cmmixlogit — Mixed logit choice model

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

Description

`cmmixlogit` fits a mixed logit choice model, also known as a mixed multinomial logit model or random-parameter logit model, which uses random coefficients to model the correlation of choices across alternatives. The random coefficients are on variables that vary across both cases and alternatives known as alternative-specific variables.

The correlation of choices across alternatives relaxes the independence of irrelevant alternatives (IIA) property imposed by the conventional multinomial logit model fit by `mlogit` and the conditional logit choice model fit by `cmclogit`.

For a mixed logit choice model for panel data, see [CM] **cmxtmixlogit**.

Quick start

Mixed logit regression of `y` on `x1`, where the coefficients on `x1` are assumed random normal, using `cmset` data

```
cmmixlogit y, random(x1)
```

Same as above, and include levels of case-specific covariate `a`

```
cmmixlogit y, random(x1) casevars(i.a)
```

Same as above, and add covariate `x2`, whose coefficients are random triangular

```
cmmixlogit y, random(x1) random(x2, triangle) casevars(i.a)
```

Mixed logit model of `y` on `x1`, `x2`, and `x3`, where the random coefficients for `x2` and `x3` are correlated

```
cmmixlogit y x1, random(x2 x3, correlated)
```

Same as above, but omit the alternative-specific constants

```
cmmixlogit y x1, random(x2 x3, correlated) noconstant
```

Menu

Statistics > Choice models > Mixed logit model

Syntax

`cmmixlogit` *depvar* [*indepvars*] [*if*] [*in*] [*weight*] [, *options*]

depvar equal to 1 identifies the chosen alternatives, whereas a 0 indicates the alternatives that were not chosen. There can be only one chosen alternative for each case.

indepvars specifies the alternative-specific covariates that have fixed coefficients.

options	Description
Model	
`casevars(`*varlist*`)`	case-specific variables
`random(`*varlist* [, *distribution*]`)`	specify variables that are to have random coefficients and the coefficients' distribution
`corrmetric(`*metric*`)`	correlation metric for correlated random coefficients
`basealternative(`#|*lbl*|*str*`)`	alternative used for normalizing location
`noconstant`	suppress the alternative-specific constant terms
`altwise`	use alternativewise deletion instead of casewise deletion
`constraints(`*constraints*`)`	apply specified linear constraints
SE/Robust	
`vce(`*vcetype*`)`	*vcetype* may be `oim`, `robust`, `cluster` *clustvar*, `opg`, `bootstrap`, or `jackknife`
Reporting	
`level(`#`)`	set confidence level; default is `level(95)`
`or`	report odds ratios and relative-risk ratios
`nocnsreport`	do not display constraints
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Integration	
`intmethod(`*seqspec*`)`	specify point set for Monte Carlo integration
`intpoints(`#`)`	specify number of points in each sequence
`intburn(`#`)`	specify starting index in the Hammersley or Halton sequence
`intseed(`#`)`	specify random-number seed for pseudo–random sequence
`favor(speed|space)`	favor speed or space when generating integration points
Maximization	
maximize_options	control the maximization process; seldom used
`collinear`	keep collinear variables
`coeflegend`	display legend instead of statistics

distribution	Description
`normal`	Gaussian-distributed random coefficients; the default
`correlated`	correlated Gaussian-distributed random coefficients
`lnormal`	lognormal distributed random coefficients
`tnormal`	truncated normal distributed random coefficients
`uniform`	uniform distributed random coefficients
`triangle`	triangular distributed random coefficients

metric	Description
`correlation`	standard deviation and correlation; the default
`covariance`	variance and covariance
`cholesky`	Cholesky factor

seqspec is

seqtype[, antithetics | mantithetics]

seqtype	Description
hammersley	Hammersley point set; the default
halton	Halton point set
random	uniform pseudo–random point set

You must cmset your data before using cmmixlogit; see [CM] **cmset**.

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

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

Weights are not allowed with the bootstrap prefix; see [R] **bootstrap**.

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.

Options

Model

casevars(*varlist*) specifies the case-specific variables that are constant for each case. If there are a maximum of A alternatives, there will be $A-1$ sets of coefficients associated with casevars().

random(*varlist* [, *distribution*]) specifies the alternative-specific variables that are to have random coefficients and optionally the assumed distribution of the random coefficients. The default distribution is normal, meaning Gaussian-distributed random coefficients. *distribution* may also be correlated, lnormal, tnormal, uniform, or triangle. random() may be specified more than once to specify different sets of variables that correspond to different coefficient distributions.

corrmetric(*metric*) specifies the estimation metric for correlated random coefficients. corrmetric(correlation), the default, estimates the standard deviations and correlations of the random coefficients. corrmetric(covariance) estimates variances and covariances, and corrmetric(cholesky) estimates Cholesky factors. corrmetric() is allowed only when random(*varlist*, correlated) is specified.

basealternative(# | *lbl* | *str*) sets the alternative that normalizes the level of utility. The base alternative may be specified as a number when the alternatives variable is numeric, as a label when it is numeric and has a value label, or as a string when it is a string variable. The default is the alternative with the highest frequency of being chosen. This option is ignored if neither alternative-specific constants nor case-specific variables are specified.

noconstant suppresses the $A-1$ alternative-specific constant terms.

altwise specifies that alternativewise deletion be used when omitting observations because of missing values in your variables. The default is to use casewise deletion; that is, the entire group of observations making up a case is omitted if any missing values are encountered. This option does not apply to observations that are excluded by the if or in qualifier or the by prefix; these observations are always handled alternativewise regardless of whether altwise is specified.

constraints(*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`), that allow for intragroup correlation (`cluster` *clustvar*), and that use bootstrap or jackknife methods (`bootstrap`, `jackknife`); see [R] ***vce_option***.

