

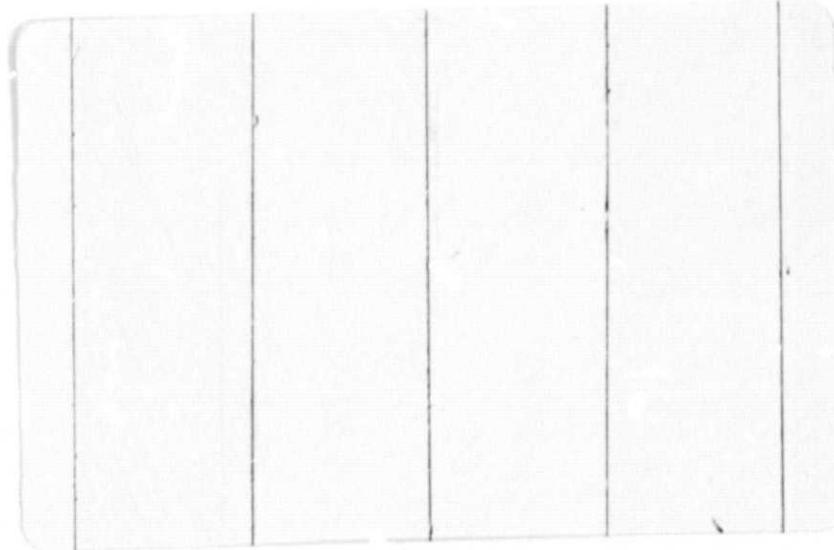
## General Disclaimer

### One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.
- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.
- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.
- This document is paginated as submitted by the original source.
- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

NASA CR-

147825



(NASA-CR-147825) A QUASI-NEWTON APPROACH TO  
OPTIMIZATION PROBLEMS WITH FEASIBILITY  
DENSITY CONSTRAINTS (Rice Univ.) 41 p  
HC \$4.00

CSC1 12A

G3/65 Unclas 44567

N70-27943



I C S A

INSTITUTE FOR COMPUTER SERVICES AND APPLICATIONS

RICE UNIVERSITY

A Quasi-Newton Approach to  
Optimization Problems with  
Probability Density Constraints

by

R. A. Tapia and D. L. Van Rooy  
Rice University

ABSTRACT:

A quasi-Newton method is presented for minimizing a non-linear functional while constraining the variables to be nonnegative and sum to one. The nonnegativity constraints are eliminated by working with the squares of the variables and the resulting problem is solved using Tapia's general theory of quasi-Newton methods for constrained optimization. A user's guide for a program implementing this algorithm is provided.

Institute for Computer Services and Applications  
Rice University  
Houston, TX 77001  
June, 1976

This work was sponsored in part by NASA contract NAS 9-12776

A Quasi-Newton Approach to  
Optimization Problems with  
Probability Density Constraints

by

R. A. Tapia and D. L. Van Rooy  
Rice University

ABSTRACT:

A quasi-Newton method is presented for minimizing a nonlinear functional while constraining the variables to be nonnegative and sum to one. The nonnegativity constraints are eliminated by working with the squares of the variables and the resulting problem is solved using Tapia's general theory of quasi-Newton methods for constrained optimization. A user's guide for a program implementing this algorithm is provided.

Institute for Computer Services and Applications  
Rice University  
Houston, TX 77001  
June, 1976

This work was sponsored in part by NASA contract NAS 9-12776

## I. INTRODUCTION

Consider the optimization problem

$$(1) \quad \begin{aligned} \min f(x_1, \dots, x_n) ; \text{ subject to } 1 - \sum_{i=1}^n x_i &= 0, \\ x_i &\geq 0, \quad i=1, \dots, n \end{aligned}$$

Problems of the form (1) occur frequently in statistical applications.

In these applications, the variables  $x_i$  usually represent a discrete probability density function and the functional  $f$  represents an optimality criterion, e.g., the negative likelihood or negative log likelihood.

