

dstat: A new command for the analysis of distributions

Ben Jann

University of Bern

2021 Stata Conference
Virtual, August 5–6, 2021

Outline

- 1 Introduction
- 2 Theory
- 3 The new command
- 4 Examples
- 5 Conclusions

Why this command?

- Stata provides great functionality for advanced statistics and econometric analyses.
- However, Stata sometimes appears a bit weak in terms of descriptive statistics.
- What do I mean by descriptive statistics?
 - ▶ Statistics describing points in a distribution; series of such statistics illustrate the shape of a distribution.
 - ★ Densities, cumulative distributions, histograms, probabilities, quantiles, lorenz ordinates, etc.
 - ▶ Summary statistics describing particular features of a distribution.
 - ★ Various types of location, dispersion, skewness, or kurtosis measures.
 - ★ Inequality, concentration, and poverty measures.
 - ★ Association measures
 - ★ ...?
 - ▶ I primarily mean univariate statistics (although some of the measures covered by `dstat` are bivariate and `dstat` has some other features to support “multivariate” analyses).

What do I mean by “weak”?

- With “weak” I do not mean that descriptive statistics cannot be computed in Stata. I just mean that things could be a bit improved in terms of functionality and convenience.
 - ▶ Different types of statistics are scattered across various commands.
 - ▶ Each command has its own logic (idiosyncratic syntax, idiosyncratic output, idiosyncratic returns).
 - ▶ Support for statistical inference greatly varies (some commands do not even allow weights, others fully support complex survey estimation).
 - ▶ Often difficult to create tables and graphs without too much effort (particularly if interested in confidence intervals).
 - ▶ Some topics such as, e.g., inequality measures are not covered at all in official Stata. In general, there is a large number of user add-ons for descriptive statistics which, again, all have their own idiosyncrasies, and also greatly vary in terms of functionality and quality.

Guiding principles for the new command

- Descriptive statistics are estimates and should be treated as such.
 - ▶ Provide standard errors/confidence intervals for everything.
 - ▶ Standardized output like any other estimation command.
 - ▶ Return results in `e()` like any other estimation command.
- Highly standardized and consistent, but as flexible as possible.
- Graphs are important in descriptive analysis; provide convenient graphing functionality.
- Framework should be easy to extend (integrate further statistics without too much effort).
- Personal interest: Provide support for covariate balancing a.k.a. compositional standardization (\rightarrow treatment effect estimation, counterfactual decompositions).

- 1 Introduction
- 2 Theory**
- 3 The new command
- 4 Examples
- 5 Conclusions

Moment conditions and influence functions

- Think of statistic θ as a functional of a distribution F , that is $\theta = T(F)$.
- The influence function of θ is defined as the limit of the change in θ if a small amount of data mass is added at a specific point in the distribution:

$$IF(x, \theta, F) = \lim_{\epsilon \rightarrow 0} \frac{T((1 - \epsilon)F + \epsilon\delta_x) - T(F)}{\epsilon}$$

where δ_x is a distribution with all its mass at point x .

- Influence functions have been developed in robust statistics (e.g. Hampel 1974) to study the robustness properties of estimators.
- However, influence functions are super useful because the sampling variance of an estimator is equal to the sampling variance of the expected value of its influence function (e.g. Deville 1999)
- This means that the standard error of a mean estimate of the “empirical” influence function provides a consistent estimate of the standard error of $\hat{\theta}$.

Moment conditions and influence functions

- But how to compute influence functions in practice?
- Let h_i^θ be the moment condition of θ such that

$$\frac{1}{W} \sum_{i=1}^n w_i h_i^{\hat{\theta}} = 0$$

where w_i are sampling weights and W is the sum of weights.

- For example, the moment condition is as follows:

$$\bar{y} = \frac{1}{W} \sum_{i=1}^N w_i Y_i \Rightarrow \frac{1}{W} \sum_{i=1}^N w_i (Y_i - \bar{y}) = 0 \Rightarrow h_i^{\bar{y}} = Y_i - \bar{y}$$

- The “empirical” influence function can then be obtained as

$$IF_i(\hat{\theta}) = \frac{1}{G} h_i^{\hat{\theta}} \quad \text{with} \quad G = -\frac{1}{W} \sum_{i=1}^n w_i \left. \frac{\partial h_i^\theta}{\partial \theta} \right|_{\theta=\hat{\theta}}$$

- In case of the mean, G boils down to 1, such that the influence function simply is $IF_i(\bar{y}) = Y_i - \bar{y}$.

