A SAS Macro for Hierarchical Data Simulation

Wei Sun, University of Southern California, Los Angeles, CA
Chih-Ping Chou, University of Southern California, Los Angeles, CA

ABSTRACT

Behavioral and social data usually have a hierarchical (multilevel) structure. Within the hierarchical model, each of the levels in the data structure is represented by a sub model. Each sub-model contains the parameters on the structural relationship. The parameters contained at the lower level are used as the outcome variables at the higher level model.

A SAS Macro has been developed to simulate normal/non-normal data with hierarchical structure. Skewness and Kurtosis are used to specify the non-normality.

INTRODUCTION

The hierarchical (multilevel) structured data is very common in behavioral and social field. The hierarchical structure indicates that the sources from which data were collected are often "nested" within other larger social context. For example, students are nested within school; residents are nested within community. For a two-level hierarchical model, it consists of two sub-models: one sub-model for each level. Each sub-model contains the parameters on the structural relationship among units in each level. The parameters contained at the lower level, e.g. student or resident, are used as the outcome variables at the higher level , e.g. school or community, model.

A general two-level model[1] can be expressed as:

Level-1 model:

$$Y_{ij}=\beta_{0j}+\Sigma\beta_{qj}x_{qij}+e_{ij}$$

where

Y_{ij}: level-1 outcome variable for case i in group j

β_{qj}: level-1 regression coefficients, q=0, 1 ,…, Q; β_{0j}: refers to intercept;

x_{qij} : level-1 predictor q for case i in unit j;

e_{ij} : level-1 random effect; and

σ_e^2 : variance of e_{ij}

Level-2 model:

$$\beta_{qj}=r_{q0}+\Sigma r_{qs}w_{sj}+u_{qj}$$

where

r_{qs}: level-2 coefficients, q=0,1,…,Q;

w_{sj} : level-2 predictors for group j;

u_{qj}: level-2 random effect.

A typical application of this two-level model is that the level-1 model would represent the relationships at the student-level, or within school level and the level-2 model would capture the influence of school-level factors on the relationships obtained in level-1 model[1]. The applications of hierarchical linear models could be used to improve estimation of individual effects, model cross-level effects, and partition variance-covariance components[2].

To understand the statistical properties of hierarchical model, Monte Carlo Study is needed. With the simulated data, better understanding of the hierarchical model can be instrumental in conducting the empirical research. A SAS Macro has been developed to simulate normal/non-

normal data with hierarchical structure. Skewness and Kurtosis are used to simulate the non-normally distributed data[3] which can be incorporated in multilevel structure. For the predicators in each level, the correlation matrix could also be specified.

In this SAS Macro, although the simulated data is for the basic two-level models, it could be easily extended for the simulation of hierarchical data with 3 or more levels

CONCLUSION

This SAS Macro has the advantages of generating both normal and non-normal data with hierarchical structure. Statistician and researcher can benefit from it to develop and evaluate various statistical techniques to analyze the data with hierarchical structure.

REFERENCES

1. Anthony Bryk, Stephen Raudenbush, and R. Congdon, *Hierarchical Linear and Nonlinear Modeling with HLM/2L and HLM/3L Programs*. 2 ed. 0-89498-038-6, ed. Second. 1996: SSI. 170.
2. Anthony Bryk and S. Raudenbush, *Hierarchical Linear Models*. 1992.
3. Chou C-P, & Wu S-T, A SAS/IML program to simulate data with Non-Normal distribution, SAS Users Group International conference, Honolulu, April, 1992

