

**heckprobit postestimation** — Postestimation tools for heckprobit

[Description](#)      [Syntax for predict](#)      [Menu for predict](#)      [Options for predict](#)  
[Remarks and examples](#)      [Also see](#)

## Description

The following postestimation commands are available after `heckprobit`:

Command	Description
<code>contrast</code>	contrasts and ANOVA-style joint tests of estimates
<code>estat ic</code>	Akaike's and Schwarz's Bayesian information criteria (AIC and BIC)
<code>estat summarize</code>	summary statistics for the estimation sample
<code>estat vce</code>	variance–covariance matrix of the estimators (VCE)
<code>estat (svy)</code>	postestimation statistics for survey data
<code>estimates</code>	cataloging estimation results
<code>lincom</code>	point estimates, standard errors, testing, and inference for linear combinations of coefficients
<code>lrtest</code> <sup>1</sup>	likelihood-ratio test
<code>margins</code>	marginal means, predictive margins, marginal effects, and average marginal effects
<code>marginsplot</code>	graph the results from margins (profile plots, interaction plots, etc.)
<code>nlcom</code>	point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients
<code>predict</code>	predictions, residuals, influence statistics, and other diagnostic measures
<code>predictnl</code>	point estimates, standard errors, testing, and inference for generalized predictions
<code>pwcompare</code>	pairwise comparisons of estimates
<code>suest</code>	seemingly unrelated estimation
<code>test</code>	Wald tests of simple and composite linear hypotheses
<code>testnl</code>	Wald tests of nonlinear hypotheses

<sup>1</sup> `lrtest` is not appropriate with `svy` estimation results.

## Syntax for predict

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

```
predict [type] { stub* | newvarreg newvarsel newvarathrho } [if] [in] , scores
```

<i>statistic</i>	Description
Main	
<u>pmargin</u>	$\Phi(\mathbf{x}_j\mathbf{b})$ , success probability; the default
<u>p11</u>	$\Phi_2(\mathbf{x}_j\mathbf{b}, \mathbf{z}_j\mathbf{g}, \rho)$ , predicted probability $\Pr(y_j^{\text{probit}} = 1, y_j^{\text{select}} = 1)$
<u>p10</u>	$\Phi_2(\mathbf{x}_j\mathbf{b}, -\mathbf{z}_j\mathbf{g}, -\rho)$ , predicted probability $\Pr(y_j^{\text{probit}} = 1, y_j^{\text{select}} = 0)$
<u>p01</u>	$\Phi_2(-\mathbf{x}_j\mathbf{b}, \mathbf{z}_j\mathbf{g}, -\rho)$ , predicted probability $\Pr(y_j^{\text{probit}} = 0, y_j^{\text{select}} = 1)$
<u>p00</u>	$\Phi_2(-\mathbf{x}_j\mathbf{b}, -\mathbf{z}_j\mathbf{g}, \rho)$ , predicted probability $\Pr(y_j^{\text{probit}} = 0, y_j^{\text{select}} = 0)$
<u>pselect</u>	$\Phi(\mathbf{z}_j\mathbf{g})$ , selection probability
<u>pcond</u>	$\Phi_2(\mathbf{x}_j\mathbf{b}, \mathbf{z}_j\mathbf{g}, \rho) / \Phi(\mathbf{z}_j\mathbf{g})$ , probability of success conditional on selection
<u>xb</u>	linear prediction
<u>stdp</u>	standard error of the linear prediction
<u>xbselect</u>	linear prediction for selection equation
<u>stdpselect</u>	standard error of the linear prediction for selection equation

where  $\Phi(\cdot)$  is the standard normal distribution function and  $\Phi_2(\cdot)$  is the bivariate normal distribution function.

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

## Menu for predict

Statistics > Postestimation > Predictions, residuals, etc.

## Options for predict

Main

pmargin, the default, calculates the univariate (marginal) predicted probability of success  $\Pr(y_j^{\text{probit}} = 1)$ .

p11 calculates the bivariate predicted probability  $\Pr(y_j^{\text{probit}} = 1, y_j^{\text{select}} = 1)$ .

p10 calculates the bivariate predicted probability  $\Pr(y_j^{\text{probit}} = 1, y_j^{\text{select}} = 0)$ .

p01 calculates the bivariate predicted probability  $\Pr(y_j^{\text{probit}} = 0, y_j^{\text{select}} = 1)$ .

p00 calculates the bivariate predicted probability  $\Pr(y_j^{\text{probit}} = 0, y_j^{\text{select}} = 0)$ .

pselect calculates the univariate (marginal) predicted probability of selection  $\Pr(y_j^{\text{select}} = 1)$ .

pcond calculates the conditional (on selection) predicted probability of success  $\Pr(y_j^{\text{probit}} = 1, y_j^{\text{select}} = 1) / \Pr(y_j^{\text{select}} = 1)$ .

xb calculates the probit linear prediction  $\mathbf{x}_j\mathbf{b}$ .

`stdp` calculates the standard error of the prediction, which can be thought of as the standard error of the predicted expected value or mean for the observation's covariate pattern. The standard error of the prediction is also referred to as the standard error of the fitted value.

`xbssel` calculates the linear prediction for the selection equation.

`stdpssel` calculates the standard error of the linear prediction for the selection equation.

