**biprobit — Bivariate probit regression**

**Description**

*biprobit* fits maximum-likelihood two-equation probit models—either a bivariate probit or a seemingly unrelated probit (limited to two equations).

**Quick start**

Bivariate probit regression of $y_1$ and $y_2$ on $x_1$

```
biprobit y1 y2 x1
```

Bivariate probit regression of $y_1$ and $y_2$ on $x_1$, $x_2$, and $x_3$

```
biprobit y1 y2 x1 x2 x3
```

Constrain the coefficients for $x_1$ to equality in both equations

```
constraint define 1 _b[y1:x1] = _b[y2:x1]
biprobit y1 y2 x1 x2 x3, constraints(1)
```

Seemingly unrelated bivariate probit regression

```
biprobit (y1 = x1 x2 x3) (y2 = x1 x2)
```

With robust standard errors

```
biprobit (y1 = x1 x2 x3) (y2 = x1 x2), vce(robust)
```

Poirier partial observability model with difficult option

```
biprobit (y1 = x1 x2) (y2 = x2 x3), partial difficult
```

**Menu**

**biprobit**

Statistics > Binary outcomes > Bivariate probit regression

**seemingly unrelated biprobit**

Statistics > Binary outcomes > Seemingly unrelated bivariate probit regression
Bivariate probit regression

Syntax

```
biprobit  depvar_1  depvar_2  [ indepvars ]  [ if ]  [ in ]  [ weight ] [,  options ]
```

Seemingly unrelated bivariate probit regression

```
biprobit  equation_1  equation_2  [ if ]  [ in ]  [ weight ] [ ,  su_options ]
```

where `equation_1` and `equation_2` are specified as

```
[  eqname:  ]  depvar  =  [ indepvars ]  [ ,  noconstant  offset(varname) ]
```

**options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model</strong></td>
</tr>
<tr>
<td>noconstant</td>
</tr>
<tr>
<td>partial</td>
</tr>
<tr>
<td>offset1(varname)</td>
</tr>
<tr>
<td>offset2(varname)</td>
</tr>
<tr>
<td>constraints(constraints)</td>
</tr>
<tr>
<td>collinear</td>
</tr>
<tr>
<td><strong>SE/Robust</strong></td>
</tr>
<tr>
<td>vce(vcetype)</td>
</tr>
<tr>
<td><strong>Reporting</strong></td>
</tr>
<tr>
<td>level(#)</td>
</tr>
<tr>
<td>noskip</td>
</tr>
<tr>
<td>nocnsreport</td>
</tr>
<tr>
<td>display_options</td>
</tr>
<tr>
<td><strong>Maximization</strong></td>
</tr>
<tr>
<td>maximize_options</td>
</tr>
<tr>
<td>coeflegend</td>
</tr>
</tbody>
</table>
su_options  | Description
-----------|---------
**Model**

*partial*  | fit partial observability model
*constraints*(constraints)  | apply specified linear constraints
*collinear*  | keep collinear variables

**SE/Robust**

*vce(vcetype)*  | vcetype may be oim, robust, cluster clustvar, opg, bootstrap, or jackknife

**Reporting**

*level(#)*  | set confidence level; default is level(95)
*noskip*  | perform likelihood-ratio test
*noconsreport*  | do not display constraints
*display_options*  | control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling

**Maximization**

*maximize_options*  | control the maximization process; seldom used
*coeflegend*  | display legend instead of statistics

---

**Options**

---

**Model**

*noconstant*; see [R] estimation options.

*partial* specifies that the partial observability model be fit. This particular model commonly has poor convergence properties, so we recommend that you use the *difficult* option if you want to fit the Poirier partial observability model; see [R] maximize.

This model computes the product of the two dependent variables so that you do not have to replace each with the product.

*offset1(varname), offset2(varname), constraints(constraints), collinear*; see [R] estimation options.

