cmroprobit postestimation — Postestimation tools for cmroprobit

Postestimation commands predict margins estat
Remarks and examples Also see

Postestimation commands

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

Command	Description
`estat covariance`	covariance matrix of the utility errors for the alternatives
`estat correlation`	correlation matrix of the utility errors for the alternatives
`estat facweights`	covariance factor weights matrix

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
`hausman`	Hausman's specification test
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
`lrtest`	likelihood-ratio test
`margins`	adjusted predictions, predictive margins, and 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`	probabilities, 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 such as probabilities, linear predictions, and standard errors.

Menu for predict

Statistics > Postestimation

Syntax for predict

`predict` [*type*] *newvar* [*if*] [*in*] [, *statistic*]

`predict` [*type*] *stub*`*` [*if*] [*in*], `scores`

statistic	Description
Main	
`pr`	probability of each ranking, by case; the default
`pr1`	probability alternative is preferred
`xb`	linear prediction
`stdp`	standard error of the linear prediction

These statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample.

`predict` omits missing values casewise if `cmroprobit` used casewise deletion (the default); if `cmroprobit` used alternativewise deletion (option `altwise`), `predict` uses alternativewise deletion.

Options for predict

Main

`pr`, the default, calculates the probability of each ranking. For each case, one probability is computed for the ranks in `e(depvar)`.

`pr1` calculates the probability that each alternative is preferred.

`xb` calculates the linear prediction $\mathbf{x}_{ij}\boldsymbol{\beta} + \mathbf{z}_i\boldsymbol{\alpha}_j$ for alternative j and case i.

`stdp` calculates the standard error of the linear prediction.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new variable list of length equal to the number of columns in `e(b)`. Otherwise, use the *stub*`*` syntax to have `predict` generate enumerated variables with prefix *stub*.

margins

Description for margins

`margins` estimates margins of response for probabilities and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

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

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

statistic	Description
`pr`	not allowed with `margins`
`pr1`	probability alternative is preferred; the default
`xb`	linear prediction
`stdp`	not allowed with `margins`
`scores`	not allowed with `margins`

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

For more details, see [CM] **margins**.

estat

Description for estat

`estat covariance` computes the estimated variance–covariance matrix of the utility (latent-variable) errors for the alternatives. The estimates are displayed, and the variance–covariance matrix is stored in `r(cov)`.

`estat correlation` computes the estimated correlation matrix of the utility (latent-variable) errors for the alternatives. The estimates are displayed, and the correlation matrix is stored in `r(cor)`.

`estat facweights` displays the covariance factor weights matrix and stores it in `r(C)`.

Menu for estat

Statistics > Postestimation

Syntax for estat

Covariance matrix of the utility errors for the alternatives

`estat covariance [ , format(%fmt) border(bspec) left(#) ]`

Correlation matrix of the utility errors for the alternatives

`estat correlation [ , format(%fmt) border(bspec) left(#) ]`

Covariance factor weights matrix

`estat facweights [ , format(%fmt) border(bspec) left(#) ]`

`collect` is allowed with all `estat` commands; see **[U] 11.1.10 Prefix commands**.

Options for estat covariance, estat correlation, and estat facweights

`format(%fmt)` sets the matrix display format. The default for `estat covariance` and `estat facweights` is `format(%9.0g)`; the default for `estat correlation` is `format(%9.4f)`.

`border(bspec)` sets the matrix display border style. The default is `border(all)`. See [P] **matlist**.

`left(#)` sets the matrix display left indent. The default is `left(2)`. See [P] **matlist**.

Remarks and examples

After fitting a rank-ordered probit choice model, you can use `predict` to obtain the probabilities of the observed rankings of the alternatives or the probabilities of each alternative being preferred.

When evaluating the multivariate normal probabilities via Monte Carlo, `predict` uses the same method to generate the random sequence of numbers as the previous call to `cmroprobit`. For example, if you specified `intmethod(halton)` when fitting the model, `predict` also uses Halton sequences.

In example 1 of [CM] **cmroprobit**, we fit a model of job characteristic preferences. This is a study of Wisconsin high school graduates who were asked to rate their relative preference of four job characteristics: esteem, variety, autonomy, and security. The alternatives are ranked such that 1 is the preferred alternative and 4 is the least preferred.

The case-specific covariates are gender, `female`, an indicator variable for females, and `score`, a score on a general mental ability test measured in standard deviations. From the variables `high` and `low`, we create an alternative-specific variable, `currentjob`, that indicates whether the respondent's current job is high, low, or neither in esteem, variety, autonomy, or security.

