Assignment for Labor Economics II

Jiazhe CHEN[*]

July 2018

1.2 Problems

1.

```
#simulation of childbearing
set.seed(15)
n_it=rbinom(500,1,0.5)
```

2.

$Pr(d_{it}=0|n_{it}=0)$ can be calculated as

$$\begin{aligned} Pr(d_{it}=0|n_{it}=0) &= Pr(u_0(n_{it})>u_1(n_{it})|n_{it}=0) = Pr(\epsilon_{1,it}<\epsilon_{0,it}) \\ &= \int_{-\infty}^{+\infty} exp(-exp(-\epsilon_{0,it}))\times exp(-\epsilon_{0,it}-exp(-\epsilon_{0,it}))d\epsilon_{0,it} \end{aligned}$$

Let $exp(-\epsilon_{0,it})=u, d\epsilon_{0,it}=-\frac{1}{u}du$, by integral by substitution, we have

$$\begin{aligned} Pr(d_{it}=0|n_{it}=0) &= \int_{+\infty}^{0}[u\times exp(-2u)(-\frac{1}{u})]du \\ &= \frac{1}{2}\times exp(-2u)\Big|_{+\infty}^{0} \\ &= \frac{1}{2} \end{aligned}$$

Similarly,

$$\begin{aligned} Pr(d_{it}=1|n_{it}=0) &= \frac{1}{2}, \\ Pr(d_{it}=0|n_{it}=1) &= \frac{e}{1+e}, \\ Pr(d_{it}=1|n_{it}=1) &= \frac{1}{1+e}. \end{aligned}$$

[*]Student ID: 29-176034. Zehua SHI and I collaborated to write the R code part. The "maxLik" package comes from Henningsen, Arne and Toomet, Ott (2011), "maxLik: A package for maximum likelihood estimation in R", Computational Statistics 26(3), 443-458. DOI 10.1007/s00180-010-0217-1.

3.

```
#simulation of work decisions
library(VGAM)
set.seed(111)
epsilon_1it=rgumbel(500)
epsilon_0it=rgumbel(500)
u1=-n_it+epsilon_1it
u0=epsilon_0it
d=u1-u0
#let negative numbers be 0, positive ones be 1
d[d<0]=0
d[d>0]=1
#count 0 in d vector and 0 in n_it vector
d0n0=length(which(d==0&n_it==0))
d1n0=length(which(d==1&n_it==0))
d0n1=length(which(d==0&n_it==1))
d1n1=length(which(d==1&n_it==1))
n0=length(which(n_it==0))
n1=length(which(n_it==1))
prd0n0=d0n0/n0;prd0n0
prd1n0=d1n0/n0;prd1n0
prd0n1=d0n1/n1;prd0n1
prd1n1=d1n1/n1;prd1n1
```

Therefore, the labor force participation given $n_{it} = 0$ and $n_{it} = 1$ are

$$Pr(d_{it} = 1|n_{it} = 0) = 0.48,$$
$$Pr(d_{it} = 1|n_{it} = 1) = 0.25,$$

which are approximately the same as the theoretical values we've calculated before.

4.

Since n_{it} for $i = 1, \ldots, 500$ are exogenously given to all individuals, the log-likelihood function should only contain the probability of d_{it}. The log-likelihood function is as follows.

$$\begin{aligned}\ell &= \sum_{i=1}^{500}\{d_{it}ln(Pr(d_{it}=1)) + (1-d_{it})ln(Pr(d_{it}=0))\} \\ &= \sum_{i=1}^{500}\{d_{it}ln[\frac{exp(\alpha_1+\alpha_2 n_{it})}{1+exp(\alpha_1+\alpha_2 n_{it})}] + (1-d_{it})ln[1-\frac{exp(\alpha_1+\alpha_2 n_{it})}{1+exp(\alpha_1+\alpha_2 n_{it})}]\}\end{aligned}$$

5.

```
#maximum likelihood estimation of static model
library(maxLik)
loglik=function(alpha){
logL=sum(d*log(exp(alpha[1]+alpha[2]*n_it)/(1+exp(alpha[1]+alpha[2]*n_it)))+
```

```
        (1-d)*log(1-exp(alpha[1]+alpha[2]*n_it)/(1+exp(alpha[1]+alpha[2]*n_it))))
  return(logL)
}
alpha=maxLik(loglik,start=c(0,-1))
summary(alpha)
```

The maximum likelihood estimation result is $\hat{\alpha_1} = 0.02, \hat{\alpha_2} = -0.91$, and the standard errors are $s_{\hat{\alpha_1}} = 0.13, s_{\hat{\alpha_2}} = 0.19$, which are pretty close to the true values.

2.2 Problems

1.

1.(a)

The following steps describes simulation method for any types.

Step 1:

Draw d_{i0}, x_{i1}, n_{i1} of all individuals' for the zero period randomly. Note that x_{i0} must be less than or equal to 46. Then draw $\epsilon_{1,i0}, \epsilon_{0,i0}$ of all individuals' for the zero period randomly.

Step 2:

Individual i decides to work or not in the first period by the following way. Calculate $u_1(\{x_{i1}, n_{i1}\})$ first, and then compare it with $u_0(\{x_{i1}, n_{i1}\})$. If $u_1(\{x_{i1}, n_{i1}\}) > u_0(\{x_{i1}, n_{i1}\})$, then she will work in the first period. In such case, $d_{i1} = 1$ and $x_{i2} = x_{i1} + 1$. Otherwise, she will stay at home in the first period, and $d_{i1} = 0$, $x_{i2} = x_{i1}$.

Step 3:

Repeat Step 2 for $t = 2, \ldots, 5$, we have $d_{it}, t = 0, \ldots, 5$, and $x_{it}, t = 1, \ldots, 5$.

Step 4:

Childbearing status is exogenously given by the transition probability $Pr(n_{it+1} = 1|n_{it})$ for each individual.

Step 5:

Repeat Step 4, and we have $n_{it}, t = 0, \ldots, 5$.

```r
#draw the zero period data for two types
set.seed(15)
l1d_i0=rbinom(300,1,0.5)
set.seed(16562)
l2d_i0=rbinom(300,1,0.5)
set.seed(561515)
l1x_i1=sample(seq(0,45,by=1),300,replace=TRUE)
set.seed(1989565)
l2x_i1=sample(seq(0,45,by=1),300,replace=TRUE)
set.seed(5646)
l1n_i1=rbinom(300,1,0.5)
set.seed(753)
l2n_i1=rbinom(300,1,0.5)
l1gamma=c(-1,-2.5);l2gamma=c(1,-2.5)

#define the transition probability of childbearing n_it for two types
f1=function(l1n_it){
  exp(l1gamma[1]+l1gamma[2]*l1n_it)/(1+exp(l1gamma[1]+l1gamma[2]*l1n_it))
}
f2=function(l2n_it){
  exp(l2gamma[1]+l2gamma[2]*l2n_it)/(1+exp(l2gamma[1]+l2gamma[2]*l2n_it))
}
```

```
#use transition probability to get n_it for t=2,...,5 for both types
l1n_it=matrix(l1n_i1,ncol = 5,nrow = 300)
for (t in 1:4){
  l1n_it[,t+1]=rbinom(300,1,sapply(l1n_it[,t], f1))
}

l2n_it=matrix(l2n_i1,ncol = 5,nrow = 300)
for (t in 1:4){
  l2n_it[,t+1]=rbinom(300,1,sapply(l2n_it[,t], f2))
}

library(VGAM)
set.seed(15)
l1epsilon1i1=rgumbel(300)
set.seed(987)
l1epsilon0i1=rgumbel(300)
set.seed(32135)
l2epsilon1i1=rgumbel(300)
set.seed(465156)
l2epsilon0i1=rgumbel(300)

l1d_it=matrix(l1d_i0,ncol = 6, nrow=300)
l1u1sit=matrix(data=NA,ncol=5,nrow=300)
l1epsilon1it=matrix(l1epsilon1i1,ncol=5,nrow=300)
l1epsilon0it=matrix(l1epsilon0i1,ncol=5,nrow=300)
l1x_it=matrix(l1x_i1,ncol=6,nrow=300)

#matrix of type 2
l2d_it=matrix(l2d_i0,ncol = 6, nrow=300)
l2u1sit=matrix(data=NA,ncol=5,nrow=300)
l2epsilon1it=matrix(l2epsilon1i1,ncol=5,nrow=300)
l2epsilon0it=matrix(l2epsilon0i1,ncol=5,nrow=300)
l2x_it=matrix(l2x_i1,ncol=6,nrow=300)

#error term of type 1 for t=1,...,5
for(t in 1:4){
  l1epsilon1it[,t+1]=rgumbel(300)
  l1epsilon0it[,t+1]=rgumbel(300)
  l2epsilon1it[,t+1]=rgumbel(300)
  l2epsilon0it[,t+1]=rgumbel(300)
}

#type 1
for(t in 1:5){
  alphal1=c(-2,2,-2,0.1);
  l1u1sit[,t]=alphal1[1]+alphal1[2]*l1d_it[,t]+alphal1[3]*l1n_it[,t]+
              alphal1[4]*log(l1x_it[,t]+1)+l1epsilon1it[,t]
  l1d_it[,t+1]=l1u1sit[,t]-l1epsilon0it[,t]

l1d_it[,t+1][l1d_it[,t+1]<0]=0
l1d_it[,t+1][l1d_it[,t+1]>0]=1

  for (i in 1:300)
```

```
    if(l1d_it[i,t+1]==1){
  l1x_it[i,t+1]=l1x_it[i,t]+1
}else{
  l1x_it[i,t+1]=l1x_it[i,t]
}
}

#type 2
for(t in 1:5){
 alphal2=c(0,2,-2,0.1)
  l2u1sit[,t]=alphal2[1]+alphal2[2]*l2d_it[,t]+alphal2[3]*l2n_it[,t]+
              alphal2[4]*log(l2x_it[,t]+1)+l2epsilon1it[,t]
  l2d_it[,t+1]=l2u1sit[,t]-l2epsilon0it[,t]

l2d_it[,t+1][l2d_it[,t+1]<0]=0
l2d_it[,t+1][l2d_it[,t+1]>0]=1

for (i in 1:300)

if(l2d_it[i,t+1]==1){
  l2x_it[i,t+1]=l2x_it[i,t]+1
}else{
  l2x_it[i,t+1]=l2x_it[i,t]
}
}

l1d01=length(which(l1d_it[,1]==1))
l1d11=length(which(l1d_it[,2]==1))
l1d21=length(which(l1d_it[,3]==1))
l1d31=length(which(l1d_it[,4]==1))
l1d41=length(which(l1d_it[,5]==1))
l1d51=length(which(l1d_it[,6]==1))
l1prd1=c(l1d01,l1d11,l1d21,l1d31,l1d41,l1d51)/300
l1prd1

l2d01=length(which(l2d_it[,1]==1))
l2d11=length(which(l2d_it[,2]==1))
l2d21=length(which(l2d_it[,3]==1))
l2d31=length(which(l2d_it[,4]==1))
l2d41=length(which(l2d_it[,5]==1))
l2d51=length(which(l2d_it[,6]==1))
l2prd1=c(l2d01,l2d11,l2d21,l2d31,l2d41,l2d51)/300
l2prd1

prd1=(l1prd1+l2prd1)/2
prd1
```

