The Stata Journal (2022)
**22**, Number 2, pp. 430–445 DOI: 10.1177/1536867X221106418

# Binned scatterplots with marginal histograms: binscatterhist

Matteo Pinna
Center for Law and Economics
ETH Zürich
Zurich, Switzerland
matteo.pinna@gess.ethz.ch

**Abstract.** I introduce `binscatterhist`, a command that extends the functionality of the popular `binscatter` command (Stepner, 2013, Statistical Software Components S457709, Department of Economics, Boston College). `binscatter` allows researchers to summarize the relationship between two variables in an informative and versatile way by collapsing scattered points into bins. However, information about the variables' frequencies gets lost in the process. `binscatterhist` solves this issue by allowing the user to further enrich the graphs by plotting the variables' underlying distribution. The `binscatterhist` command includes options for different regression methods, including `reghdfe` (Correia, 2014, Statistical Software Components S457874, Department of Economics, Boston College) and `areg`, and robust and clustered standard errors, with automatic reporting of estimation results and sample size.

**Keywords:** gr0091, binscatterhist, binscatter, histogram, scatter, ggscatterhist, scatterhist, binned scatterplots, marginal histograms

## 1 Introduction

`binscatterhist` adds functionality to `binscatter` (Stepner 2013). Its main contribution is that it allows the user to include marginal histograms of each considered variable to the margins of the `binscatter` plot. Further improvements include additional regression methods and automatic reporting of estimation results and sample size. `binscatter` is a useful tool for examining the joint and marginal distributions of pairs of variables, and it is widely used in social sciences (for example, Ash et al. [2020] and Chetty, Friedman, and Saez [2013]). Scatterplots (`twoway scatter`) allow users to visualize the relationship between pairs of variables but can be difficult to interpret with large sample sizes. `binscatter` solves this issue by collapsing observations into bins and fitting a regression line. The cost of such compact representation is a loss of information on the plotted variables. `binscatterhist` accounts for this loss of information by allowing the user to plot the variables' underlying distribution. Similar commands exist for other statistical software, for example, `ggscatterhist` for R or `scatterhist` for MATLAB.

Figure 1 offers an example of a `binscatterhist` of `wage` and `tenure` from the web-available National Longitudinal Survey of Women 1988. It shows a positive correlation between wage and tenure, with a slope of 0.15 (0.02), significant at 1%, and a sample size of 2,228. The graph is produced by the following code:

```
. webuse nlsw88
(NLSW, 1988 extract)
. binscatterhist wage tenure, absorb(grade) vce(robust) coefficient(0.01)
> sample histogram(wage tenure) xmin(-2.2) ymin(5) xhistbarheight(15)
> yhistbarheight(15) xhistbins(40) yhistbins(40)
```

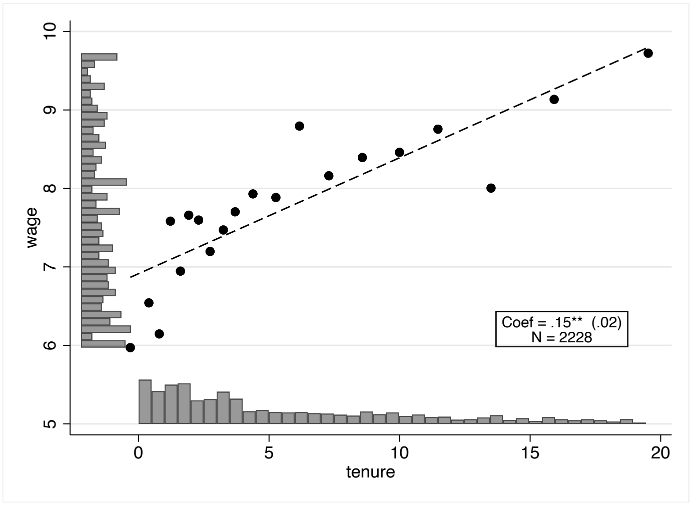

Figure 1. Example of a binscatterhist

The figure is generated including the regression options for fixed effects and robust standard errors (absorb(grade) and vce(robust), respectively). binscatterhist uses these options in the regression to generate the residuals that are then plotted after adding back each variable's sample mean and after the bins have been generated. A histogram for the regression variables is created by the option histogram(wage tenure), and the coefficient and sample size are included in the figure through the options coefficient(0.01) and sample. The additional options adjust the positioning and characteristics of the histogram.

