Appendix B

Appendix B1

Fitting MLMM using the MDLV MATLAB toolbox (cf. Table 2, Table 3, and Table 4)

The script for fitting MLMM is included in 'EVSdata_MLMM.m' as listed below. The required 4D data (Data4D) is created earlier (see Appendix A2). Four models are fitted to the data by specifying the number of clusters to be 2 or 3 and the number of classes to be 2 or 3. The required starting values are randomly generated using the function 'StartingValue_RandomV1'. The main function to estimate the parameters of the MLMM is 'MLMM.m'. The subfunction 'Loglikelihood_MLMM.m' is required to compute the likelihood value at each iteration.

EVSdata_MLMM.m

```
%% ======   Fit MLMM   ======
%% === 2-clutsers 2-Classes MLMM ===
% Specify the parameters:
G = 31; % Number of groups
Ng= 50; % Number of subjects per group
J=10;   % Number of items
T=2;    % Number of time points
L = 2;  % Number of latent clusters
M = 2;  % Number of latent classes

% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

%  Fit the MLMM
[lk,iteration,G,Ng,J,T,L,M,PH_g,PXgivenH,TauGivenH,CondResProd,Est_h,np,AIC,B
IC]=...
    MLMM(Data4D,Data2D,G,Ng,L,M,J,T,PH_g,PXgivenH,TauGivenH,CondResProd)
%% === 2-clutsers 3-Classes MLMM ===
% Specify the parameters:
G = 31; Ng= 50; J=10;  T=2;
L = 2;  % Number of latent clusters
M = 3;  % Number of latent classes

% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

%  Fit the MLMM
[lk,iteration,G,Ng,J,T,L,M,PH_g,PXgivenH,TauGivenH,CondResProd,Est_h,np,AIC,B
IC]=...
    MLMM(Data4D,Data2D,G,Ng,L,M,J,T,PH_g,PXgivenH,TauGivenH,CondResProd)
```

```
%% === 3-clutsers 2-Classes MLMM ===
% Specify the parameters:
G = 31; Ng= 50; J=10;  T=2;
L = 3;  % Number of latent clusters
M = 2;  % Number of latent classes

% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

%  Fit the MLMM
[lk,iteration,G,Ng,J,T,L,M,PH_g,PXgivenH,TauGivenH,CondResProd,Est_h,np,AIC,B
IC]=...
    MLMM(Data4D,Data2D,G,Ng,L,M,J,T,PH_g,PXgivenH,TauGivenH,CondResProd)

%% === 3-clutsers 3-Classes MLMM ===
% Specify the parameters:
G = 31; Ng= 50; J=10;  T=2;
L = 3;  % Number of latent clusters
M = 3;  % Number of latent classes

% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

%  Fit the MLMM
[lk,iteration,G,Ng,J,T,L,M,PH_g,PXgivenH,TauGivenH,CondResProd,Est_h,np,AIC,B
IC]=...
    MLMM(Data4D,Data2D,G,Ng,L,M,J,T,PH_g,PXgivenH,TauGivenH,CondResProd)
```

Appendix B2

Fitting MLCM using the MDLV MATLAB toolbox (cf. Table 2, Table 3, and Table 4)

The script for fitting MLCM is provided below in 'EVSdata_MLCM.m'. The data ItemsT1_3D and ItemsT2_3D contain responses to the 10 items for Wave 3 and Wave 4, respectively. Each data set is a matrix of $G \times Ng \times J$. Four models are fitted to each data set: 2 clusters by 2 classes, 2 clusters by 3 classes, 3 clusters by 2 classes, and 3 clusters by 3 classes. After specifying the model parameters, the starting values can be generated using the function 'StartingValue_RandomV1'. The parameters of MLCM are estimated using the function 'MLCM.m' which requires the sub-function 'Loglikelihood_MLCM.m' to calculate the log-likelihood value.

