

## Title

**regress** — Linear regression

## Syntax

`regress depvar [indepvars] [if] [in] [weight] [, options]`

<i>options</i>	description
Model	
<code>noconstant</code>	suppress constant term
<code>hascons</code>	has user-supplied constant
<code>tsscons</code>	compute total sum of squares with constant; seldom used
SE/Robust	
<code>vce(vctype)</code>	<i>vctype</i> may be <code>ols</code> , <code>robust</code> , <code>cluster clustvar</code> , <code>bootstrap</code> , <code>jackknife</code> , <code>hc2</code> , or <code>hc3</code>
Reporting	
<code>level(#)</code>	set confidence level; default is <code>level(95)</code>
<code>beta</code>	report standardized beta coefficients
<code>eform(string)</code>	report exponentiated coefficients and label as <i>string</i>
<code>noheader</code>	suppress table header
<code>plus</code>	make table extendable
<code>depname(varname)</code>	substitute dependent variable name; programmer's option
<code>†mse1</code>	force mean squared error to 1

*depvar* and *indepvars* may contain time-series operators; see [U] **11.4.3 Time-series varlists**.

`bootstrap`, `by`, `jackknife`, `nestreg`, `rolling`, `statsby`, `stepwise`, `svy`, and `xi` are allowed; see [U] **11.1.10 Prefix commands**.

Weights are not allowed with the `bootstrap` prefix.

`aweight`s are not allowed with the `jackknife` prefix.

`hascons`, `tsscons`, `vce()`, `beta`, `noheader`, `plus`, `depname()`, `mse1`, and weights are not allowed with the `svy` prefix.

`†mse1` does not appear in the dialog box.

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

See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

## Description

`regress` fits a model of *depvar* on *indepvars* using linear regression.

Here is a short list of other regression commands that may be of interest. See [I] **estimation commands** for a complete list.

---

<code>areg</code>	[R] <b>areg</b>	an easier way to fit regressions with many dummy variables
<code>arch</code>	[TS] <b>arch</b>	regression models with ARCH errors
<code>arima</code>	[TS] <b>arima</b>	ARIMA models
<code>boxcox</code>	[R] <b>boxcox</b>	Box–Cox regression models
<code>cnreg</code>	[R] <b>cnreg</b>	censored-normal regression
<code>cnsgreg</code>	[R] <b>cnsgreg</b>	constrained linear regression
<code>eivreg</code>	[R] <b>eivreg</b>	errors-in-variables regression
<code>frontier</code>	[R] <b>frontier</b>	stochastic frontier models
<code>heckman</code>	[R] <b>heckman</b>	Heckman selection model
<code>intreg</code>	[R] <b>intreg</b>	interval regression
<code>ivregress</code>	[R] <b>ivregress</b>	single-equation instrumental-variables regression
<code>ivtobit</code>	[R] <b>ivtobit</b>	tobit regression with endogenous variables
<code>newey</code>	[TS] <b>newey</b>	regression with Newey–West standard errors
<code>qreg</code>	[R] <b>qreg</b>	quantile (including median) regression
<code>reg3</code>	[R] <b>reg3</b>	three-stage least-squares (3SLS) regression
<code>rreg</code>	[R] <b>rreg</b>	a type of robust regression
<code>sureg</code>	[R] <b>sureg</b>	seemingly unrelated regression
<code>tobit</code>	[R] <b>tobit</b>	tobit regression
<code>treatreg</code>	[R] <b>treatreg</b>	treatment-effects model
<code>truncreg</code>	[R] <b>truncreg</b>	truncated regression
<code>xtabond</code>	[XT] <b>xtabond</b>	Arellano–Bond linear dynamic panel-data estimation
<code>xtdpd</code>	[XT] <b>xtdpd</b>	linear dynamic panel-data estimation
<code>xtfrontier</code>	[XT] <b>xtfrontier</b>	panel-data stochastic frontier models
<code>xtgls</code>	[XT] <b>xtgls</b>	panel-data GLS models
<code>xthtaylor</code>	[XT] <b>xthtaylor</b>	Hausman–Taylor estimator for error-components models
<code>xtintreg</code>	[XT] <b>xtintreg</b>	panel-data interval regression models
<code>xtivreg</code>	[XT] <b>xtivreg</b>	panel-data instrumental variables (2SLS) regression
<code>xtpcse</code>	[XT] <b>xtpcse</b>	linear regression with panel-corrected standard errors
<code>xtreg</code>	[XT] <b>xtreg</b>	fixed- and random-effects linear models
<code>xtregar</code>	[XT] <b>xtregar</b>	fixed- and random-effects linear models with an AR(1) disturbance
<code>xttobit</code>	[XT] <b>xttobit</b>	panel-data tobit models

---

## Options

Model

`noconstant`; see [R] **estimation options**.

`hascons` indicates that a user-defined constant or its equivalent is specified among the independent variables in *indepvars*. Some caution is recommended when specifying this option, as resulting estimates may not be as accurate as they otherwise would be. Use of this option requires “sweeping” the constant last, so the moment matrix must be accumulated in absolute rather than deviation form.

This option may be safely specified when the means of the dependent and independent variables are all reasonable and there is not much collinearity between the independent variables. The best procedure is to view `hascons` as a reporting option—estimate with and without `hascons` and verify that the coefficients and standard errors of the variables not affected by the identity of the constant are unchanged.

`tsscons` forces the total sum of squares to be computed as though the model has a constant, that is, as deviations from the mean of the dependent variable. This is a rarely used option that has an effect only when specified with `noconstant`. It affects the total sum of squares and all results derived from the total sum of squares.

---

#### SE/Robust

---

`vce(vctype)` specifies the type of standard error reported, which includes types that are derived from asymptotic theory, that are robust to some kinds of misspecification, that allow for intragroup correlation, and that use bootstrap or jackknife methods; see [R] *vce\_option*.

`vce(ols)`, the default, uses the standard variance estimator for ordinary least-squares regression. `regress` also allows the following:

`vce(hc2)` and `vce(hc3)` specify an alternative bias correction for the robust variance calculation. `vce(hc2)` and `vce(hc3)` may not be specified with `svy` prefix. In the unclustered case, `vce(robust)` uses  $\hat{\sigma}_j^2 = \{n/(n-k)\}u_j^2$  as an estimate of the variance of the  $j$ th observation, where  $u_j$  is the calculated residual and  $n/(n-k)$  is included to improve the overall estimate's small-sample properties.

`vce(hc2)` instead uses  $u_j^2/(1-h_{jj})$  as the observation's variance estimate, where  $h_{jj}$  is the diagonal element of the hat (projection) matrix. This estimate is unbiased if the model really is homoskedastic. `vce(hc2)` tends to produce slightly more conservative confidence intervals.

`vce(hc3)` uses  $u_j^2/(1-h_{jj})^2$  as suggested by Davidson and MacKinnon (1993), who report that this method tends to produce better results when the model really is heteroskedastic. `vce(hc3)` produces confidence intervals that tend to be even more conservative.

See Davidson and MacKinnon (1993, 554–556) for more discussion on these two bias corrections.

---

#### Reporting

---

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

`beta` asks that standardized beta coefficients be reported instead of confidence intervals. The beta coefficients are the regression coefficients obtained by first standardizing all variables to have a mean of 0 and a standard deviation of 1. `beta` may not be specified with `vce(cluster clustvar)` or the `svy` prefix.

`eform(string)` is used only in programs and ado-files that use `regress` to fit models other than linear regression. `eform()` specifies that the coefficient table be displayed in exponentiated form as defined in [R] *maximize* and that `string` be used to label the exponentiated coefficients in the table.

`noheader` suppresses the display of the ANOVA table and summary statistics at the top of the output; only the coefficient table is displayed. This option is often used in programs and ado-files.

`plus` specifies that the output table be made extendable. This option is often used in programs and ado-files.

