melogit — Multilevel mixed-effects logistic regression

Description

`melogit` fits mixed-effects models for binary and binomial responses. The conditional distribution of the response given the random effects is assumed to be Bernoulli, with success probability determined by the logistic cumulative distribution function.

Quick start

Without weights

Two-level logistic regression of `y` on `x` with random intercepts by `lev2`

```
melogit y x || lev2:
```

Mixed-effects model adding random coefficients for `x`

```
melogit y x || lev2: x
```

Same as above, but allow the random effects to be correlated

```
melogit y x || lev2: x, covariance(unstructured)
```

Three-level random-intercept model of `y` on `x` with `lev2` nested within `lev3`

```
melogit y x || lev3: || lev2:
```

Crossed-effects model of `y` on `x` with two-way crossed random effects by factors `a` and `b`

```
melogit y x || _all:R.a || b:
```

With weights

Two-level logistic regression of `y` on `x` with random intercepts by `lev2` and observation-level frequency weights `wvar1`

```
melogit y x [fweight=wvar1] || lev2:
```

Two-level random-intercept model from a two-stage sampling design with PSUs identified by `psu` using PSU-level and observation-level sampling weights `wvar2` and `wvar1`, respectively

```
melogit y x [pweight=wvar1] || psu:, pweight(wvar2)
```

Add secondary sampling stage with units identified by `ssu` having weights `wvar2` and PSU-level weights `wvar3` for a three-level random-intercept model

```
melogit y x [pw=wvar1] || psu:, pw(wvar3) || ssu:, pw(wvar2)
```

Same as above, but `svyset` data first

```
svyset psu, weight(wvar3) || ssu, weight(wvar2) || _n, weight(wvar1)
svy: melogit y x || psu: || ssu:
```

Menu

Statistics > Multilevel mixed-effects models > Logistic regression

Syntax

```
melogit depvar fe_equation [ || re_equation ] [ || re_equation ... ] [ , options ]
```

where the syntax of *fe_equation* is

```
[ indepvars ] [ if ] [ in ] [ weight ] [ , fe_options ]
```

and the syntax of *re_equation* is one of the following:

for random coefficients and intercepts

```
levelvar : [ varlist ] [ , re_options ]
```

for random effects among the values of a factor variable in a crossed-effects model

```
levelvar : R.varname
```

levelvar is a variable identifying the group structure for the random effects at that level or is `_all` representing one group comprising all observations.

fe_options	Description
Model	
`noconstant`	suppress constant term from the fixed-effects equation
`offset(`*varname*`)`	include *varname* in model with coefficient constrained to 1
`asis`	retain perfect predictor variables

re_options	Description
Model	
`covariance(`*vartype*`)`	variance–covariance structure of the random effects
`noconstant`	suppress constant term from the random-effects equation
`fweight(`*varname*`)`	frequency weights at higher levels
`iweight(`*varname*`)`	importance weights at higher levels
`pweight(`*varname*`)`	sampling weights at higher levels

options	Description
Model	
binomial(*varname* | #)	set binomial trials if data are in binomial form
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)
or	report fixed-effects coefficients as odds ratios
nocnsreport	do not display constraints
notable	suppress coefficient table
noheader	suppress output header
nogroup	suppress table summarizing groups
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(*intmethod*)	integration method
intpoints(#)	set the number of integration (quadrature) points for all levels; default is intpoints(7)
Maximization	
maximize_options	control the maximization process; seldom used
startvalues(*svmethod*)	method for obtaining starting values
startgrid[(*gridspec*)]	perform a grid search to improve starting values
noestimate	do not fit the model; show starting values instead
dnumerical	use numerical derivative techniques
collinear	keep collinear variables
coeflegend	display legend instead of statistics

vartype	Description
independent	one unique variance parameter per random effect and all covariances 0; the default unless the R. notation is used
exchangeable	equal variances for random effects and one common pairwise covariance
identity	equal variances for random effects and all covariances 0; the default if the R. notation is used
unstructured	all variances and covariances to be distinctly estimated
fixed(*matname*)	user-selected variances and covariances constrained to specified values; the remaining variances and covariances unrestricted
pattern(*matname*)	user-selected variances and covariances constrained to be equal; the remaining variances and covariances unrestricted

intmethod	Description
mvaghermite	mean–variance adaptive Gauss–Hermite quadrature; the default unless a crossed random-effects model is fit
mcaghermite	mode-curvature adaptive Gauss–Hermite quadrature
pcaghermite	Pinheiro–Chao mode-curvature adaptive Gauss–Hermite quadrature
ghermite	nonadaptive Gauss–Hermite quadrature
laplace	Laplacian approximation; the default for crossed random-effects models
pclaplace	Pinheiro–Chao Laplacian approximation

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

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

`bayes`, `by`, `collect`, and `svy` are allowed; see **[U] 11.1.10 Prefix commands**. For more details, see [BAYES] **bayes: melogit**.

`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**. Only one type of weight may be specified. Weights are not supported under the Laplacian approximation or for crossed models.

`startvalues()`, `startgrid`, `noestimate`, `dnumerical`, `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

`noconstant` suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.

`offset(`*varname*`)` specifies that *varname* be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

`asis` forces retention of perfect predictor variables and their associated, perfectly predicted observations and may produce instabilities in maximization; see [R] **probit**.

`covariance(`*vartype*`)` specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. *vartype* is one of the following: `independent`, `exchangeable`, `identity`, `unstructured`, `fixed(`*matname*`)`, or `pattern(`*matname*`)`.

`covariance(independent)` covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is `covariance(independent)` unless a crossed random-effects model is fit, in which case the default is `covariance(identity)`.

`covariance(exchangeable)` structure specifies one common variance for all random effects and one common pairwise covariance.

`covariance(identity)` is short for "multiple of the identity"; that is, all variances are equal and all covariances are 0.

`covariance(unstructured)` allows for all variances and covariances to be distinct. If an equation consists of p random-effects terms, the unstructured covariance matrix will have $p(p+1)/2$ unique parameters.

`covariance(fixed(`*matname*`))` and `covariance(pattern(`*matname*`))` covariance structures provide a convenient way to impose constraints on variances and covariances of random effects. Each specification requires a *matname* that defines the restrictions placed on variances and covariances. Only elements in the lower triangle of *matname* are used, and row and column names of *matname* are ignored. A missing value in *matname* means that a given element is unrestricted. In a `fixed(`*matname*`)` covariance structure, (co)variance (i, j) is constrained to equal the value specified in the i, jth entry of *matname*. In a `pattern(`*matname*`)` covariance structure, (co)variances (i, j) and (k, l) are constrained to be equal if $matname[i, j] = matname[k, l]$.

