

A Journey to Latent Class Analysis (LCA)

Jeff Pitblado

StataCorp LLC

2017 Nordic and Baltic Stata Users Group Meeting
Stockholm, Sweden

Outline

Motivation

by: prefix

if clause

suest command

Factor variables

sem command

gsem command

fmm: prefix

Latent class models

Motivation

Observed groups

What can you do with a variable that identifies groups in your data?

Latent groups (classes)

What can you do when the groups are not deterministically identified by variables in your data?

Example dataset

Observed variables

- ▶ **y** is the dependent variable of interest.
Suppose it is a count outcome.
We want to use the Poisson model.
- ▶ **x1** and **x2** are continuous independent variables.
We are interested in how they are associated with **y**.
- ▶ **grp** identifies group membership.
We have observed two groups, say 1 and 2.

by: prefix

Description

- ▶ Repeat model fit on subsets of the data.

Features

- ▶ Syntax is easy to learn and use.

Limitations

- ▶ Testing parameters between groups is not easy.
- ▶ Constraints on parameters between groups is not possible.

by: example

```
. use data  
(Simulated data--A Journey to Latent Class Analysis)  
. sort grp  
. by grp: poisson y x1 x2, nolog
```

```
-> grp = 1
```

```
Poisson regression                Number of obs   =          122  
                                LR chi2(2)       =          131.20  
                                Prob > chi2        =           0.0000  
Log likelihood = -212.8328        Pseudo R2      =           0.2356
```

y	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
x1	.0962749	.0127086	7.58	0.000	.0713666	.1211832
x2	-.1814847	.0206201	-8.80	0.000	-.2218993	-.1410701
_cons	2.956803	.2116169	13.97	0.000	2.542041	3.371564

```
-> grp = 2
```

```
Poisson regression                Number of obs   =          178  
                                LR chi2(2)       =          619.25  
                                Prob > chi2        =           0.0000  
Log likelihood = -410.976        Pseudo R2      =           0.4297
```

y	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
x1	-.0974296	.0062866	-15.50	0.000	-.1097511	-.0851081
x2	-.1929588	.0099521	-19.39	0.000	-.2124646	-.173453
_cons	4.968026	.095185	52.19	0.000	4.781467	5.154585

if clause

Description

- ▶ Fit model to each group separately.

Features

- ▶ Syntax is easy to learn and use.
- ▶ Group-specific outcome models.
- ▶ Use **estimates table** to report fitted parameters side-by-side.

Limitations

- ▶ Testing parameters between groups is not easy.
- ▶ Constraints on parameters between groups is not possible.

if example

```
. poisson y x1 x2 if grp==1, nolog
```

```
Poisson regression
```

```
Number of obs      =      122  
LR chi2(2)         =     131.20  
Prob > chi2        =      0.0000  
Pseudo R2         =      0.2356
```

```
Log likelihood = -212.8328
```

y	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
x1	.0962749	.0127086	7.58	0.000	.0713666	.1211832
x2	-.1814847	.0206201	-8.80	0.000	-.2218993	-.1410701
_cons	2.956803	.2116169	13.97	0.000	2.542041	3.371564

```
. estimates store g1
```

```
. poisson y x1 x2 if grp==2, nolog
```

```
Poisson regression
```

```
Number of obs      =      178  
LR chi2(2)         =     619.25  
Prob > chi2        =      0.0000  
Pseudo R2         =      0.4297
```

```
Log likelihood = -410.976
```

y	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
x1	-.0974296	.0062866	-15.50	0.000	-.1097511	-.0851081
x2	-.1929588	.0099521	-19.39	0.000	-.2124646	-.173453
_cons	4.968026	.095185	52.19	0.000	4.781467	5.154585

```
. estimates store g2
```


if example

```
. estimates table g1 g2, b se stat(ll N)
```

Variable	g1	g2
x1	.0962749	-.0974296
	.01270857	.0062866
x2	-.1814847	-.19295883
	.02062008	.00995211
_cons	2.9568027	4.9680258
	.21161692	.09518502
ll	-212.8328	-410.976
N	122	178

legend: b/se

suest command

Description

- ▶ Combine estimation results into a seemingly unified result.

Features

- ▶ **test** equality of parameters between groups.
- ▶ Support for group-specific outcome models.

