menl postestimation — Postestimation tools for menl

Postestimation commands

The following postestimation commands are of special interest after `menl`:

Command	Description
`estat group`	summarize the composition of the nested groups
`estat recovariance`	display the estimated random-effects covariance matrices
`estat sd`	display variance components as standard deviations and correlations
`estat wcorrelation`	display within-cluster correlations and standard deviations

The following standard postestimation commands are also available:

Command	Description
`contrast`	contrasts and ANOVA-style joint tests of parameters
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estimates`	cataloging estimation results
`etable`	table of estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
`lrtest`	likelihood-ratio test
`margins`	marginal means, predictive margins, marginal effects, and average marginal effects
`marginsplot`	graph the results from margins (profile plots, interaction plots, etc.)
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	predictions and their SEs, residuals, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`pwcompare`	pairwise comparisons of parameters
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

predict

Description for predict

`predict` creates a new variable containing predictions of mean values, residuals, or standardized residuals. It can also create multiple new variables containing estimates of random effects and their standard errors or containing predicted named substitutable expressions.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of the outcome and other statistics

`predict` [*type*] *newvar* [*if*] [*in*] [, *statistic* `fixedonly` `relevel(`*levelvar*`)` *options*]

Syntax for predicting named substitutable expressions (parameters)

Predict all parameters

`predict` [*type*] {*stub*`*`|*newvarlist*} [*if*] [*in*], `parameters`
[`fixedonly` `relevel(`*levelvar*`)` *options*]

Predict specific parameters

`predict` [*type*] (*newvar* = {*param*`:`}) [(*newvar* = {*param*`:`})] [...] [*if*] [*in*]
[, `fixedonly` `relevel(`*levelvar*`)` *options*]

`predict` [*type*] {*stub*`*`|*newvarlist*} [*if*] [*in*], `parameters(`*paramnames*`)`
[`fixedonly` `relevel(`*levelvar*`)` *options*]

Syntax for obtaining predictions of random effects and their standard errors

`predict` [*type*] {*stub*`*`|*newvarlist*} [*if*] [*in*], `reffects` [`relevel(`*levelvar*`)`
`reses(`*stub*`*`|*newvarlist*`)` *options*]

paramnames is *param* [*param* [...]] and *param* is a name of a substitutable expression as specified in one of `menl`'s `define()` options.

statistic	Description
Main	
yhat	prediction for the expected response conditional on the random effects
mu	synonym for yhat
residuals	residuals, response minus predicted values
* rstandard	standardized residuals

Unstarred statistics are available both in and out of sample; type predict ... if e(sample) ... if wanted only for the estimation sample. Starred statistics are calculated only for the estimation sample, even when if e(sample) is not specified.

options	Description
Options	
iterate(#)	maximum number of iterations when computing random effects; default is iterate(10)
tolerance(#)	convergence tolerance when computing random effects; default is tolerance(1e-6)
nrtolerance(#)	scaled gradient tolerance when computing random effects; default is nrtolerance(1e-5)
nonrtolerance	ignore the nrtolerance() option

Options for predict

Main

yhat calculates the predicted values, which are the mean-response values conditional on the random effects, $\mu(\mathbf{x}'_{ij}, \widehat{\boldsymbol{\beta}}, \hat{\mathbf{u}}_j)$. By default, the predicted values account for random effects from all levels in the model; however, if the relevel(*levelvar*) option is specified, then the predicted values are fit beginning with the topmost level down to and including level *levelvar*. For example, if classes are nested within schools, then typing

```
. predict yhat_school, yhat relevel(school)
```

would produce school-level predictions. That is, the predictions would incorporate school-specific random effects but not those for each class nested within each school. If the fixedonly option is specified, predicted values conditional on zero random effects, $\mu(\mathbf{x}'_{ij}, \widehat{\boldsymbol{\beta}}, \mathbf{0})$, are calculated based on the estimated fixed effects (coefficients) in the model when the random effects are fixed at their theoretical mean value of $\mathbf{0}$.

mu is a synonym for yhat.

residuals calculates residuals, equal to the responses minus the predicted values yhat. By default, the predicted values account for random effects from all levels in the model; however, if the relevel(*levelvar*) option is specified, then the predicted values are fit beginning at the topmost level down to and including level *levelvar*.