`fweight(`*varname*`)` specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, `[fw=`*fwtvar1*`]`. *varname* can be any valid Stata variable name, and you can specify `fweight()` at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [fw = wt1] || school: ... , fweight(wt2) ...
```

the variable `wt1` would hold the first-level (the observation-level) frequency weights, and `wt2` would hold the second-level (the school-level) frequency weights.

`iweight(`*varname*`)` specifies importance weights at higher levels in a multilevel model, whereas importance weights at the first level (the observation level) are specified in the usual manner, for example, `[iw=`*iwtvar1*`]`. *varname* can be any valid Stata variable name, and you can specify `iweight()` at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [iw = wt1] || school: ... , iweight(wt2) ...
```

the variable `wt1` would hold the first-level (the observation-level) importance weights, and `wt2` would hold the second-level (the school-level) importance weights.

`pweight(`*varname*`)` specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, `[pw=`*pwtvar1*`]`. *varname* can be any valid Stata variable name, and you can specify `pweight()` at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [pw = wt1] || school: ... , pweight(wt2) ...
```

variable `wt1` would hold the first-level (the observation-level) sampling weights, and `wt2` would hold the second-level (the school-level) sampling weights.

`binomial(`*varname* | #`)` specifies that the data are in binomial form; that is, *depvar* records the number of successes from a series of binomial trials. This number of trials is given either as *varname*, which allows this number to vary over the observations, or as the constant #. If `binomial()` is not specified (the default), *depvar* is treated as Bernoulli, with any nonzero, nonmissing values indicating positive responses.

`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`), and that allow for intragroup correlation (`cluster` *clustvar*); see [R] ***vce_option***. If `vce(robust)` is specified, robust variances are clustered at the highest level in the multilevel model.

Reporting

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

or reports estimated fixed-effects coefficients transformed to odds ratios, that is, $\exp(\beta)$ rather than β. Standard errors and confidence intervals are similarly transformed. This option affects how results are displayed, not how they are estimated. or may be specified either at estimation or upon replay.

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

notable suppresses the estimation table, either at estimation or upon replay.

noheader suppresses the output header, either at estimation or upon replay.

nogroup suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.

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(*intmethod*) specifies the integration method to be used for the random-effects model. mvaghermite performs mean–variance adaptive Gauss–Hermite quadrature; mcaghermite and pcaghermite perform mode-curvature adaptive Gauss–Hermite quadrature; ghermite performs nonadaptive Gauss–Hermite quadrature; and laplace and pclaplace perform the Laplacian approximation, equivalent to mode-curvature adaptive Gaussian quadrature with one integration point. Techniques pcaghermite and pclaplace obtain the random-effects mode and curvature using the efficient hierarchical decomposition algorithm described in Pinheiro and Chao (2006). For hierarchical models, this algorithm takes advantage of the design structure to minimize memory use and utilizes a series of orthogonal triangulations to compute the factored random-effects Hessian indirectly, avoiding the sparse full Hessian. Techniques mcaghermite and laplace use Cholesky factorization on the full Hessian. For four- and higher-level hierarchical designs, there can be dramatic computation-time differences.

The default integration method is mvaghermite unless a crossed random-effects model is fit, in which case the default integration method is laplace. The Laplacian approximation has been known to produce biased parameter estimates; however, the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects.

For crossed random-effects models, estimation with more than one quadrature point may be prohibitively intensive even for a small number of levels. For this reason, the integration method defaults to the Laplacian approximation. You may override this behavior by specifying a different integration method.

intpoints(#) sets the number of integration points for quadrature. The default is intpoints(7), which means that seven quadrature points are used for each level of random effects. This option is not allowed with intmethod(laplace).

The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. In crossed random-effects models and in models with many levels or many random coefficients, this increase can be substantial.

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**. Those that require special mention for melogit are listed below.

from() accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.

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

startvalues(*svmethod*), startgrid[(*gridspec*)], noestimate, and dnumerical; see [ME] **meglm**.

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

Remarks and examples

For a general introduction to me commands, see [ME] **me**.

melogit is a convenience command for meglm with a logit link and a bernoulli or binomial family; see [ME] **meglm**.

Remarks are presented under the following headings:

Introduction

Mixed-effects logistic regression is logistic regression containing both fixed effects and random effects. In longitudinal data and panel data, random effects are useful for modeling intracluster correlation; that is, observations in the same cluster are correlated because they share common cluster-level random effects.

melogit allows for many levels of random effects. However, for simplicity, for now we consider the two-level model, where for a series of M independent clusters, and conditional on a set of random effects $\mathbf{u}_j$,

$$\Pr(y_{ij} = 1|\mathbf{x}_{ij}, \mathbf{u}_j) = H(\mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_{ij}\mathbf{u}_j) \tag{1}$$

for $j = 1, \ldots, M$ clusters, with cluster j consisting of $i = 1, \ldots, n_j$ observations. The responses are the binary-valued y_{ij}, and we follow the standard Stata convention of treating $y_{ij} = 1$ if $\textit{depvar}_{ij} \neq 0$ and treating $y_{ij} = 0$ otherwise. The $1 \times p$ row vector $\mathbf{x}_{ij}$ are the covariates for the fixed effects, analogous to the covariates you would find in a standard logistic regression model, with regression coefficients (fixed effects) $\boldsymbol{\beta}$. For notational convenience here and throughout this manual entry, we suppress the dependence of y_{ij} on $\mathbf{x}_{ij}$.

The $1 \times q$ vector $\mathbf{z}_{ij}$ are the covariates corresponding to the random effects and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, $\mathbf{z}_{ij}$ is simply the scalar 1. The random effects $\mathbf{u}_j$ are M realizations from a multivariate normal distribution with mean $\mathbf{0}$ and $q \times q$ variance matrix $\boldsymbol{\Sigma}$. The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of $\boldsymbol{\Sigma}$, known as variance components. One special case of (1) places $\mathbf{z}_{ij} = \mathbf{x}_{ij}$ so that all covariate effects are essentially random and distributed as multivariate normal with mean $\boldsymbol{\beta}$ and variance $\boldsymbol{\Sigma}$.

Finally, because this is logistic regression, $H(\cdot)$ is the logistic cumulative distribution function, which maps the linear predictor to the probability of a success ($y_{ij} = 1$), with $H(v) = \exp(v)/\{1 + \exp(v)\}$.

Model (1) may also be stated in terms of a latent linear response, where only $y_{ij} = I(y^*_{ij} > 0)$ is observed for the latent

$$y^*_{ij} = \mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_{ij}\mathbf{u}_j + \epsilon_{ij}$$

The errors ϵ_{ij} are distributed as logistic with mean 0 and variance $\pi^2/3$ and are independent of $\mathbf{u}_j$.

A two-level logistic model can also be fit using `xtlogit` with the `re` option; see [XT] **xtlogit**. In the absence of random effects, mixed-effects logistic regression reduces to standard logistic regression; see [R] **logit**.

Two-level models

▷ Example 1: Two-level random-intercept model

Ng et al. (2006) analyze a subsample of data from the 1989 Bangladesh fertility survey (Huq and Cleland 1990), which polled 1,934 Bangladeshi women on their use of contraception.

