Title

> **meqrlogit** — Multilevel mixed-effects logistic regression (QR decomposition)

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

Description

meqrlogit is a legacy command for fitting mixed-effects models to binary or binomial responses. melogit is the modern command, and it offers more functionality; see [ME] **melogit**. The two commands use different but equivalent estimation methods. melogit performs optimization using variance components in their original metric, whereas meqrlogit uses the QR decomposition of the variance-components matrix.

Quick start

Two-level logistic regression of y on x with random intercepts by lev2 using QR decomposition
      meqrlogit y x || lev2:

Add random coefficients for x
      meqrlogit y x || lev2: x

As above, but allow the random effects to be correlated
      meqrlogit y x || lev2: x, covariance(unstructured)

Three-level random-intercept model of y on x with lev2 nested within lev3
      meqrlogit y x || lev3: || lev2:

Crossed-effects model of y on x with two-way crossed random effects by factors a and b
      meqrlogit y x || _all:R.a || b:

Menu

Statistics > Multilevel mixed-effects models > Estimation by QR decomposition > Logistic regression

Syntax

>   meqrlogit *depvar* *fe_equation* || *re_equation* [|| *re_equation* ...] [, *options*]

where the syntax of *fe_equation* is

>   [*indepvars*] [*if*] [*in*] [, *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

>   *levelvar*: R.*varname* [, *re_options*]

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

re_options	Description
Model	
covariance(*vartype*)	variance–covariance structure of the random effects
noconstant	suppress constant term from the random-effects equation
collinear	keep collinear variables

options	Description
Model	
<u>bin</u>omial(*varname* | *#*)	set binomial trials if data are in binomial form
Reporting	
<u>level</u>(*#*)	set confidence level; default is level(95)
or	report fixed-effects coefficients as odds ratios
<u>var</u>iance	show random-effects parameter estimates as variances and covariances; the default
<u>stddev</u>iations	show random-effects parameter estimates as standard deviations and correlations
<u>nore</u>table	suppress random-effects table
<u>nofe</u>table	suppress fixed-effects table
estmetric	show parameter estimates as stored in e(b)
<u>nohe</u>ader	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	
<u>intp</u>oints(*#* [*#* ...])	set the number of integration (quadrature) points; default is intpoints(7)
laplace	use Laplacian approximation; equivalent to intpoints(1)
Maximization	
maximize_options	control the maximization process; seldom used
<u>retol</u>erance(*#*)	tolerance for random-effects estimates; default is retolerance(1e-8); seldom used
<u>reiter</u>ate(*#*)	maximum number of iterations for random-effects estimation; default is reiterate(50); seldom used
matsqrt	parameterize variance components using matrix square roots; the default
matlog	parameterize variance components using matrix logarithms
refineopts(*maximize_options*)	control the maximization process during refinement of starting values
<u>coefl</u>egend	display legend instead of statistics

vartype	Description
<u>ind</u>ependent	one unique variance parameter per random effect, all covariances 0; the default unless the R. notation is used
<u>exch</u>angeable	equal variances for random effects, and one common pairwise covariance
identity	equal variances for random effects, all covariances 0; the default if the R. notation is used
<u>un</u>structured	all variances and covariances to be distinctly estimated

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

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

`bootstrap`, `by`, `jackknife`, `mi estimate`, `rolling`, and `statsby` are allowed; see [U] **11.1.10 Prefix commands**.

`coeflegend` does 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.

`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`, and `unstructured`.

　　`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)`, except when the `R.` notation is used, in which case the default is `covariance(identity)` and only `covariance(identity)` and `covariance(exchangeable)` are allowed.

　　`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.

`collinear` specifies that `meqrlogit` not omit collinear variables from the random-effects equation. Usually, there is no reason to leave collinear variables in place; in fact, doing so usually causes the estimation to fail because of the matrix singularity caused by the collinearity. However, with certain models (for example, a random-effects model with a full set of contrasts), the variables may be collinear, yet the model is fully identified because of restrictions on the random-effects covariance structure. In such cases, using the `collinear` option allows the estimation to take place with the random-effects equation intact.

`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.

