

## Stata FAQ

### How can I perform the likelihood ratio, Wald, and Lagrange multiplier (score) test in Stata?

The likelihood ratio (lr) test, Wald test, and Lagrange multiplier test (sometimes called a score test) are commonly used to evaluate the difference between nested models. One model is considered nested in another if the first model can be generated by imposing restrictions on the parameters of the second. Most often, the restriction is that the parameter is equal to zero. In a regression model restricting a parameter to zero is accomplished by removing the predictor variables from the model. For example, in the models below, the model with the predictor variables **female**, and **read**, is nested within the model with the predictor variables **female**, **read**, **math**, and **science**. The lr, Wald, and Lagrange multiplier tests ask the same basic question, which is, does constraining these parameters to zero (i.e., leaving out these predictor variables) significantly reduce the fit of the model? To perform a likelihood ratio test, one must estimate both of the models one wishes to compare. The advantage of the Wald and score tests is that they approximate the lr test but require that only one model be estimated. When computing power was much more limited, and many models took a long time to run, this was a fairly major advantage. Today, for most of the models researchers are likely to want to compare, this is not an issue, and we generally recommend running the likelihood ratio test in most situations. This is not to say that one should never use the Wald or score tests. For example, the Wald test is commonly used to perform multiple degree of freedom tests on sets of dummy variables used to model categorical variables in regression (for more information see our webbook on Regression with Stata, specifically [Chapter 3 - Regression with Categorical Predictors](#)). Another example is the "modification indices" used in structural equation modeling, they are Lagrange multiplier tests.

As we mentioned above, the lr test requires that two models be run, one of which has a set of parameters (variables), and a second model with all of the parameters from the first, plus one or more other variables. The Wald test examines a model with more parameters and assesses whether restricting those parameters (generally to zero, by removing the associated variables from the model) seriously harms the fit of the model. In contrast, the score test examines the results of a smaller model and asks whether adding one or more omitted variables would improve the fit of the model. In general, the three tests should come to the same conclusion (because the Wald and score test, at least in theory, approximate the lr test). As an example, we will test for a statistically significant difference between two models, using all three tests.

The dataset for this example includes demographic data, as well as standardized test scores for 200 high school students. We will compare two models. The dependent variable for both models is **hiwrite** (to be nested two models must share the same dependent variable), which is a dichotomous variable indicating that the student had a writing score that was above the mean. There are four possible predictor variables, **female**, a dummy variable which indicates that the student is female, and the continuous variables **read**, **math**, and **science**, which are the student's standardized test scores in reading, math, and science, respectively. We will test a model containing just the predictor variables **female** and **read**, against a model that contains the predictor variables **female** and **read**, as well as, the additional predictor variables, **math** and **science**.

#### Example of a likelihood ratio test.

As discussed above, the lr test involves estimating two models and comparing them. Fixing one or more parameters to zero, by removing the variables associated with that parameter from the model, will almost always make the model fit less well, so a change in the log likelihood does not necessarily mean the model with more variables fits significantly better. The lr test compares the log likelihoods of the two models and tests whether this difference is statistically significant. If the difference is statistically significant, then the less restrictive model (the one with more variables) is said to fit the data significantly better than the more restrictive model. The lr test statistic is calculated in the following way:

$$LR = -2 \ln(L(m1)/L(m2)) = 2(\ln(m2) - \ln(m1))$$

Where  $L(m^*)$  denotes the likelihood of the respective model, and  $\ln(m^*)$  the natural log of the models' likelihood.

This statistic is distributed chi-squared with degrees of freedom equal to the difference in the number of degrees

of freedom between the two models (i.e., the number of variables added to the model).

