```
 _____
|  Sτaτa®    |
|_____|
Statistics/Data Analysis
```

help binstest
_____

Title

binstest —— Data-Driven Nonparametric Shape Restriction and Parametric Model
Specification Testing using Binscatter.

Syntax

> **binstest** _depvar_ _indvar_ [_covars_] [_if_] [_in_] [_weight_] [,
> **estmethod(**_cmdname_**)** **deriv(**_v_**)** **at(**_position_**)** **nolink**
> **absorb(**_absvars_**)** **reghdfeopt(**_reghdfe_option_**)**
> **testmodel(**_p s_**)** **testmodelparfit(**_filename_**)** **testmodelpoly(**_p_**)**
> **testshape(**_p s_**)** **testshapel(**_numlist_**)** **testshaper(**_numlist_**)**
> **testshape2(**_numlist_**)** **lp(**_metric_**)**
> **bins(**_p s_**)** **nbins(**#**)** **binspos(**_position_**)** **binsmethod(**_method_**)** **nbinsrot(**#**)**
> **randcut(**#**)**
> **nsims(**#**)** **simsgrid(**#**)** **simsseed(**_seed_**)**
> **dfcheck(**_n1 n2_**)** **masspoints(**_masspointsoption_**)**
> **vce(**_vcetype_**)** **asyvar(**_on/off_**)** **usegtools(**_on/off_**)**]

where _depvar_ is the dependent variable, _indvar_ is the independent variable for
   binning, and _covars_ are other covariates to be controlled for.

p, s and v are integers satisfying 0 <= s,v <= p, which can take different values
   in each case.

fweights, **aweight**s and **pweight**s are allowed; see _weight_.

Description

binstest implements binscatter-based hypothesis testing procedures for parametric
   functional forms and nonparametric shape restrictions on of the regression
   function estimators, following the results in Cattaneo, Crump, Farrell and
   Feng (2021a). If the binning scheme is not set by the user, the companion
   command binsregselect is used to implement binscatter in a data-driven
   (optimal) way and inference procedures are based on robust bias correction.
   Binned scatter plots based on different models can be constructed using the
   companion commands binsreg, _binsqreg, binslogit and binsprobit.

A detailed introduction to this command is given in Cattaneo, Crump, Farrell and
   Feng (2021b). A companion R package with the same capabilities is available
   (see website below).

Companion commands: binsreg for binscatter regression with robust inference
   procedures and plots, binsqreg for binscatter quantile regression with robust
   inference procedures and plots, binslogit for binscatter logit estimation with
   robust inference procedures and plots, binsprobit for binscatter probit
   estimation with robust inference procedures and plots, and binsregselect
   data-driven (optimal) binning selection.

Related Stata and R packages are available in the following website:

> https://nppackages.github.io/

Options

```
        ┌──────────┐
────────┤ Estimand ├──────────────────────────────────────────────────────
        └──────────┘
```

estmethod(_cmdname_**)** specifies the binscatter model. The default is **estmethod(reg)**,
   which corresponds to the binscatter least squares regression. Other options
   are: **estmethod(qreg #)** for binscatter quantile regression where # is the
   quantile to be estimated, **estmethod(logit)** for binscatter logistic regression
   and **estmethod(probit)** for binscatter probit regression.

deriv(*v***)** specifies the derivative order of the regression function for estimation, testing and plotting. The default is **deriv(0),** which corresponds to the function itself.

at(*position***)** specifies the values of *covars* at which the estimated function is evaluated for plotting. The default is **at(mean),** which corresponds to the mean of *covars*. Other options are: **at(median)** for the median of *covars*, **at(0)** for zeros, and **at(filename)** for particular values of *covars* saved in another file.

Note: when **at(mean)** or **at(median)** is specified, all factor variables in *covars* (if specified) are excluded from the evaluation.

nolink specifies that the function within the inverse link (logistic) function be reported instead of the conditional probability function. This option is used only if logit or probit model is specified in **estmethod().**

Reghdfe

absorb(*absvars***)** specifies categorical variables (or interactions) representing the fixed effects to be absorbed. This is equivalent to including an indicator/dummy variable for each category of each *absvar*. When **absorb()** is specified, the community-contributed command **reghdfe** instead of the command **regress** is used.

reghdfeopt(*reghdfe_option***)** options to be passed on to the command **reghdfe.** Important: **absorb()** and **vce()** should not be specified within this option.