By default, binscatterhist creates 20 equal-sized bins; thus, if the scattered points are closer to each other like in the left side of the plot, the underlying number of observations is higher. The horizontal distance between points, however, is hard to read promptly, especially when comparing points far apart or with strongly different $y$-axis heights. binscatterhist offers a fix for this, adding to the graph two histograms showing the variables' distribution. In figure 1, this more clearly shows tenure to be skewed and wage to be slightly skewed but with several high frequency bins.

## 2 The binscatterhist command

`binscatterhist` generates binned scatterplots via a nonparametric visualization of the relationship between two variables. `binscatterhist`'s essential features are the same as `binscatter`'s: the command "groups the $x$ axis variable into equal-sized bins, computes the mean of the $x$ axis and $y$ axis variables within each bin, and then creates a scatterplot of these data points" (Stepner 2013). The novel features and differences from the original command are discussed below.

### 2.1 Marginal histograms

The main new feature is the possibility to add to the graph the distribution of the plotted variables; these histograms are personalizable and allow for adjustments through the options of a `twoway bar` graph. When adding the option `histogram(`*varlist*`)`, a histogram is generated for the variables in *varlist*, and densities and bins are saved. These values are then rescaled to fit within the margins of the `scatterplot` and added to the `binscatter` with the community-contributed command `addplot` (Jann 2014), available through the Statistical Software Components Archive.

An important caveat in the interpretation of the histograms is that they are plotted on a different coordinates system than the `scatterplot`'s. For the $x$ variable on a `scatterplot`, the $x$ axis of its histogram corresponds to the $x$ axis of the `scatterplot`, and for the $y$ variable, the $x$ axis of its histogram corresponds to the $y$ axis of the `scatterplot`. Instead, the scale of the histograms' $y$ axes is a frequency and therefore not related to the variables in the `scatterplot`. The height of the histograms' bins can only be interpreted in relative terms to the other bins' heights within each histogram.

By default, the command sets the distributions in a slightly lower position than the lowest value of the scattered points, a number of bins equal to 20, a bar height equal to roughly 10% of the graph area, a bar width of 100%, meaning bars are close to each other, and a `xcolor(teal%50)` and `ycolor(maroon%50)`. See figures 3–4 for the default output (but shown in grayscale).

### 2.2 Residualization method

If the option `absorb()` is specified, `binscatterhist` by default runs a regression using the community-contributed command `reghdfe` (Correia 2014). Alternatively, regressions can be run using the Stata command `areg` by adding the option `regtype(areg)`. This improves the original command `binscatter`, which only allowed for regressions to be run with `areg`. Usability is expanded, because `reghdfe` is a generalization of `areg` for multiple levels of fixed effects and additional robust standard errors. Binning is performed after residualization when combined with the options `controls()` and `absorb()`. To do so, `binscatterhist` runs a regression on the control variables included in `controls()` and with the fixed effects included in `absorb()`. After generating the residuals, `binscatterhist` adds the sample mean of each variable back to its residuals. When including controls but no fixed effects, the residualization is performed via a standard regression.

## 2.3 Estimation results and sample size reporting

The options `coefficient(#)` and `sample` request that `binscatterhist` include a text box reporting the coefficient of the fitted line (approximated at #), its standard error, and the sample size. The text box adjusts its position based on the coefficient.

## 2.4 Clustered and robust standard errors

`binscatterhist` allows for robust or clustered standard errors. These enter the estimation in two points: First, clustered standard errors indirectly affect the residualization, because they might change the sample if the clusters' variable is missing for some observations; second, they are included in the estimation of the fit line so that the standard error and sample reported in the text box are adjusted accordingly.

## 3 The binscatterhist command

## 3.1 Syntax

The command syntax is

`binscatterhist` *varlist* [*if*] [*in*] [*weight*] [, *options*]

where *varlist* is

*y_1* [*y_2* [...]] *x*

`aweight`s and `fweight`s are allowed; see [U] **11.1.6 weight**.

## 3.2 Options

### 3.2.1 Main

`by(`*varname*`)` plots a separate series for by-value. Both numeric and string by-variables are supported, but numeric by-variables will have faster run times.

