The Stata Journal (2002)
2, Number 4, pp. 403–410

From the help desk: Demand system estimation

Brian P. Poi
Stata Corporation

Abstract. This article provides an example illustrating how to use Stata to estimate systems of household demand equations. More generally, the techniques developed here can be used to estimate any system of nonlinear equations using Stata's maximum likelihood routines.

Keywords: st0029, nonlinear estimation, maximum likelihood, demand equations

1 Household demand analysis

The goal of demand analysis is to model households' expenditure patterns on a group of related items in order to obtain estimates of price and income elasticities and to estimate consumer welfare. In some applications, broad categories, such as food, clothing, housing, and transportation, are used. Other applications study the demand for various categories of food. Although a bit dated, Deaton and Muellbauer (1980b) remains a classic introduction to the subject.

Since Stone's (1954) linear expenditure system, there has been widespread interest in choosing an estimable system of equations to represent household demand for various goods. Two of the most well-known include the translog system of Christensen et al. (1975) and the Deaton and Muellbauer (1980a) almost ideal demand system (AIDS). Of these two approaches, the AIDS has proven to be more popular, because it permits exact aggregation over households and is easier to estimate. More recently, Banks et al. (1997) have presented a generalization of the AIDS model that includes a quadratic expenditure term, and so they called their model the QUAIDS; this paper discusses estimation of that demand system.

Demand systems are typically specified with expenditure shares as the dependent variables. A household's expenditure share for good i is defined as

$$w_i \equiv \frac{p_i q_i}{m}$$

where p_i is the price paid for good i, q_i is the quantity of good i purchased or consumed, and m is the total expenditure on all goods in the demand system. With this definition of m,

$$\sum_{i=1}^{K} w_i = 1$$

where K is the number of goods in the system. In the QUAIDS model, expenditure share equations have the form

$$w_i = \alpha_i + \sum_{j=1}^{K} \gamma_{ij} \ln p_j + \beta_i \ln \left\{ \frac{m}{P(\boldsymbol{p})} \right\} + \frac{\lambda_i}{b(\boldsymbol{p})} \left[\ln \left\{ \frac{m}{P(\boldsymbol{p})} \right\} \right]^2 \tag{1}$$

where $\boldsymbol{p}$ is the vector of all prices, $b(\boldsymbol{p})$ is defined as

$$b(\boldsymbol{p}) \equiv \prod_{i=1}^{K} p_i^{\beta_i} \tag{2}$$

and $\ln P(\boldsymbol{p})$ is a price index defined as

$$\ln P(\boldsymbol{p}) \equiv \alpha_0 + \sum_{i=1}^{K} \alpha_i \ln p_i + \frac{1}{2} \sum_{i=1}^{K} \sum_{j=1}^{K} \gamma_{ij} \ln p_i \ln p_j \tag{3}$$

Economic theory imposes several constraints on the parameters. The fact that $\sum_i w_i = 1$, often called the adding-up condition, requires that

$$\sum_{i=1}^{K} \alpha_i = 1 \qquad \sum_{i=1}^{K} \beta_i = 0 \qquad \sum_{i=1}^{K} \lambda_i = 0 \quad \text{and} \quad \sum_{i=1}^{K} \gamma_{ij} = 0 \;\; \forall\, j \tag{4}$$

Moreover, since demand functions are homogeneous of degree zero in $(\boldsymbol{p}, m)$,

$$\sum_{j=1}^{K} \gamma_{ij} = 0 \;\; \forall\, j \tag{5}$$

Slutsky symmetry implies that

$$\gamma_{ij} = \gamma_{ji} \tag{6}$$

Usually, α_0 is difficult to estimate directly and so is set equal to the minimum level of expenditure that would be needed for subsistence if all prices were equal to one; for a justification of this, see Deaton and Muellbauer (1980a).

