```
###データ読み込み
dall<-read.table("E:¥¥dataallcsv1.csv",header=TRUE,sep=",")

##-1 行目：ID
##-2-4 行目：目的変数
##-5-7 行目：説明変数

d<-dall

module(nuopt)
minSeg<-2##計算する最小セグメント数
MSeg<-2##計算する最大セグメント数

####----関数定義ここから----####
###---ロジットモデルの重みつき対数尤度関数
BinLogit<-function(beta,x,y,pz){

        ones<-rep(1,nrow(y))

        beta1<-beta[1:v]
        beta2<-beta[(1+v):(2*v)]

        u1<-x%*%beta1
        u2<-x%*%beta2

        sumV<-exp(u1)+exp(u2)+ones

        p1<-exp(u1)/sumV
        p2<-exp(u2)/sumV
        p3<-ones/sumV

        p<-cbind(p1,p2,p3)
        LogLis<-y*log(p)

        LogL<-sum(pz*LogLis)
```

```
        return(LogL)
}

###---ロジットモデルの尤度関数,[[1]]で尤度, [[2]]で選択確率
BinLogitPP<-function(beta,x,y){

        ones<-rep(1,nrow(y))

        beta1<-beta[1:v]
        beta2<-beta[(1+v):(2*v)]

        u1<-x%*%beta1
        u2<-x%*%beta2

        sumV<-exp(u1)+exp(u2)+ones

        p1<-exp(u1)/sumV
        p2<-exp(u2)/sumV
        p3<-ones/sumV

        p<-cbind(p1,p2,p3)

        L<-prod(p^y)
        return(list(L,p))
}

###---スコア計算用関数
CalcU<-function(b,x,y,z,r){
Uisk<-array(0,dim=c(N,2*v,S))

#---パラメータのスコア
for(i in 1:N){
        p<-rep(0,nrow(y[y[,1]==i,]))

        for(s in 1:S){
                for(cn in 1:2){
```

```
                        p<-BinLogitPP(beta=b[,s],
                                x=x[x[,1]==i,-1],
                                y=y[y[,1]==i,-1]
                                )[[2]][,cn]

                        for(k in 1:v){
                                ymp<-(y[y[,1]==i,(cn+1)]-p)*x[x[,1]==i,(k+1)]
                                a<-(cn-1)*4+k

                                Uisk[i,a,s]<-z[i,s]*sum(ymp)
                        }
                }
        }
}

for(s in 1:S){
        if(s==1){
                temp<-Uisk[,,s]
        }else{
                temp<-cbind(temp,Uisk[,,s])
        }
}

#---セグメントサイズのスコア
if(S>1){
Ur<-matrix(0,N,(S-1))
        for(s in 1:(S-1)){
                Ur[,s]<-z[,s]/r[s]-z[,S]/r[S]
        }
}

if(S>1){
        Ui<-cbind(Ur,temp)
        U<-apply(Ui,2,sum)
}else{
        Ui<-temp
```

```
        U<-apply(Ui,2,sum)
}
Infomat<-crossprod(Ui)-outer(U,U)/N
print(paste("情報行列の行列式",det(Infomat)))
if(is.na(det(Infomat))==TRUE | det(Infomat)<0.0001){
        Infose<-0
        print("逆行列が計算できません")
}else{

        Infose<-sqrt(diag(solve(Infomat)))
}
return(Infose)
}

###---Nuopt 用関数---###
ML <- function(YY,XX,ZZ,BB1,BB2)
                        {
                        I <- Set()
                        ch <-Set()
                        Y <- Parameter(index=dprod(I,ch),YY)
                        P <- Set()
                        X <- Parameter(index=dprod(I,P),XX)