Moment conditions and influence functions

- Many statistics are constructed in a way such that they depend on a number of auxiliary estimates. For example, the trimmed mean depends on the quantiles at which the data is trimmed.
- Influence functions for such statistics can be derived using the chain rule.
- Let θ depend on additional parameters $\gamma_1, \dots, \gamma_k$. The influence function of θ can then be obtained as

$$\text{IF}_i(\hat{\theta}) = \frac{1}{G} \left(h_i^{\hat{\theta}} - \sum_{j=1}^k G_j \text{IF}_i(\hat{\gamma}_j) \right) \quad \text{with} \quad G_j = -\frac{1}{W} \sum_{i=1}^n w_i \left. \frac{\partial h_i^{\hat{\theta}}}{\partial \gamma_j} \right|_{\gamma_j = \hat{\gamma}_j}$$

- If γ_j itself depends on further parameters, its influence function will have a similar form. In this way we can easily piece together influence functions also for very complex statistics.

Moment conditions and influence functions

- More generally (see Jann 2020), if θ is a vector of estimates $\theta_1, \dots, \theta_k$ that may or may not depend on each other, the (k dimensional) influence function of θ can be obtained as

$$\text{IF}_i(\hat{\theta}) = \mathbf{G}^{-1} \mathbf{h}_i^{\hat{\theta}}$$

where

$$\mathbf{h}_i^{\theta} = \begin{bmatrix} h_i^{\theta_1} \\ \vdots \\ h_i^{\theta_k} \end{bmatrix} \quad \text{and} \quad \mathbf{G} = -\frac{1}{W} \sum_{i=1}^n w_i \left. \frac{\partial \mathbf{h}_i^{\theta}}{\partial \theta'} \right|_{\theta=\hat{\theta}}$$

- As David Drukker would say:

“Stack the moment equations!”

Moment conditions and influence functions

- Although things might look complicated at first sight, obtaining influence functions is actually quite easy in most cases. Also subpopulation estimation can be integrated without much trouble and there is a simple solution to take account of covariate balancing based on reweighting.
- Hence, influence functions are the method of choice for the new command.
- One great thing about influence functions is that support for complex survey estimation (`svy`) comes for free.
- Another great thing is that the influence function of a linear or nonlinear combination of several statistics can be obtained by linear combination of the individual influence functions (as implied by the chain rule).
- Furthermore, the RIFs (recentered influence functions) can be used in regression models to study approximate effects of covariates on a statistic (Firpo et al. 2009).

- 1 Introduction
- 2 Theory
- 3 The new command**
- 4 Examples
- 5 Conclusions

Estimation

- Distribution functions:

```
dstat subcmd varlist [if] [in] [weight] [, options]
```

where *subcmd* is one of density (or pdf), histogram, proportion, frequency, cdf, ccdf, quantile, lorenz, share, tip.

- Summary statistics:

```
dstat [(stats)] varlist [(stats) varlist ...] [if] [in]  
      [weight] [, options]
```

where *stats* is a space-separated list of statistics. A large collection of statistics is available ...

Points in the distribution

quantile(p)	$p/100$ quantile; p in $[0,100]$
p(p)	alias for quantile()
hdquantile(p)	$p/100$ Harrell/Davis (1982) quantile; p in $[0,100]$
density(x)	kernel density at value x
hist(x1,x2)	histogram density of data within $(x1,x2]$
cdf[*](x)	cumulative distribution function (CDF) at value x ; suffix $*$ is empty for default, m for mid-adjusted CDF, f for floor CDF
ccdf[*](x)	complementary CDF at value x ; suffix $*$ is empty for default, m for mid-adjusted CCDF, f for floor CCDF
prop(x1[,x2])	proportion of data equal to $x1$ or within $[x1,x2]$
pct(x1[,x2])	percent of data equal to $x1$ or within $[x1,x2]$
freq(x1[,x2])	frequency of data equal to $x1$ or within $[x1,x2]$
total[(x1[,x2])]	overall total, or total of data equal to $x1$ or within $[x1,x2]$
min	observed minimum (standard error set to zero)
max	observed maximum (standard error set to zero)
range	$\text{max}-\text{min}$ (standard error set to zero)
midrange	$(\text{min}+\text{max})/2$ (standard error set to zero)

Location measures

mean	arithmetic mean
gmean	geometric mean (data must be positive)
hmean	harmonic mean (data must be positive)
trim(p1[,p2])	trimmed mean with $p1/100$ lower trimming and $p2/100$ upper trimming; $p1$ and $p2$ in $[0,50]$; $p2=p1$ if omitted; default is $p1=p2=25$
winsor(p1[,p2])	winsorized mean with $p1/100$ lower winsorizing and $p2/100$ upper winsorizing; $p1$ and $p2$ in $[0,50]$; $p2=p1$ if omitted; default is $p1=p2=25$
median	median; equal to q50
huber[(p)]	Huber M estimate with gaussian efficiency p in $[63.7,99.9]$; default is $p=95$
biweight[(p)]	biweight M estimate with gaussian efficiency p in $[.01,99.9]$; default is $p=95$
hl	Hodges-Lehmann location measure (Hodges and Lehmann 1963)

