STATA FINITE MIXTURE MODELS REFERENCE MANUAL

RELEASE 19

A Stata Press Publication
StataCorp LLC
College Station, Texas

Published by Stata Press, 4905 Lakeway Drive, College Station, Texas 77845

ISBN-10: 1-59718-425-X
ISBN-13: 978-1-59718-425-0

The suggested citation for this software is

StataCorp. 2025. *Stata 19*. Statistical software. StataCorp LLC.

The suggested citation for this manual is

StataCorp. 2025. *Stata 19 Finite Mixture Models Reference Manual*. College Station, TX: Stata Press.

Contents

Cross-referencing the documentation

When reading this manual, you will find references to other Stata manuals, for example, **[U] 27 Overview of Stata estimation commands**; **[R] regress**; and **[D] reshape**. The first example is a reference to chapter 27, *Overview of Stata estimation commands*, in the *User's Guide*; the second is a reference to the `regress` entry in the *Base Reference Manual*; and the third is a reference to the `reshape` entry in the *Data Management Reference Manual*.

All the manuals in the Stata Documentation have a shorthand notation:

[GSM]	*Getting Started with Stata for Mac*
[GSU]	*Getting Started with Stata for Unix*
[GSW]	*Getting Started with Stata for Windows*
[U]	*Stata User's Guide*
[R]	*Stata Base Reference Manual*
[ADAPT]	*Stata Adaptive Designs: Group Sequential Trials Reference Manual*
[BAYES]	*Stata Bayesian Analysis Reference Manual*
[BMA]	*Stata Bayesian Model Averaging Reference Manual*
[CAUSAL]	*Stata Causal Inference and Treatment-Effects Estimation Reference Manual*
[CM]	*Stata Choice Models Reference Manual*
[D]	*Stata Data Management Reference Manual*
[DSGE]	*Stata Dynamic Stochastic General Equilibrium Models Reference Manual*
[ERM]	*Stata Extended Regression Models Reference Manual*
[FMM]	*Stata Finite Mixture Models Reference Manual*
[FN]	*Stata Functions Reference Manual*
[G]	*Stata Graphics Reference Manual*
[H2OML]	*Machine Learning in Stata Using H2O: Ensemble Decision Trees Reference Manual*
[IRT]	*Stata Item Response Theory Reference Manual*
[LASSO]	*Stata Lasso Reference Manual*
[XT]	*Stata Longitudinal-Data/Panel-Data Reference Manual*
[META]	*Stata Meta-Analysis Reference Manual*
[ME]	*Stata Multilevel Mixed-Effects Reference Manual*
[MI]	*Stata Multiple-Imputation Reference Manual*
[MV]	*Stata Multivariate Statistics Reference Manual*
[PSS]	*Stata Power, Precision, and Sample-Size Reference Manual*
[P]	*Stata Programming Reference Manual*
[RPT]	*Stata Reporting Reference Manual*
[SP]	*Stata Spatial Autoregressive Models Reference Manual*
[SEM]	*Stata Structural Equation Modeling Reference Manual*
[SVY]	*Stata Survey Data Reference Manual*
[ST]	*Stata Survival Analysis Reference Manual*
[TABLES]	*Stata Customizable Tables and Collected Results Reference Manual*
[TS]	*Stata Time-Series Reference Manual*
[I]	*Stata Index*
[M]	*Mata Reference Manual*

Description Remarks and examples Acknowledgment References Also see

Description

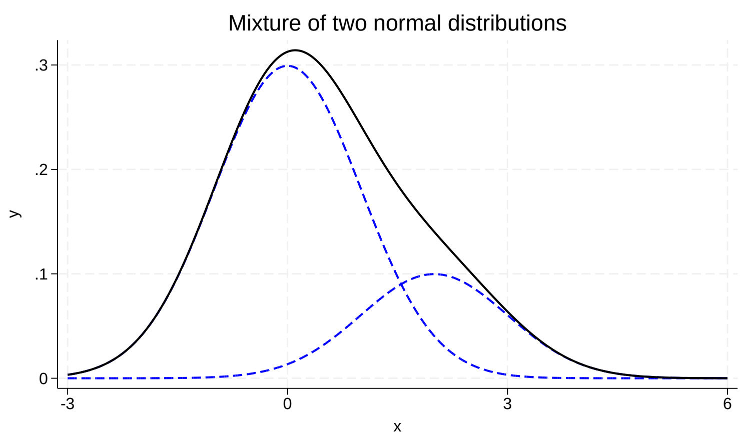
Finite mixture models (FMMs) are used to classify observations, to adjust for clustering, and to model unobserved heterogeneity. In finite mixture modeling, the observed data are assumed to belong to unobserved subpopulations called classes, and mixtures of probability densities or regression models are used to model the outcome of interest. After fitting the model, class membership probabilities can also be predicted for each observation. This entry discusses some fundamental and theoretical aspects of FMMs and illustrates these aspects with a worked example.

Remarks and examples

Remarks are presented under the following headings:

Introduction

The main concept in finite mixture modeling is that the observed data come from distinct, but unobserved, subpopulations. To illustrate, we plot the observed distribution of a whole population (solid line) and the unobserved densities of two underlying subpopulations (dashed lines).



The observed distribution looks approximately normal, with a slight asymmetry because of more values falling above zero than below. This asymmetry occurs because the distribution is a mixture of two normal densities; the right-hand density skews the distribution to the right. We can use FMMs to estimate the means and variances of the two underlying densities along with their proportions in the overall population.

More generally, we can use FMMs to model mixtures containing any number of subpopulations, and the subpopulation-specific models need not be limited to a mixture of normal densities. FMMs allow mixtures of linear and generalized linear regression models, including models for binary, ordinal, nominal, and count responses, and allow the inclusion of covariates with subpopulation-specific effects. We can also make inferences about each subpopulation and classify individual observations into a subpopulation.

Because of their flexibility, FMMs have been used extensively in various fields to classify observations, to adjust for clustering, and to model unobserved heterogeneity. Mixtures of normal densities with equal variances can be used to approximate any arbitrary continuous distribution, which makes FMMs a popular tool to model multimodal, skewed, or asymmetrical data. A mixture of regression models can be used to model phenomena such as clustering of internet traffic (Jorgensen 2004), demand for medical care (Deb and Trivedi 1997), disease risk (Schlattmann, Dietz, and Böhning 1996), and perceived consumer risk (Wedel and DeSarbo 1993). A mixture of a count model and a degenerate point mass distribution is often used for modeling zero-inflated and truncated count outcomes; see, for example, Jones et al. (2013, chap. 11). McLachlan and Peel (2000) and Frühwirth-Schnatter (2006) provide a comprehensive treatment of finite mixture modeling.

From a broader statistical perspective, FMMs are related to latent class analysis (LCA) models; both are used to identify classes using information from manifest (observed) variables. The difference is that FMMs allow parameters in a regression model for a single dependent variable to differ across classes while traditional LCA fits intercept-only models to multiple dependent variables. FMM is also a subset of structural equation modeling (SEM) where the latent variable is assumed to be categorical; see [SEM] **Intro 1**, [SEM] **Intro 2**, [SEM] **gsem**, and Skrondal and Rabe-Hesketh (2004, chap. 3) for a theoretical discussion. If your latent variable is continuous and your manifest variables are discrete, you can use item response theory models; see [IRT] **irt**. If both your latent variable and manifest variables are continuous, you can fit a structural equation model; see [SEM] **sem**.

Throughout this manual, we use the terms “class”, “group”, “type”, or “component” to refer to an unobserved subpopulation. We use the terms “class probability” or “component probability” to refer to the probability of belonging to a given component in the mixture. Class probabilities are also referred to in the literature as “mixing weights” or “mixing proportions”.

Finite mixture models

FMMs are probabilistic models that combine two or more density functions. In an FMM, the observed responses $\boldsymbol{y}$ are assumed to come from g distinct classes $f_1, f_2, \ldots, f_g$ in proportions $\pi_1, \pi_2, \ldots, \pi_g$. In its simplest form, we can write the density of a g-component mixture model as

$$f(\boldsymbol{y}) = \sum_{i=1}^{g} \pi_i f_i(\boldsymbol{y}|\boldsymbol{x}'\boldsymbol{\beta}_i)$$

where π_i is the probability for the ith class, $0 \leq \pi_i \leq 1$ and $\sum \pi_i = 1$, and $f_i(\cdot)$ is the conditional probability density function for the observed response in the ith class model.

`fmm` uses the multinomial logistic distribution to model the probabilities for the latent classes. The probability for the ith latent class is given by

$$\pi_i = \frac{\exp(\gamma_i)}{\sum_{j=1}^{g} \exp(\gamma_j)}$$

where γ_i is the linear prediction for the ith latent class. By default, the first latent class is the base level so that $\gamma_1 = 0$ and $\exp(\gamma_1) = 1$.

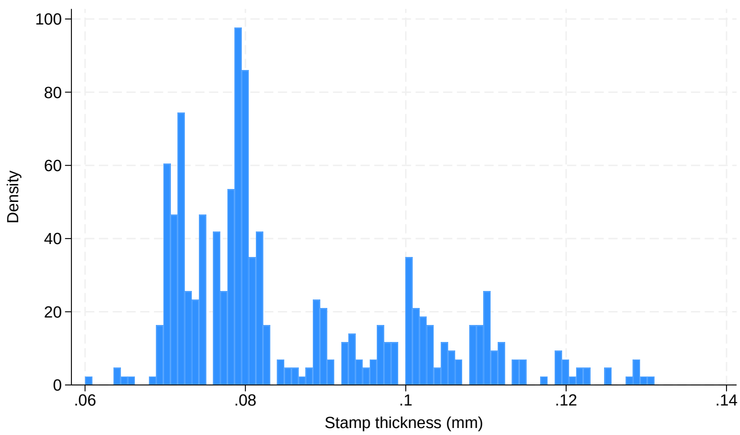
The likelihood is computed as the sum of the probability-weighted conditional likelihood from each latent class; see *Methods and formulas* in [FMM] **fmm** for details.

Mixture of normal distributions—FMM by example

The 1872 Hidalgo stamp of Mexico was printed on different paper types, which was typical of stamps of that era. For collectors, a stamp from a printing that used thicker paper is more valuable. We can use an FMM to predict the probability that a stamp is from a printing that used thick paper.

`stamp.dta` contains data on 485 measurements of stamp thickness, recorded to a thousandth of a millimeter. Here we plot the histogram of the measurements.

```
. use https://www.stata-press.com/data/r19/stamp
(1872 Hidalgo stamp of Mexico)
. histogram thickness, bins(80)
(bin=80, start=.06, width=.0008875)
```



At a minimum, the histogram suggests bimodality in the data, but we follow Izenman and Sommer (1988) and fit a mixture of three normal distributions to the data, each with its own mean and variance. We also estimate the proportion that each distribution contributes to the overall density. You can think of the three distributions as representing three different types of paper (thick, medium, thin) that the stamps were printed on. More specifically, our model is

$$f(\boldsymbol{y}) = \pi_1 N(\mu_1, \sigma_1^2) + \pi_2 N(\mu_2, \sigma_2^2) + \pi_3 N(\mu_3, \sigma_3^2)$$

The probability of being in each class is estimated using multinomial logistic regression

$$\pi_1 = \frac{1}{1 + \exp(\gamma_2) + \exp(\gamma_3)}$$

$$\pi_2 = \frac{\exp(\gamma_2)}{1 + \exp(\gamma_2) + \exp(\gamma_3)}$$

$$\pi_3 = \frac{\exp(\gamma_3)}{1 + \exp(\gamma_2) + \exp(\gamma_3)}$$

where the γ_i are intercepts in the multinomial logit model. By default, the first class is treated as the base, so $\gamma_1 = 0$.

To fit this model, we type

```
. fmm 3: regress thickness
```

We type `fmm 3:` because we have a mixture of three components. We type `regress thickness` to tell `fmm` to fit a linear regression model for each component. With no covariates, `regress` reduces to estimating the mean and variance of a Gaussian (normal) density for each component.

The result of typing our estimation command is

```
. fmm 3: regress thickness
Fitting class model:
Iteration 0:  (class) log likelihood =  -532.8249
Iteration 1:  (class) log likelihood =  -532.8249
Fitting outcome model:
Iteration 0:  (outcome) log likelihood =  1949.1228
Iteration 1:  (outcome) log likelihood =  1949.1228
Refining starting values:
Iteration 0:  (EM) log likelihood =  1396.8814
Iteration 1:  (EM) log likelihood =  1404.8995
Iteration 2:  (EM) log likelihood =  1412.4626
Iteration 3:  (EM) log likelihood =  1416.9678
Iteration 4:  (EM) log likelihood =  1419.0044
Iteration 5:  (EM) log likelihood =  1419.0582
Iteration 6:  (EM) log likelihood =  1417.9719
Iteration 7:  (EM) log likelihood =  1416.4213
Iteration 8:  (EM) log likelihood =  1414.8176
Iteration 9:  (EM) log likelihood =  1413.3462
Iteration 10: (EM) log likelihood =  1412.0695
Iteration 11: (EM) log likelihood =   1410.992
Iteration 12: (EM) log likelihood =  1410.0961
Iteration 13: (EM) log likelihood =  1409.3574
Iteration 14: (EM) log likelihood =  1408.7518
Iteration 15: (EM) log likelihood =  1408.2578
Iteration 16: (EM) log likelihood =  1407.8564
Iteration 17: (EM) log likelihood =  1407.5315
Iteration 18: (EM) log likelihood =  1407.2694
Iteration 19: (EM) log likelihood =  1407.0695
Iteration 20: (EM) log likelihood =  1406.9013
note: EM algorithm reached maximum iterations.
```

```
Fitting full model:

Iteration 0:  Log likelihood =  1516.5252
Iteration 1:  Log likelihood =  1517.1348  (not concave)
Iteration 2:  Log likelihood =  1517.8203  (not concave)
Iteration 3:  Log likelihood =   1518.153
Iteration 4:  Log likelihood =  1518.6491
Iteration 5:  Log likelihood =  1518.8474
Iteration 6:  Log likelihood =  1518.8484
Iteration 7:  Log likelihood =  1518.8484

Finite mixture model                                  Number of obs = 485
Log likelihood = 1518.8484
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
1.Class	(base outcome)					
2.Class						
_cons	.6410696	.1625089	3.94	0.000	.3225581	.9595812
3.Class						
_cons	.8101538	.1493673	5.42	0.000	.5173992	1.102908

```
Class:     1
Response:  thickness
Model:     regress
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
thickness						
_cons	.0712183	.0002011	354.20	0.000	.0708242	.0716124
var(e.thic~s)	1.71e-06	4.49e-07			1.02e-06	2.86e-06

```
Class:     2
Response:  thickness
Model:     regress
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
thickness						
_cons	.0786016	.0002496	314.86	0.000	.0781123	.0790909
var(e.thic~s)	5.74e-06	9.98e-07			4.08e-06	8.07e-06

```
Class:     3
Response:  thickness
Model:     regress
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
thickness						
_cons	.0988789	.0012583	78.58	0.000	.0964127	.1013451
var(e.thic~s)	.0001967	.0000223			.0001575	.0002456

The output shows four iteration logs. The first three are for models that are fit to obtain starting values. Finding good starting values is often challenging for mixture models. `fmm` provides a variety of options for specifying and computing starting values; see *Options* in [FMM] **fmm** for more information.

The first output table presents the estimated class probabilities on a multinomial logistic scale. We can transform these estimates into probabilities as follows:

$$\pi_1 = \frac{1}{1 + \exp(0.64) + \exp(0.81)} \approx 0.19$$

$$\pi_2 = \frac{\exp(0.64)}{1 + \exp(0.64) + \exp(0.81)} \approx 0.37$$

$$\pi_3 = \frac{\exp(0.81)}{1 + \exp(0.64) + \exp(0.81)} \approx 0.44$$

More conveniently, we can use the `estat lcprob` command, which calculates these probabilities and the associated standard errors and confidence intervals; see [FMM] **estat lcprob**.

```
. estat lcprob
Latent class marginal probabilities                    Number of obs = 485
```

	Margin	Delta-method std. err.	[95% conf.	interval]
Class				
1	.1942968	.0221242	.1545535	.2413428
2	.3688746	.0286318	.3147305	.4265356
3	.4368286	.027885	.383149	.49203

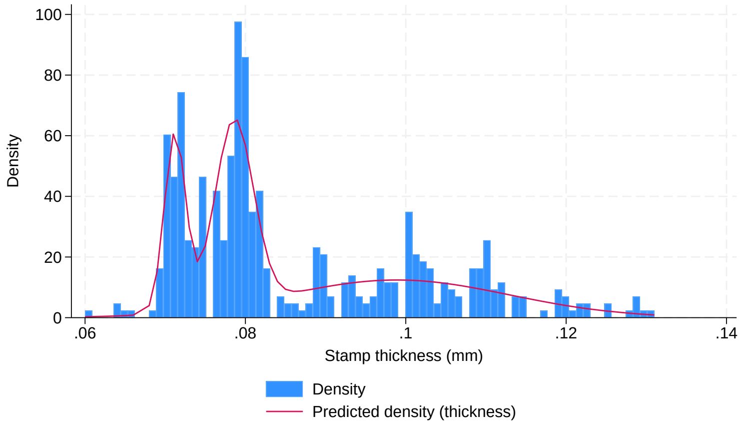
The three remaining tables of the `fmm` output show the estimated means and variances of each normal distribution.

The resulting mixture density, with maximum likelihood estimates of means, variances, and class probabilities, is given by

$$0.19 \times N(0.071, 0.0000017) + 0.37 \times N(0.079, 0.0000057) + 0.44 \times N(0.099, 0.0001967)$$

This equation gives the predicted density of stamp thickness, and we can plot it against the empirical distribution of stamp thickness as follows:

```
. predict den, density marginal
. histogram thickness, bins(80) addplot(line den thickness) legend(pos(6))
(bin=80, start=.06, width=.0008875)
```



We see that the first two components with small variances model the left-hand side of the empirical distribution, whereas the third component with much larger variance covers the long tail on the right-hand side of the empirical distribution.

We can use the predictions of the posterior probability of class membership to evaluate the probability of being in each class for each stamp. For the first stamp in our dataset, the probability of being in class 3, the thick paper type, is 1.

```
. predict pr*, classposteriorpr
. format %4.3f pr*
. list thickness pr* in 1, abbreviate(10)
```

	thickness	pr1	pr2	pr3
1.	.06	0.000	0.000	1.000

Because there are no covariates in the model, the posterior probabilities are the same for any stamp with a given thickness and are as follows.