Therefore, the simulated labor force participation rate over time is as follows.

Period	0	1	2	3	4	5
Type 1	0.57	0.29	0.20	0.18	0.17	0.20
Type 2	0.51	0.52	0.56	0.61	0.61	0.58
Pooled	0.54	0.41	0.38	0.39	0.39	0.39

1.(b)

```
#count the number of individuals whose d_it=0 & d_it+1=0 for each period
l1td0d0=length(which(l1d_it[,1]==0&l1d_it[,2]==0))+
        length(which(l1d_it[,2]==0&l1d_it[,3]==0))+
        length(which(l1d_it[,3]==0&l1d_it[,4]==0))+
        length(which(l1d_it[,4]==0&l1d_it[,5]==0))+
        length(which(l1d_it[,5]==0&l1d_it[,6]==0))

l2td0d0=length(which(l2d_it[,1]==0&l2d_it[,2]==0))+
        length(which(l2d_it[,2]==0&l2d_it[,3]==0))+
        length(which(l2d_it[,3]==0&l2d_it[,4]==0))+
        length(which(l2d_it[,4]==0&l2d_it[,5]==0))+
        length(which(l2d_it[,5]==0&l2d_it[,6]==0))

prd0d0=(l1td0d0+l2td0d0)/(length(which(l1d_it[,1]==0))+
       length(which(l1d_it[,2]==0))+length(which(l1d_it[,3]==0))+
       length(which(l1d_it[,4]==0))+length(which(l1d_it[,5]==0))+
       length(which(l2d_it[,1]==0))+length(which(l2d_it[,2]==0))+
       length(which(l2d_it[,3]==0))+length(which(l2d_it[,4]==0))+
       length(which(l2d_it[,5]==0)) )

#count the number of individuals whose d_it=0 & d_it+1=1 for each period
l1td0d1=length(which(l1d_it[,1]==0&l1d_it[,2]==1))+
        length(which(l1d_it[,2]==0&l1d_it[,3]==1))+
        length(which(l1d_it[,3]==0&l1d_it[,4]==1))+
        length(which(l1d_it[,4]==0&l1d_it[,5]==1))+
        length(which(l1d_it[,5]==0&l1d_it[,6]==1))

l2td0d1=length(which(l2d_it[,1]==0&l2d_it[,2]==1))+
        length(which(l2d_it[,2]==0&l2d_it[,3]==1))+
        length(which(l2d_it[,3]==0&l2d_it[,4]==1))+
        length(which(l2d_it[,4]==0&l2d_it[,5]==1))+
        length(which(l2d_it[,5]==0&l2d_it[,6]==1))

prd0d1=(l1td0d1+l2td0d1)/(length(which(l1d_it[,1]==0))+
       length(which(l1d_it[,2]==0))+length(which(l1d_it[,3]==0))+
       length(which(l1d_it[,4]==0))+length(which(l1d_it[,5]==0))+
       length(which(l2d_it[,1]==0))+length(which(l2d_it[,2]==0))+
       length(which(l2d_it[,3]==0))+length(which(l2d_it[,4]==0))+
       length(which(l2d_it[,5]==0)) )

#count the number of individuals whose d_it=1 & d_it+1=0 for each period
l1td1d0=length(which(l1d_it[,1]==1&l1d_it[,2]==0))+
        length(which(l1d_it[,2]==1&l1d_it[,3]==0))+
        length(which(l1d_it[,3]==1&l1d_it[,4]==0))+
        length(which(l1d_it[,4]==1&l1d_it[,5]==0))+
        length(which(l1d_it[,5]==1&l1d_it[,6]==0))

l2td1d0=length(which(l2d_it[,1]==1&l2d_it[,2]==0))+
        length(which(l2d_it[,2]==1&l2d_it[,3]==0))+
        length(which(l2d_it[,3]==1&l2d_it[,4]==0))+
        length(which(l2d_it[,4]==1&l2d_it[,5]==0))+
        length(which(l2d_it[,5]==1&l2d_it[,6]==0))
```

```
prd1d0=(l1td1d0+l2td1d0)/(length(which(l1d_it[,1]==1))+
        length(which(l1d_it[,2]==1))+length(which(l1d_it[,3]==1))+
        length(which(l1d_it[,4]==1))+length(which(l1d_it[,5]==1))+
        length(which(l2d_it[,1]==1))+length(which(l2d_it[,2]==1))+
        length(which(l2d_it[,3]==1))+length(which(l2d_it[,4]==1))+
        length(which(l2d_it[,5]==1)) )

#count the number of individuals whose d_it=1 & d_it+1=1 for each period
l1td1d1=length(which(l1d_it[,1]==1&l1d_it[,2]==1))+
        length(which(l1d_it[,2]==1&l1d_it[,3]==1))+
        length(which(l1d_it[,3]==1&l1d_it[,4]==1))+
        length(which(l1d_it[,4]==1&l1d_it[,5]==1))+
        length(which(l1d_it[,5]==1&l1d_it[,6]==1))

l2td1d1=length(which(l2d_it[,1]==1&l2d_it[,2]==1))+
        length(which(l2d_it[,2]==1&l2d_it[,3]==1))+
        length(which(l2d_it[,3]==1&l2d_it[,4]==1))+
        length(which(l2d_it[,4]==1&l2d_it[,5]==1))+
        length(which(l2d_it[,5]==1&l2d_it[,6]==1))

prd1d1=(l1td1d1+l2td1d1)/(length(which(l1d_it[,1]==1))+
       length(which(l1d_it[,2]==1))+
       length(which(l1d_it[,3]==1))+length(which(l1d_it[,4]==1))+
       length(which(l1d_it[,5]==1))+length(which(l2d_it[,1]==1))+
       length(which(l2d_it[,2]==1))+length(which(l2d_it[,3]==1))+
       length(which(l2d_it[,4]==1))+length(which(l2d_it[,5]==1)) )

l1prd0d0=l1td0d0/(length(which(l1d_it[,1]==0))+
         length(which(l1d_it[,2]==0))+
         length(which(l1d_it[,3]==0))+
         length(which(l1d_it[,4]==0))+
         length(which(l1d_it[,5]==0)) )

l2prd0d0=l2td0d0/(length(which(l2d_it[,1]==0))+
         length(which(l2d_it[,2]==0))+length(which(l2d_it[,3]==0))+
         length(which(l2d_it[,4]==0))+length(which(l2d_it[,5]==0)) )


l1prd0d1=l1td0d1/(length(which(l1d_it[,1]==0))+length(which(l1d_it[,2]==0))+
         length(which(l1d_it[,3]==0))+length(which(l1d_it[,4]==0))+
         length(which(l1d_it[,5]==0)) )
l2prd0d1=l2td0d1/(length(which(l2d_it[,1]==0))+length(which(l2d_it[,2]==0))+
         length(which(l2d_it[,3]==0))+length(which(l2d_it[,4]==0))+
         length(which(l2d_it[,5]==0)) )



l1prd1d0=l1td1d0/(length(which(l1d_it[,1]==1))+length(which(l1d_it[,2]==1))+
         length(which(l1d_it[,3]==1))+length(which(l1d_it[,4]==1))+
         length(which(l1d_it[,5]==1)) )

l2prd1d0=l2td1d0/(length(which(l2d_it[,1]==1))+length(which(l2d_it[,2]==1))+
         length(which(l2d_it[,3]==1))+length(which(l2d_it[,4]==1))+
```

```
         length(which(l2d_it[,5]==1)) )

l1prd1d1=l1td1d1/(length(which(l1d_it[,1]==1))+length(which(l1d_it[,2]==1))+
         length(which(l1d_it[,3]==1))+length(which(l1d_it[,4]==1))+
         length(which(l1d_it[,5]==1)))
l2prd1d1=l2td1d1/(length(which(l2d_it[,1]==1))+length(which(l2d_it[,2]==1))+
         length(which(l2d_it[,3]==1))+length(which(l2d_it[,4]==1))+
         length(which(l2d_it[,5]==1)) )
```

