Author(s) : W. Mönch, B. Schorr **Library:** MATHLIB
Submitter : W. Mönch **Submitted:** 15.03.1993
Language : Fortran **Revised:**

Constrained Non-Linear Least Squares and Maximum Likelihood Estimation

`LEAMAX` is a portable collection of subprograms for solving general non-linear least squares problems, non-linear data fitting problems, and maximum likelihood estimations.

Subroutine subprograms `RSUMSQ`, `RFUNFT`, `RMAXLK` and `DSUMSQ`, `DFUNFT`, `DMAXLK` calculate an approximation to a minimum of an objective function φ , with respect to n unknown parameters $a = (a_1, ..., a_n)^{\mathbf{T}} \in \mathbf{R^n}$:

(S) The general non-linear least squares problem

$$\varphi_S(a) \;=\; \frac{1}{2}\sum_{i=1}^{m}[f_i(a)]^2,$$

(F) The least squares data fitting problem

$$\varphi_F(a) \;=\; \frac{1}{2}\sum_{i=1}^{m}\left[\frac{y_i - f(x_i, a)}{\sigma_i}\right]^2,$$

(M) The maximum likelihood estimation

$$\varphi_M(a) \;=\; -\sum_{i=1}^{m}\ln(f(x_i, a)),$$

subject to bounds on the variables of the form

$$\underline{a}_j \le a_j \le \overline{a}_j \qquad (j = 1, 2, \ldots, n).$$

The functions $f_i : \mathbf{R^n} \to \mathbf{R^1}$ $(i = 1, ..., m)$ and $f : \mathbf{R^k} \times \mathbf{R^n} \to \mathbf{R^1}$ are arbitrary non-linear functions with respect to the argument a. In the case of the data fitting problem **(F)**, a set of observation data $\{(x_i, y_i)|x_i \in \mathbf{R^k}, y_i \in \mathbf{R^1}, i = 1, ..., m\}$ with their corresponding weights σ_i $(i = 1, ..., m)$ has to be provided, whereas for the maximum likelihood estimation **(M)**, the set of data points $\{(x_i)|x_i \in \mathbf{R^k}, i = 1, ..., m\}$ belongs to the input of the problem.

These subprograms are based on the algorithm described by Moré (Ref. 1) for finding the solution of a general non-linear least squares problem by using the Levenberg-Marquardt algorithm. The method is completed by an active set strategy for handling simple box constraints to the unknown parameters (see **Long Write-up** for details). The necessary derivatives can either be supplied by the user (subprogram `SUB`) or are approximated numerically. In the case of a non-linear data fitting problem, approximations to the covariance matrix and the standard deviations of the model parameter estimators are also provided.

On computers other than CDC or Cray, only the double-precision versions `DSUMSQ`, `DFUNFT`, `DMAXLK` are available. On CDC and Cray computers, only the single-precision versions `RSUMSQ`, `RFUNFT`, `RMAXLK` are available.

Structure:

SUBROUTINE subprograms
User Entry Names: RSUMSQ, RFUNFT, RMAXLK, DSUMSQ, DFUNFT, DMAXLK
Internal Entry Names: D501L1, D501L2, D501SF, D501P1, D501P2, D501N1, D501N2
External References: RGEQPF (F001), RORMQR (F001), RTRTRS (F001), DGEQPF (F001), DORMQR (F001), DTRTRS (F001), RVSET (F002), RVSCL (F002), RVSUB (F002), RVCPY (F002), RVMPY (F002), DVSET (F002), DVSCL (F002), DVSUB (F002), DVCPY (F002), DVMPY (F002), RMSET (F003), RMSCL (F003), RMCPY (F003), RMMPY (F003), RMBIL (F003), DMMLT (F003), DMSET (F003), DMSCL (F003), DMCPY (F003), DMMPY (F003), DMBIL (F003), RMMLT (F004), DMMLT (F004), RSINV (F012), DSINV (F012)
User-supplied SUBROUTINE subprogram

Usage:

For t $=$ R (type REAL), t $=$ D (type DOUBLE PRECISION):

(S) General non-linear least squares problem

```
      CALL tSUMSQ(SUB,M,N,NC,A,AL,AU,MODE,EPS,MAXIT,IPRT,
     +            MFR,IAFR,PHI,DPHI,COV,STD,W,NERROR)
```

(F) Least squares data fitting problem

```
      CALL tFUNFT(SUB,K,M,N,NX,NC,X,Y,SY,A,AL,AU,MODE,EPS,MAXIT,IPRT,
     +            MFR,IAFR,PHI,DPHI,COV,STD,W,NERROR)
```

(M) Maximum likelihood estimation

```
      CALL tMAXLK(SUB,K,M,N,NX,X,A,AL,AU,MODE,EPS,MAXIT,IPRT,
     +            MFR,IAFR,PHI,DPHI,W,NERROR)
```

SUB Name of user-supplied SUBROUTINE subprogram, declared EXTERNAL in the calling program. This subprogram must provide the values of the functions $f_i(a)$ $(i = 1, \ldots, m)$, $f(\cdot, a)$, and, if MODE $= 1$, the values of the derivatives $\partial f_i(a)/\partial a_j$ and $\partial f(\cdot, a)/\partial a_j$ $(i = 1, \ldots, m; j = 1, \ldots, n)$ (see **Examples**).

