ML (nonlinear options) *log likelihood equation name* ;
or
ML (nonlinear options) *procedurename list of parameters* ;

Function:

ML is a general purpose maximum likelihood estimation procedure. It can be used to estimate the parameters of any (identified) model for which you can write down the logarithm of the likelihood in a TSP equation (FRML), or evaluate the log likelihood in a procedure (PROC).

Usage:

FRML method. Usually the simplest approach is to write the log likelihood equation in a FRML with LOGL as the dependent variable. Note that this equation is for each observation in the current SMPL vector. If there are different equations, depending on different cases, write the equation as the sum of the individual ones, with dummy variables multiplying each equation to select the appropriate one for any given observation. Often the "case" will be determined by a dependent discrete choice variable, and observations are usually i.i.d., but the likelihood function could be made different for different parts of the SMPL by using more general time-dependent dummy variables. Of course, the log likelihood must be additively separable over the SMPL for this method to work.

Then use a PARAM statement to specify which of the variables in the LOGL equation are to be estimated and supply their starting values if desired. Follow this by an ML command with any of the standard NONLINEAR options and the name of the equation which specifies the likelihood function. ML will maximize this function with respect to the parameters using a standard gradient method; the exact form of the Hessian approximation used as a weighting matrix depends on the HITER option. The default is to use the BHHH method, a method of scoring, but with the sample covariance of the gradient of the likelihood used in place of its expectation.

Good Applications for the FRML method:

1. Truncation models involving CNORM(), such as two-limit Tobit.

2. Nonlinear equations for PROBIT, TOBIT, LOGIT, etc. This includes parameter restrictions and holding parameters fixed.

3. Checking your own second derivatives when you are writing a FORTRAN routine for user maximization.

4. Robust models like LOGL = ABS(Y-XB).

5. Minimization problems (just negate the equation).

6. General maximum likelihood problems, using any functions recognized by TSP (including new ones like SQRT, POS, Factorial, and Gamma function, which can be used for the gamma, chi-squared, beta, t, and F densities). Even complicated likelihood functions on large datasets which have been traditionally estimated with a FORTRAN routine may be estimable with less overall cost by using ML. Even though more computer time may be required, programming time is considerably reduced (and the relative price of CPU time is usually small and shrinking). Maximization with analytic derivatives is usually much faster than with numeric derivatives (the method which used to be lowest in programming cost). See Timing example below.

PROC method. Sometimes it is extremely difficult, or impossible to write down the log likelihood in a single FRML (even with use of EQSUB). See the list below for some examples. For this form of the ML command, write a PROC which evaluates the log likelihood and stores it in @LOGL. Give the name of this PROC as the first argument (after any options) of the ML command, and follow it with a list of PARAMs which are to be estimated. Write the PROC so that it starts by checking any constraints on the PARAMs. If any constraint is violated, set @LOGL to @MISS before exiting from the PROC. OPTIONS DOUBLE; is advised if you want to use double precision to form intermediate results such as residuals.

The main disadvantage of using the PROC instead of the FRML method is that analytic derivatives are not available. However, numeric derivatives (default HITER=F and GRAD=C2) will often be quite adequate. A slight disadvantage is that you have to explicitly list the PARAMs to be estimated in the command line.

Good Applications for the PROC method:

1. Time series models like ARMA and GARCH, where the equations are recursive (depend on residuals or variance from the previous time period(s). Models which can be evaluated by the KALMAN command also fit into this category (ML thus allows estimation of the hyperparameters).

2. Multi-equation models like FIML. These involve Jacobians, matrix inverses, and determinants, which would have to be written into the log likelihood equation by hand (very difficult for more than about 4 equations unless the Jacobian is sparse).

3. Models which require several diverse commands to evaluate, such as multivariate normal integrals via simulation, or other functions that are not built in to TSP. Another example in this class is a concentrated log likelihood function (the FRML method can only handle the unconcentrated log likelihood, which is usually more nonlinear and often harder to write).

4. Models with complicated constraints. A good example would be ARCH models, where the conditional variance must be positive for every observation.

Bad Applications for either method:

1. Existing linear models in TSP (PROBIT, TOBIT, LOGIT, SAMPSEL). The regular TSP commands are more efficient, more resistant to numerical problems, often have better starting values, and provide model-specific statistics. See Timing example below.

To give an idea of how much this convenience costs in terms of CPU time, here is a timing example run on the VAX 11/780 of a Probit on 385 observations, 8 variables.

Time in CPU seconds	**Procedure**
05.24	PROBIT command.
65.65	ML(HITER=B,HCOV=N)
75.97	ML(HITER=N,HCOV=N)

The moral is that ML should not be used when you have a Fortran-coded alternative estimation program, but could be useful if you don't want to spend your time developing such a program. Also, in this case, the method of scoring was somewhat faster than Newton's method, although the latter is more powerful (it takes fewer iterations).

