Maximum Likelihood Estimation

Richard Williams, University of Notre Dame, http://www3.nd.edu/~rwilliam/
Last revised January 10, 2017

[This handout draws very heavily from Regression Models for Categorical and Limited Dependent Variables, 1997, by J. Scott Long. See Long's book, especially sections 2.6, 3.5 and 3.6 for additional details.]

Most of the models we will look at are (or can be) estimated via maximum likelihood.

Brief Definition. The maximum likelihood estimate is that value of the parameter that makes the observed data most likely.

Example. This is adapted from J. Scott Long's Regression Models for Categorical and Limited Dependent Variables.

Define p_i as the probability of observing whatever value of y was actually observed for a given observation, i.e.

$$p_i = \begin{cases} \Pr(y_i = 1 \mid x_i) & \text{if } y_i = 1 \text{ is observed} \\ 1 - \Pr(y_i = 1 \mid x_i) & \text{if } y_i = 0 \text{ is observed} \end{cases}$$

So, for example, if the predicted probability of the event occurring for case i was .7, and the event did occur, then $p_i = .7$. If, on the other hand, the event did not occur, then $p_i = .30$.

If the observations are independent, the likelihood equation is

$$L(\boldsymbol{\beta} \mid \mathbf{y}, \mathbf{X}) = \prod_{i=1}^{N} p_i$$

The likelihood tends to be an incredibly small number, and it is generally easier to work with the log likelihood. Ergo, taking logs, we obtain the log likelihood equation:

$$\ln L(\boldsymbol{\beta} \mid \mathbf{y}, \mathbf{X}) = \sum_{i=1}^{N} \ln p_i$$

Before proceeding, let's see how this works in practice! Here is how you compute p_i and the log of p_i using Stata:

```
. use http://www3.nd.edu/~rwilliam/statafiles/logist.dta, clear
. logit grade gpa tuce psi

Iteration 0:   log likelihood =  -20.59173
Iteration 1:   log likelihood = -13.259768
Iteration 2:   log likelihood = -12.894606
Iteration 3:   log likelihood = -12.889639
Iteration 4:   log likelihood = -12.889633
Iteration 5:   log likelihood = -12.889633
```

```
Logistic regression                             Number of obs     =         32
                                                LR chi2(3)        =      15.40
                                                Prob > chi2       =     0.0015
Log likelihood = -12.889633                     Pseudo R2         =     0.3740

------------------------------------------------------------------------------
       grade |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         gpa |   2.826113   1.262941     2.24   0.025     .3507938    5.301432
        tuce |   .0951577   .1415542     0.67   0.501    -.1822835    .3725988
         psi |   2.378688   1.064564     2.23   0.025       .29218    4.465195
       _cons |  -13.02135   4.931325    -2.64   0.008    -22.68657    -3.35613
------------------------------------------------------------------------------

. * Compute probability that y = 1
. predict prob1
(option pr assumed; Pr(grade))

 .* If y = 1, pi = probability y = 1
. gen pi = prob1 if grade == 1
(21 missing values generated)
. * If y = 0, replace pi with probability y = 0
. replace pi = (1 - prob1) if grade == 0
(21 real changes made)

. * compute log of pi
. gen lnpi = ln(pi)

. list  grade  gpa tuce psi prob1 pi lnpi, sep(8)

     +-------------------------------------------------------------+
     | grade    gpa   tuce   psi      prob1         pi        lnpi |
     |-------------------------------------------------------------|
  1. |     0   2.06     22     1   .0613758   .9386242   -.0633401 |
  2. |     1   2.39     19     1   .1110308   .1110308   -2.197947 |
  3. |     0   2.63     20     0   .0244704   .9755296   -.0247748 |
     |                --- Output Deleted ---                       |
 30. |     1      4     21     0    .569893    .569893   -.5623066 |
 31. |     1      4     23     1   .9453403   .9453403   -.0562103 |
 32. |     1   3.92     29     0   .6935114   .6935114   -.3659876 |
     +-------------------------------------------------------------+
```