```
. use https://www.stata-press.com/data/r19/bangladesh
(Bangladesh Fertility Survey, 1989)
. describe
Contains data from https://www.stata-press.com/data/r19/bangladesh.dta
 Observations:         1,934                  Bangladesh Fertility Survey,
                                                1989
    Variables:             8                  28 May 2024 20:27
                                              (_dta has notes)
------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------
district        byte    %9.0g                 District
c_use           byte    %9.0g      yesno      Use contraception
urban           byte    %9.0g      urban      Urban or rural
age             float   %6.2f                 Age, mean centered
child1          byte    %9.0g                 1 child
child2          byte    %9.0g                 2 children
child3          byte    %9.0g                 3 or more children
children        byte    %18.0g     childlbl   Number of children
------------------------------------------------------------------------------
Sorted by: district
```

The women sampled were from 60 districts, identified by the variable `district`. Each district contained either urban or rural areas (variable `urban`) or both. The variable `c_use` is the binary response, with a value of 1 indicating contraceptive use. Other covariates include mean-centered `age` and a factor variable for the number of children.

Consider a standard logistic regression model, amended to have random effects for each district. Defining $\pi_{ij} = \Pr(\texttt{c\_use}_{ij} = 1)$, we have

$$\begin{aligned}\text{logit}(\pi_{ij}) = \beta_0 + \beta_1 1.\texttt{urban}_{ij} + \beta_2 \texttt{age}_{ij} + \beta_3 1.\texttt{children}_{ij} + \beta_4 2.\texttt{children}_{ij} + \\ \beta_5 3.\texttt{children}_{ij} + u_j\end{aligned} \tag{2}$$

for $j = 1, \ldots, 60$ districts, with $i = 1, \ldots, n_j$ women in district j.

```
. melogit c_use i.urban age i.children, nofvlabel|| district:
Fitting fixed-effects model:
Iteration 0:  Log likelihood = -1229.5485
Iteration 1:  Log likelihood = -1228.5268
Iteration 2:  Log likelihood = -1228.5263
Iteration 3:  Log likelihood = -1228.5263
Refining starting values:
Grid node 0:  Log likelihood = -1219.2681
Fitting full model:
Iteration 0:  Log likelihood = -1219.2681  (not concave)
Iteration 1:  Log likelihood = -1207.5978
Iteration 2:  Log likelihood = -1206.8428
Iteration 3:  Log likelihood = -1206.8322
Iteration 4:  Log likelihood = -1206.8322
Mixed-effects logistic regression               Number of obs     =      1,934
Group variable: district                        Number of groups  =         60
                                                Obs per group:
                                                              min =          2
                                                              avg =       32.2
                                                              max =        118
Integration method: mvaghermite                 Integration pts.  =          7
                                                Wald chi2(5)      =     109.60
Log likelihood = -1206.8322                     Prob > chi2       =     0.0000
```

c_use	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
1.urban	.7322765	.1194857	6.13	0.000	.4980888	.9664641
age	-.0264981	.0078916	-3.36	0.001	-.0419654	-.0110309
children						
1	1.116001	.1580921	7.06	0.000	.8061465	1.425856
2	1.365895	.1746691	7.82	0.000	1.02355	1.70824
3	1.344031	.1796549	7.48	0.000	.9919139	1.696148
_cons	-1.68929	.1477591	-11.43	0.000	-1.978892	-1.399687
district						
var(_cons)	.215618	.0733222			.1107208	.4198954

```
LR test vs. logistic model: chibar2(01) = 43.39       Prob >= chibar2 = 0.0000
```

The estimation table reports the fixed effects and the estimated variance components. The fixed effects can be interpreted just as you would the output from `logit`. You can also specify the `or` option at estimation or on replay to display the fixed effects as odds ratios instead. If you did display results as odds ratios, you would find urban women to have roughly double the odds of using contraception as that of their rural counterparts. Having any number of children will increase the odds from three- to fourfold when compared with the base category of no children. Contraceptive use also decreases with age. The `nofvlabel` option requested the values of factor variables `urban` and `children` be displayed instead of the value labels.

Underneath the fixed effect, the table shows the estimated variance components. The random-effects equation is labeled `district`, meaning that these are random effects at the `district` level. Because we have only one random effect at this level, the table shows only one variance component. The estimate of σ_u^2 is 0.22 with standard error 0.07.

A likelihood-ratio test comparing the model with ordinary logistic regression is provided and is highly significant for these data.

We now store our estimates for later use.

```
. estimates store r_int
```

◁

In what follows, we will be extending (2), focusing on the variable `urban`. Before we begin, to keep things short we restate (2) as

$$\text{logit}(\pi_{ij}) = \beta_0 + \beta_1 \texttt{1.urban}_{ij} + \mathcal{F}_{ij} + u_j$$

where $\mathcal{F}_{ij}$ is merely shorthand for the portion of the fixed-effects specification having to do with age and children.

▷ Example 2: Two-level random-slope model

Extending (2) to allow for a random slope on the indicator variable `1.urban` yields the model

$$\text{logit}(\pi_{ij}) = \beta_0 + \beta_1 \texttt{1.urban}_{ij} + \mathcal{F}_{ij} + u_j + v_j \texttt{1.urban}_{ij} \tag{3}$$

which we can fit by typing

```
. melogit c_use i.urban age i.children, nofvlabel || district: i.urban
  (output omitted)
. estimates store r_urban
```

Extending the model was as simple as adding `i.urban` to the random-effects specification so that the model now includes a random intercept and a random coefficient on `1.urban`. We dispense with the output because, although this is an improvement over the random-intercept model (2),

```
. lrtest r_int r_urban
Likelihood-ratio test
Assumption: r_int nested within r_urban
 LR chi2(1) =   3.66
Prob > chi2 = 0.0558
Note: The reported degrees of freedom assumes the null hypothesis is not on
      the boundary of the parameter space. If this is not true, then the
      reported test is conservative.
```

we find the default covariance structure for (u_j, v_j), `covariance(independent)`,

$$\boldsymbol{\Sigma} = \text{Var}\begin{bmatrix} u_j \\ v_j \end{bmatrix} = \begin{bmatrix} \sigma_u^2 & 0 \\ 0 & \sigma_v^2 \end{bmatrix}$$

to be inadequate. We state that the random-coefficient model is an "improvement" over the random-intercept model because the null hypothesis of the likelihood-ratio comparison test ($H_0 : \sigma_v^2 = 0$) is on the boundary of the parameter test. This makes the reported p-value, 5.6%, an upper bound on the actual p-value, which is actually half of that; see *Distribution theory for likelihood-ratio test* in [ME] **me**.

We see below that we can reject this model in favor of one that allows correlation between u_j and v_j.

