SESUG 2022 Paper 186

A SAS Macro That Automates Model Fitting of Group-Based Trajectory Modeling Using Proc TRAJ

Donald E. Warden, MPH and Yu Jiang, PhD, Division of Epidemiology and Biostatistics, School of Public Health, The University of Memphis

ABSTRACT

The third-party SAS procedure TRAJ developed by Jones et.al gives users a convenient tool for group-based trajectory modeling. The procedure allows the user to fit any number of classes across five polynomial orders that each class follows. To fit these models, users must assess model fit by comparing the Bayesian Information Criteria (BIC) of the models with different polynomial orders. Moreover, the parameter estimate for each class polynomial must be statistically significant – generally presumed as a T test alpha less than 0.05. With each increase in the number of classes, the number of models to check increases exponentially. Traditional methodology employs bootstrapping to generate valid, parsimonious models without checking every model. However, this methodology likely does not find the best fitting model.

In this paper, we develop a SAS macro, autoTRAJ, which assesses all possible polynomial orders and outputs a list with the best fitting model based on BIC up to eight classes. Further, the macro checks for common statistical abnormalities such as false or singular convergences. The macro utilizes latent output files to determine the statistical significance of the polynomial T statistics and only shows users statistically significant models. By automating the traditional methodology, the programmer will no longer exclude better fitting models due to time constraints. This allows for more precise trajectory modeling with no uncertainty that the selected model fits best. The intended audience for this presentation is intermediate SAS users with knowledge of Base and IML SAS and the TRAJ procedure.

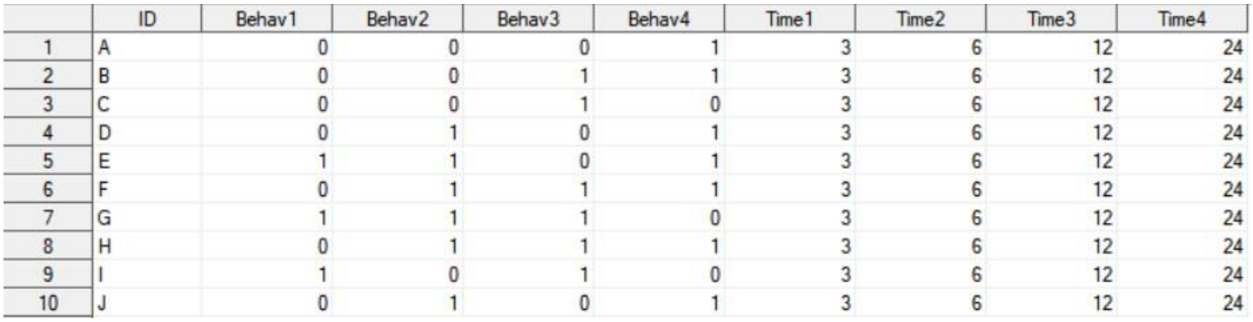
INTRODUCTION

Group-based trajectory modeling (GBTM) allows for the analysis of developmental trajectories that can account for age and time-based data (Nagin 2014). This field of growth mixture modeling was introduced in 1993 by Nagin and Land, initially to track behavioral patterns among incarcerated individuals; however, mathematically, these models are applicable to developmental trajectories for any condition with multivariate longitudinal data (Nagin and Odgers 2010; Nagin et al. 2018). The third-party SAS procedure, Proc TRAJ, introduces this analysis into the SAS environment. The procedure has been updated since its inception (Jones and Nagin 2007) and can be downloaded at: https://www.andrew.cmu.edu/user/bjones/

However, this procedure requires users to manipulate polynomial orders to fit models individually, which are subsequently compared to other models of interest. As the number of desired trajectories increases, it becomes cumbersome to analyze and compare these models to determine the best fitting model. This macro automates the analytic and comparative processes leaving the best fitting model as the result.

UNDERSTANDING THE MACRO

The minimum data requirements for Proc TRAJ and thus this macro is an observational identifier, dependent variables measured across timepoints, and independent variables indicating the time at which the dependent variables were collected. The macro, attached as an appendix is appropriately flagged for reference within this paper. This is a main topic in the body of the paper.