There are two ways in which `binscatterhist` does *not* condition on by-values:

1. When combined with `controls()` or `absorb()`, the command residualizes using the restricted model in which each covariate has the same coefficient in each by-value sample. It does not run separate regressions for each by-value. If you wish to control for covariates by using a different model, you can residualize your $x$ and $y$ variables beforehand with your desired model and then run `binscatterhist` on the residuals you constructed.

2. When not combined with `discrete` or `xq()`, the command constructs a single set of bins using the unconditional quantiles of the $x$ variable. It does not bin the $x$ variable separately for each by-value. If you wish to use a different binning procedure (such as constructing equal-sized bins separately for each by-value), you can construct a variable containing your desired bins beforehand and then run `binscatterhist` with `xq()`.

`medians` creates the binned scatterplot using the median $x$ and $y$ values within each bin rather than the mean. This option only affects the scatter points; it does not, for instance, cause `linetype(lfit)` to use quantile regression instead of ordinary least squares when drawing a fit line.

### 3.2.2 Bins

`nquantiles(#)` specifies the number of equal-sized bins to be created. This is equivalent to the number of points in each series. The default is `nquantiles(20)`. If the $x$ variable has fewer unique values than the number of bins specified, then `discrete` will be automatically invoked and no binning will be performed. `nquantiles()` may not be combined with `discrete` or `xq()`.

Binning is performed after residualization when combined with `controls()` or `absorb()`. Note that the binning procedure is equivalent to running Stata's `xtile` command, which in certain cases will generate fewer quantile categories than specified. (For example, `sysuse auto`; `xtile temp = mpg, nq(20)`; and `tab temp`.)

`genxq(`*varname*`)` creates a categorical variable containing the computed bins. `genxq()` may not be combined with `discrete` or `xq()`.

`discrete` specifies that the $x$ variable is discrete and that each $x$ value be treated as a separate bin. `binscatterhist` will therefore plot the mean $y$ value associated with each $x$ value. `discrete` may not be combined with `nquantiles()`, `genxq()`, or `xq()`.

In most cases, `discrete` should not be combined with `controls()` or `absorb()`, because residualization occurs before binning and, in general, the residual of a discrete variable will not be discrete.

`xq(`*varname*`)` specifies a categorical variable that contains the bins to be used instead of having `binscatterhist` generate them. This option is typically used to avoid recomputing the bins needlessly when `binscatterhist` is being run repeatedly on the same sample and with the same $x$ variable; it may be convenient to use `genxq()` in the first iteration and specify `xq()` in subsequent iterations. Computing quantiles is computationally intensive in large datasets, so avoiding repetition can reduce run times considerably. `xq()` may not be combined with `nquantiles()`, `genxq()`, or `discrete`.

Take care when combining `xq()` with `controls()` or `absorb()`. Binning takes place after residualization, so if the sample or the control variables change, the bins should be recomputed as well.

### 3.2.3 Residuals computation

`regtype(`*string*`)` specifies the type of regression to use to compute the residuals. *string* may be `reghdfe` or `areg`. `regtype()` requires `absorb()` to be specified. When `reghdfe` is specified, `absorb()` allows for more than one *varname*; however, interactions are not allowed, including tricks like, for example, `one##control`, to include controls in the absorb. Such controls must be included in the `controls()` option. `reghdfe` drops singleton observations with regard to the included fixed effects; therefore, sample size might differ between `reghdfe` and `areg`. The default is `regtype(reghdfe)`.

### 3.2.4 Standard errors/robust

`cluster(`*varname*`)` specifies the variable that identifies clusters. Clustered standard errors affect both the sample for residualization and the computation of standard errors for slope reporting.

`vce(robust)` specifies to calculate robust standard errors, which affect both the sample for residualization and the computation of standard errors for slope reporting.

### 3.2.5 Controls

`controls(`*varlist*`)` residualizes the $x$ and $y$ variables on the specified controls before binning and plotting. To do so, `binscatterhist` runs a regression of each variable on the controls, generates the residuals, and adds the sample mean of each variable back to its residuals.

`absorb(`*varlist*`)` absorbs fixed effects in the categorical variable from the $x$ and $y$ variables before binning and plotting. To do so, `binscatterhist` runs an `areg` of each variable with `absorb()` and any `controls()` specified. It then generates the residuals and adds the sample mean of each variable back to its residuals.