So, this tells us that the predicted probability of the first case being 0 was .9386. The probability of the second case being a 1 was .111. The probability of the 3rd case being a 0 was .9755; and so on. The likelihood is therefore

$$L(\boldsymbol{\beta} \mid \mathbf{y}, \mathbf{X}) = \prod_{i=1}^{N} p_i = .9386 * .1110 * .9755 * ... * .6935 = .000002524$$

which is a really small number; indeed so small that your computer or calculator may have trouble calculating it correctly (and this is only 32 cases; imagine the difficulty if you have hundreds of thousands). Much easier to calculate is the log likelihood, which is

$$\ln L(\boldsymbol{\beta} \mid \mathbf{y}, \mathbf{X}) = \sum_{i=1}^{N} \ln p_i = -.0633 + -2.198 + ... + -.366 = -12.88963$$

Stata's `total` command makes this calculation easy for us:

```
. total lnpi

Total estimation                    Number of obs    =      32

--------------------------------------------------------------
             |      Total   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
        lnpi |  -12.88963   3.127734     -19.26869   -6.510578
--------------------------------------------------------------
```

Note this is the exact same value that logit reported as the log likelihood for the model. We call this number LL_M, i.e. the log likelihood for the model.

Incidentally, if we repeated this process for the constant-only model, i.e. if we instead ran the logistic regression

```
. logit grade
```

we would eventually get

```
. total lnpi

Total estimation                    Number of obs    =         32

--------------------------------------------------------------
             |      Total   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
        lnpi |  -20.59173    1.76514     -24.19176    -16.9917
--------------------------------------------------------------
```

This is the same value that the original logit reported as the log likelihood for iteration 0. We call this LL_0. As other handouts will show, LL_0 and LL_M can be used to compute various statistics of interest.

Expanded Definition. The maximum likelihood estimates are those values of the parameters that make the observed data most likely. That is, the maximum likelihood estimates will be those values which produce the largest value for the likelihood equation (i.e. get it as close to 1 as possible; which is equivalent to getting the log likelihood equation as close to 0 as possible).

Notes

- Maximum likelihood estimation is generally more complicated than this. You usually have more cases and are often estimating several parameters simultaneously. Nonetheless, the general idea is the same: the ML estimates are those values that make the observed data most likely. See Long for the technical details of how ML estimation is done.

- To say that something is the most likely value is not the same as saying it is likely; there are, after all, an infinite number of other parameter values that would produce almost the same observed data. As we've just seen, the likelihood that is reported for models is a very small number (which is part of the reason you'll see the log of the likelihood, or the deviance [-2LL, i.e. -2 * the log of the likelihood] reported instead).

- In the case of OLS regression, the maximum likelihood estimates and the OLS estimates are one and the same.

Properties of ML estimators

- The ML estimator is consistent. As the sample size grows large, the probability that the ML estimator differs from the true parameter by an arbitrarily small amount tends toward 0.

- The ML estimator is asymptotically efficient, which means that the variance of the ML estimator is the smallest possible among consistent estimators.

- The ML estimator is asymptotically normally distributed, which justifies various statistical tests.

ML and Sample Size. For ML estimation, the desirable properties of consistency, normality and efficiency are asymptotic, i.e. these properties have been proven to hold as the sample size approaches infinity. The small sample behavior of ML estimators is largely unknown. Long says there are no hard and fast rules for sample size.

- He says it is risky to use ML with samples smaller than 100, while samples over 500 seem adequate.

- More observations are needed if there are a lot of parameters – he suggests that at least 10 observations per parameter seems reasonable for the models he discusses.

- If the data are ill-conditioned (e.g. IVs are highly collinear) or if there is little variation in the DV (e.g. nearly all outcomes are 1) a larger sample is required.

- Some models seem to require more cases, e.g. ordinal regression models.

- Both Long and Allison agree that the standard advice is that with small samples you should accept larger p-values as evidence against the null hypothesis. Given that the degree to which ML estimates are normally distributed in small samples is unknown, it is more reasonable to require smaller p-values in small samples.