EVSdata_MLCM.m:

```
%% ======   Fit MLCM   ======
%% === Wave 3 ===
%% W3 === 2-clutsers 2-Classes MLCM ===
% Specify the parameters:
G = 31; % Number of groups
```

```
Ng= 50; % Number of subjects per group
J=10;   % Number of items
T=1;    % Number of time points
L = 2;  % Number of latent clusters
M = 2;  % Number of latent classes
% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

[lk,iteration,G,n_g,J,L,M,est_PH_g,est_PXgivenH,est_CondResProd,Est_h,np,AIC,
BIC]=...
    MLCM(ItemsT1_3D,G,Ng,L,M,J,PH_g,PXgivenH,CondResProd)
%% W3 === 2-clutsers 3-Classes MLCM ===
% Specify the parameters:
G = 31; Ng= 50; J=10; T=1;
L = 2;  % Number of latent clusters
M = 3;  % Number of latent classes

% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

[lk,iteration,G,n_g,J,L,M,est_PH_g,est_PXgivenH,est_CondResProd,Est_h,np,AIC,
BIC]=...
    MLCM(ItemsT1_3D,G,Ng,L,M,J,PH_g,PXgivenH,CondResProd)

%% W3 === 3-clutsers 2-Classes MLCM ===
% Specify the parameters:
G = 31; Ng= 50; J=10; T=1;
L = 3;  % Number of latent clusters
M = 2;  % Number of latent classes

% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

[lk,iteration,G,n_g,J,L,M,est_PH_g,est_PXgivenH,est_CondResProd,Est_h,np,AIC,
BIC]=...
    MLCM(ItemsT1_3D,G,Ng,L,M,J,PH_g,PXgivenH,CondResProd)

%% W3 === 3-clutsers 3-Classes MLCM ===
% Specify the parameters:
G = 31; Ng= 50; J=10; T=1;
L = 3;  % Number of latent clusters
M = 3;  % Number of latent classes

% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

[lk,iteration,G,n_g,J,L,M,est_PH_g,est_PXgivenH,est_CondResProd,Est_h,np,AIC,
BIC]=...
    MLCM(ItemsT1_3D,G,Ng,L,M,J,PH_g,PXgivenH,CondResProd)
%% === Wave 4 ===
%% W4 === 2-clutsers 2-Classes MLCM ===
```

```
% Specify the parameters:
G = 31; Ng= 50; J=10; T=1;
L = 2;  % Number of latent clusters
M = 2;  % Number of latent classes
% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

[lk,iteration,G,n_g,J,L,M,est_PH_g,est_PXgivenH,est_CondResProd,Est_h,np,AIC,
BIC]=...
    MLCM(ItemsT2_3D,G,Ng,L,M,J,PH_g,PXgivenH,CondResProd)

%% W4 === 2-clutsers 3-Classes MLCM ===
% Specify the parameters:
G = 31; Ng= 50; J=10; T=1;
L = 2;  % Number of latent clusters
M = 3;  % Number of latent classes
% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

[lk,iteration,G,n_g,J,L,M,est_PH_g,est_PXgivenH,est_CondResProd,Est_h,np,AIC,
BIC]=...
    MLCM(ItemsT2_3D,G,Ng,L,M,J,PH_g,PXgivenH,CondResProd)
%% W4 === 3-clutsers 2-Classes MLCM ===
% Specify the parameters:
G = 31; Ng= 50; J=10; T=1;
L = 3;  % Number of latent clusters
M = 2;  % Number of latent classes

% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

[lk,iteration,G,n_g,J,L,M,est_PH_g,est_PXgivenH,est_CondResProd,Est_h,np,AIC,
BIC]=...
    MLCM(ItemsT2_3D,G,Ng,L,M,J,PH_g,PXgivenH,CondResProd)
%% W4 === 3-clutsers 3-Classes MLCM ===
% Specify the parameters:
G = 31; Ng= 50; J=10; T=1;
L = 3;  % Number of latent clusters
M = 3;  % Number of latent classes
% generate the starting values
[PH_g,PXgivenH,TauGivenH,TauGivenHT,CondResProd]=StartingValue_RandomV1(1,M,L
,T,J);

[lk,iteration,G,n_g,J,L,M,est_PH_g,est_PXgivenH,est_CondResProd,Est_h,np,AIC,
BIC]=...
    MLCM(ItemsT2_3D,G,Ng,L,M,J,PH_g,PXgivenH,CondResProd)
```

Appendix B3

Fitting LMM using the MDLV MATLAB toolbox (cf. Table 2 and Table 5)

The script for fitting LMM is included in 'EVSdata_LMM.m' as shown below. Data3D is created with the script 'EVSdata.m' (see Appendix A2). The starting values for model parameters are required (Start_PX, Start_Tau, and Start_CondResProd) and are randomly generated in this example. Four models with number of latent classes ranging from 2 to 5 are fitted. The number of latent classes is specified by assigning a value to M. The main function to estimate the parameters of the LMM is 'LMM.m'. The sub-function 'Loglikelihood_LMM.m' (nested in 'LMM.m') is used to calculate the log-likelihood value.