```
. melogit c_use i.urban age i.children, nofvlabel
> || district: i.urban, covariance(unstructured)

Fitting fixed-effects model:

Iteration 0:  Log likelihood = -1229.5485
Iteration 1:  Log likelihood = -1228.5268
Iteration 2:  Log likelihood = -1228.5263
Iteration 3:  Log likelihood = -1228.5263

Refining starting values:

Grid node 0:  Log likelihood = -1215.8592

Fitting full model:

Iteration 0:  Log likelihood = -1215.8592  (not concave)
Iteration 1:  Log likelihood = -1201.0652
Iteration 2:  Log likelihood = -1199.6394
Iteration 3:  Log likelihood = -1199.3157
Iteration 4:  Log likelihood =  -1199.315
Iteration 5:  Log likelihood =  -1199.315

Mixed-effects logistic regression               Number of obs     =      1,934
Group variable: district                        Number of groups  =         60

                                                Obs per group:
                                                              min =          2
                                                              avg =       32.2
                                                              max =        118

Integration method: mvaghermite                 Integration pts.  =          7

                                                Wald chi2(5)      =      97.50
Log likelihood = -1199.315                      Prob > chi2       =     0.0000
```

c_use	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
1.urban	.8157875	.1715519	4.76	0.000	.4795519	1.152023
age	-.026415	.008023	-3.29	0.001	-.0421398	-.0106902
children						
1	1.13252	.1603285	7.06	0.000	.818282	1.446758
2	1.357739	.1770522	7.67	0.000	1.010723	1.704755
3	1.353827	.1828801	7.40	0.000	.9953882	1.712265
_cons	-1.71165	.1605618	-10.66	0.000	-2.026345	-1.396954
district						
var(1.urban)	.6663237	.3224689			.258074	1.720387
var(_cons)	.3897448	.1292463			.203473	.7465413
district						
cov(1.urban, _cons)	-.4058861	.1755414	-2.31	0.021	-.7499408	-.0618313

```
LR test vs. logistic model: chi2(3) = 58.42               Prob > chi2 = 0.0000

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

. estimates store r_urban_corr

. lrtest r_urban r_urban_corr

Likelihood-ratio test
Assumption: r_urban nested within r_urban_corr

 LR chi2(1) =  11.38
Prob > chi2 = 0.0007
```

By specifying `covariance(unstructured)` above, we told `melogit` to allow correlation between random effects at the `district` level; that is,

$$\boldsymbol{\Sigma} = \text{Var}\begin{bmatrix} u_j \\ v_j \end{bmatrix} = \begin{bmatrix} \sigma_u^2 & \sigma_{uv} \\ \sigma_{uv} & \sigma_v^2 \end{bmatrix}$$

◁

▷ Example 3: Alternative parameterization of random slopes

The purpose of introducing a random coefficient on the binary variable `urban` in (3) was to allow for separate random effects, within each district, for the urban and rural areas of that district. Hence, if we turn off base levels for factor variable `i.urban` via `ibn.urban`, then we can reformulate (3) as

$$\text{logit}(\pi_{ij}) = \beta_0 \texttt{0.urban}_{ij} + (\beta_0 + \beta_1)\texttt{1.urban}_{ij} + \mathcal{F}_{ij} + u_j \texttt{0.urban}_{ij} + (u_j + v_j)\texttt{1.urban}_{ij} \quad (3a)$$

where we have translated both the fixed portion and the random portion to be in terms of `0.urban` rather than a random intercept. Translating the fixed portion is not necessary to make the point we make below, but we do so anyway for uniformity.

Translating the estimated random-effects parameters from the previous output to ones appropriate for (3a), we get $\text{Var}(u_j) = \hat{\sigma}_u^2 = 0.39$,

$$\begin{aligned} \text{Var}(u_j + v_j) &= \hat{\sigma}_u^2 + \hat{\sigma}_v^2 + 2\hat{\sigma}_{uv} \\ &= 0.39 + 0.67 - 2(0.41) = 0.24 \end{aligned}$$

and $\text{Cov}(u_j, u_j + v_j) = \hat{\sigma}_u^2 + \hat{\sigma}_{uv} = 0.39 - 0.41 = -0.02$.

An alternative that does not require remembering how to calculate variances and covariances involving sums—and one that also gives you standard errors—is to let Stata do the work for you:

```
. melogit c_use ibn.urban age i.children, noconstant nofvlabel
> || district: ibn.urban, noconstant cov(unstructured)

Fitting fixed-effects model:

Iteration 0:  Log likelihood = -1229.5485
Iteration 1:  Log likelihood = -1228.5268
Iteration 2:  Log likelihood = -1228.5263
Iteration 3:  Log likelihood = -1228.5263

Refining starting values:

Grid node 0:  Log likelihood = -1208.3922

Fitting full model:

Iteration 0:  Log likelihood = -1208.3922  (not concave)
Iteration 1:  Log likelihood =  -1203.556  (not concave)
Iteration 2:  Log likelihood = -1200.5896
Iteration 3:  Log likelihood = -1199.7288
Iteration 4:  Log likelihood = -1199.3373
Iteration 5:  Log likelihood = -1199.3151
Iteration 6:  Log likelihood =  -1199.315

