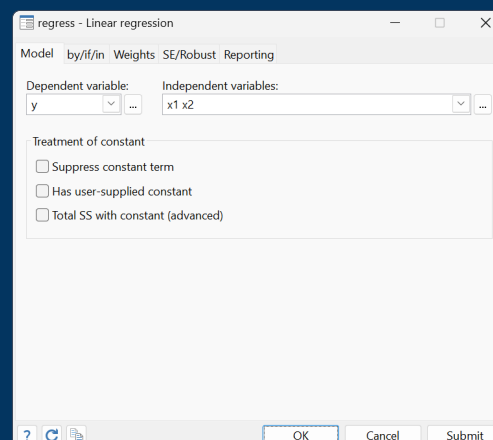


Maximum likelihood estimation

- Over a thousand built-in estimators
- Intuitive and consistent syntax
- Survey data support
- Program your own estimators
- Numerical or analytical derivatives
- Standard errors: OIM, OPG, Robust–Huber/White/sandwich, cluster–robust, bootstrap, jackknife, and more
- Powerful postestimation features
- Use point and click or type commands



Stata offers over a thousand built-in ML estimators

All follow elegant and intuitive syntax and have consistent output. Learn one command, know how to use them all.

Linear regression

```
. regress y x1 x2
```

Logistic regression

```
. logistic y x1 x2
```

Poisson regression

```
. poisson y x1 x2
```

Poisson regression with identity link (GLM)

```
. glm y x1 x2, family(poisson) link(identity)
```

ARIMA/ARMAX

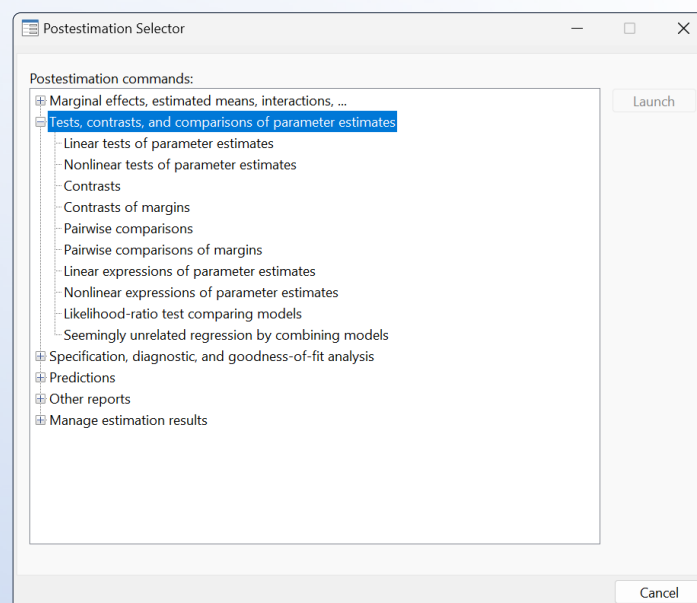
```
. arima y x1 x2, arima(2,1,3)
```

Logistic regression with survey data

```
. svy: logistic y x1 x2
```

After estimation, easily access powerful postestimation features.

```
. postest
```

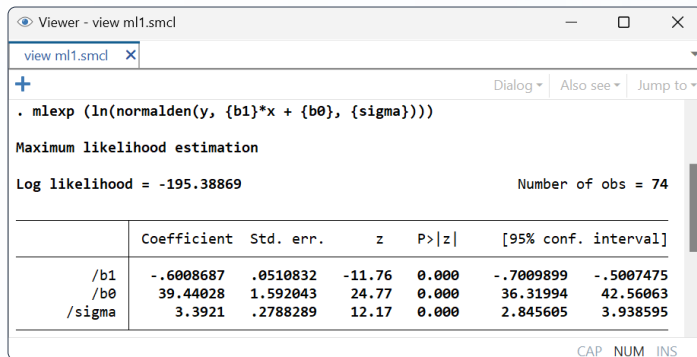


Write your own ML estimators

Stata offers a powerful environment for you to add your own ML estimators. For log likelihoods that can be written as simple expressions, just type the expression in the **mlexp** command. For more complicated expressions, you can write a program in Stata's scripting or matrix language and use the **ml** suite to do the rest for you. You can even turn your ML evaluator into a command.

Type a simple expression

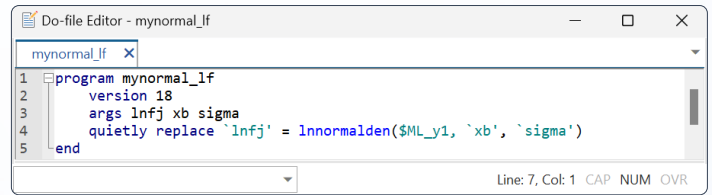
Use **mlexp** when your log likelihood can be expressed simply. For example, for normal linear regression, type