Specifying `vce(robust)` is equivalent to specifying `vce(cluster` *caseid*`)`.

If specifying `vce(bootstrap)` or `vce(jackknife)`, you must also specify `basealternative()`.

Reporting

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

`or` reports the estimated coefficients transformed to odds ratios for alternative-specific variables and relative-risk ratios for case-specific variables. That is, e^b rather than b is reported. Standard errors and confidence intervals are transformed accordingly. This option affects how results are displayed, not how they are estimated. `or` may be specified at estimation or when replaying previously estimated results.

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

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**.

Integration

`intmethod(`*seqtype*`[ , antithetics | mantithetics ])` specifies the method of generating the point sets used in the Monte Carlo integration. `intmethod(hammersley)`, the default, uses the Hammersley sequence; `intmethod(halton)` uses the Halton sequence; and `intmethod(random)` uses a sequence of uniform random numbers.

`antithetics` and `mantithetics` specify that a unidimensional antithetic sequence or a multidimensional antithetic sequence, respectively, be generated instead of the standard implementation of the requested *seqtype*. These methods improve the accuracy of the Monte Carlo integration at the cost of additional computation time; see *Methods and formulas*.

`intpoints(#)` specifies the number of points to use in the Monte Carlo integration. The default number of points is a function of model complexity and integration method. If `intmethod(hammersley)` or `intmethod(halton)` is used, the default is $500 + \text{floor}[2.5\sqrt{N_c\{\ln(r+5)+v\}}]$, where N_c is the number of cases, r is the number of random coefficients in the model, and v is the number of variance parameters. If `intmethod(hammersley, mantithetics)` or `intmethod(halton, mantithetics)` is used, the number of integration points is $250 + \text{floor}[0.5\sqrt{N_c\{\ln(r+5)+v\}}]$. If `intmethod(random)` is used, the number of points is twice the number of points used by `intmethod(hammersley)` and `intmethod(halton)`. Larger values of `intpoints()` provide better approximations of the log likelihood at the cost of additional computation time.

`intburn(#)` specifies where in the Hammersley or Halton sequence to start, which helps reduce the correlation between the sequences of each dimension. The default is to discard the first n initial elements from each sequence, where n is the largest prime used to generate the sequences. This option may not be specified with `intmethod(random)`.

`intseed(#)` specifies the seed to use for generating uniform pseudo–random sequences. This option may be specified only with `intmethod(random)`. # must be an integer between 0 and $2^{31} - 1$. The default is to use the current seed value from Stata's uniform random-number generator; see [R] **set seed**.

`favor(speed | space)` instructs `cmmixlogit` to favor either speed or space when generating the integration points. `favor(speed)` is the default. When favoring speed, the integration points are generated once and stored in memory, thus increasing the speed of evaluating the likelihood. This speed increase can be seen when there are many cases or when the user specifies many integration points in `intpoints(#)`. When favoring space, the integration points are generated repeatedly with each likelihood evaluation.

Maximization

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

Setting the optimization type to `technique(bhhh)` resets the default *vcetype* to `vce(opg)`.

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

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

Remarks and examples

`cmmixlogit` fits a mixed logit choice model, in the following simply referred to as a mixed logit model. The mixed logit model is most frequently used to model the probability that an individual chooses one of several unordered alternatives. It is also known as the mixed multinomial logit model (McFadden and Train 2000), the random-parameters logit model (Cameron and Trivedi 2005), the logit kernel model (Ben-Akiva, Bolduc, and Walker 2001), or the hybrid logit model (Ben-Akiva et al. 1997).

The mixed logit model is often used in the context of discrete choice models. These models represent how decision makers choose among a finite set of alternatives. The decision maker, called a case, is often an individual. The mixed logit model can incorporate attributes that vary across individuals, known as case-specific variables. Income, educational attainment, and age are examples of case-specific variables.

The model can also incorporate observed attributes that vary by alternative or by alternative and individual, known as alternative-specific variables. The size of the lake at a fishing site is an example of an alternative-specific covariate that varies only by alternative. The travel distance to any given fishing site is an example of an alternative-specific covariate that varies by alternative and individual.

In the mixed logit model, the coefficients on alternative-specific variables can be treated as fixed or random. Specifying random coefficients can model correlation of choices across alternatives, thereby relaxing the IIA property that is imposed by the multinomial logit models described in [R] **mlogit** and the conditional logit models described in [R] **clogit** and [CM] **cmclogit**. In this sense, the mixed logit model fit by `cmmixlogit` is more flexible than the models fit by `mlogit`, `clogit`, and `cmclogit`.

McFadden and Train (2000) show that the mixed logit model can approximate a wide class of choice representations. Although the mixed logit model was derived under a utility framework and is most often used for these applications, it also can be applied in contexts that lack this individual-choice motivation, for example, classification problems. For an introduction to mixed logit models, see Cameron

and Trivedi (2005) and Train (2009). See Hole (2007) for a previous implementation of mixed logit models via the community-contributed `mixlogit` command. For the panel-data mixed logit model, see [CM] **cmxtmixlogit**.

In discrete choice, an individual chooses the alternative that yields the highest value of an unobserved ranking index known as utility. Utility is a latent variable that is a function of observed attributes of the individuals, the alternatives, random coefficients, and a random component. In other contexts, such as classification analysis, utility is just an unobserved random index. We call it utility here because this model is most frequently applied to discrete choice data.