**SE/Robust**

*vce(vcetype)* specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim, opg), that are robust to some kinds of misspecification (robust), that allow for intragroup correlation (cluster clustvar), and that use bootstrap or jackknife methods (bootstrap, jackknife); see [R] vce_option.
Reporting

level(#); see [R] estimation options.

noskip specifies that a full maximum-likelihood model with only a constant for the regression equation be fit. This model is not displayed but is used as the base model to compute a likelihood-ratio test for the model test statistic displayed in the estimation header. By default, the overall model test statistic is an asymptotically equivalent Wald test of all the parameters in the regression equation being zero (except the constant). For many models, this option can substantially increase estimation time.

cocnsreport; see [R] estimation options.

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.

Maximization

maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs); see [R] maximize. These options are seldom used.

Setting the optimization type to technique(bhhh) resets the default vcetype to vce(opg).

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

coeflegend; see [R] estimation options.

Remarks and examples


Example 1

We use the data from Pindyck and Rubinfeld (1998, 332). In this dataset, the variables are whether children attend private school (private), number of years the family has been at the present residence (years), log of property tax (logptax), log of income (loginc), and whether the head of the household voted for an increase in property taxes (vote).

We wish to model the bivariate outcomes of whether children attend private school and whether the head of the household voted for an increase in property tax based on the other covariates.
The output shows several iteration logs. The first iteration log corresponds to running the univariate probit model for the first equation, and the second log corresponds to running the univariate probit for the second model. If $\rho = 0$, the sum of the log likelihoods from these two models will equal the log likelihood of the bivariate probit model; this sum is printed in the iteration log as the comparison log likelihood.

The final iteration log is for fitting the full bivariate probit model. A likelihood-ratio test of the log likelihood for this model and the comparison log likelihood is presented at the end of the output. If we had specified the `vce(robust)` option, this test would be presented as a Wald test instead of as a likelihood-ratio test.

We could have fit the same model by using the seemingly unrelated syntax as

```
. biprobit (private=years logptax loginc) (vote=years logptax loginc)
```
biprobit stores the following in e():

Scalars

- \texttt{e(N)}: number of observations
- \texttt{e(k)}: number of parameters
- \texttt{e(k_eq)}: number of equations in \texttt{e(b)}
- \texttt{e(k_aux)}: number of auxiliary parameters
- \texttt{e(k_eq_model)}: number of equations in overall model test
- \texttt{e(k_dv)}: number of dependent variables
- \texttt{e(ll)}: log likelihood
- \texttt{e(ll_0)}: log likelihood, constant-only model (\texttt{noskip} only)
- \texttt{e(ll_c)}: log likelihood, comparison model
- \texttt{e(N_clust)}: number of clusters
- \texttt{e(chi2)}: \(\chi^2\)
- \texttt{e(chi2_c)}: \(\chi^2\) for comparison test
- \texttt{e(p)}: significance
- \texttt{e(rho)}: \(\rho\)
- \texttt{e(rank)}: rank of \texttt{e(V)}
- \texttt{e(rank0)}: rank of \texttt{e(V)} for constant-only model
- \texttt{e(ic)}: number of iterations
- \texttt{e(rc)}: return code
- \texttt{e(converged)}: 1 if converged, 0 otherwise

Macros

- \texttt{e(cmd)}: \texttt{biprobit}
- \texttt{e(cmdline)}: command as typed
- \texttt{e(depvar)}: names of dependent variables
- \texttt{e(wtype)}: weight type
- \texttt{e(wexp)}: weight expression
- \texttt{e(title)}: title in estimation output
- \texttt{e(clustvar)}: name of cluster variable
- \texttt{e(offset1)}: offset for first equation
- \texttt{e(offset2)}: offset for second equation
- \texttt{e(chi2type)}: Wald or LR; type of model \(\chi^2\) test
- \texttt{e(chi2_ct)}: Wald or LR; type of model \(\chi^2\) test corresponding to \texttt{e(chi2_c)}
- \texttt{e(vce)}: \texttt{vcetype} specified in \texttt{vce()}
- \texttt{e(vcetype)}: title used to label Std. Err.
- \texttt{e(opt)}: type of optimization
- \texttt{e(which)}: \texttt{max} or \texttt{min}; whether optimizer is to perform maximization or minimization
- \texttt{e(ml_method)}: type of \texttt{ml} method
- \texttt{e(user)}: name of likelihood-evaluator program
- \texttt{e(technique)}: maximization technique
- \texttt{e(properties)}: \texttt{b V}
- \texttt{d(predict)}: program used to implement \texttt{predict}
- \texttt{e(asbalanced)}: factor variables \texttt{fvset} as \texttt{asbalanced}
- \texttt{e(asobserved)}: factor variables \texttt{fvset} as \texttt{asobserved}