`depname` (*varname*) is used only in programs and ado-files that use `regress` to fit models other than linear regression. `depname()` may be specified only at estimation time. *varname* is recorded as the identity of the dependent variable, even though the estimates are calculated using *depvar*. This method affects the labeling of the output—not the results calculated—but could affect subsequent calculations made by `predict`, where the residual would be calculated as deviations from *varname* rather than *depvar*. `depname()` is most typically used when *depvar* is a temporary variable (see [P] **macro**) used as a proxy for *varname*.

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

`mse1` is used only in programs and ado-files that use `regress` to fit models other than linear regression and is not allowed with the `svy` prefix. `mse1` sets the mean squared error to 1, forcing the variance–covariance matrix of the estimators to be  $(\mathbf{X}'\mathbf{D}\mathbf{X})^{-1}$  (see *Methods and Formulas* below) and affecting calculated standard errors. Degrees of freedom for *t* statistics are calculated as *n* rather than  $n - k$ .

## Remarks

Remarks are presented under the following headings:

- Ordinary least squares*
- Treatment of the constant*
- Robust standard errors*
- Weighted regression*
- Instrumental variables and two-stage least-squares regression*

`regress` performs linear regression, including ordinary least squares and weighted least squares. For a general discussion of linear regression, see Draper and Smith (1998), Johnston and DiNardo (1997), or Kmenta (1997).

See Wooldridge (2006) for an excellent treatment of estimation, inference, interpretation, and specification testing in linear regression models. This presentation stands out for its clarification of the statistical issues, as opposed to the algebraic issues. See Wooldridge (2002, chap. 4) for a more advanced discussion along the same lines.

See Hamilton (2006, chap. 6) for an introduction to linear regression using Stata. Dohoo, Martin, and Stryhn (2003) discuss linear regression using examples from epidemiology, and Stata datasets and do-files used in the text are available.

Chatterjee and Hadi (2006) explain regression analysis by using examples containing typical problems that you might encounter when performing exploratory data analysis. We also recommend Weisberg (2005), who emphasizes the importance of the assumptions of linear regression and problems resulting from these assumptions. For a discussion of model-selection techniques and exploratory data analysis, see Mosteller and Tukey (1977). For a mathematically rigorous treatment, see Peracchi (2001, chap. 6). Finally, see Plackett (1972) if you are interested in the history of regression. Least squares, which dates back to the 1790s, was discovered independently by Legendre and Gauss.

## Ordinary least squares

### ▷ Example 1

Suppose that we have data on the mileage rating and weight of 74 automobiles. The variables in our data are `mpg`, `weight`, and `foreign`. The last variable assumes the value 1 for foreign and 0 for domestic automobiles. We wish to fit the model

$$\text{mpg} = \beta_0 + \beta_1 \text{weight} + \beta_2 \text{weight}^2 + \beta_3 \text{foreign} + \epsilon$$

We do this by creating a new variable called `weightsq` and then typing `regress mpg weight weightsq foreign`:

```
. use http://www.stata-press.com/data/r10/auto
(1978 Automobile Data)
. generate weightsq=weight^2
. regress mpg weight weightsq foreign
```

Source	SS	df	MS			
Model	1689.15372	3	563.05124	Number of obs =	74	
Residual	754.30574	70	10.7757963	F( 3, 70) =	52.25	
Total	2443.45946	73	33.4720474	Prob > F =	0.0000	
				R-squared =	0.6913	
				Adj R-squared =	0.6781	
				Root MSE =	3.2827	

  

mpg	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
weight	-.0165729	.0039692	-4.18	0.000	-.0244892	-.0086567
weightsq	1.59e-06	6.25e-07	2.55	0.013	3.45e-07	2.84e-06
foreign	-2.2035	1.059246	-2.08	0.041	-4.3161	-.0909002
_cons	56.53884	6.197383	9.12	0.000	44.17855	68.89913

◀

`regress` produces a variety of summary statistics along with the table of regression coefficients. At the upper left, `regress` reports an analysis-of-variance (ANOVA) table. The column headings `SS`, `df`, and `MS` stand for “sum of squares”, “degrees of freedom”, and “mean square”, respectively. In the previous example, the total sum of squares is 2,443.5: 1,689.2 accounted for by the model and 754.3 left unexplained. Since the regression included a constant, the total sum reflects the sum after removal of means, as does the sum of squares due to the model. The table also reveals that there are 73 total degrees of freedom (counted as 74 observations less 1 for the mean removal), of which 3 are consumed by the model, leaving 70 for the residual.

To the right of the ANOVA table are presented other summary statistics. The  $F$  statistic associated with the ANOVA table is 52.25. The statistic has 3 numerator and 70 denominator degrees of freedom. The  $F$  statistic tests the hypothesis that all coefficients *excluding the constant* are zero. The chance of observing an  $F$  statistic that large or larger is reported as 0.0000, which is Stata’s way of indicating a number smaller than 0.00005. The  $R$ -squared ( $R^2$ ) for the regression is 0.6913, and the  $R$ -squared adjusted for degrees of freedom ( $R_a^2$ ) is 0.6781. The root mean squared error, labeled Root MSE, is 3.2827. It is the square root of the mean squared error reported for the residual in the ANOVA table.

Finally, Stata produces a table of the estimated coefficients. The first line of the table indicates that the left-hand-side variable is `mpg`. Thereafter follow the four estimated coefficients. Our fitted model is

$$\text{mpg}_{\text{hat}} = 56.54 - 0.0166 \text{weight} + 1.59 \cdot 10^{-6} \text{weightsq} - 2.20 \text{foreign}$$

Reported to the right of the coefficients in the output are the standard errors. For instance, the standard error for the coefficient on `weight` is 0.0039692. The corresponding  $t$  statistic is  $-4.18$ , which has a two-sided significance level of 0.000. This number indicates that the significance is less than 0.0005. The 95% confidence interval for the coefficient is  $[-.024, -.009]$ .

## ▷ Example 2

`regress` shares the features of all estimation commands. Among other things, this means that after running a regression, we can use `test` to test hypotheses about the coefficients, `estat vce` to examine the covariance matrix of the estimators, and `predict` to obtain predicted values, residuals, and influence statistics. See [U] **20 Estimation and postestimation commands**. Options that affect how estimates are displayed, such as `beta` or `level()`, can be used when replaying results.

Suppose that we meant to specify the `beta` option to obtain beta coefficients (regression coefficients normalized by the ratio of the standard deviation of the regressor to the standard deviation of the dependent variable). Even though we forgot, we can specify the option now:

```
. regress, beta
```

Source	SS	df	MS		
Model	1689.15372	3	563.05124	Number of obs =	74
Residual	754.30574	70	10.7757963	F( 3, 70) =	52.25
Total	2443.45946	73	33.4720474	Prob > F =	0.0000
				R-squared =	0.6913
				Adj R-squared =	0.6781
				Root MSE =	3.2827

  

mpg	Coef.	Std. Err.	t	P> t	Beta
weight	-.0165729	.0039692	-4.18	0.000	-2.226321
weightsq	1.59e-06	6.25e-07	2.55	0.013	1.32654
foreign	-2.2035	1.059246	-2.08	0.041	-.17527
_cons	56.53884	6.197383	9.12	0.000	.

◀

## Treatment of the constant

By default, `regress` includes an intercept (constant) term in the model. The `noconstant` option suppresses it, and the `hascons` option tells `regress` that the model already has one.

## ▷ Example 3

We wish to fit a regression of the `weight` of an automobile against its `length`, and we wish to impose the constraint that the weight is zero when the length is zero.