┌─── 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.

variance, the default, displays the random-effects parameter estimates as variances and covariances.

stddeviations displays the random-effects parameter estimates as standard deviations and correlations.

noretable suppresses the random-effects table.

nofetable suppresses the fixed-effects table.

estmetric displays all parameter estimates in one table using the metric in which they are stored in e(b). The results are stored in the same metric regardless of the parameterization of the variance components, matsqrt or matlog, used at estimation time. Random-effects parameter estimates are stored as log-standard deviations and hyperbolic arctangents of correlations, with equation names that organize them by model level. Note that fixed-effects estimates are always stored and displayed in the same metric.

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

intpoints(*#* [*#* ...]) sets the number of integration points for adaptive Gaussian quadrature. The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases with the number of quadrature points, and in models with many levels or many random coefficients, this increase can be substantial.

You may specify one number of integration points applying to all levels of random effects in the model, or you may specify distinct numbers of points for each level. intpoints(7) is the default; that is, by default seven quadrature points are used for each level.

laplace specifies that log likelihoods be calculated using the Laplacian approximation, equivalent to adaptive Gaussian quadrature with one integration point for each level in the model; laplace is equivalent to intpoints(1). Computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. The computational time saved by using laplace can thus be substantial, especially when you have many levels or random coefficients.

The Laplacian approximation has been known to produce biased parameter estimates, but the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects. If your interest lies primarily with the fixed-effects estimates, the Laplace approximation may be a viable faster alternative to adaptive quadrature with multiple integration points.

When the R.*varname* notation is used, the dimension of the random effects increases by the number of distinct values of *varname*. Even when this number is small to moderate, it increases the total random-effects dimension to the point where estimation with more than one quadrature point is prohibitively intensive.

For this reason, when you use the R. notation in your random-effects equations, the laplace option is assumed. You can override this behavior by using the intpoints() option, but doing so is not recommended.

     Maximization

maximize_options: <u>difficult</u>, <u>tech</u>nique(*algorithm_spec*), <u>iter</u>ate(*#*), [<u>no</u>]<u>log</u>, <u>trace</u>,
    <u>grad</u>ient, <u>showstep</u>, <u>hess</u>ian, <u>showtol</u>erance, <u>tol</u>erance(*#*), <u>ltol</u>erance(*#*),
    <u>nrtol</u>erance(*#*), <u>nonrtol</u>erance, and <u>from</u>(*init_specs*); see [R] **maximize**. Those that require
    special mention for meqrlogit are listed below.

    For the technique() option, the default is technique(nr). The bhhh algorithm may not be
    specified.

    from(*init_specs*) is particularly useful when combined with refineopts(iterate(0)) (see the
    description below), which bypasses the initial optimization stage.

<u>retol</u>erance(*#*) specifies the convergence tolerance for the estimated random effects used by adaptive
    Gaussian quadrature. Although not estimated as model parameters, random-effects estimators are
    used to adapt the quadrature points. Estimating these random effects is an iterative procedure,
    with convergence declared when the maximum relative change in the random effects is less than
    retolerance(). The default is retolerance(1e-8). You should seldom have to use this option.

<u>reiter</u>ate(*#*) specifies the maximum number of iterations used when estimating the random effects
    to be used in adapting the Gaussian quadrature points; see the retolerance() option. The default
    is reiterate(50). You should seldom have to use this option.

<u>matsqrt</u> (the default), during optimization, parameterizes variance components by using the matrix
    square roots of the variance–covariance matrices formed by these components at each model level.

<u>matlog</u>, during optimization, parameterizes variance components by using the matrix logarithms of
    the variance–covariance matrices formed by these components at each model level.

    The matsqrt parameterization ensures that variance–covariance matrices are positive semidefinite,
    while matlog ensures matrices that are positive definite. For most problems, the matrix square root
    is more stable near the boundary of the parameter space. However, if convergence is problematic,
    one option may be to try the alternate matlog parameterization. When convergence is not an issue,
    both parameterizations yield equivalent results.