An error term ϵ_i is added to the right-hand side of equation (1) for estimation purposes. In addition, $\boldsymbol{\epsilon} \equiv [\epsilon_1, \ldots, \epsilon_k]$ is usually assumed to have a multivariate normal distribution with covariance matrix $\boldsymbol{\Sigma}$. However, the adding-up condition implies that $\boldsymbol{\Sigma}$ is singular. Therefore, one of the K demand equations is dropped from the system, the remaining $(K-1)$ equations are estimated by maximum likelihood, and then the parameters of the final equation are recovered using the parameter constraints mentioned above. Barten (1969) showed that it makes no difference which equation is dropped. A straightforward application of the delta method then yields the covariance matrix, including the parameters for the Kth equation.

The concentrated log-likelihood function for the $(K-1)$ equations in a sample of N households is

$$\ln L = -\frac{N}{2} \Big[(K-1) \left\{ 1 + \ln(2\pi) \right\} + \ln |\boldsymbol{S}| \Big] \tag{7}$$

where N is the number of households or individuals and

$$\boldsymbol{S} \equiv \frac{1}{N} \sum_{t=1}^{N} \widehat{\boldsymbol{\epsilon}}_t^* \widehat{\boldsymbol{\epsilon}}_t^{*\prime} \tag{8}$$

where t indexes households and $\widehat{\boldsymbol{\epsilon}}_t^* \equiv [w_{1t} - \widehat{w}_{1t}, \ldots, w_{K-1,t} - \widehat{w}_{K-1,t}]$.

2 An example

This section illustrates the technique outlined in the previous section by estimating a four-equation demand system using data from the 1987–1988 Nationwide Food Consumption Survey conducted by the United States Department of Agriculture. Demands for four categories of food are estimated: meats, fruits and vegetables, breads and cereals, and miscellaneous. The sample used here consists of 4,048 households. Poi (2002) contains a more detailed discussion of the data and fits a slightly larger model, which included a separate category for dairy products.

Stata's `ml` commands provide a very convenient set of tools for fitting user-defined maximum likelihood models. Gould and Sribney (1999) provides an excellent tutorial. When the equations to be estimated are linear or nearly so, the `ml` commands' θ-notation makes coding likelihood functions particularly simple. In method `lf`, the quantity $\boldsymbol{x}_i'\boldsymbol{\beta}$ is computed automatically for each equation. With methods `d0`, `d1`, and `d2`, the `mltheta` command is used to evaluate that inner product.

In the case of the system estimated here, however, the demand equations are inherently nonlinear. Moreover, cross-equation restrictions must be placed on the parameters. To reduce the total number of parameters to estimate, the concentrated form of the likelihood function is maximized. What is needed, then, is an approach that allows the likelihood evaluator to receive an arbitrary vector of parameters that will then be manipulated to compute the likelihood function. While a large majority of statistical models can be written in terms of $\boldsymbol{x}_i'\boldsymbol{\beta}$ directly, in this application, having that expression evaluated automatically provides virtually no benefit. Here, complete flexibility to manipulate the parameters of the model is required.

With $K = 4$ equations, there are a total of $3(K-1) + \frac{1}{2}K(K-1) = 15$ parameters to estimate. For estimation purposes, these parameters are best viewed as a single vector $\boldsymbol{\theta}$ of parameters consisting of the first three αs, then the first three βs, then vech($\boldsymbol{\Gamma}^*$), where $\boldsymbol{\Gamma}^*$ contains only the first $(K-1)$ rows and columns of $\boldsymbol{\Gamma}$, and finally the first three λs. The parameters for the fourth equation will then be obtained using the restrictions shown in equations (4), (5), and (6).

The dataset `food.dta` consists of four variables named `w1`, `w2`, `w3`, and `w4`, which contain the four expenditure shares for each household. Variables `lnp1` through `lnp4` contain the logarithms of prices paid by each household for the four food categories. Finally, the variable `lnexp` contains the logarithm of expenditures on the foods.