For the mixed logit model, the utility that individual i receives from alternative a, $a = 1, 2, \ldots, A$, denoted by U_{ia}, is

$$U_{ia} = \mathbf{x}_{ia}\boldsymbol{\beta}_i + \mathbf{w}_{ia}\boldsymbol{\alpha} + \mathbf{z}_i\boldsymbol{\delta}_a + \epsilon_{ia}$$

$\boldsymbol{\beta}_i$ are random coefficients that vary over individuals in the population, and $\mathbf{x}_{ia}$ is a vector of alternative-specific variables. $\boldsymbol{\alpha}$ are fixed coefficients on $\mathbf{w}_{ia}$, a vector of alternative-specific variables. $\boldsymbol{\delta}_a$ are fixed alternative-specific coefficients on $\mathbf{z}_i$, a vector of case-specific variables. ϵ_{ia} is a random term that follows a type I extreme value distribution.

`cmmixlogit` estimates the fixed coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\delta}_a$ and the parameters of $f(\boldsymbol{\beta})$, the distribution of the random coefficients. The mixed logit model can estimate only the parameters of $f(\boldsymbol{\beta})$, not the $\boldsymbol{\beta}_i$ per se. For example, if the random coefficients $\boldsymbol{\beta}_i$ follow a normal distribution, $\boldsymbol{\beta}_i \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, then the mixed logit model estimates $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$.

Note that only the rank order of the utilities for each alternative matters; that is, the location and scale of utility are irrelevant. The data reveal only the chosen alternative, so we must normalize for location by taking differences with respect to a base alternative k. The assumed type I extreme value distribution implies that the difference in the errors for alternative a and base k, $\epsilon_{ia} - \epsilon_{ik}$, follows a logistic distribution. Assuming a standard logistic distribution normalizes for scale.

The choice probabilities are the standard logistic probabilities integrated over the density $f(\boldsymbol{\beta})$. That is, the probability of choosing alternative a for individual i is

$$P_{ia} = \int P_{ia}(\boldsymbol{\beta}) f(\boldsymbol{\beta}) d\boldsymbol{\beta} \tag{1}$$

where

$$P_{ia}(\boldsymbol{\beta}) = \frac{e^{\mathbf{x}_{ia}\boldsymbol{\beta}_i + \mathbf{w}_{ia}\boldsymbol{\alpha} + \mathbf{z}_i\boldsymbol{\delta}_a}}{\sum_{a=1}^{A} e^{\mathbf{x}_{ia}\boldsymbol{\beta}_i + \mathbf{w}_{ia}\boldsymbol{\alpha} + \mathbf{z}_i\boldsymbol{\delta}_a}}$$

are the logistic probabilities, evaluated at parameters $\boldsymbol{\beta}$.

The integral in (1) must be approximated because it has no closed-form solution. `cmmixlogit` approximates (1) by simulation and estimates the model parameters by maximum simulated likelihood (MSL). Consistency of the MSL estimator requires that the number of random draws in the simulation method be sufficiently large. More draws will produce more precise estimates by reducing approximation error, at the cost of increased computation time. See *Methods and formulas* for further details, and see Cameron and Trivedi (2005) for an introduction to MSL estimation.

▷ Example 1: Mixed logit model with fixed and random coefficients

inschoice.dta records information about available insurance plans and the selected plan for 250 individuals. Each individual selected an insurance plan from the five alternatives that are recorded in the insurance variable. The binary variable choice records the chosen alternative; choice is 1 for the chosen alternative and 0 otherwise. For each individual, we have one observation for each alternative. Here are the data for the first two individuals:

```
. use https://www.stata-press.com/data/r19/inschoice
(Fictional health insurance data)
. list in 1/10, sepby(id) abbreviate(10)
```

	id	premium	deductible	income	insurance	choice
1.	1	2.87	1.70	5.74	Health	1
2.	1	3.13	2.14	5.74	HCorp	0
3.	1	2.03	2.26	5.74	SickInc	0
4.	1	1.65	2.94	5.74	MGroup	0
5.	1	0.87	3.56	5.74	MoonHealth	0
6.	2	3.52	1.24	2.89	Health	0
7.	2	3.23	1.52	2.89	HCorp	0
8.	2	2.81	2.31	2.89	SickInc	0
9.	2	1.04	2.58	2.89	MGroup	1
10.	2	0.93	3.17	2.89	MoonHealth	0

Insurance premiums (premium) and deductibles (deductible) vary over alternatives and are thus alternative-specific variables. In this example, they also vary over individuals. Income (income) varies only over individuals and is thus a case-specific variable.

We wish to estimate the effect of health insurance premiums, insurance deductibles, and personal income on the choice of health insurance plans. We assume that preferences with respect to deductibles vary over individuals in the population but that preferences with respect to premiums are constant over individuals in the population.

Before we can fit a model, we must cmset our data. The first argument to cmset is the case ID variable. For these data, it is the variable id, which identifies individuals. The second argument is the alternatives variable, which identifies the alternatives that could have been chosen. In this instance, it is the variable insurance, which identifies the insurance plans available.

```
. cmset id insurance
     Case ID variable: id
Alternatives variable: insurance
```