Numerical Methods for ML Estimation. For OLS regression, you can solve for the parameters using algebra. Algebraic solutions are rarely possible with nonlinear models. Consequently, numeric methods are used to find the estimates that maximize the log likelihood function. Numerical methods start with a guess of the values of the parameters and iterate to improve on that guess. The iterative process stops when estimates do not change much from one step to the next. Long (1997) describes various algorithms that can be used when computing ML estimates.

Occasionally, there are problems with numerical methods:

- It may be difficult or impossible to reach convergence, e.g. you'll get a message like "Convergence not obtained after 250 iterations."

- Convergence does occur, but you get the wrong solution (this is rare, but still, you might want to be suspicious if the numbers just don't look right to you).

- In some cases, ML estimates do not exist for a particular pattern of data. For example, with a binary outcome and a single binary IV, ML estimates are not possible if there is no variation in the IV for one of the outcomes (e.g. everybody coded 1 on the IV is also coded 1 on the DV). Long and Freese (2006) also discuss this problem. Here is an example:

```
. list

     +--------------+
     | x   y   freq |
     |--------------|
  1. | 1   1     25 |
  2. | 1   0      0 |
  3. | 0   1     30 |
  4. | 0   0     12 |
     +--------------+

. logit y x [fw=freq]

note: x != 0 predicts success perfectly
      x dropped and 1 obs not used

Iteration 0:   log likelihood = -25.127323

Logit estimates                                   Number of obs   =         42
                                                  LR chi2(0)      =       0.00
                                                  Prob > chi2     =          .
Log likelihood = -25.127323                       Pseudo R2       =     0.0000

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
       _cons |   .9162907    .341565     2.68   0.007     .2468356    1.585746
------------------------------------------------------------------------------

. dis invlogit(_b[_cons])
.71428571
```

In the above example, whenever x = 1, y = 1. ML estimates are impossible. So, what Stata does, is it drops x from the model along with all observations where x = 1. As a result, you are left analyzing the 42 cases where x = 0 and y = 0 or 1. For those 42 cases, p(y = 1) = 30/42 = .71428571.

If you seem to be having problems with your estimation, Long suggests the following:

- *Model specification.* Make sure the software is estimating the model you want to estimate, i.e. make sure you haven't made a mistake in specifying what you want to run. (And, if it is running what you wanted it to run, make sure that what you wanted it to do actually makes sense!)

- *Incorrect variables.* Make sure the variables are correct, e.g. variables have been computed correctly. Check the descriptive statistics. Long says his experience is that most problems with numerical methods are due to data that have not been "cleaned."

- *Number of observations.* Convergence generally occurs more rapidly when there are more observations. Not that there is much you can do about sample size, but this may explain why you are having problems.

- *Scaling of variables.* Scaling can be important. The larger the ratio between the largest standard deviation and the smallest standard deviation, the more problems you will have with numerical methods. For example, if you have income measured in dollars, it may have a very large standard deviation relative to other variables. Recoding income to thousands of dollars may solve the problem. Long says that, in his experience, problems are much more likely when the ratio between the largest and smallest standard deviations exceeds 10. (You may want to rescale for presentation purposes anyway, e.g. the effect of 1 dollar of income may be extremely small and have to be reported to several decimal places; coding income in thousands of dollars may make your tables look better.)

- *Distribution of outcomes.* If a large proportion of cases are censored in the tobit model or if one of the categories of a categorical variable has very few cases, convergence may be difficult. Long says you can't do much about this, although for the latter problem I think you could sometimes combine categories.

When the model is appropriate for the data, Long says that ML estimation tends to work well and convergence is often achieved within 5 iterations. Rescaling can solve some problems. If still having problems, you can try using a different program that uses a different algorithm; a problem that may be very difficult for one algorithm may work quite well for another.

The troubleshooting FAQ for my `gologit2` program also has several suggestions for what to do when you have convergence problems, and much of this advice will apply to programs besides `gologit2`. See

http://www3.nd.edu/~rwilliam/gologit2/tsfaq.html