Several program files accompany this article and are downloadable from the *Stata Journal* web site. Because the key to understanding nonlinear multivariate estimation in Stata is being able to program the likelihood function, those programs are explained in some detail here. The program `vec_sum.ado` simply returns the sum of the elements of a vector and is self-explanatory. As with all type `d0` likelihood evaluators, `lnl_quaids.ado` is passed the argument `todo`, the parameter vector `b`, and the scalar `lnf`, which is to be filled with the overall log likelihood. The evaluator first passes the vector of parameters to program `quaids_params.ado`, which recovers the actual $K \times 1$ vectors $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\lambda}$ and the $K \times K$ matrix $\boldsymbol{\Gamma}$. This program splits up `b` according to the scheme discussed in the previous paragraph and then, with the help of the `vec_sum.ado` program, imposes the necessary parameter constraints. In all of these programs, the global variable `$NEQN` represents the number of equations K.

Once `lnl_quaids.ado` has the parameters in a usable form, the next step is to compute the price index variable defined in equation (3); the global variable `$ANOT` is used to hold the parameter α_0. The program then calculated deflated incomes $m/P(\boldsymbol{p})$ and the $b(\boldsymbol{p})$ term defined in equation (2). With these preliminaries, K variables containing the residuals from the expenditure share equations are computed, and Stata's `matrix accum` command and a few matrix manipulations yield the covariance matrix (8). Computing the actual likelihood (7) then requires just one more line of code.

After these programs have been saved, estimation is straightforward and proceeds as with most other `ml` estimation. To start, load the data and initialize the global variables `$A_NOT` and `$NEQN`:

```
. use food, clear
. global A_NOT = 5
. global NEQN = 4
```

Next, specify the model using the `ml model` command:

```
. ml model d0 lnl_quaids () /a2 /a3 /b1 /b2 /b3 /g11 /g21 /g31 /*
>     */  /g22 /g32 /g33 /l1 /l2 /l3
```

Stata's `ml model` command requires that at least one equation be specified with the `(eq)` syntax. Therefore, one cannot use `/a1` to refer to α_1. After the model has been specified, give instructions to find initial parameters and then maximize the likelihood function. Displaying the coefficients will be handled later, so for now, use the `nooutput` option with `ml maximize`.

```
. ml search
initial:       log likelihood =    4475.42
improve:       log likelihood =    4475.42
alternative:   log likelihood = -18282.737
rescale:       log likelihood =   6077.236
rescale eq:    log likelihood =   10683.67
```

(*Continued on next page*)

```
. ml maximize, noclear nooutput
initial:       log likelihood =   10683.67
rescale:       log likelihood =   10683.67
rescale eq:    log likelihood =  10785.383
Iteration 0:   log likelihood =  10785.383  (not concave)
Iteration 1:   log likelihood =  11816.321  (not concave)
Iteration 2:   log likelihood =  12291.816
Iteration 3:   log likelihood =  12445.116
Iteration 4:   log likelihood =  12705.123
Iteration 5:   log likelihood =  13070.869
Iteration 6:   log likelihood =  13093.257
Iteration 7:   log likelihood =  13093.487
Iteration 8:   log likelihood =  13093.487
```

The `noclear` option is included so that `ml report` can be used if desired.

One could now list the estimated parameter vector `e(b)` and covariance matrix `e(V)` produced by `mloutput`, but they do not include the parameters for the Kth equation. Program `quaids_vec.ado` takes the parameter vector for the first $K-1$ equations and returns a vector that includes the parameters for all K equations. Program `quaids_params.ado` can also be used to obtain separate matrices for $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\Gamma}$, and $\boldsymbol{\lambda}$. To get the entire parameter vector, type

```
. matrix b = e(b)
. quaids_vec b theta
```

The vector theta now contains all estimated parameters.