APPENDIX

1: Main Macro to Get Simulation Data for 2-level Hierarchical Linear Model (normal or non-normal distribution)

```
/*    %getdata(Gam_m= Matrix of Gam {0.1 0.2
                                     0.5 0.3}
            Var_U_m= Matrix of U     {0.1 0.2}
             Var_e= Matrix of e      {.2}
               Lev_num= Number of students in each
          school                     {20@30}
          seed= seed for generating simulation data
                                     {01012000},
          mean_z= Mean Of Z          {0}
                  v_z= Variance Of Z {1}
                  r_z= Correlation Matrix of Z   {1}
             skew_z= Skewness Of Z   {missing}
             kurt_z= Kurtosis Of Z   {missing}
        mean_x= Mean Of X            {0}
          v_x= Variance Of X         {1}
          r_x= Correlation Matrix of X  {1}
             skew_x= Skewness Of X    {missing}
             kurt_x= Kurtosis Of X    {missing}
        outdata= Output Dataset      {t}
             outfile= Output File Name  {t.txt}
                   mod= Mod Flag     {No}
              debug= Debug Flag      {0}
             );                                  */

%macro getdata(Gam_m=0.1 0.2
                     0.5 0.3
                     0.3 0.4 ,
               Var_U_m=0.1 0.2 0.3,
               Var_e=.2,
               Lev_num=20@30,
                 seed=01012000,
mean_z=0, v_z=1, r_z=1, skew_z=,kurt_z=,
mean_x=0, v_x=1, r_x=1, skew_x=,kurt_x=,
          outdata=t,outfile=T.txt, mod=No,
          debug=0);

%if &debug=1 %then %do;
   option mprint mlogic; %end;
%if &debug^=1 %then %do;
   option nomprint nomlogic nonotes;   %end;

* _nx:  number of X;  * _nz:  number of Z;
%local _nx _nz;
%local Lev_num1 Lev_num2 _Gam_m;
%local num_obs;  %local num_T num_U num_e;

 * num of x, z;
data _null_ ;
  tt1=translate("&mean_x",'aaaaaaaaaa_','0123456789.');
  call symput( '_Cx',left(tt1) ) ;
  tt2=translate("&mean_z",'aaaaaaaaaa_','0123456789.');
  call symput( '_Cz',left(tt2) ) ;
run;

%local mean_1 mean_n_x mean_n_z;
 * to get parameters for system;
%macro read_s1;

data _null_ ;
  t1=translate("&Gam_m",'_',',');
  t2=translate("&Var_U_m",'_',',');
  t3=translate("&Var_e",'_',',');
  call symput('_CGam_m',left(t1) ) ;
  call symput('_CVar_U',left(t2) ) ;
  call symput('_CVar_e',left(t3) ) ;
  tt1=translate("&Gam_m",'aaaaaaaaaP','0123456789.');
 tt2=translate("&Var_U_m",'aaaaaaaaaP','0123456789.');
 tt3=translate("&Var_e",'aaaaaaaaaP','0123456789.');
  call symput('_CG',left(tt1) ) ;
```

```
  call symput('_Cu',left(tt2) ) ;
  call symput( '_Ce',left(tt3) );
run;

data _null_;
  array _a_ &_CG _b_ &_CU _c_ &_Ce;
  call symput('num_T',left(dim(_a_)));
  call symput('num_U',left(dim(_b_)) ;
  call symput('num_e',left(dim(_c_)));
run;

%do i=1 %to &num_T;   %local t&i;
%let t&i=%scan(&_CGam_m,&i,' '); %end;

%do i=1 %to &num_U;   %local U&i;
%let U&i=%scan(&_CVar_U,&i,' '); %end;

%do i=1 %to &num_e;   %local e&i;
%let e&i=%scan(&_CVar_e,&i,' '); %end;

data _null_;      tt=&num_u-1;
  call symput('_nx',tt);  tt=&num_T/&num_u-1;
  call symput('_nz',tt);
run;

%let mean_n_x=&_nx;    %let mean_n_z=&_nz;
%do i=1 %to &num_u;
  %local Gam_&i; %let Gam_&i= ;