Scale measures

<code>sd[(df)]</code>	standard deviation; <i>df</i> applies small-sample adjustment; default is <i>df</i> =1
<code>variance[(df)]</code>	variance; default is <i>df</i> =1
<code>mse[{x},(df)]</code>	mean squared deviation from value <i>x</i> (mean squared error); default is <i>x</i> =0 and <i>df</i> =0
<code>smse[{x},(df)]</code>	square-root of mean squared deviation from value <i>x</i> ; default is <i>x</i> =0 and <i>df</i> =0
<code>iqr[n]([p1,p2])</code>	interquantile range; default is <code>iqr(25,75)</code> (interquartile range); specify <code>iqrn</code> for normalized IQR, equal to $1/(\text{invnormal}(p2) - \text{invnormal}(p1)) * \text{iqr}$
<code>mad[n]([l],[t])</code>	median (or mean if <i>l</i> !=0) absolute deviation from the median (or mean if <i>t</i> !=0); specify <code>madn</code> for normalized MAD, equal to $1/\text{invnormal}(0.75) * \text{mad}$ (or $\sqrt{\text{pi}/2} * \text{mad}$ if <i>l</i> !=0)
<code>mae[n]([l],[x])</code>	median (or mean if <i>l</i> !=0) absolute deviation from value <i>x</i> ; default is <i>x</i> =0; specify <code>maen</code> for normalized MAE, equal to $1/\text{invnormal}(0.75) * \text{mae}$ or $(\sqrt{\text{pi}/2}) * \text{mae}$ if <i>l</i> !=0
<code>md[n]</code>	mean absolute pairwise difference; equal to $2 * \text{mean} * \text{gini}$; specify <code>mdn</code> for normalized MD, equal to $\sqrt{\text{pi}}/2 * \text{md}$
<code>mscale[(bp)]</code>	M estimate of scale with breakdown point <i>bp</i> in [1,50]; default is <i>bp</i> =50
<code>qn</code>	Qn scale coefficient (Rousseeuw and Croux 1993)

Skewness measures

<code>skewness</code>	skewness
<code>qskew[(alpha)]</code>	quantile skewness (Hinkley 1975); <i>alpha</i> in [0,50]; default is <i>alpha</i> =25
<code>mc</code>	medcouple (Brys et al. 2004)

Kurtosis measures

<code>kurtosis</code>	kurtosis
<code>qw[(alpha)]</code>	quantile tail weight; <i>alpha</i> in [0,50]; default is <i>alpha</i> =25
<code>lqw[(alpha)]</code>	left quantile tail weight; <i>alpha</i> in [0,50]; default is <i>alpha</i> =25
<code>rqw[(alpha)]</code>	right quantile tail weight; <i>alpha</i> in [0,50]; default is <i>alpha</i> =25
<code>lmc</code>	left medcouple tail weight measure (Brys et al. 2006)
<code>rmc</code>	right medcouple tail weight measure (Brys et al. 2006)

Inequality measures

hoover	Hoover index (Robin Hood index)
gini [(<i>df</i>)]	Gini coefficient; <i>df</i> applies small-sample adjustment; default is <i>df</i> =0
agini [(<i>df</i>)]	absolute Gini coefficient
mld	mean log deviation; equal to ge (0)
theil	Theil index; equal to ge (1)
cv [(<i>df</i>)]	coefficient of variation; default is <i>df</i> =1; cv (0)=sqrt(2* ge (1))
ge [(<i>alpha</i>)]	generalized entropy (Shorrocks 1980) with parameter <i>alpha</i>
atkinson [(<i>epsilon</i>)]	Atkinson index with parameter <i>epsilon</i> >=0; default is <i>epsilon</i> =1
lvar [(<i>df</i>)]	logarithmic variance; default is <i>df</i> =1
vlog [(<i>df</i>)]	variance of logarithm; default is <i>df</i> =1
top [(<i>p</i>)]	outcome share of top <i>p</i> percent; default is <i>p</i> =10
bottom [(<i>p</i>)]	outcome share of bottom <i>p</i> percent; default is <i>p</i> =40
mid [(<i>p1,p2</i>)]	outcome share of mid <i>p1</i> to <i>p2</i> percent; default is <i>p1</i> =40 and <i>p2</i> =90
palma	palma ratio; equal to top / bottom or sratio (40,90)
qratio [(<i>p1,p2</i>)]	quantile ratio $q(p2)/q(p1)$; default is <i>p1</i> =10 and <i>p2</i> =90
sratio [(<i>l1,u1,l2,u2</i>)]	percentile share ratio; default is <i>l1</i> =0, <i>u1</i> =10, <i>l2</i> =90, <i>u2</i> =100; can also specify sratio (<i>u1,l2</i>)
[*]lorenz (<i>p</i>)	Lorenz ordinate, <i>p</i> in [0,100]; prefix * is empty for default, g for generalized, t for total, a for absolute, e for equality gap
[*]share (<i>p1,p2</i>)	percentile share, <i>p1</i> and <i>p2</i> in [0,100]; prefix * is empty for default, d for density, g for generalized, t for total, a for average