<u>refineopts</u>(*maximize_options*) controls the maximization process during the refinement of starting
    values. Estimation in meqrlogit takes place in two stages. In the first stage, starting values
    are refined by holding the quadrature points fixed between iterations. During the second stage,
    quadrature points are adapted with each evaluation of the log likelihood. Maximization options
    specified within refineopts() control the first stage of optimization; that is, they control the
    refining of starting values.

    maximize_options specified outside refineopts() control the second stage.

    The one exception to the above rule is the nolog option, which when specified outside refine-
    opts() applies globally.

    from(*init_specs*) is not allowed within refineopts() and instead must be specified globally.

    Refining starting values helps make the iterations of the second stage (those that lead toward the so-
    lution) more numerically stable. In this regard, of particular interest is refineopts(iterate(*#*)),
    with two iterations being the default. Should the maximization fail because of instability in the
    Hessian calculations, one possible solution may be to increase the number of iterations here.

The following option is available with meqrlogit but is not shown in the dialog box:

<u>coeflegend</u>; see [R] **estimation options**.

Remarks and examples

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.

`meqrlogit` allows for many levels of random effects. For example, in a three-level model you can specify random effects for schools and then random effects for classes nested within schools. In this model, the observations (presumably, the students) comprise the first level, the classes comprise the second level, and the schools comprise the third.

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 $\mathbf{u}_j$,

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

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 *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) $\boldsymbol{\beta}$.

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 $\mathbf{u}_j$.

Model (1) is an example of a generalized linear mixed model (GLMM), which generalizes the linear mixed-effects (LME) model to non-Gaussian responses. You can fit LMEs in Stata by using `mixed` and fit GLMMs by using `meglm`. Because of the relationship between LMEs and GLMMs, there is insight to be gained through examination of the linear mixed model. This is especially true for Stata users because the terminology, syntax, options, and output for fitting these types of models are nearly identical. See [ME] **mixed** and the references therein, particularly in *Introduction*, for more information.

Multilevel models with binary responses have been used extensively in the health and social sciences. As just one example, Leyland and Goldstein (2001, sec. 3.6) describe a study of equity of healthcare in Great Britain. Multilevel models with binary and other limited dependent responses also have a long history in econometrics; Rabe-Hesketh, Skrondal, and Pickles (2005) provide an excellent survey.

Log-likelihood calculations for fitting any generalized mixed-effects model require integrating out the random effects. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. The estimation method

used by meqrlogit is a multicoefficient and multilevel extension of one of these quadrature types, namely, adaptive Gaussian quadrature (AGQ) based on conditional modes, with the multicoefficient extension from Pinheiro and Bates (1995) and the multilevel extension from Pinheiro and Chao (2006); see *Methods and formulas*.

▷ 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 http://www.stata-press.com/data/r15/bangladesh
(Bangladesh Fertility Survey, 1989)

. describe

Contains data from http://www.stata-press.com/data/r15/bangladesh.dta
  obs:         1,934                          Bangladesh Fertility Survey, 1989
 vars:             7                          28 May 2016 20:27
 size:        19,340                          (_dta has notes)
```

variable name	storage type	display format	value 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

```
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 three indicator variables recording number of children.

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

$$\text{logit}(\pi_{ij}) = \beta_0 + \beta_1 \text{urban}_{ij} + \beta_2 \text{age}_{ij} + \beta_3 \text{child1}_{ij} + \beta_4 \text{child2}_{ij} + \beta_5 \text{child3}_{ij} + u_j \quad (2)$$

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

```
. meqrlogit c_use urban age child* || district:

Refining starting values:

Iteration 0:   log likelihood = -1219.2682
Iteration 1:   log likelihood = -1209.3544
Iteration 2:   log likelihood = -1207.1895

Performing gradient-based optimization:

Iteration 0:   log likelihood = -1207.1895
Iteration 1:   log likelihood = -1206.8323
Iteration 2:   log likelihood = -1206.8322
Iteration 3:   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 points =   7                    Wald chi2(5)      =     109.60
Log likelihood = -1206.8322                 Prob > chi2       =     0.0000
```

c_use	Coef.	Std. Err.	z	P>|z|	[95% Conf. Interval]	
urban	.7322764	.1194857	6.13	0.000	.4980887	.9664641
age	-.0264982	.0078916	-3.36	0.001	-.0419654	-.0110309
child1	1.116002	.1580921	7.06	0.000	.8061466	1.425856
child2	1.365895	.174669	7.82	0.000	1.02355	1.70824
child3	1.344031	.1796549	7.48	0.000	.9919141	1.696148
_cons	-1.68929	.1477592	-11.43	0.000	-1.978892	-1.399687