  %do j=1 %to %eval(&_nz+1);
    %local tttt; %let tttt=%eval((&i-1)*(&_nz+1)+&j);
    %let Gam_&i=&&Gam_&i &&T&tttt;       %end;

  %if &i = 1 %then %do; %let Gam__= &&Gam_&i;%end;
  %else %if &i ^= 1  %then %do;
    %let Gam__=&Gam__ , &&Gam_&i ; %end;
  %end;

  %local clev_num;
  data _null_ ;
    t1=translate("&Lev_num",'_________','123456789');
    call symput( 'clev_num',left(t1) ) ;
  run;

  *within schools;
  %if %index(&Lev_num,@)>0  %then %do;
  %let Lev_num1=%scan(&Lev_num,1,'@') ;
  %let Lev_num2= %scan(&Lev_num,2,'@') ;

  %do i=1 %to &Lev_num1;
    %global num_s&i;
    %let num_s&i=&Lev_num2;
  %end;
%end;
%else %if %index(&Lev_num,@)=0 %then
%do;
  data _null_;
     array _a_ &clev_num;  call
     symput('Lev_num1',left(dim(_a_)));
  run;

  %do i=1 %to &Lev_num1 ;
   %global num_s&i;
   %let num_s&i=%scan(&Lev_num, &i);
  %end;
%end;

data _null_ ;
  t1=translate("&Gam__",':',' ');
  call symput('_Gam_m',left(t1) ) ;
run;

%mend read_s1;

* corr ;
%local _r_x _r_z;

%macro read_z;
  %let r=&r_z;
  * charactized r;
  %if &r=  %then  %let r=1;
  data _null_ ;
  t1=translate("&r",'_',',');
  call symput('_C_r',left(t1) ) ;    run;

  * to get mean;    %local _num1 _num2  ;
  %if &mean_z=  %let mean_z=0;
  * default value of mean;
  %if %index(&mean_z,@)=1  %then %do;
   *n@m format;       %let _num1 =
   %scan(&mean_z,1,'@') ; %let _num2 =
   %scan(&mean_z,2,'@') ;

  %do i=1 %to &_num1;
    %local m_&i; %let m_&i=&_num2;%end;
    %end;
  %else %if %index(&mean_z,@)=0 %then
  %do;     * n1 n2 ...format;
    %do i=1 %to &mean_n_z ;
      %let m_&i=%scan(&mean_z, &i); %end; %end;

  %let _num_m=&mean_n_z;
  %if &v_z=  %then %do;
   * default value of v;
   %do i=1 %to &_num_m ;
     %let v_z=&v_z 1;  %end;  %end;

*to get the element from r;
%local _ttt; %let _ttt=%eval(&_num_m*&_num_m);
%do i=1 %to &_ttt;        %local r__&i;
  %let r__&i=%scan(&_C_r,&i,' ');     %end;

%do i=0 %to (&_num_m-1); %local r_&i;%let r_&i=;

%do j=1 %to &_num_m;
  %local tttt;
  %let tttt=%eval(&i*&_num_m+&j);
  %let r_&i=&&r_&i &&r__&tttt; %end;

%if &i = 0 %then %do;  %let _r_z= &&r_&i; %end;
%else %if &i ^= 0  %then %do;
%let _r_z=&_r_z , &&r_&i ;  %end;    %end;

 data _null_ ;
 t1=translate("&_r_z",',',' ');
 call symput('_r_z',left(t1) ) ;    run;

%mend read_z;

%macro read_x;

 %let r=&r_x;

 * charactized r;
 %if &r=  %then %do; %let r=1;    %end;
 data _null_ ;
  t1=translate("&r",'_',',');
  call symput('_C_r',left(t1) ) ;    run;

 * to get mean;    %local _num1 _num2  ;
 %if &mean_x=  %then %do;  %do j=1 %to &_nx;
```

```
 %let mean_x=&mean_x 0;  %end;    %end;

* default value of mean;
%if %index(&mean_x,@)=1  %then %do;
 *  n@m format;
 %let _num1=%scan(&mean_x,1,'@');
 %let _num2=%scan(&mean_x,2,'@');
 %do i=1 %to &_num1;  %local m_&i;
 %let m_&i=&_num2;  %end;    %end;
 %else %if %index(&mean_x,@)=0 %then %do;
 * n1 n2 ... format;
 %do i=1 %to &mean_n_x ;
 %let m_&i=%scan(&mean_x, &i);    %end;  %end;

 %let _num_m=&mean_n_x;  %if &v_x= %then %do;
 * default value of v;
 %do i=1 %to &_num_m;%let v_x=&v_x 1;%end;%end;

 %if &v_x=1 and &_num_m>1 %then %do;
 * default value of v;
 %do i=2 %to &_num_m;%let v_x=&v_x 1;%end;%end;