Concentration measures

gci [(<i>zvar</i> , <i>df</i>)]	Gini concentration index; <i>zvar</i> specifies the sort variable; default is as set by option zvar (); <i>df</i> applies small-sample adjustment; default is <i>df</i> =0; can also specify gci (<i>df</i>)
aci [(<i>zvar</i> , <i>df</i>)]	absolute Gini concentration index; syntax as for gci
[*]ccurve (<i>p</i> , <i>zvar</i>)	concentration curve ordinate, <i>p</i> in [0,100]; prefix * is empty for default, g for generalized, t for total, a for absolute, e for equality gap
[*]cshare (<i>p1,p2</i> , <i>zvar</i>)	concentration share, <i>p1</i> and <i>p2</i> in [0,100]; prefix * is empty for default, d for density, g for generalized, t for total, a for average

Poverty measures

<code>hcr[(pline)]</code>	head count ratio (i.e. proportion poor); <i>pline</i> specifies the poverty line > 0; <i>pline</i> can be <i>varname</i> or #; default is as set by option <code>pline()</code>
<code>[a]pgap[(pline)]</code>	poverty gap (proportion by which mean outcome of poor is below <i>pline</i>); specify <code>apgap</code> for absolute poverty gap (<i>pline</i> - mean outcome of poor)
<code>[a]pgi[(pline)]</code>	poverty gap index; equal to <code>hcr*pgap</code> ; specify <code>apgi</code> for absolute poverty gap index, equal to <code>hcr*apgap</code>
<code>fgt[(a[,pline])]</code>	Foster-Greer-Thorbecke index with $a \geq 0$ (Foster et al. 1984, 2010); default is $a=0$ (head count ratio); $a=1$ is equivalent to <code>pgi</code>
<code>sen[(pline)]</code>	Sen poverty index (Sen 1976; using the replication invariant version of the index, also see Shorrocks 1995)
<code>sst[(pline)]</code>	Sen-Shorrocks-Thon poverty index (see, e.g., Osberg and Xu 2008)
<code>takayama[(pline)]</code>	Takayama poverty index (Takayama 1979)
<code>watts[(pline)]</code>	Watts index (see, e.g., Saisana 2014)
<code>chu[(a[,pline])]</code>	Clark-Hemming-Ulph poverty index with a in $[0,100]$ (Clark et al. 1981); default is $a=50$; $a=0$ is equivalent to $1-\exp(-watts)$; $a=100$ is equivalent to <code>fgt(1)</code>
<code>[a]tip(ρ,pline)</code>	TIP ordinate, ρ in $[0,100]$; specify <code>atip()</code> for absolute TIP ordinates

Association

<code>corr[(zvar)]</code>	correlation coefficient; <i>zvar</i> specifies the secondary variable; default is as set by option <code>zvar()</code>
<code>cov[(zvar[,df])]</code>	covariance; <i>df</i> applies small-sample adjustment; default is $df=1$; can also specify <code>cov(df)</code>
<code>spearman[(zvar)]</code>	Spearman's rank correlation

Some main options

- `over(overvar [, options]) [total]`
 - ▶ compute results by subpopulations, possibly including total, possibly accumulating or taking contrasts
- `balance([method:] varlist [, options])`
 - ▶ balance covariates using entropy balancing (see Jann 2021) or IPW
- `nocasewise`
 - ▶ exclude missing values for each variable individually (no casewise deletion of observations)
- `vce(vcetype [, options])`
 - ▶ note: specify `vce(svy)` for complex survey estimation instead of applying the `svy` prefix!
- There are many more options; see `help dstat` for details.

After estimation

- Draw graph:

```
dstat graph [ , graph_options ]
```

or apply option `graph()` when estimating. The graphs will be created by an internal call to `coefplot` (Jann 2014).