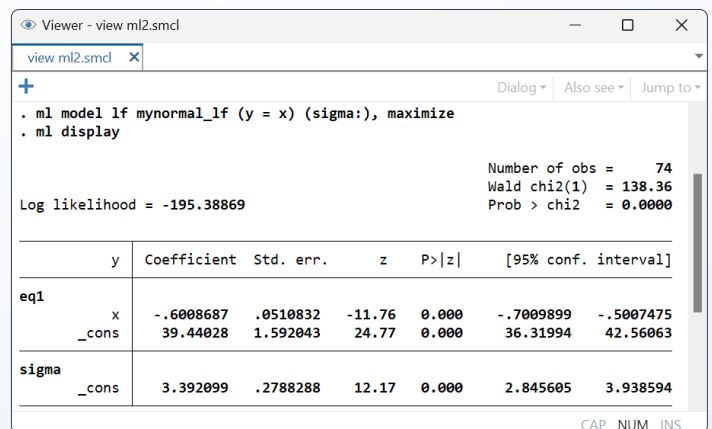
```
view ml1.smcl  
view ml1.smcl  
+ Dialog Also see Jump to  
. mlexp (ln(normalden(y, {b1}*x + {b0}, {sigma})))  
Maximum likelihood estimation  
Log likelihood = -195.38869 Number of obs = 74  
+-----+-----+-----+-----+-----+-----+  
/b1      Coefficient  Std. err.   z    P>|z|    [95% conf. interval]  
/b0      Coefficient  Std. err.   z    P>|z|    [95% conf. interval]  
/sigma   Coefficient  Std. err.   z    P>|z|    [95% conf. interval]  
+-----+-----+-----+-----+-----+-----+  
eq1      x      -.6008687   .0510832  -11.76  0.000   -.7009899   -.5007475  
_cons    39.44028   1.592043   24.77   0.000   36.31994   42.56063  
sigma    _cons    3.392099   .2788289   12.17   0.000   2.845605   3.938595  
CAP NUM INS
```

Write a program

Write a program to evaluate more complicated likelihood functions.



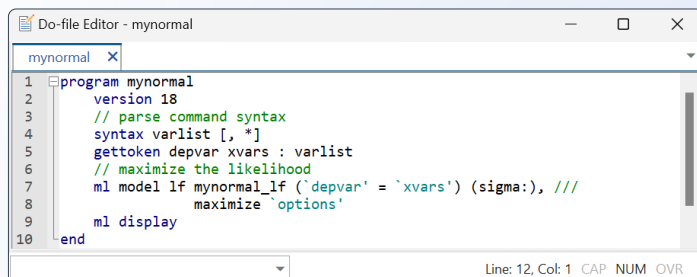
```
Do-file Editor - mynormal_lf  
mynormal_lf  
1 program mynormal_lf  
2 version 18  
3 args lnfj xb sigma  
4 quietly replace `lnfj' = lnnormalden($ML_y1, `xb', `sigma')  
5 end  
Line: 7, Col: 1 CAP NUM OVR
```



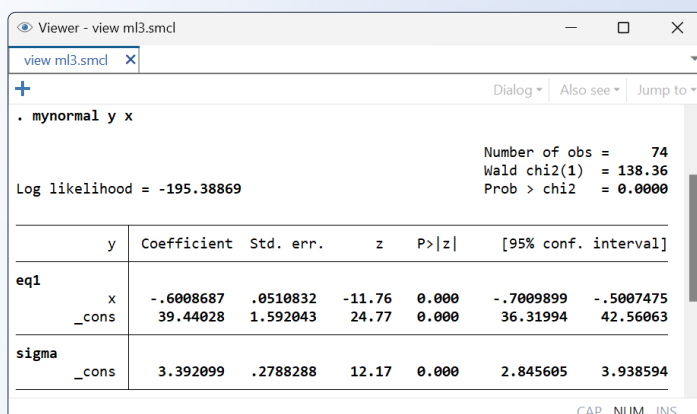
```
Viewer - view ml2.smcl  
view ml2.smcl  
+ Dialog Also see Jump to  
. ml model lf mynormal_lf (y = x) (sigma:), maximize  
. ml display  
Number of obs = 74  
Wald chi2(1) = 138.36  
Prob > chi2 = 0.0000  
Log likelihood = -195.38869  
+-----+-----+-----+-----+-----+-----+  
y      Coefficient  Std. err.   z    P>|z|    [95% conf. interval]  
eq1    x      -.6008687   .0510832  -11.76  0.000   -.7009899   -.5007475  
_cons  39.44028   1.592043   24.77   0.000   36.31994   42.56063  
sigma  _cons    3.392099   .2788288   12.17   0.000   2.845605   3.938594  
CAP NUM INS
```

Type a simple expression

With another small program, you can turn your likelihood-evaluation program into a full-fledged Stata command.



```
Do-file Editor - mynormal  
mynormal  
1 program mynormal  
2 version 18  
3 // parse command syntax  
4 syntax varlist [, *]  
5 gettoken depvar xvars : varlist  
6 // maximize the likelihood  
7 ml model lf mynormal_lf (`depvar' = `xvars') (sigma:), ///  
8 maximize `options'  
9 ml display  
10 end  
Line: 12, Col: 1 CAP NUM OVR
```



```
Viewer - view ml3.smcl  
view ml3.smcl  
+ Dialog Also see Jump to  
. mynormal y x  
Number of obs = 74  
Wald chi2(1) = 138.36  
Prob > chi2 = 0.0000  
Log likelihood = -195.38869  
+-----+-----+-----+-----+-----+-----+  
y      Coefficient  Std. err.   z    P>|z|    [95% conf. interval]  
eq1    x      -.6008687   .0510832  -11.76  0.000   -.7009899   -.5007475  
_cons  39.44028   1.592043   24.77   0.000   36.31994   42.56063  
sigma  _cons    3.392099   .2788288   12.17   0.000   2.845605   3.938594  
CAP NUM INS
```

Your new command automatically has many nice features such as options for robust and cluster-robust standard errors without any extra programming effort.

- `. mynormal y x, vce(robust)`
- `. mynormal y x, vce(cluster id)`

With a few more lines of code, your command can even support survey data,

- `. svy: mynormal y x`

Your command will also automatically work with postestimation features such as Wald tests, likelihood-ratio tests, contrasts, and much more.