 %if &_C_r or (&_C_r=1 and &_num_m>1) %then %do;
 * default value of r;%local _xx;
 %let _C_r=;       %do i=1 %to &_num_m ;
 %do j=1 %to &_num_m ;
  %if &i=&j %then %let _xx=1;
  %else  %let _xx=0;
  %let _C_r=&_C_r &_xx; %end; %end; %end;

 *to get the element from r;     %local _ttt;
 %let _ttt=%eval(&_num_m*&_num_m);
 %do i=1 %to &_ttt;        %local r__&i;
 %let r__&i=%scan(&_C_r,&i,' ');     %end;

 %do i=0 %to (&_num_m-1);%local r_&i;%let r_&i=;

 %do j=1 %to &_num_m; %local tttt;
   %let tttt=%eval(&i*&_num_m+&j);
   %let r_&i=&&r_&i &&r__&tttt;  %end;

 %if &i = 0 %then %do;   %let _r_x= &&r_&i;%end;
 %else %if &i ^= 0  %then %do;
 %let _r_x=&_r_x , &&r_&i ;  %end;  %end;

 data _null_ ;
  t1=translate("&_r_x",'.','_');
  call symput('_r_x',left(t1) ) ;     run;

%mend read_x;

%macro runmain;

 proc iml;

   * to input data;
   START in_data(TGam,U,e,x_num,z_num);
   * u;  tu={&Var_U_m};  U=tu`;
   *x_num:  number of X   z_num:  number of Z;
   x_num=ncol(tu)-1; z_num=ncol(TGam)/ncol(tu)-1;
   * e;        e_={&Var_e};        e=e_`;
   * TGam;        TGam={&_Gam_m};
   finish;

   * skew & kurt are not in;
   run in_data(TGam,U,e,x_num,z_num);

   * between school level;
   %let r=&_r_z;
   %local seedj; %let seedj=&seed;

   %if &skew_x^= and &skew_z^= %then  %do;
     %local flag_x;  %let flag_x=1; %end;
   %if &skew_x^= and &skew_z= %then %do;
     %local flag_x; %let flag_x=0;          %end;

%if &skew_z= and &kurt_z= %then %do;
  %simula(seed1=&seedj,num_obs=&Lev_num1,
  mean=&mean_z,  mean_num=&mean_n_z,
  v=&v_z,outmat=zeta,flag=0);        %end;
 %else %if &skew_z^= and &kurt_z^= %then %do;
  _t1=translate("&r",'@',',');
  call symput("R1",_t1);
  %if &debug=1 %then
  %put &seedj &Lev_num1 &mean_z &skew_z &kurt_z
     &r1;
  %simnnor( seed1=&seedj,
     num_obs=&Lev_num1,
     mean=&mean_z,skew=&skew_z,kurt=&kurt_z,r=&r1,
     outmat=zeta, debug=&debug, flag=0);%end;

  %local iu ju; %let r=;
  %do iu=1 %to &num_U;
   %do ju=1 %to &num_U;
     %if &iu^=&ju %then %do; %let r=&r 0 ;%end;
     %else  %if  &iu=&ju  %then  %do;  %let  r=&r  1;
          %end;%end;
  %if &iu^=&num_U %then %do;%let r=&r ,; %end;%end;

  * adjusted seed; %let seedj=%eval(&seedj+1);
   %simula(seed1=&seedj,
   num_obs=&Lev_num1,mean=0,mean_num=&num_U,
   v=&Var_U_m,outmat=U_e,flag=0);

    _t1=TGam[,1];
    %if &_nz^=1 %then %do;_t2=TGam[,2:%eval(&_nz+1)];
    %end;
    %if &_nz=1 %then %do;    _t2=TGam[,2];%end;
    _tt1=_t1`; gama0=repeat(_tt1,&Lev_num1,1);
    beta=gama0+zeta*_t2`+U_e;

    * within school;
   %do iiii=1 %to &Lev_num1;
   %let num_obs=&&num_s&iiii; %let r=&_r_x;

   * adjusted seed; %let seedj=%eval(&seedj+1);
   %if &skew_x= and &kurt_x= %then %do;
    %simula(seed1=&seedj,num_obs=&num_obs,
     mean=&mean_x,mean_num=&mean_n_x,
     v=&v_x,outmat=x,flag=0);          %end;

   %else %if &skew_x^= and &kurt_x^= %then %do;
     _t1=translate("&r",'@',',');
     call symput("R1",_t1);  %put r1=&r1;
     %simnnor(seed1=&seedj,
            num_obs=&num_obs,
            mean=&mean_x,skew=&skew_x,
            kurt=&kurt_x,r=&r1,outmat=x, debug=&debug,
             flag=&flag_x); %end;

* to get y corresponding to x and beta;
_ttt=beta[&iiii,];          _t1=_ttt[,1];