If we simply type `regress weight length`, we are fitting the model

$$\text{weight} = \beta_0 + \beta_1 \text{length} + \epsilon$$

Here a `length` of zero corresponds to a `weight` of  $\beta_0$ . We want to force  $\beta_0$  to be zero or, equivalently, estimate an equation that does not include an intercept:

$$\text{weight} = \beta_1 \text{length} + \epsilon$$

We do this by specifying the `noconstant` option:

```
. regress weight length, noconstant
```

Source	SS	df	MS			
Model	703869302	1	703869302	Number of obs = 74		
Residual	14892897.8	73	204012.299	F( 1, 73) = 3450.13		
Total	718762200	74	9713002.7	Prob > F = 0.0000		
				R-squared = 0.9793		
				Adj R-squared = 0.9790		
				Root MSE = 451.68		
weight	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
length	16.29829	.2774752	58.74	0.000	15.74528	16.8513

In our data, `length` is measured in inches and `weight` in pounds. We discover that each inch of length adds 16 pounds to the weight.

◀

Sometimes there is no need for Stata to include a constant term in the model. Most commonly, this occurs when the model contains a set of mutually exclusive indicator variables. `hascons` is a variation of the `noconstant` option—it tells Stata not to add a constant to the regression because the regression specification already has one, either directly or indirectly.

For instance, we now refit our model of `weight` as a function of `length` and include separate constants for foreign and domestic cars. Our dataset already has an indicator variable called `foreign` marking foreign-made automobiles, so we create a corresponding `domestic` variable and fit the regression:

```
. generate domestic=!foreign
. regress weight length domestic foreign, hascons
```

Source	SS	df	MS			
Model	39647744.7	2	19823872.3	Number of obs = 74		
Residual	4446433.7	71	62625.8268	F( 2, 71) = 316.54		
Total	44094178.4	73	604029.841	Prob > F = 0.0000		
				R-squared = 0.8992		
				Adj R-squared = 0.8963		
				Root MSE = 250.25		
weight	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
length	31.44455	1.601234	19.64	0.000	28.25178	34.63732
domestic	-2850.25	315.9691	-9.02	0.000	-3480.274	-2220.225
foreign	-2983.927	275.1041	-10.85	0.000	-3532.469	-2435.385

## □ Technical Note

There is a subtle distinction between the `hascons` and `noconstant` options. We can most easily reveal it by refitting the last regression, specifying `noconstant` rather than `hascons`:

(Continued on next page)

```
. regress weight length domestic foreign, noconstant
```

Source	SS	df	MS			
Model	714315766	3	238105255	Number of obs =	74	
Residual	4446433.7	71	62625.8268	F( 3, 71) =	3802.03	
				Prob > F =	0.0000	
				R-squared =	0.9938	
				Adj R-squared =	0.9936	
Total	718762200	74	9713002.7	Root MSE =	250.25	

  

weight	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
length	31.44455	1.601234	19.64	0.000	28.25178	34.63732
domestic	-2850.25	315.9691	-9.02	0.000	-3480.274	-2220.225
foreign	-2983.927	275.1041	-10.85	0.000	-3532.469	-2435.385

Comparing this output with that produced by the previous `regress` command, we see that they are almost, but not quite, identical. The parameter estimates and their associated statistics—the second half of the output—are identical. The overall summary statistics and the ANOVA table—the first half of the output—are different, however.

In the first case, the  $R^2$  is shown as 0.8992; here it is shown as 0.9938. In the first case, the  $F$  statistic is 316.54; now it is 3802.03. The numerator degrees of freedom are different as well. In the first case, the numerator degrees of freedom are 2; now they are 3. Which is correct?

Both are. Specifying the `hascons` option causes `regress` to adjust the ANOVA table and its associated statistics for the explanatory power of the constant. The regression in effect has a constant; it is just written in such a way that a separate constant is unnecessary. No such adjustment is made with the `noconstant` option.

□

## □ Technical Note

When the `hascons` option is specified, `regress` checks to make sure that the model does in fact have a constant term. If `regress` cannot find a constant term, it automatically adds one. Fitting a model of `weight` on `length` and specifying the `hascons` option, we obtain

```
. regress weight length, hascons
(note: hascons false)
```

Source	SS	df	MS			
Model	39461306.8	1	39461306.8	Number of obs =	74	
Residual	4632871.55	72	64345.4382	F( 1, 72) =	613.27	
				Prob > F =	0.0000	
				R-squared =	0.8949	
				Adj R-squared =	0.8935	
Total	44094178.4	73	604029.841	Root MSE =	253.66	

  

weight	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
length	33.01988	1.333364	24.76	0.000	30.36187	35.67789
_cons	-3186.047	252.3113	-12.63	0.000	-3689.02	-2683.073

Even though we specified `hascons`, `regress` included a constant, anyway. It also added a note to our output: “note: hascons false”.

□

## □ Technical Note

Even if the model specification effectively includes a constant term, we need not specify the `hascons` option. `regress` is always on the lookout for collinear variables and drops them as necessary. For instance,

```
. regress weight length domestic foreign
```

Source	SS	df	MS			
Model	39647744.7	2	19823872.3	Number of obs =	74	
Residual	4446433.7	71	62625.8268	F( 2, 71) =	316.54	
Total	44094178.4	73	604029.841	Prob > F =	0.0000	
				R-squared =	0.8992	
				Adj R-squared =	0.8963	
				Root MSE =	250.25	

  

weight	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
length	31.44455	1.601234	19.64	0.000	28.25178	34.63732
domestic	(dropped)					
foreign	-133.6775	77.47615	-1.73	0.089	-288.1605	20.80555
_cons	-2850.25	315.9691	-9.02	0.000	-3480.274	-2220.225

□

## Robust standard errors

`regress` with the `vce(robust)` option substitutes a robust variance matrix calculation for the conventional calculation, or if `vce(cluster clustvar)` is specified, allows relaxing the assumption of independence within groups. How this method works is explained in [U] [20.15 Obtaining robust variance estimates](#). Below we show how well this approach works.

### ▷ Example 4

Specifying the `vce(robust)` option is equivalent to requesting White-corrected standard errors in the presence of heteroskedasticity. We use the automobile data and, in the process of looking at the energy efficiency of cars, analyze a variable with considerable heteroskedasticity.

We will examine the amount of energy—measured in gallons of gasoline—that the cars in the data need to move 1,000 pounds of their weight 100 miles. We are going to examine the relative efficiency of foreign and domestic cars.

```
. gen gpmw = ((1/mpg)/weight)*100*1000
. summarize gpmw
```

Variable	Obs	Mean	Std. Dev.	Min	Max
gpmw	74	1.682184	.2426311	1.09553	2.30521

In these data, the engines consume between 1.10 and 2.31 gallons of gas to move 1,000 pounds of the car's weight 100 miles. If we ran a regression with conventional standard errors of `gpmw` on `foreign`, we would obtain

(Continued on next page)

```
. regress gpmw foreign
```

Source	SS	df	MS			
Model	.936705572	1	.936705572	Number of obs =	74	
Residual	3.36079459	72	.046677703	F( 1, 72) =	20.07	
Total	4.29750017	73	.058869865	Prob > F =	0.0000	
				R-squared =	0.2180	
				Adj R-squared =	0.2071	
				Root MSE =	.21605	

  

gpmw	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
foreign	.2461526	.0549487	4.48	0.000	.1366143	.3556909
_cons	1.609004	.0299608	53.70	0.000	1.549278	1.66873

regress with the `vce(robust)` option, on the other hand, reports

```
. regress gpmw foreign, vce(robust)
```

Linear regression

gpmw	Coef.	Robust Std. Err.	t	P> t	[95% Conf. Interval]	
foreign	.2461526	.0679238	3.62	0.001	.1107489	.3815563
_cons	1.609004	.0234535	68.60	0.000	1.56225	1.655758

The point estimates are the same (foreign cars need one-quarter gallon more gas), but the standard errors differ by roughly 20%. Conventional regression reports the 95% confidence interval as [.14, .36], whereas the robust standard errors make the interval [.11, .38].