Random-effects Parameters	Estimate	Std. Err.	[95% Conf. Interval]	
district: Identity				
var(_cons)	.2156188	.0733234	.1107202	.4199007

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

Notes:

1. The estimation log consists of two parts:

   (a) A set of iterations aimed at refining starting values. These are designed to be relatively quick iterations aimed at getting the parameter estimates within a neighborhood of the eventual solution, making the iterations in (b) more numerically stable.

   (b) A set of gradient-based iterations. By default, these are Newton–Raphson iterations, but other methods are available by specifying the appropriate *maximize_options*; see [R] **maximize**.

2. The first estimation table reports the fixed effects, and these 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.

3. The second estimation table shows the estimated variance components. The first section of the table is labeled district: Identity, meaning that these are random effects at the district level and that their variance–covariance matrix is a multiple of the identity matrix; that is, $\Sigma = \sigma_u^2 \mathbf{I}$.

Because we have only one random effect at this level, meqrlogit knew that Identity is the only possible covariance structure. In any case, σ_u^2 was estimated as 0.22 with standard error 0.07.

4. A likelihood-ratio test comparing the model to ordinary logistic regression, (2) without u_j, is provided and is highly significant for these data.

◁

The above extends to models with more than two levels of nesting by adding more random-effects equations, each separated by ||.

Stored results

meqrlogit stores the following in e():

Scalars

e(N)	number of observations
e(k)	number of parameters
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(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(reparm_rc)	return code, final reparameterization
e(rc)	return code
e(converged)	1 if converged, 0 otherwise

Macros

e(cmd)	meqrlogit
e(cmdline)	command as typed
e(depvar)	name of dependent variable
e(ivars)	grouping variables
e(model)	logistic
e(title)	title in estimation output
e(offset)	offset
e(binomial)	binomial number of trials
e(redim)	random-effects dimensions
e(vartypes)	variance-structure types
e(revars)	random-effects covariates
e(n_quad)	number of integration points
e(laplace)	laplace, if Laplace approximation
e(chi2type)	Wald; type of model χ^2
e(method)	ML
e(opt)	type of optimization
e(ml_method)	type of ml method
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(marginsdefault)	default predict() specification for margins

e(asbalanced)	factor variables fvset as asbalanced
e(asobserved)	factor variables fvset as asobserved

Matrices

e(b)	coefficient 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

Functions

e(sample)	marks estimation sample

Methods and formulas

Model (1) assumes Bernoulli data, a special case of the binomial. Because binomial data are also supported by meqrlogit (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

$$
f(\mathbf{y}_j|\mathbf{u}_j) = \prod_{i=1}^{n_j} \left[\binom{r_{ij}}{y_{ij}} \left\{ H\left(\mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_{ij}\mathbf{u}_j\right) \right\}^{y_{ij}} \left\{ 1 - H\left(\mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_{ij}\mathbf{u}_j\right) \right\}^{r_{ij} - y_{ij}} \right]
$$

$$
= \exp\left(\sum_{i=1}^{n_j} \left[y_{ij} \left(\mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_{ij}\mathbf{u}_j\right) - r_{ij} \log\left\{ 1 + \exp\left(\mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_{ij}\mathbf{u}_j\right) \right\} + \log\binom{r_{ij}}{y_{ij}} \right] \right)
$$

for $H(v) = \exp(v)/\{1 + \exp(v)\}$.