The transition matrix for labor force participation is as follows.

Type 1	$d_{t-1}=0$	$d_{t-1}=1$	Type 2	$d_{t-1}=0$	$d_{t-1}=1$	Pooled	$d_{t-1}=0$	$d_{t-1}=1$
$d_t=0$	0.89	0.53	$d_t=0$	0.61	0.28	$d_t=0$	0.79	0.36
$d_t=1$	0.11	0.47	$d_t=1$	0.39	0.72	$d_t=1$	0.21	0.64

2.

2.(a)

For type 1 individuals, the likelihood is $Pr(\{d_{i1},\ldots,d_{i5};x_{i2},\ldots,x_{i5};n_{i2},\ldots,n_{i5}\})$.

Calculate the likelihood of working decision of type 1 individuals.

$$\begin{aligned}\ell(\alpha_{l=1}|\{d_{i1},\ldots,d_{i5}\}) &= \sum_{t=1}^{5}\{d_{it}ln[Pr(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1)+\epsilon_{1,it}>\epsilon_{0,it})]\\ &\quad+(1-d_{it})ln[Pr(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1)+\epsilon_{1,it}<\epsilon_{0,it})]\}\\ &= \sum_{i=1}^{5}\{d_{it}ln[\frac{exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1))}{1+exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1))}]\\ &\quad+(1-d_{it})ln[1-\frac{exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1))}{1+exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1))}]\}\end{aligned}$$

Once the profile of working decision $\{d_{i1},\ldots,d_{i5}\}$ is made, $\{x_{i2},\ldots,x_{i5}\}$ will also be determined, hence there is no likelihood function for labor market experience.

Calculate the likelihood of childbearing process of type 1 individuals.

$$\begin{aligned}\ell(\gamma_{l=1}|n_{i2},\ldots,n_{i5}) &= \sum_{t=2}^{5}\{n_{it}ln[Pr(n_{it}=1|n_{it-1})]+(1-n_{it})ln[Pr(n_{it}=0|n_{it-1})]\}\\ &= \sum_{t=2}^{5}\{n_{it}ln[\frac{exp(\gamma_{1,i}+\gamma_2 n_{it-1})}{1+exp(\gamma_{1,i}+\gamma_2 n_{it-1})}]+(1-n_{it})ln[1-\frac{exp(\gamma_{1,i}+\gamma_2 n_{it-1})}{1+exp(\gamma_{1,i}+\gamma_2 n_{it-1})}]\}\end{aligned}$$

2.(b)

Estimate all the partial likelihood function listed above. The estimation values are pretty close to the true parameter values.

Parameter	True Value	Estimate	Standard Error
α_1	-2	-2.10	0.27
α_2	2	2.41	0.16
α_3	-2	-1.80	0.20
α_4	0.1	0.07	0.08
γ_1	-1	-0.82	0.07
γ_2	-2.5	-2.6 1	0.31

```
library(maxLik)

loglikalphal1=function(halphal1){
  halpha1l1=halphal1[1]
  halpha2l1=halphal1[2]
  halpha3l1=halphal1[3]
  halpha4l1=halphal1[4]
logLal=sum(l1d_it[,2:6]*log(exp(halphal1[1]+halphal1[2]*l1d_it[,1:5]+halphal1[3]*
          l1n_it+halphal1[4]*log(1+l1x_it[,1:5]))/(1+exp(halphal1[1]+halphal1[2]*
          l1d_it[,1:5]+ halphal1[3]*l1n_it+halphal1[4]*log(1+l1x_it[,1:5])))+
          (1-l1d_it[,2:6])* log(1-exp(halphal1[1]+halphal1[2]*l1d_it[,1:5]+halphal1[3]*
          l1n_it+halphal1[4]*log(1+l1x_it[,1:5]))/
          (1+exp(halphal1[1]+halphal1[2]*l1d_it[,1:5]+halphal1[3]
          *l1n_it+halphal1[4]*log(1+l1x_it[,1:5])))))}

halphal1=maxLik(loglikalphal1,start=c(-2,2,-2,0.1))
summary(halphal1)

loglikgammal1=function(hgammal1){
  hgamma1l1=hgammal1[1]
  hgamma2l1=hgammal1[2]
  logLga=sum(l1n_it[,2:5]*log(exp(hgammal1[1]+hgammal1[2]*l1n_it[,1:4])/
        (1+exp(hgammal1[1]+hgammal1[2]*l1n_it[,1:4])))+(1-l1n_it[,2:5])*
        log(1-exp(hgammal1[1]+hgammal1[2]*l1n_it[,1:4])/
        (1+exp(hgammal1[1]+hgammal1[2]*l1n_it[,1:4]))))
}
hgammal1=maxLik(loglikgammal1,start=c(-1,-2.5))
summary(hgammal1)
```

3.

3.(a)

We follow 5 steps to get the log-likelihood function for mixed type individual i.

Step 1:

Calculate the probability of d_{it} for type 1 individuals.

$$Pr(d_{it}|l=1) = \prod_{t=1}^{5}\left(\frac{exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1)}{1+exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1)}\right)^{d_{it}} \times \left(1-\frac{exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1)}{1+exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1)}\right)^{1-d_{it}}$$

Step 2:

Calculate the probability of n_{it} for type 1 individuals.

$$Pr(n_{it}|l=1) = \prod_{t=2}^{5}(\frac{exp(\gamma_{1,i}+\gamma_2 n_{it-1})}{1+exp(\gamma_{1,i}+\gamma_2 n_{it-1})})^{n_{it}}(1-\frac{exp(\gamma_{1,i}+l1\gamma_2 n_{it-1})}{1+exp(\gamma_{1,i}+\gamma_2 n_{it-1})})^{1-n_{it}})$$

Step 3:

Calculate the probability of d_{it} for type 2 individuals.

$$\begin{aligned} Pr(d_{it}|l=2) = \prod_{t=1}^{5}(&\frac{exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1))}{1+exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1))})^{d_{it}} \\ &\times (1-\frac{exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1))}{1+exp(\alpha_{1i}+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1))})^{1-d_{it}} \end{aligned}$$

Step 4:

Calculate the probability of n_{it} for type 2 individuals.

$$Pr(n_{it}|l=2) = \prod_{t=2}^{5}(\frac{exp(\gamma_{1,i}+\gamma_2 n_{it-1})}{1+exp(\gamma_{1,i}+\gamma_2 n_{it-1})})^{n_{it}}(1-\frac{exp(\gamma_{1,i}+l1\gamma_2 n_{it-1})}{1+exp(\gamma_{1,i}+\gamma_2 n_{it-1})})^{1-n_{it}})$$

Step 5:

The whole log-likelihood function for mixed type is as follows.

$$\begin{aligned} \ell(\alpha_{l=1,2},\gamma_{l=1,2}) = \sum_{i=1}^{300} ln(\{&Pr(d_{it}|l=1)\times Pr(l1 n_{it})\times Pr(l=1) \\ &+ Pr(d_{it}|l=2)\times Pr(n_{it}|l=2)\times Pr(l=2))\} \end{aligned}$$

To simulate 300 individuals with mixing types, simulate 300 Bernoulli experiments with success probability $1-\frac{0.5}{1+e}$ first, then get the simulation result, which shows that there are 257 type 1 individuals and 43 those of type 2. Finally, let the first 257 individuals be type 1, and the rest 43 individuals be type 2.