- Obtain influence functions (optionally recentered):

```
predict { stub* | newvarlist } [if] [in] [ , predict_options ]
```

or apply option `generate()` when estimating.

- 1 Introduction
- 2 Theory
- 3 The new command
- 4 Examples**
- 5 Conclusions

```
. use sess16, clear
(Sample from Swiss Earnings Structure Survey 2016)
```

```
. describe
```

```
Contains data from sess16.dta
```

```
Observations:      50,000      Sample from Swiss Earnings Structure
                               Survey 2016
Variables:          5          24 Jun 2021 21:38
```

Variable name	Storage type	Display format	Value label	Variable label
earnings	long	%10.0g		monthly earnings in CHF (full-time equivalent)
gender	byte	%8.0g	gender	gender
educ	byte	%27.0g	educ	highest educational degree
tenure	byte	%8.0g		tenure (in years)
wgt	double	%10.0g		sampling weight

```
Sorted by:
```

```
. summarize
```

Variable	Obs	Mean	Std. dev.	Min	Max
earnings	47,600	7848.055	4189.382	2314	103998
gender	49,771	.5547608	.4969972	0	1
educ	49,503	2.797063	1.304769	1	5
tenure	48,525	8.599588	8.934825	0	61
wgt	50,000	33.19645	61.75064	8.435029	2991.433

- If you specify no subcommand and no statistics, `dstat` behaves like official `mean`:

```
. dstat earnings [pw=wt], over(gender)
```

```
mean                               Number of obs   =   47,383
```

earnings	Coefficient	Std. err.	[95% conf. interval]	
gender				
female	6511.388	31.48064	6449.686	6573.091
male	8007.006	53.82319	7901.511	8112.5

```
. mean earnings [pw=wt], over(gender)
```

```
Mean estimation                               Number of obs = 47,383
```

	Mean	Std. err.	[95% conf. interval]	
c.earnings@gender				
female	6511.388	31.48064	6449.686	6573.091
male	8007.006	53.82319	7901.511	8112.5

- Other than mean it allows you to include the total across subpopulations or to take contrasts:

```
. dstat earnings [pw=wt], over(gender, contrast(1) ratio)
```

```
Ratio of mean          Number of obs   =    47,383
                       Contrast           =     1.gender
```

earnings	Coefficient	Std. err.	[95% conf. interval]	
gender				
female	.8132114	.0067335	.8000137	.8264091

```
. dstat earnings [pw=wt], over(gender) total
```

```
mean          Number of obs   =    47,383
```

earnings	Coefficient	Std. err.	[95% conf. interval]	
gender				
female	6511.388	31.48064	6449.686	6573.091
male	8007.006	53.82319	7901.511	8112.5
total	7366.241	34.06905	7299.465	7433.017

- You can also select and reorder subpopulations (total will still be across all subpopulations):

```
. dstat earnings [pw=wt], over(educ) total
```

```
mean                               Number of obs   =   47,129
```

earnings	Coefficient	Std. err.	[95% conf. interval]	
educ				
Lower secondary	5093.716	38.91248	5017.447	5169.985
Upper secondary: vocational	6380.539	34.30033	6313.31	6447.768
Upper secondary: general	7438.123	141.6259	7160.534	7715.711
Tertiary: vocational	9019.856	56.53847	8909.039	9130.672
Tertiary: academic	11471.11	170.9918	11135.96	11806.25
total	7369.873	34.33644	7302.573	7437.173

```
. dstat earnings [pw=wt], over(educ, select(5 4)) total
```

```
mean                               Number of obs   =   47,129
```

earnings	Coefficient	Std. err.	[95% conf. interval]	
educ				
Tertiary: academic	11471.11	170.9918	11135.96	11806.25
Tertiary: vocational	9019.856	56.53847	8909.039	9130.672
total	7369.873	34.33644	7302.573	7437.173

- Furthermore, `dstat` supports IPW covariate balancing. Here is an “ATT” of being male (earnings gap if women’s sample is reweighted):

```
. dstat earnings [pw=wgt], over(gender, contrast) balance(i.educ tenure, ref(1))
Difference in mean      Number of obs      =    45,530
                       Contrast          =     0.gender
                       Balancing:
                           method =      ipw
                           reference =  1.gender
                           controls = e(balance)
```

earnings	Coefficient	Std. err.	t	P> t	[95% conf. interval]	
gender male	1261.559	55.857	22.59	0.000	1152.078	1371.039

```
. teffects ipw (earnings) (gender i.educ tenure) [pw=wgt], atet nolog
Treatment-effects estimation      Number of obs      =    45,530
Estimator      : inverse-probability weights
Outcome model  : weighted mean
Treatment model: logit
```

earnings		Coefficient	Robust std. err.	z	P> z	[95% conf. interval]	
ATET	gender (male vs female)	1261.559	55.85639	22.59	0.000	1152.082	1371.035
POmean	gender female	6755.25	34.66481	194.87	0.000	6687.308	6823.192