We fit a model for this outcome by using cmmixlogit. We specify random(deductible) to include random coefficients on deductible, and we include premium as *indepvar* to include a fixed coefficient on premium.

```
. cmmixlogit choice premium, random(deductible)
Fitting fixed parameter model:
Fitting full model:
Iteration 0:  Log simulated-likelihood = -296.14208  (not concave)
Iteration 1:  Log simulated-likelihood = -295.74886
Iteration 2:  Log simulated-likelihood = -295.05013
Iteration 3:  Log simulated-likelihood = -295.04639
Iteration 4:  Log simulated-likelihood = -295.04639
Mixed logit choice model                    Number of obs      =      1,250
Case ID variable: id                        Number of cases    =        250
Alternatives variable: insurance            Alts per case: min =          5
                                                           avg =        5.0
                                                           max =          5
Integration sequence:      Hammersley
Integration points:              567        Wald chi2(2)       =      99.32
Log simulated-likelihood = -295.04639       Prob > chi2        =     0.0000
```

choice	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
insurance						
premium	-2.672185	.269542	-9.91	0.000	-3.200478	-2.143893
deductible	-1.109975	.337406	-3.29	0.001	-1.771278	-.4486709
/Normal						
sd(deducti~e)	.8886822	.3641774			.3980416	1.984104
Health						
_cons	.5208962	.2979741	1.75	0.080	-.0631222	1.104915
HCorp	(base alternative)					
SickInc						
_cons	-.8426238	.2909169	-2.90	0.004	-1.412811	-.2724371
MGroup						
_cons	-2.107124	.4436973	-4.75	0.000	-2.976755	-1.237494
MoonHealth						
_cons	-3.36121	.6795993	-4.95	0.000	-4.6932	-2.02922

```
LR test vs. fixed parameters: chibar2(01) =     2.99  Prob >= chibar2 = 0.0420
```

The estimated fixed coefficient on premium is −2.67, so an increase in a plan's premium reduces the probability that it is chosen. The estimated mean of the normally distributed coefficients on deductible is −1.11. The estimated standard deviation of these random coefficients is 0.89, indicating heterogeneity across individuals in the population with respect to the effect of a plan's deductible.

The likelihood-ratio (LR) test in the footer shows the result of a test against a model with only fixed coefficients and indicates that we can reject the null hypothesis that the coefficients on deductible are fixed.

cmmixlogit computes the likelihood using Monte Carlo integration. Several options are available to control how the integration is done. In almost all cases, you will never have to change any of the defaults. However, there is one of them, intpoints(#), the number of integration points, that you may

sometimes want to set. The default value is set to a number that is typically adequate, but when you have a model you consider final, you should run the estimation again, setting `intpoints()` to a bigger number to confirm the numerical soundness of the estimates.

We see from the header on the earlier output that `cmmixlogit` used 567 integration points by default. Let's run it again using `intpoints(1000)`.

```
. cmmixlogit choice premium, random(deductible) intpoints(1000)
Fitting fixed parameter model:
Fitting full model:
Iteration 0:  Log simulated-likelihood = -296.14179  (not concave)
Iteration 1:  Log simulated-likelihood = -295.74946
Iteration 2:  Log simulated-likelihood = -295.05088
Iteration 3:  Log simulated-likelihood = -295.04713
Iteration 4:  Log simulated-likelihood = -295.04713
Mixed logit choice model                    Number of obs      =      1,250
Case ID variable: id                        Number of cases    =        250
Alternatives variable: insurance            Alts per case: min =          5
                                                           avg =        5.0
                                                           max =          5
Integration sequence:      Hammersley
Integration points:              1000       Wald chi2(2)       =      99.30
Log simulated-likelihood = -295.04713       Prob > chi2        =     0.0000
```

choice	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
insurance						
premium	-2.672171	.2695616	-9.91	0.000	-3.200502	-2.14384
deductible	-1.109943	.337404	-3.29	0.001	-1.771243	-.4486433
/Normal						
sd(deducti~e)	.8885033	.3643278			.3977651	1.984684
Health						
_cons	.5208928	.2979772	1.75	0.080	-.0631319	1.104917
HCorp	(base alternative)					
SickInc						
_cons	-.8426217	.2909165	-2.90	0.004	-1.412808	-.2724358
MGroup						
_cons	-2.107116	.4437085	-4.75	0.000	-2.976769	-1.237463
MoonHealth						
_cons	-3.361211	.6796514	-4.95	0.000	-4.693304	-2.029119

```
LR test vs. fixed parameters: chibar2(01) =     2.99  Prob >= chibar2 = 0.0420
```

The estimates are almost exactly the same. It does not matter which output we report, but we will use the last one. See the discussion in *Setting the number of integration points* in [CM] **Intro 5** for more information.

◁

❑ Technical note

The LR test vs. fixed parameters is a test of sd(deductible) = 0. This is a boundary test and thus requires careful consideration concerning the calculation of its p-value. In particular, the null distribution of the LR test statistic is not the usual χ^2_1 but rather is a 50:50 mixture of a χ^2_0 (point mass at 0) and a χ^2_1, denoted as $\overline{\chi}^2_{01}$. See Gutierrez, Carter, and Drukker (2001) for more details.

❑

Based on the model we just fit and assuming that we have a random or otherwise representative sample of individuals, we can use margins to estimate the proportion of people who will select each insurance plan.

```
. margins
Predictive margins                                      Number of obs = 1,250
Model VCE: OIM
Expression: Pr(insurance), predict()
```

	Margin	Delta-method std. err.	z	P>|z|	[95% conf.	interval]
_outcome						
Health	.2039552	.0224378	9.09	0.000	.1599779	.2479325
HCorp	.2573735	.0240836	10.69	0.000	.2101705	.3045764
SickInc	.219662	.0223481	9.83	0.000	.1758606	.2634634
MGroup	.187416	.0218151	8.59	0.000	.1446592	.2301729
MoonHealth	.1315932	.0192139	6.85	0.000	.0939346	.1692518

We expect about 26% of individuals to select the HCorp insurance plan. Suppose that HCorp is planning to increase their premiums by 10%. What effect would we expect this to have on the percentage of individuals selecting each alternative? We estimate this by using the at(premium=generate(premium*1.10)) and alternative(HCorp) options with margins.

```
. margins, at(premium=generate(premium*1.10)) alternative(HCorp)
Predictive margins                                      Number of obs = 1,250
Model VCE: OIM
Expression:  Pr(insurance), predict()
Alternative: HCorp
At: premium = premium*1.10
```

	Margin	Delta-method std. err.	z	P>|z|	[95% conf.	interval]
_outcome						
Health	.2287178	.0240772	9.50	0.000	.1815274	.2759083
HCorp	.1751858	.0188653	9.29	0.000	.1382104	.2121612
SickInc	.2439653	.0238211	10.24	0.000	.1972769	.2906538
MGroup	.2073507	.0233282	8.89	0.000	.1616282	.2530732
MoonHealth	.1447803	.020706	6.99	0.000	.1041972	.1853634

If HCorp makes this planned increase to their premiums, we expect that only about 18% of individuals would select their plan.