	ID	Behav1	Behav2	Behav3	Behav4	Time1	Time2	Time3	Time4
1	A	0	0	0	1	3	6	12	24
2	B	0	0	1	1	3	6	12	24
3	C	0	0	1	0	3	6	12	24
4	D	0	1	0	1	3	6	12	24
5	E	1	1	0	1	3	6	12	24
6	F	0	1	1	1	3	6	12	24
7	G	1	1	1	0	3	6	12	24
8	H	0	1	1	1	3	6	12	24
9	I	1	0	1	0	3	6	12	24
10	J	0	1	0	1	3	6	12	24

Figure 1. A Sample Dataset

DETAILED MACRO WALKTHROUGH

1. This IML code generates all possible polynomial orders using cartesian products via the EXPANDGRID function. This dataset is this used as a reference for enumerating models and using the corresponding polynomial order. Initially, this produces all orders to 8 group models.

```
/********************************/

/*                                              */
/*           autoTRAJ Start               */

/*                                              */
/********************************/
%macro autoTRAJ(c_start,c_num);
/********************************/
/*                                              */
/*           Creating Perms               */

/*                                              */
/********************************/
proc iml; set =
T( 0:5 );
/* generate all possible triplets (x1,x2,x3) where the x_i are in
{0,1,2,3,4,5} */ p1
= set;
p2 = expandgrid(set, set); p3 = expandgrid(set, set,
set); p4 = expandgrid(set, set, set, set); p5 =
expandgrid(set, set, set, set, set); p6 =
expandgrid(set, set, set, set, set, set); p7 =
expandgrid(set, set, set, set, set, set, set); p8 =
expandgrid(set, set, set, set, set, set, set, set);
/* write to data set */
create Values var {Length, x1, x2, x3, x4, x5, x6, x7, x8};

N = nrow(p1);Length=j(N,1,1); x1 =
p1;  x2=j(N,1,.); x3=j(N,1,.);
append;
N = nrow(p2); Length=j(N,1,2); x1 =
p2[,1];  x2=p2[,2]; x3=j(N,1,.);
append;
```

```
N = nrow(p3); Length=j(N,1,3); x1 =
p3[,1];  x2=p3[,2]; x3=p3[,3];
append;
N = nrow(p4); Length=j(N,1,4);
x1 = p4[,1];  x2=p4[,2]; x3=p4[,3]; x4=p4[,4]; append;
N = nrow(p5); Length=j(N,1,5);
x1 = p5[,1];  x2=p5[,2]; x3=p5[,3]; x4=p5[,4] ;x5=p5[,5]; append;
N = nrow(p6); Length=j(N,1,6);
x1 = p6[,1];  x2=p6[,2]; x3=p6[,3]; x4=p6[,4] ;x5=p6[,5]; x6 = p6[,6];
append;
N = nrow(p7); Length=j(N,1,7);
x1 = p7[,1];  x2=p7[,2]; x3=p7[,3]; x4=p7[,4] ;x5=p7[,5]; x6 = p7[,6];
x7 = p7[,7]; append;
N = nrow(p8); Length=j(N,1,8);
x1 = p8[,1];  x2=p8[,2]; x3=p8[,3]; x4=p8[,4] ;x5=p8[,5]; x6 = p8[,6];
x7 = p8[,7]; x8 = p8[,8]; append; close; quit;  data Perms;
      set values;
c_num = LENGTH;
p_num = _N_;
            perm = catx(' ', of x: );
            perm = trim(compress(perm,".", ));
            INDEX + 1;
by LENGTH;
            if first.LENGTH then INDEX = 1;
            keep C_NUM P_NUM INDEX
perm; run;   dm log "clear"; dm output
"clear";
%do c = &c_start %to &c_num %by 1;
FILENAME el
"&autoTRAJdir.\error_log.txt";
```

2. If additional modifications are needed for analysis such as the inclusion of risk factors or other features of Proc TRAJ, they can be added here. This section calls the TRAJ procedure by pulling in the appropriate variables from user inputs and the polynomial order data.

```
/*********************************/

/*                                              */
/*          Trajectories                  */

/*                                              */
/*********************************/
data Perms_&c; /*creates permutation dataset where only perms of class
number c exist*/  set Perms;              where c_num = &c;
run;
data _null_;
      %let numo = 0;    set
Perms_&c end = eof;
call
symputx(cats('Perm',_n_),Perm
);
            if eof then call symputx('numo',_n_);  run;
```

```
%do p = 1 %to &numo; dm
log "clear";
PROC PRINTTO LOG = el NEW;
RUN; dm log "clear";
Proc Traj data = &dataset
                  outplot =
OP_&c._&p               outstat =
OS_&c._&p               out =
OF_&c._&p               outest =
OE_&c._&p;        id &idvar;
var &varlist;           indep
&indeplist;             model
&model;           ngroups &c;
            order &&Perm&p;
run;  PROC
PRINTTO;
 RUN;  dm log
  "clear";
```