rstandard calculates standardized residuals, equal to the residuals multiplied by the inverse square root of the estimated error covariance matrix.

parameters and parameters(*paramnames*) calculate predictions for all or a subset of the named substitutable expressions in the model. By default, the predictions account for random effects from all levels in the model; however, if the relevel(*levelvar*) option is specified, then the predictions would incorporate random effects from the topmost level down to and including level *levelvar*. Option parameters(*param*) is useful with margins. parameters() does not appear in the dialog box.

reffects calculates predictions of the random effects. For the Lindstrom–Bates estimation method of menl, these are essentially the best linear unbiased predictions (BLUPs) of the random effects in the LME approximated log likelihood; see *Inference based on linearization* in [ME] **menl**. By default, estimates of all random effects in the model are calculated. However, if the relevel(*levelvar*) option is specified, then estimates of random effects for only level *levelvar* in the model are calculated. For example, if classes are nested within schools, then typing

```
. predict b*, reffects relevel(school)
```

would produce estimates at the school level. You must specify q new variables, where q is the number of random-effects terms in the model (or level). However, it is much easier to just specify *stub*∗ and let Stata name the variables *stub*1, *stub*2, …, *stubq* for you.

fixedonly specifies that all random effects be set to zero, equivalent to using only the fixed portion of the model.

relevel(*levelvar*) specifies the level in the model at which predictions involving random effects are to be obtained; see the options above for the specifics. *levelvar* is the name of the model level; it is the name of the variable describing the grouping at that level.

reses(*stub*∗ | *newvarlist*) calculates the standard errors of the estimates of the random effects. By default, standard errors for all random effects in the model are calculated. However, if the relevel(*levelvar*) option is specified, then standard errors of the estimates of the random effects for only level *levelvar* in the model are calculated; see the reffects option.

You must specify q new variables, where q is the number of random-effects terms in the model (or level). However, it is much easier to just specify *stub*∗ and let Stata name the variables *stub*1, *stub*2, …, *stubq* for you. The new variables will have the same storage type as the corresponding random-effects variables.

The reffects and reses() options often generate multiple new variables at once. When this occurs, the random effects (or standard errors) contained in the generated variables correspond to the order in which the variance components are listed in the output of menl. Still, examining the variable labels of the generated variables (with the describe command, for instance) can be useful in deciphering which variables correspond to which terms in the model.

Options

iterate(#) specifies the maximum number of iterations when computing estimates of the random effects. The default is iterate(10). This option is relevant only to predictions that depend on random effects. This option is not allowed if the fixedonly option is specified.

tolerance(#) specifies a convergence tolerance when computing estimates of the random effects. The default is tolerance(1e-6). This option is relevant only to predictions that depend on random effects. This option is not allowed if the fixedonly option is specified.

nrtolerance(#) and nonrtolerance control the tolerance for the scaled gradient when computing estimates of the random effects.

`nrtolerance(#)` specifies the tolerance for the scaled gradient. Convergence is declared when $g(-H^{-1})g'$ is less than `nrtolerance(#)`, where g is the gradient row vector and H is the approximated Hessian matrix from the current iteration. The default is `nrtolerance(1e-5)`.

`nonrtolerance` specifies that the default `nrtolerance()` criterion be turned off.

margins

Description for margins

`margins` estimates margins of response for predicted mean values or named substitutable expressions.

Menu for margins

Statistics > Postestimation

Syntax for margins

`margins` [*marginlist*] [`,` *options*]

`margins` [*marginlist*] `,` `predict(`*statistic* `...)` [*options*]

statistic	Description
`yhat`	predicted values conditional on zero random effects; the default
`mu`	synonym for `yhat`
`residuals`	not allowed with `margins`
`rstandard`	not allowed with `margins`
`parameters`	predicted parameters
`parameters(`*param*`)`	predicted named substitutable expression *param* conditional on zero random effects
`reffects`	not allowed with `margins`

The `fixedonly` option is assumed for the predictions used with `margins`.

Statistics not allowed with `margins` are functions of stochastic quantities other than `e(b)`.

For the full syntax, see [R] **margins**.