Mixed-effects logistic regression               Number of obs     =      1,934
Group variable: district                        Number of groups  =         60
                                                Obs per group:
                                                              min =          2
                                                              avg =       32.2
                                                              max =        118
Integration method: mvaghermite                 Integration pts.  =          7
                                                Wald chi2(6)      =     120.24
Log likelihood = -1199.315                      Prob > chi2       =     0.0000
 ( 1)  [c_use]_cons = 0
```

c_use	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
urban						
0	-1.711652	.1605617	-10.66	0.000	-2.026347	-1.396956
1	-.8958623	.1704954	-5.25	0.000	-1.230027	-.5616974
age	-.026415	.008023	-3.29	0.001	-.0421398	-.0106903
children						
1	1.13252	.1603285	7.06	0.000	.8182819	1.446758
2	1.357739	.1770522	7.67	0.000	1.010724	1.704755
3	1.353827	.18288	7.40	0.000	.9953883	1.712265
_cons	0	(omitted)				
district						
var(0.urban)	.3897485	.1292403			.2034823	.7465212
var(1.urban)	.2442899	.1450625			.0762871	.7822759
district						
cov(0.urban, 1.urban)	-.0161411	.1057462	-0.15	0.879	-.2233999	.1911177

```
LR test vs. logistic model: chi2(3) = 58.42                 Prob > chi2 = 0.0000

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

The above output demonstrates an equivalent fit to that we displayed for model (3), with the added benefit of a more direct comparison of the parameters for rural and urban areas.

◁

❑ Technical note

Our model fits for (3) and (3a) are equivalent only because we allowed for correlation in the random effects for both. Had we used the default independent covariance structure, we would be fitting different models; in (3) we would be making the restriction that $\text{Cov}(u_j, v_j) = 0$, whereas in (3a) we would be assuming that $\text{Cov}(u_j, u_j + v_j) = 0$.

The moral here is that although `melogit` will do this by default, one should be cautious when imposing an independent covariance structure, because the correlation between random effects is not invariant to model translations that would otherwise yield equivalent results in standard regression models. In our example, we remapped an intercept and binary coefficient to two complementary binary coefficients, something we could do in standard logistic regression without consequence but that here required more consideration.

Rabe-Hesketh and Skrondal (2022, sec. 11.4) provide a nice discussion of this phenomenon in the related case of recentering a continuous covariate.

❑

Other covariance structures

In the above examples, we demonstrated the `independent` and `unstructured` covariance structures. Also available are `identity` (seen previously in output but not directly specified), which restricts random effects to be uncorrelated and share a common variance, and `exchangeable`, which assumes a common variance and a common pairwise covariance.

You can also specify multiple random-effects equations at the same level, in which case the above four covariance types can be combined to form more complex blocked-diagonal covariance structures. This could be used, for example, to impose an equality constraint on a subset of variance components or to otherwise group together a set of related random effects.

Continuing the previous example, typing

```
. melogit c_use i.urban age i.children,
>         || district: i.children, cov(exchangeable)
>         || district:
```

would fit a model with the same fixed effects as (3) but with random-effects structure

$$\text{logit}(\pi_{ij}) = \beta_0 + \cdots + u_{1j}\texttt{1.children}_{ij} + u_{2j}\texttt{2.children}_{ij} + u_{3j}\texttt{3.children}_{ij} + v_j$$

That is, we have random coefficients on the children factor levels (the first `district:` specification) and an overall district random intercept (the second `district:` specification). The above syntax fits a model with overall covariance structure

$$\boldsymbol{\Sigma} = \text{Var}\begin{bmatrix} u_{1j} \\ u_{2j} \\ u_{3j} \\ v_j \end{bmatrix} = \begin{bmatrix} \sigma_u^2 & \sigma_c & \sigma_c & 0 \\ \sigma_c & \sigma_u^2 & \sigma_c & 0 \\ \sigma_c & \sigma_c & \sigma_u^2 & 0 \\ 0 & 0 & 0 & \sigma_v^2 \end{bmatrix}$$

reflecting the relationship among the random coefficients for children. We did not have to specify `noconstant` on the first `district:` specification. `melogit` automatically avoids collinearity by including an intercept on only the final specification among repeated-level equations.

Of course, if we fit the above model, we would heed our own advice from the previous technical note and make sure that not only our data but also our specification characterization of the random effects permitted the above structure. That is, we would check the above against a model that had an `unstructured` covariance for all four random effects and then perhaps against a model that assumed an `unstructured` covariance among the three random coefficients on children, coupled with independence with the random intercept. All comparisons can be made by storing estimates (command `estimates store`) and then using `lrtest`, as demonstrated previously.

Three-level models

▷ Example 4: Three-level random-intercept model

Rabe-Hesketh, Toulopoulou, and Murray (2001) analyzed data from a study measuring the cognitive ability of patients with schizophrenia compared with their relatives and control subjects. Cognitive ability was measured as the successful completion of the "Tower of London", a computerized task, measured at three levels of difficulty. For all but one of the 226 subjects, there were three measurements (one for each difficulty level). Because patients' relatives were also tested, a family identifier, `family`, was also recorded.

```
. use https://www.stata-press.com/data/r19/towerlondon, clear
(Tower of London data)
. describe
Contains data from https://www.stata-press.com/data/r19/towerlondon.dta
 Observations:           677                  Tower of London data
    Variables:             5                  31 May 2024 10:41
                                              (_dta has notes)
------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------
family          int     %8.0g                 Family ID
subject         int     %9.0g                 Subject ID
dtlm            byte    %9.0g                 1 = task completed
difficulty      byte    %9.0g                 Level of difficulty: -1, 0, or 1
group           byte    %8.0g                 1: controls; 2: relatives; 3:
                                                schizophrenics
------------------------------------------------------------------------------
Sorted by: family  subject
```

We fit a logistic model with response `dtlm`, the indicator of cognitive function, and with covariates `difficulty` and a set of indicator variables for `group`, with the controls (`group==1`) being the base category. We allow for random effects due to families and due to subjects within families, and we request to see odds ratios.

```
. melogit dtlm difficulty i.group || family: || subject: , or
Fitting fixed-effects model:

Iteration 0:  Log likelihood = -317.35042
Iteration 1:  Log likelihood = -313.90007
Iteration 2:  Log likelihood = -313.89079
Iteration 3:  Log likelihood = -313.89079

Refining starting values:

Grid node 0:  Log likelihood = -310.28429

Fitting full model:

Iteration 0:  Log likelihood = -310.28429
Iteration 1:  Log likelihood = -307.36653
Iteration 2:  Log likelihood = -305.19357
Iteration 3:  Log likelihood = -305.12073
Iteration 4:  Log likelihood = -305.12041
Iteration 5:  Log likelihood = -305.12041

Mixed-effects logistic regression               Number of obs     =        677
        Grouping information
```

