Maximum Likelihood Module

See the `max.e` files in the directory `c:\gauss\examples` for examples of how to use MAXLIK.

Procedure For Computing Likelihood Function

The user must provide a procedure for computing the log-likelihood for either one observation, or for a matrix of observations. The procedure must have two input arguments, first, a vector of parameter values, and second, one or more rows of the data matrix. The output argument is the log-likelihood for the observation or observations in the second argument evaluated at the parameters values in the first argument. Suppose that the function procedure has been named fct, the following considerations apply:

```
FORMAT
        logl=fct(x,y)

INPUT
        x - vector of parameters of model

        y - one or more rows of the data set (if the data set has
            been transformed, or if vars /= 0, i.e., there is
            selection, then y is a transformed, selected observation)
            if __row == 1,  one row of the data set
            if __row >= 2,  if data set is stored in memory then
                            all of the data set will be passed to FCT;
                            if data set is stored in GAUSS data file
                            then __row will be passed to passed to
                            FCT.
            if __row <= 0, For data set is stored in memory same as __row>= 2,
                            for GAUSS data file the maximum number of
                            rows that will fit in memory will be
                            computed by MAXLIK.
           if _max_Lag >= 1,  a matrix of observations, the first is
                           the i-_max_Lag row, and the final row is
                           the i-th row.

OUTPUT
        logl - the log-likelihood
               if __row == 1 or _max_Lag >= 1,  a scalar value for
               a given observation, otherwise a vector of
               log-likelihoods.
```

REMARKS

If you have written the procedure such that it must compute the log-likelihood of one observation at a time then you must set __row = 1. But if you are able to write the procedure so that a vector of log-likelihoods may be returned then set __row=0; If you are getting "insufficient memory" messages when the data are being read from a GAUSS data file then either set __row ==1 or to some positive value. Also, if the data set is stored in a GAUSS data set and the selected data set will fit into memory, then MAXLIK will read it in and store it before beginning the iterations. In this case the setting of __row will follow the rules of a data set stored in memory. Significant reduction in computation time may be achieved when the data set can be stored in memory and procedure is written to compute vectors of log-probabilities.

Calling MAXLIK Recursively

The procedure that computes the log-likelihood may itself call MAXLIK. When calling MAXLIK recursively the following considerations apply:

If a data set is being analyzed and it is to be transformed or deleted for missing data or cases are to be selected, then this can be done only on the outermost version of MAXLIK, i.e., the version called in the original command file. Variable selection (as opposed to case selection) can be done on any level through the second argument in the call to each version of MAXLIK. Data sets can be opened by nested versions of MAXLIK. If a nested version of MAXLIK is going to use the data set opened by the outer version of MAXLIK then pass a null string (i.e., "") in the first argument in the call. If it is going to analyze a different data set from the outer version then pass it the data set name in a string. You may also load and store a data set in memory in the command file and pass it as the first argument in the nested call to MAXLIK.

Before the call to the nested version of MAXLIK, the global variables may be re-set by calling MAXCLR. You must not use MAXSET because that will clear information about the data sets opened and processed in the outer version. The only differences between MAXSET and MAXCLR are references to these globals.

You may also want to disable the keyboard control of the nested versions. This is done by setting the global _max_key = 0 after the call to MAXCLR and before the call to the nested MAXLIK.

PROC MAXLIK

```
FORMAT
       { x,f,g,cov,retcode } = MAXLIK(dataset,vars,&fct,start)

INPUT

 dataset - string containing name of GAUSS data set, or
           name of data matrix stored in memory

    vars - character vector of labels selected for analysis, or
           numeric vector of column numbers in data set
           of variables selected for analysis

     fct - the name of a procedure that returns either
           the log-likelihood for one observation or a vector of
           log-likelihoods for a matrix of observations

   start - a Kx1 vector of start values

OUTPUT
```

```
         x - Kx1 vector, estimated parameters
         f - scalar, function at minimum (mean log-likelihood)
         g - Kx1 vector, gradient evaluated at x
       cov - KxK matrix, covariance matrix of the parameters
   retcode - scalar, return code:

          0   normal convergence
          1   forced exit
          2   maximum number of iterations exceeded
          3   function calculation failed
          4   gradient calculation failed
          5   Hessian calculation failed
          6   step length calculation failed
          7   function cannot be evaluated at initial parameter values
          8   number of elements in the gradient vector inconsistent
              with number of starting values
          9   gradient function returned a column vector rather than
              the required row vector
         10   secant update failed
         11   maximum time exceeded
         12   weights could not be found
         20   Hessian failed to invert
         34   data set could not be opened
         99   termination condition unknown
```