For more information about the community-contributed command **reghdfe,** please see http://scorreia.com/software/reghdfe/.

Parametric Model Specification Testing

testmodel(*p s***)** sets a piecewise polynomial of degree *p* with *s* smoothness constraints for parametric model specification testing. The default is **testmodel(3 3),** which corresponds to a cubic B-spline estimate of the regression function of interest for testing against the fitting from a parametric model specification.

testmodelparfit(*filename***)** specifies a dataset which contains the evaluation grid and fitted values of the model(s) to be tested against. The file must have a variable with the same name as *indvar*, which contains a series of evaluation points at which the binscatter model and the parametric model of interest are compared with each other. Each parametric model is represented by a variable named as *binsreg_fit**, which must contain the fitted values at the corresponding evaluation points.

testmodelpoly(*p***)** specifies the degree of a global polynomial model to be tested against.

Nonparametric Shape Restriction Testing

testshape(*p s***)** sets a piecewise polynomial of degree *p* with *s* smoothness constraints for nonparametric shape restriction testing. The default is **testmodel(3 3),** which corresponds to a cubic B-spline estimate of the regression function of interest for one-sided or two-sided testing.

testshapel(*numlist***)** specifies a numlist of null boundary values for hypothesis testing. Each number *a* in the *numlist* corresponds to one boundary of a one-sided hypothesis test to the left of the form H0: *sup_x mu(x)<=a*.

testshaper(*numlist***)** specifies a numlist of null boundary values for hypothesis testing. Each number *a* in the *numlist* corresponds to one boundary of a one-sided hypothesis test to the right of the form H0: *inf_x mu(x)>=a*.

testshape2(*numlist***)** specifies a numlist of null boundary values for hypothesis testing. Each number *a* in the *numlist* corresponds to one boundary of a two-sided hypothesis test of the form H0: *sup_x |mu(x)-a|=0*.

Metric for Hypothesis Testing

lp(*metric***)** specifies a Lp metric used for (two-sided) parametric model
specification testing and/or shape restriction testing. The default is
lp(inf), which corresponds to the sup-norm. Other options are Lp(q) for a
positive integer q.

Partitioning/Binning Selection

bins(*p s***)** sets a piecewise polynomial of degree *p* with *s* smoothness constraints
for data-driven (IMSE-optimal) selection of the partitioning/binning scheme.
The default is **bins(0 0),** which corresponds to piecewise constant (canonical
binscatter).

nbins(*#***)** sets the number of bins for partitioning/binning of *indvar*. If not
specified, the number of bins is selected via the companion command
binsregselect in a data-driven, optimal way whenever possible.

binspos(*position***)** specifies the position of binning knots. The default is
binspos(qs), which corresponds to quantile-spaced binning (canonical
binscatter). Other options are: **es** for evenly-spaced binning, or a numlist
for manual specification of the positions of inner knots (which must be within
the range of *indvar*).

binsmethod(*method***)** specifies the method for data-driven selection of the number of
bins via the companion command binsregselect. The default is **binsmethod(dpi),**
which corresponds to the IMSE-optimal direct plug-in rule. The other option
is: **rot** for rule of thumb implementation.

nbinsrot(*#***)** specifies an initial number of bins value used to construct the DPI
number of bins selector. If not specified, the data-driven ROT selector is
used instead.

randcut(*#***)** specifies the upper bound on a uniformly distributed variable used to
draw a subsample for bins selection. Observations for which **runiform()<=#** are
used. # must be between 0 and 1.

Simulation

nsims(*#***)** specifies the number of random draws for constructing confidence bands
and hypothesis testing. The default is **nsims(500),** which corresponds to 500
draws from a standard Gaussian random vector of size [(p+1)*J − (J−1)*s].

simsgrid(*#***)** specifies the number of evaluation points of an evenly-spaced grid
within each bin used for evaluation of the supremum (or infimum) operation
needed to construct confidence bands and hypothesis testing procedures. The
default is **simsgrid(20),** which corresponds to 20 evenly-spaced evaluation
points within each bin for approximating the supremum (or infimum) operator.

simsseed(*#***)** sets the seed for simulations.