Which is right? Notice that `gpmw` is a variable with considerable heteroskedasticity:

```
. tabulate foreign, summarize(gpmw)
```

Car type	Summary of gpmw		Freq.
	Mean	Std. Dev.	
Domestic	1.6090039	.16845182	52
Foreign	1.8551565	.30186861	22
Total	1.6821844	.24263113	74

Thus here we favor the robust standard errors. In [U] **20.15 Obtaining robust variance estimates**, we show another example using linear regression where it makes little difference whether we specify `vce(robust)`. The linear-regression assumptions were true, and we obtained nearly linear-regression results. The advantage of the robust estimate is that in neither case did we have to check assumptions. ◀

## □ Technical Note

`regress` purposefully suppresses displaying the ANOVA table when `vce(robust)` is specified, as it is no longer appropriate in a statistical sense, even though, mechanically, the numbers would be unchanged. That is, sums of squares remain unchanged, but the meaning of those sums is no longer relevant. The  $F$  statistic, for instance, is no longer based on sums of squares; it becomes a Wald test

based on the robustly estimated variance matrix. Nevertheless, `regress` continues to report the  $R^2$  and the root MSE even though both numbers are based on sums of squares and are, strictly speaking, irrelevant. In this, the root MSE is more in violation of the spirit of the robust estimator than is  $R^2$ . As a goodness-of-fit statistic,  $R^2$  is still fine; just do not use it in formulas to obtain  $F$  statistics because those formulas no longer apply. The root MSE is valid in a literal sense—it is the square root of the mean squared error, but it is no longer an estimate of  $\sigma$  because there is no single  $\sigma$ ; the variance of the residual varies observation by observation. □

### ► Example 5

Options `vce(hc2)` and `vce(hc3)` modify the robust variance calculation. In the context of linear regression without clustering, the idea behind the robust calculation is somehow to measure  $\sigma_j^2$ , the variance of the residual associated with the  $j$ th observation, and then to use that estimate to improve the estimated variance of  $\hat{\beta}$ . Since residuals have (theoretically and practically) mean 0, one estimate of  $\sigma_j^2$  is the observation's squared residual itself— $u_j^2$ . A finite-sample correction could improve that by multiplying  $u_j^2$  by  $n/(n-k)$ , and, as a matter of fact, `vce(robust)` uses  $\{n/(n-k)\}u_j^2$  as its estimate of the residual's variance.

`vce(hc2)` and `vce(hc3)` use alternative estimators of the observation-specific variances. For instance, if the residuals are homoskedastic, we can show that the expected value of  $u_j^2$  is  $\sigma^2(1-h_{jj})$ , where  $h_{jj}$  is the  $j$ th diagonal element of the projection (hat) matrix.  $h_{jj}$  has average value  $k/n$ , so  $1-h_{jj}$  has average value  $1-k/n = (n-k)/n$ . Thus the default robust estimator  $\hat{\sigma}_j = \{n/(n-k)\}u_j^2$  amounts to dividing  $u_j^2$  by the average of the expectation.

`vce(hc2)` divides  $u_j^2$  by  $1-h_{jj}$  itself, so it should yield better estimates if the residuals really are homoskedastic. `vce(hc3)` divides  $u_j^2$  by  $(1-h_{jj})^2$  and has no such clean interpretation. Davidson and MacKinnon (1993) show that  $u_j^2/(1-h_{jj})^2$  approximates a more complicated estimator that they obtain by jackknifing (MacKinnon and White 1985).

Here are the results of refitting our efficiency model using `vce(hc2)` and `vce(hc3)`:

```
. regress gpmw foreign, vce(hc2)
Linear regression                               Number of obs =      74
                                                F( 1,    72) =    12.93
                                                Prob > F      =    0.0006
                                                R-squared     =    0.2180
                                                Root MSE     =    .21605
```

gpmw	Robust HC2		t	P> t	[95% Conf. Interval]	
	Coef.	Std. Err.				
foreign	.2461526	.0684669	3.60	0.001	.1096662	.3826389
_cons	1.609004	.0233601	68.88	0.000	1.562437	1.655571

(Continued on next page)

```
. regress gpmw foreign, vce(hc3)
```

```
Linear regression
```

```
Number of obs =      74
F( 1, 72) = 12.38
Prob > F = 0.0008
R-squared = 0.2180
Root MSE = .21605
```

gpmw	Robust HC3		t	P> t	[95% Conf. Interval]	
	Coef.	Std. Err.				
foreign	.2461526	.069969	3.52	0.001	.1066719	.3856332
_cons	1.609004	.023588	68.21	0.000	1.561982	1.656026

◀

### ▶ Example 6

The `vce(cluster clustvar)` option relaxes the assumption of independence. Below we have 28,534 observations on 4,711 women aged 14–46 years. Data were collected on these women between 1968 and 1988. We are going to fit a classic earnings model, and we begin by ignoring that each woman appears an average of 6.056 times in the data.

```
. use http://www.stata-press.com/data/r10/regsmpl
(NLS Women 14-26 in 1968)
```

```
. regress ln_wage age age2 tenure
```

Source	SS	df	MS	Number of obs = 28101		
Model	1054.52501	3	351.508335	F( 3, 28097) = 1842.45		
Residual	5360.43962	28097	.190783344	Prob > F = 0.0000		
Total	6414.96462	28100	.228290556	R-squared = 0.1644		
				Adj R-squared = 0.1643		
				Root MSE = .43679		

ln_wage	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
age	.0752172	.0034736	21.65	0.000	.0684088	.0820257
age2	-.0010851	.0000575	-18.86	0.000	-.0011979	-.0009724
tenure	.0390877	.0007743	50.48	0.000	.0375699	.0406054
_cons	.3339821	.0504413	6.62	0.000	.2351148	.4328495

The number of observations in our model is 28,101 because Stata drops observations that have a missing value for one or more of the variables in the model. We can be reasonably certain that the standard errors reported above are meaningless. Without a doubt, a woman with higher-than-average wages in one year typically has higher-than-average wages in other years, and so the residuals are not independent. One way to deal with this would be to fit a random-effects model—and we are going to do that—but first we fit the model using `regress` specifying `vce(cluster id)`, which treats only observations with different person ids as truly independent:

```
. regress ln_wage age age2 tenure, vce(cluster id)
Linear regression                               Number of obs = 28101
                                                F( 3, 4698) = 748.82
                                                Prob > F      = 0.0000
                                                R-squared    = 0.1644
                                                Root MSE    = .43679
                                                (Std. Err. adjusted for 4699 clusters in idcode)
```

ln_wage	Coef.	Robust Std. Err.	t	P> t	[95% Conf. Interval]	
age	.0752172	.0045711	16.45	0.000	.0662557	.0841788
age2	-.0010851	.0000778	-13.94	0.000	-.0012377	-.0009325
tenure	.0390877	.0014425	27.10	0.000	.0362596	.0419157
_cons	.3339821	.0641918	5.20	0.000	.208136	.4598282

For comparison, we focus on the tenure coefficient, which in economics jargon can be interpreted as the rate of return for keeping your job. The 95% confidence interval we previously estimated—an interval we do not believe—is [.038, .041]. The robust interval is twice as wide, being [.036, .042].

As we said, one correct way to fit this model is by random-effects regression. Here is the random-effects result:

```
. xtreg ln_wage age age2 tenure, re
Random-effects GLS regression                 Number of obs   = 28101
Group variable: idcode                       Number of groups = 4699
R-sq:  within = 0.1370                       Obs per group: min = 1
        between = 0.2154                       avg             = 6.0
        overall = 0.1608                       max             = 15
Random effects u_i ~ Gaussian                 Wald chi2(3)    = 4717.05
corr(u_i, X) = 0 (assumed)                   Prob > chi2     = 0.0000
```

ln_wage	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
age	.0568296	.0026958	21.08	0.000	.0515459	.0621132
age2	-.0007566	.0000447	-16.93	0.000	-.0008441	-.000669
tenure	.0260135	.0007477	34.79	0.000	.0245481	.0274789
_cons	.6136792	.0394611	15.55	0.000	.5363368	.6910216
sigma_u	.33542449					
sigma_e	.29674679					
rho	.56095413	(fraction of variance due to u_i)				

Robust regression estimated the 95% interval [.036, .042], and xtreg estimates [.025, .027]. Which is better? The random-effects regression estimator assumes a lot. We can check some of these assumptions by performing a Hausman test. Using `estimates`, we save the random-effects estimation results, and then we run the required fixed-effects regression to perform the test.