3.(b)

The estimation code is as follows. It seems that the estimated parameters are not setting near by the true values.

```
library(maxLik)

#generate 300 individuals randomly, 1 is for type 1 and 0 is for type 2
set.seed(123)
l=rbinom(300,1,1-0.5/(1+exp(1)))
l1=length(which(l==1))#l1=257,l2=43
#create a matrix, with 257 of type 1 and 43 of type 2 individuals
```

```
d_it=rbind(l1d_it[1:257,],l2d_it[1:43,])
n_it=rbind(l1n_it[1:257,],l2n_it[1:43,])
x_it=rbind(l1x_it[1:257,],l2x_it[1:43,])
```

```
#estimate the mixed type likelihood function
logLikall=function(param){
  halpha1=param[1]
  halpha2=param[2]
  halpha3=param[3]
  halpha4=param[4]
  halpha5=param[5]
  halpha6=param[6]
  halpha7=param[7]
  halpha8=param[8]
  hgamma1=param[9]
  hgamma2=param[10]
  hgamma3=param[11]
  hgamma4=param[12]
  hdelta=param[13]

  l1d=(exp(param[1]+param[2]*d_it[,1:5]+param[3]*n_it+param[4]*log(1+x_it[,1:5]))/
      (1+ exp(param[1]+param[2]*d_it[,1:5]+param[3]*n_it+param[4]*log(1+x_it[,1:5]))))
      ^(d_it[,2:6])* (1-exp(param[1]+param[2]*d_it[,1:5]+param[3]*n_it+param[4]*
      log(1+x_it[,1:5])))/1+exp(param[1]+param[2]*d_it[,1:5]+param[3]*n_it+
      param[4]*log(1+x_it[,1:5])) )^(1-d_it[,2:6])
    l1d15=apply(l1d,1,prod)

  l1n=(exp(param[9]+param[10]*n_it[,1:4])/(1+exp(param[9]+param[10]*n_it[,1:4])))
    ^(n_it[,2:5])* (1-exp(param[9]+param[10]*n_it[,1:4])/
    (1+exp(param[9]+param[10]*n_it[,1:4])))^(1-n_it[,2:5])
    l1n25=apply(l1n,1,prod)

  l2d=(exp(param[5]+param[6]*d_it[,1:5]+param[7]*n_it+param[8]*log(1+x_it[,1:5]))/
      (1+ exp(param[5]+param[6]*d_it[,1:5]+param[7]*n_it+param[8]*log(1+x_it[,1:5]))))^
      (d_it[,2:6])* (1-exp(param[5]+param[6]*d_it[,1:5]+param[7]*n_it+param[8]*
      log(1+x_it[,1:5])))/1+exp(param[5]+param[6]*d_it[,1:5]+param[7]*n_it+param[8]*
      log(1+x_it[,1:5])) )^(1-d_it[,2:6])
      l2d15=apply(l2d,1,prod)

  l2n=(exp(param[11]+param[12]*n_it[,1:4])/(1+exp(param[11]+param[12]*n_it[,1:4])))^
      (n_it[,2:5])*(1-exp(param[11]+param[12]*n_it[,1:4])/
      (1+exp(param[11]+param[12]*n_it[,1:4])))^(1-n_it[,2:5])
    l2n25=apply(l2n,1,prod)

   LogL=sum(log(l1d15*l1n25*(1-0.5/(1+exp(param[13])))+
            l2d15*l2n25*(0.5/(1+exp(param[13])))))
}
param=maxLik(logLikall,start=c(-2,2,-2,0.1,0,2,-2,0.1,-1,-2.5,1,-2.5,1))
summary(param)
```

3.(c)

To adopt EM algorithm, we first calculate the posterior probility of two types given the initial values of theta,

$\theta^0 = \{\delta^0, \alpha^0_{l=1}, \gamma^0_{l=1}, \alpha^0_{l=2}, \gamma^0_{l=2}\}$.

The posterior probility of type 1 is

$$Pr(l=1) = \frac{(1-\frac{0.5}{1+e^{\delta^0}}) \times \prod_{t=1}^{5}\{[Pr(d_{it}=1)]^{d_{it}}[Pr(d_{it}=0)]^{1-d_{it}}\} \times \prod_{t=2}^{5} Pr(n_{it})}{\prod_{t=1}^{5} Pr(d_{it}) \times \prod_{t=2}^{5} Pr(n_{it}) \times Pr(l=1) + \prod_{t=1}^{5} Pr(d_{it}) \times \prod_{t=2}^{5} Pr(n_{it}) \times Pr(l=2)}$$

The posterior probility of type 2 is

$$Pr(l=2) = \frac{(1-\frac{0.5}{1+e^{\delta^0}}) \times \prod_{t=1}^{5}\{[Pr(d_{it}=1)]^{d_{it}}[Pr(d_{it}=0)]^{1-d_{it}}\} \times \prod_{t=2}^{5} Pr(n_{it})}{\prod_{t=1}^{5} Pr(d_{it}) \times \prod_{t=2}^{5} Pr(n_{it}) \times Pr(l=1) + \prod_{t=1}^{5} Pr(d_{it}) \times \prod_{t=2}^{5} Pr(n_{it}) \times Pr(l=2)}$$

The ϵ function can be defined as

$$\begin{aligned}\epsilon(\theta|\theta_{t-1}) = \sum_{i=1}^{300}(Pr(l=1)[ln(1-\frac{0.5}{1+e^{\delta}}) + \sum_{t=1}^{5} ln(Pr(d_{it})) + \sum_{t=2}^{5} ln(Pr(n_{it}))] \\ + Pr(l=2)[ln(\frac{0.5}{1+e^{\delta}}) + \sum_{t=1}^{5} ln(Pr(d_{it})) + \sum_{t=2}^{5} ln(Pr(n_{it}))]).\end{aligned}$$

Then $\delta, \alpha_{l=1}, \gamma_{l=1}, \alpha_{l=2}$ and $\gamma_{l=2}$ can be estimated separately.

The likelihood function $\sum_i [Pr(l=1) \times ln(1-\frac{0.5}{1+e^{\delta}}) + Pr(l=2) \times ln(\frac{0.5}{1+e^{\delta}})]$ can be estimated to get δ^1 for the first round.

Same process can be conducted to obtain estimates of $\alpha^1_{l=1}, \gamma^1_{l=1}, \alpha^1_{l=2}, \gamma^1_{l=2}$.

Next, subsitute the first-round estimates into $Pr(l=1)$ and $Pr(l=2)$. Repeat the partial estimation process until estimated values converge.

We failed to write the code nicely to reduce the computing time here, so after iteration for 5 rounds, the estimation result is as follows.

$\hat{\alpha}^5_{(l=1)} = (-15.0361379, -1.9052144, -2.8929304, -34.4098994)$,

$\hat{\gamma}^5_{(l=1)} = (-11.2861093, 3.2885060)$,

$\hat{\alpha}^5_{(l=2)} = (9.2319668, 12.6651221, 10.8282671, -7.9744749)$,

$\hat{\gamma}^5_{(l=2)} = (16.4988095, 0.5688028)$,

$\hat{\delta}^5 = 8.4513753$.