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,

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

where $\mathbf{X}_j$ is formed by stacking the row vectors $\mathbf{x}_{ij}$ and $\mathbf{Z}_j$ is formed by stacking the row vectors $\mathbf{z}_{ij}$. 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

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

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. The Laplacian approximation (Tierney and Kadane 1986; Pinheiro and Bates 1995) is based on a second-order Taylor expansion of $h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j\right)$ about the value of $\mathbf{u}_j$ that maximizes it. Taking first and second derivatives, we obtain

$$h'\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j\right) = \frac{\partial h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j\right)}{\partial \mathbf{u}_j} = \mathbf{Z}_j'\left\{\mathbf{y}_j - \mathbf{m}(\boldsymbol{\beta}, \mathbf{u}_j)\right\} - \boldsymbol{\Sigma}^{-1}\mathbf{u}_j$$

$$h''\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j\right) = \frac{\partial^2 h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j\right)}{\partial \mathbf{u}_j \partial \mathbf{u}_j'} = -\left\{\mathbf{Z}_j'\mathbf{V}(\boldsymbol{\beta}, \mathbf{u}_j)\mathbf{Z}_j + \boldsymbol{\Sigma}^{-1}\right\}$$

where $\mathbf{m}(\boldsymbol{\beta}, \mathbf{u}_j)$ is the vector function with the ith element equal to the conditional mean of y_{ij} given $\mathbf{u}_j$, that is, $r_{ij}H(\mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_{ij}\mathbf{u}_j)$. $\mathbf{V}(\boldsymbol{\beta}, \mathbf{u}_j)$ is the diagonal matrix whose diagonal entries v_{ij} are the conditional variances of y_{ij} given $\mathbf{u}_j$, namely,

$$v_{ij} = r_{ij}H\left(\mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_{ij}\mathbf{u}_j\right)\left\{1 - H\left(\mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_{ij}\mathbf{u}_j\right)\right\}$$

The maximizer of $h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j\right)$ is $\widehat{\mathbf{u}}_j$ such that $h'\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j\right) = \mathbf{0}$. The integrand in (5) is proportional to the posterior density $f(\mathbf{u}_j|\mathbf{y}_j)$, so $\widehat{\mathbf{u}}_j$ also represents the posterior mode, a plausible estimator of $\mathbf{u}_j$ in its own right.

Given the above derivatives, the second-order Taylor approximation then takes the form

$$h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j\right) \approx h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j\right) + \frac{1}{2}\left(\mathbf{u}_j - \widehat{\mathbf{u}}_j\right)' h''\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j\right)\left(\mathbf{u}_j - \widehat{\mathbf{u}}_j\right) \tag{6}$$

The first-derivative term vanishes because $h'\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j\right) = \mathbf{0}$. Therefore,

$$\int \exp\left\{h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j\right)\right\} d\mathbf{u}_j \approx \exp\left\{h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j\right)\right\}$$

$$\times \int \exp\left[-\frac{1}{2}\left(\mathbf{u}_j - \widehat{\mathbf{u}}_j\right)'\left\{-h''\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j\right)\right\}\left(\mathbf{u}_j - \widehat{\mathbf{u}}_j\right)\right] d\mathbf{u}_j \tag{7}$$

$$= \exp\left\{h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j\right)\right\}(2\pi)^{q/2}\left|-h''\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j\right)\right|^{-1/2}$$

because the latter integrand can be recognized as the "kernel" of a multivariate normal density.

Combining the above with (5) (and taking logs) gives the Laplacian log-likelihood contribution of the jth cluster,

$$\mathcal{L}_j^{\mathrm{Lap}}(\boldsymbol{\beta}, \boldsymbol{\Sigma}) = -\frac{1}{2}\log|\boldsymbol{\Sigma}| - \log|\mathbf{R}_j| + h\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j\right) + c(\mathbf{y}_j, \mathbf{r}_j)$$

where $\mathbf{R}_j$ is an upper-triangular matrix such that $-h''\left(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j\right) = \mathbf{R}_j\mathbf{R}_j'$. Pinheiro and Chao (2006) show that $\widehat{\mathbf{u}}_j$ and $\mathbf{R}_j$ can be efficiently computed as the iterative solution to a least-squares problem by using matrix decomposition methods similar to those used in fitting LME models (Bates and Pinheiro 1998; Pinheiro and Bates 2000; [ME] **mixed**).