We can use margins to answer many other interesting questions as well. See [CM] **Intro 1** and [CM] **margins** for more information and examples.

▷ Example 2: Correlated random coefficients

Continuing with example 1, we now assume that preferences for premiums also vary, and we estimate the parameters of a model with random coefficients on both `premium` and `deductible`. We allow the random coefficients to be correlated, assuming a multivariate normal distribution, by specifying `random(deductible premium, correlated)`.

```
. cmmixlogit choice, random(deductible premium, correlated)
Fitting fixed parameter model:
Fitting full model:
Iteration 0:  Log simulated-likelihood = -295.85468  (not concave)
Iteration 1:  Log simulated-likelihood =  -295.8229
Iteration 2:  Log simulated-likelihood = -294.48628  (not concave)
Iteration 3:  Log simulated-likelihood =  -294.3334
Iteration 4:  Log simulated-likelihood = -294.20658
Iteration 5:  Log simulated-likelihood = -294.04148
Iteration 6:  Log simulated-likelihood = -294.03592
Iteration 7:  Log simulated-likelihood = -294.03592
Refining estimates:
Iteration 0:  Log simulated-likelihood = -294.03592
Iteration 1:  Log simulated-likelihood = -294.03592
Mixed logit choice model                    Number of obs      =      1,250
Case ID variable: id                        Number of cases    =        250
Alternatives variable: insurance            Alts per case: min =          5
                                                           avg =        5.0
                                                           max =          5
Integration sequence:      Hammersley
Integration points:              588        Wald chi2(2)       =      43.46
Log simulated-likelihood = -294.03592       Prob > chi2        =     0.0000
```

choice	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
insurance						
deductible	-1.323078	.4620074	-2.86	0.004	-2.228596	-.4175604
premium	-3.043368	.4667707	-6.52	0.000	-3.958222	-2.128514
/Normal						
sd(deducti~e)	1.250886	.6835422			.4286292	3.650512
corr(deduc~e, premium)	.5492364	.5488277	1.00	0.317	-.7273401	.9736263
sd(premium)	1.066745	.6118013			.3466394	3.28279
Health						
_cons	.4995312	.3252348	1.54	0.125	-.1379174	1.13698
HCorp	(base alternative)					
SickInc						
_cons	-.8885082	.3081869	-2.88	0.004	-1.492544	-.2844729
MGroup						
_cons	-2.302517	.5068546	-4.54	0.000	-3.295934	-1.309101
MoonHealth						
_cons	-3.692866	.8062486	-4.58	0.000	-5.273085	-2.112648

```
LR test vs. fixed parameters: chi2(3) =       5.01        Prob > chi2 = 0.1713
Note: LR test is conservative and provided only for reference.
```

The estimated means of the random coefficients on `deductible` and `premium` are −1.32 and −3.04, respectively. Beneath the estimated means, we see the estimated standard deviations of the random coefficients and their estimated correlation, which are 1.25, 1.07, and 0.55, respectively. The high standard errors on these parameters indicate that they are not precisely estimated.

◁

▷ Example 3: Lognormal random coefficients and case-specific variables

In the previous example, we assumed that the coefficients on `premium` are normally distributed. Assuming a normal distribution implies that the random coefficients could be both positive and negative. However, it is typically more plausible to assume that increasing prices do not have positive effects on the probability of choosing a corresponding alternative. We can constrain the `premium` coefficients to be negative using a lognormal distribution.

Because the lognormal distribution is defined only over positive real values, the coefficient values coming from this distribution will only be positive. We want the coefficients to be negative, so we reverse the sign of `premium`. Then positive coefficients for `-premium` are the same as negative coefficients for `premium`.

```
. generate negpremium = -premium
```

We again assume a normal distribution for random coefficients on `deductible`, and we include `income` as a case-specific variable in the `casevars()` option.

```
. cmmixlogit choice, random(deductible) random(negpremium, lnormal)
> casevars(income)

Fitting fixed parameter model:

Fitting full model:

Iteration 0:  Log simulated-likelihood = -290.56142
Iteration 1:  Log simulated-likelihood = -288.84285
Iteration 2:  Log simulated-likelihood =  -288.7695
Iteration 3:  Log simulated-likelihood = -288.76877
Iteration 4:  Log simulated-likelihood = -288.76877

Mixed logit choice model                    Number of obs      =      1,250
Case ID variable: id                        Number of cases    =        250

Alternatives variable: insurance            Alts per case: min =          5
                                                           avg =        5.0
                                                           max =          5

Integration sequence:      Hammersley
Integration points:               579       Wald chi2(6)       =      84.14
Log simulated-likelihood = -288.76877       Prob > chi2        =     0.0000
```