Since the estimation result has not converged yet, most of the estimates are still far from the true values.

```
library(stats)

#theta is a 13*1 vector
thetaz=function(z,theta){
  if (z==1) {
     return (c(theta[1:6],theta[13]))
  }
  if (z==2) {
     return (c(theta[7:12],theta[13]))
```

```
  }
}


#likelihood function of individual i's d_it and n_it for all periods
pr_di=function(i,z,theta){
   di=(exp(thetaz(z,theta)[1]+thetaz(z,theta)[2]*d_it[i,1:5]+thetaz(z,theta)[3]*n_it[i,]+
      thetaz(z,theta)[4]*log(1+x_it[i,1:5]))/(1+exp(thetaz(z,theta)[1]+thetaz(z,theta)[2]*
      d_it[i,1:5]+thetaz(z,theta)[3]*n_it[i,]+
      thetaz(z,theta)[4]*log(1+x_it[i,1:5])))^(d_it[i,2:6])*
      (1-exp(thetaz(z,theta)[1]+thetaz(z,theta)[2]*d_it[i,1:5]+thetaz(z,theta)[3]*n_it[i,]+
      thetaz(z,theta)[4]*log(1+x_it[i,1:5]))/1+exp(thetaz(z,theta)[1]+thetaz(z,theta)[2]*
      d_it[i,1:5]+thetaz(z,theta)[3]*n_it[i,]+thetaz(z,theta)[4]*log(1+x_it[i,1:5])))^
      (1-d_it[i,2:6])
   di15=prod(di)
   return (di15)
}

pr_ni=function(i,z,theta){
   ni=(exp(thetaz(z,theta)[5]+thetaz(z,theta)[6]*n_it[i,1:4])/(1+exp(thetaz(z,theta)[5]+
      thetaz(z,theta)[6]*n_it[i,1:4])))^(n_it[i,2:5])*
      (1-exp(thetaz(z,theta)[5]+thetaz(z,theta)[6]*n_it[i,1:4])/(1+exp(thetaz(z,theta)[5]+
      thetaz(z,theta)[6]*n_it[i,1:4])))^(1-n_it[i,2:5])
   ni25=prod(ni)
   return (ni25)
}

#probability for both types
fz=function(z,theta){
  if (z==1){
    return (1-0.5/(1+exp(theta[13])))
  }
  if (z==2){
    return (0.5/(1+exp(theta[13])))
  }
}

#individual i's posterior probability
h_i=function(i, z, theta){
   return (pr_di(i,z,theta)*pr_ni(i,z,theta)*fz(z,theta)/(pr_di(i,z=1,theta)*
           pr_ni(i,z=1,theta)*fz(z=1,theta)+pr_di(i,z=2,theta)*
          pr_ni(i,z=2,theta)*fz(z=2,theta)))
}

theta_ini=c(-2,2,-2,0.1,-1,-2.5,0,2,-2,0.1,1,-2.5,1)


#epsilon function, need to sum i up
epsilon=function(theta_starlag1,theta_star){
  sumemi=0
  for (i in 1:300){
    sumemi=h_i(i,z=1,theta=theta_starlag1)*(log((fz(z=1,theta_star)))+
           log(pr_di(i,z=1,theta_star))+log(pr_ni(i,z=1,theta_star)))+
```

```
           h_i(i,z=2,theta=theta_starlag1)*(log(fz(z=2,theta_star))+
           log(pr_di(i,z=2,theta_star))+log(pr_ni(i,z=2,theta_star)))+sumemi
  }
  return (sumemi)
}

theta_try<-theta_ini
for (k in 1:5){
 theta_starlag1<-theta_try
 iterepsilon=function(theta_star){
    epsilon(theta_starlag1,theta_star)
 }
 theta_try<-optim(theta_ini,iterepsilon)$par

 distance_theta=dist(t(cbind(theta_try,theta_starlag1)))
 if (k==1){
    print(theta_try)
 }
 if (k==2){
    print(theta_try)
 }
 if (k==3){
    print(theta_try)
  }
 if (distance_theta<10^(-7)){
    print("converged successfully")
    print(k)
    break
  }

 if (k==5){
    print("not converged until 2000 times")
  }
}
```

3.2 Problems

1.

Step 1:

Specify the transition matrices $Fx(d_{it}=1)$ and $Fx(d_{it}=0)$.

Since state variables are $S_s=(d_{it-1},x_{it},n_{it})$, there are $2\times 51\times 2=204$ kinds of states in all. Denote S_s in the following way:

$$S_1=(0,0,0),S_2=(0,0,1),S_3=(0,1,0),S_4=(0,1,1),\ldots,$$
$$S_{101}=(0,50,0),S_{102}=(0,50,1),S_{103}=(1,0,0),S_{104}=(1,0,1),\ldots,$$
$$S_{203}=(1,50,0),S_{204}=(1,50,1).$$

Then $Fx(d_{it}=1)$ for both types can be defined as

$$Fx(d_{it}=1)=\begin{pmatrix} S1\to S_1 & S_1\to S_2 & \cdots & S_1\to S_{204}\\ & \vdots & \vdots & \\ & \vdots & \vdots & \\ & \vdots & \vdots & \\ S_{204}\to S_1 & \cdots & \cdots & S_{204}\to S_{204}\end{pmatrix}$$
$$=\begin{pmatrix} 0&0&\cdots&a&b&0&0&0&\cdots&0&0\\ 0&0&\cdots&c&d&0&0&0&\cdots&0&0\\ 0&0&\cdots&0&0&a&b&0&\cdots&0&0\\ 0&0&\cdots&0&0&a&b&0&\cdots&0&0\\ & & & & & \vdots & & & & & \\ & & & & & \vdots & & & & & \\ 0&0&\cdots&\cdots&\cdots&\cdots&\cdots&0&0&a&b\\ 0&0&\cdots&\cdots&\cdots&\cdots&\cdots&0&0&c&d\\ 0&\cdots&\cdots&\cdots&\cdots&a&b&0&\cdots&0&0\\ 0&\cdots&\cdots&\cdots&\cdots&c&d&0&\cdots&0&0\\ 0&\cdots&\cdots&\cdots&\cdots&\cdots&\cdots&a&b&\cdots&0\\ 0&\cdots&\cdots&\cdots&\cdots&\cdots&\cdots&c&d&\cdots&0\\ & & & & & \vdots & & & & & \\ & & & & & \vdots & & & & & \\ 0&0&\cdots&\cdots&\cdots&\cdots&\cdots&0&0&a&b\\ 0&0&\cdots&\cdots&\cdots&\cdots&\cdots&0&0&c&d\\ 0&0&\cdots&\cdots&\cdots&\cdots&\cdots&\cdots&\cdots&0&0\\ 0&0&\cdots&\cdots&\cdots&\cdots&\cdots&\cdots&\cdots&0&0\end{pmatrix}.$$

a,b,c,d are the transition probability of any particular status from 0 child to 0 child, 0 child to 1 child, 1 child to 0 child and 1 child to 1 child, respectively.

Similarly, $Fx(d_{it}=0)$ can also be defined for both types.

Step 2:

Write the Bellman Equation of $\overline{V}$ for both types.

$$\overline{V} = ln\left\{\sum_{d_{it}} exp[u(d_{it}) + \beta Fx(d_{it})\overline{V}]\right\}$$

Step 3:

Use the true value as initial values, and substitute them into the Bellman Equation. Then take some initial guess $\overline{V}^0$. We use a 204×1 zero vector here. After that, a new guess $\overline{V}^1$ can be calculated.

Step 4:

Repeat Step 3 till $\overline{V}$ for both types converges.

Step 5:

Following the way we defined states, $(d_{it-1} = 0, x_{it} = 10, n_{it} = 1)$ is the 22nd value in the $\overline{V}$ vector of type 1 and $(d_{it-1} = 1, x_{it} = 20, n_{it} = 0)$ is the 143rd value in the $\overline{V}$ vector of type 2.

```
library(SQUAREM)

#define Fx(d_it=0) and Fx(d_it=1) as 2 204*204 matrices
#2*2 small matrix for two types. each has elements of a,b,c,d.
f3=matrix(0,nrow=100,ncol=2)
f6=matrix(0,nrow=2,ncol=102)
f8=matrix(0,nrow=204,ncol=102)

Fxdit1=function(z,theta){
  if (z==1){
  return (cbind(f8,rbind(rbind(kronecker(diag(1,51),
        matrix(c(1/(1+exp(theta[5]+theta[6]*0)),1/(1+exp(theta[5]+theta[6]*1)),
        exp(theta[5]+theta[6]*0)/(1+exp(theta[5]+theta[6]*0)),exp(theta[5]+theta[6]*1)/
        (1+exp(theta[5]+theta[6]*1))),nrow=2,ncol=2)),
        cbind(f3,kronecker(diag(1,50),matrix(c(1/(1+exp(theta[5]+theta[6]*0)),
        1/(1+exp(theta[5]+theta[6]*1)),exp(theta[5]+theta[6]*0)/(1+exp(theta[5]+theta[6]*0)),
        exp(theta[5]+theta[6]*1)/(1+exp(theta[5]+theta[6]*1))),nrow=2,ncol=2)))),f6)))
  }

  if(z==2){
  return (cbind(f8,rbind(rbind(kronecker(diag(1,51),
         matrix(c(1/(1+exp(theta[11]+theta[12]*0)),1/(1+exp(theta[11]+theta[12]*1)),
         exp(theta[11]+theta[12]*0)/(1+exp(theta[11]+theta[12]*0)),
         exp(theta[11]+theta[12]*1)/(1+exp(theta[11]+theta[12]*1))),nrow=2,ncol=2)),
         cbind(f3,kronecker(diag(1,50),matrix(c(1/(1+exp(theta[11]+theta[12]*0)),
         1/(1+exp(theta[11]+theta[12]*1)),exp(theta[11]+theta[12]*0)/
        (1+exp(theta[11]+theta[12]*0)),exp(theta[11]+theta[12]*1)/
        (1+exp(theta[11]+theta[12]*1))),nrow=2,ncol=2)))),f6)))
  }
}

Fxdit0=function(z,theta){
```

```
      if (z==1){
      return (cbind(rbind(rbind(kronecker(diag(1,51),
             matrix(c(1/(1+exp(theta[5]+theta[6]*0)),1/(1+exp(theta[5]+theta[6]*1)),
            exp(theta[5]+theta[6]*0)/(1+exp(theta[5]+theta[6]*0)),
            exp(theta[5]+theta[6]*1)/(1+exp(theta[5]+theta[6]*1))),nrow=2,ncol=2)),
            cbind(f3,kronecker(diag(1,50),matrix(c(1/(1+exp(theta[5]+theta[6]*0)),
            1/(1+exp(theta[5]+theta[6]*1)),exp(theta[5]+theta[6]*0)/
            (1+exp(theta[5]+theta[6]*0)),exp(theta[5]+theta[6]*1)/
            (1+exp(theta[5]+theta[6]*1))),nrow=2,ncol=2))),f6),f8))
     }
  if (z==2) {
      return (cbind(rbind(rbind(kronecker(diag(1,51),
             matrix(c(1/(1+exp(theta[11]+theta[12]*0)),1/(1+exp(theta[11]+theta[12]*1)),
             exp(theta[11]+theta[12]*0)/(1+exp(theta[11]+theta[12]*0)),
             exp(theta[11]+theta[12]*1)/(1+exp(theta[11]+theta[12]*1))),nrow=2,ncol=2)),
             cbind(f3,kronecker(diag(1,50),matrix(c(1/(1+exp(theta[11]+theta[12]*0)),
             1/(1+exp(theta[11]+theta[12]*1)),exp(theta[11]+theta[12]*0)/
             (1+exp(theta[11]+theta[12]*0)),exp(theta[11]+theta[12]*1)/
             (1+exp(theta[11]+theta[12]*1))),nrow=2,ncol=2))),f6),f8))
    }
}

truetheta=c(-2,2,-2,0.1,-1,-2.5,0,2,-2,0.1,1,-2.5,1)
truel1Fxdit0=Fxdit0(z=1,theta=truetheta)
truel1Fxdit1=Fxdit1(z=1,theta=truetheta)
truel2Fxdit0=Fxdit0(z=2,theta=truetheta)
truel2Fxdit1=Fxdit1(z=2,theta=truetheta)

vbarini=c(rep(0,204))
vbarl1=vbarini
for (k in 1:2000){
  vbarl1_lag=vbarl1
   u0=c(rep(0,204));beta=0.95

  s1=c(rep(1,204))

  sditlag1=c(rep(0,102),rep(1,102))
  sxit=c(rep(0:50,each=2),rep(0:50,each=2))
  snit=c(rep(0:1,102) )
  su1=cbind(s1,sditlag1,sxit,snit)

  l1u1=su1%*%alphal1

#define the Bellman Equation of Vbar for type 1
  vbarl1=log(  exp(u0+beta*truel1Fxdit0%*%vbarl1_lag)+
               exp(l1u1+beta*truel1Fxdit1%*%vbarl1_lag)   )
  distance=dist(t(cbind(vbarl1,vbarl1_lag)))
   if (distance<10^(-7)){
    print("converged successfully")
    print(k)
    break
  }
```

```
   if (k==2000){
    print("not converged until 2000 times")
  }
}
vbarl1[22]

vbarl2=vbarini
for (k in 1:2000){
  vbarl2_lag=vbarl2
  l2u1=su1%*%alphal2

#define the Bellman Equation of Vbar for type 2
  vbarl2=log(  exp(beta*truel2Fxdit0%*%vbarl2_lag)+
               exp(l2u1+beta*truel2Fxdit1%*%vbarl2_lag)   )
  distance=dist(t(cbind(vbarl2,vbarl2_lag)))
   if (distance<10^(-7)){
    print("converged successfully")
    print(k)
    break
  }

   if (k==2000){
    print("not converged until 2000 times")
  }
}
vbarl2[143]
```

2.

2.(a)

Here, the individual i decide whether to work or not in each period t after the value function of the whole sample periods rather than some instantaneous utility function.

Period	0	1	2	3	4	5
Type 1	0.57	0.29	0.20	0.18	0.17	0.20
Type 2	0.51	0.52	0.56	0.61	0.60	0.58
Pooled	0.54	0.41	0.38	0.39	0.39	0.39