- IPW does not perfectly balance the data ...

```
. dstat (pr1 pr2 pr3 pr4 pr5) educ (mean) tenure [pw=wt] if earnings<., ///
> over(gender, contrast) balance(i.educ tenure, ref(1))
```

Difference in summary statistics

```
Number of obs      =    45,530
Contrast            =    0.gender
Balancing:
  method            =    ipw
  reference          =    1.gender
  controls           =    e(balance)
```

0: gender = female
1: gender = male

	Coefficient	Std. err.	t	P> t	[95% conf. interval]	
1-educ						
pr1	-.0017249	.0005537	-3.12	0.002	-.0028101	-.0006396
pr2	.0006886	.0009046	0.76	0.447	-.0010844	.0024616
pr3	-.0002834	.0003077	-0.92	0.357	-.0008864	.0003197
pr4	.0003433	.000665	0.52	0.606	-.0009601	.0016466
pr5	.0009764	.0004809	2.03	0.042	.0000338	.0019189
1-tenure						
mean	.1103688	.0254176	4.34	0.000	.0605498	.1601877

- ... so we may prefer entropy balancing (see Jann 2021):

```
. dstat earnings [pw=wt], over(gender, contrast) balance(eb:i.educ tenure, ref(1))
Difference in mean      Number of obs      =      45,530
                       Contrast              =      0.gender
                       Balancing:
                           method =          eb
                           reference =    1.gender
                           controls = e(balance)
```

earnings	Coefficient	Std. err.	t	P> t	[95% conf. interval]	
gender						
male	1247.068	55.99405	22.27	0.000	1137.318	1356.817

```
. kmatch eb gender i.educ tenure (earnings = i.educ tenure) [pw=wt], att nomtable
(fitting balancing weights ... done)
```

```
Entropy balancing      Number of obs = 45,530
                       Balance tolerance = .00001
```

```
Treatment   : gender = 1
Targets     : 1
Covariates  : i.educ tenure
RA equations: earnings = i.educ tenure _cons
```

Treatment-effects estimation

earnings	Coefficient	Std. err.	t	P> t	[95% conf. interval]	
ATT	1247.068	55.99405	22.27	0.000	1137.318	1356.817

- Perfect balance!

```
. dstat (pr1 pr2 pr3 pr4 pr5) educ (mean) tenure [pw=wgt] if earnings<., ///
> over(gender, contrast) balance(eb:i.educ tenure, ref(1))
```

```
Difference in summary statistics          Number of obs    =    45,530
                                           Contrast          =     0.gender
                                           Balancing:
                                           method =          eb
                                           reference =    1.gender
                                           controls = e(balance)
```

```
0: gender = female
1: gender = male
```

	Coefficient	Std. err.	t	P> t	[95% conf. interval]	
1-educ						
pr1	-3.61e-16	2.12e-17	-17.05	0.000	-4.02e-16	-3.19e-16
pr2	2.22e-16	1.48e-17	14.95	0.000	1.93e-16	2.51e-16
pr3	2.08e-17	1.41e-18	14.78	0.000	1.81e-17	2.36e-17
pr4	1.39e-16	6.77e-18	20.50	0.000	1.26e-16	1.52e-16
pr5	6.94e-17	3.99e-18	17.39	0.000	6.16e-17	7.72e-17
1-tenure						
mean	-3.55e-15	2.22e-16	-15.99	0.000	-3.99e-15	-3.12e-15

- All of the above you can do with any other statistic, also with multiple statistics and multiple variables at the same time!

```
. generate llearn = ln(earnings)
(2,400 missing values generated)
. dstat (mean gini mld vlog) earnings (mean var) llearn [pw=wgt], ///
> over(gender, contrast) balance(eb:i.educ tenure, ref(1))
```

Difference in summary statistics

```
Number of obs      =      45,530
Contrast           =      0.gender
Balancing:
    method         =          eb
reference          =      1.gender
controls           = e(balance)
```

0: gender = female
1: gender = male

	Coefficient	Std. err.	t	P> t	[95% conf. interval]	
1-earnings						
mean	1247.068	55.99405	22.27	0.000	1137.318	1356.817
gini	.0393687	.0038698	10.17	0.000	.0317838	.0469536
mld	.0298174	.0031885	9.35	0.000	.0235679	.036067
vlog	.0421788	.0043786	9.63	0.000	.0335967	.0507609
1-llearn						
mean	.1392601	.0056122	24.81	0.000	.1282601	.1502602
var	.0421788	.0043786	9.63	0.000	.0335967	.0507609