Let us make the change of variables

$$(2) \quad x_i = \frac{1}{2} y_i^2, \quad i=1, \dots, n$$

and consider the optimization problem

$$(3) \quad \min \hat{f}(y_1, \dots, y_n) ; \text{ subject to } 1 - \sum_{i=1}^n \frac{1}{2} y_i^2 = 0$$

where

$$(4) \quad \hat{f}(y_1, \dots, y_n) = f(\frac{1}{2} y_1^2, \dots, \frac{1}{2} y_n^2)$$

### Proposition 1.1

If  $y = (y_1, \dots, y_n)^T$  solves problem (3), then  $x = (\frac{1}{2} y_1^2, \dots, \frac{1}{2} y_n^2)^T$  solves problem (1). Conversely, if  $x = (x_1, \dots, x_n)^T$  solves problem (1), then  $y = (\sqrt{2x_1}, \dots, \sqrt{2x_n})^T$  solves problem (3).

### Proof

The proof is not difficult.

Problem (3), in contrast to problem (1), does not involve inequality constraints. The price one pays for this simplification is that problem (3) will have more critical points than problem (1) and the equality constraint is quadratic instead of linear. We maintain, that according to Proposition 1.1, if one starts sufficiently near the solution, the extraneous critical points will not affect the algorithm. Moreover, when  $f$  is nonlinear, the quadratic constraints should not significantly increase the degree of complexity of the problem.

Tapia's approach to quasi-Newton methods for constrained optimization ([1] and [2]) depends heavily on the invertibility of the Hessian of the Lagrangian at the solution. Hence, we are certainly interested in determining the relationship between the invertibility of the Hessian of the Lagrangian for problem (1) and the invertibility of the Hessian of the Lagrangian for problem (3). Toward this end let

$$(5) \quad L(x, \mu, \lambda) = f(x) - \sum_{i=1}^n \mu_i x_i + \lambda(1 - \sum x_i)$$

and

$$(6) \quad \hat{L}(y, \lambda) = \hat{f}(y) + \lambda(1 - \sum_{i=1}^n \frac{1}{2} y_i^2)$$

we use the notation

$$\Lambda = (\lambda_1, \dots, \lambda_n)^T,$$

$$Y = \text{diag}(y_1, \dots, y_n)$$

and

$$X = \text{diag}(x_1, \dots, x_n)$$

Straightforward calculations show that

$$(7) \quad \nabla_y \hat{L}(y, \lambda) = Y (\nabla f(x) - \Lambda) ,$$

$$(8) \quad \nabla_y^2 L(y, \lambda) = Y \nabla^2 f(x) Y + \text{diag}(\nabla f(x) - \Lambda) ,$$

$$(9) \quad \nabla_x L(x, \mu, \lambda) = \nabla f(x) - \Lambda - \mu ,$$

and

$$(10) \quad \nabla_x^2 L(x, \mu, \lambda) = \nabla^2 f(x) .$$

The first order necessary conditions for problem (1) are

$$(11) \quad \begin{aligned} \nabla f(x) - \Lambda - \mu &= 0 \\ 1 - \sum_{i=1}^n x_i &= 0 \\ \mu_i &\geq 0 \\ x_i &\geq 0 \\ \mu_i x_i &= 0, \quad i=1, \dots, n ; \end{aligned}$$

while the first order necessary conditions for problem (3) are

$$(12) \quad \begin{aligned} Y (\nabla f(x) - \Lambda) &= 0 \\ 1 - \sum_{i=1}^n \frac{1}{w_i} y_i^2 &= 0 \end{aligned}$$

Recall that strict complementarity means that in (11) we do not have  $\mu_i = x_i = 0$  for any  $i$ .

Proposition 1.2

Let  $(x, u, \lambda)$  be a solution of problem (1) and  $(y, \lambda)$  the corresponding solution of problem (3). Then

- (i) we have strict complementarity and  $\nabla^2 f(x)$  is positive definite  $\Leftrightarrow \nabla_y^2 \hat{L}(y, \lambda)$  is positive definite
- (ii) we have strict complementarity and  $\nabla^2 f(x)$  is invertible  $\Leftrightarrow \nabla_y^2 \hat{L}(y, \lambda)$  is invertible.

Proof

Again the proof is not difficult. One merely works with (8), (10), (11) and (12).

Proposition 1.2 is very satisfying and warns us that in the algorithm we must guard against the loss of strict complementarity. This will be accomplished by working with the variable

$$(13) \quad x_i = \frac{1}{E_i} y_i^{E_i}$$

where  $E_i = 2$  if  $|x_i| + |\mu_i| \neq 0$  and  $E_i = 1$  if  $|x_i| + |\mu_i| = 0$ . Observe that, when  $E_i = 1$  in (13), we have effectively ignored the nonnegativity constraint on  $x_i$ .