The delta method is used to compute the covariance matrix. Letting $f(\boldsymbol{\theta})$ be the function that transforms $\boldsymbol{\theta}$ into the parameter vector for all K equations, the delta method requires computation of $\partial f(\boldsymbol{\theta})/\partial\boldsymbol{\theta}'$. Program `quaids_delta.ado` computes this derivative matrix for the present application, where $K = 4$; generalizing the program to accept any value of K is difficult because of the terms involving the γs. To get the covariance matrix, do the following:

```
. matrix v = e(V)
. quaids_delta r
. matrix var = r*v*r'
```

The final step before displaying the coefficients is to provide row and column names to the parameter vector and covariance matrix so that they can be posted and displayed using `estimates post` and `estimates display`. The next several lines of code accomplish all of this:

```
. forvalues i = 1/$NEQN {
  2.        global anames "$anames alpha:`i'"
  3.        global bnames "$bnames beta:`i'"
  4.        global lnames "$lnames lambda:`i'"
  5.        forvalues j = `i'/$NEQN {
  6.                global gnames "$gnames gamma:`j'`i'"
  7.        }
  8. }
```

```
. global names "$anames $bnames $gnames $lnames"
. matrix colnames theta = $names
. matrix colnames var = $names
. matrix rownames var = $names
. estimates post theta var
. estimates display
------------------------------------------------------------------------------
             |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
alpha        |
           1 |   .3136616   .0087845    35.71   0.000     .2964443     .330879
           2 |   .2712567   .0067258    40.33   0.000     .2580743    .2844391
           3 |   .1052015   .0034402    30.58   0.000     .0984587    .1119442
           4 |   .3098802   .0064653    47.93   0.000     .2972084    .3225519
-------------+----------------------------------------------------------------
beta         |
           1 |   .0099745   .0112141     0.89   0.374    -.0120047    .0319537
           2 |  -.0261289    .008397    -3.11   0.002    -.0425868   -.0096711
           3 |   .0041683   .0043783     0.95   0.341     -.004413    .0127496
           4 |   .0119861   .0089786     1.33   0.182    -.0056117    .0295839
-------------+----------------------------------------------------------------
gamma        |
          11 |   .1214999   .0057281    21.21   0.000      .110273    .1327268
          21 |  -.0522583   .0039358   -13.28   0.000    -.0599724   -.0445443
          31 |  -.0351566   .0021892   -16.06   0.000    -.0394473   -.0308659
          41 |   -.034085   .0036337    -9.38   0.000    -.0412069   -.0269631
          22 |   .0644288   .0044602    14.45   0.000     .0556869    .0731707
          32 |   -.001202   .0019829    -0.61   0.544    -.0050883    .0026843
          42 |  -.0109685   .0029765    -3.69   0.000    -.0168023   -.0051346
          33 |   .0425055   .0017735    23.97   0.000     .0390295    .0459815
          43 |  -.0061469   .0016456    -3.74   0.000    -.0093723   -.0029215
          44 |   .0512004    .003652    14.02   0.000     .0440425    .0583582
-------------+----------------------------------------------------------------
lambda       |
           1 |  -.0025218   .0043894    -0.57   0.566     -.011125    .0060813
           2 |  -.0000235   .0032918    -0.01   0.994    -.0064753    .0064282
           3 |   .0011219   .0017151     0.65   0.513    -.0022396    .0044835
           4 |   .0014234   .0035208     0.40   0.686    -.0054771     .008324
------------------------------------------------------------------------------
```