                        Z <- Parameter(index=I,as.array(ZZ))
                        Bint1 <- Parameter(index=P,as.array(BB1))
                        Bint2 <- Parameter(index=P,as.array(BB2))
                        i <- Element(set=I)
                        j <- Element(set=P)
                        b1 <- Variable(index=j)
                        b2 <- Variable(index=j)
                        b1[j]~Bint1[j]
                        b2[j]~Bint2[j]

                        u1 <-Expression(index=i)
                        p1 <-Expression(index=i)
                        u2 <-Expression(index=i)
                        p2 <-Expression(index=i)
```

```
                        p3 <-Expression(index=i)
                        u1[i]~Sum(X[i,j]*b1[j],j)
                        u2[i]~Sum(X[i,j]*b2[j],j)
                        p1[i]~exp(u1[i])/(1+exp(u1[i])+exp(u2[i]))
                        p2[i]~exp(u2[i])/(1+exp(u1[i])+exp(u2[i]))
                        p3[i]~1/(1+exp(u1[i])+exp(u2[i]))
                        Lopt <- Objective(type="maximize")

        Lopt~Sum(Z[i]*Y[i,1]*log(p1[i])+Z[i]*Y[i,2]*log(p2[i])+Z[i]*Y[i,3]*log(p
3[i]),i)
                }

####----関数定義ここまで----####


###----出力記録用----###
listb<-list(0)
listr<-list(0)
listz<-list(0)
listznt<-list(0)
listbse<-list(0)
listg<-list(0)
cBIC<-rep(0,MSeg)
listp<-list(0)

###---変数定義----###
#-v：ロジットモデルの変数の数
v<-ncol(d)-3

#-Tn：データの縦のサイズ
Tn<-nrow(d)

#-group：所属グループ
group<-rep(0,Tn)
```

```
###---1から連番のID作成----###
###---データはIDが昇順に並んでいる---###
ID<-rep(0,Tn)

temp<-d[1,1]
id<-1
for(n in 1:Tn){
        if(temp==d[n,1]){
                ID[n]<-id
        }else{
                id<-id+1
                temp<-d[n,1]
                ID[n]<-id
        }
}

#-N：人数
N<-id
print(paste("人数",N))

###---切片作成など----###
y<-as.matrix(cbind(ID[1:Tn],as.matrix(d[,2:4])))
x<-as.matrix(cbind(ID[1:Tn],rep(1,Tn),as.matrix(d[,c(-1,-2,-3,-4)])))



###----第sセグメント計算開始
for(S in minSeg:MSeg){

#-r：セグメントサイズ用ベクトル（s*1）
r<-c(rep(1/S,S))

#-z：各セグメントへの所属確率用行列（N*s）
#-znt：各セグメントへの所属確率用行列（Tn*(s+1)),+1はIDの分
zint<-runif(N*S,0,1)
z<-matrix(zint,N,S)
znt<-cbind(ID[1:Tn],matrix(runif(S*Tn,0,1),Tn,S))
```

```
#-b：ロジットモデルのパラメータ用行列（v*s）
b<-matrix(runif(2*v*S,-0.5,0.5),2*v,S)

#-NC.Lis：人セグメントごとの尤度（N*S）
NC.Lis<-matrix(rep(0,N*S),N,S)
rNC.Lis<-matrix(rep(0,N*S),N,S)
LO<-0##尤度

###---EM アルゴリズム開始----###

diff<-100##diff：対数尤度の差
while(abs(diff)>=0.1){



##--セグメントサイズ r の推定
        r<-apply(z,2,sum)/N

##--第 s セグメントのパラメータ b の推定
for(s in 1:S){

module(nuopt,unload=T)
module(nuopt)

zopt<-as.vector(znt[,(s+1)])
yopt<-as.matrix(y[,-1])
xopt<-as.matrix(x[,-1])
bi1<-b[(1:4),s]
bi2<-b[(5:8),s]
sys <-System(model=ML,yopt,xopt,zopt,bi1,bi2)
sol <-solve(sys,trace=F)