Mass Points and Degrees of Freedom

dfcheck(*n1 n2***)** sets cutoff values for minimum effective sample size checks, which
take into account the number of unique values of *indvar* (i.e., adjusting for
the number of mass points), number of clusters, and degrees of freedom of the
different statistical models considered. The default is **dfcheck(20 30).** See
Cattaneo, Crump, Farrell and Feng (2021b) for more details.

masspoints(*masspointsoption***)** specifies how mass points in *indvar* are handled. By
default, all mass point and degrees of freedom checks are implemented.
Available options:
masspoints(*noadjust***)** omits mass point checks and the corresponding effective
sample size adjustments.
masspoints(*nolocalcheck***)** omits within-bin mass point and degrees of freedom
checks.
masspoints(*off***)** sets **masspoints(***noadjust***)** and **masspoints(***nolocalcheck***)**
simultaneously.
masspoints(*veryfew***)** forces the command to proceed as if *indvar* has only a few
number of mass points (i.e., distinct values). In other words, forces the
command to proceed as if the mass point and degrees of freedom checks were
failed.

Other Options

---

vce(*vcetype***)** specifies the *vcetype* for variance estimation used by the commands
regress, logit or qreg. The default is **vce(robust)**.

asyvar(*on/off***)** specifies the method used to compute standard errors. If
asyvar(on) is specified, the standard error of the nonparametric component is
used and the uncertainty related to other control variables *covars* is omitted.
Default is **asyvar(off)**, that is, the uncertainty related to *covars* is taken
into account.

usegtools(*on/off***)** forces the use of several commands in the community-distributed
Stata package **gtools** to speed the computation up, if *on* is specified. Default
is **usegtools(off)**.

For more information about the package **gtools**, please see
https://gtools.readthedocs.io/en/latest/index.html.

Examples

Setup
        . sys use auto

Test for linearity
        . binstest mpg weight foreign, testmodelpoly(1)

Test for monotonicity
        . binstest mpg weight foreign, deriv(1) bins(1 1) testshapel(0)

Stored results

Scalars
  e(N)   number of observations
  e(Ndist)   number of distince values
  e(Nclust)   number of clusters
  e(nbins)   number of bins
  e(p)   degree of polynomial for bin selection
  e(s)   smoothness of polynomial for bin selection
  e(testshape_p) degree of polynomial for testing shape
  e(testshape_s) smoothnes of polynomial for testing shape
  e(testmodel_p) degree of polynomial for testing models
  e(testmodel_s) smoothness of polynomial for testing models
  e(testpolyp)   degree of polynomial regression model
  e(stat_poly)   statistic for testing global polynomial model
  e(pval_poly)   p value for testing global polynomial model
Locals
  e(testvalueL)   values in **testshapel()**
  e(testvalueR)   values in **testshaper()**
  e(testvalue2)   values in **testshape2()**
  e(testvarlist) varlist found in **testmodel()**
Matrices
  e(stat_shapeL) statistics for **testshapel()**
  e(pval_shapeL) p values for **testshapel()**
  e(stat_shapeR) statistics for **testshaper()**
  e(pval_shapeR) p values for **testshaper()**
  e(stat_shape2) statistics for **testshape2()**

e(pval_shape2)   p values for **testshape2()**
e(stat_model)   statistics for **testmodel()**
e(pval_model)   p values for **testmodel()**

References

Cattaneo, M. D., R. K. Crump, M. H. Farrell, and Y. Feng. 2021a. <u>On Binscatter</u>. *arXiv:1902.09608*.

Cattaneo, M. D., R. K. Crump, M. H. Farrell, and Y. Feng. 2021b. <u>Binscatter Regressions</u>. *arXiv:1902.09615*.

Authors

Matias D. Cattaneo, Princeton University, Princeton, NJ. <u>cattaneo@princeton.edu</u>.

Richard K. Crump, Federal Reserve Band of New York, New York, NY. <u>richard.crump@ny.frb.org</u>.

Max H. Farrell, University of Chicago, Chicago, IL. <u>max.farrell@chicagobooth.edu</u>.

Yingjie Feng, Princeton University, Princeton, NJ. <u>yingjief@princeton.edu</u>.