3. The macro then checks for false and singular convergences. Models with these errors are flagged.

```
/********************************/

/*                                              */
/*          Error Checks                  */

/*                                              */
/********************************/
*Adapted from Blaze, T.J 2014.  http://d-
scholarship.pitt.edu/21158/1/blazetj_etd_2014.pdf;
*check for errors and/or warnings;
 DATA e_log_&c._&p;
 INFILE el dlm='10'X dsd;
 LENGTH message $256;
 INPUT message;
 IF FIND(message, "ERROR: Singular convergence")~=0 THEN issue
= 1;
 ELSE IF FIND(message,"WARNING: Unable to calculate standard errors")~=0
THEN issue = 2;
 ELSE IF FIND (message,"ERROR: Floating Point Overflow")~=0 THEN
issue = 3;
 ELSE IF FIND (message,"ERROR: Floating Point Zero
Divide")~=0 THEN issue = 4;
ELSE IF FIND (message,"ERROR: False convergence")~=0 THEN issue = 5;
ELSE issue = 0;
 IF (issue = 0 & (FIND(message,
"ERROR")~=0|FIND(message,"WARNING")~=0)) THEN issue = 6;
 DROP message;
 RUN;
 PROC SORT DATA=e_log_&c._&p nodupkey;
 BY issue;
 RUN;
 DATA e_log_&c._&p;
SET e_log_&c._&p;  flag
= 1;
```

```
 RUN;
 PROC TRANSPOSE DATA=e_log_&c._&p OUT=e_log_&c._&p PREFIX=issue;
 ID issue;
 VAR flag;
 RUN;  DATA
_null_;
 dset=open("e_log_&c._&p");
 CALL SYMPUT ('chk1',varnum(dset,'issue1'));
 CALL SYMPUT ('chk2',varnum(dset,'issue2'));
 CALL SYMPUT ('chk3',varnum(dset,'issue3'));
 CALL SYMPUT ('chk4',varnum(dset,'issue4'));
 CALL SYMPUT ('chk5',varnum(dset,'issue5'));
CALL SYMPUT ('chk6',varnum(dset,'issue6'));
 RUN;

 DATA e_log_&c._&p;
 RETAIN issue1 issue2 issue3 issue4 issue5 issue6;
 SET e_log_&c._&p;
 IF &chk1 = 0 THEN issue1 = 0;
 IF &chk2 = 0 THEN issue2 = 0;
 IF &chk3 = 0 THEN issue3 = 0;
 IF &chk4 = 0 THEN issue4 = 0;
 IF &chk5 = 0 THEN issue5 = 0;
IF &chk6 = 0 THEN issue6 = 0;
 DROP _NAME_ issue0;
RUN; dm log "clear";
```

4. Based on the polynomial order and the number of desired trajectories, the appropriate variables are pulled from the output datasets for further analysis and to determine model fit.

```
/********************************/

/*                                              */
/*           Perm datasets                */

/*                                              */
/********************************/

data Perms_&C._&p._classes; *Creates dataset describing c number of
classes;
      set Perms_&C;           if
perm = "&&PERM&p";
%do r = 1 %to &c;
                  %let uno = 1;
                  %let rnum = %eval((2*&r)-1);
            Class&r = substr(Perm,&rnum, &uno);
                        call symputx(cats("Class&r"),Class&r);
            %end;
run;

/********************************/
/*                                              */
/*           SE and Betas                 */
/*                                              */
```

```
/********************************/

Data Check&c;
      set Perms_&C._&p._classes;
%do r = 1 %to &c;
      *Adds a string for the class r Beta and SE variable in the
OE output;  if &&class&r = 0 then stringClass&r = "INTERC0&r";
if &&class&r = 1 then stringClass&r = "LINEAR0&r";    if &&class&r
= 2 then stringClass&r = "QUADRA0&r";     if &&class&r = 3 then
stringClass&r = "CUBIC0&r";   if &&class&r = 4 then stringClass&r
= "QUARTI0&r";    if &&class&r = 5 then stringClass&r =
"QUINTI0&r";      call symputx ("stringClass&r",stringClass&r);
%end; run;
%do r = 1 %to &c;
*This selects the correct string for Beta; proc
sql noprint;
      select &&stringclass&r into :classbeta&r from OE_&c._&p
      where _TYPE_ = "PARMS";
*This selects the correct string for SE; proc
sql noprint;
      select &&stringclass&r into :classSE&r from OE_&c._&p
      where _TYPE_ = "STDERR";
quit;
      %end;
```