Limitations

- ▶ Constraints on parameters between groups is not possible.
- ▶ No support for **predict** or **margins**.
- ▶ No support for random effects, mixed-effects, or multilevel models.

suest example

```
. suest g1 g2
```

Simultaneous results for g1, g2

Number of obs = 300

		Coef.	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
g1_y	x1	.0962749	.0036486	26.39	0.000	.0891238	.103426
	x2	-.1814847	.0060325	-30.08	0.000	-.1933083	-.1696611
	_cons	2.956803	.0564931	52.34	0.000	2.846078	3.067527
g2_y	x1	-.0974296	.0021771	-44.75	0.000	-.1016967	-.0931625
	x2	-.1929588	.0034863	-55.35	0.000	-.1997919	-.1861258
	_cons	4.968026	.0357811	138.84	0.000	4.897896	5.038155

suest example

```
. suest, coeflegend
```

```
Simultaneous results for g1, g2
```

```
Number of obs = 300
```

	Coef.	Legend
g1_y		
x1	.0962749	_b[g1_y:x1]
x2	-.1814847	_b[g1_y:x2]
_cons	2.956803	_b[g1_y:_cons]
g2_y		
x1	-.0974296	_b[g2_y:x1]
x2	-.1929588	_b[g2_y:x2]
_cons	4.968026	_b[g2_y:_cons]

```
. test _b[g1_y:x1] = _b[g2_y:x1]
```

```
( 1) [g1_y]x1 - [g2_y]x1 = 0
      chi2( 1) = 2078.51
      Prob > chi2 = 0.0000
```

```
. test _b[g1_y:x2] = _b[g2_y:x2]
```

```
( 1) [g1_y]x2 - [g2_y]x2 = 0
      chi2( 1) = 2.71
      Prob > chi2 = 0.0996
```

Factor variables

Description

- ▶ Use factor variables notation to fit group-specific slopes and intercepts.

Features

- ▶ **test** equality of parameters between groups.
- ▶ Impose equality constraints between groups.
- ▶ Use **lrtest** to compare model fits with different group constraint patterns.
- ▶ Supported by models with random effects, mixed-effects, or multilevel models.
- ▶ **margins** and **contrast** were designed for this.

Factor variables

Limitations

- ▶ No support for group-specific outcome models.
- ▶ Support for group-specific auxiliary parameters is limited to models that support predictors in the auxiliary parameter equations.
- ▶ Random effects, mixed-effects, and multilevel parameters are group invariant.

Factor variables example

```
. poisson y bn.grp#c. (x1 x2) bn.grp, noconstant nolog
```

```
Poisson regression                Number of obs   =           300  
                                Wald chi2(6)     =       25499.84  
Log likelihood = -623.8088        Prob > chi2    =           0.0000
```

y	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
grp#c.x1						
1	.0962749	.0127086	7.58	0.000	.0713666	.1211832
2	-.0974296	.0062866	-15.50	0.000	-.1097511	-.0851081
grp#c.x2						
1	-.1814847	.0206201	-8.80	0.000	-.2218993	-.1410701
2	-.1929588	.0099521	-19.39	0.000	-.2124646	-.173453
grp						
1	2.956803	.2116169	13.97	0.000	2.542041	3.371564
2	4.968026	.095185	52.19	0.000	4.781467	5.154585

```
. estimates store free
```

Factor variables example

```
. constraint 1 _b[1.grp#x2] = _b[2.grp#x2]
. poisson y bn.grp#c.(x1 x2) bn.grp, noconstant constr(1) nolog
```

```
Poisson regression              Number of obs      =           300
                               Wald chi2(5)            =    25490.18
Log likelihood = -623.93426     Prob > chi2        =           0.0000

( 1) [y]1bn.grp#c.x2 - [y]2.grp#c.x2 = 0
```

y	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
grp#c.x1						
1	.0966158	.0126846	7.62	0.000	.0717545	.1214771
2	-.0973955	.0062872	-15.49	0.000	-.1097183	-.0850728
grp#c.x2						
1	-.1907964	.0089568	-21.30	0.000	-.2083513	-.1732414
2	-.1907964	.0089568	-21.30	0.000	-.2083513	-.1732414
grp						
1	3.04447	.1183256	25.73	0.000	2.812556	3.276384
2	4.948426	.0867631	57.03	0.000	4.778373	5.118478

```
. lrtest free .
```

```
Likelihood-ratio test              LR chi2(1) =           0.25
(Assumption: . nested in free)     Prob > chi2 =           0.6164
```


sem command

Description

- ▶ Fit combined linear outcome models across subgroups of the data while allowing some parameters to vary and constraining others to be equal across subgroups.