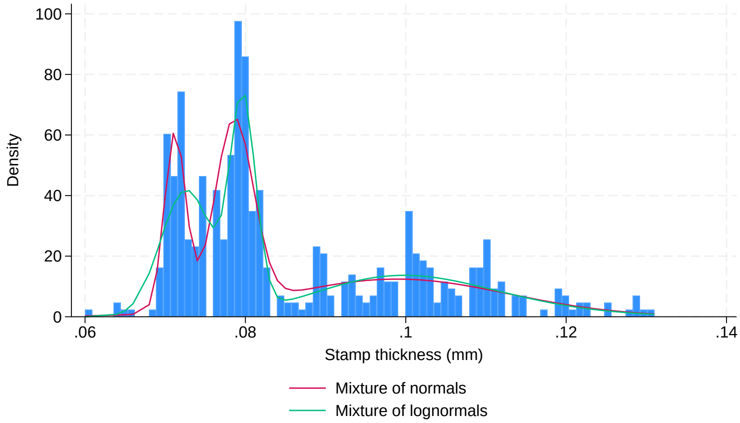
thickness	pr1	pr2	pr3
.06	0.000	0.000	1.000
.064	0.000	0.000	1.000
.065	0.001	0.000	0.999
.066	0.026	0.000	0.974
.068	0.723	0.001	0.276
.069	0.915	0.001	0.083
.07	0.960	0.002	0.037
.071	0.965	0.007	0.028
.072	0.937	0.026	0.037
.073	0.789	0.134	0.076
.074	0.335	0.525	0.140
.075	0.038	0.838	0.123
.076	0.002	0.910	0.088
.077	0.000	0.930	0.070
.078	0.000	0.936	0.064
.079	0.000	0.930	0.070
.08	0.000	0.912	0.088
.081	0.000	0.871	0.129
.082	0.000	0.788	0.212
.083	0.000	0.635	0.365
.084	0.000	0.406	0.594
.085	0.000	0.185	0.815
.086	0.000	0.060	0.940
.087	0.000	0.015	0.985
.088	0.000	0.003	0.997
.089	0.000	0.001	0.999
.09-.131	0.000	0.000	1.000

The third mixture component has a relatively large variance, so the four thinnest measures end up being incorrectly classified into the thick paper type. Because stamp thickness cannot be negative, we can improve the model fit if we use a density with support only on the positive real line, such as the lognormal distribution.

```
. fmm 3: glm thickness, family(lognormal)
```

(*output omitted*)

We plot the predicted density from the mixture of normals with the density from the mixture of lognormals.



The mixture of lognormals correctly classifies the thinnest stamps into the thin paper type, which is confirmed by the predicted posterior probabilities.

thickness	pr1	pr2	pr3
.06	.889	0	.111
.064	.992	0	.008
.065	.994	0	.006
.066	.996	0	.004
.068	.997	0	.003
.069	.997	0	.003
.07	.996	0	.004
.071	.996	0	.004
.072	.995	0	.005
.073	.992	0	.008
.074	.987	.001	.011
.075	.965	.017	.018
.076	.849	.124	.027
.077	.532	.437	.031
.078	.233	.741	.026
.079	.102	.874	.024
.08	.056	.915	.028
.081	.041	.911	.048
.082	.039	.85	.111
.083	.042	.654	.305
.084	.034	.288	.678
.085	.017	.056	.928
.086	.006	.006	.988
.087	.002	0	.998
.088	.001	0	.999
.89-.131	0	0	1

Beyond mixtures of distributions

We have just scratched the surface of what can be done with `fmm`. We can fit mixtures of linear and generalized linear regression models where the effect of the covariates and the covariates themselves differ by class; see [FMM] **fmm estimation** for a list of supported outcome models. We can also model class probabilities with common or class-specific covariates.

More complicated FMMs can be fit using `gsem` within the LCA framework. `gsem` allows more than one response variable per component and more than one categorical latent variable; see, for instance, [SEM] **Example 54g**, where we fit a mixture of Poisson regression models to multiple responses. See *Latent class analysis (LCA)* in [SEM] **Intro 2** and *Latent class models* in [SEM] **Intro 5** for an overview of latent class modeling with `gsem`.

Acknowledgment

We gratefully acknowledge the previous work by Partha Deb at Hunter College and the Graduate Center, City University of New York; see Deb (2007).

References

Cerulli, G., R. Simone, F. Di Iorio, D. Piccolo, and C. F. Baum. 2022. Fitting mixture models for feeling and uncertainty for rating data analysis. *Stata Journal* 22: 195–223.

Deb, P. 2007. fmm: Stata module to estimate finite mixture models. Statistical Software Components S456895, Department of Economics, Boston College. https://ideas.repec.org/c/boc/bocode/s456895.html.

Deb, P., and P. K. Trivedi. 1997. Demand for medical care by the elderly: A finite mixture approach. *Journal of Applied Econometrics* 12: 313–336. https://doi.org/10.1002/(SICI)1099-1255(199705)12:3<313::AID-JAE440>3.0.CO;2-G.

Frühwirth-Schnatter, S. 2006. *Finite Mixture and Markov Switching Models*. New York: Springer.

Izenman, A. J., and C. J. Sommer. 1988. Philatelic mixtures and multimodal densities. *Journal of the American Statistical Association* 83: 941–953. https://doi.org/10.2307/2290118.

Jenkins, S. P., and F. Rios-Avila. 2023. Finite mixture models for linked survey and administrative data: Estimation and postestimation. *Stata Journal* 23: 53–85.

Jones, A. M., N. Rice, T. Bago D'Uva, and S. Balia. 2013. *Applied Health Economics*. 2nd ed. New York: Routledge.

Jorgensen, M. 2004. Using multinomial mixture models to cluster Internet traffic. *Australian and New Zealand Journal of Statistics* 46: 205–218. https://doi.org/10.1111/j.1467-842X.2004.00325.x.

McLachlan, G. J., and D. Peel. 2000. *Finite Mixture Models*. New York: Wiley. https://doi.org/10.1002/0471721182.

Saint-Cyr, L. D. F., and L. Piet. 2019. mixmcm: A community-contributed command for fitting mixtures of Markov chain models using maximum likelihood and the EM algorithm. *Stata Journal* 19: 294–334.

Schlattmann, P., E. Dietz, and D. Böhning. 1996. Covariate adjusted mixture models and disease mapping with the program dismapwin. *Statistics in Medicine* 15: 919–929. https://doi.org/10.1002/(SICI)1097-0258(19960415)15:7/9<919::AID-SIM260>3.0.CO;2-W.

Skrondal, A., and S. Rabe-Hesketh. 2004. *Generalized Latent Variable Modeling: Multilevel, Longitudinal, and Structural Equation Models*. Boca Raton, FL: Chapman and Hall/CRC.

Wedel, M., and W. S. DeSarbo. 1993. A latent class binomial logit methodology for the analysis of paired comparison choice data: An application reinvestigating the determinants of perceived risk. *Decision Sciences* 24: 1157–1170. https://doi.org/10.1111/j.1540-5915.1993.tb00508.x.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **Example 1a** — Mixture of linear regression models

[FMM] **Example 2** — Mixture of Poisson regression models

[FMM] **Example 3** — Zero-inflated models

[FMM] **Example 4** — Mixture cure models for survival data

[FMM] **Glossary**

[SEM] **gsem** — Generalized structural equation model estimation command

fmm estimation — Fitting finite mixture models

Description   Also see

Description

Fitting finite mixture models in Stata is similar to standard estimation—simply prefix the estimation commands with `fmm #:`, where # is the number of mixtures; see [FMM] **fmm**.

The following estimation commands support the `fmm` prefix.

Command	Entry	Description
Linear regression models		
`regress`	[FMM] **fmm: regress**	Linear regression
`truncreg`	[FMM] **fmm: truncreg**	Truncated regression
`intreg`	[FMM] **fmm: intreg**	Interval regression
`tobit`	[FMM] **fmm: tobit**	Tobit regression
`ivregress`	[FMM] **fmm: ivregress**	Instrumental-variables regression
Binary-response regression models		
`logit`	[FMM] **fmm: logit**	Logistic regression, reporting coefficients
`probit`	[FMM] **fmm: probit**	Probit regression
`cloglog`	[FMM] **fmm: cloglog**	Complementary log–log regression
Ordinal-response regression models		
`ologit`	[FMM] **fmm: ologit**	Ordered logistic regression
`oprobit`	[FMM] **fmm: oprobit**	Ordered probit regression
Categorical-response regression models		
`mlogit`	[FMM] **fmm: mlogit**	Multinomial (polytomous) logistic regression
Count-response regression models		
`poisson`	[FMM] **fmm: poisson**	Poisson regression
`nbreg`	[FMM] **fmm: nbreg**	Negative binomial regression
`tpoisson`	[FMM] **fmm: tpoisson**	Truncated Poisson regression
Generalized linear models		
`glm`	[FMM] **fmm: glm**	Generalized linear models
Fractional-response regression models		
`betareg`	[FMM] **fmm: betareg**	Beta regression
Survival regression models		
`streg`	[FMM] **fmm: streg**	Parametric survival models

`fmm:` allows different regression models for different components of the mixture; see [FMM] **fmm**. `fmm:` also allows one or more components to be a degenerate distribution taking on a single integer value with probability one; see [FMM] **fmm: pointmass**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **Glossary**

Description

The `fmm` prefix fits finite mixture models; see [FMM] **fmm estimation** for the list of supported commands.

Quick start

Mixture of three normal distributions of `y`

```
fmm 3: regress y
```

Mixture of three linear regression models of `y` on `x1` and `x2`

```
fmm 3: regress y x1 x2
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 3, lcprob(z1 z2): regress y x1 x2
```

Same as above, but with additional class-specific regression covariates `x3`, `x4`, and `x5`

```
fmm, lcprob(z1 z2): (regress y x1 x2 x3)    ///
          (regress y x1 x2 x4)    ///
          (regress y x1 x2 x5)
```

Same as above, but with additional class-specific probability covariates `z3` and `z4`

```
fmm: (regress y x1 x2 x3)            ///
   (regress y x1 x2 x4, lcprob(z1 z2 z3))  ///
   (regress y x1 x2 x5, lcprob(z1 z2 z4))
```

Menu

Statistics > FMM (finite mixture models) > General estimation and regression

Syntax

Standard syntax

fmm # [*if*] [*in*] [*weight*] [, *fmmopts*] : *component*

Hybrid syntax

fmm [*if*] [*in*] [*weight*] [, *fmmopts*] : (*component*$_1$) (*component*$_2$) ...

where the standard syntax for *component* is

model depvar indepvars [, *options*]

the hybrid syntax for *component* is

model depvar indepvars [, lcprob(*varlist*) *options*]

model is an estimation command, and *options* are *model*-specific estimation options.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see [U] **11.4.3 Factor variables**.

`by`, `collect`, `statsby`, and `svy` are allowed; see [U] **11.1.10 Prefix commands**.

`vce()` and weights are not allowed with the `svy` prefix; see [SVY] **svy**.

`fweight`s, `iweight`s, and `pweight`s are allowed; see [U] **11.1.6 weight**.

`collinear` and `coeflegend` do not appear in the dialog box.

See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

pclassname	Description
`cons`	intercepts and cutpoints
`coef`	fixed coefficients
`errvar`	covariances of errors
`scale`	scaling parameters
`all`	all the above
`none`	none of the above; the default

Options

Model

`lcinvariant(`*pclassname*`)` specifies which parameters of the model are constrained to be equal across the latent classes; the default is `lcinvariant(none)`.

`lcprob(`*varlist*`)` specifies that the linear prediction for a given latent class probability include the variables in *varlist*. `lcinvariant()` has no effect on these parameters.

In the standard syntax, *varlist* is used in the linear prediction for each latent class probability.

In the hybrid syntax, specify `lcprob(`$varlist_i$`)` in $component_i$ to include $varlist_i$ in the linear prediction for the ith latent class probability. `lcprob()` is not allowed to be specified in *fmmopts* if it is being used in one or more *component* specifications.

In the hybrid syntax, if you specify `lcprob()` in the component that corresponds with the base latent class, the option is ignored.

`lclabel(`*name*`)` specifies a name for the categorical latent variable; the default is `lclabel(Class)`.

`lcbase(`*#*`)` specifies that # is to be treated as the base latent class.

In the standard syntax, the default is `lcbase(1)`.

In the hybrid syntax, the default base is the latent class corresponding to the first *component* that does not have `lcprob()` specified. If all components have `lcprob()`, the first *component* is the base and the `lcprob()` option specified for the first *component* is ignored.

`constraints()`; see [R] **Estimation options**.

SE/Robust

`vce(`*vcetype*`)` specifies the type of standard error reported, which includes types that are derived from asymptotic theory (`oim`, `opg`), that are robust to some kinds of misspecification (`robust`), and that allow for intragroup correlation (`cluster` *clustvar*); see [R] ***vce_option***.

Reporting

level(#); see [R] **Estimation options**.

nocnsreport suppresses the display of the constraints. Fixed-to-zero constraints that are automatically set by fmm are not shown in the report to keep the output manageable.

noheader suppresses the header above the parameter table, the display that reports the final log-likelihood value, number of observations, etc.

nodvheader suppresses the dependent variables information from the header above each parameter table.

notable suppresses the parameter tables.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fvwrap(#), fvwrapon(*style*), cformat(*%fmt*), pformat(*%fmt*), sformat(*%fmt*), and nolstretch; see [R] **Estimation options**.

Maximization

maximize_options: difficult, technique(*algorithm_spec*), iterate(#), [no]log, trace, gradient, showstep, hessian, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(*init_specs*); see [R] **Maximize**. These options are seldom used.

startvalues() specifies how starting values are to be computed. Starting values specified in from() override the computed starting values.

startvalues(factor [, *maxopts*]) specifies that starting values are computed by assigning each observation to an initial latent class that is determined by running a factor analysis on all the observed variables in the specified model. This is the default.

startvalues(classid *varname*[, *maxopts*]) specifies that starting values are computed by assigning each observation to an initial latent class specified in *varname*. *varname* is required to have each class represented in the estimation sample.

startvalues(classpr *varlist*[, *maxopts*]) specifies that starting values are computed using the initial class probabilities specified in *varlist*. *varlist* is required to contain g variables for a model with g latent classes. The values in *varlist* are normalized to sum to 1 within each observation.

startvalues(randomid [, draws(#) seed(#) *maxopts*]) specifies that starting values are computed by randomly assigning observations to initial classes.

startvalues(randompr [, draws(#) seed(#) *maxopts*]) specifies that starting values are computed by randomly assigning initial class probabilities.

startvalues(jitter [$\#_c$ [$\#_v$]], draws(#) seed(#) *maxopts*]) specifies that starting values are constructed by randomly perturbing the values from a Gaussian approximation to each outcome.

$\#_c$ is the magnitude for randomly perturbing coefficients, intercepts, cutpoints, and scale parameters; the default value is 1.

$\#_v$ is the magnitude for randomly perturbing variances for Gaussian outcomes; the default value is 1.

startvalues(zero) specifies that starting values are to be set to 0. This option is only useful if you use from() to specify starting values for some parameters and want the remaining starting values to be 0.

Most starting values options have suboptions that allow for tuning the starting values calculations:

maxopts is a subset of the standard *maximize_options*: `difficult`, `technique(`*algorithm_spec*`)`, `iterate(#)`, [`no`]`log`, `trace`, `gradient`, `showstep`, `hessian`, `showtolerance`, `tolerance(#)`, `ltolerance(#)`, and `nrtolerance(#)`; see [R] **Maximize**.

`draws(#)` specifies the number of random draws. For `startvalues(randomid)`, `startvalues(randompr)`, and `startvalues(jitter)`, `fmm` will generate # random draws and select the starting values from the draw with the best log-likelihood value from the EM iterations. The default is `draws(1)`.

`seed(#)` sets the random-number seed.

`emopts(`*maxopts*`)` controls maximization of the log likelihood for the EM algorithm. *maxopts* is the same subset of *maximize_options* that are allowed in the `startvalues()` option, but some of the defaults are different for the EM algorithm. The default maximum number of iterations is `iterate(20)`. The default coefficient vector tolerance is `tolerance(1e-4)`. The default log-likelihood tolerance is `ltolerance(1e-6)`.

`noestimate` specifies that the model is not to be fit. Instead, starting values are to be shown (as modified by the above options if modifications were made), and they are to be shown using the `coeflegend` style of output. An important use of this option is before you have modified starting values at all; you can type the following:

```
. fmm ..., ... noestimate : ...
. matrix b = e(b)
. ... (modify elements of b) ...
. fmm ..., ... from(b) : ...
```

The following options are available with `fmm` but are not shown in the dialog box:

`collinear`; see [R] **Estimation options**.

`coeflegend` displays the legend that reveals how to specify estimated coefficients in `_b[]` notation, which you are sometimes required to type when specifying postestimation commands.

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For the list of estimation commands supported by the `fmm` prefix, see [FMM] **fmm estimation**.

Examples using `fmm` can be found at

[FMM] **Example 1a** — Mixture of linear regression models
[FMM] **Example 1b** — Covariates for class membership
[FMM] **Example 1c** — Testing coefficients across class models
[FMM] **Example 1d** — Component-specific covariates
[FMM] **Example 2** — Mixture of Poisson regression models
[FMM] **Example 3** — Zero-inflated models
[FMM] **Example 4** — Mixture cure models for survival data

Stored results

fmm stores the following in e():

Scalars

e(N)	number of observations
e(k)	number of parameters
e(k_eq)	number of equations in e(b)
e(k_dv)	number of dependent variables
e(k_cat#)	number of categories for the #th *depvar*, ordinal
e(k_out#)	number of categories for the #th *depvar*, mlogit
e(ll)	log likelihood
e(N_clust)	number of clusters
e(rank)	rank of e(V)
e(ic)	number of iterations
e(rc)	return code
e(converged)	1 if target model converged, 0 otherwise

Macros

e(cmd)	gsem
e(cmd2)	fmm
e(cmdline)	command as typed
e(prefix)	fmm
e(depvar)	names of dependent variables
e(eqnames)	names of equations
e(wtype)	weight type
e(wexp)	weight expression
e(title)	title in estimation output
e(clustvar)	name of cluster variable
e(model#)	model for the #th component
e(offset#)	offset for the #th *depvar*
e(vce)	*vcetype* specified in vce()
e(vcetype)	title used to label Std. err.
e(opt)	type of optimization
e(which)	max or min; whether optimizer is to perform maximization or minimization
e(method)	estimation method: ml
e(ml_method)	type of ml method
e(user)	name of likelihood-evaluator program
e(technique)	maximization technique
e(properties)	b V
e(estat_cmd)	program used to implement estat
e(predict)	program used to implement predict
e(covariates)	list of covariates
e(lclass)	name of latent class variable
e(marginsnotok)	predictions not allowed by margins
e(marginsdefault)	default predict() specification for margins
e(footnote)	program used to implement the footnote display
e(asbalanced)	factor variables fvset as asbalanced
e(asobserved)	factor variables fvset as asobserved

Matrices

e(b)	parameter vector
e(b_pclass)	parameter class
e(cat#)	categories for the #th *depvar*, ordinal
e(out#)	outcomes for the #th *depvar*, mlogit
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(V)	covariance matrix of the estimators
e(V_modelbased)	model-based variance
e(lclass_k_levels)	number of levels for latent class variables

`e(lclass_bases)`	base levels for latent class variables
`e(_N)`	sample size for each component
Functions	
`e(sample)`	marks estimation sample

In addition to the above, the following is stored in `r()`:

Matrices	
`r(table)`	matrix containing the coefficients with their standard errors, test statistics, p-values, and confidence intervals

Note that results stored in `r()` are updated when the command is replayed and will be replaced when any r-class command is run after the estimation command.

Methods and formulas

Methods and formulas are presented under the following headings:

The likelihood
The EM algorithm
Survey data
Predictions

The likelihood

`fmm` fits finite mixture models via maximum likelihood estimation. The likelihood for the specified model is derived under the assumption that, within a given latent class, each response variable is independent and identically distributed across the estimation sample. These assumptions are conditional on the latent classes and the observed exogenous variables.