_t1=repeat(_t1,&num_obs,1);
%if &_nx^=1 %then %do;
     _t2=_ttt[,2:%eval(&_nx+1)];    %end;
   %if &_nx=1 %then %do;     _t2=_ttt[,2]; %end;

 * to get e;  %let r=1;
* adjusted seed; %let seedj=%eval(&seedj+1);
%simula(seed1=&seedj,num_obs=&num_obs,
     mean=0,mean_num=1,
```

```
    v=&Var_e,outmat=e_e,flag=0);

* to get y ; _t3=x;
   %if &debug=1 %then %do;
    print zeta; print _t1; print _t2;
    print x ;print e_e; %end;
    y=_t1+x*_t2`+e_e;
    * to get zeta for each school;
    _tz=zeta[&iiii,]; zeta1=repeat(_tz,&num_obs,1);

* final MATRIX;
%IF &iiii=1 %THEN %DO; CREATE _z FROM zeta1;
 * z; APPEND FROM zeta1; CREATE _x FROM x;
 * x; APPEND FROM x; CREATE _y FROM y;
 * y; APPEND FROM y; %END;
 %ELSE %DO; SETOUT _z;
 APPEND FROM zeta1; SETOUT _x;
 APPEND FROM x;   SETOUT _y;
 APPEND FROM y;  %END;  %end;

quit;

data t;
  %do i=1 %to &Lev_num1;  %do j=1 %to &&num_s&i;
   sch_ID=&i; stu_ID=&j; output t; %end;  %end;
run;

data t; set t;i=_n_;   run;

data _x; set _x; i=_n_; run;
data _x(drop=i); merge t _x; by i; run;

data _y; set _y; rename col1=y; run;
data _x; set _x; %do i=1 %to &mean_n_x;
   rename col&i=x&i;    %end;    run;

data _z; set _z; %do i=1 %to &mean_n_z;
   rename col&i=z&i;    %end;    run;

data &outdata;  merge _x _y _z; run;

%if &outfile^=  %then %do;
   %if %upcase(&mod)=NO %then %let mod=;
data _null_;
   file "&outfile" &mod;
   put '_____';
   put 'Model Specified';       put "Gam=&Gam_m";
   put "U=&Var_U_m";    put "e=&Var_e";
   put "num: &Lev_num";  put "seed=&seed";
   put          "mean_x=&mean_x          skew_x=&skew_x
          kurt_x=&kurt_x
   v_x=&v_x r_x=&r_x"; put "mean_z=&mean_z
   skew_z=&skew_z kurt_z=&kurt_z v_z=&v_z r_z=&r_z";
run;
option ls=78 ps=500;

proc anova data=&outdata noprint outstat=_t;
  class sch_id;  model y=sch_id;run;
data _t1(keep=MSWG df1) _t2(keep=MSBG F df2);
 set _t;       if _source_='ERROR' then do ;
   MSWG=ss/df;df1=df;output _t1;       end;
   if _source_='SCH_ID' then do;
   MSBG=ss/df; df2=df;  output _t2;       end;
run;
data _t(keep=mswg msbg icc); merge _t1 _t2;
  ICC=(f-1)/(f+(df1+df2+1)/(df2+1)-1);  run;
data _null_;  file "&outfile" mod; set _t;
   format ICC mswg msbg 10.4 ;
   put '_____';
   put 'ICC from generated simulation data';
   put ICC=;put MSWG= MSBG=;      run;

proc printto print="&outfile"; run;
proc sort data=&outdata; by sch_ID;    run;

 %local zzz xxx;      %do i=1 %to &mean_n_z;
 %let zzz=&zzz z&i;      %end;
 %do i=1 %to &mean_n_x;  %let xxx=&xxx x&i; %end;

 data __t1; set &outdata; by sch_id;
 if first.sch_id; run;
 title '==== all subjects====';
 proc means data=__t1;  var &zzz;   run;
 proc means data=&outdata;
  var sch_id stu_id y &xxx ; run;
 PROC MEANS MEAN STD SKEWNESS
  KURTOSIS DATA=__t1(keep=&zzz);
  OUTPUT MEAN=mean_z
  STD=std_z SKEWNESS=skew_z KURTOSIS=kurt_z;
 run;

 PROC MEANS MEAN STD SKEWNESS KURTOSIS
  DATA=&outdata(keep=&xxx);OUTPUT
          MEAN=mean_x
  STD=std_x SKEWNESS=skew_x KURTOSIS=kurt_x;
 run;

 title " ; proc printto ;run;    %end;

 proc datasets library=work memtype=data nolist;
 delete _x _y _z;   quit ;
%mend runmain;

%read_s1;  %read_z;  %read_x;
* to get simulation data;  %runmain;

%mend getdata;
```