5. T Scores are calculated from the output files and the p-values are calculated to assess significance.

```
/********************************/
/*                                             */
/*           Calc T Score                 */
/*                                             */
/********************************/

Data OE_&c._&p;
      set OE_&c._&p;
%do r = 1 %to &c;
            CSIG&r = &&classbeta&r / &&classSE&r;           if
CSIG&r ge 1.96 or CSIG&r le -1.96 then SIGC_&r = 1;
            else SIGC_&r = 0;
      %end;
            CSIG = sum(of SIGC_:);
if CSIG ne &c then MSIG_&c = 0;
                  if CSIG = &c then MSIG_&c =
            1; run;
```

6. Models are then sorted by BIC and the top 10 significant models are output for each number of trajectories. Models are not culled by errors, but the errors are noted for further investigation.

```
/********************************/
/*                                             */
```

```
  /* Determine Significance          */
/*                                               */
/********************************/

data OE_&c._&p;
      set OE_&c._&p;
      length model_id $30;          if
_type_ = 'PARMS';             model_id =
cats(&c,"Class",&p);          p_num = &p;
      c_num = &c;
run;
proc sql;
      create table OE_&c._&p as
select a.*, b.* from
      OE_&c._&p as a, e_log_&c._&p as
b; quit;  %end; *ends p;
 data
Combined_&c;
      set
      %do i = 1 %to &numo;
            OE_&c._&i
      %end;;
run;
proc sql;
      create table combined2_&c as        select
a.*, b.perm from        combined_&c as a, perms as
b     where a.p_num = b.index and a.c_num =
b.c_num; quit;   proc sort data = combined2_&c;
      by descending _BIC1_; run;
data ft_combined_&c;    set
combined2_&c (obs=10);
      where MSIG_&C = 1; *Only keeps significant
ones; run;  proc print data = ft_combined_&c noobs
label;      var model_id _BIC1_ MSIG_&c perm issue1
issue5;
label       issue1 = "Singular Convergence"
issue5 = "False Convergence"
            _BIC1_ = "Bayesian Information Criterion"
            perm = "Polynomial Order"
            model_id = "Model ID";
title "Model Fit Data for &c Classes";
footnote j = l "autoTRAJ Macro created by Donald E. Warden, MPH and Yu
(Joyce) Jiang, PhD";
run;  data Export;
      set
      %do i = 1 %to &c;
ft_combined_&c
      %end; ;
run;   proc
export
      data = Export
      outfile = "&autoTRAJdir.\autoTraj_output.csv"
dbms = csv
      replace;
```

```
run;
%end; *ends c;
            %mend autoTRAJ;
```

7. There are six macrovariables that must be specified prior to running the macro. A model error log and a full list of the output in csv format are output to the specified file path.

```
/********************************/

/*                                              */
/*          Specify Macrovars       */
/*                                              */
/********************************/

/*Dependent Variables*/
%let varlist = Behav1-Behav4;
/*Independent Variables*/
%let indeplist = Time1-Time4;
/*Dataset of Interest*/
%let dataset = dataset;
/*Model Type: LOGIT, CNORM, or ZIP. See Proc TRAJ documentation for
more information*/ %let model = LOGIT;
/*Observation Identifier*/
%let idvar = ID;
/*Error log filepath*/
      %let autoTRAJdir = Filepath;
```