The likelihood is computed by combining the conditional likelihoods from each latent class weighted by the associated latent-class probabilities. Let $\boldsymbol{\theta}$ be the vector of model parameters. For a given observation, let $\boldsymbol{y}$ be the vector of observed response variables, and $\boldsymbol{x}$ be the vector of independent variables. Let C be the categorical latent variable with g latent classes $1,\dots,g$. The marginal likelihood for a given observation looks something like

$$\mathcal{L}_C(\boldsymbol{\theta}) = \sum_{i=1}^{g} \pi_i f_i(\boldsymbol{y}|\boldsymbol{x}, c_i = 1, \boldsymbol{\theta})$$

where π_i is the probability for the ith latent class, $f_i(\cdot)$ is the conditional probability density function for the observed response variables in the ith latent class, and $\boldsymbol{c}' = (c_1,\dots,c_g)$ is the vector of latent class indicators. When $c_i = 1$, all other elements of $\boldsymbol{c}$ are zero. All auxiliary parameters are fit directly without any further parameterization, so we simply acknowledge that the auxiliary parameters are among the elements of $\boldsymbol{\theta}$.

The $\boldsymbol{y}$ variables are assumed to be independent, conditional on $\boldsymbol{x}$ and C, so $f_i(\cdot)$ is the product of the individual conditional densities. One exception to this is when $\boldsymbol{y}$ contains the outcome and endogenous covariates for `ivregress`, in which case the Gaussian responses are actually modeled using a multivariate normal density to allow for correlated errors. This one exception does not meaningfully change the following discussion, so we make no effort to represent this distinction in the formulas.

For the ith latent class with n response variables, the conditional joint density function for a given observation is

$$f_i(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}) = \prod_{j=1}^{n} f_{ij}(y_{ij}|\boldsymbol{x}, \boldsymbol{\theta})$$

All estimation commands supported by fmm model the dependence of y_{ij} on $\boldsymbol{x}$ through the linear prediction

$$z_{ij} = \boldsymbol{x}'\boldsymbol{\beta}_{ij}$$

where $\boldsymbol{\beta}_{ij}$ is the vector of the coefficients for y_{ij}. For notational convenience, we will overload the definitions of $f_i(\cdot)$ and $f_{ij}(\cdot)$ so that they are functions of the responses and model parameters through the linear predictions $\boldsymbol{z}_i' = (z_{i1}, \ldots, z_{in})$. Thus $f_i(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta})$ is equivalently specified as $f_i(\boldsymbol{y}, \boldsymbol{z}_i, \boldsymbol{\theta})$, and $f_{ij}(y_{ij}|\boldsymbol{x}, \boldsymbol{\theta})$ is equivalently specified as $f_{ij}(y_{ij}, z_{ij}, \boldsymbol{\theta})$. In this new notation, the likelihood for a given observation is

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{g} \pi_i \prod_{j=1}^{n} f_{ij}(y_{ij}, z_{ij}, \boldsymbol{\theta}) \tag{1}$$

fmm uses the multinomial logistic distribution to model the probabilities for the latent classes. For the ith latent class, the probability is given by

$$\pi_i = \Pr(c_i = 1|\boldsymbol{x}) = \frac{\exp(z_i)}{\sum_{j=1}^{g} \exp(z_j)}$$

where the linear prediction for the ith latent class is

$$z_i = \boldsymbol{x}'\boldsymbol{\gamma}_i$$

and $\boldsymbol{\gamma}_i$ is the associated vector of coefficients. If the first latent class is the base level, $\boldsymbol{\gamma}_1$ is a vector of zeros so that $z_1 = 0$ and $\exp(z_1) = 1$.

The vector $\boldsymbol{\theta}$ is therefore the set of unique model parameters taken from the following:

$\boldsymbol{\gamma}_i$ is the vector of coefficients for the ith latent class.

$\boldsymbol{\beta}_{ij}$ is the vector of coefficients for y_{ij}.

Auxiliary parameters are parameters that result from some of the distribution families.

Each latent class will have its own set of these parameters.

The EM algorithm

fmm uses the EM algorithm to refine starting values before maximizing the likelihood in (1).

The EM algorithm uses the complete-data likelihood, a likelihood where it is as if we have observed values for the latent class indicator variables $\boldsymbol{c}$. In the complete-data case, the likelihood for a given observation is

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{g} \{\pi_i f_i(\boldsymbol{y}, \boldsymbol{z}_i, \boldsymbol{\theta})\}^{c_i}$$

so the complete-data log likelihood is

$$\log L(\boldsymbol{\theta}) = \sum_{i=1}^{g} c_i \{\log \pi_i + \log f_i(\boldsymbol{y}, \boldsymbol{z}_i, \boldsymbol{\theta})\}$$

We intend to maximize the expected complete-data log likelihood given the observed variables $\boldsymbol{y}$ and $\boldsymbol{x}$. This is an iterative process in which we use the kth guess of the model parameters, denoted $\boldsymbol{\theta}_{(k)}$, then compute the next guess, $\boldsymbol{\theta}_{(k+1)}$.

In the expectation (E) step, we derive the functional form of the expected complete-data log likelihood. The complete-data log likelihood is a linear function of the latent class indicator variables, so

$$E(c_i|\boldsymbol{y},\boldsymbol{x},\boldsymbol{\theta}_{(k)}) = \frac{\pi_i f_i(\boldsymbol{y},\boldsymbol{z}_i,\boldsymbol{\theta}_{(k)})}{\sum_{j=1}^{g} \pi_j f_j(\boldsymbol{y},\boldsymbol{z}_j,\boldsymbol{\theta}_{(k)})}$$

We denote this posterior probability by p_i, so the expected complete-data log likelihood for a given observation is given by

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}_{(k)}) = \sum_{i=1}^{g} p_i \left\{ \log \pi_i + \log f_i(\boldsymbol{y},\boldsymbol{z}_i,\boldsymbol{\theta}) \right\}$$

Note that $Q(\boldsymbol{\theta}|\boldsymbol{\theta}_{(k)})$ is a function of $\boldsymbol{\theta}_{(k)}$ solely through the posterior probabilities p_i.

Now that we have the conditional complete-data log likelihood, the maximization (M) step is to maximize $Q(\boldsymbol{\theta}|\boldsymbol{\theta}_{(k)})$ with respect to $\boldsymbol{\theta}$ to find $\boldsymbol{\theta}_{(k+1)}$.

Survey data

fmm supports estimation with survey data. However, only the linearized variance estimator is supported. For details on VCEs with survey data, see [SVY] **Variance estimation**.

Predictions

The predicted mean for a given response within a latent class is computed in the standard way. For example, the predicted mean for regress is the linear prediction, the predicted mean for glm is computed by applying the link function to the linear prediction, and for ologit, the predicted mean for a given response level is the predicted probability for that level. For survival outcomes, the formulas for predicted means (expected values) are provided in the *Survival distributions* section in [SEM] **Methods and formulas for gsem**.

Let $\hat{z}_i$ be the linear prediction for the ith latent class. The predicted probability for the ith latent class is then given by

$$\hat{\pi}_i = \frac{\exp(\hat{z}_i)}{\sum_{j=1}^{g} \exp(\hat{z}_j)}$$

The predicted posterior probability for the ith latent class is given by

$$\tilde{\pi}_i = \frac{\hat{\pi}_i f_i(\boldsymbol{y},\hat{\boldsymbol{z}}_i,\hat{\boldsymbol{\theta}})}{\sum_{j=1}^{g} \hat{\pi}_j f_j(\boldsymbol{y},\hat{\boldsymbol{z}}_j,\hat{\boldsymbol{\theta}})}$$

Let $\hat{\mu}_i$ be the predicted mean of response y in the ith latent class. The predicted overall mean of y, using the fitted latent class probabilities, is given by

$$\hat{\mu} = \sum_{i=1}^{g} \hat{\pi}_i \hat{\mu}_i$$

The predicted overall mean of y, using the posterior latent class probabilities, is given by

$$\tilde{\mu} = \sum_{i=1}^{g} \tilde{\pi}_i \hat{\mu}_i$$

Also see

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm estimation** — Fitting finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[SVY] **svy estimation** — Estimation commands for survey data

Description Quick start Menu Syntax Remarks and examples
Stored results Methods and formulas Reference Also see

Description

`fmm: betareg` fits mixtures of beta regression models to a fractional outcome whose values are greater than 0 and less than 1; see [FMM] **fmm** and [R] **betareg** for details.

Quick start

Mixture of two beta distributions of `y`

```
fmm 2: betareg y
```

Mixture of two beta regression models of `y` on `x1` and `x2`

```
fmm 2: betareg y x1 x2
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): betareg y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): betareg y x1 x2
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): betareg y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Beta regression

Syntax

Basic syntax

`fmm #: betareg` *depvar* [*indepvars*] [, *options*]

Full syntax

`fmm #` [*if*] [*in*] [*weight*] [, *fmmopts*]`: betareg` *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`noconstant`	suppress the constant term
`link(`*linkname*`)`	specify link function for the conditional mean; default is `link(logit)`

indepvars may contain factor variables; see [U] **11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see [U] **11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **betareg**.

linkname	Description
`logit`	logit
`probit`	probit
`cloglog`	complementary log–log

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about beta regression, see [R] **betareg**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Reference

Gray, L. A., and M. Hernández-Alava. 2018. A command for fitting mixture regression models for bounded dependent variables using the beta distribution. *Stata Journal* 18: 51–75.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **betareg** — Beta regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: cloglog — Finite mixtures of complementary log–log regression models

Description

`fmm: cloglog` fits mixtures of complementary log–log regression models; see [FMM] **fmm** and [R] **cloglog** for details.

Quick start

Mixture of two complementary log–log regression models of `y` on `x1` and `x2`

```
fmm 2: cloglog y x1 x2
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): cloglog y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): cloglog y x1 x2
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): cloglog y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Binary outcomes > Complementary log–log regression

Syntax

Basic syntax

`fmm` # : `cloglog` *depvar* [*indepvars*] [, *options*]

Full syntax

`fmm` # [*if*] [*in*] [*weight*] [, *fmmopts*] : `cloglog` *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`noconstant`	suppress the constant term
`offset(`*varname*`)`	include *varname* in model with coefficient constrained to 1
`asis`	retain perfect predictor variables

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **cloglog**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about complementary log–log regression, see [R] **cloglog**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **cloglog** — Complementary log–log regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: glm — Finite mixtures of generalized linear regression models

Description

fmm: glm fits mixtures of generalized linear regression models; see [FMM] **fmm** and [R] **glm** for details.

Quick start

Mixture of two normal distributions of y

```
fmm 2: glm y, family(gaussian) link(identity)
```

Mixture of two gamma distributions of y

```
fmm 2: glm y, family(gamma)
```

Mixture of two gamma regression models of y on x1 and x2

```
fmm 2: glm y x1 x2, family(gamma)
```

Same as above, but with class probabilities depending on z1 and z2

```
fmm 2, lcprob(z1 z2): glm y x1 x2, family(gamma)
```

With robust standard errors

```
fmm 2, vce(robust): glm y x1 x2, family(gamma)
```

Constrain coefficients on x1 and x2 to be equal across classes

```
fmm 2, lcinvariant(coef): glm y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Generalized linear model (GLM)

Syntax

Basic syntax

fmm # : glm *depvar* [*indepvars*] [, *options*]

Full syntax

fmm # [*if*] [*in*] [*weight*] [, *fmmopts*] : glm *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
family(*familyname*)	distribution of *depvar*; default is family(gaussian)
link(*linkname*)	link function; default varies per family
noconstant	suppress the constant term
exposure($varname_e$)	include $\ln(varname_e)$ in model with coefficient constrained to 1
offset($varname_o$)	include $varname_o$ in model with coefficient constrained to 1
asis	retain perfect predictor variables

indepvars may contain factor variables; see [U] **11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see [U] **11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **glm**.

familyname	Description
gaussian	Gaussian (normal); the default
bernoulli	Bernoulli
beta	beta
binomial [# | *varname*]	binomial; default number of binomial trials is 1
poisson	Poisson
nbinomial [mean | constant]	negative binomial; default dispersion is mean
exponential	exponential
gamma	gamma
lognormal	lognormal
loglogistic	loglogistic
weibull	Weibull

bernoulli, beta, exponential, lognormal, loglogstic, and weibull are extensions available with fmm: glm that are not available with glm.

linkname	Description
identity	identity
log	log
logit	logit
probit	probit
cloglog	complementary log–log

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about generalized linear regression, see [R] **glm**. For examples using `fmm`, see examples in *Contents*.

If you specify both `family()` and `link()`, not all combinations make sense. You may choose from the following combinations:

	identity	log	logit	probit	cloglog
Gaussian	D	x			
Bernoulli			D	x	x
beta			D	x	x
binomial			D	x	x
Poisson		D			
negative binomial		D			
exponential		D			
gamma		D			
lognormal		D			
loglogistic		D			
Weibull		D			

D denotes the default.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **glm** — Generalized linear models

[SEM] **gsem** — Generalized structural equation model estimation command

[SVY] **svy estimation** — Estimation commands for survey data

fmm: intreg — Finite mixtures of interval regression models

Description

fmm: intreg fits mixtures of interval regression models; see [FMM] **fmm** and [R] **intreg** for details.

Quick start

Mixture of two interval regressions on x1 of the interval-measured dependent variable with lower endpoint y_lower and upper endpoint y_upper

```
fmm 2: intreg y_lower y_upper x1
```

Same as above, but with class probabilities depending on z1 and z2

```
fmm 2, lcprob(z1 z2): intreg y_lower y_upper x1
```

With robust standard errors

```
fmm 2, vce(robust): intreg y_lower y_upper x1
```

Constrain coefficients on x1 to be equal across classes

```
fmm 2, lcinvariant(coef): intreg y_lower y_upper x1
```

Menu

Statistics > FMM (finite mixture models) > Continuous outcomes > Interval regression

Syntax

Basic syntax

`fmm #: intreg` *depvar*$_{\text{lower}}$ *depvar*$_{\text{upper}}$ [*indepvars*] [, *options*]

Full syntax

`fmm #` [*if*] [*in*] [*weight*] [, *fmmopts*] :

`intreg` *depvar*$_{\text{lower}}$ *depvar*$_{\text{upper}}$ [*indepvars*] [, *options*]

where # specifies the number of class models.

The values in *depvar*$_{\text{lower}}$ and *depvar*$_{\text{upper}}$ should have the following form:

Type of data		*depvar*$_{\text{lower}}$	*depvar*$_{\text{upper}}$
point data	$a = [a, a]$	a	a
interval data	$[a, b]$	a	b
left-censored data	$(-\infty, b]$	.	b
right-censored data	$[a, +\infty)$	a	.
missing		.	.

options	Description
Model	
`noconstant`	suppress the constant term
`offset(`*varname*`)`	include *varname* in model with coefficient constrained to 1

indepvars may contain factor variables; see [U] **11.4.3 Factor variables**.

depvar$_{\text{lower}}$, *depvar*$_{\text{upper}}$, and *indepvars* may contain time-series operators; see [U] **11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **intreg**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about interval regression, see [R] **intreg**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **intreg** — Interval regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: ivregress — Finite mixtures of linear regression models with endogenous covariates

Description | Quick start | Menu | Syntax
Remarks and examples | Stored results | Methods and formulas | Also see

Description

`fmm: ivregress` fits mixtures of linear regression models with endogenous covariates; see [FMM] **fmm** and [R] **ivregress** for details.

Quick start

Mixture of two linear regressions of `y1` on `x1` with endogenous regressor `y2` that is instrumented by `w1`

```
fmm 2: ivregress y1 x1 (y2 = w1)
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): ivregress y1 x1 (y2 = w1)
```

With robust standard errors

```
fmm 2, vce(robust): ivregress y1 x1 (y2 = w1)
```

Constrain coefficients on `x1`, `w1`, and `y2` to be equal across classes

```
fmm 2, lcinvariant(coef): ivregress y1 x1 (y2 = w1)
```

Menu

Statistics > FMM (finite mixture models) > Continuous outcomes > Linear regression with endogenous covariates

Syntax

Basic syntax

fmm # : ivregress *depvar* [*varlist*$_1$] (*varlist*$_2$ = *varlist_iv*) [, *options*]

Full syntax

fmm # [*if*] [*in*] [*weight*] [, *fmmopts*] :

ivregress *depvar* [*varlist*$_1$] (*varlist*$_2$ = *varlist_iv*) [, *options*]

where # specifies the number of class models.

options	Description
Model	
noconstant	suppress the constant term

varlist$_1$ and *varlist_iv* may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar, *varlist*$_1$, and *varlist_iv* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **ivregress**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about linear regression with endogenous covariates, see [R] **ivregress**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **ivregress** — Single-equation instrumental-variables regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: logit — Finite mixtures of logistic regression models

Description

`fmm: logit` fits mixtures of logistic regression models; see [FMM] **fmm** and [R] **logit** for details.

Quick start

Mixture of two logistic regression models of `y` on `x1` and `x2`

```
fmm 2: logit y x1 x2
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): logit y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): logit y x1 x2
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): logit y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Binary outcomes > Logistic regression

Syntax

Basic syntax

fmm # : logit *depvar* [*indepvars*] [, *options*]

Full syntax

fmm # [*if*] [*in*] [*weight*] [, *fmmopts*] : logit *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
noconstant	suppress the constant term
offset(*varname*)	include *varname* in model with coefficient constrained to 1
asis	retain perfect predictor variables

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **logit**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about logistic regression, see [R] **logit**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **logit** — Logistic regression, reporting coefficients

[SVY] **svy estimation** — Estimation commands for survey data

fmm: mlogit — Finite mixtures of multinomial (polytomous) logistic regression models

Description	Quick start	Menu	Syntax
Remarks and examples	Stored results	Methods and formulas	Also see

Description

`fmm: mlogit` fits mixtures of multinomial logistic regression models; see [FMM] **fmm** and [R] **mlogit** for details.

Quick start

Mixture of two `mlogit` distributions of `y`

```
fmm 2: mlogit y
```

Mixture of two `mlogit` models of `y` on `x1` and `x2`

```
fmm 2: mlogit y x1 x2
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): mlogit y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): mlogit y x1 x2
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): mlogit y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Multinomial logistic regression

Syntax

Basic syntax `fmm # : mlogit` *depvar* [*indepvars*] [, *options*]

Full syntax

`fmm #` [*if*] [*in*] [*weight*] [, *fmmopts*] `: mlogit` *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`noconstant`	suppress the constant term
`baseoutcome(#)`	value of *depvar* that will be the base outcome

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **mlogit**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about multinomial logistic regression, see [R] **mlogit**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **mlogit** — Multinomial (polytomous) logistic regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: nbreg — Finite mixtures of negative binomial regression models

Description

`fmm: nbreg` fits mixtures of negative binomial regression models; see [FMM] **fmm** and [R] **nbreg** for details.

Quick start

Mixture of two negative binomial distributions of `y`

```
fmm 2: nbreg y
```

Mixture of two negative binomial regression models of `y` on `x1` and `x2`

```
fmm 2: nbreg y x1 x2
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): nbreg y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): nbreg y x1 x2
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): nbreg y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Count outcomes > Negative binomial regression

Syntax

Basic syntax

`fmm #: nbreg` *depvar* [*indepvars*] [, *options*]

Full syntax

`fmm #` [*if*] [*in*] [*weight*] [, *fmmopts*] `: nbreg` *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`noconstant`	suppress the constant term
`dispersion(mean)`	parameterization of dispersion; the default
`dispersion(constant)`	constant dispersion for all observations
`exposure(`$varname_e$`)`	include $\ln(varname_e)$ in model with coefficient constrained to 1
`offset(`$varname_o$`)`	include $varname_o$ in model with coefficient constrained to 1

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options for nbreg* in [R] **nbreg**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see [U] **11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see [U] **11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see [U] **11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about negative binomial regression, see [R] **nbreg**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **nbreg** — Negative binomial regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: ologit — Finite mixtures of ordered logistic regression models

Description | Quick start | Menu | Syntax
Remarks and examples | Stored results | Methods and formulas | Also see

Description

`fmm: ologit` fits mixtures of ordered logistic regression models; see [FMM] **fmm** and [R] **ologit** for details.

