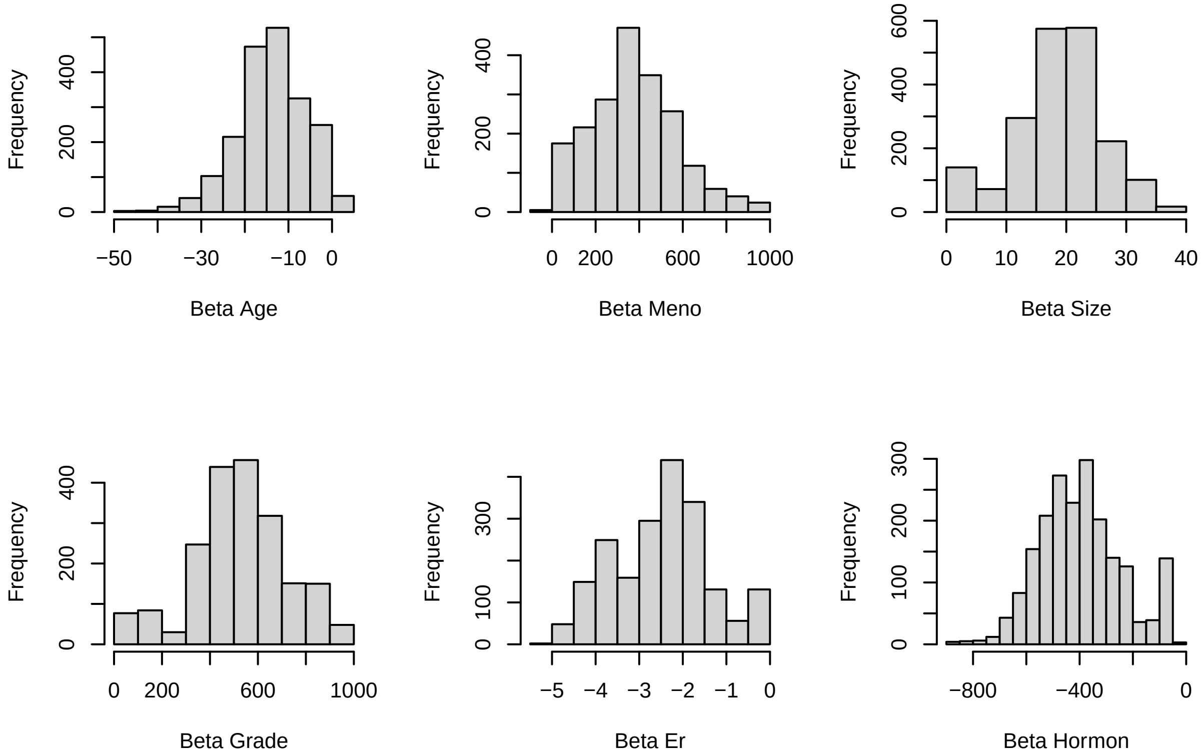
```
sigma0 = rep(500,dim(A)[2])
m = 2200
B = 200
system.time({
  result6 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)

})
```

```
##    user  system elapsed
##   64.89   20.09  106.19
```