`noaddmean` prevents the sample mean of each variable from being added back to its residuals when combined with `controls()` or `absorb()`.

### 3.2.6 Fit line

`linetype(`*string*`)` specifies the type of line plotted on each series. The default is `linetype(lfit)`, which plots a linear fit line. Other options are `linetype(qfit)` for a quadratic fit line, `linetype(connect)` for connected points, and `linetype(none)` for no line.

Linear or quadratic fit lines are estimated using the underlying data, not the binned scatter points. When combined with `controls()` or `absorb()`, the fit line is estimated after the variables have been residualized.

`rd(`*numlist*`)` draws a dashed vertical line at the specified $x$ values and generates regression discontinuities when combined with `linetype(lfit)` or `linetype(qfit)`. Separate fit lines will be estimated below and above each discontinuity. These estimations are performed using the underlying data, not the binned scatter points.

The regression discontinuities do not affect the binned scatter points in any way. Specifically, a bin may contain a discontinuity within its range and, therefore, may include data from both sides of the discontinuity.

`reportreg` displays in the Results window the regressions used to estimate the fit lines.

### 3.2.7 Coefficient and sample reporting

`coefficient(`#`)` reports the slope of the fitted line with its standard error, rounded at # using `round(coefficient,` #`)`. See `help round()`.

`ci(`#`)` reports the #% confidence interval, rounded as the `coefficient()`.

`pvalue` reports the $p$-value of the regression on residualized variables.

`sample` reports the sample size of the regression on residualized variables.

`stars(`*string*`)` reports the $p$-value using stars. *string* may be `nostars`, `1` (*5% **1%), `2` (+10% *5% **1%), `3` (+10% *5% **1% ***0.1%), or `4` (*5% **1% ***0.1%). The default is `stars(1)`.

### 3.2.8 Graph style

`colors(`*colorstyle*`)` specifies an ordered list of colors for each series.

`mcolors(`*colorstyle*`)` specifies an ordered list of colors for the markers of each series, which overrides any list provided in `colors()`.

`lcolors(`*colorstyle*`)` specifies an ordered list of colors for the lines of each series, which overrides any list provided in `colors()`.

`msymbols(`*symbolstyle*`)` specifies an ordered list of symbols for each series.

*twoway_options* controls the graph titles, legends, axes, added lines and text, regions, name, aspect ratio, etc.; see [G-3] ***twoway_options***.

### 3.2.9 Histogram

`histogram(`*varlist*`)` plots a histogram for each of the selected variables (max = 2). Selected variables have to be the scattered ones.

`xmin(`*value*`)` sets the base position of the $y$ histogram in terms of the $x$ axis. Option `xmin()` is only allowed with `histogram()`.

`ymin(`*value*`)` sets the base position of the $x$ histogram in terms of the $y$ axis. Option `ymin()` is only allowed with `histogram()`.

`xhistbarheight(`*value*`)` sets the height of the $x$ histogram as a percentage. The default is `xhistbarheight(10)`.

`yhistbarheight(`*value*`)` sets the height of the $y$ histogram as a percentage. The default is `yhistbarheight(10)`.

`xhistbarwidth(`*value*`)` sets the width of the $x$ histogram as a percentage. The default is `xhistbarwidth(100)`.

`yhistbarwidth(`*value*`)` sets the width of the $y$ histogram as a percentage. The default is `yhistbarwidth(100)`.

`xhistbins(`*#*`)` sets the number of bins to be created in the $x$ histogram. The default is `xhistbins(20)`.

`yhistbins(`*#*`)` sets the number of bins to be created in the $y$ histogram. The default is `yhistbins(20)`.

The following options require the *axis* to be specified as `x` or `y`, for example, `xcolor()` or `ylpattern()`.

*axis*`color(`*colorstyle*`)` sets the outline and fill color and opacity. The defaults are `xcolor(teal%50)` and `ycolor(maroon%50)`.

*axis*`fcolor(`*colorstyle*`)` sets the fill color and opacity.

*axis*`fintensity(`*intensitystyle*`)` sets the fill intensity.

*axis*`lcolor(`*colorstyle*`)` sets the outline color and opacity.

*axis*`lwidth(`*linewidthstyle*`)` sets the thickness of the outline.

*axis*`lpattern(`*linepatternstyle*`)` sets the outline pattern (solid, dashed, etc.).