Quick start

Mixture of two ordered logistic regression models of `y` on `x1` and `x2`

```
fmm 2: ologit y x1 x2
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): ologit y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): ologit y x1 x2
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): ologit y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Ordinal outcomes > Ordered logistic regression

Syntax

Basic syntax

`fmm #: ologit` *depvar* [*indepvars*] [, *options*]

Full syntax

`fmm #` [*if*] [*in*] [*weight*] [, *fmmopts*]`: ologit` *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`offset(`*varname*`)`	include *varname* in model with coefficient constrained to 1

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **ologit**.

fmmopts	Description
Model	
`lcinvariant(`*pclassname*`)`	specify parameters that are equal across classes; default is `lcinvariant(none)`
`lcprob(`*varlist*`)`	specify independent variables for class probabilities
`lclabel(`*name*`)`	name of the categorical latent variable; default is `lclabel(Class)`
`lcbase(#)`	base latent class
`constraints(`*constraints*`)`	apply specified linear constraints
SE/Robust	
`vce(`*vcetype*`)`	*vcetype* may be `oim`, `opg`, `robust`, or `cluster` *clustvar*
Reporting	
`level(#)`	set confidence level; default is `level(95)`
`nocnsreport`	do not display constraints
`noheader`	do not display header above parameter table
`nodvheader`	do not display dependent variables information in the header
`notable`	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
`startvalues(`*svmethod*`)`	method for obtaining starting values; default is `startvalues(factor)`
`emopts(`*maxopts*`)`	control EM algorithm for improved starting values
`noestimate`	do not fit the model; show starting values instead
`collinear`	keep collinear variables
`coeflegend`	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

`by`, `collect`, `statsby`, and `svy` are allowed; see **[U] 11.1.10 Prefix commands**.

`vce()` and weights are not allowed with the `svy` prefix; see [SVY] **svy**.

`fweight`s, `iweight`s, and `pweight`s are allowed; see **[U] 11.1.6 weight**.

`collinear` and `coeflegend` do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
`cons`	intercepts and cutpoints
`coef`	fixed coefficients
`errvar`	covariances of errors
`scale`	scaling parameters
`all`	all the above
`none`	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about ordered logistic regression, see [R] **ologit**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **ologit** — Ordered logistic regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: oprobit — Finite mixtures of ordered probit regression models

Description

`fmm: oprobit` fits mixtures of ordered probit regression models; see [FMM] **fmm** and [R] **oprobit** for details.

Quick start

Mixture of two ordered probit regression models of `y` on `x1` and `x2`

```
fmm 2: oprobit y x1 x2
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): oprobit y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): oprobit y x1 x2
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): oprobit y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Ordinal outcomes > Ordered probit regression

Syntax

Basic syntax

`fmm #: oprobit` *depvar* [*indepvars*] [, *options*]

Full syntax

`fmm #` [*if*] [*in*] [*weight*] [, *fmmopts*] `: oprobit` *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`offset(`*varname*`)`	include *varname* in model with coefficient constrained to 1

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **oprobit**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about ordered probit regression, see [R] **oprobit**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **oprobit** — Ordered probit regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: pointmass — Finite mixtures models with a density mass at a single point

Description

`fmm: pointmass` is a degenerate distribution that takes on a single integer value with probability one. This distribution cannot be used by itself and is always combined with other `fmm` distributions, often to model zero-inflated outcomes.

Quick start

Zero-inflated Poisson regression of `y` on `x1` and `x2`

```
fmm : (pointmass y) (poisson y x1 x2)
```

Same as above, but add predictors `w1` and `w2` to model the pointmass class probability

```
fmm : (pointmass y, lcprob(w1 w2)) (poisson y x1 x2)
```

Ordered logistic regression of `y` on `x1` and `x2` with inflation at 1

```
fmm : (pointmass y, value(1)) (ologit y x1 x2)
```

Menu

Statistics > FMM (finite mixture models) > General estimation and regression

Syntax

fmm [*if*] [*in*] [*weight*] [, *fmmopts*] : (pointmass *depvar* [, *options*])
(*component*$_1$) [(*component*$_2$) ...]

component is defined in [FMM] **fmm**.

options	Description
lcprob(*varlist*)	specify independent variables for class probability
value(#)	integer-valued location of the point mass

depvar may contain time-series operators; see [U] **11.4.4 Time-series varlists**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see [U] **11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see [U] **11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see [U] **11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Options

lcprob(*varlist*) specifies that the linear prediction for belonging to the point mass component includes the variables in *varlist*. lcinvariant() has no effect on these parameters.

value(#) specifies the value of *depvar* at which the latent class has a singular point mass. The default is value(0). Only integer values are allowed for #.

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. See [FMM] **Example 3** where pointmass is used to fit a zero-inflated Poisson model. See [FMM] **Example 4** where pointmass is used to fit a mixture cure model to survival data. Other examples are available; see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Example 3** — Zero-inflated models

[FMM] **Example 4** — Mixture cure models for survival data

[FMM] **Glossary**

[R] **zinb** — Zero-inflated negative binomial regression

[R] **zioprobit** — Zero-inflated ordered probit regression

[R] **zip** — Zero-inflated Poisson regression

[SVY] **svy estimation** — Estimation commands for survey data

Description

fmm: poisson fits mixtures of Poisson regression models; see [FMM] **fmm** and [R] **poisson** for details.

Quick start

Mixture of two Poisson distributions of y

```
fmm 2: poisson y
```

Mixture of two Poisson regression models of y on x1 and x2

```
fmm 2: poisson y x1 x2
```

Same as above, but with class probabilities depending on z1 and z2

```
fmm 2, lcprob(z1 z2): poisson y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): poisson y x1 x2
```

Constrain coefficients on x1 and x2 to be equal across classes

```
fmm 2, lcinvariant(coef): poisson y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Count outcomes > Poisson regression

Syntax

Basic syntax

`fmm #: poisson` *depvar* [*indepvars*] [, *options*]

Full syntax

`fmm #` [*if*] [*in*] [*weight*] [, *fmmopts*] `: poisson` *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`noconstant`	suppress the constant term
`exposure(`*varname*$_e$`)`	include $\ln(varname_e)$ in model with coefficient constrained to 1
`offset(`*varname*$_o$`)`	include *varname*$_o$ in model with coefficient constrained to 1

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see Options in [R] **poisson**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about Poisson regression, see [R] **poisson**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Example 2** — Mixture of Poisson regression models

[FMM] **Example 3** — Zero-inflated models

[FMM] **Glossary**

[R] **poisson** — Poisson regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: probit — Finite mixtures of probit regression models

Description

`fmm: probit` fits mixtures of probit regression models; see [FMM] **fmm** and [R] **probit** for details.

Quick start

Mixture of two probit regression models of `y` on `x1` and `x2`

```
fmm 2: probit y x1 x2
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): probit y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): probit y x1 x2
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): probit y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Binary outcomes > Probit regression

Syntax

Basic syntax

`fmm` # : `probit` *depvar* [*indepvars*] [, *options*]

Full syntax

`fmm` # [*if*] [*in*] [*weight*] [, *fmmopts*] : `probit` *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`noconstant`	suppress the constant term
`offset(`*varname*`)`	include *varname* in model with coefficient constrained to 1
`asis`	retain perfect predictor variables

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **probit**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about probit regression, see [R] **probit**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **probit** — Probit regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: regress — Finite mixtures of linear regression models

Description | Quick start | Menu | Syntax
Remarks and examples | Stored results | Methods and formulas | Also see

Description

`fmm: regress` fits mixtures of linear regression models; see [FMM] **fmm** and [R] **regress** for details.

Quick start

Mixture of two normal distributions of `y`

```
fmm 2: regress y
```

Mixture of seven normal distributions of `y` with variances constrained to be equal

```
fmm 7, lcinvariant(errvar): regress y
```

Mixture of two linear regression models of `y` on `x1` and `x2`

```
fmm 2: regress y x1 x2
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): regress y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): regress y x1 x2
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): regress y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Continuous outcomes > Linear regression

Syntax

Basic syntax

`fmm` # : `regress` *depvar* [*indepvars*] [, *options*]

Full syntax

`fmm` # [*if*] [*in*] [*weight*] [, *fmmopts*] : `regress` *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`noconstant`	suppress the constant term

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **regress**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see [U] **11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see [U] **11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see [U] **11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about linear regression, see [R] **regress**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Example 1a** — Mixture of linear regression models

[FMM] **Example 1b** — Covariates for class membership

[FMM] **Example 1c** — Testing coefficients across class models

[FMM] **Example 1d** — Component-specific covariates

[FMM] **Glossary**

[R] **regress** — Linear regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: streg — Finite mixtures of parametric survival models

Description

`fmm: streg` fits mixtures of parametric survival regression models; see [FMM] **fmm** and [ST] **streg** for details.

Quick start

Mixture of two Weibull distributions using `stset` data

```
fmm 2: streg, distribution(weibull)
```

Mixture of two exponential distributions

```
fmm 2: streg, distribution(exponential)
```

Mixture of two Weibull survival models with covariates `x1` and `x2`

```
fmm 2: streg y x1 x2, distribution(weibull)
```

Same as above, but with class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): streg y x1 x2, distribution(weibull)
```

With robust standard errors

```
fmm 2, vce(robust): streg y x1 x2, distribution(weibull)
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): streg y x1 x2, distribution(weibull)
```

Menu

Statistics > FMM (finite mixture models) > Parametric survival regression

Syntax

Basic syntax

`fmm #: streg` [*indepvars*] [, *options*]

Full syntax

`fmm #` [*if*] [*in*] [*weight*] [, *fmmopts*] `: streg` [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`noconstant`	suppress the constant term
* `distribution(`*distname*`)`	specify survival distribution
`time`	use accelerated failure-time metric
`offset(`*varname*`)`	include *varname* in model with coefficient constrained to 1

*`distribution(`*distname*`)` is required.

You must `stset` your data before using `fmm: streg`; see [ST] **stset**.

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [ST] **streg**.

distname	Description
`exponential`	exponential survival distribution
`loglogistic`	loglogistic survival distribution
`llogistic`	synonym for `loglogistic`
`weibull`	Weibull survival distribution
`lognormal`	lognormal survival distribution
`lnormal`	synonym for `lognormal`
* `gamma`	gamma survival distribution

*`fmm: streg` uses the gamma survival distribution and not the generalized gamma distribution that is used by `streg`.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about parametric survival models, see [ST] **streg**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Example 4** — Mixture cure models for survival data

[FMM] **Glossary**

[ST] **streg** — Parametric survival models

[ST] **stset** — Declare data to be survival-time data

[SVY] **svy estimation** — Estimation commands for survey data

fmm: tobit — Finite mixtures of tobit regression models

Description

`fmm: tobit` fits mixtures of tobit regression models; see [FMM] **fmm** and [R] **tobit** for details.

Quick start

Mixture of two tobit regression models of `y` on `x1` and `x2` where `y` is censored at the minimum of `y`

```
fmm 2: tobit y x1 x2, ll
```

Same as above, but where the lower-censoring limit is zero

```
fmm 2: tobit y x1 x2, ll(0)
```

Same as above, but where `lower` and `upper` are variables containing the censoring limits

```
fmm 2: tobit y x1 x2, ll(lower) ul(upper)
```

With class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): tobit y x1 x2, ll
```

With robust standard errors

```
fmm 2, vce(robust): tobit y x1 x2, ll
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): tobit y x1 x2, ll
```

Menu

Statistics > FMM (finite mixture models) > Continuous outcomes > Tobit regression

Syntax

Basic syntax

`fmm #: tobit` *depvar* [*indepvars*] [, *options*]

Full syntax

`fmm #` [*if*] [*in*] [*weight*] [, *fmmopts*] `: tobit` *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
`noconstant`	suppress the constant term
`ll`[`(`*varname* | #`)`]	left-censoring variable or limit
`ul`[`(`*varname* | #`)`]	right-censoring variable or limit
`offset(`*varname*`)`	include *varname* in model with coefficient constrained to 1

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in [R] **tobit**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about tobit regression, see [R] **tobit**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **tobit** — Tobit regression

[SVY] **svy estimation** — Estimation commands for survey data

Description Quick start Menu Syntax
Remarks and examples Stored results Methods and formulas Also see

Description

`fmm: tpoisson` fits mixtures of truncated Poisson regression models; see [FMM] **fmm** and [R] **tpoisson** for details.

Quick start

Mixture of two truncated Poisson distributions with default truncation point at 0

```
fmm 2: tpoisson y
```

Mixture of two truncated Poisson regression models of `y` on `x1` and `x2` with truncation at 0

```
fmm 2: tpoisson y x1 x2
```

Same as above, but with truncation at 3

```
fmm 2: tpoisson y x1 x2, ll(3)
```

With class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): tpoisson y x1 x2
```

With robust standard errors

```
fmm 2, vce(robust): tpoisson y x1 x2
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): tpoisson y x1 x2
```

Menu

Statistics > FMM (finite mixture models) > Count outcomes > Truncated Poisson regression

Syntax

Basic syntax

fmm # : tpoisson *depvar* [*indepvars*] [, *options*]

Full syntax

fmm # [*if*] [*in*] [*weight*] [, *fmmopts*] : tpoisson *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
noconstant	suppress the constant term
ll(*varname* | #)	truncation point; default value is ll(0), zero truncation
exposure(*varname*$_e$)	include ln(*varname*$_e$) in model with coefficient constrained to 1
offset(*varname*$_o$)	include *varname*$_o$ in model with coefficient constrained to 1

indepvars may contain factor variables; see [U] 11.4.3 Factor variables.

depvar and *indepvars* may contain time-series operators; see [U] 11.4.4 Time-series varlists.

For a detailed description of *options*, see Options in [R] tpoisson.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see **[U] 11.4.3 Factor variables**.

by, collect, statsby, and svy are allowed; see **[U] 11.1.10 Prefix commands**.

vce() and weights are not allowed with the svy prefix; see [SVY] **svy**.

fweights, iweights, and pweights are allowed; see **[U] 11.1.6 weight**.

collinear and coeflegend do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] **fmm**.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about truncated Poisson regression, see [R] **tpoisson**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **tpoisson** — Truncated Poisson regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm: truncreg — Finite mixtures of truncated linear regression models

Description

`fmm: truncreg` fits mixtures of truncated linear regression models; see [FMM] **fmm** and [R] **truncreg** for details.

Quick start

Mixture of two truncated normal distributions of `y` with truncation from below at 0

```
fmm 2: truncreg y, ll(0)
```

Mixture of two truncated regression models of `y` on `x1` and `x2` with truncation from below at 0

```
fmm 2: truncreg y x1 x2, ll(0)
```

Same as above, but where `lower` is a variable containing the truncation point for each observation

```
fmm 2: truncreg y x1 x2, ll(lower)
```

With class probabilities depending on `z1` and `z2`

```
fmm 2, lcprob(z1 z2): truncreg y x1 x2, ll(0)
```

With robust standard errors

```
fmm 2, vce(robust): truncreg y x1 x2, ll(0)
```

Constrain coefficients on `x1` and `x2` to be equal across classes

```
fmm 2, lcinvariant(coef): truncreg y x1 x2, ll(0)
```

Menu

Statistics > FMM (finite mixture models) > Continuous outcomes > Truncated regression

Syntax

Basic syntax

fmm # : truncreg *depvar* [*indepvars*] [, *options*]

Full syntax

fmm # [*if*] [*in*] [*weight*] [, *fmmopts*] : truncreg *depvar* [*indepvars*] [, *options*]

where # specifies the number of class models.

options	Description
Model	
noconstant	suppress the constant term
ll(*varname* | #)	left-truncation variable or limit
ul(*varname* | #)	right-truncation variable or limit
offset(*varname*)	include *varname* in model with coefficient constrained to 1

indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.

depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.

For a detailed description of *options*, see *Options* in **[R] truncreg**.

fmmopts	Description
Model	
lcinvariant(*pclassname*)	specify parameters that are equal across classes; default is lcinvariant(none)
lcprob(*varlist*)	specify independent variables for class probabilities
lclabel(*name*)	name of the categorical latent variable; default is lclabel(Class)
lcbase(#)	base latent class
constraints(*constraints*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim, opg, robust, or cluster *clustvar*
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
noheader	do not display header above parameter table
nodvheader	do not display dependent variables information in the header
notable	do not display parameter table
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process
startvalues(*svmethod*)	method for obtaining starting values; default is startvalues(factor)
emopts(*maxopts*)	control EM algorithm for improved starting values
noestimate	do not fit the model; show starting values instead
collinear	keep collinear variables
coeflegend	display legend instead of statistics

varlist may contain factor variables; see [U] 11.4.3 Factor variables.

by, collect, statsby, and svy are allowed; see [U] 11.1.10 Prefix commands.

vce() and weights are not allowed with the svy prefix; see [SVY] svy.

fweights, iweights, and pweights are allowed; see [U] 11.1.6 weight.

collinear and coeflegend do not appear in the dialog box.

See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

For a detailed description of *fmmopts*, see *Options* in [FMM] fmm.

pclassname	Description
cons	intercepts and cutpoints
coef	fixed coefficients
errvar	covariances of errors
scale	scaling parameters
all	all the above
none	none of the above; the default

Remarks and examples

For a general introduction to finite mixture models, see [FMM] **fmm intro**. For general information about truncated regression, see [R] **truncreg**. For examples using `fmm`, see examples in *Contents*.

Stored results

See *Stored results* in [FMM] **fmm**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

[FMM] **Glossary**

[R] **truncreg** — Truncated regression

[SVY] **svy estimation** — Estimation commands for survey data

fmm postestimation — Postestimation tools for fmm

Postestimation commands

The following postestimation commands are of special interest after estimation with `fmm`:

Command	Description
`estat eform`	display exponentiated parameters
`estat lcmean`	latent class marginal means
`estat lcprob`	latent class marginal probabilities
`lcstats`	latent class model-comparison statistics

The following standard postestimation commands are also available:

Command	Description
`contrast`	contrasts and linear hypothesis tests
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat` (svy)	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
* `hausman`	Hausman's specification test
`lincom`	linear combination of parameters
* `lrtest`	likelihood-ratio test
`margins`	marginal means, predictive margins, marginal effects, and average marginal effects
`marginsplot`	graph the results from margins (profile plots, interaction plots, etc.)
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	predictions, residuals, influence statistics, and other diagnostic measures
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`pwcompare`	pairwise comparisons of parameters
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

* `hausman` and `lrtest` are not appropriate with `svy` estimation results.