2.(b)

```
#count the number of people from d_t=0 to d_t+1=0 for both types
l1vd0d0=length(which(l1vd_it[,1]==0&l1vd_it[,2]==0))+
  length(which(l1vd_it[,2]==0&l1vd_it[,3]==0))+
  length(which(l1vd_it[,3]==0&l1vd_it[,4]==0))+
  length(which(l1vd_it[,4]==0&l1vd_it[,5]==0))+
  length(which(l1vd_it[,5]==0&l1vd_it[,6]==0))
l1vd0d0

l2vd0d0=length(which(l2vd_it[,1]==0&l2vd_it[,2]==0))+
  length(which(l2vd_it[,2]==0&l2vd_it[,3]==0))+
  length(which(l2vd_it[,3]==0&l2vd_it[,4]==0))+
  length(which(l2vd_it[,4]==0&l2vd_it[,5]==0))+
```

```
  length(which(l2vd_it[,5]==0&l2vd_it[,6]==0))

prvd0d0=(l1vd0d0+l2vd0d0)/(length(which(l1vd_it[,1]==0))+length(which(l1vd_it[,2]==0))+
        length(which(l1vd_it[,3]==0))+length(which(l1vd_it[,4]==0))+
        length(which(l1vd_it[,5]==0))+length(which(l2vd_it[,1]==0))+
        length(which(l2vd_it[,2]==0))+length(which(l2vd_it[,3]==0))+
        length(which(l2vd_it[,4]==0))+length(which(l2vd_it[,5]==0)))

#count the number of people from d_t=0 to d_t+1=1 for both types
l1vd0d1=length(which(l1vd_it[,1]==0&l1vd_it[,2]==1))+
  length(which(l1vd_it[,2]==0&l1vd_it[,3]==1))+
  length(which(l1vd_it[,3]==0&l1vd_it[,4]==1))+
  length(which(l1vd_it[,4]==0&l1vd_it[,5]==1))+
  length(which(l1vd_it[,5]==0&l1vd_it[,6]==1))

l2vd0d1=length(which(l2vd_it[,1]==0&l2vd_it[,2]==1))+
  length(which(l2vd_it[,2]==0&l2vd_it[,3]==1))+
  length(which(l2vd_it[,3]==0&l2vd_it[,4]==1))+
  length(which(l2vd_it[,4]==0&l2vd_it[,5]==1))+
  length(which(l2vd_it[,5]==0&l2vd_it[,6]==1))


prvd0d1=(l1vd0d1+l2vd0d1)/(length(which(l1vd_it[,1]==0))+
        length(which(l1vd_it[,2]==0))+length(which(l1vd_it[,3]==0))+
        length(which(l1vd_it[,4]==0))+length(which(l1vd_it[,5]==0))+
        length(which(l2vd_it[,1]==0))+length(which(l2vd_it[,2]==0))+
        length(which(l2vd_it[,3]==0))+length(which(l2vd_it[,4]==0))+
        length(which(l2vd_it[,5]==0))                  )


#count the number of people from d_t=1 to d_t+1=0 for both types
l1vd1d0=length(which(l1vd_it[,1]==1&l1vd_it[,2]==0))+
  length(which(l1vd_it[,2]==1&l1vd_it[,3]==0))+
  length(which(l1vd_it[,3]==1&l1vd_it[,4]==0))+
  length(which(l1vd_it[,4]==1&l1vd_it[,5]==0))+
  length(which(l1vd_it[,5]==1&l1vd_it[,6]==0))

l2vd1d0=length(which(l2vd_it[,1]==1&l2vd_it[,2]==0))+
  length(which(l2vd_it[,2]==1&l2vd_it[,3]==0))+
  length(which(l2vd_it[,3]==1&l2vd_it[,4]==0))+
  length(which(l2vd_it[,4]==1&l2vd_it[,5]==0))+
  length(which(l2vd_it[,5]==1&l2vd_it[,6]==0))

prvd1d0=(l1vd1d0+l2vd1d0)/(length(which(l1vd_it[,1]==1))+
        length(which(l1vd_it[,2]==1))+length(which(l1vd_it[,3]==1))+
        length(which(l1vd_it[,4]==1))+length(which(l1vd_it[,5]==1))+
        length(which(l2vd_it[,1]==1))+length(which(l2vd_it[,2]==1))+
        length(which(l2vd_it[,3]==1))+length(which(l2vd_it[,4]==1))+
        length(which(l2vd_it[,5]==1))                  )

#count the number of people from d_t=1 to d_t+1=1 for both types
l1vd1d1=length(which(l1vd_it[,1]==1&l1vd_it[,2]==1))+
  length(which(l1vd_it[,2]==1&l1vd_it[,3]==1))+
```

```
  length(which(l1vd_it[,3]==1&l1vd_it[,4]==1))+
  length(which(l1vd_it[,4]==1&l1vd_it[,5]==1))+
  length(which(l1vd_it[,5]==1&l1vd_it[,6]==1))

l2vd1d1=length(which(l2vd_it[,1]==1&l2vd_it[,2]==1))+
  length(which(l2vd_it[,2]==1&l2vd_it[,3]==1))+
  length(which(l2vd_it[,3]==1&l2vd_it[,4]==1))+
  length(which(l2vd_it[,4]==1&l2vd_it[,5]==1))+
  length(which(l2vd_it[,5]==1&l2vd_it[,6]==1))

prvd1d1=(l1vd1d1+l2vd1d1)/(length(which(l1vd_it[,1]==1))+
        length(which(l1vd_it[,2]==1))+length(which(l1vd_it[,3]==1))+
        length(which(l1vd_it[,4]==1))+length(which(l1vd_it[,5]==1))+
        length(which(l2vd_it[,1]==1))+length(which(l2vd_it[,2]==1))+
        length(which(l2vd_it[,3]==1))+length(which(l2vd_it[,4]==1))+
        length(which(l2vd_it[,5]==1))                    )

l1prvd0d0=l1vd0d0/(length(which(l1vd_it[,1]==0))+
                     length(which(l1vd_it[,2]==0))+
                     length(which(l1vd_it[,3]==0))+
                     length(which(l1vd_it[,4]==0))+
                     length(which(l1vd_it[,5]==0)))

l2prvd0d0=l2vd0d0/(length(which(l2vd_it[,1]==0))+
                     length(which(l2vd_it[,2]==0))+
                     length(which(l2vd_it[,3]==0))+
                     length(which(l2vd_it[,4]==0))+
                     length(which(l2vd_it[,5]==0)))


l1prvd0d1=l1vd0d1/(length(which(l1vd_it[,1]==0))+
                     length(which(l1vd_it[,2]==0))+length(which(l1vd_it[,3]==0))+
                     length(which(l1vd_it[,4]==0))+length(which(l1vd_it[,5]==0)))
l2prvd0d1=l2vd0d1/(length(which(l2vd_it[,1]==0))+length(which(l2vd_it[,2]==0))+
                     length(which(l2vd_it[,3]==0))+length(which(l2vd_it[,4]==0))+
                     length(which(l2vd_it[,5]==0)))



l1prvd1d0=l1vd1d0/(length(which(l1vd_it[,1]==1))+length(which(l1vd_it[,2]==1))+
                     length(which(l1vd_it[,3]==1))+length(which(l1vd_it[,4]==1))+
                     length(which(l1vd_it[,5]==1)))

l2prvd1d0=l2vd1d0/(length(which(l2vd_it[,1]==1))+length(which(l2vd_it[,2]==1))+
                     length(which(l2vd_it[,3]==1))+length(which(l2vd_it[,4]==1))+
                     length(which(l2vd_it[,5]==1)))

l1prvd1d1=l1vd1d1/(length(which(l1vd_it[,1]==1))+length(which(l1vd_it[,2]==1))+
                     length(which(l1vd_it[,3]==1))+length(which(l1vd_it[,4]==1))+
                     length(which(l1vd_it[,5]==1)))
l2prvd1d1=l2vd1d1/(length(which(l2vd_it[,1]==1))+length(which(l2vd_it[,2]==1))+
                     length(which(l2vd_it[,3]==1))+length(which(l2vd_it[,4]==1))+
                     length(which(l2vd_it[,5]==1)))
```

```
l1prvd0d0;l1prvd0d1;l1prvd1d0;l1prvd1d1;l2prvd0d0;l2prvd0d1;
l2prvd1d0;l2prvd1d1;prvd0d0;prvd0d1;prvd1d0;prvd1d1
```

Therefore, the transition matrices of working decision for all types are as follows.

Type 1	$d_{t-1}=0$	$d_{t-1}=1$
$d_t=0$	0.89	0.53
$d_t=1$	0.11	0.47

Type 2	$d_{t-1}=0$	$d_{t-1}=1$
$d_t=0$	0.61	0.28
$d_t=1$	0.39	0.72

Pooled	$d_{t-1}=0$	$d_{t-1}=1$
$d_t=0$	0.79	0.36
$d_t=1$	0.21	0.64

3.

3.(a)

To obtain the log-likelihood function, $Pr(d_{it}=1)$ needs to be calculated first.

$$\begin{aligned} Pr(d_{it}=1) &= \frac{e^{V(d_{it}=1)}}{e^{V(d_{it}=1)}+e^{V(d_{it}=0)}} \\ &= \frac{e^{\alpha_1+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1)+\beta Fx(d_{it}=1)_S\overline{V}}}{e^{\alpha_1+\alpha_2 d_{it-1}+\alpha_3 n_{it}+\alpha_4 ln(x_{it}+1)+\beta Fx(d_{it}=1)_S\overline{V}}+e^{\beta Fx(d_{it}=0)_S\overline{V}}} \end{aligned}$$

$Fx(d_{it}=1)_S$ is the S-th row of $Fx(d_{it}=1)$, where $S=(d_{it-1},x_{it},n_{it})$.

$Pr(d_{it}=0)=[1-Pr(d_{it}=1)]$ can also be calculated in the same way.

The log-likelihood function can then be written as follows.

$$\begin{aligned} \ell = \sum_i \Bigg\{ &\sum_t (d_{it} lnPr(d_{it}=1)+(1-d_{it})lnPr(d_{it}=0)) \\ &+\sum_t (n_{it} lnPr(n_{it}=1)+(1-n_{it})lnPr(n_{it}=0)) \Bigg\} \end{aligned}$$

Take an initial guess of α^0 and γ^0 before calculating the first round estimation, where γ^0 is used to calculate $Fx(d_{it})$. Then use the following equation to obtain a converged $\overline{V^1}$.

$$\overline{V}_{n+1} = ln\Bigg\{ \sum_{d_{it}} exp[u(d_{it})+\beta Fx(d_{it})\overline{V}_n] \Bigg\}$$