Features

- ▶ Easy syntax for constraints, and option `ginvariant()`.
- ▶ Test group invariance with postestimation command `estat ginvariant`.
- ▶ Use `lrtest` to compare model fits with different group constraint patterns.
- ▶ Fit multiple outcomes simultaneously.
- ▶ Support for CFA and SEM.

sem command

Limitations

- ▶ This framework is a linear outcome model, not all outcomes are usefully fit using a linear model.
- ▶ No support for random effects, mixed-effects, or multilevel models.

sem example

```
. generate logy = log(y)
. sem (logy <- x1 x2), group(grp) nolog noddescribe noheader nofootnote
Group          : 1                      Number of obs   =          122
```

	Coef.	OIM Std. Err.	z	P> z	[95% Conf. Interval]	
Structural						
logy						
x1	.0967704	.0033914	28.53	0.000	.0901234	.1034174
x2	-.1797791	.0054579	-32.94	0.000	-.1904764	-.1690819
_cons	2.930972	.0590996	49.59	0.000	2.815139	3.046805
var(e.logy)	.0138026	.0017672			.0107393	.0177397

```
Group          : 2                      Number of obs   =          178
```

	Coef.	OIM Std. Err.	z	P> z	[95% Conf. Interval]	
Structural						
logy						
x1	-.0958377	.0022986	-41.69	0.000	-.1003428	-.0913325
x2	-.1911029	.0034952	-54.68	0.000	-.1979535	-.1842524
_cons	4.939567	.0369362	133.73	0.000	4.867173	5.011961
var(e.logy)	.0088759	.0009408			.0072108	.0109255

```
. estimates store free
```

sem example

```
. quietly sem (logy <- x1 x2@a), group(grp)  
. lrtest free .
```

```
Likelihood-ratio test  
(Assumption: . nested in free)
```

```
LR chi2(1) = 3.03  
Prob > chi2 = 0.0817
```

sem example

```
. estat ginvariant
```

Tests for group invariance of parameters

	Wald Test			Score Test		
	chi2	df	p>chi2	chi2	df	p>chi2
Structural						
logy						
x1	2180.641	1	0.0000	.	.	.
x2	.	.	.	3.010	1	0.0827
_cons	6413.147	1	0.0000	.	.	.
var(e.logy)	6.268	1	0.0123	.	.	.

gsem command

Description

- ▶ Fit combined models across subgroups of the data while allowing some parameters to vary and constraining others to be equal across subgroups.

Features

- ▶ Easy syntax for constraints, and option **ginvariant()**.
- ▶ **test** equality of parameters between groups.
- ▶ Use **lrtest** to compare model fits with different group constraint patterns.
- ▶ Fit multiple outcomes simultaneously.
- ▶ Fit the outcome model of interest.
- ▶ Group-specific outcome models.
- ▶ Support for CFA, IRT, generalized SEM, random effects, multilevel latent variables

gsem example

```
. gsem (y <- x1 x2), poisson group(grp) ginvariant(none) nolog noheader
```

Group		:	1	Number of obs			=	122
			Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
y	x1		.0962749	.0127086	7.58	0.000	.0713666	.1211832
	x2		-.1814847	.0206201	-8.80	0.000	-.2218993	-.1410701
	_cons		2.956803	.2116169	13.97	0.000	2.542041	3.371564

Group		:	2	Number of obs			=	178
			Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
y	x1		-.0974296	.0062866	-15.50	0.000	-.1097511	-.0851081
	x2		-.1929588	.0099521	-19.39	0.000	-.2124646	-.173453
	_cons		4.968026	.095185	52.19	0.000	4.781467	5.154585

```
. estimates store free
```

gsem example

```
. quietly gsem (y <- x1 x2@a), poisson group(grp) ginvariant(none)  
. lrtest free .
```

```
Likelihood-ratio test  
(Assumption: . nested in free)
```

```
LR chi2(1) = 0.25  
Prob > chi2 = 0.6164
```


fmm: prefix

Description

- ▶ If the group membership is not observed, you can use a finite mixture model.

Features

- ▶ Prefix syntax using estimation commands you are already familiar with.
- ▶ Easy syntax for constraints, and option `lcinvariant()`.
- ▶ Fit the outcome model of interest.
- ▶ Class specific outcome models.
- ▶ Predict class membership probabilities.

fmm: example

```
. fmm 2, lcinvariant(none) nolog : poisson y x1 x2
```

```
Finite mixture model                               Number of obs   =           300  
Log likelihood = -765.56337
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
1.Class	(base outcome)					
2.Class						
_cons	-.5790036	.1573549	-3.68	0.000	-.8874136	-.2705936