In order to perform the likelihood ratio test we will need to run both models and make note of their final log likelihoods. We will run the models using Stata and use commands to store the log likelihoods. We could also just copy the likelihoods down (i.e., by writing them down, or cutting and pasting), but using commands is a little easier and is less likely to result in errors. The first line of syntax below reads in the dataset from our website. The second line of syntax runs a logistic regression model, predicting **hiwrite** based on students' gender (**female**), and reading scores (**read**). The third line of code stores the value of the log likelihood for the model, which is temporarily stored as the returned estimate **e(ll)** (for more information type **help return** in the Stata command window), in the scalar named **m1**.

```
use http://www.ats.ucla.edu/stat/stata/faq/nested_tests, clear
logit hiwrite female read
scalar m1 = e(ll)
```

Below is the output. In order to perform the likelihood ratio test we will need to keep track of the log likelihood (-102.44), the syntax for this example (above) does this by storing the value in a scalar. Since it is not our primary concern here, we will skip the interpretation of the rest logistic regression model. Note that storing the returned estimate does not produce any output.

```
Iteration 0:   log likelihood = -137.41698
Iteration 1:   log likelihood = -104.79885
Iteration 2:   log likelihood = -102.52269
Iteration 3:   log likelihood = -102.44531
Iteration 4:   log likelihood = -102.44518
```

```
Logistic regression               Number of obs   =          200
                                LR chi2(2)         =          69.94
                                Prob > chi2         =          0.0000
Log likelihood = -102.44518       Pseudo R2       =          0.2545
```

	hiwrite	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
	female	1.403022	.3671964	3.82	0.000	.6833301 2.122713
	read	.1411402	.0224042	6.30	0.000	.0972287 .1850517
	_cons	-7.798179	1.235685	-6.31	0.000	-10.22008 -5.376281

The first line of syntax below runs the second model, that is, the model with all four predictor variables. The second line of code stores the value of the log likelihood for the model (-84.4), which is temporarily stored as the returned estimate ( **e(ll)** ), in the scalar named **m2**. Again, we won't say much about the output except to note that the coefficients for both **math** and **science** are both statistically significant. So we know that, individually, they are statistically significant predictors of **hiwrite**.

```
logit hiwrite female read math science
scalar m2 = e(ll)
```

```
Iteration 0:   log likelihood = -137.41698
Iteration 1:   log likelihood = -90.166892
Iteration 2:   log likelihood = -84.909776
Iteration 3:   log likelihood = -84.42653
Iteration 4:   log likelihood = -84.419844
Iteration 5:   log likelihood = -84.419842
```

```
Logistic regression               Number of obs   =          200
                                LR chi2(4)         =         105.99
                                Prob > chi2         =          0.0000
Log likelihood = -84.419842       Pseudo R2       =          0.3857
```

	hiwrite	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
	female	1.805528	.4358101	4.14	0.000	.9513555 2.6597
	read	.0529536	.0275925	1.92	0.055	-.0011268 .107034

math		.1319787	.0318836	4.14	0.000	.069488	.1944694
science		.0577623	.027586	2.09	0.036	.0036947	.1118299
_cons		-13.26097	1.893801	-7.00	0.000	-16.97275	-9.549188

---

Now that we have the log likelihoods from both models, we can perform a likelihood ratio test. The first line of syntax below calculates the likelihood ratio test statistic. The second line of syntax below finds the p-value associated with our test statistic with two degrees of freedom. Looking below we see that the test statistic is 36.05, and that the associated p-value is very low (less than 0.0001). The results show that adding **math** and **science** as predictor variables together (not just individually) results in a statistically significant improvement in model fit. Note that if we performed a likelihood ratio test for adding a single variable to the model, the results would be the same as the significance test for the coefficient for that variable presented in the above table.

```
di "chi2(2) = " 2*(m2-m1)
di "Prob > chi2 = "chi2tail(2, 2*(m2-m1))

chi2(2) = 36.050677
Prob > chi2 = 1.485e-08
```

### Using Stata's postestimation commands to calculate a likelihood ratio test

As you have seen, it is easy enough to calculate a likelihood ratio test "by hand." However, you can also use Stata to store the estimates and run the test for you. This method is easier still, and probably less error prone. The first line of syntax runs a logistic regression model, predicting **hiwwrite** based on students' gender (**female**), and reading scores (**read**). The second line of syntax asks Stata to store the estimates from the model we just ran, and instructs Stata that we want to call the estimates **m1**. It is necessary to give the estimates a name, since Stata allows users to store the estimates from more than one analysis, and we will be storing more than one set of estimates.

```
use http://www.ats.ucla.edu/stat/stata/faq/nested_tests, clear
logit hiwwrite female read
estimates store m1
```