II. THE QUASI-NEWTON ALGORITHM

The algorithm we are about to present is a special case of the approach given by Tapia in [2] applied to problem (3). The reader is referred to that paper for a better understanding of the theory and algorithm. In describing an iterative method, it is convenient to denote the present iterate by  $x$  and the subsequent iterate by  $\bar{x}$  and similarly for other variables. Also, we let  $I$

denote the identity matrix  $I$ ,  $\langle \cdot, \cdot \rangle$  denotes the Euclidean inner product,  $e_1 = (1, 0, \dots)^T, \dots, e_n = (0, \dots, 1)^T$ ,  $(a_{ij})$  denotes the matrix whose  $(i, j)$ -th element is  $a_{ij}$  and  $b_i$  or  $(b)_i$  denotes the  $i$ -th element of the vector  $b$ .

Consider the constrained optimization problem

$$(14) \quad \min f(x), \quad \text{subject to } g(x) = 0;$$

where  $f, g : \mathbb{R}^n \rightarrow \mathbb{R}^1$ . We first describe the quasi-Newton method BFGS given in [2] for problem (14). Let

$$(15) \quad L(x, \lambda) = f(x) + \lambda g(x)$$

### Quasi-Newton BFGS

Step 1: (Initialization) Determine  $x$  and  $B$ .

Step 2: (Update  $x$  and  $\lambda$ ) Let

$$\bar{\lambda} = \frac{g(x) - \langle \nabla g(x), B^{-1} \nabla f(x) \rangle}{\langle \nabla g(x), B^{-1} \nabla g(x) \rangle}$$

$$\bar{x} = x - B^{-1} \nabla_x L(x, \bar{\lambda}).$$

Step 3: (Update approximate Hessian)

$$\bar{B} = \left( B + \frac{y y^T}{\langle y, s \rangle} - \frac{B s s^T B}{\langle s, Bs \rangle} \right)$$

where

$$s = \bar{x} - x$$

and

$$y = \nabla_x L(\bar{x}, \bar{\lambda}) - \nabla_x L(x, \bar{\lambda}).$$

Step 4: GO TO STEP 2.

In implementing this algorithm, one would have to make provisions for output and a stopping criterion. In [2], Tapia has demonstrated that the above algorithm is locally superlinearly convergent.

Remark 1:

If the initial  $B$  depended on  $\lambda$ , we would make an initial approximation of  $\lambda$  according to the projection formula (see [2]):

$$(16) \quad \lambda = - \frac{\langle \nabla g(x), \nabla f(x) \rangle}{\langle \nabla g(x), \nabla g(x) \rangle}$$

Remark 2:

In updating  $B$  in Step 3, it is advantageous to store  $B$  factored according to a Cholesky decomposition and then merely update these factors. Moreover,  $B$  can be slightly modified so that the resultant matrix is always positive definite. For a detailed description of these modifications, the reader is referred to Gill et al [3], Van Rooy et al [4], Fletcher and Powell [5] and Van Rooy [6]. The quantities  $B^{-1} \nabla f(x)$  and  $B^{-1} \nabla g(x)$  in Step 2 are then obtained in the standard back-substitution manner.

We now apply the above algorithm for problem (3) with the additional feature of obtaining a discrete Newton approximation to the Hessian every  $M$  iterations (where  $M$  is an input parameter).

Define the following functions

$$(17) \quad x_i = \frac{y_i}{E_i}$$

$$(18) \quad \nabla g_i = - [2 - E_i + (E_i - 1)y_i], \\ i=1, \dots, n$$

and

$$(19) \quad g = 1 - \sum_{i=1}^n x_i$$

The quantity  $\nabla g$  is merely the gradient of  $g$  with respect to  $y$  and (13) and (17) are the same.

### Quasi-Newton Algorithm for Problem (3)

#### Step 1: (Initialization)

Read	CRC	(constraint refection criterion)
	h	(discretization step)
	$\epsilon$	(stopping criterion)
	DI	(descent indicator)
	IOUTPUT	(output indicator)
	M	(discrete Hessian indicator)
	MAX	(maximum number of iterations)

Let

$$E_i = 2 \\ y_i = \sqrt{\frac{2}{n}}$$

$$\lambda = \langle \nabla f(x), x \rangle$$

$$B = I$$

#### Step 2: Output according to IOUTPUT, check stopping criterion

IF  $(\|\nabla_y \hat{L}(y, \lambda)\| + g^2 < \epsilon \text{ or}$   
 ITERATION > MAX ) stop

Step 3: Calculate discrete Hessian according to M

$$B = \left( \frac{\nabla_y \hat{L}(y + h e_j, \lambda)_i - \nabla_y \hat{L}(y, \lambda)_i}{h} \right)$$

Calculate modified Cholesky decomposition of B.