After that, substitute $\overline{V^1}$ into the log-likelihood function to obtain estimation of α^1 and γ^1. Repeat these steps till $\hat{\alpha}$ and $\hat{\gamma}$ converge.

```
vbar=function(theta){
  vbarini=c(rep(0,204))
  vbarl1=vbarini
  for (k in 1:2000){
    vbarl1_lag=vbarl1
    u0=c(rep(0,204));beta=0.95

    s1=c(rep(1,204))

    sditlag1=c(rep(0,102),rep(1,102))
    sxit=c(rep(0:50,each=2),rep(0:50,each=2))
    snit=c(rep(0:1,102) )
    su1=cbind(s1,sditlag1,sxit,snit)

    l1u1estim=su1%*%theta[1:4]

#define the Bellman Equation of Vbar for type 1
    vbarl1=log(  exp(u0+beta*Fxdit0(z=1,theta)%*%vbarl1_lag)+
                 exp(l1u1estim+beta*Fxdit1(z=1,theta)%*%vbarl1_lag)   )
    distance_vbar=dist(t(cbind(vbarl1,vbarl1_lag)))

    if (distance_vbar<10^(-7)){
      break
    }

    if (k==2000){
      print("not converged until 2000 times")
    }
  }
  return(vbarl1)
}

#likelihood for individual i's d_it and n_it of all periods
pr_vdi=function(i,z,theta,vbar){
   fxdit115=t(c(Fxdit1(z=1,theta)[(1+102*l1vd_it[i,1]+2*l1vx_it[i,1]+l1n_it[i,1]),]%*%vbar
             ,Fxdit1(z=1,theta)[(1+102*l1vd_it[i,2]+2*l1vx_it[i,2]+l1n_it[i,2]),]%*%vbar
             ,Fxdit1(z=1,theta)[(1+102*l1vd_it[i,3]+2*l1vx_it[i,3]+l1n_it[i,3]),]%*%vbar
             ,Fxdit1(z=1,theta)[(1+102*l1vd_it[i,4]+2*l1vx_it[i,4]+l1n_it[i,4]),]%*%vbar
             ,Fxdit1(z=1,theta)[(1+102*l1vd_it[i,5]+2*l1vx_it[i,5]+l1n_it[i,5]),]%*%vbar))
   fxdit015=t(c(Fxdit0(z=1,theta)[(1+102*l1vd_it[i,1]+2*l1vx_it[i,1]+l1n_it[i,1]),]%*%vbar
             ,Fxdit0(z=1,theta)[(1+102*l1vd_it[i,2]+2*l1vx_it[i,2]+l1n_it[i,2]),]%*%vbar
             ,Fxdit0(z=1,theta)[(1+102*l1vd_it[i,3]+2*l1vx_it[i,3]+l1n_it[i,3]),]%*%vbar
             ,Fxdit0(z=1,theta)[(1+102*l1vd_it[i,4]+2*l1vx_it[i,4]+l1n_it[i,4]),]%*%vbar
             ,Fxdit0(z=1,theta)[(1+102*l1vd_it[i,5]+2*l1vx_it[i,5]+l1n_it[i,5]),]%*%vbar))
   di=(exp(theta[1+6*(z-1)]+theta[2+6*(z-1)]*l1vd_it[i,1:5]+theta[3+6*(z-1)]*l1n_it[i,]+
            theta[4+6*(z-1)]*log(1+l1vx_it[i,1:5])+beta*fxdit115)/
         (exp(theta[1+6*(z-1)]+theta[2+6*(z-1)]*l1vd_it[i,1:5]+theta[3+6*(z-1)]*l1n_it[i,]
              +theta[4+6*(z-1)]*log(1+l1vx_it[i,1:5])+beta*fxdit115)+
         (exp(beta*fxdit015))))  ^(l1vd_it[i,2:6])*


      ((exp(beta*fxdit015))/
         (exp(theta[1+6*(z-1)]+theta[2+6*(z-1)]*l1vd_it[i,1:5]+theta[3+6*(z-1)]*l1n_it[i,]+
```

```
                theta[4+6*(z-1)]*log(1+l1vx_it[i,1:5])+beta*fxdit115)+
         (exp(beta*fxdit015))))^(1-l1vd_it[i,2:6])
   di15=prod(di)
   return (log(di15))
}

pr_ni=function(i,z,theta){
   ni=(exp(theta[5+6*(z-1)]+theta[6+6*(z-1)]*l1n_it[i,1:4])/(1+exp(theta[5+6*(z-1)]+
      theta[6+6*(z-1)]*l1n_it[i,1:4])))^(l1n_it[i,2:5])*
      (1-exp(theta[5+6*(z-1)]+theta[6+6*(z-1)]*l1n_it[i,1:4])/
      (1+exp(theta[5+6*(z-1)]+theta[6+6*(z-1)]*l1n_it[i,1:4])))^(1-l1n_it[i,2:5])
   ni25=prod(ni)
   return (log(ni25))
}

l1loglik=function(theta,vbar){
  sumi=0
  for (i in 1:300){
    sumi=pr_vdi(i,z=1,theta,vbar)+pr_ni(i,z=1,theta)+sumi
  }
  return(sumi)
}
truethetal1=c(-2,2,-2,0.1,-1,-2.5)
theta_new_guess=truethetal1
for (k in 1:5){
  theta_guess=theta_new_guess
  iterl1loglik=function(theta){
    return(l1loglik(theta,vbar=vbar(theta_guess)))
  }
  theta_new_guess=optim(truethetal1,iterl1loglik)$par
  distance_theta_guess=dist(t(cbind(theta_new_guess,theta_guess)))

    if (k==1){
    print(theta_new_guess)
    }
    if (k==2){
    print(theta_new_guess)
    }
    if (k==3){
    print(theta_new_guess)
    }
  if (distance_theta_guess<10^(-7)){
     print("converged successfully")
     print(k)
     break
  }

  if (k==5){
     print("not converged until 5 times")
  }
}
```

3.(b)

q-NPL Algorithm

Step 1:

Take an initial guess $\theta^0 = (\alpha_1^0, \alpha_2^0, \alpha_3^0, \alpha_4^0, \gamma_1^0, \gamma_2^0)$ and $\overline{V}^0$, then use following Bellman Equation for 3 times(q=3). Denote the result as $\overline{V}^3$.

$$\overline{V}_{n+1} = ln\left\{ \sum_{d_{it}} exp[u(d_{it}) + \beta Fx(d_{it})\overline{V}_n] \right\}$$

Step 2:

Substitute $\overline{V}^3$ into the log-likelihood function to obtain a new estimation of parameter vector θ^1.

$$\ell = \sum_i \left\{ \sum_t (d_{it} lnPr(d_{it} = 1) + (1 - d_{it}) lnPr(d_{it} = 0)) \right.$$
$$\left. + \sum_t (n_{it} lnPr(n_{it} = 1) + (1 - n_{it}) lnPr(n_{it} = 0)) \right\}$$

Step 3:

Repeat Step 1 and Step 2 until $\theta^\star_{q=3}$ converges.

4.

4.(a)

The log-likelihood function can be written as follows.

$$\ell = \sum_i ln \left\{ \prod_t Pr(d_{it}) \prod_t Pr(n_{it}) \times Pr(l = 1) + \prod_t Pr(d_{it}) \prod_t Pr(n_{it}) \times Pr(l = 2) \right\}$$

Bellman Equation for each type can be written as follows.

$$\overline{V}_{n+1}(\cdot|l = 1) = ln\left\{ \sum_{d_{it}} exp[u(d_{it}) + \beta Fx(d_{it}|l = 1)\overline{V}_n(\cdot|l = 1)] \right\}$$

$$\overline{V}_{n+1}(\cdot|l = 2) = ln\left\{ \sum_{d_{it}} exp[u(d_{it}) + \beta Fx(d_{it}|l = 2)\overline{V}_n(\cdot|l = 2)] \right\}$$

Take an initial guess for $\alpha^0_{l=1}, \gamma^0_{l=1}, \delta^0$, and some converged $\overline{V}(\cdot|l = 1)^1$ can be calculated using Bellman Equation for type 1 individuals. Same process can be done for type 2 individuals.

Subsitute $\overline{V}(\cdot|l = 1)^1$ and $\overline{V}(\cdot|l = 2)^1$ into the log-likelihood function to estimate all 13 parameters denoted by θ^1. Substitute θ^1 into Bellman Equation to obtain converged $\overline{V}^2$ for both types. Repeat these steps till a vector of converged parameters are obtained.

4.(b)

The ϵ function can be defined as

$$\epsilon(\theta|\theta_{t-1}) = \sum_{i=1}^{300}(Pr(l=1)[ln(1-\frac{0.5}{1+e^{\delta}}) + \sum_{t=1}^{5} ln(Pr(d_{it})) + \sum_{t=2}^{5} ln(Pr(n_{it}))]$$
$$+ Pr(l=2)[ln(\frac{0.5}{1+e^{\delta}}) + \sum_{t=1}^{5} ln(Pr(d_{it})) + \sum_{t=2}^{5} ln(Pr(n_{it}))]).$$

Take an initial guess for all 13 parameters, denoted by θ^0, and then calculate $Pr(l=1)$ and $Pr(l=2)$.

The following objective function is maximized to obtain estimation for δ^1.

$$\sum_i \left\{ Pr(l=1) \times ln(1-\frac{0.5}{1+e^{\delta}}) + Pr(l=2) \times ln(\frac{0.5}{1+e^{\delta}}) \right\}$$

Obtain $\overline{V}(\cdot|l=1)$ using Bellman Equation with θ^0, then substitute $\overline{V}(\cdot|l=1)$ into the following objective function. Maximize it to obtain the estimation for $\alpha^1_{l=1}$.

$$\sum_i \left\{ Pr(l=1) \times \sum_t ln(Pr(d_{it})) \right\}$$

Similar objective function can be constructed to obtain the estimation for $\gamma^1_{l=1}$.

$$\sum_i \left\{ Pr(l=1) \times \sum_t ln(Pr(n_{it})) \right\}$$

Same process can be done to obtain the estimation of $\alpha^1_{l=2}$ and $\gamma^1_{l=2}$ for type 2 individuals.

Finally, repeat these steps to obtain a vector of estimated parameters with converged values.