Below is the output. Since it is not our primary concern here, we will skip the interpretation of the logistic regression model. Note that storing the estimates does not produce any output.

```
Iteration 0:  log likelihood = -137.41698-137.41698-137.41698
Iteration 1:  log likelihood = -104.79885
Iteration 2:  log likelihood = -102.52269
Iteration 3:  log likelihood = -102.44531
Iteration 4:  log likelihood = -102.44518

Logistic regression                                Number of obs   =           200
                                                    LR chi2(2)      =           69.94
                                                    Prob > chi2     =           0.0000
Log likelihood = -102.44518                        Pseudo R2      =           0.2545
```

---

	hiwwrite		Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
	female		1.403022	.3671964	3.82	0.000	.6833301 2.122713
	read		.1411402	.0224042	6.30	0.000	.0972287 .1850517
	_cons		-7.798179	1.235685	-6.31	0.000	-10.22008 -5.376281

---

The first line of syntax below this paragraph runs the second model, that is the model with all four predictor variables. The second line of syntax saves the estimates from this model, and names them **m2**. Below the syntax is the output generated. Again, we won't say much about the output except to note that the coefficients for both **math** and **science** are both statistically significant. So we know that, individually, they are statistically significant predictors of **hiwwrite**. The tests below will allow us to test whether adding both of these variables to the model significantly improves the fit of the model, compared to a model that contains just **female** and **read**.

```
logit hiwwrite female read math science
```

**estimates store m2**

```

Iteration 0:  log likelihood = -137.41698
Iteration 1:  log likelihood = -90.166892
Iteration 2:  log likelihood = -84.909776
Iteration 3:  log likelihood = -84.42653
Iteration 4:  log likelihood = -84.419844
Iteration 5:  log likelihood = -84.419842

```

```

Logistic regression                                Number of obs   =          200
                                                    LR chi2(4)      =        105.99
                                                    Prob > chi2     =         0.0000
Log likelihood = -84.419842                        Pseudo R2      =         0.3857

```

hiwrite	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
female	1.805528	.4358101	4.14	0.000	.9513555	2.6597
read	.0529536	.0275925	1.92	0.055	-.0011268	.107034
math	.1319787	.0318836	4.14	0.000	.069488	.1944694
science	.0577623	.027586	2.09	0.036	.0036947	.1118299
_cons	-13.26097	1.893801	-7.00	0.000	-16.97275	-9.549188

The first line of syntax below tells Stata that we want to run an lr test, and that we want to compare the estimates we have saved as **m1** to those we have saved as **m2**. The output reminds us that this test assumes that A is nested in B, which it is. It also gives us the chi-squared value for the test (36.05) as well as the p-value for a chi-squared of 36.05 with two degrees of freedom. Note that the degrees of freedom for the lr test, along with the other two tests, is equal to the number of parameters that are constrained (i.e., removed from the model), in our case, 2. Note that the results are the same as when we calculated the lr test by hand above. Adding **math** and **science** as predictor variables together (not just individually) results in a statistically significant improvement in model fit. As noted when we calculated the likelihood ratio test by hand, if we performed a likelihood ratio test for adding a single variable to the model, the results would be the same as the significance test for the coefficient for that variable presented in the table above.

**lrtest m1 m2**

```

Likelihood-ratio test                                LR chi2(2)  =        36.05
(Assumption: A nested in B)                        Prob > chi2 =         0.0000

```

The entire syntax for a likelihood ratio test, all in one block, looks like this:

```

logit hiwrite female read
estimates store m1
logit hiwrite female read math science
estimates store m2
lrtest m1 m2

```

**Example of a Wald test.**

As was mentioned above, the Wald test approximates the lr test, but with the advantage that it only requires estimating one model. The Wald test works by testing that the parameters of interest are simultaneously equal to zero. If they are, this strongly suggests that removing them from the model will not substantially reduce the fit of that model, since a predictor whose coefficient is very small relative to its standard error is generally not doing much to help predict the dependent variable.