choice	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
insurance						
deductible	-1.141117	.3648257	-3.13	0.002	-1.856162	-.426072
negpremium	1.065557	.1175425	9.07	0.000	.8351775	1.295936
/Normal						
sd(deducti~e)	.7946425	.4502895			.2617175	2.412742
/Lognormal						
sd(negprem~m)	.3132935	.1398278			.1306312	.7513737
Health						
income	.1434246	.164893	0.87	0.384	-.1797597	.4666089
_cons	-.212565	.8944708	-0.24	0.812	-1.965695	1.540566
HCorp	(base alternative)					
SickInc						
income	-.3091895	.1557857	-1.98	0.047	-.6145239	-.003855
_cons	.6005832	.7992584	0.75	0.452	-.9659346	2.167101
MGroup						
income	-.3490833	.2014309	-1.73	0.083	-.7438807	.0457141
_cons	-.6984732	1.006771	-0.69	0.488	-2.671708	1.274762
MoonHealth						
income	-.506413	.2502777	-2.02	0.043	-.9969483	-.0158776
_cons	-1.463112	1.267173	-1.15	0.248	-3.946725	1.020502

```
LR test vs. fixed parameters: chi2(2) =       4.33        Prob > chi2 = 0.1149

Note: LR test is conservative and provided only for reference.
```

The estimated coefficient for `income` (our case-specific variable) shows that individuals are more likely to choose insurance plan `Health` over `HCorp` as income increases. The remaining alternatives are less likely to be chosen over `HCorp` as income increases. However, note that the coefficients on `income` for `Health` and for `MGroup` are not significantly different from 0 at the 5% level.

Because we assumed a lognormal distribution for the random coefficients on `negpremium`, we have to transform the estimated mean and standard deviation before we can interpret them. The lognormal distribution is parameterized in terms of the underlying normal distribution. The estimates shown above are the mean and standard deviation of the natural logarithm of the `premium` coefficients. The mean of the coefficients is $e^{\mu+\sigma^2/2}$, and their standard deviation is $\sqrt{e^{2\mu+\sigma^2}(e^{\sigma^2}-1)}$. We can calculate point estimates and standard errors by using `nlcom`. To account for the reversed sign of `premium`, we multiply the mean by -1.

```
. local m _b[insurance:negpremium]
. local s _b[/Lognormal:sd(negpremium)]
. nlcom mean: -exp(`m'+`s'^2/2)
       mean: -exp(_b[insurance:negpremium]+_b[/Lognormal:sd(negpremium)]^2/2)
```

choice	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
mean	-3.048449	.4138273	-7.37	0.000	-3.859536	-2.237362

```
. nlcom sd: sqrt(exp(2*`m'+`s'^2)*(exp(`s'^2)-1)), nopval
         sd: sqrt(exp(2*_b[insurance:negpremium]+_b[/Lognormal:sd(negpremium)]
> ^2)*(exp(_b[/Lognormal:sd(negpremium)]^2)-1))
```

choice	Coefficient	Std. err.	[95% conf.	interval]
sd	.978981	.5431803	-.0856328	2.043595

We estimate a mean of -3.05, which in this case is almost the same result as in example 2, where we assumed a normal distribution for the `premium` coefficients.

◁

▷ Example 4: Random intercepts

At first glance, the random coefficient model may seem limiting. What if we believe there is correlation of the errors of the utilities across alternatives but we are unsure which of the alternative-specific variables to model with random coefficients and correlated distributions.

The constants in the model are alternative specific, and we can turn them into random intercepts. To model random intercepts, we simply add dummies for the alternatives to `random()` using factor-variable notation with the alternatives variable `insurance`; see **[U] 11.4.3 Factor variables**. We specify `correlated` as a suboption to `random()` to estimate both standard deviations and correlations. The option `noconstant` is needed because otherwise we would have collinearity between the random intercepts means and automatically created constant terms.

```
. cmmixlogit choice deductible premium, casevars(income)
> random(i.insurance, correlated) noconstant intpoints(1000)
```

(*output omitted*)

```
Mixed logit choice model                    Number of obs      =      1,250
Case ID variable: id                        Number of cases    =        250

Alternatives variable: insurance            Alts per case: min =          5
                                                           avg =        5.0
                                                           max =          5

Integration sequence:      Hammersley
Integration points:              1000       Wald chi2(10)      =      42.24
Log simulated-likelihood = -287.28864       Prob > chi2        =     0.0000
```

choice	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
insurance						
deductible	-1.27205	.4228305	-3.01	0.003	-2.100782	-.4433171
premium	-3.385171	.5534678	-6.12	0.000	-4.469948	-2.300394
insurance						
HCorp	.7807315	1.264681	0.62	0.537	-1.697997	3.25946
SickInc	.9118253	1.367183	0.67	0.505	-1.767805	3.591455
MGroup	-.9128333	1.584654	-0.58	0.565	-4.018698	2.193032
MoonHealth	-1.210673	1.724469	-0.70	0.483	-4.590571	2.169224
/Normal						
sd(2.insur~e)	1.732887	.914604			.615904	4.875596
corr(2.ins~e, 3.insurance)	.8538065	.2610271	3.27	0.001	-.5495268	.9963905
corr(2.ins~e, 4.insurance)	.72224	.408129	1.77	0.077	-.6409974	.9886836
corr(2.ins~e, 5.insurance)	.8850927	.2495122	3.55	0.000	-.6956714	.9986672
sd(3.insur~e)	2.082722	1.116996			.7279793	5.958593
corr(3.ins~e, 4.insurance)	.2685434	.5308222	0.51	0.613	-.6889524	.8846027
corr(3.ins~e, 5.insurance)	.5291393	.4334075	1.22	0.222	-.5305008	.9434711
sd(4.insur~e)	2.680528	1.237726			1.084378	6.626133
corr(4.ins~e, 5.insurance)	.9534027	.1765808	5.40	0.000	-.9590828	.9999762
sd(5.insur~e)	2.368159	1.293019			.8121816	6.905076
Health						
income	.2345847	.2143513	1.09	0.274	-.1855362	.6547055
HCorp	(base alternative)					
SickInc						
income	-.3319407	.1783297	-1.86	0.063	-.6814604	.0175791
MGroup						
income	-.3150546	.2292982	-1.37	0.169	-.7644708	.1343617
MoonHealth						
income	-.461337	.2296455	-2.01	0.045	-.911434	-.01124

```
LR test vs. fixed parameters: chi2(10) =       7.29       Prob > chi2 = 0.6981
Note: LR test is conservative and provided only for reference.
```