Postestimation commands such `lincom` and `nlcom` require referencing estimated parameter values, which are accessible via `_b[`*name*`]`. To find out what the names are, type `fmm, coeflegend`.

predict

Description for predict

`predict` after `fmm` creates new variables containing predictions such as means, probabilities, linear predictions, densities, or latent class probabilities.

Menu for predict

Statistics > Postestimation

Syntax for predict

`predict` [*type*] { *stub*`*` | *newvarlist* } [*if*] [*in*] [, *statistic options*]

`predict` [*type*] *stub*`*` [*if*] [*in*] , `scores`

statistic	Description
Main	
`mu`	expected value of *depvar*; the default
`eta`	linear prediction of *depvar*
`density`	density function at *depvar*
`distribution`	distribution function at *depvar*
`survival`	survivor function at *depvar*
`classpr`	latent class probability
`classposteriorpr`	posterior latent class probability

options	Description
Main	
`marginal`	compute *statistic* marginally with respect to the latent classes
`pmarginal`	compute `mu` marginally with respect to the posterior latent class probabilities
`nooffset`	make calculation ignoring offset or exposure
* `outcome(`*depvar* [#]`)`	specify observed response variable (default all)
`class(`#`)`	specify latent class (default all)

*`outcome(`*depvar* #`)` is allowed only if *depvar* is from `mlogit`, `ologit`, or `oprobit`.
`outcome(`*depvar* #`)` may also be specified as `outcome(`#.*depvar*`)` or `outcome(`*depvar* `##)`.
`outcome(`*depvar* `#3)` means the third outcome value. `outcome(`*depvar* `#3)` would mean the same as `outcome(`*depvar* `4)` if outcomes were 1, 3, and 4.

Options for predict

Main

`mu`, the default, calculates the expected value of the outcomes.

`eta` calculates the fitted linear prediction.

`density` calculates the density function. This prediction is computed using the current values of the observed variables, including the dependent variable.

`distribution` calculates the distribution function. This prediction is computed using the current values of the observed variables, including the dependent variable. This option is not allowed for `mlogit` outcomes.

`survival` calculates the survivor function. This prediction is computed using the current values of the observed variables, including the dependent variable. This option is allowed only for `streg` outcomes.

`classpr` calculates predicted probabilities for each latent class.

`classposteriorpr` calculates predicted posterior probabilities for each latent class. The posterior probabilities are a function of the latent-class predictors and the fitted outcome densities.

`marginal` specifies that the prediction be computed marginally with respect to the latent classes. The marginal prediction is computed by combining the class specific predictions using the latent-class probabilities.

This option is allowed only with `mu` and `density`.

`pmarginal` specifies that the prediction is computed by combining the class specific expected values using the posterior latent-class probabilities.

This option is allowed only with `mu`.

`nooffset` is relevant only if option `offset()` or `exposure()` was specified at estimation time. `nooffset` specifies that `offset()` or `exposure()` be ignored, which produces predictions as if all subjects had equal exposure.

`outcome(`*depvar* [#]`)` specifies the *depvar* for which predictions should be calculated. Predictions for all observed response variables are computed by default. Most models have only one *depvar*. If *depvar* is an `mlogit`, `ologit`, or `oprobit` outcome, then # optionally specifies which outcome level to predict. The default is the first level.

`class(`#`)` specifies that predictions for latent class # be calculated. Predictions for all latent classes are computed by default.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new variable list of length equal to the number of columns in `e(b)`. Otherwise, use *stub*`*` to have `predict` generate enumerated variables with prefix *stub*.

margins

Description for margins

margins estimates margins of response for outcome means, outcome probabilities, and latent-class probabilities.

Menu for margins

Statistics > Postestimation

Syntax for margins

margins [*marginlist*] [, *options*]

margins [*marginlist*] , predict(*statistic* ...) [predict(*statistic* ...) ...] [*options*]

statistic	Description
default	calculate expected values for each *depvar*
mu	calculate expected value of *depvar*
eta	calculate expected value of linear prediction of *depvar*
classpr	calculate latent class prior probabilities
density	not allowed with margins
distribution	not allowed with margins
survival	not allowed with margins
classposteriorpr	not allowed with margins
scores	not allowed with margins

mu defaults to the first *depvar* if option outcome() is not specified. If *depvar* is mlogit, ologit, or oprobit, the default is the first level of the outcome. The default is the first latent class if class() is not specified.

eta defaults to the first *depvar* if option outcome() is not specified. If *depvar* is mlogit, the default is the first level of the outcome.

classpr defaults to the first latent class if option class() is not specified.

predict's option marginal is assumed if predict's option class() is not specified.

Statistics not allowed with margins are functions of stochastic quantities other than e(b).

For the full syntax, see [R] **margins**.

Remarks and examples

For examples using estimates stats to compare models based on Akaike information criterion and Bayesian information criterion, see [FMM] **Example 1a**, [FMM] **Example 1b**, and [FMM] **Example 1d**.

For examples using estat lcprob to obtain marginal latent class probabilities and estat lcmean to obtain marginal predicted means, see [FMM] **Example 2** and [FMM] **Example 3**.

For examples using test and contrast to test equality of coefficients across classes, see [FMM] **Example 1c**.

For examples using predict, see [FMM] **Example 2**, [FMM] **Example 3**, and [FMM] **Example 4**.

Methods and formulas

See *Methods and formulas* in [FMM] **fmm**.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm estimation** — Fitting finite mixture models

[FMM] **fmm intro** — Introduction to finite mixture models

estat eform — Display exponentiated coefficients

Description | Menu for estat | Syntax | Options
Remarks and examples | Also see

Description

`fmm` reports coefficients. You can obtain exponentiated coefficients and their standard errors by using `estat eform` after estimation to redisplay results.

Menu for estat

Statistics > Postestimation

Syntax

`estat eform` [*eqnamelist*] [, `level(#)` *display_options*]

where *eqnamelist* is a list of equation names. With `fmm`, equation names correspond to the names of the response variables. If no *eqnamelist* is specified, exponentiated results for the first equation are shown.

Options

`level(#)`; see [R] **Estimation options**.

display_options control the display of factor variables and more. Allowed *display_options* are `noci`, `nopvalues`, `noomitted`, `vsquish`, `noemptycells`, `baselevels`, `allbaselevels`, `nofvlabel`, `fvwrap(#)`, `fvwrapon(`*style*`)`, `cformat(`*%fmt*`)`, `pformat(`*%fmt*`)`, `sformat(`*%fmt*`)`, and `nolstretch`. See [R] **Estimation options**.

Remarks and examples

For some commands that support the `fmm` prefix, exponentiated coefficients have a special meaning. Those special meanings are as follows:

Command	Meaning of exp(coef)
`logit`	odds ratio
`ologit`	odds ratio
`mlogit`	relative-risk ratio
`poisson`	incidence-rate ratio
`nbreg`	incidence-rate ratio

For `fmm: glm`, the interpretation of exponentiated coefficients depends on the family and link as follows:

Family	Link	Meaning of exp(coef)
Bernoulli	logit	odds ratio
Poisson	log	incidence-rate ratio
nbreg	log	incidence-rate ratio

For `fmm: streg`, the interpretation of exponentiated coefficients depends on the survival distribution and whether the proportional hazards or accelerated failure-time parameterization is used.

Survival distribution	Parameterization	Meaning of exp(coef)
exponential	PH	hazard ratio
exponential	AFT	time ratio
Weibull	PH	hazard ratio
Weibull	AFT	time ratio
gamma	AFT	time ratio
loglogistic	AFT	time ratio
lognormal	AFT	time ratio

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

estat lcmean — Latent class marginal means

Description

`estat lcmean` reports a table of the marginal predicted means of the outcome within each latent class. For `ivregress`, `mlogit`, `oprobit`, and `ologit`, a table is produced for each outcome.

`marginsplot` can be used after `estat lcmean` to plot the marginal predicted means for each class.

Menu for estat

Statistics > Postestimation

Syntax

`estat lcmean` [, *options*]

options	Description
`nose`	do not estimate SEs
`post`	post margins and their VCE as estimation results
display_options	control column formats, row spacing, and line width

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

Options

`nose` suppresses calculation of the VCE and standard errors.

`post` causes `estat lcmean` to behave like a Stata estimation (e-class) command. `estat lcmean` posts the vector of estimated margins along with the estimated variance–covariance matrix to `e()`, so you can treat the estimated margins just as you would results from any other estimation command.

display_options: `vsquish`, `fvwrap(#)`, `fvwrapon(`*style*`)`, `cformat(%`*fmt*`)`, `pformat(%`*fmt*`)`, `sformat(%`*fmt*`)`, and `nolstretch`.

Remarks and examples

`estat lcmean` is illustrated in **[FMM] Example 2** and **[FMM] Example 3**.

Stored results

`estat lcmean` stores the following in `r()`:

Scalars	
`r(N)`	number of observations
Macros	
`r(title)`	title in output
Matrices	
`r(b)`	estimates
`r(V)`	variance–covariance matrix of the estimates
`r(table)`	matrix containing the margins with their standard errors, test statistics, p-values, and confidence intervals

`estat lcmean` with the `post` option also stores the following in `e()`:

Scalars	
`e(N)`	number of observations
Macros	
`e(title)`	title in output
`e(properties)`	`b V`
Matrices	
`e(b)`	estimates
`e(V)`	variance–covariance matrix of the estimates

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

estat lcprob — Latent class marginal probabilities

Description | Menu for estat | Syntax | Options
Remarks and examples | Stored results | Also see

Description

estat lcprob reports a table of the marginal predicted latent class probabilities.

marginsplot can be used after estat lcprob to plot the marginal predicted latent class probabilities.

Menu for estat

Statistics > Postestimation

Syntax

estat lcprob [, *options*]

options	Description
classpr	latent class probability; the default
classposteriorpr	posterior latent class probability
nose	do not estimate SEs
post	post margins and their VCE as estimation results
display_options	control column formats, row spacing, and line width

collect is allowed; see **[U] 11.1.10 Prefix commands**.

Options

classpr, the default, calculates marginal predicted probabilities for each latent class.

classposteriorpr calculates marginal predicted posterior probabilities for each latent class. The posterior probabilities are a function of the latent-class predictors and the fitted outcome densities.

nose suppresses calculation of the VCE and standard errors.

post causes estat lcprob to behave like a Stata estimation (e-class) command. estat lcprob posts the vector of estimated margins along with the estimated variance–covariance matrix to e(), so you can treat the estimated margins just as you would results from any other estimation command.

display_options: vsquish, fvwrap(#), fvwrapon(*style*), cformat(%*fmt*), pformat(%*fmt*), sformat(%*fmt*), and nolstretch.

Remarks and examples

estat lcprob is illustrated in **[FMM] Example 1a**, **[FMM] Example 2**, and **[FMM] Example 3**.

Stored results

estat lcprob stores the following in r():

Scalars	
r(N)	number of observations
Macros	
r(title)	title in output
Matrices	
r(b)	estimates
r(V)	variance–covariance matrix of the estimates
r(table)	matrix containing the margins with their standard errors, test statistics, p-values, and confidence intervals

estat lcprob with the post option also stores the following in e():

Scalars	
e(N)	number of observations
Macros	
e(title)	title in output
e(classposteriorpr)	classposteriorpr or empty
e(properties)	b V
Matrices	
e(b)	estimates
e(V)	variance–covariance matrix of the estimates

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

Description | Quick start | Menu | Syntax
Options | Remarks and examples | Stored results | Methods and formulas
References | Also see

Description

`lcstats` calculates model-comparison statistics for latent class models fit using `fmm` or `gsem`. You can specify which statistics to show in the reported table, including the number of classes, estimation sample size, log likelihood, rank, entropy, Akaike information criterion (AIC), Schwarz Bayesian information criterion (BIC), corrected AIC (AICc), consistent AIC (CAIC), Vuong–Lo–Mendell–Rubin (VLMR) likelihood-ratio test, and Lo–Mendell–Rubin (LMR)-adjusted likelihood-ratio test.

The VLMR and LMR tests are commonly used to determine the number of latent classes your data supports for similarly specified models. To conduct the VLMR and LMR tests, you must store the estimation results using `estimates store`. `lcstats` also works with the current estimation results.

Quick start

Report the default statistics—number of classes, sample size, log likelihood, rank, and entropy—for a linear regression model with two latent classes

```
fmm 2 : regress y x1 x2 x3
lcstats .
```

Compare linear regression models with 1 latent class to linear regression model with 2 latent classes; report default statistics, including the LMR-adjusted likelihood-ratio test for 2 classes versus 1 class

```
fmm 1 : regress y x1 x2 x3
estimate store m1
fmm 2 : regress y x1 x2 x3
lcstats m1 .
```

Same as above, but also show AIC and BIC

```
lcstats m1 ., aic bic
```

Same as above, but split the output into two tables

```
lcstats m1 ., aic bic split
```

Same as above, but specify how to split the output

```
lcstats m1 ., results(N rank aic bic entropy) results(k_classes ll df lmr p_lmr)
```

Specify a single table, and select statistics of interest and column order

```
lcstats m1 ., results(k_classes bic lmr p_lmr entropy)
```

Menu

Statistics > Postestimation

Syntax

`lcstats` [*namelist*] [, *options*]

namelist is a name, a list of names, `_all`, or `*`. A name may be `.`, meaning the current (active) estimates. `_all` and `*` mean the same thing. If *namelist* is not specified, the current (active) estimates is used; this is equivalent to specifying *namelist* as "`.`".

name is the name under which estimation results were stored using `estimates store` (see [R] **estimates store**), and "`.`" refers to the last estimation results, whether or not these were already stored.

options	Description
Main	
`all`	report all available statistics
`noentropy`	do not report entropy
`allic`	report AIC, BIC, AICc, and CAIC
`aic`	report AIC
`bic`	report BIC
`aicc`	report AICc
`caic`	report CAIC
`noicnotes`	suppress notes for information criteria
`nolrtests`	do not report likelihood-ratio tests
`lmr`	report the LMR-adjusted likelihood-ratio test
`vlmr`	report the VLMR likelihood-ratio test
`nolrnotes`	suppress notes for likelihood-ratio tests
Formats	
* `pformat(`[*%fmt*] [`, ...`]`)`	specify numeric format for p-values
`nformat(`*%fmt* [*results*][`, basestyle`]`)`	specify numeric format
Split tables	
`split`	split output into two tables
`results(`*results*`)`	specify results and their order for separate tables
Options	
[`no`]`shownames`	show or hide estimates' names
`extraspace(#)`	specify the number of extra spaces between columns
`name(`*cname*`)`	work with collection *cname*; default is `name(LCStats)`
`replace`	replace the collection
`label(`*filename*`)`	specify the collection labels
`style(`*filename*[`, override`]`)`	specify the collection style

The full specification is `pformat(`[%fmt*] [`, minimum(`[#][`, label(`*string*`)`]`)`]`)`.

results is a list of result names and may include any of the following:

results	Definition
k_classes	number of classes
N	sample size
ll	log likelihood
rank	rank of e(V)
aic	AIC
bic	BIC
aicc	AICc
caic	CAIC
entropy	measure of separation between latent classes
df	degrees of freedom for the likelihood-ratio tests
vlmr	VLMR likelihood-ratio test statistic
p_vlmr	p-value for VLMR
lmr	LMR-adjusted likelihood-ratio test statistic
p_lmr	p-value for LMR

Options

Main

all specifies that all available statistics be reported in the output. This option is a shortcut for specifying aic, bic, aicc, caic, entropy, lmr, and vlmr.

noentropy specifies that entropy not be reported.

allic, aic, bic, aicc, caic, and noicnotes control the reporting of information criteria and their notes. The default is to not report information criteria.

allic specifies that all information criteria be reported in the output. This option is a shortcut for specifying aic, bic, aicc, and caic.

aic specifies that AIC be reported.

bic specifies that BIC be reported.

aicc specifies that AICc be reported. This information criterion is a second-order approximation and is recommended for small sample sizes.

caic specifies that CAIC be reported. This information criterion is a consistent version of AIC; that is, the probability of selecting the "true model" approaches 1 as sample size increases.

noicnotes suppresses the notes for the information criteria.

nolrtests, lmr, vlmr, and nolrnotes control reporting of likelihood-ratio tests comparing models with C versus $C-1$ latent classes. The default is lmr.

nolrtests specifies that no likelihood-ratio test be reported.

lmr specifies that the LMR-adjusted likelihood-ratio test be reported.

vlmr specifies that the VLMR likelihood-ratio test be reported.

nolrnotes suppresses the likelihood-ratio test notes.

Formats

`pformat([%fmt] [, minimum([#][, label(string)])])` changes the numeric format, such as the number of decimal places, for *p*-value results `p_lmr` and `p_vlmr`.

`minimum([#][, label(string)])` specifies that *p*-values less than # be displayed as "<#", where # is formatted according to *%fmt*.

If suboption `label(string)` is specified, then "*string*" is used instead of "<#". If *string* contains `%s`, then `%s` is replaced by # formatted according to *%fmt*.

The default style is equivalent to `pformat(%6.3f, minimum(.001))`.

`nformat(%fmt [results][, basestyle])` changes the numeric format, such as the number of decimal places, for specified results. If *results* are not specified, the numeric format is changed for all results.

This option is repeatable, and when multiple formats apply to one result, the rightmost specification is applied. Note that specifying a `pformat()` option will override any `nformat()` option applied to the *p*-value results `p_lmr` and `p_vlmr`, regardless of the order that the options are specified.

`basestyle` indicates that the format be applied to results that do not already have their own format instead of overriding the format for all results.

The default style is equivalent to

```
nformat(%9.0g, basestyle)
nformat(%6.4f entropy)
nformat(%21.0fc N k_classes rank df)
nformat(%21.2fc aic bic aicc caic)
nformat(%21.2fc ll lmr vlmr)
```

Split tables

`split` and `results(results)` control how to split the reported statistics into multiple tables.

`split` is a shortcut for splitting the results into two tables: entropy and the information criteria are reported in the first table; likelihood-ratio test results are reported in the second table.

By default, `split` is a shortcut for

```
results(N rank entropy)
results(k_classes ll df lmr p_lmr)
```

With option `all`, `split` is a shortcut for

```
results(N rank aic bic aicc caic entropy)
results(k_classes ll df vlmr p_vlmr lmr p_lmr)
```

`results(results)` specifies the results to report in the table columns. This option is repeatable, and each specification defines a separate table. Results not selected in any of the specified `results()` options are suppressed from the output. Repeating results is not allowed.

Options

`shownames` and `noshownames` control reporting of estimates' names in the table row headers. The default is to show the estimates' names in the table row headers.

`extraspace(#)` specifies extra spaces to pad columns in each reported table. The first and middle columns get # extra spaces added on both sides. The last column gets # extra spaces added on the left. The default is `extraspace(1)`.

This column property is also respected by collect export when publishing your collection to SMCL and plain text.

name(*cname*) specifies the collection for lcstats to work with. The default is name(LCStats).

replace permits lcstats to overwrite the existing collection. This option is implied for name(LCStats).

label(*filename*) specifies the *filename* containing the collection labels to use for your table. Labels in *filename* will be loaded for the table, and default labels will be used for any labels not specified in *filename*.

style(*filename*[, override]) specifies the *filename* containing the collection styles to use for your table. This might be a style you saved with collect style save or a predefined style shipped with Stata. The lcstats collection styles will be discarded, and only the collection styles in *filename* will be applied. Note that the layout specification saved in *filename* will not be applied; lcstats will always specify the layout.