*axis*`lalign(`*linealignmentstyle*`)` sets the outline alignment (inside, outside, or center).

*axis*`lstyle(`*linestyle*`)` sets the overall look of the outline.

*axis*`bstyle(`*areastyle*`)` sets the overall look of the bars, all settings above.

*axis*`pstyle(`*pstyle*`)` sets the overall plot style, including area style.

### 3.2.10 Save output

`savegraph(`*filename*`)` saves the graph to a file. The format is automatically detected from the extension (for example, `.gph`, `.jpg`, or `.png`), and either `graph save` or `graph export` is run. By default, `.gph` is assumed.

`savedata(`*filename*`)` saves *filename*`.csv` containing scatter point data and *filename*`.do` to process the data into a graph.

`replace` specifies that files be overwritten if they already exist.

### 3.2.11 fastxtile options

`nofastxtile` forces the use of `xtile` instead of `fastxtile` to compute bins. There is no situation where this should be necessary or useful. The `fastxtile` command generates identical results to `xtile` but runs faster on large datasets and has additional options for random sampling that may be useful to increase speed.

`fastxtile` is built into the `binscatterhist` code but may also be installed separately for use outside of `binscatterhist`. It is available from the Statistical Software Components Archive.

`randvar(`*varname*`)` requests that *varname* be used to select a sample of observations when computing the quantile boundaries. Sampling increases the speed of the binning procedure but generates bins that, because of sampling error, are only approximately equal sized. It is possible to omit this option and still perform random sampling from $U[0,1]$, as described below in `randcut()` and `randn()`.

`randcut(`#`)` specifies the upper bound on the variable contained in `randvar()`. Quantile boundaries are approximated using observations for which `randvar()` $\leq \#$. If no variable is specified in `randvar()`, a standard uniform random variable is generated. The default is `randcut(1)`. `randcut()` may not be combined with `randn()`.

`randn(`#`)` specifies an approximate number of observations to sample when computing the quantile boundaries. Quantile boundaries are approximated using observations for which a uniform random variable is $\leq \#/N$. The exact number of observations sampled may therefore differ from #, but it equals # in expectation. When this option is combined with `randvar()`, *varname* should be distributed $U[0,1]$. Otherwise, a standard uniform random variable is generated. `randn()` may not be combined with `randcut()`.

## 4 Examples

The following figures use the 1988 National Longitudinal Survey of Women to show some basic examples of the new features of `binscatterhist`.

Figure 2 shows a binned scatterplot of two variables from the survey, `wage` and `tenure`, generated by the code

```
. binscatterhist wage tenure
```

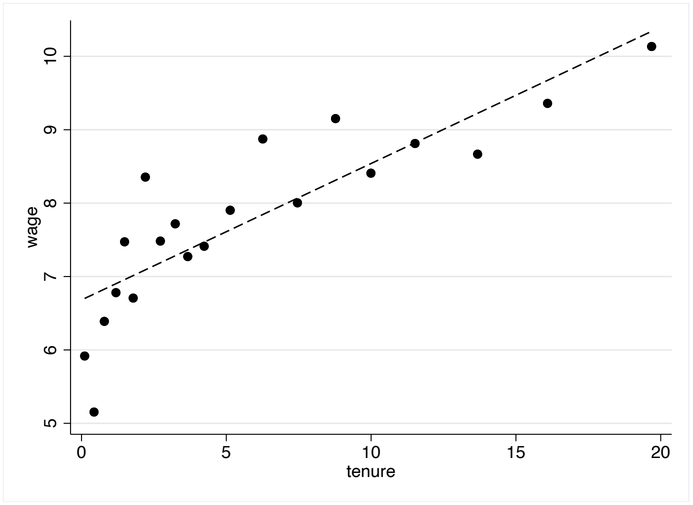

Figure 2. Simple binned scatterplot with a fitted line

Together with the fitted line, the binned scatterplot provides a graphical visualization of the positive correlation between the outcome variable `wage` and the regressor `tenure`. With no options specified in the command, `binscatterhist` produces a similar output to `binscatter`. Differences between the outputs of the two packages could arise based on the default use of `reghdfe` by `binscatterhist` compared with `areg` by `binscatter`.