Group variable	No. of groups	Observations per group Minimum	Average	Maximum
family	118	2	5.7	27
subject	226	2	3.0	3

```
Integration method: mvaghermite                 Integration pts.  =          7
                                                Wald chi2(3)      =      74.90
Log likelihood = -305.12041                     Prob > chi2       =     0.0000
```

dtlm	Odds ratio	Std. err.	z	P>|z|	[95% conf.	interval]
difficulty	.1923372	.037161	-8.53	0.000	.1317057	.2808806
group						
2	.7798263	.2763763	-0.70	0.483	.3893369	1.561961
3	.3491318	.13965	-2.63	0.009	.15941	.764651
_cons	.226307	.0644625	-5.22	0.000	.1294902	.3955112
family						
var(_cons)	.5692105	.5215654			.0944757	3.429459
family>subject						
var(_cons)	1.137917	.6854853			.3494165	3.705762

```
Note: Estimates are transformed only in the first equation to odds ratios.
Note: _cons estimates baseline odds (conditional on zero random effects).
LR test vs. logistic model: chi2(2) = 17.54                 Prob > chi2 = 0.0002

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

This is a three-level model with two random-effects equations, separated by `||`. The first is a random intercept (constant only) at the `family` level, and the second is a random intercept at the `subject` level. The order in which these are specified (from left to right) is significant—`melogit` assumes that `subject` is nested within `family`.

The information on groups is now displayed as a table, with one row for each upper level. Among other things, we see that we have 226 subjects from 118 families.

After adjusting for the random-effects structure, the odds of successful completion of the Tower of London decrease dramatically as the level of difficulty increases. Also, schizophrenics (group==3) tended not to perform as well as the control subjects. Of course, we would make similar conclusions from a standard logistic model fit to the same data, but the odds ratios would differ somewhat.

◁

❑ Technical note

In the previous example, the subjects are coded with unique values between 1 and 251 (with some gaps), but such coding is not necessary to produce nesting within families. Once we specified the nesting structure to melogit, all that was important was the relative coding of subject within each unique value of family. We could have coded subjects as the numbers 1, 2, 3, and so on, restarting at 1 with each new family, and melogit would have produced the same results.

Group identifiers may also be coded using string variables.

❑

The above extends to models with more than two levels of nesting by adding more random-effects equations, each separated by ||. The order of nesting goes from left to right as the groups go from biggest (highest level) to smallest (lowest level).

Crossed-effects models

▷ Example 5: Crossed-effects model

Rabe-Hesketh and Skrondal (2022, 493–512) perform an analysis on school data from Fife, Scotland. The data, originally from Paterson (1991), are from a study measuring students' attainment as an integer score from 1 to 10, based on the Scottish school exit examination taken at age 16. The study comprises 3,435 students who first attended any one of 148 primary schools and then any one of 19 secondary schools.

```
. use https://www.stata-press.com/data/r19/fifeschool
(School data from Fife, Scotland)
. describe
Contains data from https://www.stata-press.com/data/r19/fifeschool.dta
 Observations:         3,435                  School data from Fife, Scotland
    Variables:             5                  28 May 2024 10:08
                                              (_dta has notes)
------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------
pid             int     %9.0g                 Primary school ID
sid             byte    %9.0g                 Secondary school ID
attain          byte    %9.0g                 Attainment score at age 16
vrq             int     %9.0g                 Verbal-reasoning score from final
                                                year of primary school
sex             byte    %9.0g                 1: female; 0: male
------------------------------------------------------------------------------
Sorted by:
. generate byte attain_gt_6 = attain > 6
```

To make the analysis relevant to our present discussion, we focus not on the attainment score itself but instead on whether the score is greater than 6. We wish to model this indicator as a function of the fixed effect sex and of random effects due to primary and secondary schools.

For this analysis, it would make sense to assume that the random effects are not nested, but instead crossed, meaning that the effect due to primary school is the same regardless of the secondary school attended. Our model is thus

$$\text{logit}\{\Pr(\texttt{attain}_{ijk} > 6)\} = \beta_0 + \beta_1 \texttt{sex}_{ijk} + u_j + v_k \tag{4}$$

for student i, $i = 1, \ldots, n_{jk}$, who attended primary school j, $j = 1, \ldots, 148$, and then secondary school k, $k = 1, \ldots, 19$.

Because there is no evident nesting, one solution would be to consider the data as a whole and fit a two-level, one-cluster model with random-effects structure

$$\mathbf{u} = \begin{bmatrix} u_1 \\ \vdots \\ u_{148} \\ v_1 \\ \vdots \\ v_{19} \end{bmatrix} \sim N(\mathbf{0}, \boldsymbol{\Sigma}); \quad \boldsymbol{\Sigma} = \begin{bmatrix} \sigma_u^2 \mathbf{I}_{148} & \mathbf{0} \\ \mathbf{0} & \sigma_v^2 \mathbf{I}_{19} \end{bmatrix}$$

We can fit such a model by using the group designation `_all:`, which tells `melogit` to treat the whole dataset as one cluster, and the `R.`*varname* notation, which mimics the creation of indicator variables identifying schools:

```
. melogit attain_gt_6 sex || _all:R.pid || _all:R.sid, or
```

But we do not recommend fitting the model this way because of high total dimension ($148 + 19 = 167$) of the random effects. This would require working with matrices of column dimension 167, which is probably not a problem for most current hardware, but would be a problem if this number got much larger.

An equivalent way to fit (4) that has a smaller dimension is to treat the clusters identified by primary schools as nested within all the data, that is, as nested within the `_all` group.

```
. melogit attain_gt_6 sex || _all:R.sid || pid:, or
note: crossed random-effects model specified; option intmethod(laplace)
      implied.

Fitting fixed-effects model:

Iteration 0:  Log likelihood = -2320.2374
Iteration 1:  Log likelihood = -2317.9062
Iteration 2:  Log likelihood = -2317.9059
Iteration 3:  Log likelihood = -2317.9059

Refining starting values:

Grid node 0:  Log likelihood = -2234.6403

Fitting full model:

Iteration 0:  Log likelihood = -2234.6403  (not concave)
Iteration 1:  Log likelihood = -2227.9507  (not concave)
Iteration 2:  Log likelihood = -2227.9287  (not concave)
Iteration 3:  Log likelihood = -2227.9265  (not concave)
Iteration 4:  Log likelihood = -2227.9263
Iteration 5:  Log likelihood = -2221.6884  (not concave)
Iteration 6:  Log likelihood = -2221.1707  (not concave)
Iteration 7:  Log likelihood = -2221.1232
Iteration 8:  Log likelihood = -2220.1709  (not concave)
Iteration 9:  Log likelihood = -2220.1556
Iteration 10: Log likelihood = -2220.0522
Iteration 11: Log likelihood = -2220.0039
Iteration 12: Log likelihood = -2220.0035
Iteration 13: Log likelihood = -2220.0035