(Continued on next page)

```

. estimates store random
. xtreg ln_wage age age2 tenure, fe
Fixed-effects (within) regression      Number of obs   =   28101
Group variable: idcode                 Number of groups =   4699
R-sq:  within = 0.1375                 Obs per group:  min =    1
      between = 0.2066                   avg             =   6.0
      overall  = 0.1568                   max             =   15
                                         F(3,23399)      =  1243.00
corr(u_i, Xb) = 0.1380                 Prob > F         =   0.0000

```

ln_wage	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
age	.0522751	.002783	18.78	0.000	.0468202	.05773
age2	-.0006717	.0000461	-14.56	0.000	-.0007621	-.0005813
tenure	.021738	.000799	27.21	0.000	.020172	.023304
_cons	.687178	.0405944	16.93	0.000	.6076103	.7667456
sigma_u	.38743138					
sigma_e	.29674679					
rho	.6302569 (fraction of variance due to u_i)					

F test that all u\_i=0: F(4698, 23399) = 7.98 Prob > F = 0.0000

```
. hausman . random
```

	Coefficients			
	(b)	(B)	(b-B)	sqrt(diag(V_b-V_B))
	.	random	Difference	S.E.
age	.0522751	.0568296	-.0045545	.0006913
age2	-.0006717	-.0007566	.0000849	.0000115
tenure	.021738	.0260135	-.0042756	.0002816

```

                b = consistent under Ho and Ha; obtained from xtreg
                B = inconsistent under Ha, efficient under Ho; obtained from xtreg
Test:  Ho:  difference in coefficients not systematic
        chi2(3) = (b-B)'[(V_b-V_B)^(-1)](b-B)
                =      336.62
        Prob>chi2 =      0.0000

```

The Hausman test casts grave suspicions on the random-effects model we just fitted, so we should be careful in interpreting those results.

Meanwhile, our robust regression results still stand, as long as we are careful about the interpretation. The correct interpretation is that, if the data collection were repeated (on women sampled the same way as in the original sample), and if we were to refit the model, 95% of the time we would expect the estimated coefficient on tenure to be in the range [.036, .042].

Even with robust regression, we must be careful about going beyond that statement. Here the Hausman test is probably picking up something that differs within and between person, which would cast doubt on our robust regression model in terms of interpreting [.036, .042] to contain the rate of return for keeping a job, economywide, for all women, without exception.

## Weighted regression

`regress` can perform weighted and unweighted regression. We indicate the weight by specifying the `[weight]` qualifier. By default, `regress` assumes analytic weights; see the technical note below.

### ▷ Example 7

We have census data recording the death rate (`drate`) and median age (`medage`) for each state. The data also record the region of the country in which each state is located and the overall population of the state:

```
. use http://www.stata-press.com/data/r10/census9
(1980 Census data by state)
. describe
Contains data from http://www.stata-press.com/data/r10/census9.dta
  obs:                50                1980 Census data by state
  vars:                5                13 Jan 2007 22:03
  size:                1,550 (99.9% of memory free)
```

variable name	storage type	display format	value label	variable label
state	str14	%-14s		State
drate	float	%9.0g		Death Rate
pop	long	%12.0gc		Population
medage	float	%9.2f		Median age
region	byte	%-8.0g	cenreg	Census region

Sorted by:

We can use the `xi` command to automatically create and include dummy variables for region. Since the variables in the regression reflect means rather than individual observations, the appropriate method of estimation is analytically weighted least squares (Johnston and DiNardo 1997), where the weight is total population:

```
. xi: regress drate medage I.region [w=pop]
I.region      _Iregion_1-4      (naturally coded; _Iregion_1 omitted)
(analytic weights assumed)
(sum of wgt is  2.2591e+08)
```

Source	SS	df	MS	Number of obs = 50		
Model	4096.6093	4	1024.15232	F( 4, 45) =	37.21	
Residual	1238.40987	45	27.5202192	Prob > F =	0.0000	
Total	5335.01916	49	108.877942	R-squared =	0.7679	
				Adj R-squared =	0.7472	
				Root MSE =	5.246	

  

drate	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
medage	4.283183	.5393329	7.94	0.000	3.196911	5.369455
_Iregion_2	.3138738	2.456431	0.13	0.899	-4.633632	5.26138
_Iregion_3	-1.438452	2.320244	-0.62	0.538	-6.111663	3.234758
_Iregion_4	-10.90629	2.681349	-4.07	0.000	-16.30681	-5.505777
_cons	-39.14727	17.23613	-2.27	0.028	-73.86262	-4.431915

To weight the regression by population, we added the qualifier `[w=pop]` to the end of the `regress` command. Our qualifier was vague (we did not say `[aweight=pop]`), but unless told otherwise, Stata assumes analytic weights for `regress`. Stata informed us that the sum of the weight is  $2.2591 \cdot 10^8$ ; there were approximately 226 million people residing in the United States according to our 1980 data.

`xi` provides one way to include dummy variables and can be used with any estimation command. In the special case of linear regression, another alternative would be to use `anova` with the `regress` option. This would probably be better, but only because `anova` has special logic to fit models with many dummy variables, which uses less memory and computer time.

◀

### □ Technical Note

Once we fit a weighted regression, we can obtain the appropriately weighted variance–covariance matrix of the estimators using `estat vce` and perform appropriately weighted hypothesis tests using `test`.

In the weighted regression in the previous example, we see that `_Iregion_4` is statistically significant but that `_Iregion_2` and `_Iregion_3` are not. We use `test` to test the joint significance of the region variables:

```
. test _Iregion_2 _Iregion_3 _Iregion_4
( 1)  _Iregion_2 = 0
( 2)  _Iregion_3 = 0
( 3)  _Iregion_4 = 0
      F( 3, 45) = 9.84
      Prob > F = 0.0000
```

The results indicate that the region variables are jointly significant.

□

`regress` also accepts frequency weights (`fweights`). Frequency weights are appropriate when the data do not reflect cell means, but instead represent replicated observations. Specifying `aweight` or `fweights` will not change the parameter estimates, but it will change the corresponding significance levels.

For instance, if we specified [`fweight=pop`] in the weighted regression example above—which would be statistically incorrect—Stata would treat the data as if the data represented 226 million independent observations on death rates and median age. The data most certainly do not represent that—they represent 50 observations on state averages.

With `aweight`, Stata treats the number of observations on the process as the number of observations in the data. When we specify `fweights`, Stata treats the number of observations as if it were equal to the sum of the weights; see *Methods and Formulas* below.