2: Example of Using This Macro

```
%getdata(Gam_m =0.1 0.1
          0.5 0.2,
    Var_U_m=0.1 0.2,
     Var_e=.2,
    Lev_num=20@30,
     seed_x=01012000,mean_x=0, v_x=1,
       r_x=1,
     seed_z=01012000,mean_z=0, v_z=1,
       r_z=1,
    outdata=_t_,
    outfile=c:\remwork\wei\remw002.lst,mod=);
```

3: Example of Output

```
_____
Model Specified
Gam=0.1 0.1              0.5 0.2
U=0.1 0.2
e=.2
num: 20@30
seed=01012000
mean_x=0 skew_x= kurt_x= v_x=1 r_x=1
mean_z=0 skew_z= kurt_z= v_z=1 r_z=1
_____
ICC from generated simulation data
```

ICC=0.2686
MSWG=0.4988 MSBG=5.9942

Analysis Variable : Z1

N	Mean	Std Dev	Minimum	Maximum
20	-0.2988	1.1151	-2.0294	1.3454

Variable	N	Mean	Std Dev	Minimum	Maximum
SCH_ID	600	10.500	5.77	1	20
STU_ID	600	15.500	8.663	1	30
Y	600	0.036	0.820	-2.378	2.594
X1	600	-0.030	0.983	-2.630	3.192

Notes:

1: Macro simula and simnnor used in Getdata are not put in appendix. Simula is applied to create simulation data with correlation and simnnor is applied to create simulation data with non normal distribution.

Please contact Wei Sun at wsun@hsc.usc.edu for this full macro.

2: Output format modified for the paper.