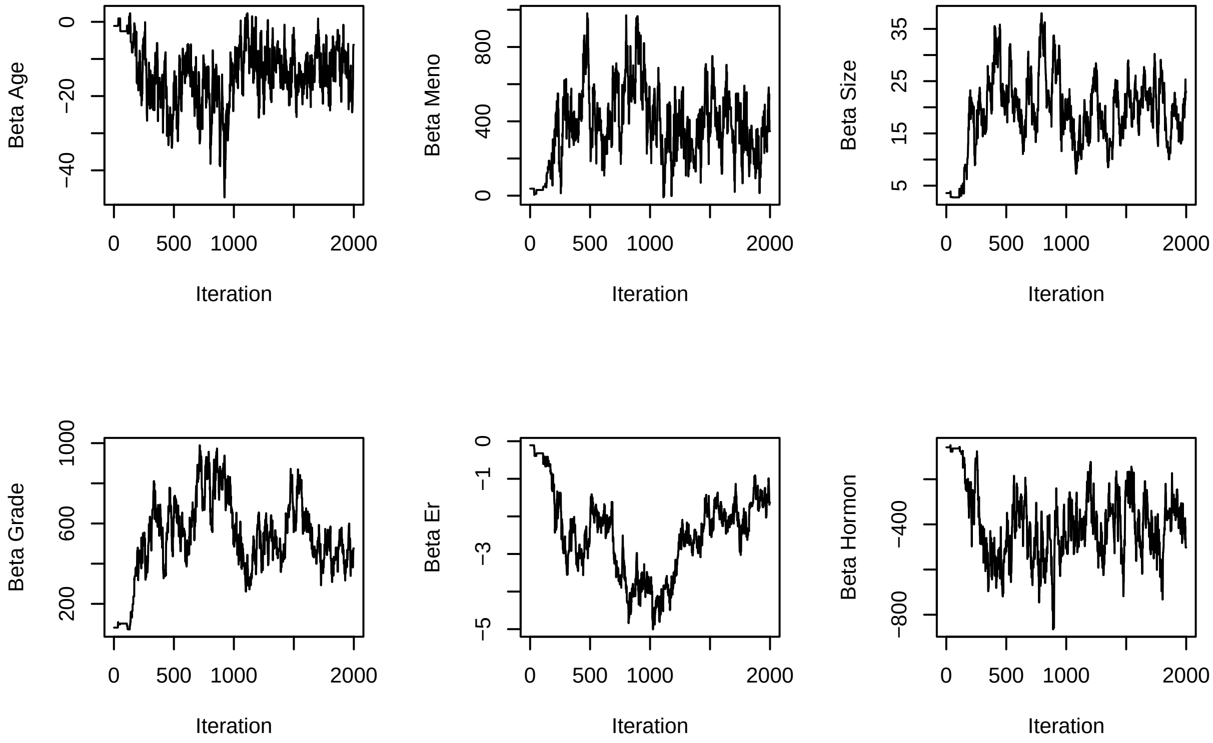
```
result6$accept_rate
```

```
## [1] 0.4909091
```

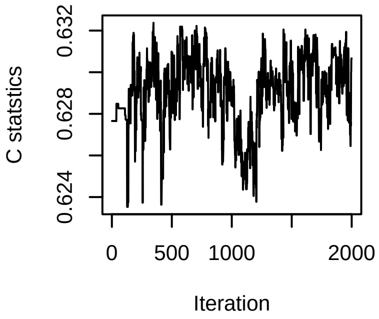
```
beta_hist(result6)
```



```
beta_iter(result6,m-B)
```



```
plot(1:(m-B),result6$C_stat,xlab = "Iteration",
     ylab = "C statstics",type = "l")
```



```
mean(result4$C_stat)
```

```
## [1] 0.6287485
```

```
mean(result6$C_stat)
```

```
## [1] 0.6290651
```

C statistics slightly better? (Need more experiments)

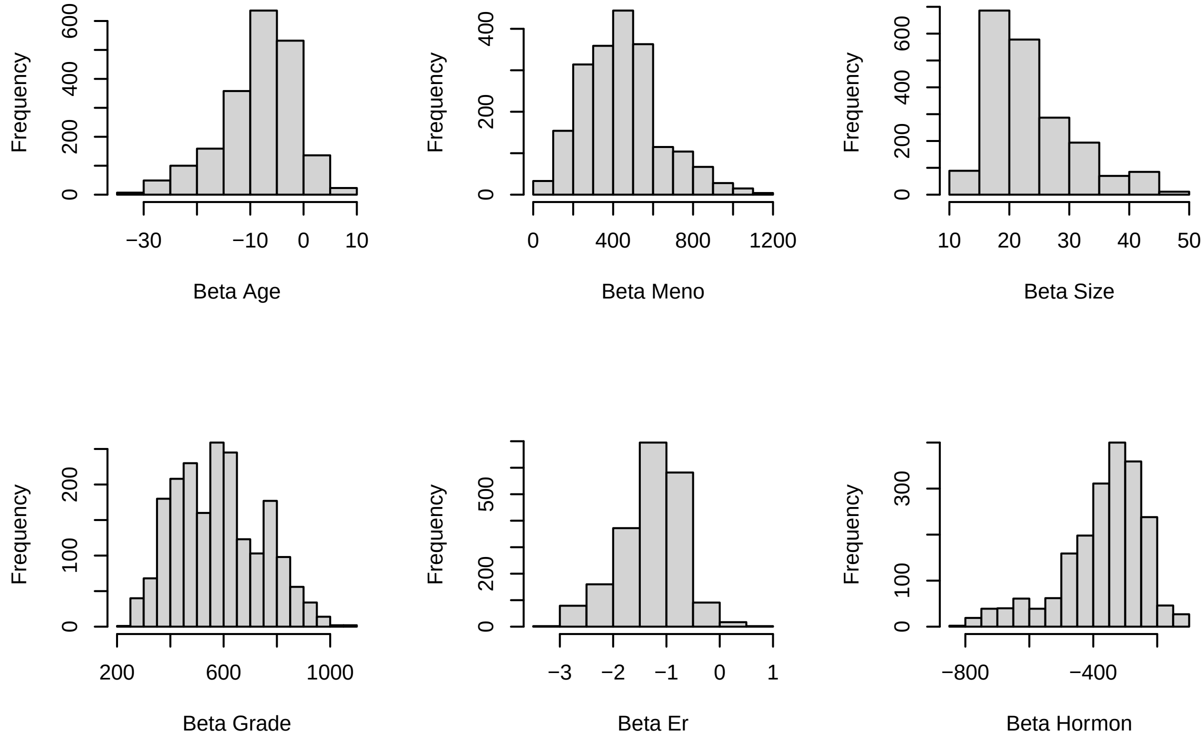
Result 7 is to test Uno C statistics (Compared to Result 6)

```
Wmat_option = 1
result7 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)
```