- Some other stuff. Here is the educational distribution by gender:

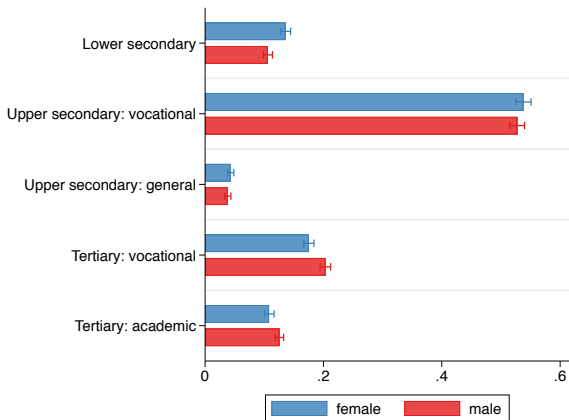
```
. dstat prop educ [pw=wt], over(gender) graph(merge)
```

Proportion Number of obs = 49,277

0: gender = female

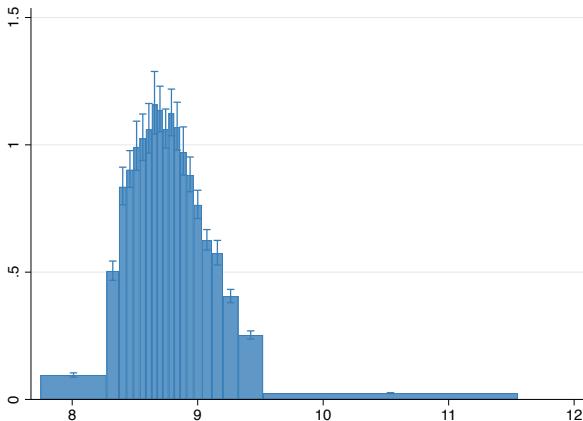
1: gender = male

(coefficients table suppressed)



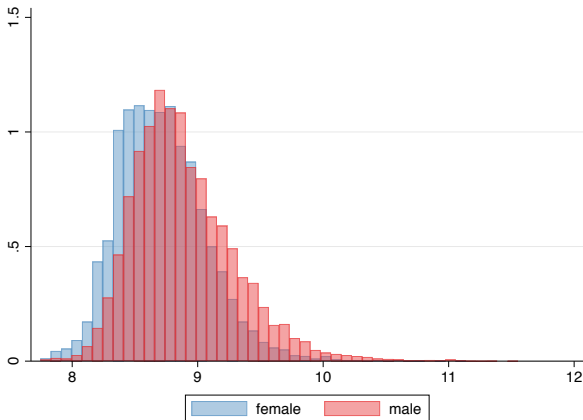
- Or an equal probability histogram of log earnings:

```
. dstat histogram lnearn [pw=wt], ep n(20) graph  
Histogram (density)          Number of obs   =    47,600  
(coefficients table suppressed)
```



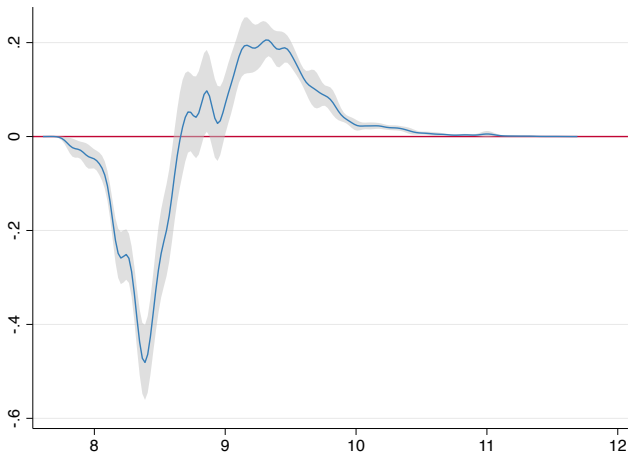
- Or histograms of log earnings by gender printed on top of each other using the same bin definitions:

```
. dstat histogram lnearn [pw=wt], over(gender) common nose graph(merge)
Histogram (density)          Number of obs   =    47,383
      0: gender = female
      1: gender = male
(coefficients table suppressed)
```