GLOBAL VARIABLES

```
      ----- Options -----
_max_Options  - string array, specification of options,
                     default is equivalent to:

                 string _max_Options = { bfgs stepbt forward info screen }

    ----- Descent ------
_max_Algorithm  - scalar, determines descent algorithm (2)
_max_Delta      - scalar, floor for Hessian Eigenvalues in Newton (.1)

     ----- Line Search -----
_max_LineSearch - scalar, determines line search method (2)
_max_MaxTry     - scalar, maximum # of tries in step length methods (10
_max_Extrap     - scalar, extrapolation constant for BRENT (2.0)
_max_Interp     - scalar, interpolation constant for BRENT (.25)
_max_RandRadius - scalar, radius of random direction (0)
_max_UserSearch - scalar, enables user defined line search (0)

    ----- Covariance Matrix of Parameters -----
_max_CovPar    - scalar, determines type of covariance matrix of
                 parameters (1)
_max_XprodCov  - KxK matrix, cross-product covariance matrix of
                 parameters when _max_CovPar = 3
_max_HessCov   - KxK matrix, information matrix covariance matrix
                  of parameters when _max_CovPar = 3
_max_FinalHess - KxK matrix, stores hessian used for covariance

    ----- Gradients -----
_max_GradMethod  - determines type of numerical gradient (1)
_max_GradProc    - scalar, pointer to analytical gradient procedure (0)
_max_UserNumGrad - scalar, pointer to numerical gradient procedure (0)
_max_HessProc    - scalar, pointer to analytical hessian procedure (0)
_max_UserNumHess - scalar, pointer to numerical hessian procedure (0)
_max_GradStep    - scalar, increment size for computing gradient (0)
_max_GradCheck   - scalar, if nonzero, check analytical gradients (0)

   ----- Convergence Criteria -----
_max_GradTol  - scalar, convergence tolerance for gradient (1e-5)
_max_MaxIters - scalar, maximum number of iterations (1e+5)
```

```
_max_MaxTime  - scalar, maximum time in iterations in minutes (1e+5)

 ----- Data -----
_max_Active   - vector, defines fixed/active coefficients (1)
__weight      - vector, frequency of observations (1)
_max_Lag      - scalar, number of lags in model (0)
_max_NumObs   - scalar, rows of data matrix (output)
_max_ParNames - char. vector, parameter names (0)
__row         - scalar, # of rows of data set passed to procedures (0)
__rowfac      - scalar, proportion of rows of data set (1)

----- Miscellaneous -----
__title         - string, title ("")
_max_IterData   - 3x1 vector, elapsed time, # of iters, cov method
_max_Key        - scalar, controls keyboard trapping (0)
_max_Diagnostic - scalar, records current information from iterations
```

Options

```
_max_Options - string array, specification of options.  This global
               permits setting various MAXLIK options in a single global
               using string identifiers.  For example,

                string _max_Options = { brent newton central file };

                sets the line search method to BRENT, the descent method
                to NEWTON, the numerical gradient method to central
                differences, and __OUTPUT = 1.

                Algorithms:  STEEP, BFGS, DFP, NEWTON, BHHH, PRCG
                Line Search:  ONE, STEPBT, HALF, BRENT, BHHHSTEP
                Covariance Matrix :  NOCOV, INFO, XPROD, HETCON
                Gradient method:  CENTRAL, FORWARD
                Output method:  NONE, FILE, SCREEN
```