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting an NLME model using `menl`. For the most part, calculation centers on obtaining estimates of the random effects. Random effects are not estimated when the model is fit but instead need to be predicted after estimation. The estimates of the random effects are in turn used to obtain predicted values and residuals at different nesting levels. These are useful for checking model assumptions and may be used in general as model-building tools.

▷ Example 1: Testing variance components

In example 9 and example 12 of [ME] **menl**, we modeled the average leaf weight of two genotypes of soybean plants over three growing seasons as

$$\texttt{weight}_{ij} = \frac{\phi_{1j}}{1 + \exp\left\{-\left(\texttt{time}_{ij} - \phi_{2j}\right)/\phi_{3j}\right\}} + \epsilon_{ij}$$

for $j = 1, \ldots, 48$ and $i = 1, \ldots, n_j$, with $8 \leq n_j \leq 10$. Here we consider a simplified version of the stage 2 model specification from example 12,

$$\boldsymbol{\phi}_j = \begin{bmatrix} \phi_{1j} \\ \phi_{2j} \\ \phi_{3j} \end{bmatrix} = \begin{bmatrix} \beta_{11} + \beta_{12} S_{89,j} + \beta_{13} S_{90,j} + u_{1j} \\ \beta_{21} + \beta_{22} S_{89,j} + \beta_{23} S_{90,j} + \beta_{24} P_j \\ \beta_{31} + \beta_{32} S_{89,j} + \beta_{33} S_{90,j} \end{bmatrix}$$

where $P_j = I(\texttt{variety}_j = \texttt{P})$, $S_{89,j} = I(\texttt{year}_j = 1989)$, and $S_{90,j} = I(\texttt{year}_j = 1990)$. The random effects u_{1j}'s are normally distributed with mean 0 and variance σ^2_{u1} and errors ϵ_{ij}'s are normally distributed with mean 0 and error variance

$$\text{Var}\left(\epsilon_{ij}\right) = \sigma^2 (\widehat{\texttt{weight}}_{ij})^{2\delta}$$

Let's fit this model using `menl`.

```
. use https://www.stata-press.com/data/r19/soybean
(Growth of soybean plants (Davidian and Giltinan, 1995))
. menl weight = {phi1:}/(1+exp(-(time-{phi2:})/{phi3:})),
> define(phi1: i.year U1[plot])
> define(phi2: i.year i.variety)
> define(phi3: i.year, xb) resvariance(power _yhat, noconstant)
Obtaining starting values by EM:
Alternating PNLS/LME algorithm:
Iteration 1:  Linearization log likelihood = -324.21579
Iteration 2:  Linearization log likelihood = -313.89733
Iteration 3:  Linearization log likelihood = -314.76287
Iteration 4:  Linearization log likelihood =  -314.4317
Iteration 5:  Linearization log likelihood =  -314.5131
Iteration 6:  Linearization log likelihood = -314.49399
Iteration 7:  Linearization log likelihood = -314.49922
Iteration 8:  Linearization log likelihood = -314.49838
Iteration 9:  Linearization log likelihood = -314.49853
Iteration 10: Linearization log likelihood = -314.49851
```

```
Computing standard errors:
Mixed-effects ML nonlinear regression           Number of obs     =        412
Group variable: plot                            Number of groups  =         48
                                                Obs per group:
                                                              min =          8
                                                              avg =        8.6
                                                              max =         10
                                                Wald chi2(7)      =     193.98
Linearization log likelihood = -314.49851       Prob > chi2       =     0.0000
       phi1: i.year U1[plot]
       phi2: i.year i.variety
       phi3: i.year
```

weight	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
phi1						
year						
1989	-6.614797	1.195633	-5.53	0.000	-8.958195	-4.271399
1990	-3.749016	1.264716	-2.96	0.003	-6.227814	-1.270218
_cons	20.28009	.953959	21.26	0.000	18.41036	22.14981
phi2						
year						
1989	-2.623514	.9752055	-2.69	0.007	-4.534882	-.7121468
1990	-5.142726	.9879783	-5.21	0.000	-7.079128	-3.206324
variety						
P	-2.265482	.3274228	-6.92	0.000	-2.907219	-1.623745
_cons	55.25935	.7554917	73.14	0.000	53.77861	56.74008
phi3						
year						
1989	-.9538782	.1963606	-4.86	0.000	-1.338738	-.5690186
1990	-.7220007	.2081227	-3.47	0.001	-1.129914	-.3140877
_cons	8.042677	.1452638	55.37	0.000	7.757965	8.327389