Now that the correct parameter vector and covariance matrix have been posted, Stata's `test` command can be used to conduct Wald tests.

```
. test [lambda]1 = 0, notest
 ( 1)  [lambda]1 = 0.0
. test [lambda]2 = 0, notest accumulate
 ( 1)  [lambda]1 = 0.0
 ( 2)  [lambda]2 = 0.0
. test [lambda]3 = 0, notest accumulate
 ( 1)  [lambda]1 = 0.0
 ( 2)  [lambda]2 = 0.0
 ( 3)  [lambda]3 = 0.0
```

```
. test [lambda]4 = 0, accumulate

 ( 1)  [lambda]1 = 0.0
 ( 2)  [lambda]2 = 0.0
 ( 3)  [lambda]3 = 0.0
 ( 4)  [lambda]4 = 0.0
       Constraint 3 dropped

           chi2(  3) =    0.58
         Prob > chi2 =    0.9020
```

Notice how parameters are specified here. In Stata's nomenclature, `lambda` is an equation name, and `1`, `2`, `3`, and `4` are the parameter names. More importantly, why was one of the constraints dropped? First, recall that the adding-up constraint (4) causes $\mathbf{\Sigma}$ to be singular; this implies that the covariance matrix of all the parameters is also singular.

```
. matrix var = e(V)

. display det(var)
1.411e-224
```

Intuitively, since $\sum_i \lambda_i = 0$, testing

$$H_0 : \lambda_i = 0, \qquad i = 1, \ldots, K$$

is equivalent to testing

$$H_0 : \lambda_i = 0, \qquad i = 1, \ldots, K-1$$

Fortunately, Stata catches the redundancy and drops one of the test's conditions. Note that the exact same result is obtained by testing only three of the λs.

```
. test [lambda]1 = 0, notest

 ( 1)  [lambda]1 = 0.0

. test [lambda]2 = 0, notest accumulate

 ( 1)  [lambda]1 = 0.0
 ( 2)  [lambda]2 = 0.0

. test [lambda]3 = 0, accumulate

 ( 1)  [lambda]1 = 0.0
 ( 2)  [lambda]2 = 0.0
 ( 3)  [lambda]3 = 0.0

           chi2(  3) =    0.58
         Prob > chi2 =    0.9020
```

In any event, the quadratic income terms are not statistically significant in this particular application.

3 Conclusion

This article has discussed estimation of a system of household demand equations subject to a set of constraints imposed by the expenditure minimization problem. Although estimation with Stata still requires that a set of programs be written by the user, the sample programs here serve as a useful outline for other nonlinear multivariate problems. For example, the estimation of firms' cost functions often involves techniques very similar to those used in demand analysis.

By using Stata instead of a general matrix programming language, one is able to take advantage of Stata's powerful, easy-to-use set of `ml` commands instead of having to use a general optimization routine. Stata's `ml search` command makes finding initial parameters much easier, and its `ml maximize` command produces the covariance matrix automatically. Data handling is also much easier in Stata, as is hypothesis testing.

4 References

Banks, J., R. Blundell, and A. Lewbel. 1997. Quadratic Engel curves and consumer demand. *Review of Economics and Statistics* 69: 527–539.

Barten, A. 1969. Maximum likelihood estimation of a complete system of demand equations. *European Economic Review* 1: 7–73.

Christensen, L. R., D. W. Jorgenson, and L. J. Lau. 1975. Transcendental logarithmic utility functions. *American Economic Review* 65: 367–383.

Deaton, A. S. and J. Muellbauer. 1980a. An almost ideal demand system. *American Economic Review* 70: 312–326.

—. 1980b. *Economics and Consumer Behavior*. Cambridge: Cambridge University Press.

Gould, W. and W. Sribney. 1999. *Maximum Likelihood Estimation with Stata*. College Station, TX: Stata Press.

Poi, B. P. 2002. Dairy policy and consumer welfare. In *Three Essays in Applied Econometrics*, Chapter II, Doctoral thesis. Department of Economics, University of Michigan.

Stone, R. N. 1954. Linear expenditure systems and demand analysis: An application to the pattern of British demand. *Economic Journal* 64: 511–527.

About the Author

Brian Poi received his Ph.D. in economics from the University of Michigan in the summer of 2002 and joined Stata Corporation as a staff statistician.