Tips:

1. Write the equation carefully to avoid things like Log(x<=0) or Exp(x>88). These are fatal errors if they happen in the first function evaluation (using the starting values). They are not fatal during iterations (the

program automatically uses a smaller stepsize), but they can be inefficient. Often these problems can be avoided by reparametrizing the likelihood function. The standard example of this is estimating SIGMA (or SIGMA- inverse) instead of SIGMA-squared.

If you are getting numerical errors and you can't rewrite the likelihood function, try using SELECT to remove the problem observations. After you get convergence, use the converged values as starting values and reestimate using the full sample.

2. Choose starting values carefully (see previous).

3. Use EQSUB for less work rewriting equations and more efficient code.

4. The "Working space=" message is an indication of the length of the derivative code.

5. If the second derivative matrix is singular, you may have sign errors in the log likelihood function (the inversion routine assumes the second derivative matrix is negative definite).

6. If you are using derivatives, make sure the functions you are using are differentiable. Logical operations are not differentiable everywhere, although they are diffferentiable at all but a finite number of points. TSP will do the best it can with them, but if you end up on a kink (corner), it may stall.

Examples:

FRML method. For example, in the Probit model, the likelihood is CNORM(-XB) for Y<=0, and (1-CNORM(-XB)) for Y>0. This could be written as (see the **User's Guide** Section 9.5 for alternate coding and many more examples):

```
GENR Y0 = Y<=0;
GENR Y1 = Y>0;
FRML EQ1 LOGL = LOG (Y0*CNORM(-XB) + Y1*(1-CNORM(-XB)));
```

The XB expressions can be filled in later with the EQSUB command (see the examples in that section). Note that this allows for nonlinear equations, as in this example:

```
FRML NLXB XB = B0 + B1*X1 + (B2/B1)*X2;
EQSUB EQ1 NLXB;
ML EQ1;
```

PROC method. Here is a simple concentrated log likelihood function, where we estimate the mean of a time trend, and concentrate out the variance parameter to reduce the nonlinearity of the function.

```
OPTIONS DOUBLE;                    ? make sure that residuals are stored in double precision
SMPL 1,9;
TREND T;                           ? yields same results as MSD T;  or  OLSQ T C;
PARAM MT,2;
ML  NRMLC  MT;                     ? PROC form of the ML command
PROC NRMLC;
  E = T - MT;                      ? residual
  MAT SIG2 = (E'E)/@NOB;           ? sigma-squared  (variance)
  SET PI = 4*ATAN(1);              ? tan(pi/4) = 1
  SET @LOGL = -(@NOB/2)*( LOG(SIG2) + 1 + LOG(2*PI) );
ENDPROC;
```

Options:

Standard nonlinear options (see NONLINEAR section). HITER=B,HCOV=B is the default for the FRML method;

ML

HITER=F,HCOV=F,GRAD=C2 is the default for the PROC method.

Starting values are from the PARAM and SET statements. Note that the CONST command allows fixing parameters during an estimation (for the FRML method).

Output:

Name	Type	Length	Variable Description
@RNMS	list	#params	Names of right hand side variables.
@LOGL	scalar	1	Log of likelihood function.
@IFCONV	scalar	1	1 if convergence acheived, 0 otherwise.
@NCOEF	scalar	1	Number of parameters to be estimated.
@NCID	scalar	1	Number of identified parameters.
@COEF	vector	#params	Coefficient estimates.
@GRAD	vector	#params	Gradient of log L at convergence.
@SES	vector	#params	Standard errors.
@T	vector	#params	T-statistics.
@VCOV	matrix	#params* #params	Variance-covariance of estimated coefficients.

See the NONLINEAR section for the alternative names for when the HCOV option is used.

Method:

The method used is a standard gradient method, explained in somewhat more detail in Chapter 9 of the User's Manual. Briefly, at each iteration, a new parameter vector is computed by moving in the direction specified by the gradient of the likelihood (uphill), weighting this gradient by an approximation to the matrix of second derivatives at that point (in order to adjust for the curvature). Convergence is declared when the changes in the parameters are all "small", where small is defined by the TOL= option.

The ML procedure normally uses analytic first (and second) derivatives (for the FRML method). The function can also be maximized numerically (HITER=F or HITER=D). See NONLINEAR for more information on the options (HCOV=N gives standard errors based on analytic second derivatives).

References:

Berndt, E. K., B. H. Hall, R. E. Hall, and J. A. Hausman, "Estimation and Inference in Nonlinear Structural Models," **Annals of Economic and Social Measurement,** October 1974, pp. 653-665.

Gill, Philip E., Walter Murray, and Margaret H. Wright, **Practial Optimization,** Academic Press, New York, 1981.