Step 4: Calculate multiplier correction

$$\Delta \lambda = \frac{g - \langle \nabla g, B^{-1} \nabla_y \hat{L}(y, \lambda) \rangle}{\langle \nabla g, B^{-1} \nabla g \rangle}$$

Step 5: Calculate y-variable correction

$$\Delta y = - B^{-1} \nabla_y \hat{L}(y, \lambda) - \Delta \lambda B^{-1} \nabla g$$

Step 6: Update y according to descent indicator

$$\bar{\lambda} = \lambda + \alpha \Delta \lambda$$

$$\bar{y} = y + \alpha \Delta y$$

where  $\alpha = 1$  if descent is not desired, otherwise  $\alpha$  is chosen according to the descent principle on  $\|\nabla_y \hat{L}(y, -\lambda)\|^2 + g^2$  (see [2]).

Step 7: Update approximate Hessian according to BFGS  
 and modified Cholesky decomposition

$$\tilde{B} = B + \frac{Dy \ Dy^T}{\langle Dy, Ds \rangle} - \frac{B \ Ds \ Ds^T B}{\langle Ds, B Ds \rangle}$$

where  $Ds = \bar{x} - x$  and

$$Dy = \nabla_y \hat{L}(\bar{y}, \bar{\lambda}) - \nabla_y \hat{L}(y, \bar{\lambda})$$

Step 8: Update constraint rejection indicators

$$T_0 = a_1 \left[ \| \nabla_y \hat{L}(\bar{y}, \bar{\lambda}) \|^2 + \left( 1 - \sum_{i=1}^n x_i \right)^2 \right]^{a_2},$$

$$T_1 = \min \left( \frac{1}{4n}, T_0 \right),$$

$$T_2 = \min (CRC, T_0)$$

$$\text{If } ( |\bar{x}_i| < T_1 \text{ and } | \frac{\partial f}{\partial x_i} - \lambda | < T_2 ),$$

$$\text{let } E_i = 1$$

$$\text{If } E_i \text{ has changed, reset } \bar{y}_i \quad i=1, \dots, n$$

Step 9: GO TO STEP 2.

### III. USER'S GUIDE

The above algorithm has been implemented in a set of FORTRAN subroutines whose entry point is the subroutine CONOPT. In this section, we shall describe how to use this program, provide a description of the workings of the program, and illustrate its use with examples. A listing of the program is given in the appendix. This program runs on an IBM 370/155 using the FORTRAN G, G1, or H compilers. All computations are done in double precision. Unit 6 is used for output, if requested.

To utilize this program, the user supplies several parameters and two subprograms which calculate the function value and its gradient at a specified point. The program will then attempt to minimize the function subject to the constraints and return the argument at the minimum.

The calling sequence of the program is CALL CONOPT (N, DELF, FUNCT, X, WKA, MITER, STC, CRC, IDN, DH, IOUT, DSNT, A, RSTRT, USELAM, XLAM).

These parameters are described below:

N - integer variable indicating the dimension of the problem.

DELF - a user supplied subroutine which calculates the gradient of the function. The calling sequence is CALL DELF (DF, X, N), where X is the N dimensional argument and DELF returns the gradient in the vector DF. Note that DF and X are both double precision variables.

FUNCT - a user supplied double precision function subprogram which calculates the function value at a specified point. The calling sequence is  $F = \text{FUNCT}(X, N)$  where  $X$  is the (double precision) argument of length  $N$ . Though the function value is not required by the algorithm, CONOPT will output it in the convergence test step if the user requests output. (It is called only when output is requested.) If, for any reason, the user is not interested in this value, he may just supply a function subprogram that sets the function value to zero and returns.

[N.B. The two programs DELF and FUNCT must be declared external in the calling program. The two programs may have other entry points, with different arguments, referred to by the user's calling program prior to his calling CONOPT in order to make other variables or their addresses available to either or these subprograms.]