MACRO OUTPUT

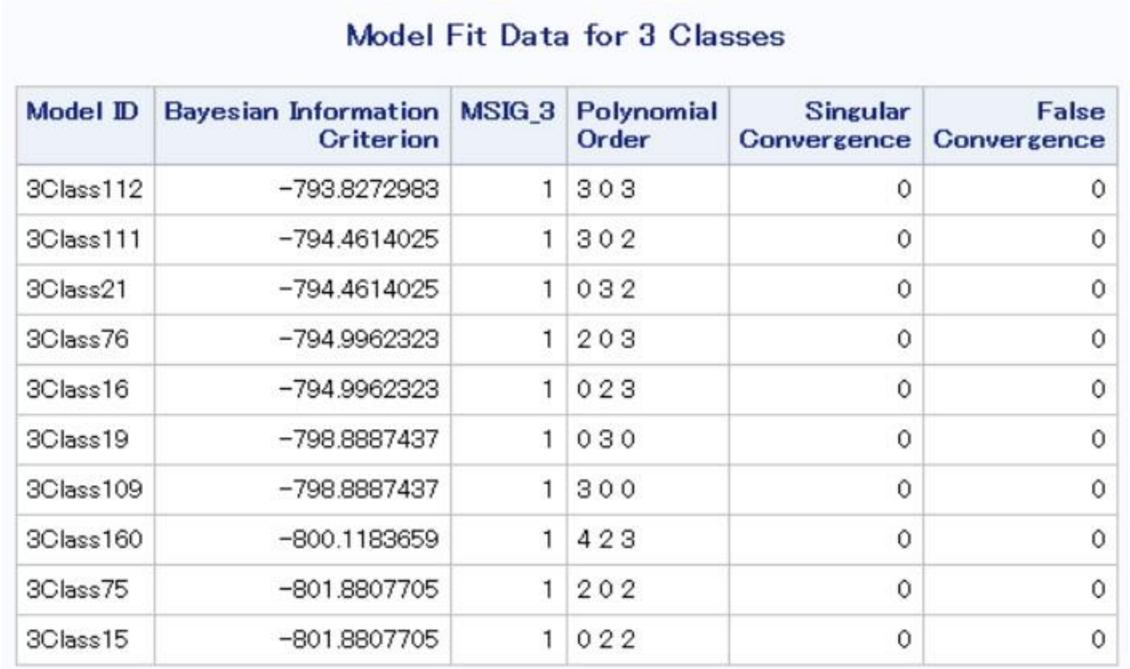
The macro is called with %AutoTRAJ(starting trajectories, ending trajectories). This line will test only models with three trajectories.

```
%AutoTraj(3,3);
```

This line will test all models with one through five trajectories.

```
%AutoTraj(1,5);
```

It is recommended that the users start with running %AutoTraj(1,1); as a test to validate the macro works with the dataset – this tests the six polynomial orders for models with only one trajectory. Running the macro on a dataset with 4 dependent variables, 4 independent variables, and 22 observations, the macro assessed every possible model (155 total) up to three trajectories in 7 minutes 58 seconds using a computer with 16GB of RAM and an 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz.The sample dataset from https://www.andrew.cmu.edu/user/bjones/logit.htm contains 23 variables and 195 subjects – similarly, it ran the models in 11 minutes 39 seconds. While the example suggests using a polynomial order of “3 3” or cubic polynomials on two trajectories, the AutoTRAJ macro has identified a three-trajectory model with a better fit.



Model Fit Data for 3 Classes

Model ID	Bayesian Information Criterion	MSIG_3	Polynomial Order	Singular Convergence	False Convergence
3Class112	-793.8272983	1	3 0 3	0	0
3Class111	-794.4614025	1	3 0 2	0	0
3Class21	-794.4614025	1	0 3 2	0	0
3Class76	-794.9962323	1	2 0 3	0	0
3Class16	-794.9962323	1	0 2 3	0	0
3Class19	-798.8887437	1	0 3 0	0	0
3Class109	-798.8887437	1	3 0 0	0	0
3Class160	-800.1183659	1	4 2 3	0	0
3Class75	-801.8807705	1	2 0 2	0	0
3Class15	-801.8807705	1	0 2 2	0	0

Figure 2. Three Trajectory Model Fit Output from the AutoTRAJ macro

CONCLUSION

Group-based trajectory modeling represents an important subfield of growth mixture modeling. This macro assists in expediting model fitting and ensures accurate models even at high numbers of trajectories. Further, the current model fitting paradigm requires active participation from the end user; whereas this macro allows the user to "run and forget" thereby freeing time for other endeavors.

REFERENCES

Jones, B. L., and Nagin, D. S. (2007), "Advances in Group-Based Trajectory Modeling and an SAS procedure for estimating them," Sociological methods & research, 35, 542–571. https://doi.org/10.1177/0049124106292364.

Nagin, D. S., Jones, B. L., Passos, V. L., and Tremblay, R. E. (2018), "Group-based multitrajectory modeling.," Statistical Methods in Medical Research, 27, 2015–2023. https://doi.org/10.1177/0962280216673085.

Nagin, D. S., and Odgers, C. L. (2010), "Group-based trajectory modeling in clinical research.," Annual Review of Clinical Psychology, 6, 109–138. https://doi.org/10.1146/annurev.clinpsy.121208.131413.

Nagin, D. S. (2014), "Group-based trajectory modeling: an overview.," Annals of Nutrition & Metabolism, 65, 205–210. https://doi.org/10.1159/000360229.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Donald E. Warden

dewarden@memphis.edu