Note that we also used intpoints(1000) with cmmixlogit. When we used the default value for intpoints(), the model would not converge. Because we specified correlated and there are 5 alternatives, we are estimating 4 standard deviations (the number of alternatives minus one because the base level is not estimated) and 6 correlations, for a total of 10 variance parameters. The default intpoints() was not large enough for this number of variance parameters with these data. See the discussion in *Setting the number of integration points* in [CM] **Intro 5**. This introduction also has another example of a random intercepts model.

◁

Stored results

cmmixlogit stores the following in e():

Scalars

e(N)	number of observations
e(N_case)	number of cases
e(N_ic)	N for Bayesian information criterion (BIC)
e(N_clust)	number of clusters
e(k)	number of parameters
e(k_alt)	number of alternatives
e(k_casevars)	number of case-specific variables
e(k_eq)	number of equations in e(b)
e(k_eq_model)	number of equations in overall model test
e(df_m)	model degrees of freedom
e(df_c)	degrees of freedom, comparison test
e(ll)	log simulated-likelihood
e(ll_c)	log likelihood, comparison model
e(const)	constant indicator
e(intpoints)	number of raw integration points
e(lsequence)	length of each integration sequence
e(intburn)	starting sequence index
e(chi2)	χ^2
e(chi2_c)	χ^2, comparison test
e(p)	p-value for model test
e(p_c)	p-value for comparison test
e(alt_min)	minimum number of alternatives
e(alt_avg)	average number of alternatives
e(alt_max)	maximum number of alternatives
e(rank)	rank of e(V)
e(ic)	number of iterations
e(rc)	return code
e(converged)	1 if converged, 0 otherwise

Macros

e(cmd)	cmmixlogit
e(cmdline)	command as typed
e(depvar)	name of dependent variable
e(caseid)	name of case ID variable
e(altvar)	name of alternatives variable
e(alteqs)	alternative equation names
e(alt#)	alternative labels
e(base)	base alternative
e(corrmetric)	correlation metric for correlated random coefficients
e(wtype)	weight type
e(wexp)	weight expression
e(marktype)	casewise or altwise, type of markout
e(key_N_ic)	cases, key for N for Bayesian information criterion (BIC)

e(title)	title in estimation output
e(clustvar)	name of cluster variable
e(chi2type)	type of χ^2
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(ml_method)	type of ml method
e(intmethod)	technique used to generate sequences
e(sequence)	type of sequences
e(mc_rngstate)	random-number state used
e(user)	name of likelihood-evaluator program
e(technique)	maximization technique
e(properties)	b V
e(predict)	program used to implement predict
e(marginsok)	predictions allowed by margins
e(marginsnotok)	predictions disallowed by margins
e(asbalanced)	factor variables fvset as asbalanced
e(asobserved)	factor variables fvset as asobserved

Matrices

e(b)	coefficient vector
e(altvals)	alternative values
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(V)	variance–covariance matrix of the estimators
e(V_modelbased)	model-based variance

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

cmmixlogit estimates the parameters of the mixed logit model by MSL. The probability that case i chooses alternative a, conditional on the random parameter $\boldsymbol{\beta}_i$, is

$$P_{ia}(\boldsymbol{\beta}) = \frac{e^{\mathbf{x}_{ia}\boldsymbol{\beta}_i + \mathbf{w}_{ia}\boldsymbol{\alpha} + \mathbf{z}_i\boldsymbol{\delta}_a}}{\sum_{a=1}^{A} e^{\mathbf{x}_{ia}\boldsymbol{\beta}_i + \mathbf{w}_{ia}\boldsymbol{\alpha} + \mathbf{z}_i\boldsymbol{\delta}_a}}$$

We get the unconditional choice probability, P_{ia}, by integrating over the mixing distribution $f(\boldsymbol{\beta})$:

$$P_{ia} = \int P_{ia}(\boldsymbol{\beta}) f(\boldsymbol{\beta}) d\boldsymbol{\beta} \tag{2}$$

This integral of dimension d, where d equals the number of random parameters, is approximated by simulation because it has no closed-form solution. The simulated likelihood for the ith case is

$$L_i = \sum_{a=1}^{A} d_{ia} \hat{P}_{ia}$$

where d_{ia} is an indicator that takes on the value 1 for the chosen alternative and 0 otherwise. The overall log simulated-likelihood is then $\sum_{i=1}^{N} \ln L_i$. $\hat{P}_{ia}$ are the simulated probabilities,

$$\hat{P}_{ia} = \frac{1}{M} \sum_{m=1}^{M} P_{ia}(\boldsymbol{\beta}^m) \tag{3}$$

where $\boldsymbol{\beta}^m$ are the random parameters drawn from $f(\boldsymbol{\beta})$ and M is the number of random draws. Equation (3) is the computation used to approximate the probabilities in (2).

Computation of $\hat{P}_{ia}$ is based on integration sequences where each point of the sequence is a draw from density $f(\boldsymbol{\beta})$. The underlying uniform sequences are either pseudo–random draws from the uniform density or deterministic sequences such as a Halton sequence. Using deterministic sequences leads to better coverage of the probability space and lower variance of the simulator and thus to having a smaller approximation error than pseudo–random sequences, given the same number of draws. `cmmixlogit` supports pseudo–random, Halton, and Hammersley sequences; see Drukker and Gates (2006) for details.