X - on output, this will contain the  $N$ -dimensional double precision argument last calculated by the program (at the minimum if the program converged). On input, the program will set each element of  $X$  to the value  $1/N$ , unless the user has specified restart (see description of the variable RSTART below), in which case

the supplied values of  $X$  will be used as the starting point.

- WKA - a double precision working storage array of length  $\geq \frac{1}{2} N^2 + (23N + 7) / 2$ . If a restart has been specified, the first  $N$  (2-byte integer) locations should contain the desired constraint rejection indicators. However, if these variables do not contain either 1's or 2's, the program will set them to 2. The final constraint rejection indicators will be in these locations.
- MITER - an integer variable indicating the maximum number of iterations to perform. If the user sets this to 0, the program will reset it to max (30,  $2 * N$ ).
- STC - a real variable specifying the stopping criterion on the norm squared of the gradient of the Lagrangian plus the constraint, i.e.  $\| \nabla L \|_2^2 + \left( 1 - \sum_{i=1}^n x_i \right)^2$ . If 0 is specified, the program will set  $STC = 10^{-8}$ .
- CRC - a real variable specifying the constraint rejection criterion. See the algorithm description section for a precise definition of its use. If 0, is specified, the program will set  $CRC = 10^{-2}$ .
- IDN - an integer variable specifying the frequency of calculation of the discrete approximation to the Hessian, i.e., this finite difference Hessian is calculated every IDN iterations. If 0 is specified, this approximation is never used.

If a negative value is specified, the program sets IDN = min ( 10 , N ).

DH - a real variable specifying the step size to be used in calculating the finite difference Hessian (see the algorithm description for a detailed definition). If 0 is specified, the program will set DH =  $10^{-2}$ .

IOUT - an integer variable specifying the frequency of output; i.e., the program will output various computed quantities every IOUT iterations. If 0 is specified, no output will be produced. If a negative value is specified, the program sets IOUT = 1.

DSNT - a one-byte logical variable indicating whether the program should implement descent in its search at each iteration.

A - a two-word real vector containing respectively the multiplier and exponent for DNORM  

$$\left( = \| \nabla L \| ^2 + ( 1 - \sum_{i=1}^n x_i ) ^2 \right)$$
 for use in the constraint rejection test; i.e., the appropriate quantity is A(1) \* DNORM \* \* A(2). The defaults for A(1) and A(2) are 1. and .2 respectively. If the user wishes to set A(2) = 0., he should set A(1) to the negative of what he wants for A(1) and then a value of zero will be used for A(2).

RSTART - a one-byte logical variable indicating whether the user is supplying an initial value for X (as in a restart case). If RSTART = .TRUE. , the first  $2N$  bytes of WKA should contain the vector ICR (see description of WKS and the algorithm description—the  $E_i$ 's ).

USELAM - a one-byte logical variable used if RESTR = . TRUE. to indicate if the user is supplying a value for XLAM . If RESTR = . TRUE. and USELAM = . FALSE. , the value  $\langle \nabla f, x \rangle$  is used for XLAM .

XLAM - the Langrange multiplier for the equality constraint. Used on input only if USELAM = RSTART = . TRUE.

The default values for the above parameters have been chosen so that many problems will converge; thus, the only input parameters with which the user need concern himself in many cases is N, IOUT, DSNT and RSTART (usually = . FALSE.).

The subprograms used and a brief description of their function is given below:

CONOPT - main driver subroutine

DIMS - computes base addresses for various working arrays used throughout the calculations.

INPT - sets default values for parameters and outputs the parameters used.

INIT initializes y, x, ICR, H and  $\lambda$  . Also fetches values of  $\nabla f$  and  $\nabla L$  .

- CHKSTP - checks the stopping criterion and outputs quantities if requested.
- DHESS - calculates a finite difference approximation to the Hessian and its modified Cholesky decomposition, if requested. The Hessian is forced to be positive definite.
- DELAMY - computes the update terms  $\nabla \lambda$  and  $\nabla y$ .
- NEWY - updates  $y$  and  $\lambda$  based optionally on descent of  $\| \nabla L \|^2 + \left( 1 - \sum_{i=1}^n x_i \right)^2$ .
- UPDTH - updates the approximate Hessian using the Broyden-Fletcher-Goldfarb-Shannon (BFGS) update technique. (Really the MCD of it is updated.) Here again the Hessian is forced to be positive definite.
- CONSTR - updates the constraint rejection indicators and, if necessary,  $y$  and  $\nabla L$ .
- GRADL - computes  $\nabla L$  given  $\nabla f$ ,  $\lambda$ ,  $y$ , and ICR.
- SUM - a function subprogram that performs various utility functions  $( \sum_{i=1}^n x_i, \sum_{i=1}^n x_i^2, \sum_{i=1}^n x_i y_i )$ .
- MCHLSK - computes the modified Cholesky decomposition (MCD) of a matrix. If the matrix is not positive definite, the MCD is modified to represent a matrix that is (see e.g. ref 4).
- PDSOLV - solves  $Ax = y$  where  $A$  contains the MCD of a matrix.