Random-effects parameters	Estimate	Std. err.	[95% conf.	interval]
plot: Identity				
var(U1)	4.260962	1.114483	2.551939	7.11451
Residual variance:				
Power _yhat				
sigma2	.046573	.0038271	.0396449	.0547118
delta	.9667451	.0229472	.9217694	1.011721

`menl` does not report tests against zeros for parameters in the random-effects table because they are not appropriate for all types of parameters such as variances. For some parameters such as power parameter δ in our example, labeled as `delta` in the output, the test of H_0: $\delta = 0$ is sensible. In fact, it corresponds to the test of homoskedastic within-plot errors because under the null hypothesis the error variance $\mathrm{Var}(\epsilon_{ij}) = \sigma^2(\widehat{\texttt{weight}}_{ij})^{2\delta}$ reduces to σ^2.

We can use the `test` command to perform this test.

```
. test _b[/Residual:delta] = 0
 ( 1)  [/Residual]delta = 0
           chi2(  1) = 1774.87
         Prob > chi2 =    0.0000
```

The Wald test strongly rejects the null hypothesis of homoskedastic errors.

◁

▷ Example 2: Obtaining predictions

Continuing with example 1, we can also obtain the estimates of the plot-level random effects u_{1j}'s. Because `menl` used the Lindstrom–Bates linearization method, the estimated random effects are essentially BLUPs; see *Inference based on linearization* in [ME] **menl**.

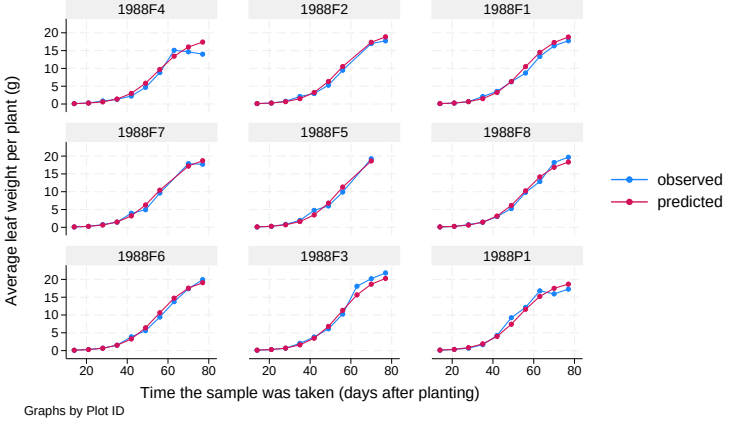
We need to specify the name of the variable to be created and then use `predict, reffects`. For example, below we obtain the predictions of random effects for the first 10 plots.

```
. predict u1, reffects
. by plot, sort: generate tolist = (_n==1)
. list plot u1 if plot <=10 & tolist
```

	plot	u1
1.	1988F4	-1.716238
11.	1988F2	-.1668753
20.	1988F1	-.2153712
30.	1988F7	-.3337681
39.	1988F5	1.331566
47.	1988F8	-.7153563
57.	1988F6	.0699629
67.	1988F3	1.353845
77.	1988P1	-.6681811
87.	1988P5	-.9152615

Next, we obtain the predicted mean values and plot them. By default, the mean response conditional on the estimated random effects is computed. Predicted values based on the fixed-effects estimates alone, that is, conditional on zero random effects, may be obtained by specifying the `fixedonly` option.

```
. predict fitweight, yhat
. twoway connected weight fitweight time if plot<=9, sort by(plot)
> ytitle("Average leaf weight per plant (g)")
> legend(order(1 "observed" 2 "predicted"))
```



The predicted values closely match the observed average leaf weights, confirming the adequacy of the model.

Also see example 13 in [ME] **menl** for how to predict parameters defined as functions of other parameters with substitutable expressions.