```
Class      : 1  
Response   : y  
Model      : poisson
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
y						
x1	-.1051311	.0064635	-16.27	0.000	-.1177994	-.0924629
x2	-.2056394	.0105883	-19.42	0.000	-.226392	-.1848868
_cons	5.084371	.0987573	51.48	0.000	4.890811	5.277932

```
Class      : 2  
Response   : y  
Model      : poisson
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
y						
x1	.1106521	.0135704	8.15	0.000	.0840547	.1372495
x2	-.1849162	.0227743	-8.12	0.000	-.2295531	-.1402794
_cons	2.981518	.228961	13.02	0.000	2.532762	3.430273

```
. matrix b = e(b)
```

```
. estimates store free
```

fmm: example

```
. estat lcp
```

```
Latent class marginal probabilities          Number of obs      =          300
```

Class	Delta-method			
	Margin	Std. Err.	[95% Conf. Interval]	
1	.6408381	.0362175	.5672386	.7083561
2	.3591619	.0362175	.2916439	.4327614

fmm: example

```
. quietly fmm 2, lcinvariant(none) from(b) : poisson y x1 x2@a  
. lrtest free .
```

```
Likelihood-ratio test  
(Assumption: . nested in free)
```

```
LR chi2(1) = 0.67  
Prob > chi2 = 0.4117
```

Latent class models

Wikipedia

- ▶ A latent class model (LCM) relates a set of observed (usually discrete) multivariate variables to a set of latent variables. It is a type of latent variable model. It is called a latent class model because the latent variable is discrete (categorical).
- ▶ Latent class analysis (LCA) is a subset of structural equation modeling, used to find groups or subtypes of cases in multivariate categorical data.

LCA via `gsem` command

Features

- ▶ Specify categorical latent variables using new `lclass()` option.
- ▶ Easy syntax for constraints, and option `lcinvariant()`.
- ▶ `test` equality of parameters between classes.
- ▶ Use `lrtest` to compare model fits with different class constraint patterns.
- ▶ Fit multiple outcomes simultaneously.
- ▶ Class-specific outcome models.

Limitations

- ▶ Support is restricted to model specifications that do not include continuous latent variables.

LCA example dataset

Observed variables

- ▶ y_1 , y_2 , and y_3 are binary dependent variables of interest.
- ▶ x is a continuous independent variable.
Use it to predict class membership and the dependent variables.
- ▶ Assume there are 2 latent classes.

LCA example

```
gsem (y* <- x) (2.C <- x), logit lclass(C 2) lcinvariant(none)  
matrix b = e(b)  
estimates store free
```


LCA example

```
. gsem (y* <- x) (2.C <- x), logit lclass(C 2) lcinvariant(none) nolog nodvheader
```

```
Generalized structural equation model      Number of obs      =      3,000  
Log likelihood = -5088.3207
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
1.C	(base outcome)					
2.C						
x	-.3669334	.0538679	-6.81	0.000	-.4725124	-.2613543
_cons	-.2821683	.0498944	-5.66	0.000	-.3799596	-.184377

```
Class           : 1
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
y1						
x	.9722039	.1006049	9.66	0.000	.775022	1.169386
_cons	-1.98065	.1169993	-16.93	0.000	-2.209965	-1.751336
y2						
x	1.073417	.1033773	10.38	0.000	.8708013	1.276033
_cons	-2.011822	.1237851	-16.25	0.000	-2.254436	-1.769208
y3						
x	1.328942	.1152357	11.53	0.000	1.103084	1.5548
_cons	-2.315209	.1443729	-16.04	0.000	-2.598175	-2.032244

```
Class           : 2
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
y1						
x	1.088556	.1259243	8.64	0.000	.841749	1.335363
_cons	2.105665	.1543028	13.65	0.000	1.803237	2.408093
y2						
x	.8507127	.1265496	6.72	0.000	.60268	1.098745
_cons	2.040876	.148814	13.71	0.000	1.749206	2.332546
y3						
x	1.11105	.1274936	8.71	0.000	.8611668	1.360933
_cons	2.231151	.1626808	13.71	0.000	1.912303	2.55

```
. matrix b = e(b)
```

```
. estimates store free
```

LCA example

```
. estat lcprob
```

```
Latent class marginal probabilities          Number of obs      =          3,000
```

	Margin	Delta-method Std. Err.	[95% Conf. Interval]	
C				
1	.5691789	.0118173	.5458825	.5921731
2	.4308211	.0118173	.4078269	.4541175

LCA example

```
. quietly gsem (y* <- x) (2.C <- x), logit lclass(C 2) lcinvariant(coef) from(b)
. lrtest free .
Likelihood-ratio test                                LR chi2(3) =          4.09
(Assumption: . nested in free)                      Prob > chi2 =        0.2523
```

What's next

- ▶ Add latent class support for models with continuous latent variables.
- ▶ ...