Mixed-effects logistic regression               Number of obs     =      3,435
        Grouping information
```

Group variable	No. of groups	Observations per group Minimum	Average	Maximum
_all	1	3,435	3,435.0	3,435
pid	148	1	23.2	72

```
Integration method: laplace

                                                Wald chi2(1)      =      14.33
Log likelihood = -2220.0035                     Prob > chi2       =     0.0002
```

attain_gt_6	Odds ratio	Std. err.	z	P>|z|	[95% conf.	interval]
sex	1.325119	.0985247	3.79	0.000	1.145425	1.533003
_cons	.5311554	.0621222	-5.41	0.000	.4223454	.6679984
_all>sid						
var(_cons)	.1239764	.0693322			.0414301	.37099
pid						
var(_cons)	.452049	.0951708			.2992129	.6829528

```
Note: Estimates are transformed only in the first equation to odds ratios.
Note: _cons estimates baseline odds (conditional on zero random effects).
LR test vs. logistic model: chi2(2) = 195.80              Prob > chi2 = 0.0000

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

Choosing the primary schools as those to nest was no accident; because there are far fewer secondary schools than primary schools, the above required only 19 random coefficients for the secondary schools and one random intercept at the primary school level, for a total dimension of 20. Our data also include a measurement of verbal reasoning, the variable `vrq`. Adding a fixed effect due to `vrq` in (4) would negate the effect due to secondary school, a fact we leave to you to verify as an exercise.

◁

See [ME] **mixed** for a similar discussion of crossed effects in the context of linear mixed models. Also see Rabe-Hesketh and Skrondal (2022) for more examples of crossed-effects models, including models with random interactions, and for more techniques on how to avoid high-dimensional estimation.

❑ Technical note

The estimation in the previous example was performed using a Laplacian approximation, even though we did not specify this. Whenever the `R.` notation is used in random-effects specifications, estimation reverts to the Laplacian method because of the high dimension induced by having the `R.` variables.

In the above example, through some creative nesting, we reduced the dimension of the random effects to 20, but this is still too large to permit estimation via adaptive Gaussian quadrature; see *Computation time and the Laplacian approximation* in [ME] **me**. Even with two quadrature points, our rough formula for computation time would contain within it a factor of $2^{20} = 1{,}048{,}576$.

The `intmethod(laplace)` option is therefore assumed when you use `R.` notation. If the number of distinct levels of your `R.` variables is small enough (say, five or fewer) to permit estimation via quadrature, you can override the imposition of `laplace` by specifying a different integration method in the `intmethod()` option.

❑

Stored results

`melogit` stores the following in `e()`:

Scalars

`e(N)`	number of observations
`e(k)`	number of parameters
`e(k_dv)`	number of dependent variables
`e(k_eq)`	number of equations in `e(b)`
`e(k_eq_model)`	number of equations in overall model test
`e(k_f)`	number of fixed-effects parameters
`e(k_r)`	number of random-effects parameters
`e(k_rs)`	number of variances
`e(k_rc)`	number of covariances
`e(df_m)`	model degrees of freedom
`e(ll)`	log likelihood
`e(N_clust)`	number of clusters
`e(chi2)`	χ^2
`e(p)`	p-value for model test
`e(ll_c)`	log likelihood, comparison model
`e(chi2_c)`	χ^2, comparison test
`e(df_c)`	degrees of freedom, comparison test
`e(p_c)`	p-value for comparison test
`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)	meglm
e(cmd2)	melogit
e(cmdline)	command as typed
e(depvar)	name of dependent variable
e(wtype)	weight type
e(wexp)	weight expression (first-level weights)
e(fweight*k*)	fweight variable for *k*th highest level, if specified
e(iweight*k*)	iweight variable for *k*th highest level, if specified
e(pweight*k*)	pweight variable for *k*th highest level, if specified
e(covariates)	list of covariates
e(ivars)	grouping variables
e(model)	logistic
e(title)	title in estimation output
e(link)	logit
e(family)	bernoulli or binomial
e(clustvar)	name of cluster variable
e(offset)	offset
e(binomial)	binomial number of trials
e(intmethod)	integration method
e(n_quad)	number of integration points
e(chi2type)	Wald; type of model 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(user)	name of likelihood-evaluator program
e(technique)	maximization technique
e(datasignature)	the checksum
e(datasignaturevars)	variables used in calculation of checksum
e(properties)	b V
e(estat_cmd)	program used to implement estat
e(predict)	program used to implement predict
e(marginsnotok)	predictions disallowed by margins
e(marginswtype)	weight type for margins
e(marginswexp)	weight expression for margins
e(asbalanced)	factor variables fvset as asbalanced
e(asobserved)	factor variables fvset as asobserved

Matrices

e(b)	coefficient vector
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(N_g)	group counts
e(g_min)	group-size minimums
e(g_avg)	group-size averages
e(g_max)	group-size maximums
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

`melogit` is a convenience command for `meglm` with a `logit` link and a `bernoulli` or `binomial` family; see [ME] **meglm**.

Model (1) assumes Bernoulli data, a special case of the binomial. Because binomial data are also supported by `melogit` (option `binomial()`), the methods presented below are in terms of the more general binomial mixed-effects model.

For a two-level binomial model, consider the response y_{ij} as the number of successes from a series of r_{ij} Bernoulli trials (replications). For cluster j, $j = 1, \ldots, M$, the conditional distribution of $\mathbf{y}_j = (y_{j1}, \ldots, y_{jn_j})'$, given a set of cluster-level random effects $\mathbf{u}_j$, is

$$\begin{aligned} f(\mathbf{y}_j|\mathbf{u}_j) &= \prod_{i=1}^{n_j} \left[\binom{r_{ij}}{y_{ij}} \left\{H(\boldsymbol{\eta}_{ij})\right\}^{y_{ij}} \left\{1 - H(\boldsymbol{\eta}_{ij})\right\}^{r_{ij}-y_{ij}} \right] \\ &= \exp\left(\sum_{i=1}^{n_j} \left[y_{ij}\boldsymbol{\eta}_{ij} - r_{ij} \log\left\{1 + \exp(\boldsymbol{\eta}_{ij})\right\} + \log\binom{r_{ij}}{y_{ij}} \right] \right) \end{aligned}$$

for $\boldsymbol{\eta}_{ij} = \mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_{ij}\mathbf{u}_j + \text{offset}_{ij}$ and $H(v) = \exp(v)/\{1 + \exp(v)\}$.