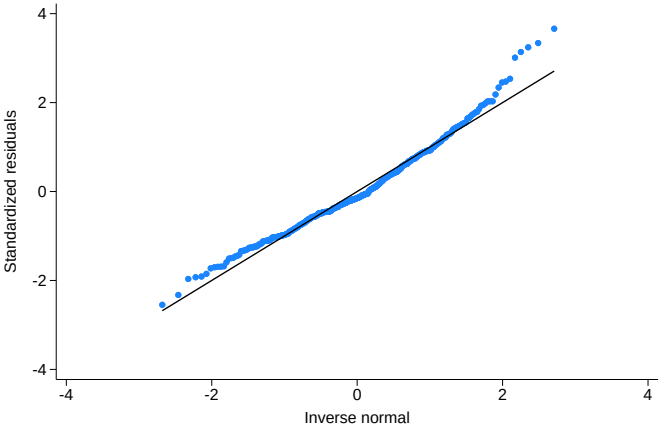
◁

▷ Example 3: Checking model assumptions based on residuals

The raw residuals are useful to check for heterogeneity of the within-group error variance; see example 10 in [ME] **menl**. They are less recommended, however, for checking normality assumptions and for detecting outlying observations. This is because raw residuals are usually correlated and have different variances. Instead, we can use standardized residuals to check for normality and outlying observations. If the normality assumption is reasonable and the model fits data well, standardized residuals should follow a standard normal distribution.

Let's check whether the standardized residuals from our model are approximately normally distributed with mean zero and variance one.

```
. predict rs, rstandard
. summarize rs
    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
          rs |        412    .0150192    .9564895  -2.547287   3.661603
. qnorm rs
```



The plot does not indicate serious departures from normality, and the estimated mean and standard deviation are close to zero and one, respectively. It appears that the power of the mean function is a reasonable choice for modeling heteroskedasticity of the within-group errors in this example.

Example 4: estat group and level-specific predictions

In example 23 of [ME] **menl**, we modeled the intensity of current at the ith level of `voltage` in the jth site within the kth wafer as

$$\texttt{current}_{ijk} = \phi_{1jk} + \phi_{2jk}\cos\left(\phi_{3jk}\texttt{voltage}_i + \pi/4\right) + \epsilon_{ijk}$$

for $k = 1,\ldots,10$, $j = 1,\ldots,8$, and $i = 1,\ldots,5$. In that example, we considered fairly complicated specifications for ϕ_j's in stage 2 with many random effects at different levels, which lead to slow execution of the command. To illustrate some of the commands available after `menl`, we will substantially simplify the stage 2 specification to speed up the estimation of the model.

$$\begin{aligned}
\phi_{1jk} &= \beta_0 + u^{(3)}_{0k} + u^{(2)}_{0j,k} + \left(\beta_1 + u^{(3)}_{1k} + u^{(2)}_{1j,k}\right)\texttt{voltage}_i \\
\phi_{2jk} &= \beta_3 \\
\phi_{3jk} &= \beta_4
\end{aligned}$$

$$\mathbf{u}^{(3)}_k = \begin{bmatrix} u^{(3)}_{0k} \\ u^{(3)}_{1k} \end{bmatrix} \sim N(\mathbf{0}, \boldsymbol{\Sigma}_3) \qquad \mathbf{u}^{(2)}_{j,k} = \begin{bmatrix} u^{(2)}_{0j,k} \\ u^{(2)}_{1j,k} \end{bmatrix} \sim N(\mathbf{0}, \boldsymbol{\Sigma}_2) \qquad \epsilon_{ijk} \sim N(0, \sigma^2_\epsilon)$$

where

$$\boldsymbol{\Sigma}_3 = \begin{bmatrix} \sigma^{(3)}_{11} & 0 \\ 0 & \sigma^{(3)}_{22} \end{bmatrix} \qquad \boldsymbol{\Sigma}_2 = \begin{bmatrix} \sigma^{(2)}_{11} & 0 \\ 0 & \sigma^{(2)}_{22} \end{bmatrix}$$