Matrices

- \texttt{e(b)}: coefficient vector
- \texttt{e(Cns)}: constraints matrix
- \texttt{e(ilog)}: iteration log (up to 20 iterations)
- \texttt{e(gradient)}: gradient vector
- \texttt{e(V)}: variance–covariance matrix of the estimators
- \texttt{e(V_modelbased)}: model-based variance

Functions

- \texttt{e(sample)}: marks estimation sample
Methods and formulas

The log likelihood, $\ln L$, is given by

$$
\begin{align*}
\xi_j^\beta &= x_j \beta + \text{offset}_j^\beta \\
\xi_j^\gamma &= z_j \gamma + \text{offset}_j^\gamma \\
q_{1j} &= \begin{cases} 
1 & \text{if } y_{1j} \neq 0 \\
-1 & \text{otherwise}
\end{cases} \\
q_{2j} &= \begin{cases} 
1 & \text{if } y_{2j} \neq 0 \\
-1 & \text{otherwise}
\end{cases} \\
\rho_j^* &= q_{1j} q_{2j} \rho \\
\ln L &= \sum_{j=1}^n w_j \ln \Phi_2 \left( q_{1j} \xi_j^\beta, q_{2j} \xi_j^\gamma, \rho_j^* \right)
\end{align*}
$$

where $\Phi_2()$ is the cumulative bivariate normal distribution function (with mean $[0 \ 0]'$) and $w_j$ is an optional weight for observation $j$. This derivation assumes that

$$
\begin{align*}
y_{1j}^* &= x_j \beta + \epsilon_{1j} + \text{offset}_j^\beta \\
y_{2j}^* &= z_j \gamma + \epsilon_{2j} + \text{offset}_j^\gamma \\
E(\epsilon_1) &= E(\epsilon_2) = 0 \\
\text{Var}(\epsilon_1) &= \text{Var}(\epsilon_2) = 1 \\
\text{Cov}(\epsilon_1, \epsilon_2) &= \rho
\end{align*}
$$

where $y_{1j}^*$ and $y_{2j}^*$ are the unobserved latent variables; instead, we observe only $y_{ij} = 1$ if $y_{ij}^* > 0$ and $y_{ij} = 0$ otherwise (for $i = 1, 2$).

In the maximum likelihood estimation, $\rho$ is not directly estimated, but $\text{atanh} \rho$ is

$$
\text{atanh} \rho = \frac{1}{2} \ln \left( \frac{1 + \rho}{1 - \rho} \right)
$$

From the form of the likelihood, if $\rho = 0$, then the log likelihood for the bivariate probit models is equal to the sum of the log likelihoods of the two univariate probit models. A likelihood-ratio test may therefore be performed by comparing the likelihood of the full bivariate model with the sum of the log likelihoods for the univariate probit models.

This command supports the Huber/White/sandwich estimator of the variance and its clustered version using \texttt{vce(robust)} and \texttt{vce(cluster clustvar)}, respectively. See \texttt{[P] \_robust}, particularly \textit{Maximum likelihood estimators} and \textit{Methods and formulas}.

\texttt{biprobit} also supports estimation with survey data. For details on VCEs with survey data, see \texttt{[SVY] variance estimation}. 
References


Also see

[R] **biprobit postestimation** — Postestimation tools for biprobit

[R] **mprobit** — Multinomial probit regression

[R] **probit** — Probit regression

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

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