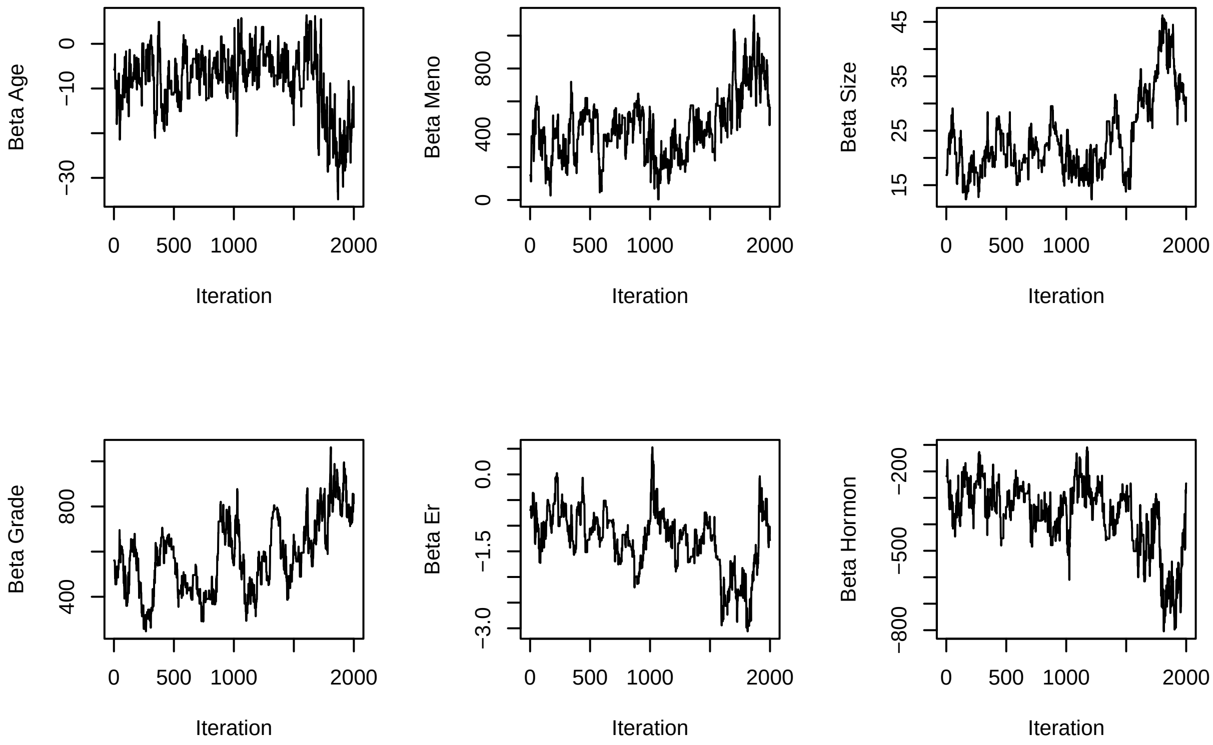
```
result7$accept_rate
```

```
## [1] 0.3718182
```