### □ Technical Note

A popular request on the help line is to describe the effect of specifying [`aweight=exp`] with `regress` in terms of transformation of the dependent and independent variables. The mechanical answer is that typing

```
. regress y x1 x2 [aweight=n]
```

is equivalent to fitting the model

$$y_j \sqrt{n_j} = \beta_0 \sqrt{n_j} + \beta_1 x_{1j} \sqrt{n_j} + \beta_2 x_{2j} \sqrt{n_j} + u_j \sqrt{n_j}$$

This regression will reproduce the coefficients and covariance matrix produced by the `aweight`d regression. The mean squared errors (estimates of the variance of the residuals) will, however, be different. The transformed regression reports  $s_t^2$ , an estimate of  $\text{Var}(u_j \sqrt{n_j})$ . The `aweight`d regression reports  $s_a^2$ , an estimate of  $\text{Var}(u_j \sqrt{n_j} \sqrt{N / \sum_k n_k})$ , where  $N$  is the number of observations. Thus

$$s_a^2 = \frac{N}{\sum_k n_k} s_t^2 = \frac{s_t^2}{\bar{n}} \quad (1)$$

The logic for this adjustment is as follows: consider the model

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + u$$

Assume that, were this model fitted on individuals,  $\text{Var}(u) = \sigma_u^2$ , a constant. Assume that individual data are not available; what is available are averages  $(\bar{y}_j, \bar{x}_{1j}, \bar{x}_{2j})$  for  $j = 1, \dots, N$ , and each average is calculated over  $n_j$  observations. Then it is still true that

$$\bar{y}_j = \beta_0 + \beta_1 \bar{x}_{1j} + \beta_2 \bar{x}_{2j} + \bar{u}_j$$

where  $\bar{u}_j$  is the average of  $n_j$  mean 0, variance  $\sigma_u^2$  deviates and has variance  $\sigma_{\bar{u}}^2 = \sigma_u^2/n_j$ . Thus multiplying through by  $\sqrt{n_j}$  produces

$$\bar{y}_j \sqrt{n_j} = \beta_0 \sqrt{n_j} + \beta_1 \bar{x}_{1j} \sqrt{n_j} + \beta_2 \bar{x}_{2j} \sqrt{n_j} + \bar{u}_j \sqrt{n_j}$$

and  $\text{Var}(\bar{u}_j \sqrt{n_j}) = \sigma_u^2$ . The mean squared error,  $s_t^2$ , reported by fitting this transformed regression is an estimate of  $\sigma_u^2$ . The coefficients and covariance matrix could also be obtained by **aweighted regress**. The only difference would be in the reported mean squared error, which from (1) is  $\sigma_u^2/\bar{n}$ . On average, each observation in the data reflects the averages calculated over  $\bar{n} = \sum_k n_k/N$  individuals, and thus this reported mean squared error is the average variance of an observation in the dataset. We can retrieve the estimate of  $\sigma_u^2$  by multiplying the reported mean squared error by  $\bar{n}$ .

More generally, **aweights** are used to solve general heteroskedasticity problems. In these cases, we have the model

$$y_j = \beta_0 + \beta_1 x_{1j} + \beta_2 x_{2j} + u_j$$

and the variance of  $u_j$  is thought to be proportional to  $a_j$ . If the variance is proportional to  $a_j$ , it is also proportional to  $\alpha a_j$ , where  $\alpha$  is any positive constant. Not quite arbitrarily, but with no loss of generality, we could choose  $\alpha = \sum_k (1/a_k)/N$ , the average value of the inverse of  $a_j$ . We can then write  $\text{Var}(u_j) = k \alpha a_j \sigma^2$ , where  $k$  is the constant of proportionality that is no longer a function of the scale of the weights.

Dividing this regression through by the  $\sqrt{a_j}$ ,

$$y_j / \sqrt{a_j} = \beta_0 / \sqrt{a_j} + \beta_1 x_{1j} / \sqrt{a_j} + \beta_2 x_{2j} / \sqrt{a_j} + u_j / \sqrt{a_j}$$

produces a model with  $\text{Var}(u_j / \sqrt{a_j}) = k \alpha \sigma^2$ , which is the constant part of  $\text{Var}(u_j)$ . This variance is a function of  $\alpha$ , the average of the reciprocal weights; if the weights are scaled arbitrarily, then so is this variance.

We can also fit this model by typing

```
. regress y x1 x2 [aweight=1/a]
```

This input will produce the same estimates of the coefficients and covariance matrix; the reported mean squared error is, from (1),  $\{N/\sum_k (1/a_k)\} k \alpha \sigma^2 = k \sigma^2$ . This variance is independent of the scale of  $a_j$ .

□

## Instrumental variables and two-stage least-squares regression

An alternate syntax for `regress` can be used to produce instrumental variables (two-stage least squares) estimates.

```
regress depvar [varlist1 [(varlist2)]] [if] [in] [weight] [, regress_options]
```

This syntax is used mainly by programmers developing estimators using the instrumental variables estimates as intermediate results. `ivregress` is normally used to directly fit these models; see [R] `ivregress`.

With this syntax, `regress` fits a structural equation of `depvar` on `varlist1` using instrumental variables regression; `(varlist2)` indicates the list of instrumental variables. With the exception of `vce(hc2)` and `vce(hc3)`, all standard `regress` options are allowed.

## Saved Results

`regress` saves the following in `e()`:

### Scalars

<code>e(N)</code>	number of observations	<code>e(F)</code>	$F$ statistic
<code>e(mss)</code>	model sum of squares	<code>e(rmse)</code>	root mean squared error
<code>e(df_m)</code>	model degrees of freedom	<code>e(ll)</code>	log likelihood under additional assumption of i.i.d. normal errors
<code>e(rss)</code>	residual sum of squares	<code>e(ll_0)</code>	log likelihood, constant-only model
<code>e(df_r)</code>	residual degrees of freedom	<code>e(N_clust)</code>	number of clusters
<code>e(r2)</code>	$R$ -squared		
<code>e(r2_a)</code>	adjusted $R$ -squared		

### Macros

<code>e(cmd)</code>	<code>regress</code>	<code>e(clustvar)</code>	name of cluster variable
<code>e(cmdline)</code>	command as typed	<code>e(vce)</code>	<code>vcetype</code> specified in <code>vce()</code>
<code>e(depvar)</code>	name of dependent variable	<code>e(vcetype)</code>	title used to label Std. Err.
<code>e(model)</code>	ols or iv	<code>e(properties)</code>	<code>b V</code>
<code>e(wtype)</code>	weight type	<code>e(estat_cmd)</code>	program used to implement <code>estat</code>
<code>e(wexp)</code>	weight expression	<code>e(predict)</code>	program used to implement <code>predict</code>
<code>e(title)</code>	title in estimation output when <code>vce()</code> is not <code>ols</code>		

### Matrices

<code>e(b)</code>	coefficient vector	<code>e(V)</code>	variance-covariance matrix of the estimators
-------------------	--------------------	-------------------	--

### Functions

<code>e(sample)</code>	marks estimation sample
------------------------	-------------------------

## Methods and Formulas

Variables printed in lowercase and not boldfaced (e.g.,  $x$ ) are scalars. Variables printed in lowercase and boldfaced (e.g.,  $\mathbf{x}$ ) are column vectors. Variables printed in uppercase and boldfaced (e.g.,  $\mathbf{X}$ ) are matrices.

Let  $\mathbf{v}$  be a column vector of weights specified by the user. If no weights are specified,  $\mathbf{v} = \mathbf{1}$ . Let  $\mathbf{w}$  be a column vector of normalized weights. If no weights are specified or if the user specified `fweights` or `iwweights`,  $\mathbf{w} = \{\mathbf{v}/(\mathbf{1}'\mathbf{v})\}(\mathbf{1}'\mathbf{1})$ .

The *number of observations*,  $n$ , is defined as  $\mathbf{1}'\mathbf{w}$ . For `weights`, this is truncated to an integer. The *sum of the weights* is  $\mathbf{1}'\mathbf{v}$ . Define  $c = 1$  if there is a constant in the regression and zero otherwise. Define  $k$  as the number of RHS variables (including the constant).