COMPT - computes a rank one update of an MCD using the composite-t method (see refs. 5 and 6).

LDMUL - multiply a vector by a matrix whose MCD is supplied.

Examples:

The following quadratic programming (i.e.,  $\min \| Ax - b \|^2$  s. t.  $x_i \geq 0$  and  $\sum x_i = 1$ ) examples were run using all the default options:

$$1. \quad A = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$x = (.111112108, .444443959, .444443959)^T$$

converged in 5 iterations. Exact answer =  
 $(1/9, 4/9, 4/9)^T$

The robustness of the algorithm is exemplified by the following two examples for which the algorithm, theoretically, should not converge since the Hessian is singular at the solution.

$$2. \quad A = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 2 & 1 & 2 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}$$

$$x = (.50000019, 0, .50000004)^T$$

converged in 14 iterations. Exact answer =  
 $(\frac{1}{2}, 0, \frac{1}{2})^T$ .

The constraint for  $x_2$  is not active in this case.

3.

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad b = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$
$$x = (0, 0, 1)^T$$

converged in 11 iterations. The constraints for  $x_1$  and  $x_2$  are not active in this case.

## REFERENCES

- [1] Tapia, R.A., "A Stable Approach to Newton's Method for General Mathematical Programming Problems in  $R^n$ ," Journal of Optimization Theory and Applications, 14 (1974), pp. 453-476.
- [2] Tapia, R.A., "Diagonalized Multiplier Methods and Quasi-Newton Methods for Constrained Optimization," Journal of Optimization Theory and Applications.
- [3] Gill, P.E., Golub, G., Murray, W., and Saunders, M.A., "Methods for Modifying Matrix Factorizations," Math. Comp., 28 (1974), pp. 505-536.
- [4] Van Rooy, D.L., Lynn, M.S., and Snyder, C.H., "The Use of the Modified Cholesky Decomposition in Divergence and Classification Calculations," Conf. Proc. Machine Processing of Remotely Sensed Data, LARS, Purdue, Oct. 16-18, 1973, pp. 3B-35 - 3B-47.
- [5] Fletcher, R. and Powell, M.J.O., "On the Modification of  $LDL^T$  Factorizations," Math. Comp., 28 (1974), pp. 1067-1087.
- [6] Van Rooy, D.L., "On the Computation and Updating of the Modified Cholesky Decomposition of a Covariance Matrix," ICSA Technical Report #275-025-024, Rice University, Houston, Texas, May, 1976

## APPENDIX

### Program Listings of CONOPT

LEVEL 21.8 ( JUN 74 )