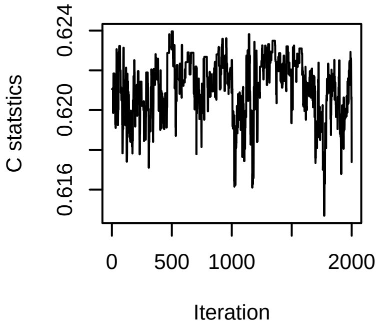
```
beta_hist(result7)
```



```
beta_iter(result7,m-B)
```



```
plot(1:(m-B),result7$C_stat,xlab = "Iteration",
     ylab = "C statstics",type = "l")
```



```
mean(result6$C_stat)
```

```
## [1] 0.6290651
```

```
mean(result7$C_stat)
```

```
## [1] 0.6211541
```

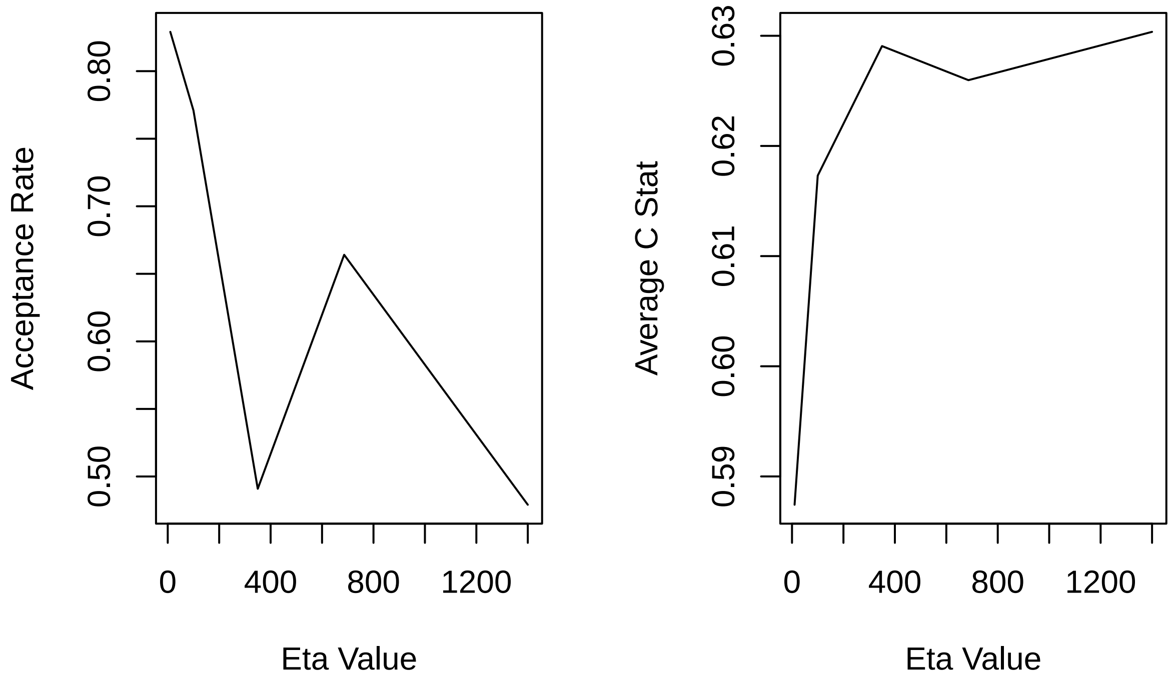
Uno C statistic seems to have a lower value than Harrell C.

Result (8-12)

```
Wmat_option = 0
sigma0 = rep(500,dim(A)[2])
m = 2200
B = 200

# In our gbsg dataset our n is approximately 700

system.time({
  eta = 10
  result8 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)

  eta = 100
  result9 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)

  eta = 350
  result10 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)

  eta = 1400
  result11 = MH_Sampling(Y,delta,tau,
                        A,beta0,sigma0,
                        var.prop,m,
                        B,eta,Wmat_option)

})
```

```
##    user  system elapsed
##  235.67   77.26  397.22
```

```
par(mfrow=c(1,2))
plot(c(10,100,350,length(Y),1400),c(result8$accept_rate,result9$accept_rate,
      result6$accept_rate,result10$accept_rate,result11$accept_rate),
     xlab = "Eta Value",ylab = "Acceptance Rate",type = "l")

plot(c(10,100,350,length(Y),1400),c(mean(result8$C_stat),mean(result9$C_stat),                  mean(re
      xlab = "Eta Value",ylab = "Average C Stat",type = "l")
```


Acceptance Rate
0.50
0.60
0.70
0.80
0
400
800
1200
Eta Value
Average C Stat
0.59
0.60
0.61
0.62
0.63
0
400
800
1200
Eta Value