K (INTEGER) Cases **(F)** and **(M)**: dimension k of a data point (observation) $x_i \in \mathbf{R^k}$.

M (INTEGER) Case **(S)**: number of non-linear functions f_i; cases **(F)** and **(M)**: number of data points (observations).

N (INTEGER) Number of unknown parameters a.

NX (INTEGER) Cases **(F)** and **(M)**: declared first dimension of array X in the calling program, NX $\geq$ K.

NC (INTEGER) Cases **(S)** and **(F)**: declared first dimension of array COV in the calling program, NC $\geq$ N.

X (Type according to t) Cases **(F)** and **(M)**: two-dimensional array of dimension (NX, $\geq$ M). On entry, X must contain the data set $\{ x_i \}$ (the i-th column of X belongs to the data point $x_i \in \mathbf{R^k}$, $i = 1, ..., m$).

Y (type according to t) Case **(F)**: one-dimensional array of length $\geq$ M, contains, on entry, the data set $\{ y_i \}$.

SY (type according to t) Case **(F)**: one-dimensional array of length $\geq$ M, contains, on entry, the weights $\{ \sigma_i \}$ of the data points.

A (Type according to t) One-dimensional array of length $\geq$ N. On entry, A(J) must contain the starting value of a_j for the Levenberg-Marquardt algorithm. On exit, A(J) contains an approximation to a_j of a minimum point (if the algorithm was successful).

AL (Type according to t) One-dimensional array of length $\geq$ N. On entry, AL(J) must contain the lower bound $\underline{a}_j$ of a_j .

AU (Type according to t) One-dimensional array of length $\geq$ N. On entry, AU(J) must contain the upper bound $\overline{a}_j$ of a_j .

MODE (INTEGER)

= 0 : The derivatives are approximated by divided differences.

= 1 : The derivatives are to be supplied by subprogram SUB.

Other values for MODE are treated as MODE = 0.

EPS (Type according to t) User-supplied tolerance used to control the termination criterion. EPS should be chosen according to the accuracy required by the problem and the machine accuracy t (recommended value on entry: between 10^{-6} for t = R , and 10^{-12} for t = D, respectively).

MAXIT (INTEGER) Maximum permitted number of iterations.

IPRT (INTEGER) Printing control.

= 0 : no printing of intermediate results,

= ±L : printing of intermediate results at every |L|-th iteration; if IPRT < 0, printing of all input parameters of tSUMSQ, tFUNFT, tMAXLK, respectively, in addition.

MFR (INTEGER) On exit, MFR contains the number of free variables at the solution point.

IAFR (INTEGER) One-dimensional array of length $\geq 2 *$ N for cases **(S)** and **(F)**, and of length $\geq$ N for case **(M)**, used as working space. On exit, the first MFR elements of IAFR contain the indices of the free variables at the solution point.

PHI (Type according to t) On exit, PHI contains the value of the objective function φ at the solution point.

DPHI (Type according to t) One-dimensional array of length $\geq$ N. On exit, DPHI(J) contains the derivative $\partial\varphi/\partial a_j$ of φ with respect to a_j (j-th component of the gradient of φ) at the solution point.

COV (Type according to t) Cases **(S)** and **(F)**: two-dimensional array of dimension (NC, $\geq$ N). On exit, COV contains an approximation to the covariance matrix.

STD (Type according to t) Cases **(S)** and **(F)**: one-dimensional array of length $\geq$ N. On exit, STD(J) contains an approximation to the standard deviation of the estimator of the model parameter a_j.

W (Type according to t) One-dimensional array of length $\geq 9*\mathrm{N}+4*\mathrm{M}+2*\mathrm{M}*\mathrm{N}+3*\mathrm{N}*\mathrm{N}$ for cases **(S)** and **(F)**, and of length $\geq 7*\mathrm{N}+2*\mathrm{N}*\mathrm{N}$ for case **(M)**, used as working space.

NERROR (INTEGER) Error indicator. On exit:

= 0 : No error or warning detected.

= 1 : At least one of the constants K, M, N, NX, NC, MAXIT is illegal or at least for one j the relation $\underline{a}_j \leq \overline{a}_j$ is not true.

= 2 : The maximum number MAXIT of iterations has been reached.

= 3 : The objective function φ or its derivative is not defined for the current values of the parameter vector a.

= 4 : Cases **(S)** and **(F)**: The routines tGEQPF, tORMQR, tTRTRS in the Linear Algebra package LAPACK (F001) were unable to solve the linear least squares problem or the subprogram tSINV (F012) was unable to compute the covariance matrix.
Case **(M)**: the routine tSINV (F012) was unable to solve the normal equations.