The following code adds the `histogram()` option. `binscatterhist` thus produces a histogram of the requested variable, as shown in figure 3.

```
. binscatterhist wage tenure, histogram(tenure)
```

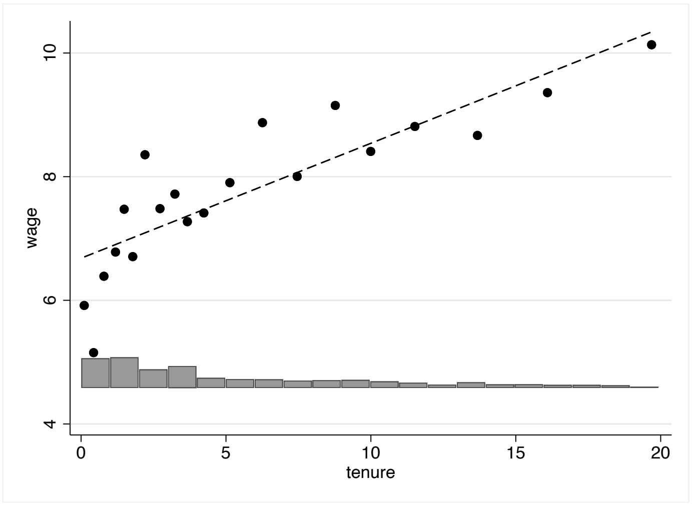

Figure 3. Adding a marginal histogram for one variable

By including the option `histogram(tenure)`, the distribution of the variable `tenure` is plotted above the $x$ axis and appears to be right-skewed. The position of the histogram can be adjusted, as shown in figure 4.

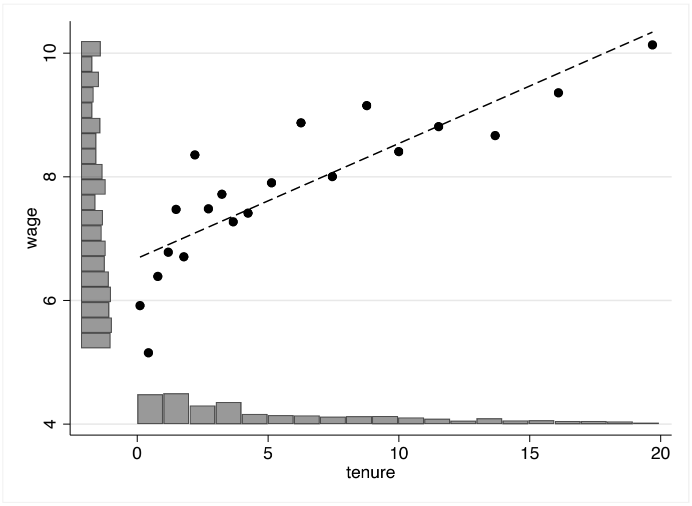

Figure 4. Adding both marginal histograms and fixing their position

In figure 4, the `histogram()` option is included for both variables. Adding the `ymin()` option fixes the misplacement of the $x$ histogram, telling `binscatterhist` to reset the base line for the histogram in terms of the $y$ axis, in this case, starting above `wage` $= 4$.

```
. binscatterhist wage tenure, histogram(wage tenure) ymin(4)
```

The code below uses the *axis*`bstyle(outline)` option and sets a bigger offset width for the histogram bars through the option `xhistbarwidth(50)`.

```
. binscatterhist wage tenure, histogram(wage tenure) ymin(4) yhistbarwidth(50)
> xhistbarwidth(50) ybstyle(outline) xbstyle(outline)
```

With the simple black and white layout, figure 5 shows a minimalist visualization of the histograms.