If you prefer the lcstats collection styles but also want to apply any styles in *filename*, specify override. If there are conflicts between the default collection styles and those in *filename*, the ones in *filename* will take precedence.

The default is to use only the collection styles defined in style-lcstats.stjson; see [TABLES] **Predefined styles**.

Remarks and examples

lcstats is illustrated in [FMM] **Example 1a**, [FMM] **Example 1b**, and [FMM] **Example 1d**.

Stored results

lcstats stores the following in r():

Matrices	
r(S)	latent class statistics

The rows of r(S) correspond with the names of the estimation results in the order they were specified. See the *results* table in *Syntax* for the complete list and order of the columns of r(S).

Methods and formulas

For each estimation result, `lcstats` collects or computes the following:

- `k_classes`: number of classes, `e(lclass_k_levels)`
- `N`: sample size, `e(N)`
- `ll`: log likelihood, `e(ll)`
- `rank`: rank of `e(V)`
- `aic`: AIC
- `bic`: BIC
- `aicc`: AICc
- `caic`: CAIC
- `entropy`: measure of separation between latent classes

Akaike's (1974) information criterion is defined as

$$\texttt{aic} = -2\ln L + 2k$$

where $\ln L$ is the maximized log likelihood of the model and k is the number of parameters estimated (that is, `rank`). Schwarz's (1978) BIC is another measure of fit defined as

$$\texttt{bic} = -2\ln L + k\ln N$$

where N is the sample size. Hurvich and Tsai (1989) derived a second-order variant of AIC called AICc,

$$\texttt{aicc} = \texttt{aic} + \frac{2k(k+1)}{N-k-1}$$

Bozdogan (1987) proposed a consistent version of AIC called CAIC,

$$\texttt{caic} = -2\ln L + k(\ln N + 1)$$

See [R] **estat ic** for a focused discussion of these information criteria.

Entropy is computed from the predicted posterior latent class probabilities, as described by Ramaswamy et al. (1993). Let C be the number of latent classes for a given estimation and $\widehat{p}_{ij}$ be the predicted posterior probability for class i in observation j, where $i = 1, \dots, C$ and $j = 1, \dots, N$. Then

$$\texttt{entropy} = 1 + \frac{1}{N\ln(C)} \sum_{j=1}^{N} \sum_{i=1}^{C} \widehat{p}_{ij} \ln(\widehat{p}_{ij})$$

`entropy` ranges from 0 to 1, and values closer to 1 indicate better separation between latent classes.

Let M_1 and M_2 denote estimation results based on the same data and model specifications for the observed outcome variables. Denote their log-likelihood values by $\ln L_1$ and $\ln L_2$ and ranks by k_1 and k_2. Suppose M_1 has $C-1$ latent classes and M_2 has C latent classes. Then, according to Vuong (1989) and Lo, Mendell, and Rubin (2001), the likelihood-ratio test statistic

$$\texttt{vlmr} = 2(\ln L_2 - \ln L_1)$$

is asymptotically distributed as a weighted sum of independent χ_1^2 variables. The LMR-adjusted likelihood-ratio test statistic is

$$\texttt{lmr} = \frac{2(\ln L_2 - \ln L_1)}{1 + 1/\{(k_2 - k_1)\ln N\}}$$

and has the same asymptotic distribution as `vlmr`. The reported degrees of freedom for these likelihood-ratio tests is

$$\texttt{df} = k_2 - k_1$$

The p-values `p_vlmr` and `p_lmr` are computed using a numerical approximation of the distribution of the weighted sum of independent χ_1^2 variables.

References

Akaike, H. 1974. A new look at the statistical model identification. *IEEE Transactions on Automatic Control* 19: 716–723. https://doi.org/10.1109/TAC.1974.1100705.

Bozdogan, H. 1987. Model selection and Akaike's information criterion (AIC): The general theory and its analytical extensions. *Psychometrika* 52: 345–370. https://doi.org/10.1007/BF02294361.

Hurvich, C. M., and C.-L. Tsai. 1989. Regression and time series model selection in small samples. *Biometrika* 76: 297–307. https://doi.org/10.1093/biomet/76.2.297.

Lo, Y., N. R. Mendell, and D. B. Rubin. 2001. Testing the number of components in a normal mixture. *Biometrika* 88: 767–778. https://doi.org/10.1093/biomet/88.3.767.

Ramaswamy, V., W. S. Desarbo, D. J. Reibstein, and W. T. Robinson. 1993. An empirical pooling approach for estimating marketing mix elasticities with PIMS data. *Marketing Science* 12: 103–124. https://doi.org/10.1287/mksc.12.1.103.

Schwarz, G. 1978. Estimating the dimension of a model. *Annals of Statistics* 6: 461–464. https://doi.org/10.1214/aos/1176344136.

Vuong, Q. H. 1989. Likelihood ratio tests for model selection and non-nested hypotheses. *Econometrica* 57: 307–333. https://doi.org/10.2307/1912557.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm postestimation** — Postestimation tools for fmm

Example 1a — Mixture of linear regression models

Description Remarks and examples References Also see

Description

In this example, we show how to fit FMMs with covariates, and we illustrate how you might determine the number of latent classes. For an example without covariates and for a conceptual overview of FMMs, see [FMM] **fmm intro**.

Remarks and examples

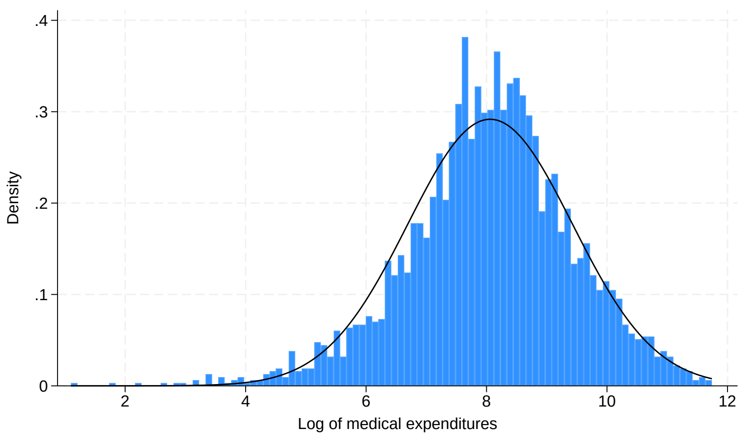
Medical expenditures vary greatly from person to person. We believe that some of the variation may be due to having different types of medical care users. We might think of these types as low spenders, average spenders, and high spenders. Because we cannot necessarily tell which group a person belongs to, an FMM may be appropriate for these data.

We use an abbreviated version of `mus03data.dta` from Cameron and Trivedi (2022, chap. 3). `mus03sub.dta` contains information on the log of medical expenditures, `lmedexp`. For brevity, we use only the variables `female`, `age`, `income`, and `totchr`, the last variable recording the number of chronic health problems.

First, let us look at the distribution of medical expenditures.

```
. use https://www.stata-press.com/data/r19/mus03sub
(Abbreviated dataset mus203mepsmedexp from Cameron and Trivedi (2022))

. histogram lmedexp, bins(100) normal
(bin=100, start=1.0986123, width=.10642325)
```



The variable `lmedexp` looks approximately normally distributed. Indeed, it looks as if it may come from a single normal distribution. However, our model includes covariates, and this histogram does not give us an indication of how the regression models may differ across groups. We start by fitting the three-group model, but we will certainly want to check whether a model with a single distribution or with two distributions is a better fit for these data.

```
. fmm 3: regress lmedexp income c.age##c.age totchr i.sex
Fitting class model:
Iteration 0:  (class) log likelihood = -3246.3993
Iteration 1:  (class) log likelihood = -3246.3993
Fitting outcome model:
Iteration 0:  (outcome) log likelihood = -4700.2736
Iteration 1:  (outcome) log likelihood = -4700.2736
Refining starting values:
Iteration 0:  (EM) log likelihood =  -7482.765
Iteration 1:  (EM) log likelihood = -7327.5583
Iteration 2:  (EM) log likelihood = -7271.2407
Iteration 3:  (EM) log likelihood = -7254.4109
Iteration 4:  (EM) log likelihood = -7246.0793
Iteration 5:  (EM) log likelihood =  -7238.679
Iteration 6:  (EM) log likelihood = -7231.9742
Iteration 7:  (EM) log likelihood = -7226.4046
Iteration 8:  (EM) log likelihood = -7222.1152
Iteration 9:  (EM) log likelihood = -7219.0098
Iteration 10: (EM) log likelihood = -7216.9001
Iteration 11: (EM) log likelihood = -7215.5809
Iteration 12: (EM) log likelihood = -7214.8641
Iteration 13: (EM) log likelihood = -7214.5912
Iteration 14: (EM) log likelihood = -7214.6342
Iteration 15: (EM) log likelihood = -7214.8937
Iteration 16: (EM) log likelihood = -7215.2936
Iteration 17: (EM) log likelihood = -7215.7769
Iteration 18: (EM) log likelihood = -7216.3017
Iteration 19: (EM) log likelihood = -7216.8377
Iteration 20: (EM) log likelihood = -7217.3632
note: EM algorithm reached maximum iterations.
Fitting full model:
Iteration 0:  Log likelihood = -4734.6429
Iteration 1:  Log likelihood = -4733.3724
Iteration 2:  Log likelihood = -4732.1323
Iteration 3:  Log likelihood = -4731.0186
Iteration 4:  Log likelihood = -4729.3225
Iteration 5:  Log likelihood = -4727.7218
Iteration 6:  Log likelihood = -4727.6741
Iteration 7:  Log likelihood = -4727.6738
Finite mixture model                                Number of obs = 2,955
Log likelihood = -4727.6738
```

	Coefficient	Std. err.	z	P>|z|	[95% conf. interval]	
1.Class	(base outcome)					
2.Class						
_cons	1.162296	.292186	3.98	0.000	.5896216	1.73497
3.Class						
_cons	-1.153202	.3188697	-3.62	0.000	-1.778175	-.5282289

```
Class:    1
Response: lmedexp
Model:    regress
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
lmedexp						
income	.0059804	.002604	2.30	0.022	.0008768	.0110841
age	.1201823	.2926979	0.41	0.681	-.4534951	.6938597
c.age#c.age	-.0007572	.0019417	-0.39	0.697	-.0045628	.0030483
totchr	.9223744	.0810612	11.38	0.000	.7634974	1.081251
sex						
Female	.0576508	.1453985	0.40	0.692	-.227325	.3426266
_cons	.6300965	10.96433	0.06	0.954	-20.8596	22.11979
var(e.lmed~p)	1.43183	.1533984			1.160642	1.766382

```
Class:    2
Response: lmedexp
Model:    regress
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
lmedexp						
income	.0023725	.0012209	1.94	0.052	-.0000205	.0047655
age	.2136658	.1075408	1.99	0.047	.0028897	.424442
c.age#c.age	-.0013195	.0007152	-1.84	0.065	-.0027213	.0000823
totchr	.3106586	.0292864	10.61	0.000	.2532583	.3680589
sex						
Female	-.0918924	.0543976	-1.69	0.091	-.1985097	.0147249
_cons	-.9546721	4.017561	-0.24	0.812	-8.828947	6.919602
var(e.lmed~p)	.7966127	.0805009			.6534764	.9711013

```
Class:    3
Response: lmedexp
Model:    regress
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
lmedexp						
income	.0009315	.0048146	0.19	0.847	-.0085049	.0103679
age	-.2645947	.2637125	-1.00	0.316	-.7814618	.2522724
c.age#c.age	.0015761	.001754	0.90	0.369	-.0018616	.0050138
totchr	.186475	.0647115	2.88	0.004	.0596427	.3133072
sex						
Female	-.1761484	.1371471	-1.28	0.199	-.4449517	.0926549
_cons	20.79524	9.853989	2.11	0.035	1.481775	40.1087
var(e.lmed~p)	.3846891	.0983236			.2331038	.634849

That is a lot of output! Let's start with the part of the output that is probably familiar if you have used regress. We have one regression table for each class. The coefficient estimates here are interpreted just as you do the coefficients from a linear regression model. Because the dependent variable is log transformed, we can interpret the coefficients in terms of a percentage change. For example, a one-unit increase in totchr results in an 18.6% increase in medical expenditures for class 3, all else held constant. The estimates for each class also include a variance term. So, we see that the first class has much higher variability than the third.

The first table in the output gives the coefficients for the latent class membership, next to 1.Class, 2.Class, and 3.Class at the top of the table. These coefficients can be interpreted in the same manner as you interpret the coefficients from multinomial logistic regression (mlogit), which is to say that they are difficult to interpret. However, the postestimation command estat lcprob will turn them into probabilities.

```
. estat lcprob, nose

Latent class marginal probabilities                  Number of obs = 2,955

--------------------------------------------------------------
             |     Margin
-------------+------------------------------------------------
       Class |
          1  |   .2215875
          2  |    .708474
          3  |   .0699385
--------------------------------------------------------------
```

We see that individuals in the population fall into the three classes in proportions 0.22, 0.71, and 0.07. Notice that we specified the nose option above. estat lcprob can be slow because it is time consuming to compute standard errors when there are a lot of covariates in the model. When fitting preliminary models, we might not be concerned about standard errors of the latent class probabilities, so we use the nose option to speed things up.

We have estimated that about 22% of observations are in group 1, about 71% are in group 2, and about 7% are in group 3. But, we still do not know which group corresponds to which spending class. If we want to calculate the level of spending for each group, we can use estat lcmean to calculate the marginal means for each class; see [FMM] **estat lcmean**.

```
. estat lcmean

Latent class marginal means                          Number of obs = 2,955

------------------------------------------------------------------------------
             |            Delta-method
             |     Margin   std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
1            |
     lmedexp |   7.185846   .1572402    45.70   0.000     6.877661    7.494031
-------------+----------------------------------------------------------------
2            |
     lmedexp |   8.143981   .0469051   173.63   0.000     8.052049    8.235914
-------------+----------------------------------------------------------------
3            |
     lmedexp |   10.15809   .1712913    59.30   0.000     9.822369    10.49382
------------------------------------------------------------------------------
```

We see that class 1 corresponds to low spenders, class 2 corresponds to average spenders, class 3 corresponds to high spenders.

Because medical expenditures for class 1 and class 2 are relatively close to each other, compared with class 3, we may be tempted to fit a model with two classes. We may also compare our model with a model with one class, which reduces to a linear regression.

First, we store our estimates from the model with three latent classes with the name `fmm3` by using `estimates store`.

```
. estimates store fmm3
```

Then, we fit a model with two classes and then a model with one class, storing the results of each model in `fmm2` and `fmm1`, respectively.

```
. fmm 2: regress lmedexp income c.age##c.age totchr i.sex
  (output omitted)
. estimates store fmm2
. fmm 1: regress lmedexp income c.age##c.age totchr i.sex
  (output omitted)
. estimates store fmm1
```

Finally, we use `lcstats` to compare these fitted models.

```
. lcstats fmm1 fmm2 fmm3
Latent class statistics
```

	Classes	N	ll	Rank	Entropy	df	LMR	P>LMR
fmm1	1	2,955	-4,807.39	7				
fmm2	2	2,955	-4,758.18	15	0.5304	8	96.90	<0.001
fmm3	3	2,955	-4,727.67	23	0.5367	8	60.07	<0.001

```
LMR is the Lo-Mendell-Rubin-adjusted likelihood-ratio test statistic.
Likelihood-ratio tests compare the given model versus the same model with
one less latent class.
```

`lcstats` reports the sample size, log likelihood, and rank for each fitted model. It also reports entropy, a measure of class separation, for models with 2 or more latent classes. Larger entropy values, closer to 1, correspond to better separation of classes. The specified estimates only differ in the number of latent classes, each having one more latent class than the previous, so `lcstats` also reports the Lo–Mendell–Rubin (LMR) adjusted likelihood-ratio test for two scenarios.

1. The first is reported in the row labeled `fmm2`, comparing this model with two latent classes versus `fmm1` with one latent class. We find evidence that the two class model fits better than the one class model..

2. The second scenario is reported in the row labeled `fmm3`, comparing this model with three latent classes versus `fmm2` with two latent classes. We find evidence that the three class model fits better than the two class model. scenario.

`lcstats` has options for reporting the usual information criteria. Here we add option `allic` to get all the information criteria. Adding these statistics makes the table wide, so we also add option `split` to request that `lcstats` partition the reported statistics into two tables.

```
. lcstats fmm1 fmm2 fmm3, allic split
Latent class statistics
```

	N	Rank	AIC	BIC	AICc	CAIC	Entropy
fmm1	2,955	7	9,628.77	9,670.71	9,628.81	9,677.71	
fmm2	2,955	15	9,546.35	9,636.22	9,546.52	9,651.22	0.5304
fmm3	2,955	23	9,501.35	9,639.15	9,501.72	9,662.15	0.5367

```
AIC is the Akaike information criterion.
BIC is the Bayesian information criterion.
AICc is the corrected Akaike information criterion.
CAIC is the consistent Akaike information criterion.
BIC, AICc, and CAIC use N = number of observations.
```

	Classes	ll	df	LMR	P>LMR
fmm1	1	-4,807.39			
fmm2	2	-4,758.18	8	96.90	<0.001
fmm3	3	-4,727.67	8	60.07	<0.001

```
LMR is the Lo-Mendell-Rubin-adjusted
likelihood-ratio test statistic.
Likelihood-ratio tests compare the given model
versus the same model with one less latent class.
```

The Akaike information criterion (AIC) and its sample-size corrected version (AICc) clearly favor the three-component model, whereas the Bayesian information criterion (BIC) and the consistent version of AIC (CAIC) marginally favor the two-component model; see [R] **estat ic** for more information about these information criteria.

We will proceed with the three-component model.

❑ Technical note

Prior to the addition of `lcstats` to Stata, we might have been tempted to use the standard likelihood-ratio test (see [R] **lrtest**) to help us decide how many latent classes to fit. However, a model with $C-1$ classes with covariates for the mean is not nested in the model extended to C classes because of the additional equation for the mean of the Cth component. The model with $C-1$ classes could be viewed as the model with C classes with variance components of the Cth class model going to zero. But the parameter value of zero lies on the boundary of the parameter space, and the standard regularity conditions necessary for the likelihood-ratio test do not hold. See McLachlan and Peel (2000, 185) for a detailed explanation.

❑

References

Cameron, A. C., and P. K. Trivedi. 2022. *Microeconometrics Using Stata*. 2nd ed. College Station, TX: Stata Press.

McLachlan, G. J., and D. Peel. 2000. *Finite Mixture Models*. New York: Wiley. https://doi.org/10.1002/0471721182.

Also see

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm: regress** — Finite mixtures of linear regression models

[FMM] **estat lcmean** — Latent class marginal means

[FMM] **estat lcprob** — Latent class marginal probabilities

[FMM] **lcstats** — Latent class model-comparison statistics

Description Remarks and examples Also see

Description

In this example, we demonstrate how to fit an FMM with covariates that model the probability of class membership.

Remarks and examples

We continue with Example 1a, where we settled on the three-component mixture model as being the best fit for these data. In that example, we used variables from our data to predict the mean of medical expenditures for each latent class. However, the prior probability of being in a given class was the same for each individual.

Assuming that the probabilities of belonging to a particular class are the same for all individuals does not seem realistic for these data. It seems more reasonable to think that individual characteristics predict the probability of being in a given group. We specify `totchr` in the `lcprob()` option to model the latent class probabilities based on the number of chronic conditions a person has.