The first step in performing a Wald test is to run the full model (i.e., the model containing all four predictor variables). The first line of syntax below does this (but uses the quietly prefix so that the output from the regression is not shown). The second line of syntax below instructs Stata to run a Wald test in order to test whether the coefficients for the variables **math** and **science** are simultaneously equal to zero. The output first gives the null hypothesis. Below that we see the chi-squared value generated by the Wald test, as well as the p-value associated with a chi-squared of 27.53 with two degrees of freedom. Based on the p-value, we are able to reject the null hypothesis, again indicating that the coefficients for **math** and **science** are not simultaneously equal to zero, meaning that including these variables create a statistically significant improvement in the fit of

the model.

```
quietly: logit hiwrite female read math science
test math science

( 1)  math = 0
( 2)  science = 0

      chi2( 2) =    27.53
Prob > chi2 =    0.0000
```

### Example of a score or Lagrange multiplier test.

In order to perform the score test, you will need to download two user written packages for Stata. These packages are called **enumopt** and **testomit**. If you have an internet connection on the computer you use Stata on, these packages can be found by typing **findit enumopt** and **findit testomit** (as separate commands) in the Stata command window. (For more information or help see our FAQ page [How do I use findit to search for programs and additional help?](#) ) Assuming the necessary packages are installed, the syntax below shows how to run a score test. The first line of syntax runs the model with just **female** and **read** as predictor variables (recall that the score test uses a model with fewer variables and tests for omitted variables). The next line uses the command **predict** to generate a new variable called **test** that contains the score for each case. Without going into too much detail, the scores here are based on the model estimated and the value of the variables in the model for each case. The third line of syntax uses the **testomit** command to examine whether the variables **math** and/or **science** are variables which were incorrectly omitted from the model. The option **score(test)** tells Stata the name of the variable containing the scores, although it is in the options section (i.e., after the comma), this is required.

```
quietly: logit hiwrite female read
predict test, score
testomit math science, score(test)

logit: score tests for omitted variables
```

Term	score	df	p
math	28.94	1	0.0000
science	15.39	1	0.0001
simultaneous test	35.51	2	0.0000

The first part of the output gives the type of model run, followed by a table of results. The results of the score test are distributed chi-squared with degrees of freedom equal to the number of variables added to the model. The table has three columns, the first giving the value of the test statistic, the second the number of degrees of freedom for the test, and the third giving the p-value associated with a chi-squared of a given value with a given number of degrees of freedom. The variables **math** and **science** appear separately in their own rows, the first two rows contain the results for a test of whether adding either (but not both) of these variables to the model would significantly improve the fit of the model. The bottom row, labeled simultaneous test, tests whether adding both variables to the model will significantly improve the fit of the model. The results shown in the table are consistent with the wald and lr tests we performed above. They are also consistent with the regression output above, in which the coefficients for **math** and **science** were statistically significant.

The command **testomit** behaves somewhat differently for different estimation commands. Below are examples of how to use **testomit** with several other regression commands. Most multiple equation commands will use a syntax similar to the syntax for **mllogit**. Two exceptions are **ologit** and **oprobit**, and **regress**, which are shown separately.

For **mllogit** and many other multiple-equation commands:

```
mllogit perform read write
predict mslo mshi, score
testomit (low: science math) (high: science math), score(mslo mshi)
```

For **ologit** and **oprobit**:

```
ologit perform read write  
predict coef cut1 cut2, score  
testomit (LP: math science), score(coef)
```

For **regress**:

```
reg read write math  
predict regs, score  
testomit (mean: science female), score(regs)
```

---

[How to cite this page](#)

[Report an error on this page](#)

UCLA Researchers are invited to our [Statistical Consulting Services](#)  
We recommend others to our list of [Other Resources for Statistical Computing Help](#)  
These pages are [Copyrighted \(c\) by UCLA Academic Technology Services](#)

---

The content of this web site should not be construed as an endorsement of any particular web site, book, or software product by the University of California.

---