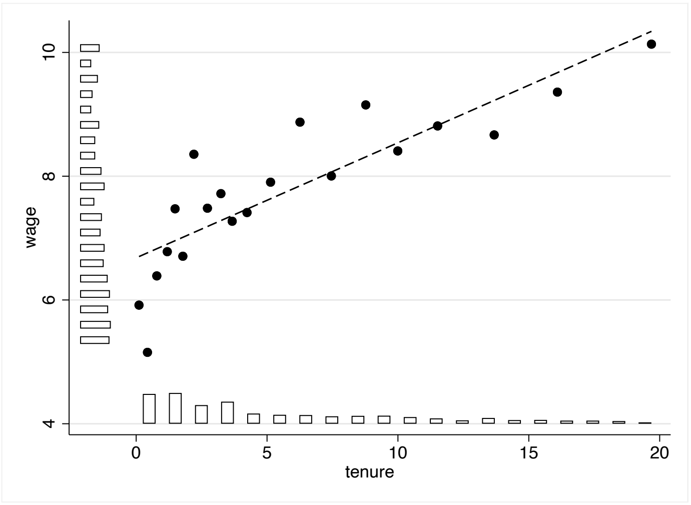

Figure 5. Black and white visualization

Furthermore, `binscatterhist` allows adjustment of the height and number of bins via the options *axis*`histbarheight()` and *axis*`histbins()`. In figure 6, a greater bar height and a higher number of bins are set; these may offer a more accurate or easy-to-read representation of the distribution of the variables.

```
. binscatterhist wage tenure, histogram(wage tenure) ymin(4) xhistbarheight(15)
> yhistbarheight(15) xhistbins(40) yhistbins(40)
```

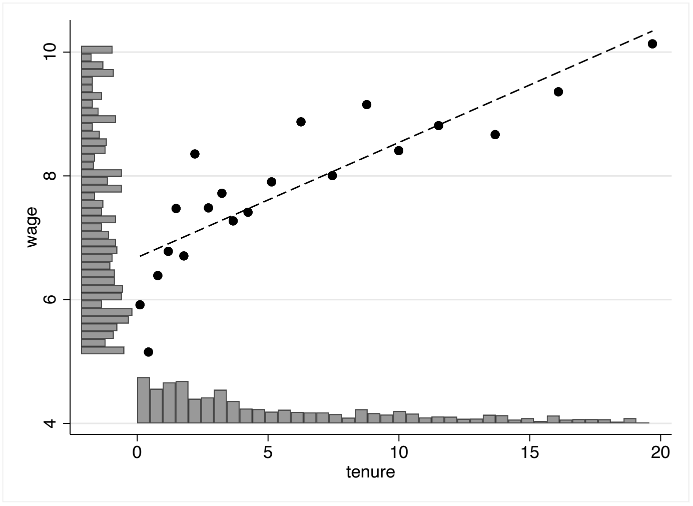

Figure 6. Adjusting the size and dimension of the bins

When the `absorb()` option for fixed effects is included, `binscatterhist` runs a regression using `reghdfe` to perform the prebinning residualization. In figure 7, the regression uses grade-level fixed effects; furthermore, the standard errors of the coefficient estimated by that regression are reported in the text box and are robust to heteroskedasticity (option `vce(robust)`). The options `coefficient(#)` and `sample` generate a text box reporting such information.

```
. binscatterhist wage tenure, absorb(grade) vce(robust) coefficient(0.01) sample
> xmin(-2.2) ymin(5) histogram(wage tenure) xhistbarheight(15)
> yhistbarheight(15) xhistbins(40) yhistbins(40)
```

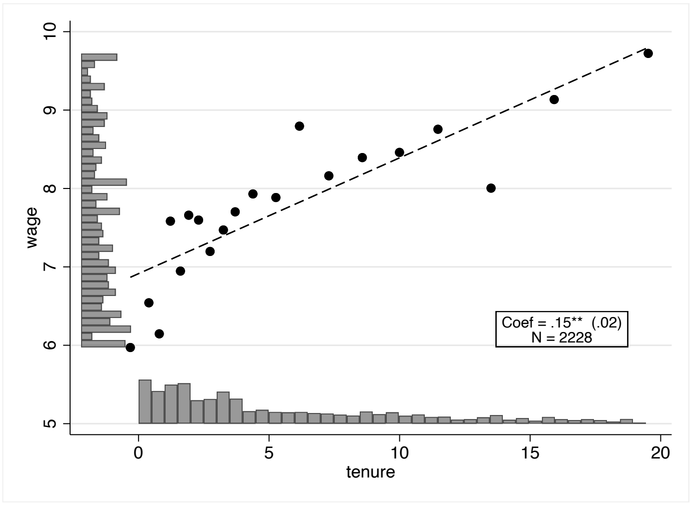

Figure 7. Reporting coefficient and sample; residualization through `reghdfe`

In figure 8, the slope of the fit line is negative. The text box automatically adjusts its positioning, while the `xmin()` option has been reset for the histogram to match the new graph composition. The option `regtype(areg)` runs the command `areg` instead of the command `reghdfe` to create the residuals. Finally, the options `ci()` and `pvalue` add the confidence interval and *p*-value to the coefficient text box.

```
. replace tenure=-abs(tenure)
(2,180 real changes made)
. binscatterhist wage tenure, regtype(areg) absorb(grade) vce(robust)
> coefficient(0.01) ci(95) pvalue sample xmin(-22) ymin(5)
> histogram(wage tenure) xhistbarheight(15) yhistbarheight(15)
> xhistbins(40) yhistbins(40)
```