We fit this model by using menl.

```
. use https://www.stata-press.com/data/r19/wafer, clear
(Modeling of analog MOS circuits)
. menl current = {phi1:}+{phi2}*cos({phi3}*voltage + _pi/4),
> define(phi1: voltage W0[wafer] S0[wafer>site]
>              c.voltage#(W1[wafer] S1[wafer>site]))
Obtaining starting values by EM:
Alternating PNLS/LME algorithm:
Iteration 1:  Linearization log likelihood =  503.60719
Iteration 2:  Linearization log likelihood =  503.60719
Computing standard errors:
Mixed-effects ML nonlinear regression           Number of obs     =        400
      Grouping information
```

Path	No. of groups	Observations per group Minimum	Average	Maximum
wafer	10	40	40.0	40
wafer>site	80	5	5.0	5

```
                                                Wald chi2(1)      =    4860.30
Linearization log likelihood =  503.60719       Prob > chi2       =     0.0000
        phi1: voltage W0[wafer] S0[wafer>site] c.voltage#W1[wafer]
              c.voltage#S1[wafer>site]
```

current	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
phi1						
voltage	-25.20026	.3614709	-69.72	0.000	-25.90873	-24.49179
_cons	64.41187	.4984303	129.23	0.000	63.43497	65.38878
/phi2	-93.6509	.6367582	-147.07	0.000	-94.89892	-92.40287
/phi3	.3828109	.001525	251.02	0.000	.379822	.3857999

Random-effects parameters	Estimate	Std. err.	[95% conf.	interval]
wafer: Independent				
var(W0)	.0048116	.0027516	.0015686	.014759
var(W1)	.0396212	.0184213	.0159284	.0985558
wafer>site: Independent				
var(S0)	.0086449	.0017812	.0057726	.0129463
var(S1)	.0118703	.0021115	.0083763	.0168217
var(Residual)	.001069	.0000952	.0008978	.0012729

We can use `estat group` to see how the data are broken down by wafer and site:

```
. estat group
Grouping information
```

Path	No. of groups	Observations per group Minimum	Average	Maximum
wafer	10	40	40.0	40
wafer>site	80	5	5.0	5

We are reminded that we have balanced data for each site (all sites were measured at 5 ascending voltages).

Suppose that we want to predict random effects at the wafer level only; that is, we want to compute $\hat{\mathbf{u}}_k^{(3)}$. This can be done by specifying the `relevel(wafer)` option:

```
. predict u_wafer*, reffects relevel(wafer)
```

Notice how `predict` labels the generated variables for you to avoid confusion.

```
. describe u_wafer*
```

Variable name	Storage type	Display format	Value label	Variable label
u_wafer1	float	%9.0g		BLUP r.e. for W0[wafer]
u_wafer2	float	%9.0g		BLUP r.e. for W1[wafer]

We can use `predict, yhat` to get the predicted values $\mu(\texttt{voltage}_i, \widehat{\boldsymbol{\beta}}, \hat{\mathbf{u}}_k^{(3)}, \hat{\mathbf{u}}_{j,k}^{(2)})$. If instead we want to predict values at the wafer level, $\mu(\texttt{voltage}_i, \widehat{\boldsymbol{\beta}}, \hat{\mathbf{u}}_k^{(3)}, \mathbf{0})$, we again need to specify the `relevel()` option:

```
. predict curr_wafer, yhat relevel(wafer)
. list wafer site current curr_wafer in 1/10
```

	wafer	site	current	curr_w~r
1.	1	1	.90088	.8898317
2.	1	1	3.8682	3.920231
3.	1	1	7.6406	7.65254
4.	1	1	11.736	11.76189
5.	1	1	15.934	15.91457
6.	1	2	1.032	.8898317
7.	1	2	4.1022	3.920231
8.	1	2	7.9316	7.65254
9.	1	2	12.064	11.76189
10.	1	2	16.294	15.91457

The predicted values `curr_wafer` do not vary across sites, because $\mu(\texttt{voltage}_i, \widehat{\boldsymbol{\beta}}, \hat{\mathbf{u}}_k^{(3)}, \mathbf{0})$ does not depend on j.