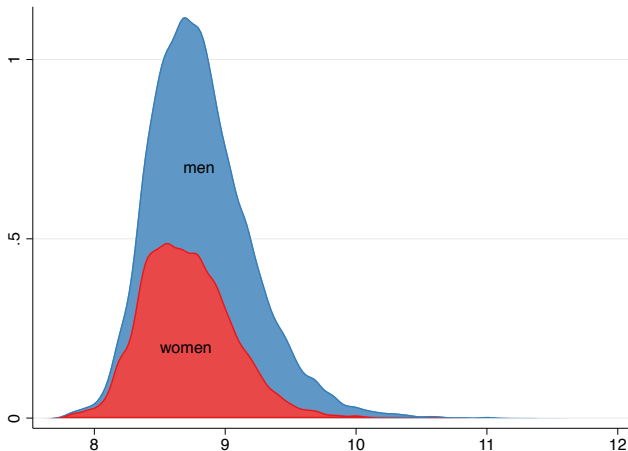
- Or the difference in the density function of log earnings by gender:

```
. dstat density llearn [pw=wgt], over(gender, contrast) n(200) ///  
> graph(ylines(0))  
(output omitted)
```



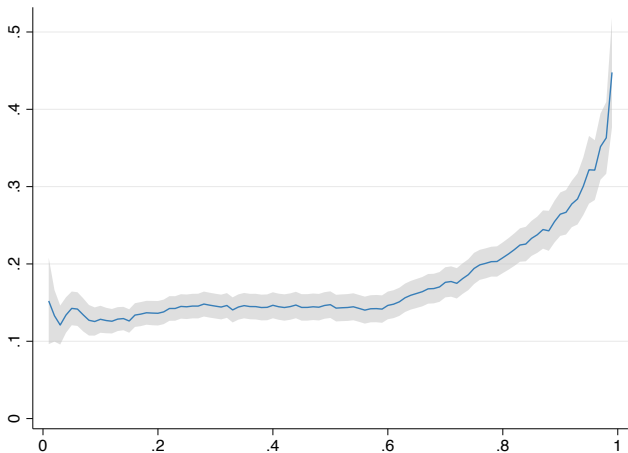
- Or the composition of the overall density of log earnings by gender:

```
. dstat density llearn [pw=wgt], over(gender) total nose n(200) unconditional ///  
> graph(recast(area) select(3 1) merge legend(off) ///  
> text(.2 8.7 "women" .7 8.8 "men"))  
(output omitted)
```



- Or the difference in quantile functions of log earnings by gender:

```
. dstat quantile llearn [pw=wgt], over(gender, contrast) graph(ylabel(0(.1).5))  
  (output omitted)
```



- `dstat` cannot compute balanced and unbalanced results at the same time. If you want include both sets of results in the same graph, you need to store the estimates and use `coefplot` manually.
- Here is how we can create a graph that shows the quantile wage gap function with and without covariate adjustment:

```
. dstat quantile llearn [pw=wtg], over(gender, contrast) notable ///
>     balance(eb:i.educ tenure)
    (output omitted)
. estimates store balanced
. dstat quantile llearn [pw=wtg] if e(sample), over(gender, contrast)
    (output omitted)
. estimates store raw
. coefplot raw balanced, se(se) at(at) keep(1:) ylabel(0(.1).5) ///
>     recast(line) ciopts(recast(rarea) pstyle(ci) color(%50) lcolor(%0)) ///
>     plotlabels("raw wage gap" "adjusted wage gap")
```



- 1 Introduction
- 2 Theory
- 3 The new command
- 4 Examples
- 5 Conclusions**

Conclusions

- I could go on forever. There would be so much more to say ...
... for example, on the cool stuff you can do with the influence functions (or RIFs) generated by `dstat`.
- Have fun with the command!
- Drop me a note if you want me to add a specific statistic.
- Install from SSC

```
. ssc install dstat, replace  
. ssc install moremata, replace  
. ssc install coefplot, replace
```

or from GitHub: <http://github.com/benjann/dstat>

References

- Deville, J.C. 1999. Variance estimation for complex statistics and estimators: linearization and residual techniques. *Survey Methodology* 25:193–203.
- Firpo, S., N.M. Fortin, T. Lemieux. 2009. Unconditional quantile regressions. *Econometrica* 77:953–973.
- Hampel, F.R. 1974. The influence curve and its role in robust estimation. *Journal of the American Statistical Association* 69:383–393.
- Jann, B. 2014. Plotting regression coefficients and other estimates. *The Stata Journal* 14:708-737.
- Jann, B. 2020. Influence functions continued. A framework for estimating standard errors in reweighting, matching, and regression adjustment. University of Bern Social Sciences Working Papers 35. Available from <https://ideas.repec.org/p/bss/wpaper/35.html>.
- Jann, B. 2021. Entropy balancing as an estimation command. University of Bern Social Sciences Working Papers 39. Available from <https://ideas.repec.org/p/bss/wpaper/39.html>.