Examples:

For the user-supplied `SUBROUTINE` subprogram `SUB` write for instance in the case `t` = `D`:

(S) Objective function (Brown badly-scaled function, $n = 2,\ m = 3$):

$$\varphi_S(a) \;=\; \frac{1}{2}\,\left[(a_1 - 10^6)^2 + (a_2 - 2 \cdot 10^{-6})^2 + (a_1 a_2 - 2)^2\right]$$

.

```
      SUBROUTINE SUB(N,A,M,F,DF,MODE,NERROR)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (ZO = 0)
      DIMENSION A(*),F(*),DF(M,*)
      NERROR=0
      F(1)=A(1)-1D6
      F(2)=A(2)-2D-6
      F(3)=A(1)*A(2)-2
      IF(MODE .NE. 1) RETURN
      CALL DMSET(M,N,ZO,DF(1,1),DF(1,2),DF(2,1))
      DF(1,1)=1
      DF(2,2)=1
      DF(3,1)=A(2)
      DF(3,2)=A(1)
      RETURN
      END
```

(F) Objective function (Bard function, $n = 3,\ m = 15,\ k = 3$):

$$\varphi_F(a) \;=\; \frac{1}{2}\sum_{i=1}^{m}\left[y_i - \left(a_1 + \frac{x_{1,i}}{x_{2,i}\,a_2 + x_{3,i}\,a_3}\right)\right]^2$$

.

```
      SUBROUTINE SUB(K,X,N,A,F,DF,MODE,NERROR)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION A(*),X(*),DF(*)
      T=X(2)*A(2)+X(3)*A(3)
      IF (T .EQ. 0) THEN
       NERROR=3
      ELSE
       NERROR=0
       F=A(1)+X(1)/T
       IF(MODE .NE. 1) RETURN
       DF(1)=1
       DF(2)=-X(1)*X(2)/T**2
       DF(3)=-X(1)*X(3)/T**2
      ENDIF
      RETURN
      END
```

(M) Objective function ($n = 1,\ m = 100,\ k = 1$):

$$\varphi_M(a) \ = \ -\sum_{i=1}^{m} \ln \left\{ \frac{1}{a_1\sqrt{\pi}} \exp\left[-\frac{1}{2}\left(\frac{x_{1,i}-1}{a_1}\right)^2 \right] \right\}$$

.

```
SUBROUTINE SUB(K,X,N,A,F,DF,MODE,NERROR)
IMPLICIT DOUBLE PRECISION (A-H,O-Z)
DIMENSION A(*),X(*),DF(*)
PARAMETER (PIR = 0.56418 95835 47756 287D0)
NERROR=3
IF(A(1) .LE. 0) RETURN
T=0.5D0*((X(1)-1)/A(1))**2
F=PIR*EXP(-T)/A(1)
NERROR=0
IF(MODE .EQ. 1) DF(1)=-F*(1-2*T)/A(1)**2
RETURN
END
```

In all three cases the parameters `K` , `N` , `A` , `M` , `MODE` , `NERROR` are as declared above. The other parameters are the following:

`F` (Type according to `t`) Case **(S)**: one-dimensional array of length $\geq$ `M`. `F(I)` must contain the function value $f_i(a)$ at a $(i = 1, \ldots, m)$, on exit.
Cases **(F)** and **(M)**: `F` must contain the function value $f(x, a)$ at (x, a), on exit.

`DF` (Type according to `t`) If `MODE = 1` values of `DF` are supplied by `SUB`. For other values of `MODE` the parameter `DF` is not referenced.
Case **(S)**: two-dimensional array of dimension (`M`, $\geq$ `N`). `DF(I,J)` must contain the value of the partial derivative $\partial f_i(a)/\partial a_j$ at a, $(i = 1, \ldots, m; j = 1, \ldots, n)$, on exit.
Cases **(F)** and **(M)**: one-dimensional array of length $\geq$ `N`. `DF(J)` must contain the value of the partial derivative $\partial f(x, a)/\partial a_j$, $(j = 1, \ldots, n)$, on exit.

`X` (Type according to `t`) Cases **(F)** and **(M)**: one-dimensional array of length $\geq$ `K` for **one** data point $x_i \in \mathbf{R^k}$ (in contrast to above declaration).

References:

1. J.J. Moré, The Levenberg-Marquardt algorithm: Implementation and theory, In: Numerical Analysis, G.A. Watson (Ed.), Lecture Notes in Mathematics 630, Springer-Verlag, New York (1977) 105-116.

2. Å.Björck: Solution of Equations in $\mathbf{R^n}$ (Part 1: Least Squares Methods). In: Handbook of Numerical Analysis, P.G.Ciarlet, J.L.Lions (Eds.), North-Holland, Amsterdam, New York, Oxford, Tokyo, 1990, 467-636.

3. R.Fletcher: Practical Methods of Optimization. John Wiley and Sons, Chichester, 2nd Edition, 1987.

•