◁

Methods and formulas

Following the notation defined throughout [ME] **menl**, estimates of random effects $\mathbf{u}_j$ are obtained by using PNLS iterations with parameters $\boldsymbol{\beta}$, $\boldsymbol{\alpha}$, and σ^2 held fixed at their values obtained at convergence. Starting with $\hat{\mathbf{u}}_j^{(0)} = \mathbf{0}$, at the kth iteration, we have

$$\hat{\mathbf{u}}_j^{(k)} = \widehat{\boldsymbol{\Sigma}}\widehat{\mathbf{Z}}_j'^{(k-1)} \left(\widehat{\mathbf{Z}}_j^{(k-1)}\widehat{\boldsymbol{\Sigma}}\widehat{\mathbf{Z}}_j'^{(k-1)} + \hat{\sigma}^2\widehat{\boldsymbol{\Lambda}}_j\right)^{-1} \left(\widehat{\mathbf{w}}_j^{(k-1)} - \widehat{\mathbf{X}}_j^{(k-1)}\widehat{\boldsymbol{\beta}}\right)$$

where $\widehat{\boldsymbol{\Sigma}}$ and $\widehat{\boldsymbol{\Lambda}}$ are $\boldsymbol{\Sigma}$ and $\boldsymbol{\Lambda}$ with maximum likelihood (ML) or restricted maximum likelihood (REML) estimates of the variance components plugged in and $\widehat{\mathbf{X}}_j^{(k-1)} = \widehat{\mathbf{X}}_j(\hat{\mathbf{u}}_j^{(k-1)})$, $\widehat{\mathbf{Z}}_j^{(k-1)} = \widehat{\mathbf{Z}}_j(\hat{\mathbf{u}}_j^{(k-1)})$, and $\widehat{\mathbf{w}}_j^{(k-1)} = \widehat{\mathbf{w}}_j(\hat{\mathbf{u}}_j^{(k-1)})$ are defined in the LME step of *Inference based on linearization* in *Methods and formulas* of [ME] **menl**. When the variance structure depends on $\mathbf{u}_j$, such as when the `resvariance(power _yhat)` option is specified during estimation, $\widehat{\boldsymbol{\Lambda}}_j$ will also be updated at each iteration; that is, $\widehat{\boldsymbol{\Lambda}}_j = \widehat{\boldsymbol{\Lambda}}_j(\hat{\mathbf{u}}_j^{(k-1)})$. The iterative process stops when the relative difference between $\hat{\mathbf{u}}_j^{(k-1)}$ and $\hat{\mathbf{u}}_j^{(k)}$ is less than `tolerance(#)` or, if the stopping rule is not met, when the maximum number of iterations in `iterate(#)` is reached.

Standard errors for the estimates of the random effects are calculated based on Bates and Pinheiro (1998, sec. 3.3). If estimation is done by REML, these standard errors account for uncertainty in the estimate of $\boldsymbol{\beta}$, whereas for ML, the standard errors treat $\boldsymbol{\beta}$ as known. As such, standard errors of REML-based estimates will usually be larger.

Predicted mean values are computed as $\boldsymbol{\mu}_j(\mathbf{X}_j, \widehat{\boldsymbol{\beta}}, \hat{\mathbf{u}}_j)$, predicted parameters as $\widehat{\boldsymbol{\phi}}_j = [\mathbf{d}](\mathbf{x}_j^b, \widehat{\boldsymbol{\beta}}, \hat{\mathbf{u}}_j)$, residuals as $\hat{\boldsymbol{\epsilon}}_j = \mathbf{y}_j - \boldsymbol{\mu}_j(\mathbf{X}_j, \widehat{\boldsymbol{\beta}}, \hat{\mathbf{u}}_j)$, and standardized residuals as

$$\hat{\boldsymbol{\epsilon}}_j^* = \hat{\sigma}^{-1}\widehat{\boldsymbol{\Lambda}}_j^{-1/2}\hat{\boldsymbol{\epsilon}}_j$$

If the `relevel(`*levelvar*`)` option is specified, predicted values, residuals, and standardized residuals consider only those random-effects terms up to and including level *levelvar* in the model. If the `fixedonly` option is specified, all statistics and named substitutable expressions are computed conditional on zero random effects; that is, their computation is based on the estimated fixed effects only.

Reference

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.

Also see

[ME] **menl** — Nonlinear mixed-effects regression

[U] 20 Estimation and postestimation commands