Let  $\mathbf{X}$  denote the matrix of observations on the RHS variables,  $\mathbf{y}$  the vector of observations on the left-hand-side variable, and  $\mathbf{Z}$  the matrix of observations on the instruments. If the user specifies no instruments, then  $\mathbf{Z} = \mathbf{X}$ . In the following formulas, if the user specifies weights, then  $\mathbf{X}'\mathbf{X}$ ,  $\mathbf{X}'\mathbf{y}$ ,  $\mathbf{y}'\mathbf{y}$ ,  $\mathbf{Z}'\mathbf{Z}$ ,  $\mathbf{Z}'\mathbf{X}$ , and  $\mathbf{Z}'\mathbf{y}$  are replaced by  $\mathbf{X}'\mathbf{D}\mathbf{X}$ ,  $\mathbf{X}'\mathbf{D}\mathbf{y}$ ,  $\mathbf{y}'\mathbf{D}\mathbf{y}$ ,  $\mathbf{Z}'\mathbf{D}\mathbf{Z}$ ,  $\mathbf{Z}'\mathbf{D}\mathbf{X}$ , and  $\mathbf{Z}'\mathbf{D}\mathbf{y}$ , respectively, where  $\mathbf{D}$  is a diagonal matrix whose diagonal elements are the elements of  $\mathbf{w}$ . We suppress the  $\mathbf{D}$  below to simplify the notation.

If no instruments are specified, define  $\mathbf{A}$  as  $\mathbf{X}'\mathbf{X}$  and  $\mathbf{a}$  as  $\mathbf{X}'\mathbf{y}$ . Otherwise, define  $\mathbf{A}$  as  $\mathbf{X}'\mathbf{Z}(\mathbf{Z}'\mathbf{Z})^{-1}(\mathbf{X}'\mathbf{Z})'$  and  $\mathbf{a}$  as  $\mathbf{X}'\mathbf{Z}(\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{y}$ .

The coefficient vector  $\mathbf{b}$  is defined as  $\mathbf{A}^{-1}\mathbf{a}$ . Although not shown in the notation, unless `hascons` is specified,  $\mathbf{A}$  and  $\mathbf{a}$  are accumulated in deviation form and the constant is calculated separately. This comment applies to all statistics listed below.

The *total sum of squares*, TSS, equals  $\mathbf{y}'\mathbf{y}$  if there is no intercept and  $\mathbf{y}'\mathbf{y} - \{(\mathbf{1}'\mathbf{y})^2/n\}$  otherwise. The *degrees of freedom* are  $n - c$ .

The *error sum of squares*, ESS, is defined as  $\mathbf{y}'\mathbf{y} - 2\mathbf{b}'\mathbf{X}'\mathbf{y} + \mathbf{b}'\mathbf{X}'\mathbf{X}\mathbf{b}$  if there are instruments and as  $\mathbf{y}'\mathbf{y} - \mathbf{b}'\mathbf{X}'\mathbf{y}$  otherwise. The *degrees of freedom* are  $n - k$ .

The *model sum of squares*, MSS, equals TSS - ESS. The *degrees of freedom* are  $k - c$ .

The *mean squared error*,  $s^2$ , is defined as ESS/( $n - k$ ). The *root mean squared error* is  $s$ , its square root.

The  $F$  statistic with  $k - c$  and  $n - k$  degrees of freedom is defined as

$$F = \frac{\text{MSS}}{(k - c)s^2}$$

if no instruments are specified. If instruments are specified and  $c = 1$ , then  $F$  is defined as

$$F = \frac{(\mathbf{b} - \mathbf{c})'\mathbf{A}(\mathbf{b} - \mathbf{c})}{(k - 1)s^2}$$

where  $\mathbf{c}$  is a vector of  $k - 1$  zeros and  $k$ th element  $\mathbf{1}'\mathbf{y}/n$ . Otherwise,  $F$  is defined as *missing*. (Here you may use the `test` command to construct any  $F$  test that you wish.)

The *R-squared*,  $R^2$ , is defined as  $R^2 = 1 - \text{ESS}/\text{TSS}$ .

The *adjusted R-squared*,  $R_a^2$ , is  $1 - (1 - R^2)(n - c)/(n - k)$ .

If `vce(robust)` is not specified, the conventional estimate of variance is  $s^2\mathbf{A}^{-1}$ . The handling of `vce(robust)` is described below.

### A general notation for the robust variance calculation

Put aside all context of linear regression and the notation that goes with it—we will return to it. First, we are going to establish a notation for describing robust variance calculations.

The calculation formula for the robust variance calculation is

$$\hat{\mathbf{V}} = q_c \hat{\mathbf{V}} \left( \sum_{k=1}^M \mathbf{u}_k^{(G)'} \mathbf{u}_k^{(G)} \right) \hat{\mathbf{V}}$$

where

$$\mathbf{u}_k^{(G)} = \sum_{j \in G_k} w_j \mathbf{u}_j$$

$G_1, G_2, \dots, G_M$  are the clusters specified by `vce(cluster clustvar)`, and  $w_j$  are the user-specified weights, normalized if `aweights` or `pweights` are specified and equal to 1 if no weights are specified.

For `fweights` without clusters, the variance formula is

$$\widehat{V} = q_c \widehat{V} \left( \sum_{j=1}^N w_j \mathbf{u}'_j \mathbf{u}_j \right) \widehat{V}$$

which is the same as expanding the dataset and making the calculation on the unweighted data.

If `vce(cluster clustvar)` is not specified,  $M = N$ , and each cluster contains 1 observation. The inputs into this calculation are

1.  $\widehat{V}$ , which is typically a conventionally calculated variance matrix;
2.  $\mathbf{u}_j$ ,  $j = 1, \dots, N$ , a row vector of scores; and
3.  $q_c$ , a constant finite-sample adjustment.

Thus we can now describe how estimators apply the robust calculation formula by defining  $\widehat{V}$ ,  $\mathbf{u}_j$ , and  $q_c$ .

Two definitions are popular enough for  $q_c$  to deserve a name. The regression-like formula for  $q_c$  (Fuller et al. 1986) is

$$q_c = \frac{N-1}{N-k} \frac{M}{M-1}$$

where  $M$  is the number of clusters and  $N$  is the number of observations. For weights,  $N$  refers to the sum of the weights if weights are frequency weights and the number of observations in the dataset (ignoring weights) in all other cases. Also note that, weighted or not,  $M = N$  when `vce(cluster clustvar)` is not specified, and, then  $q_c = N/(N-k)$ .

The asymptotic-like formula for  $q_c$  is

$$q_c = \frac{M}{M-1}$$

where  $M = N$  if `vce(cluster clustvar)` is not specified.

See [U] **20.15 Obtaining robust variance estimates** and [P] `_robust` for a discussion of the robust variance estimator and a development of these formulas.

### Robust calculation for regress

For `regress`,  $\widehat{V} = \mathbf{A}^{-1}$ . The other terms are

No instruments, `vce(robust)`, but not `vce(hc2)` or `vce(hc3)`,

$$\mathbf{u}_j = (y_j - \mathbf{x}_j \mathbf{b}) \mathbf{x}_j$$

and  $q_c$  is given by its regression-like definition.

No instruments, `vce(hc2)`,

$$\mathbf{u}_j = \frac{1}{\sqrt{1 - h_{jj}}}(y_j - \mathbf{x}_j\mathbf{b})\mathbf{x}_j$$

where  $q_c = 1$  and  $h_{jj} = \mathbf{x}_j(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_j'$ .

No instruments, `vce(hc3)`,

$$\mathbf{u}_j = \frac{1}{1 - h_{jj}}(y_j - \mathbf{x}_j\mathbf{b})\mathbf{x}_j$$

where  $q_c = 1$  and  $h_{jj} = \mathbf{x}_j(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_j'$ .

Instrumental variables,

$$\mathbf{u}_j = (y_j - \mathbf{x}_j\mathbf{b})\widehat{\mathbf{x}}_j$$

where  $q_c$  is given by its regression-like definition, and

$$\widehat{\mathbf{x}}_j = \mathbf{P}\mathbf{z}_j'$$

where  $\mathbf{P} = (\mathbf{X}'\mathbf{Z})(\mathbf{Z}'\mathbf{Z})^{-1}$ .