The fidelity of the Laplacian approximation is determined wholly by the accuracy of the approximation in (6). An alternative that does not depend so heavily on this approximation is integration via AGQ (Naylor and Smith 1982; Liu and Pierce 1994).

The application of AGQ to this particular problem is from Pinheiro and Bates (1995). When we reexamine the integral in question, a transformation of integration variables yields

$$
\int \exp \{h(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j)\} \, d\mathbf{u}_j = |\mathbf{R}_j|^{-1} \int \exp \{h(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j + \mathbf{R}_j^{-1}\mathbf{t})\} \, d\mathbf{t}
$$
$$
= (2\pi)^{q/2} |\mathbf{R}_j|^{-1} \int \exp \{h(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j + \mathbf{R}_j^{-1}\mathbf{t}) + \mathbf{t}'\mathbf{t}/2\} \, \phi(\mathbf{t}) d\mathbf{t}
\tag{8}
$$

where $\phi(\cdot)$ is the standard multivariate normal density. Because the integrand is now expressed as some function multiplied by a normal density, it can be estimated by applying the rules of standard Gauss–Hermite quadrature. For a predetermined number of quadrature points N_Q, define $a_k = \sqrt{2}a_k^*$ and $w_k = w_k^*/\sqrt{\pi}$, for $k = 1, \ldots, N_Q$, where (a_k^*, w_k^*) are a set of abscissas and weights for Gauss–Hermite quadrature approximations of $\int \exp(-x^2)f(x)dx$, as obtained from Abramowitz and Stegun (1964, 924).

Define $\mathbf{a_k} = (a_{k_1}, a_{k_2}, \ldots, a_{k_q})'$; that is, $\mathbf{a_k}$ is a vector that spans the N_Q abscissas over the dimension q of the random effects. Applying quadrature rules to (8) yields the AGQ approximation,

$$
\int \exp \{h(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \mathbf{u}_j)\} \, d\mathbf{u}_j
$$
$$
\approx (2\pi)^{q/2} |\mathbf{R}_j|^{-1} \sum_{k_1=1}^{N_Q} \cdots \sum_{k_q=1}^{N_Q} \left[\exp \{h(\boldsymbol{\beta}, \boldsymbol{\Sigma}, \widehat{\mathbf{u}}_j + \mathbf{R}_j^{-1}\mathbf{a_k}) + \mathbf{a_k}'\mathbf{a_k}/2\} \prod_{p=1}^{q} w_{k_p} \right]
$$
$$
\equiv (2\pi)^{q/2} \widehat{G}_j(\boldsymbol{\beta}, \boldsymbol{\Sigma})
$$

resulting in the AGQ log-likelihood contribution of the ith cluster,

$$
\mathcal{L}_j^{\mathrm{AGQ}}(\boldsymbol{\beta}, \boldsymbol{\Sigma}) = -\frac{1}{2} \log |\boldsymbol{\Sigma}| + \log \left\{ \widehat{G}_j(\boldsymbol{\beta}, \boldsymbol{\Sigma}) \right\} + c(\mathbf{y}_j, \mathbf{r}_j)
$$

The "adaptive" part of adaptive Gaussian quadrature lies in the translation and rescaling of the integration variables in (8) by using $\widehat{\mathbf{u}}_j$ and $\mathbf{R}_j^{-1}$, respectively. This transformation of quadrature abscissas (centered at 0 in standard form) is chosen to better capture the features of the integrand, which through (7) can be seen to resemble a multivariate normal distribution with mean $\widehat{\mathbf{u}}_j$ and variance $\mathbf{R}_j^{-1}\mathbf{R}_j^{-T}$. AGQ is therefore not as dependent as the Laplace method upon the approximation in (6). In AGQ, (6) serves merely to redirect the quadrature abscissas, with the AGQ approximation improving as the number of quadrature points N_Q increases. In fact, Pinheiro and Bates (1995) point out that AGQ with only one quadrature point ($a = 0$ and $w = 1$) reduces to the Laplacian approximation.