beta1<-as.array(sol$variables$b1$current)
beta2<-as.array(sol$variables$b2$current)
b[,s]<-c(beta1,beta2)

}
```

```
##---個人別の第ｓセグメントにおける尤度算出
        for(i in 1:N){

                XX<-x[x[,1]==i,-1]
                YY<-y[y[,1]==i,-1]

                for(s in 1:S){
                        NC.Lis[i,s]<-BinLogitPP(beta=b[,s],
                                                x=XX,
                                                y=YY)[[1]]
                }
        }
        rNC.Lis<-t(r*t(NC.Lis))

##---所属確率ｚ（N*S）算出
        z<-rNC.Lis/matrix(apply(rNC.Lis,1,sum),N,S)

##---所属確率 znt（Tn*S）算出
        for(s in 1:S){
                for(i in 1:N){
                        znt[znt[,1]==i,(s+1)]<-z[i,s]
                }
        }

##---対数尤度算出，１回前の尤度との差を記録
        LLis<-t(r*t(NC.Lis))
        LL<-sum(log(apply(LLis,1,sum)))
        diff<-LL-LO
        LO<-LL
        print(paste("対数尤度",LL))

}
###---EM アルゴリズム終わり----###
```

```
###---第ｓセグメントのパラメータの値を出力----###
print(paste("第",S,"セグメント終了"))

listb[[S]]<-b
listr[[S]]<-r
listz[[S]]<-cbind(c(1:N),z)
listznt[[S]]<-cbind(d[,1],znt)
print(b)
print(r)

###---標準誤差算出---###
Use<-CalcU(b,x,y,z,r)
listbse[[S]]<-matrix(Use[-c(1:(S-1))],2*v,S)


###---BIC算出---###
BICscore<--2*LL+(length(b)+length(r))*log(Tn)
cBIC[S]<-BICscore

###---所属グループ算出---###
for(i in 1:Tn){
group[i]<-which(znt[i,-1]==max(znt[i,-1]))
}
listg[[S]]<-cbind(d[,1],group)

}
###---第ｓセグメント計算終わり---###

###---出力---###
cBIC
listb
listbse
listr


###---あてはまり検証---###
```

```
##--検証用データ
#-検証期間のみを抽出
d1test<-read.table("E:¥¥testdata.csv",header=TRUE,sep=",")

xtest<-cbind(1,d1test[,-c(1,2,3,4)])
ytest<-d1test[,c(2,3,4)]
prob<-matrix(0,nrow(ytest),3)

##--セグメント番号と検証用データをマージ
tempg<-listg[[5]]
group2<-tempg[duplicated(tempg[,1])==FALSE,]
group1<-data.frame(UserID=group2[,1],seg=group2[,2])
colnames(d1test)<-c("UserID","cv","session","nosession","v1","v2","v3")
mdata<-merge(d1test,group1,by="UserID")
pred<-rep(0,nrow(ytest))


###---5セグメントを採用---###
S<-5
b<-listb

CalPP=function(beta, x, y)
{
        beta1 <- beta[1:v]
        beta2 <- beta[(1+v):(2*v)]
        u1 <- x %*% beta1
        u2 <- x %*% beta2
        sumV <- exp(u1) + exp(u2) + 1
        p1 <- exp(u1)/sumV
        p2 <- exp(u2)/sumV
        p3 <- 1/sumV
        p <- cbind(p1, p2, p3)
        return(p)
}

for(i in 1:nrow(ytest)){
```

```
        xt<-xtest[i,]
        yt<-ytest[i,]
        g<-mdata$seg[i]
        prob[i,]<-CalPP(c(b[,g]),xt,yt)
        pred[i]<-which(prob[i,]==max(prob[i,]))
}

choice<-rep(0,nrow(ytest))
for(i in 1:nrow(ytest)){
choice[i]<-which(d1test[i,2:4]==1)
}

##--的中率算出
hit<-table(factor(choice,levels=1:3),factor(pred,levels=1:3))
sum(diag(hit))/sum(hit)
```