## Acknowledgments

The robust estimate of variance was first implemented in Stata by Mead Over, World Bank; Dean Jolliffe, U.S. Department of Agriculture; and Andrew Foster, Department of Economics, Brown University (Over, Jolliffe, and Foster 1996).

## References

- Alexanderson, A. 1998. `gr32`: Confidence ellipses. *Stata Technical Bulletin* 46: 10–13. Reprinted in *Stata Technical Bulletin Reprints*, vol. 8, pp. 54–57.
- Chatterjee, S., and A. S. Hadi. 2006. *Regression Analysis by Example*. 4th ed. New York: Wiley.
- Davidson, R., and J. G. MacKinnon. 1993. *Estimation and Inference in Econometrics*. New York: Oxford University Press.
- Dohoo, I., W. Martin, and H. Stryhn. 2003. *Veterinary Epidemiologic Research*. Charlottetown, Prince Edward Island: AVC.
- Draper, N., and H. Smith. 1998. *Applied Regression Analysis*. 3rd ed. New York: Wiley.
- Dunnington, G. W. 1955. *Gauss: Titan of Science*. New York: Hafner Publishing.
- Fuller, W. A., W. J. Kennedy, Jr., D. Schnell, G. Sullivan, H. J. Park. 1986. *PC Carp*. Ames, IA: Iowa State University.
- Gillham, N. W. 2001. *A Life of Sir Francis Galton: From African Exploration to the Birth of Eugenics*. New York: Oxford University Press.
- Gillispie, C. C. 1997. *Pierre-Simon Laplace, 1749–1827: A Life in Exact Science*. Princeton: Princeton University Press.
- Hald, A. 1998. *A History of Mathematical Statistics from 1750 to 1930*. New York: Wiley.
- Hamilton, L. C. 2006. *Statistics with Stata (Updated for Version 9)*. Belmont, CA: Duxbury.
- Johnston, J., and J. DiNardo. 1997. *Econometric Methods*. 4th ed. New York: McGraw–Hill.

- Kmenta, J. 1997. *Elements of Econometrics*. 2nd ed. Ann Arbor: University of Michigan Press.
- Kohler, U., and F. Kreuter. 2005. *Data Analysis Using Stata*. College Station, TX: Stata Press.
- Long, J. S., and J. Freese. 2000. sg152: Listing and interpreting transformed coefficients from certain regression models. *Stata Technical Bulletin* 57: 27–34. Reprinted in *Stata Technical Bulletin Reprints*, vol. 10, pp. 231–240.
- MacKinnon, J. G., and H. White. 1985. Some heteroskedasticity consistent covariance matrix estimators with improved finite sample properties. *Journal of Econometrics* 29: 305–325.
- Mosteller, F., and J. W. Tukey. 1977. *Data Analysis and Regression: A Second Course in Statistics*. Reading, MA: Addison–Wesley.
- Over, M., D. Jolliffe, and A. Foster. 1996. sg46: Huber correction for two-stage least-squares estimates. *Stata Technical Bulletin* 29: 24–25. Reprinted in *Stata Technical Bulletin Reprints*, vol. 5, pp. 140–142.
- Peracchi, F. 2001. *Econometrics*. Chichester, UK: Wiley.
- Plackett, R. L. 1972. The discovery of the method of least squares. *Biometrika* 59: 239–251.
- Rogers, W. H. 1991. smv2: Analyzing repeated measurements—some practical alternatives. *Stata Technical Bulletin* 4: 10–16. Reprinted in *Stata Technical Bulletin Reprints*, vol. 1, pp. 123–131.
- Royston, J. P., and G. Ambler. 1998. sg79: Generalized additive models. *Stata Technical Bulletin* 42: 38–43. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, pp. 217–224.
- Schonlau, M. 2005. Boosted regression (boosting): An introductory tutorial and a Stata plugin. *Stata Journal* 5: 330–354.
- Stigler, S. M. 1986. *The History of Statistics: The Measurement of Uncertainty before 1900*. Cambridge, MA: Belknap Press.
- Tyler, J. H. 1997. sg73: Table making program. *Stata Technical Bulletin* 40: 18–23. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, pp. 186–192.
- Weesie, J. 1998. sg77: Regression analysis with multiplicative heteroskedasticity. *Stata Technical Bulletin* 42: 28–32. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, pp. 204–210.
- Weisberg, S. 2005. *Applied Linear Regression*. 3rd ed. New York: Wiley.
- Wooldridge, J. M. 2002. *Econometric Analysis of Cross Section and Panel Data*. Cambridge, MA: MIT Press.
- . 2006. *Introductory Econometrics: A Modern Approach*. 3rd ed. Cincinnati, OH: South-Western.
- Zimmerman, F. 1998. sg93: Switching regressions. *Stata Technical Bulletin* 45: 30–33. Reprinted in *Stata Technical Bulletin Reprints*, vol. 8, pp. 183–186.

## Also See

- [R] **regress postestimation** — Postestimation tools for regress
- [R] **regress postestimation time series** — Postestimation tools for regress with time series
- [R] **anova** — Analysis of variance and covariance
- [U] **20 Estimation and postestimation commands**

The history of regression is long and complicated: the books by Stigler (1986) and Hald (1998) are devoted largely to the story. Legendre published first on least squares in 1805. Gauss published later in 1809, but he had the idea earlier. Gauss, and especially Laplace, tied least squares to a normal errors assumption. The idea of the normal distribution can itself be traced back to De Moivre in 1733. Laplace discussed a variety of other estimation methods and error assumptions over his long career, while linear models long predate either innovation. Most of this work was linked to problems in astronomy and geodesy.

A second wave of ideas started when Galton used graphical and descriptive methods on data bearing on heredity to develop what he called regression. His term reflects the common phenomenon that characteristics of offspring are positively correlated with those of parents but with regression slope such that offspring “regress toward the mean”. Galton’s work was rather intuitive: contributions from Pearson, Edgeworth, Yule, and others introduced more formal machinery, developed related ideas on correlation, and extended application into the biological and social sciences. So most of the elements of regression as we know it were in place by 1900.

Pierre-Simon Laplace (1749–1827) was born in Normandy and was early recognized as a remarkable mathematician. He weathered a changing political climate well enough to rise to Minister of the Interior under Napoleon in 1799 (although only for 6 weeks) and to be made a Marquis by Louis XVIII in 1817. He made many contributions to mathematics and physics, his two main interests being theoretical astronomy and probability theory (including statistics). Laplace transforms are named for him.

Adrien-Marie Legendre (1752–1833) was born in Paris (or possibly in Toulouse) and educated in mathematics and physics. He worked in number theory, geometry, differential equations, calculus, function theory, applied mathematics, and geodesy. The Legendre polynomials are named for him. His main contribution to statistics is as one of the discoverers of least squares. He died in poverty, having refused to bow to political pressures.

Johann Carl Friedrich Gauss (1777–1855) was born in Braunschweig (Brunswick), now in Germany. He studied there and at Göttingen. His doctoral dissertation at the University of Helmstedt was a discussion of the fundamental theorem of algebra. He made many fundamental contributions to geometry, number theory, algebra, real analysis, differential equations, numerical analysis, statistics, astronomy, optics, geodesy, mechanics, and magnetism. An outstanding genius, Gauss worked mostly in isolation in Göttingen.

Francis Galton (1822–1911) was born in Birmingham, England, into a well-to-do family with many connections: he and Charles Darwin were first cousins. After an unsuccessful foray into medicine, he became independently wealthy at the death of his father. Galton traveled widely in Europe, the Middle East, and Africa, and became celebrated as an explorer and geographer. His pioneering work on weather maps helped in the identification of anticyclones, which he named. From about 1865, most of his work was centered on quantitative problems in biology, anthropology, and psychology. In a sense, Galton (re)invented regression, and he certainly named it. Galton also promoted the normal distribution, correlation approaches, and the use of median and selected quantiles as descriptive statistics. He was knighted in 1909.