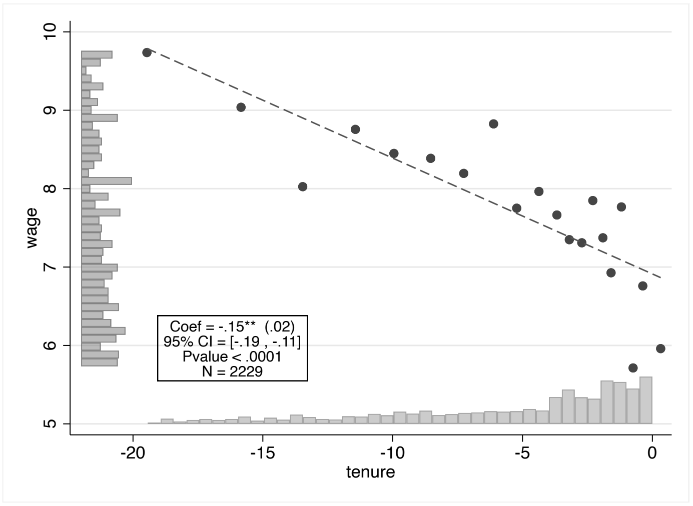

Figure 8. Other text box options; residualization through `areg`

## 5 Conclusions

In this article, I introduced the `binscatterhist` command for binned scatterplots with marginal histograms in Stata. The command provides functionality in addition to that provided by `scatterplot` and `binscatter`. `binscatterhist` helps users to interpret plots with large sample sizes without losing as much information about the distributions of the variables. The additional regression commands and user-friendly options provide a better graphical representation of regression results in a convenient way.

## 6 Acknowledgments

The author would like to thank Elliott Ash, Christopher Baum, Sergio Galletta, Malka Guillot, Suresh Naidu, and an anonymous referee for useful feedback on the command.

## 7 Programs and supplemental materials

To install a snapshot of the corresponding software files as they existed at the time of publication of this article, type

```
. net sj 22-2
. net install gr0091      (to install program files, if available)
. net get gr0091          (to install ancillary files, if available)
```

## 8 References

Ash, E., S. Galletta, D. Hangartner, Y. Margalit, and M. Pinna. 2020. The effect of Fox News on health behavior during COVID-19. SocArXiv abqe5, Center for Open Science. https://dx.doi.org/10.2139/ssrn.3636762.

Chetty, R., J. N. Friedman, and E. Saez. 2013. Using differences in knowledge across neighborhoods to uncover the impacts of the EITC on earnings. *American Economic Review* 103: 2683–2721. http://doi.org/10.1257/aer.103.7.2683.

Correia, S. 2014. reghdfe: Stata module to perform linear or instrumental-variable regression absorbing any number of high-dimensional fixed effects. Statistical Software Components S457874, Department of Economics, Boston College. https://ideas.repec.org/c/boc/bocode/s457874.html.

Jann, B. 2014. addplot: Stata module to add twoway plot objects to an existing twoway graph. Statistical Software Components S457917, Department of Economics, Boston College. https://ideas.repec.org/c/boc/bocode/s457917.html.

Stepner, M. 2013. binscatter: Stata module to generate binned scatterplots. Statistical Software Components S457709, Department of Economics, Boston College. https://ideas.repec.org/c/boc/bocode/s457709.html.

### About the author

Matteo Pinna is a doctoral candidate at the Center for Law and Economics of ETH Zurich. His research focuses on political economy, looking at the behavioral effects of information media. His practical experience as a research assistant and PhD have pushed him to write or improve preexisting community-contributed commands, such as **binscatterhist**, **eventcoefplot**, **multicoefplot**, and **inlist2**. His recent working articles have appeared on media outlets such as *The Washington Post*, *The Atlantic*, *MSNBC*, and *Vox*. A full list of research articles and packages can be found on the author's website, https://sites.google.com/view/matteopinna/.