Descent

```
_max_Algorithm  -  scalar, indicator for optimization method:
             = 1,   SD (steepest descent)
             = 2,   BFGS (Broyden, Fletcher, Goldfarb, Shanno)
             = 3,   DFP (Davidon, Fletcher, Powell)
             = 4,   NEWTON (Newton-Raphson)
             = 5,   BHHH
             = 6,   Polak-Ribiere Conjugate Gradient

_max_Delta - scalar, floor for eigenvalues of Hessian in the NEWTON
             algorithm. This will insure that the Hessian will be
             positive definite.
```

Line Search

```
_max_LineSearch - scalar, indicator determining the line search method.

           = 1,  steplength = 1
           = 2,  STEPBT  (default)
           = 3,  HALF
           = 4,  BRENT
           = 5,  BHHHSTEP
```

```
            Usually _max_Step = 2 will be best.  If the optimization
            bogs down try setting _max_Step = 1 or 3.  _max_Step = 3
            will generate slow iterations but faster convergence and
            _max_Step = 1 will generate fast iterations but slower
            convergence.

_max_MaxTry - scalar, maximum number of tries in BRENT and GOLDEN.

_max_Extrap - scalar, extrapolation constant in BRENT.

_max_Interp - scalar, interpolation constant in BRENT.

_max_RandRadius - scalar, if _max_RandRadius is set to a nonzero
           value (1e-2, say) and all other line search methods fail then
            OPTMUM will attempt a random direction with radius
            determined by _max_RandRadius.

_max_UserSearch - scalar, if nonzero and if all other line search
            methods fail MAXLIK will enter an interactive mode in which
            the user can select a line search parameter.
```

Covariance Matrix of Parameters

```
_max_CovPar  -  scalar, type of covariance matrix of parameters,
             = 0,  the inverse of the final information matrix from
                   the optimization is returned in cov (default).
             = 1,  the inverse of the second derivatives is returned.

             = 2,  the inverse of the cross-product of the first
                   derivatives is returned.

             = 3,  the hetereskedastic-consistent covariance matrix
                   is returned.

_max_XprodCov - KxK matrix, when _max_CovPar is set to 3 the
                cross-product matrix covariance matrix of the
                parameters will be returned in _max_XprodCov.

_max_HessCov - KXK matrix, when _max_CovPar is set to 3 the
            information matrix covariance matrix of the parameters,
            i.e., the inverse of the matrix of second order partial
            derivatives of the log-likelihood, will be returned in
           _max_HessCov.

_max_FinalHess - KxK matrix, the Hessian used to compute the covariance
            matrix of the parameters will be stored in _max_FinalHess.
            This will be most useful if the inversion of the hessian
            fails, which is indicated when MAXLIK returns a
            missing value for the covariance matrix of the
            parameters.  An analysis of the Hessian stored in
            _max_FinalHess can then reveal the source of the linear
            dependency responsible for the singularity.
```