`nooffset` is relevant only if you specified `offset(varname)` for `heckprobit`. It modifies the calculations made by `predict` so that they ignore the offset variable; the linear prediction is treated as  $\mathbf{x}_j\mathbf{b}$  rather than as  $\mathbf{x}_j\mathbf{b} + \text{offset}_j$ .

`scores` calculates equation-level score variables.

The first new variable will contain  $\partial \ln L / \partial (\mathbf{x}_j\boldsymbol{\beta})$ .

The second new variable will contain  $\partial \ln L / \partial (\mathbf{z}_j\boldsymbol{\gamma})$ .

The third new variable will contain  $\partial \ln L / \partial (\text{atanh } \rho)$ .

## Remarks and examples

[stata.com](http://stata.com)

### ► Example 1

It is instructive to compare the marginal predicted probabilities with the predicted probabilities that we would obtain by ignoring the selection mechanism. To compare the two approaches, we will synthesize data so that we know the “true” predicted probabilities.

First, we need to generate correlated error terms, which we can do using a standard Cholesky decomposition approach. For our example, we will clear any data from memory and then generate errors that have a correlation of 0.5 by using the following commands. We set the seed so that interested readers can type in these same commands and obtain the same results.

```
. set seed 12309
. set obs 5000
obs was 0, now 5000
. gen c1 = rnormal()
. gen c2 = rnormal()
. matrix P = (1, .5 \ .5, 1)
. matrix A = cholesky(P)
. local fac1 = A[2,1]
. local fac2 = A[2,2]
. gen u1 = c1
. gen u2 = 'fac1'*c1 + 'fac2'*c2
```

We can check that the errors have the correct correlation by using the `correlate` command. We will also normalize the errors so that they have a standard deviation of one, so we can generate a bivariate probit model with known coefficients. We do that with the following commands:

#### 4 heckprobit postestimation — Postestimation tools for heckprobit

```

. correlate u1 u2
(obs=5000)

      |          u1          u2
-----+-----
      |          1.0000         |
      |          0.5020         | 1.0000
-----+-----

. summarize u1
(output omitted)

. replace u1 = u1/r(sd)
(5000 real changes made)

. summarize u2
(output omitted)

. replace u2 = u2/r(sd)
(5000 real changes made)

. drop c1 c2

. gen x1 = runiform()-.5
. gen x2 = runiform()+1/3
. gen y1s = 0.5 + 4*x1 + u1
. gen y2s = 3 - 3*x2 + .5*x1 + u2
. gen y1 = (y1s>0)
. gen y2 = (y2s>0)

```

We have now created two dependent variables,  $y_1$  and  $y_2$ , which are defined by our specified coefficients. We also included error terms for each equation, and the error terms are correlated. We run `heckprobit` to verify that the data have been correctly generated according to the model

$$y_1 = .5 + 4x_1 + u_1$$

$$y_2 = 3 + .5x_1 - 3x_2 + u_2$$

where we assume that  $y_1$  is observed only if  $y_2 = 1$ .

```

. heckprobit y1 x1, sel(y2 = x1 x2) nolog
Probit model with sample selection          Number of obs   =   5000
                                           Censored obs     =   1762
                                           Uncensored obs   =   3238
                                           Wald chi2(1)    =   953.71
                                           Prob > chi2     =   0.0000
Log likelihood =  -3679.5

```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
<b>y1</b>						
x1	3.784705	.1225532	30.88	0.000	3.544505	4.024905
_cons	.4630922	.0453952	10.20	0.000	.3741192	.5520653
<b>y2</b>						
x1	.3693052	.0721694	5.12	0.000	.2278558	.5107547
x2	-3.05069	.0832424	-36.65	0.000	-3.213842	-2.887538
_cons	3.037696	.0777733	39.06	0.000	2.885263	3.190128
/athrho	.5186232	.083546	6.21	0.000	.354876	.6823705
rho	.4766367	.0645658			.3406927	.5930583

```
LR test of indep. eqns. (rho = 0):  chi2(1) = 40.43  Prob > chi2 = 0.0000
```

Now that we have verified that we have generated data according to a known model, we can obtain and then compare predicted probabilities from the probit model with sample selection and a (usual) probit model.

```
. predict pmarg
(option pmargin assumed; Pr(y1=1))
. probit y1 x1 if y2==1
(output omitted)
. predict phat
(option pr assumed; Pr(y1))
```

Using the (marginal) predicted probabilities from the probit model with sample selection (`pmarg`) and the predicted probabilities from the (usual) probit model (`phat`), we can also generate the “true” predicted probabilities from the synthesized `y1s` variable and then compare the predicted probabilities:

```
. gen ptrue = normal(y1s)
. summarize pmarg ptrue phat
```

Variable	Obs	Mean	Std. Dev.	Min	Max
pmarg	5000	.6071226	.3147861	.0766334	.9907113
ptrue	5000	.5974195	.348396	5.53e-06	.9999999
phat	5000	.6568175	.3025085	.1059824	.9954919

Here we see that ignoring the selection mechanism (comparing the `phat` variable with the true `ptrue` variable) results in predicted probabilities that are much higher than the true values. Looking at the marginal predicted probabilities from the model with sample selection, however, results in more accurate predictions.

◀

## Also see

[R] [heckprobit](#) — Probit model with sample selection

[U] [20 Estimation and postestimation commands](#)