The log likelihood for the entire dataset is then simply the sum of the contributions of the M individual clusters, namely, $\mathcal{L}(\boldsymbol{\beta}, \boldsymbol{\Sigma}) = \sum_{j=1}^{M} \mathcal{L}_j^{\mathrm{Lap}}(\boldsymbol{\beta}, \boldsymbol{\Sigma})$ for Laplace and $\mathcal{L}(\boldsymbol{\beta}, \boldsymbol{\Sigma}) = \sum_{j=1}^{M} \mathcal{L}_j^{\mathrm{AGQ}}(\boldsymbol{\beta}, \boldsymbol{\Sigma})$ for AGQ.

Maximization of $\mathcal{L}(\boldsymbol{\beta}, \boldsymbol{\Sigma})$ is performed with respect to $(\boldsymbol{\beta}, \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a vector comprising the unique elements of the matrix square root of $\boldsymbol{\Sigma}$. This is done to ensure that $\boldsymbol{\Sigma}$ is always positive semidefinite. If the matlog option is specified, then $\boldsymbol{\theta}$ instead consists of the unique elements of the matrix logarithm of $\boldsymbol{\Sigma}$. For well-conditioned problems, both methods produce equivalent results, yet our experience deems the former as more numerically stable near the boundary of the parameter space.

Once maximization is achieved, parameter estimates are mapped from $(\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\theta}})$ to $(\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\gamma}})$, where $\widehat{\boldsymbol{\gamma}}$ is a vector containing the unique (estimated) elements of $\boldsymbol{\Sigma}$, expressed as logarithms of standard deviations for the diagonal elements and hyperbolic arctangents of the correlations for off-diagonal elements. This last step is necessary to (a) obtain a parameterization under which parameter estimates can be displayed and interpreted individually, rather than as elements of a matrix square root (or logarithm), and (b) parameterize these elements such that their ranges each encompass the entire real line.

Parameter estimates are stored in `e(b)` as $(\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\gamma}})$, with the corresponding variance–covariance matrix stored in `e(V)`. Parameter estimates can be displayed in this metric by specifying the `estmetric` option. However, in `meqrlogit` output, variance components are displayed as variances and covariances.

The approach outlined above can be extended from two-level models to higher-level models; see Pinheiro and Chao (2006) for details.

References

Abramowitz, M., and I. A. Stegun, ed. 1964. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.* Washington, DC: National Bureau of Standards.

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.

Bates, D. M., and J. C. Pinheiro. 1998. Computational methods for multilevel modelling. In *Technical Memorandum BL0112140-980226-01TM.* Murray Hill, NJ: Bell Labs, Lucent Technologies. http://ect.bell-labs.com/sl/project/nlme/CompMulti.pdf.

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.

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 & Data Analysis* 52: 5066–5074.

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

Leyland, A. H., and H. Goldstein, ed. 2001. *Multilevel Modelling of Health Statistics.* New York: Wiley.

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.

Liu, Q., and D. A. Pierce. 1994. A note on Gauss–Hermite quadrature. *Biometrika* 81: 624–629.

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.

Naylor, J. C., and A. F. M. Smith. 1982. Applications of a method for the efficient computation of posterior distributions. *Journal of the Royal Statistical Society, Series C* 31: 214–225.

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.

Palmer, T. M., C. M. Macdonald-Wallis, D. A. Lawlor, and K. Tilling. 2014. Estimating adjusted associations between random effects from multilevel models: The reffadjust package. *Stata Journal* 14: 119–140.

Pinheiro, J. C., and D. M. Bates. 1995. Approximations to the log-likelihood function in the nonlinear mixed-effects model. *Journal of Computational and Graphical Statistics* 4: 12–35.

——. 2000. *Mixed-Effects Models in S and S-PLUS*. New York: Springer.

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.

Rabe-Hesketh, S., A. Skrondal, and A. Pickles. 2005. Maximum likelihood estimation of limited and discrete dependent variable models with nested random effects. *Journal of Econometrics* 128: 301–323.

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.

Tierney, L., and J. B. Kadane. 1986. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association* 81: 82–86.

Also see

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

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

[ME] **melogit** — Multilevel mixed-effects logistic regression

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

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

[MI] **estimation** — Estimation commands for use with mi estimate

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

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

[U] **20 Estimation and postestimation commands**