Using a higher M in (3) will produce a better approximation to the probabilities in (2), at the cost of increased computation time. M is a function of the number of raw integration points q, which may be specified using the `intpoints()` option. In the default method, $M = q$ is the number of draws used in the approximation in (3). In addition to the default method, `cmmixlogit` supports methods in which the draws are symmetric around a midpoint, known as unidimensional and multidimensional antithetic draws. These antithetic methods produce a better approximation to the probabilities in (2), at the cost of additional computation time; see Train (2009, sec. 9.3.1). For unidimensional antithetics, $M = 2q$ draws are used. For multidimensional antithetics on a problem with d random coefficients, $M = 2^d q$ draws are used.

Random coefficients with mean μ and scale parameter σ are simulated as follows,

$$\begin{aligned}
\boldsymbol{\beta}^m_{\text{normal}} &= \mu + \sigma\eta_i & \eta_i &\sim N(0,1) \\
\boldsymbol{\beta}^m_{\text{lognormal}} &= \exp(\mu + \sigma\eta_i) & \eta_i &\sim N(0,1) \\
\boldsymbol{\beta}^m_{\text{truncated_normal}} &= \mu + \sigma\eta_i & \eta_i &\sim \text{TN}(0,1,-1.96,1.96) \\
\boldsymbol{\beta}^m_{\text{uniform}} &= \mu + \sigma\eta_i & \eta_i &\sim U(-1,1) \\
\boldsymbol{\beta}^m_{\text{triangular}} &= \mu + \sigma\eta_i & \eta_i &\sim \triangle(-1,1)
\end{aligned}$$

where $N(\mu,\sigma^2)$ is the normal distribution, $\text{TN}(\mu,\sigma^2,a,b)$ is the truncated normal distribution with lower truncation point a and upper truncation point b, $U(a,b)$ is uniform over $[\,a,b\,]$, and $\triangle(a,b)$ is the triangular distribution over $[\,a,b\,]$.

Correlated random parameters drawn from the multivariate normal distribution are generated as $\boldsymbol{\beta}^m_{\text{MVN}} = \mathbf{M} + \mathbf{L}\eta_i$, where $\mathbf{M}$ is a vector of means, $\eta_i \sim N(\mathbf{0},\mathbf{I})$, and $\mathbf{L}$ is the Cholesky factor such that the variance–covariance matrix $\mathbf{V} = \mathbf{L}\mathbf{L}'$.

This command supports the clustered version of the Huber/White/sandwich estimator of the variance using `vce(robust)` and `vce(cluster` *clustvar*`)`. See [P] **_robust**, particularly *Maximum likelihood estimators* and *Methods and formulas*. Specifying `vce(robust)` is equivalent to specifying `vce(cluster` *caseid*`)`, where *caseid* is the variable that identifies the cases.

References

Ben-Akiva, M., D. Bolduc, and J. Walker. 2001. Specification, identification, and estimation of the logit kernel (or continuous mixed logit) model. Manuscript, University of California, Berkeley, https://eml.berkeley.edu/reprints/misc/multinomial2.pdf.

Ben-Akiva, M., D. L. McFadden, M. Abe, U. Böckenholt, D. Bolduc, D. Gopinath, T. Morikawa, V. Ramaswamy, V. Rao, D. Revelt, and D. Steinberg. 1997. Modeling methods for discrete choice analysis. *Marketing Letters* 8: 273–286. https://doi.org/10.1023/A:1007956429024.

Brownstone, D., and K. E. Train. 1998. Forecasting new product penetration with flexible substitution patterns. *Journal of Econometrics* 89: 109–129. https://doi.org/10.1016/S0304-4076(98)00057-8.

Cameron, A. C., and P. K. Trivedi. 2005. *Microeconometrics: Methods and Applications*. New York: Cambridge University Press.

Drukker, D. M., and R. Gates. 2006. Generating Halton sequences using Mata. *Stata Journal* 6: 214–228.

Gutierrez, R. G., S. L. Carter, and D. M. Drukker. 2001. sg160: On boundary-value likelihood-ratio tests. *Stata Technical Bulletin* 60: 15–18. Reprinted in *Stata Technical Bulletin Reprints*, vol. 10, pp. 269–273. College Station, TX: Stata Press.

Hole, A. R. 2007. Fitting mixed logit models by using maximum simulated likelihood. *Stata Journal* 7: 388–401.

McFadden, D. L., and K. E. Train. 2000. Mixed MNL models for discrete response. *Journal of Applied Econometrics* 15: 447–470. https://doi.org/10.1002/1099-1255(200009/10)15:5<447::AID-JAE570>3.0.CO;2-1.

Train, K. E. 2009. *Discrete Choice Methods with Simulation*. 2nd ed. New York: Cambridge University Press. https://doi.org/10.1017/CBO9780511805271.

Zhu, Z., A. A. Gutiérrez-Vargas, and M. Vandebroek. 2024. Fitting mixed random regret minimization models using maximum simulated likelihood. *Stata Journal* 24: 250–272.

Also see

[CM] **cmmixlogit postestimation** — Postestimation tools for cmmixlogit

[CM] **cmclogit** — Conditional logit (McFadden's) choice model

[CM] **cmmprobit** — Multinomial probit choice model

[CM] **cmset** — Declare data to be choice model data

[CM] **cmxtmixlogit** — Panel-data mixed logit choice model

[CM] **margins** — Adjusted predictions, predictive margins, and marginal effects

[CM] **nlogit** — Nested logit regression

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

[U] 20 Estimation and postestimation commands