```
. use https://www.stata-press.com/data/r19/mus03sub
(Abbreviated dataset mus203mepsmedexp from Cameron and Trivedi (2022))
. fmm 3, lcprob(totchr): regress lmedexp income c.age##c.age totchr i.sex
Fitting class model:
```

(iteration log omitted)

```
Finite mixture model                              Number of obs = 2,955
Log likelihood = -4712.3871
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
1.Class	(base outcome)					
2.Class						
totchr	.9376084	.2222695	4.22	0.000	.5019683	1.373249
_cons	-.6114399	.4542569	-1.35	0.178	-1.501767	.2788872
3.Class						
totchr	1.16097	.2588803	4.48	0.000	.6535739	1.668366
_cons	-3.270603	.6134585	-5.33	0.000	-4.47296	-2.068246

Class: 1
Response: lmedexp
Model: regress

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
lmedexp						
income	.0048917	.0026337	1.86	0.063	-.0002702	.0100537
age	.0261976	.284515	0.09	0.927	-.5314416	.5838368
c.age#c.age	-.0000843	.0018944	-0.04	0.965	-.0037973	.0036286
totchr	.5412491	.1163553	4.65	0.000	.3131969	.7693012
sex						
Female	.1793964	.1507783	1.19	0.234	-.1161237	.4749164
_cons	5.035174	10.61396	0.47	0.635	-15.76781	25.83815
var(e.lmed~p)	2.311098	.2100365			1.934015	2.761703

Class: 2
Response: lmedexp
Model: regress

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
lmedexp						
income	.0027131	.0013618	1.99	0.046	.0000439	.0053822
age	.2675077	.1152288	2.32	0.020	.0416634	.4933519
c.age#c.age	-.001688	.0007648	-2.21	0.027	-.0031869	-.0001891
totchr	.2878736	.0354297	8.13	0.000	.2184327	.3573145
sex						
Female	-.1326158	.0602376	-2.20	0.028	-.2506795	-.0145522
_cons	-2.895759	4.313613	-0.67	0.502	-11.35029	5.558767
var(e.lmed~p)	.7413402	.0801554			.5997686	.9163288

Class: 3
Response: lmedexp
Model: regress

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
lmedexp						
income	-.0061289	.0041295	-1.48	0.138	-.0142226	.0019648
age	-.2012074	.2578283	-0.78	0.435	-.7065417	.3041268
c.age#c.age	.0011186	.0017078	0.65	0.512	-.0022287	.0044659
totchr	.106383	.0878267	1.21	0.226	-.0657542	.2785202
sex						
Female	-.3027395	.1371042	-2.21	0.027	-.5714588	-.0340202
_cons	18.93315	9.651339	1.96	0.050	.0168759	37.84943
var(e.lmed~p)	.3241542	.1006027			.176432	.5955603

In the first table, we find evidence that the coefficients on `totchr` are different from zero in both class probability equations. We use `estimates store` and then `lcstats` to compare this model with the three-component one we fit in Example 1a.

```
. estimates store fmm3f

. lcstats fmm3 fmm3f, aic bic

Latent class statistics
```

	Classes	N	ll	Rank	AIC	BIC	Entropy
fmm3	3	2,955	-4,727.67	23	9,501.35	9,639.15	0.5367
fmm3f	3	2,955	-4,712.39	25	9,474.77	9,624.56	0.5018

```
AIC is the Akaike information criterion.
BIC is the Bayesian information criterion.
BIC uses N = number of observations.
```

Both the AIC and the BIC favor the model that uses a predictor to model class probabilities. We continue with this new model in Example 1c, where we illustrate some postestimation features.

Also see

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm: regress** — Finite mixtures of linear regression models

[FMM] **estat lcmean** — Latent class marginal means

[FMM] **estat lcprob** — Latent class marginal probabilities

[FMM] **lcstats** — Latent class model-comparison statistics

Description Remarks and examples Also see

Description

In this example, we demonstrate how to use `test` and `contrast` to test the equality of coefficients across classes after fitting an FMM.

Remarks and examples

We continue with Example 1b, where we fit a three-component mixture model for the logarithm of medical expenditures. The best model we found was one in which we used total chronic conditions (`totchr`) in the `lcprob()` option of `fmm` to predict latent class probabilities and additional covariates to predict the means for the latent classes.

At this point, we may want to begin looking at how the effect of covariates differs by class. For example, we may want to know if being female has the same effect on medical expenditures in the low-, medium-, and high-spending classes. To do this, we can test the coefficient on `1.sex` in the equations for the class means.

Many of Stata's postestimation commands require you to specify an expression if you want, for example, to perform a test of equality (`test`), compute a difference between estimates (`lincom`), or compute a ratio of coefficients (`nlcom`). Remembering how to specify the names of estimates can be difficult. We first redisplay the estimation output with the `coeflegend` option so we can see the legend of the coefficients and how to specify them in an expression.

```
. fmm, coeflegend

Finite mixture model                               Number of obs = 2,955
Log likelihood = -4712.3871

------------------------------------------------------------------------------
             |  Coefficient  Legend
-------------+----------------------------------------------------------------
1.Class      |  (base outcome)
-------------+----------------------------------------------------------------
2.Class      |
      totchr |   .9376084  _b[2.Class:totchr]
       _cons |  -.6114399  _b[2.Class:_cons]
-------------+----------------------------------------------------------------
3.Class      |
      totchr |    1.16097  _b[3.Class:totchr]
       _cons |  -3.270603  _b[3.Class:_cons]
------------------------------------------------------------------------------

Class:    1
Response: lmedexp
Model:    regress

------------------------------------------------------------------------------
             |  Coefficient  Legend
-------------+----------------------------------------------------------------
lmedexp      |
      income |   .0048917  _b[lmedexp:1.Class#c.income]
         age |   .0261976  _b[lmedexp:1.Class#c.age]
             |
 c.age#c.age |  -.0000843  _b[lmedexp:1.Class#c.age#c.age]
             |
      totchr |   .5412491  _b[lmedexp:1.Class#c.totchr]
             |
         sex |
     Female  |   .1793964  _b[lmedexp:1.sex#1.Class]
       _cons |   5.035174  _b[lmedexp:1.Class]
-------------+----------------------------------------------------------------
var(e.lmed~p)|   2.311098  _b[/var(e.lmedexp)#1.Class]
------------------------------------------------------------------------------
```

(*output omitted*)

Here we test individually whether the effect of being female in class 1 is the same as the effect of being female in class 2 and whether the effect of being female in class 2 is the same as the effect of being female in class 3.

```
. test (_b[lmedexp:1.Class#1.sex] = _b[lmedexp:2.Class#1.sex])

 ( 1)  [lmedexp]1.sex#1bn.Class - [lmedexp]1.sex#2.Class = 0

           chi2(  1) =    3.04
         Prob > chi2 =    0.0811

. test (_b[lmedexp:2.Class#1.sex] = _b[lmedexp:3.Class#1.sex])

 ( 1)  [lmedexp]1.sex#2.Class - [lmedexp]1.sex#3.Class = 0

           chi2(  1) =    1.46
         Prob > chi2 =    0.2270
```

Neither test is significant; therefore, we cannot reject the null of the coefficients being equal. We can also do a joint test.

```
. test (_b[lmedexp:1.Class#1.sex] = _b[lmedexp:2.Class#1.sex])
>      (_b[lmedexp:2.Class#1.sex] = _b[lmedexp:3.Class#1.sex])

 ( 1)  [lmedexp]1.sex#1bn.Class - [lmedexp]1.sex#2.Class = 0
 ( 2)  [lmedexp]1.sex#2.Class - [lmedexp]1.sex#3.Class = 0

           chi2(  2) =    5.11
         Prob > chi2 =    0.0775
```

The joint test is also not significant.

Alternatively, `contrast` can do all the work for us without the need of remembering coefficient names. Here we use the `a.` operator on `Class` to compare adjacent class categories. See [R] **contrast** for additional comparisons that we could make.

```
. contrast sex#a.Class, equation(lmedexp)

Contrasts of marginal linear predictions

Margins: asbalanced
```

	df	chi2	P>chi2
lmedexp			
sex#Class			
(joint) (1 vs 2)	1	3.04	0.0811
(joint) (2 vs 3)	1	1.46	0.2270
Joint	2	5.11	0.0775

We obtain exactly the same results reported by `test` but in a more succinct format.

Also see

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm: regress** — Finite mixtures of linear regression models

[FMM] **fmm postestimation** — Postestimation tools for fmm

Description Remarks and examples Also see

Description

In this example, we demonstrate how to fit FMMs with class-specific covariates using the hybrid syntax; see [FMM] **fmm** for details.

Remarks and examples

We continue with Example 1b, where we settled on the three-component mixture model with the variable `totchr` modeling class probabilities as being the best fit for these data. We notice that the variable `sex` in our model from Example 1b is not significant in the class 1 model. To omit this variable from the class 1 equation but keep it for the class 2 and class 3 equations, we use the hybrid syntax.

```
. fmm, lcprob(totchr): (regress lmedexp income c.age##c.age totchr)
>                      (regress lmedexp income c.age##c.age totchr i.sex)
>                      (regress lmedexp income c.age##c.age totchr i.sex)
```

(iteration log omitted)

```
Finite mixture model                                Number of obs = 2,955
Log likelihood = -4713.1378

------------------------------------------------------------------------------
             | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
1.Class      |  (base outcome)
-------------+----------------------------------------------------------------
2.Class      |
      totchr |   .9462362   .2230292     4.24   0.000      .509107    1.383366
       _cons |  -.6516843   .4582362    -1.42   0.155    -1.549811    .2464422
-------------+----------------------------------------------------------------
3.Class      |
      totchr |    1.18053   .2592234     4.55   0.000     .6724612    1.688598
       _cons |  -3.351777   .6142948    -5.46   0.000    -4.555773   -2.147781
------------------------------------------------------------------------------

Class:    1
Response: lmedexp
Model:    regress

------------------------------------------------------------------------------
             | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
lmedexp      |
      income |   .0044082   .0025775     1.71   0.087    -.0006437    .0094601
         age |   .0112209   .2807385     0.04   0.968    -.5390164    .5614582
             |
 c.age#c.age |   .0000205   .0018687     0.01   0.991    -.0036421    .0036831
             |
      totchr |   .5379611   .1147846     4.69   0.000     .3129875    .7629347
       _cons |   5.699667   10.47167     0.54   0.586    -14.82444    26.22377
-------------+----------------------------------------------------------------
var(e.lmed~p)|   2.326567   .2087898                      1.951315    2.773983
------------------------------------------------------------------------------
```

```
Class:    2
Response: lmedexp
Model:    regress
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
lmedexp						
income	.0027704	.0013668	2.03	0.043	.0000915	.0054492
age	.2714012	.115707	2.35	0.019	.0446196	.4981828
c.age#c.age	-.0017135	.0007679	-2.23	0.026	-.0032185	-.0002085
totchr	.2870954	.0351779	8.16	0.000	.218148	.3560428
sex						
Female	-.1060824	.0560499	-1.89	0.058	-.2159383	.0037734
_cons	-3.057941	4.331862	-0.71	0.480	-11.54823	5.432352
var(e.lmed~p)	.7398619	.0805511			.5976923	.9158486

```
Class:    3
Response: lmedexp
Model:    regress
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
lmedexp						
income	-.006469	.0041191	-1.57	0.116	-.0145423	.0016044
age	-.185511	.2573091	-0.72	0.471	-.6898276	.3188057
c.age#c.age	.0010118	.0017054	0.59	0.553	-.0023306	.0043543
totchr	.1000723	.0861764	1.16	0.246	-.0688303	.2689748
sex						
Female	-.2824174	.1344932	-2.10	0.036	-.5460192	-.0188156
_cons	18.37937	9.628842	1.91	0.056	-.4928137	37.25155
var(e.lmed~p)	.3186378	.098786			.1735412	.5850485

We store our estimates and compare this model with the model in Example 1b.

```
. estimates store fmm3ff
. lcstats fmm3f fmm3ff, aic bic
Latent class statistics
```

	Classes	N	ll	Rank	AIC	BIC	Entropy
fmm3f	3	2,955	-4,712.39	25	9,474.77	9,624.56	0.5018
fmm3ff	3	2,955	-4,713.14	24	9,474.28	9,618.07	0.5006

```
AIC is the Akaike information criterion.
BIC is the Bayesian information criterion.
BIC uses N = number of observations.
```

The AIC for this more parsimonious model is about the same as the previous model (`fmm3f`), which was our best model. The BIC here appears to be rewarding us for our parsimony.

Also see

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm: regress** — Finite mixtures of linear regression models

[FMM] **estat lcmean** — Latent class marginal means

[FMM] **estat lcprob** — Latent class marginal probabilities

[FMM] **lcstats** — Latent class model-comparison statistics

Description Remarks and examples References Also see

Description

In this example, we demonstrate how to fit a two-component mixture of Poisson regressions models. We also use `estat lcmean` to estimate marginal predicted counts and `estat lcprob` to estimate the proportion of individuals in each class.

Remarks and examples

We are interested in fitting a Poisson regression to model the annual number of doctor visits. We hypothesize that there are two distinct groups or classes in the population that differ in their healthcare utilization—frequent users and infrequent users—and we believe that the model may differ across these two groups.

We do not have any information that tells us which individuals in our sample belong to which group. With FMM, we can specify two latent classes in our model to identify these groups. To account for differences between the latent classes, we include predictor variables in our model to fit potentially different Poisson distributions for each class.

Here we replicate the finite mixture Poisson regression example from [SEM] **Example 53g**. We use the following data:

```
. use https://www.stata-press.com/data/r19/gsem_mixture
(US Medical Expenditure Panel Survey (2003))
. describe
Contains data from https://www.stata-press.com/data/r19/gsem_mixture.dta
 Observations:         3,677                  US Medical Expenditure Panel
                                                Survey (2003)
    Variables:            12                  26 Jan 2025 08:46
                                              (_dta has notes)
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
drvisits        int     %9.0g                 Number of doctor visits
private         byte    %8.0g                 Has private supplementary
                                                insurance
medicaid        byte    %8.0g                 Has Medicaid public insurance
age             byte    %8.0g                 Age in years
educ            byte    %8.0g                 Years of education
actlim          byte    %8.0g                 Has activity limitations
chronic         byte    %8.0g                 Number of chronic conditions
income          float   %9.0g                 Income in $1,000s
offer           byte    %8.0g                 Employer offers insurance
hpvisits        int     %8.0g                 Number of visits to health
                                                professionals other than
                                                doctors
female          byte    %8.0g                 Female
phylim          byte    %8.0g                 Has physical limitation
-------------------------------------------------------------------------------
Sorted by:
. notes
_dta:
  1.  Data on annual number of doctor visits for individuals age 65 and older
      from the US Medical Expenditure Panel Survey for 2003.
  2.  Data are analyzed in Cameron, A. C., and P. K. Trivedi. 2010.
      Microeconometrics Using Stata. Rev. ed. College Station, TX: Stata Press.
  3.  Additional information on finite mixture models for count data and a
      similar example are found in Deb, P., and P. K. Trivedi. 1997. Demand for
      medical care by the elderly: A finite mixture approach. Journal of
      Applied Econometrics 12: 313-336.
      https://doi.org/10.1002/(SICI)1099-1255(199705)12:3<313::AID-JAE440>3.0.C
      > O;2-G.
```

Following Cameron and Trivedi (2022), we fit an FMM with a Poisson regression component for each latent class. We model the number of doctor visits as a function of whether an individual has private supplementary insurance, whether he or she has Medicaid, age, age squared, education level, whether he or she has activity limitations, and the number of chronic conditions.

We add the `startvalues(randomid, draws(5) seed(15))` option to specify that five random draws are taken when computing starting values. The class assignment is selected from the draw that has the best log likelihood after the EM iterations. When fitting FMMs, taking multiple draws of random starting values can help to prevent convergence at a local maximum rather than the global maximum. `fmm` provides a variety of options for obtaining starting values; see [FMM] **fmm** for more information on starting values.

```
. fmm 2, startvalues(randomid, draws(5) seed(15)):
> poisson drvisits private medicaid c.age##c.age educ actlim chronic
```

(iteration log omitted)

```
Finite mixture model                                  Number of obs = 3,677
Log likelihood = -11502.686
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
1.Class	(base outcome)					
2.Class						
_cons	.877227	.0494614	17.74	0.000	.7802845	.9741696

```
Class:    1
Response: drvisits
Model:    poisson
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
drvisits						
private	.138229	.0247626	5.58	0.000	.0896951	.1867629
medicaid	.1269723	.0341525	3.72	0.000	.0600345	.19391
age	.2628874	.0466774	5.63	0.000	.1714014	.3543735
c.age#c.age	-.0017418	.0003108	-5.60	0.000	-.002351	-.0011326
educ	.0241679	.0030705	7.87	0.000	.0181499	.030186
actlim	.1831598	.0238817	7.67	0.000	.1363525	.2299671
chronic	.1970511	.0088783	22.19	0.000	.17965	.2144523
_cons	-8.051256	1.741677	-4.62	0.000	-11.46488	-4.637632

```
Class:    2
Response: drvisits
Model:    poisson
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
drvisits						
private	.2077415	.0306353	6.78	0.000	.1476974	.2677856
medicaid	.1071618	.0407211	2.63	0.008	.02735	.1869736
age	.3798087	.0562035	6.76	0.000	.269652	.4899655
c.age#c.age	-.0024869	.0003736	-6.66	0.000	-.0032191	-.0017547
educ	.029099	.003972	7.33	0.000	.021314	.0368841
actlim	.1244235	.0310547	4.01	0.000	.0635574	.1852895
chronic	.3191166	.0089757	35.55	0.000	.3015247	.3367086
_cons	-14.25713	2.101964	-6.78	0.000	-18.37691	-10.13736

The first table in the output provides the estimated coefficients in the multinomial logit model for the latent class probabilities. The next two tables are the results for the Poisson regression models for the first and second classes. The estimated coefficients from these tables are interpreted just as you would coefficients from `poisson`; see [R] **poisson**.

To better understand these classes, we use `estat lcmean` to estimate the marginal predicted counts (means) for each class.