Defining $\mathbf{r}_j = (r_{j1}, \ldots, r_{jn_j})'$ and

$$c\left(\mathbf{y}_j, \mathbf{r}_j\right) = \sum_{i=1}^{n_j} \log\binom{r_{ij}}{y_{ij}}$$

where $c(\mathbf{y}_j, \mathbf{r}_j)$ does not depend on the model parameters, we can express the above compactly in matrix notation,

$$f(\mathbf{y}_j|\mathbf{u}_j) = \exp\left[\mathbf{y}_j'\boldsymbol{\eta}_j - \mathbf{r}_j' \log\left\{\mathbf{1} + \exp(\boldsymbol{\eta}_j)\right\} + c\left(\mathbf{y}_j, \mathbf{r}_j\right)\right]$$

where $\boldsymbol{\eta}_j$ is formed by stacking the row vectors $\boldsymbol{\eta}_{ij}$. We extend the definitions of the functions $\log(\cdot)$ and $\exp(\cdot)$ to be vector functions where necessary.

Because the prior distribution of $\mathbf{u}_j$ is multivariate normal with mean $\mathbf{0}$ and $q \times q$ variance matrix $\boldsymbol{\Sigma}$, the likelihood contribution for the jth cluster is obtained by integrating $\mathbf{u}_j$ out of the joint density $f(\mathbf{y}_j, \mathbf{u}_j)$,

$$\begin{aligned} \mathcal{L}_j(\boldsymbol{\beta}, \boldsymbol{\Sigma}) &= (2\pi)^{-q/2} \left|\boldsymbol{\Sigma}\right|^{-1/2} \int f(\mathbf{y}_j|\mathbf{u}_j) \exp\left(-\mathbf{u}_j'\boldsymbol{\Sigma}^{-1}\mathbf{u}_j/2\right) d\mathbf{u}_j \\ &= \exp\left\{c\left(\mathbf{y}_j, \mathbf{r}_j\right)\right\} (2\pi)^{-q/2} \left|\boldsymbol{\Sigma}\right|^{-1/2} \int \exp\left\{h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j\right)\right\} d\mathbf{u}_j \end{aligned} \tag{5}$$

where

$$h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j\right) = \mathbf{y}_j'\boldsymbol{\eta}_j - \mathbf{r}_j' \log\left\{\mathbf{1} + \exp(\boldsymbol{\eta}_j)\right\} - \mathbf{u}_j'\boldsymbol{\Sigma}^{-1}\mathbf{u}_j/2$$

and for convenience, in the arguments of $h(\cdot)$ we suppress the dependence on the observable data $(\mathbf{y}_j, \mathbf{r}_j, \mathbf{X}_j, \mathbf{Z}_j)$.

The integration in (5) has no closed form and thus must be approximated; see *Methods and formulas* in [ME] **meglm** for details.

`melogit` supports multilevel weights and survey data; see *Methods and formulas* in [ME] **meglm** for details.

References

Andrews, M. J., T. Schank, and R. Upward. 2006. Practical fixed-effects estimation methods for the three-way error-components model. *Stata Journal* 6: 461–481.

Harbord, R. M., and P. Whiting. 2009. metandi: Meta-analysis of diagnostic accuracy using hierarchical logistic regression. *Stata Journal* 9: 211–229.

Huq, N. M., and J. Cleland. 1990. *Bangladesh Fertility Survey 1989 (Main Report).* National Institute of Population Research and Training.

Joe, H. 2008. Accuracy of Laplace approximation for discrete response mixed models. *Computational Statistics and Data Analysis* 52: 5066–5074. https://doi.org/10.1016/j.csda.2008.05.002.

Laird, N. M., and J. H. Ware. 1982. Random-effects models for longitudinal data. *Biometrics* 38: 963–974. https://doi.org/10.2307/2529876.

Lin, X., and N. E. Breslow. 1996. Bias correction in generalized linear mixed models with multiple components of dispersion. *Journal of the American Statistical Association* 91: 1007–1016. https://doi.org/10.2307/2291720.

Marchenko, Y. V. 2006. Estimating variance components in Stata. *Stata Journal* 6: 1–21.

McCulloch, C. E., S. R. Searle, and J. M. Neuhaus. 2008. *Generalized, Linear, and Mixed Models.* 2nd ed. Hoboken, NJ: Wiley.

McLachlan, G. J., and K. E. Basford. 1988. *Mixture Models: Inference and Applications to Clustering.* New York: Dekker.

Ng, E. S.-W., J. R. Carpenter, H. Goldstein, and J. Rasbash. 2006. Estimation in generalised linear mixed models with binary outcomes by simulated maximum likelihood. *Statistical Modelling* 6: 23–42. https://doi.org/10.1191/1471082X06st106oa.

Paterson, L. 1991. Socio-economic status and educational attainment: A multidimensional and multilevel study. *Evaluation and Research in Education* 5: 97–121. https://doi.org/10.1080/09500799109533303.

Pinheiro, J. C., and E. C. Chao. 2006. Efficient Laplacian and adaptive Gaussian quadrature algorithms for multilevel generalized linear mixed models. *Journal of Computational and Graphical Statistics* 15: 58–81. https://doi.org/10.1198/106186006X96962.

Rabe-Hesketh, S., and A. Skrondal. 2022. *Multilevel and Longitudinal Modeling Using Stata*. 4th ed. College Station, TX: Stata Press.

Rabe-Hesketh, S., T. Toulopoulou, and R. M. Murray. 2001. Multilevel modeling of cognitive function in schizophrenic patients and their first degree relatives. *Multivariate Behavioral Research* 36: 279–298. https://doi.org/10.1207/S15327906MBR3602_07.

Self, S. G., and K.-Y. Liang. 1987. Asymptotic properties of maximum likelihood estimators and likelihood ratio tests under nonstandard conditions. *Journal of the American Statistical Association* 82: 605–610. https://doi.org/10.2307/2289471.

Also see

[ME] **melogit postestimation** — Postestimation tools for melogit

[ME] **mecloglog** — Multilevel mixed-effects complementary log–log regression

[ME] **meprobit** — Multilevel mixed-effects probit regression

[ME] **me** — Introduction to multilevel mixed-effects models

[BAYES] **bayes: melogit** — Bayesian multilevel logistic regression

[SEM] **Intro 5** — Tour of models (*Multilevel mixed-effects models*)

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

[XT] **xtlogit** — Fixed-effects, random-effects, and population-averaged logit models

[U] 20 Estimation and postestimation commands