Gradients

```
_max_GradMethod  - scalar, method for computing numerical gradient.
              = 0, central difference
              = 1, forward difference (default)

_max_GradProc - scalar, pointer to a procedure that computes the
            gradient of the function with respect to the parameters.
            For example, the instruction:
```

```
                    _max_GradProc=&gradproc

            tells MAXLIK that a gradient procedure exists as well
            where to find it.  The user-provided procedure has
            two input arguments, a Kx1 vector of parameter values and
            an NxP matrix of data.  The procedure returns a single
            output argument, an NxK matrix of gradients of the log-
            likelihood function with respect to the parameters evaluated
            at the vector of parameter values.

            Default = 0, i.e., no gradient procedure has been provided.

_max_UserNumGrad - scalar, pointer to user provided numerical gradient
            procedure.  The instruction

                    _max_GradProc=&gradproc

            tells MAXLIK that a procedure for computing the
            numerical gradients exists. The user-provided procedure
            three input arguments, a pointer to a function that
            computes the log-likelihood function, a Kx1 vector of
            parameter values, and an NxP matrix of data.  The procedure
            returns a single output argument, an NxK matrix of
            gradients of each row of the input data matrix with
            respect to each parameter.

_max_HessProc - scalar, pointer to a procedure that computes the
            hessian, i.e., the matrix of second order partial derivatives
            of the function with respect to the parameters.  For example,
            the instruction:

                  _max_HessProc=&hessproc

            tells OPTMUM that a procedure has been provided for the
            computation of the hessian and where to find it.  The
            procedure that is provided by the user has two
            input arguments, a Kx1 vector of parameter values and an
            NxK data matrix.  The procedure returns a single
            output argument, the KxK symmetric matrix of second order
            derivatives of the function evaluated at the parameter
            values.

_max_UserNumHess - scalar, pointer to user provided numerical Hessian
            procedure.  The instruction

                    _max_GradProc=&hessproc

            tells MAXLIK that a procedure for computing the
            numerical Hessian exists. The user-provided procedure has
            three input arguments, a pointer to a function that
            computes the log-likelihood function, a Kx1 vector of
            parameter values, and an NxK matrix of data. The procedure
            returns a single output argument, a KxK Hessian matrix of
            the function with respect to the parameters.

_max_GradStep    - increment size for computing numerical gradient.

_max_GradCheck   - scalar, if nonzero and if proc's exist for
              computing the gradient or Hessian, their calculations
              will be compared with numerical gradients and Hessians
              in order to determine their correctness.
```

Convergence Criteria

```
_max_GradTol  - scalar, convergence tolerance for gradient of estimated
            coefficients.  Default = 1e-5.  When this criterion has been
```

```
           satisifed OPTMUM will exit the iterations.

_max_MaxIters - scalar, maximum number of iterations.

_max_MaxTime - scalar, maximum time in iterations in minutes.
           Default = 1e+5, about 10 weeks.
```

Data

```
_max_Active - vector, 0 = fixed coefficient, 1 = active coefficient.
              By default all coefficients are active.

__weight - vector, frequency of observations.  By default all
           observations have a frequency of 1.  zero frequencies
           are allowed.  It is assumed that the elements of __weight
           sum to the number of observations.

_max_Lag   - scalar, if the function includes lagged values of the
           variables _max_Lag may be set to the number of lags.  When
           _max_Lag is set to a nonzero value then __row is set to 1
           (that is, the function must evaluated one observation at a
           time), and MAXLIK will pass a matrix to the user-provided
           function and gradient procedures.  The first row in this
           matrix will be (i - _max_Lag)-th observation and the last
           row will be the i-th observation.  The read loop will begin
           with the (_max_Lag+1)-th observation.  Default = 0.

_max_NumObs  - scalar, number of cases in the data set that was analyzed.

_max_ParNames - Kx1 character vector, parameter labels.

__row    - determines the number of rows in the data set to be passed
           to the user-provided procedures. Default = 0.

__rowfac - If MAXLIK fails due to insufficient memory while attempting
           to read a GAUSS data set, then __rowfac may be set to some
           value between 0 and 1 to read a proportion of the original
           number of rows of the GAUSS data set.
```

Miscellaneous

```
__title  -  title of run

_max_Diagnostic - scalar. If 1, current estimates ("coeffs"),
                  gradient ("gradient"), direction ("direct"),
                  function value ("function"), Hessian ("Hessian"),
                  and step length computed in the line search ("step")
                  are printed to the screen.  If 2, they are
                  stored in _nlpmax_Diagnostic using VPUT.  Use
                  VREAD to extract.  If 3, both 1 and 2 occur.

_max_IterData - 3x1 vector, contains information about the iterations.
           The first element contains the elapsed time in minutes of the
           iterations, the second element contains the # of iterations,
           and the third element contains a character variable indicating
           the type of covariance matrix of the parameters.

_max_Key   - scalar, controls keyboard capture.  Useful for recursively
           nested version of MAXLIK.  Setting _max_key = 0 for the
           nested versions will turn off their key board captures
           permitting the outside version to retain control of the
           keyboard.
```