EVSdata_LMM.m:

```
%% ======   Fit LMM   ======
%% === 2-Classes LMM ===
N=(31*50);    % Number of Total subjects (G*Ng)
T=2;          % Number of time points
J=10;         % Number of items
M=2;          % Number of latent classes
% generate the starting values: PX
Start_PX=rand(M,1); Start_PX=Start_PX/csum(Start_PX);

% Starting value for Tau (random start)
Start_Tau = (0.1*rand(M,M)+ones(M,M))/M;
Start_Tau = rdiv(Start_Tau,rsum(Start_Tau));
Start_Tau=reshape(repmat(Start_Tau,1,T),M,M,T);

%Start_CondResProd;
Start_CondResProd =rand(M,J);

[lk,iteration,n,T,J,M,pi1,TranMatrix,rho,np,AIC,BIC]...
    = LMM(Data3D,Start_PX,Start_Tau,Start_CondResProd,N,T,J,M)

%% === 3-Classes LMM ===
N=(31*50);    T=2;    J=10;
M=3;          % Number of latent classes

Start_PX=rand(M,1); Start_PX=Start_PX/csum(Start_PX);
Start_Tau = (0.1*rand(M,M)+ones(M,M))/M;
Start_Tau = rdiv(Start_Tau,rsum(Start_Tau));
Start_Tau=reshape(repmat(Start_Tau,1,T),M,M,T);
Start_CondResProd =rand(M,J);

[lk,iteration,n,T,J,M,pi1,TranMatrix,rho,np,AIC,BIC]...
    = LMM(Data3D,Start_PX,Start_Tau,Start_CondResProd,N,T,J,M)

%% === 4-Classes LMM ===
N=(31*50);    T=2;    J=10;
M=4;          % Number of latent classes

Start_PX=rand(M,1); Start_PX=Start_PX/csum(Start_PX);
Start_Tau = (0.1*rand(M,M)+ones(M,M))/M;
Start_Tau = rdiv(Start_Tau,rsum(Start_Tau));
Start_Tau=reshape(repmat(Start_Tau,1,T),M,M,T);
Start_CondResProd =rand(M,J);
```

```
[lk,iteration,n,T,J,M,pi1,TranMatrix,rho,np,AIC,BIC]...
    = LMM(Data3D,Start_PX,Start_Tau,Start_CondResProd,N,T,J,M)

%% === 5-Classes LMM ===
N=(31*50);    T=2;    J=10;
M=5;          % Number of latent classes

Start_PX=rand(M,1); Start_PX=Start_PX/csum(Start_PX);
Start_Tau = (0.1*rand(M,M)+ones(M,M))/M;
Start_Tau = rdiv(Start_Tau,rsum(Start_Tau));
Start_Tau=reshape(repmat(Start_Tau,1,T),M,M,T);
Start_CondResProd =rand(M,J);

[lk,iteration,n,T,J,M,pi1,TranMatrix,rho,np,AIC,BIC]...
    = LMM(Data3D,Start_PX,Start_Tau,Start_CondResProd,N,T,J,M)
```

Appendix B4

Fitting LCM using the MDLV MATLAB toolbox (cf. Table 2 and Table 5)

The script for fitting LCM to the sample data from EVS is given below as "`EVSdata_LCM.m`". Data `ItemsT1_2D` and `ItemsT2_2D` can be obtained by running the script '`EVSdata.m`' provided in Appendix A2. Four models, ranging from 2 to 5 latent classes, are fitted to the data for Wave 3 and Wave 4 separately. The function '`aggregate.m`' is used to construct a matrix of unique response patterns (for the 10 items) and a vector of the corresponding frequencies for the patterns. The main function 'LCM.m' is used to estimate the model parameters.

`EVSdata_LCM.m:`

```
%% ======   Fit LCM   ======
% [lk,it,n,J,pi1,Rho,np,AIC,BIC] = LCM_V2(S,yv,M,start)
% start: 1 -> random starts
% Data LCM:  Wave3:ItemsT1_2D
%            Wave4:ItemsT2_2D
[S_T1,yv_T1] = aggregate(ItemsT1_2D);
[S_T2,yv_T2] = aggregate(ItemsT2_2D);
% aggregate function: obtain matrix with unique response patterns and
% corresponding frequencies for each pattern.

%% === Wave 3 ===
% Specify the parameters:
M = 2;  % Number of latent classes = 2
[lk,it,n,J,pi1,Rho,np,AIC,BIC]=LCM(S_T1,yv_T1,M,1)
M = 3;  % Number of latent classes = 3
[lk,it,n,J,pi1,Rho,np,AIC,BIC]=LCM(S_T1,yv_T1,M,1)
M = 4;  % Number of latent classes = 4
[lk,it,n,J,pi1,Rho,np,AIC,BIC]=LCM(S_T1,yv_T1,M,1)
M = 5;  % Number of latent classes = 5
[lk,it,n,J,pi1,Rho,np,AIC,BIC]=LCM(S_T1,yv_T1,M,1)
```

```
%% === Wave 4 ===
% Specify the parameters:
M = 2;  % Number of latent classes = 2
[lk,it,n,J,pi1,Rho,np,AIC,BIC]=LCM(S_T2,yv_T2,M,1)
M = 3;  % Number of latent classes = 3
[lk,it,n,J,pi1,Rho,np,AIC,BIC]=LCM(S_T2,yv_T2,M,1)
M = 4;  % Number of latent classes = 4
[lk,it,n,J,pi1,Rho,np,AIC,BIC]=LCM(S_T2,yv_T2,M,1)
M = 5;  % Number of latent classes = 5
[lk,it,n,J,pi1,Rho,np,AIC,BIC]=LCM(S_T2,yv_T2,M,1)
```