```
. estat lcmean
Latent class marginal means                          Number of obs = 3,677
```

	Margin	Delta-method std. err.	z	P>|z|	[95% conf. interval]	
1						
drvisits	13.95943	.1767506	78.98	0.000	13.613	14.30585
2						
drvisits	3.801692	.0587685	64.69	0.000	3.686508	3.916876

We see that class 1 represents those who visit the doctor frequently and class 2 represents those who visit the doctor less frequently. We can use `estat lcprob` to estimate the proportion of individuals in each class.

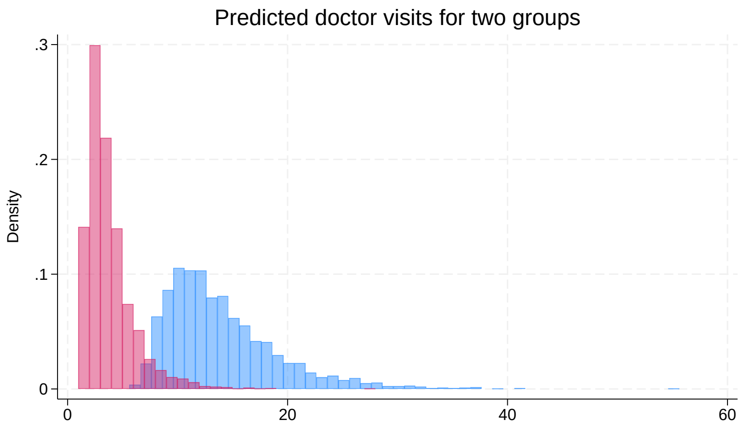
```
. estat lcprob
Latent class marginal probabilities                  Number of obs = 3,677
```

	Margin	Delta-method std. err.	[95% conf. interval]	
Class				
1	.2937527	.0102614	.2740502	.3142586
2	.7062473	.0102614	.6857414	.7259498

We find that about 29% of the population is in the group that visits the doctor frequently (class 1) and about 71% is in the group that visits the doctor less frequently (class 2).

We can visually compare the resulting distributions of the means by plotting the predicted number of doctor visits.

```
. predict mu*
(option mu assumed)
. twoway histogram mu1, width(1) color(stblue) fcolor(%50) lcolor(%50) ||
>        histogram mu2, width(1) color(stred)  fcolor(%50) lcolor(%50)
>        legend(off) title("Predicted doctor visits for two groups")
```



We can clearly see the two groups. The frequent user group exhibits more variability, which is expected in a Poisson process where the variance is equal to the mean.

References

Cameron, A. C., and P. K. Trivedi. 2022. *Microeconometrics Using Stata*. 2nd ed. College Station, TX: Stata Press.

Deb, P., and P. K. Trivedi. 1997. Demand for medical care by the elderly: A finite mixture approach. *Journal of Applied Econometrics* 12: 313–336. https://doi.org/10.1002/(SICI)1099-1255(199705)12:3<313::AID-JAE440>3.0.CO;2-G.

Also see

[FMM] **fmm intro** — Introduction to finite mixture models

[FMM] **fmm: poisson** — Finite mixtures of Poisson regression models

[FMM] **estat lcmean** — Latent class marginal means

[FMM] **estat lcprob** — Latent class marginal probabilities

[SEM] **Example 53g** — Finite mixture Poisson regression

[SEM] **Example 54g** — Finite mixture Poisson regression, multiple responses

[SEM] **gsem** — Generalized structural equation model estimation command

Description Remarks and examples References Also see

Description

In this example, we demonstrate how to fit a zero-inflated Poisson model as a two-component mixture model. We use `estat lcprob` to estimate marginal class probabilities and `estat lcmean` to estimate marginal predicted counts. A likelihood-ratio test is performed to compare models with and without predictors of class membership.

Remarks and examples

Two-component mixture models are often used to model counts that include book sales through direct mail (Wedel et al. 1993), healthcare utilization (Deb and Trivedi 1997), and modeling of risk behavior (Lanza, Kugler, and Mathur 2011). In the FMM framework, a zero-inflated count model is represented by a mixture of a component that models both zero and nonzero counts and a degenerate point mass distribution that models the zeros; see [FMM] **fmm: pointmass** for details.

The most popular zero-inflated count model is the zero-inflated Poisson (ZIP) model. Here we fit this model to the data on the number of fish caught by park visitors. Almost 57% of visitors reported zero catch, but we do not know whether they fished in the first place. In other words, zero counts can either be from a Poisson distribution or are hard zeros from a point mass distribution. Using a zero-inflated FMM, we can make probabilistic statements about which distribution a given zero came from.

Using `fish2.dta`, we fit a two-component mixture model where the nonfishing group (class 1) is modeled using a degenerate point mass distribution with the default value zero and the fishing group (class 2) is modeled using a Poisson distribution. For the latter group, we model the number of fish caught as a function of whether the visitor brought a boat (`boat`) and the number of persons in the party (`persons`).

By default, the reference probability is the class 1 probability. We specify `lcbase(2)` to make the reference probability be the probability for class 2. This will allow us to more easily compare the mixing proportions when we add covariates to model the probability of being in the nonfishing group.

```
. use https://www.stata-press.com/data/r19/fish2
(Fictional fishing data)
. fmm, lcbase(2): (pointmass count) (poisson count persons boat)
```

(iteration log omitted)

```
Finite mixture model                                    Number of obs = 250
Log likelihood = -882.31198
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
1.Class						
_cons	.0867958	.1390251	0.62	0.532	-.1856884	.35928
2.Class	(base outcome)					

```
Class:    2
Response: count
Model:    poisson
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
count						
persons	.750919	.0422907	17.76	0.000	.6680307	.8338072
boat	1.813785	.2648584	6.85	0.000	1.294672	2.332898
_cons	-2.024982	.2974941	-6.81	0.000	-2.608059	-1.441904

The first table in the output provides the estimated coefficients on the logit scale for the class probabilities. The coefficient on `1.Class` represents the probability of being in the nonfishing group which is about 52% [$\text{invlogit}(0.087) \approx 0.52$]. Because we have only two groups, the fishing fraction is 48%. Recall that the fraction of zeros in the data is 0.57, thus the model suggests that some zero counts are due to the Poisson component.

The second output table presents the results for the Poisson model component. The coefficients here are interpreted just as those from a standard Poisson regression; see [R] **poisson**. For example, we see that having a boat increases the expected number of fish caught by around six [$\exp(1.814) \approx 6.14$] for those who did fish, holding other covariates constant.

We store our estimates for later use.

```
. estimates store model1
```

In the model above, we did not model class probabilities. By modeling class probabilities with covariates, we can further differentiate between visitors who did not fish and those who fished without success. Here we make the mixing probability for the point mass component depend on covariates by using the `lcprob()` option with covariates `child` and `camper`. The default reference probability now switches to the Poisson component; therefore, we no longer need to specify `lcbase(2)`.

```
. fmm: (pointmass count, lcprob(child camper)) (poisson count persons boat)
```

(*iteration log omitted*)

```
Finite mixture model                                    Number of obs = 250
Log likelihood = -850.70142
```

	Coefficient	Std. err.	z	P>|z|	[95% conf. interval]	
1.Class						
child	1.602571	.2797719	5.73	0.000	1.054228	2.150913
camper	-1.015698	.365259	-2.78	0.005	-1.731593	-.2998039
_cons	-.4922872	.3114562	-1.58	0.114	-1.10273	.1181558
2.Class	(base outcome)					

```
Class:    2
Response: count
Model:    poisson
```

	Coefficient	Std. err.	z	P>|z|	[95% conf. interval]	
count						
persons	.8068853	.0453288	17.80	0.000	.7180424	.8957281
boat	1.757289	.2446082	7.18	0.000	1.277866	2.236713
_cons	-2.178472	.2860289	-7.62	0.000	-2.739078	-1.617865

The coefficients for the Poisson component are close to those from the previous model.

The coefficients of interest for the class 1 probability are both significant. A positive coefficient on the `child` variable means people with children in their party tended do to something other than fish. A negative coefficient on the `camper` variable means people camping at the park were more likely to go fishing.

Because we modeled the probability of being in the point mass component with covariates, calculating the marginal probabilities of belonging to a given component is more involved than before. We use `estat lcprob` to display marginal class probabilities on a probability scale.

```
. estat lcprob
Latent class marginal probabilities                     Number of obs = 250
```

	Margin	Delta-method std. err.	[95% conf. interval]	
Class				
1	.4786335	.0341083	.4125554	.5454678
2	.5213665	.0341083	.4545322	.5874446

We find that about 48% of the park visitors are in the nonfishing group, which is slightly lower than the 52% we found previously.

We can use `lrtest` to compare the current model with the previous one.

```
. lrtest model1 .

Likelihood-ratio test
Assumption: model1 nested within .

 LR chi2(2) =  63.22
Prob > chi2 = 0.0000
```

The likelihood-ratio test favors the model that includes covariates in the modeling of the probability of being in the nonfishing group.

We can also estimate the marginal predicted counts (means) for the fishing group using `estat lcmean`.

```
. estat lcmean

Latent class marginal means                          Number of obs = 250

Expression: Predicted mean (number of fish caught in class 2.Class),
            predict(outcome(count) class(2))
```

	Margin	Delta-method std. err.	z	P>|z|	[95% conf. interval]	
2						
count	6.490014	.2361623	27.48	0.000	6.027144	6.952884

The marginal predicted count for the fishing group is 6.49. This is much higher than the sample mean of 3.30 that is based on the fishing and nonfishing populations combined. If we were advertising fishing opportunities in the park, we know which number we would use!

References

Deb, P., and P. K. Trivedi. 1997. Demand for medical care by the elderly: A finite mixture approach. *Journal of Applied Econometrics* 12: 313–336. https://doi.org/10.1002/(SICI)1099-1255(199705)12:3<313::AID-JAE440>3.0.CO;2-G.

Lanza, S. T., K. C. Kugler, and C. Mathur. 2011. Differential effects for sexual risk behavior: An application of finite mixture regression. *Open Family Studies Journal* 4 (Suppl. 1-M9): 81–88. https://doi.org/10.2174/1874922401104010081.

Wedel, M., W. S. DeSarbo, J. R. Bult, and V. Ramaswamy. 1993. A latent class poisson regression model for heterogeneous count data. *Journal of Applied Econometrics* 8: 397–411. https://doi.org/10.1002/jae.3950080407.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[R] **zip** — Zero-inflated Poisson regression

Example 4 — Mixture cure models for survival data

Description Remarks and examples References Also see

Description

Cure models, or split-population models, are used to model survival data where a fraction of the population will never experience a failure. Mixture cure models represent the population as a combination of two types of individuals: a short-term survivor (noncured) group and a long-term survivor (cured) group. These models allow us to detect covariates associated with class membership (being cured or not) and to investigate the impact of covariates on the hazard for the noncured group as well.

In this example, we demonstrate how to fit a cure model as a two-component FMM with one component being a parametric survival model and one component being a point mass density that represents the cured group.

Remarks and examples

Implantation of intraocular lenses is a common surgery used to treat cataracts. One possible complication after this procedure is calcification of the lenses. Some patients will experience calcification during the follow-up period and some will not. Just because patients have not experienced calcification during the follow-up period does not mean that they truly are cured. It is still possible that they might experience calcification after the follow-up period ends. Thus, the cured group must be considered right-censored, with some individuals not observed to have calcification possibly belonging to this group.

In the language of FMM, we have two latent groups: a cured group and a noncured group. We know that patients who experience calcification are members of the noncured group. We do not know which group that patients who remain healthy belong to. That is, some of the patients we observe as healthy are truly cured, whereas others are members of the noncured group who are right-censored because they happened to not experience calcification during the study.

With a mixture cure model, we can predict the probability that an individual who did not experience calcification during the study is noncured. Let π be the probability of being in the noncured group, and let $S_1(t)$ be the survivor function for the noncured group. For the noncured group, the time to failure is modeled with a parametric distribution accounting for right-censoring, such as exponential or Weibull. If we let $S(t)$ be the probability of not failing before time t for an individual in the population, our model is

$$S(t) = (1 - \pi) + \pi S_1(t)$$

To illustrate the model, we use the artificial dataset, `lenses.dta`, with some of the characteristics of the calcification study described in Ma (2009). About 46% of the patients did not have postsurgery calcification of lenses during the follow-up period. We will predict how many of those are likely to have calcification after the follow-up period.

In our model, $S_1(t)$ is a Weibull using a proportional hazards parameterization. The covariates of interest are patient's sex (`sex`), patient's age at implantation divided by 10 (`age10`), and incision length (`inclength`).

The variable `fail` in the dataset contains an indicator for failure (occurrence of calcification). When `fail` $= 1$, we know that an individual belongs to the noncured group. When `fail` $= 0$, the individual is observed as healthy, but we cannot say they are a member of the cured group.

We first use stset to declare the data to be survival-time data. We specify t as the variable that contains analysis time and fail as the variable that indicates failure; see [ST] **stset** for details.

```
. use https://www.stata-press.com/data/r19/lenses
(Simulated calcification data)

. stset t, failure(fail)

Survival-time data settings

         Failure event: fail!=0 & fail<.
Observed time interval: (0, t]
     Exit on or before: failure

--------------------------------------------------------------------------
        770  total observations
          0  exclusions
--------------------------------------------------------------------------
        770  observations remaining, representing
        415  failures in single-record/single-failure data
 20,133.467  total analysis time at risk and under observation
                                                At risk from t =         0
                                     Earliest observed entry t =         0
                                          Last observed exit t =        36
```

To model time-to-calcification in the noncured group, we fit a Weibull model for right-censored data where the dependent variable is the time variable. This includes the patients observed as noncured and those who appear healthy. To model the probability of being cured, we use a point mass density at fail $= 0$ because this indicates that calcification was not observed. See [FMM] **fmm: pointmass** for details about the point mass distribution.

```
. fmm: (pointmass fail) (streg inclength i.sex age10, distribution(weibull))
  (iteration log omitted)

Finite mixture model                                    Number of obs = 770
Log likelihood = -1980.1495
```

	Coefficient	Std. err.	z	P>|z|	[95% conf. interval]	
1.Class	(base outcome)					
2.Class						
_cons	1.01863	.2703434	3.77	0.000	.4887664	1.548493

```
Class:     2
Response:  _t                                     No. of failures =       415
Model:     streg, dist(weibull)                   Time at risk    = 20,133.47
```

	Coefficient	Std. err.	z	P>|z|	[95% conf. interval]	
_t						
inclength	-.5922698	.2273662	-2.60	0.009	-1.037899	-.1466402
sex						
male	.3314051	.1259957	2.63	0.009	.0844581	.5783522
age10	.1600672	.032798	4.88	0.000	.0957843	.2243502
_cons	-4.939691	.940024	-5.25	0.000	-6.782104	-3.097278
/_t						
ln_p	.4683771	.058332			.3540485	.5827056

The first table in the output shows the estimated coefficient on the logit scale for the class 2 (noncured group) probability. This probability is 0.73 [invlogit(1.019) $\approx$ 0.73], which implies that the probability of being in the cured group is 0.27.

The second table presents the results for the Weibull regression model for the noncured group. We see that longer incisions decrease the hazard of calcification, while being male and being older increase the hazard of calcification.

We may want to know the probability that patients who have not experienced calcification will do so in the future. We can predict the posterior probability of being in class 2. We list the first 10 patients for the cured group.

```
. predict pprob2, classposterior class(2.Class)
. sort fail, stable
. list fail pprob2 in 1/10

     ┌───────────────────┐
     │ fail       pprob2 │
     ├───────────────────┤
  1. │    0     .2569577 │
  2. │    0     .4447927 │
  3. │    0     .3233174 │
  4. │    0     .4677424 │
  5. │    0     .4549083 │
     ├───────────────────┤
  6. │    0     .4183038 │
  7. │    0     .3161573 │
  8. │    0     .4540032 │
  9. │    0     .2782425 │
 10. │    0     .5745969 │
     └───────────────────┘
```

We see that the posterior probability of having calcification in the future is over 50% for the last patient.

We generate an indicator variable `prfail` that takes on value 1 if the posterior probability of calcification is greater than 50% and zero otherwise. We construct a classification table where we tabulate our variable against the indicator of failure `fail`.

```
. generate prfail = pprob2 > .5
. tabulate prfail fail
           │  failed=1, didn't
           │       fail=0
    prfail │         0          1 │     Total
───────────┼──────────────────────┼──────────
         0 │       257          0 │       257
         1 │        98        415 │       513
───────────┼──────────────────────┼──────────
     Total │       355        415 │       770
```

Out of 355 individuals who did not experience calcification during the study, we estimate that 98 are more likely than not to have calcification in the future.

References

Lambert, P. C. 2007. Modeling of the cure fraction in survival studies. *Stata Journal* 7: 351–375.

Ma, S. 2009. Cure model with current status data. *Statistica Sinica* 19: 233–249.

Also see

[FMM] **fmm** — Finite mixture models using the fmm prefix

[FMM] **fmm postestimation** — Postestimation tools for fmm

Glossary

categorical latent variable. A categorical latent variable has levels that represent unobserved groups in the population. Latent classes are identified with the levels of the categorical latent variables and may represent healthy and unhealthy individuals, consumers with different buying preferences, or different motivations for delinquent behavior.

class model. A class model is a regression model that is applied to one component in a mixture model. In the absence of covariates, the regression model reduces to a distribution function.

Class model is also referred to in the literature as a "component model", "component density", or "component distribution".

class probability. In the context of FMM, the probability of belonging to a given class. `fmm` uses multinomial logistic regression to model class probabilities.

Class probability is also referred to in the literature as a "latent class probability", "component probability", "mixture component probability", "mixing probability", "mixing proportion", "mixing weight", or "mixture probability".

EM algorithm. See *expectation-maximization algorithm*.

entropy. A measure of separation between latent classes. It ranges from 0 to 1, and values closer to 1 indicate better separation between latent classes.

expectation-maximization algorithm. In the context of FMM, an iterative procedure for refining starting values before maximizing the likelihood. The EM algorithm uses the complete-data likelihood as if we have observed values for the latent class indicator variable.

finite mixture model. A finite mixture model (FMM) is a statistical model that assumes the presence of unobserved groups, called latent classes, within an overall population. Each latent class can be fit with its own regression model, which may have a linear or generalized linear response function. We can compare models with differing numbers of latent classes and different sets of constraints on parameters to determine the best fitting model. For a given model, we can compare parameter estimates across classes. We can estimate the proportion of the population in each latent class, and we can predict the probabilities that the observations in our sample belong to each latent class.

FMM. See *finite mixture model*.

generalized linear response functions. Generalized linear response functions include linear functions and include functions such as probit, logit, multinomial logit, ordered probit, ordered logit, Poisson, and more.

These generalized linear functions are described by a link function $g(\cdot)$ and statistical distribution F. The link function $g(\cdot)$ specifies how the response variable y_i is related to a linear equation of the explanatory variables, $\mathbf{x}_i\boldsymbol{\beta}$, and the family F specifies the distribution of y_i:

$$g\{E(y_i)\} = \mathbf{x}_i\boldsymbol{\beta} \qquad y_i \sim F$$

If we specify that $g(\cdot)$ is the identity function and F is the Gaussian (normal) distribution, then we have linear regression. If we specify that $g(\cdot)$ is the logit function and F the Bernoulli distribution, then we have logit (logistic) regression.

In this generalized linear structure, the family may be Gaussian, gamma, Bernoulli, binomial, Poisson, negative binomial, ordinal, or multinomial. The link function may be the identity, log, logit, probit, or complementary log–log.

latent class. A latent class is an unobserved group identified by a level of a categorical latent variable.

Latent class is also referred to in the literature as a “class”, “group”, “type”, or “mixture component”.

latent variable. See *categorical latent variable*.

pointmass density. In the context of FMM, a degenerate distribution that takes on a single integer value with probability one. A pointmass density is used in combination with other FMM distributions to model, most commonly, zero-inflated outcomes.

Subject and author index

See the combined subject index and the combined author index in the *Stata Index*.