We load the data and `cmset` them. For speed of running this example, we keep only untied rankings. Then, we fit our `cmroprobit` model.

```
. use https://www.stata-press.com/data/r19/wlsrank
(1992 Wisconsin Longitudinal Study data on job values)
. cmset id jobchar
     Case ID variable: id
Alternatives variable: jobchar
. keep if noties
(11,244 observations deleted)
. generate currentjob = 1 if low==1
(1,304 missing values generated)
. replace currentjob = 2 if low==0 & high==0
(805 real changes made)
. replace currentjob = 3 if high==1
(499 real changes made)
. label define current 1 "Low" 2 "Neither" 3 "High"
. label values currentjob current
. cmroprobit rank i.currentjob, casevars(i.female score) reverse
note: variable 2.currentjob has 69 cases that are not alternative-specific;
      there is no within-case variability.
note: variable 3.currentjob has 107 cases that are not alternative-specific;
      there is no within-case variability.
Iteration 0:  Log simulated-likelihood = -1102.9667
Iteration 1:  Log simulated-likelihood = -1089.1146  (backed up)
Iteration 2:  Log simulated-likelihood = -1085.7877  (backed up)
Iteration 3:  Log simulated-likelihood = -1083.0085  (backed up)
Iteration 4:  Log simulated-likelihood = -1082.5081  (backed up)
Iteration 5:  Log simulated-likelihood = -1082.1977  (backed up)
Iteration 6:  Log simulated-likelihood = -1082.1208  (backed up)
Iteration 7:  Log simulated-likelihood = -1082.0995
Iteration 8:  Log simulated-likelihood = -1082.0442
Iteration 9:  Log simulated-likelihood = -1081.8316  (backed up)
Iteration 10: Log simulated-likelihood = -1081.6816
Iteration 11: Log simulated-likelihood = -1081.5777
Iteration 12: Log simulated-likelihood = -1081.5137
Iteration 13: Log simulated-likelihood = -1081.1495
Iteration 14: Log simulated-likelihood = -1081.0503
```

```
Iteration 15: Log simulated-likelihood = -1080.7247
Iteration 16: Log simulated-likelihood = -1080.1485
Iteration 17: Log simulated-likelihood = -1080.0215
Iteration 18: Log simulated-likelihood = -1080.0179
Iteration 19: Log simulated-likelihood = -1079.9916
Iteration 20: Log simulated-likelihood = -1079.9733
Iteration 21: Log simulated-likelihood = -1079.9733
```

```
Rank-ordered probit choice model              Number of obs      =      1,660
Case ID variable: id                          Number of cases    =        415

Alternatives variable: jobchar                Alts per case: min =          4
                                                             avg =        4.0
                                                             max =          4

Integration sequence:      Hammersley
Integration points:               642         Wald chi2(8)       =      33.92
Log simulated-likelihood = -1079.9733         Prob > chi2        =     0.0000
```

rank	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
jobchar						
currentjob						
Neither	.0694818	.1092531	0.64	0.525	-.1446503	.2836138
High	.4435911	.121678	3.65	0.000	.2051066	.6820757
Esteem	(base alternative)					
Variety						
female						
Female	.1354259	.1843808	0.73	0.463	-.2259537	.4968055
score	.14071	.0977659	1.44	0.150	-.0509077	.3323278
_cons	1.734496	.1449852	11.96	0.000	1.45033	2.018661
Autonomy						
female						
Female	.2562822	.1645938	1.56	0.119	-.0663158	.5788801
score	.189963	.0873586	2.17	0.030	.0187433	.3611827
_cons	.7007559	.1203087	5.82	0.000	.4649553	.9365566
Security						
female						
Female	.2326342	.2055864	1.13	0.258	-.1703079	.6355762
score	-.1779831	.1101985	-1.62	0.106	-.3939682	.0380021
_cons	1.34348	.1598787	8.40	0.000	1.030123	1.656836
/lnl2_2	.1813086	.0756934	2.40	0.017	.0329522	.329665
/lnl3_3	.4843953	.0793885	6.10	0.000	.3287967	.639994
/l2_1	.6060303	.1158474	5.23	0.000	.3789735	.8330871
/l3_1	.4491585	.1435157	3.13	0.002	.167873	.7304441
/l3_2	.2305503	.121713	1.89	0.058	-.0080029	.4691034

```
(jobchar=Esteem is the alternative normalizing location)
(jobchar=Variety is the alternative normalizing scale)
```

We obtain the probabilities of the observed alternative rankings using `predict` with the `pr` option. The probabilities of each alternative being preferred is given by the `pr1` option.

```
. predict prob, pr
. predict prob1, pr1
. list id jobchar rank prob prob1 in 1/12, sepby(id)
```

	id	jobchar	rank	prob	prob1
1.	13	Esteem	4	.0424396	.0159974
2.	13	Variety	2	.0424396	.6024934
3.	13	Autonomy	1	.0424396	.1029332
4.	13	Security	3	.0424396	.278576
5.	19	Esteem	3	.0942127	.0140184
6.	19	Variety	2	.0942127	.4026075
7.	19	Autonomy	4	.0942127	.1232482
8.	19	Security	1	.0942127	.4601093
9.	22	Esteem	4	.1416861	.0255156
10.	22	Variety	1	.1416861	.455048
11.	22	Autonomy	2	.1416861	.2565435
12.	22	Security	3	.1416861	.2629159

The `prob` variable is constant for each case because it contains the probability of the observed ranking in the `rank` variable. The `prob1` variable contains the estimated probability of each alternative being preferred. The sum of the values in `prob1` will be approximately 1 for each case. They do not add up to exactly 1 because of approximations due to the GHK algorithm.

For examples of the specialized `estat` subcommands `covariance` and `correlation`, see [CM] **Intro 6** and [CM] **cmroprobit**.

Also see

[CM] **cmroprobit** — Rank-ordered probit choice model

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

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

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

[U] 20 Estimation and postestimation commands

Stata, Stata Press, Mata, NetCourse, and NetCourseNow are registered trademarks of StataCorp LLC. Stata and Stata Press are registered trademarks with the World Intellectual Property Organization of the United Nations. StataNow is a trademark of StataCorp LLC. Other brand and product names are registered trademarks or trademarks of their respective companies. Copyright © 1985–2025 StataCorp LLC, College Station, TX, USA. All rights reserved.

For suggested citations, see the FAQ on citing Stata documentation.