DS/360 FORTRAN H

DATE 71

```

COMPILER OPTIONS - NAME= MAIN.OPT=00,LINECNT=60,SIZE=0000K,
      SOURCE=EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT, ID,XREF
      SUBROUTINE CONOPT(N,DELFL,FUNCT,X,WKA,TMITER,TSTC,TCRC,TIDN,TDH,
      1 TIOUT,TDSNT,A,RSTRT,USELAM,XLAM)
      IMPLICIT REAL*8 (A=H,D=Z)

ISN 0002          THIS ROUTINE MINIMIZES F(X) S.T. X(I).GE.0 & SUM(X(I))=1.
                  FOR A DETAILED DESCRIPTION OF THE PROFRAM SEE ICSA TECHNICAL
                  REPORT NO. 275-025-029. RICE UNIV. HOUSTON, TEX. JUNE. 1976.

ISN 0003          C FUNCT IS A FUNCTION WHICH RETURNS F(X), ITS ARGUMENTS ARE X,N
                  DELF IS A SUBROUTINE WHICH COMPUTES THE GRADIENT OF F(X). ITS
                  ARGUMENTS ARE DF (THE GRAD), X, & N
                  DELF & FUNCT MUST BE DECLARED EXTERNAL IN THE MAIN PROGRAM.
                  WKA IS A WORK AREA OF LENGTH *GE* .5*N**2+(23*N+7)/2
                  X ON RETURN IS THE ARGUMENT AT THE MINIMUM
                  N IS THE DIMENSION OF X
                  A(1) & A(2) ARE USED IN THE CONSTRAINT REJECTION TEST. AS
                  A(1)*DNORM**A(2) DEFAULTS ARE A(1)=1. & A(2)=.2. IF THE USER
                  WISHES A(2) TO =0, HE SHOULD SET A(1) TO THE NEGATIVE OF WHAT HE
                  WANTS FOR A(1).
                  RSTRT - A LOGICAL VARIABLE INDICATING WHETHER THE USER IS
                  RESTARTING. IF SO. THE X*S MUST BE SUPPLIED & THE CONSTRAINT
                  REJECTION INDICATORS AS INTEGER#2 VARIABLES SHOULD BE IN THE FIRST
                  N LOCATIONS OF WKA
                  USELAM - A LOGICAL VARIABLE INDICATING WHETHER OR NOT THE USER
                  IS SUPPLYING A VALUE FOR XLAM IN THE CASE OF A RESTART.
                  XLAM - THE LAGRANGE MULTIPLIER FOR THE EQUALITY CONSTRAINT.

ISN 0004          REAL*8 X(N)*WKA(1)
ISN 0005          REAL*4 TSTC,TCRC,TDH
ISN 0006          REAL*4 STC,CRC,DH
ISN 0007          REAL*4 A(2)
ISN 0008          INTEGER*4 TMITER,TIDN,TIOUT
ISN 0009          LOGICAL#1 DSNT,TDSNT,RSTRT,USELAM
ISN 0010          COMMON /INPUT/MITER,STC,CRC,IDL,DH,IOUT,DSNT
ISN 0011          EXTERNAL DELF,FUNCT
ISN 0012          COMMON /ARANGE/IICR,IY,IH,IDL,IGL.
ISN 0013          1 IHGL,IHGY,IDL,IDS
ISN 0014          C COMPUTE THE STARTING ADDRESSES OF VARIOUS WORKING ARRYS.
ISN 0015          CALL DIMSN(N)
ISN 0016          C SET DEFAULT VALUES & OUTPUT (STEP 1)
ISN 0017          CALL INPT(TMITER,TSTC,TCRC,TION, TDH,TIOUT,TDSNT,N,A,RSTRT)
ISN 0018          C INITIALIZE VARIABLES
ISN 0019          C CALL INIT(X,WKA(IDF),WKA(IY),WKA(IH),WKA(IGL),WKA(IICR),XLAM,
ISN 0020          1 NITER,N,DELFL,RSTRT,USELAM)
ISN 0021          C STEP 2 - CHECK STOPPING CRITERION & OUTPUT IF REQUESTED
ISN 0022          C 100 CALL CHKSTP (X,N,G,DNORM,NITER,XLAM,WKA(IGL),
ISN 0023          E200, FUNCT).
ISN 0024          C
ISN 0025          C
ISN 0026          C
ISN 0027          C
ISN 0028          C
ISN 0029          C
ISN 0030          C
ISN 0031          C
ISN 0032          C
ISN 0033          C
ISN 0034          C
ISN 0035          C
ISN 0036          C
ISN 0037          C
ISN 0038          C
ISN 0039          C
ISN 0040          C
ISN 0041          C
ISN 0042          C
ISN 0043          C
ISN 0044          C
ISN 0045          C
ISN 0046          C
ISN 0047          C
ISN 0048          C
ISN 0049          C
ISN 0050          C
ISN 0051          C
ISN 0052          C
ISN 0053          C
ISN 0054          C
ISN 0055          C
ISN 0056          C
ISN 0057          C
ISN 0058          C
ISN 0059          C
ISN 0060          C
ISN 0061          C
ISN 0062          C
ISN 0063          C
ISN 0064          C
ISN 0065          C
ISN 0066          C
ISN 0067          C
ISN 0068          C
ISN 0069          C
ISN 0070          C
ISN 0071          C
ISN 0072          C
ISN 0073          C
ISN 0074          C
ISN 0075          C
ISN 0076          C
ISN 0077          C
ISN 0078          C
ISN 0079          C
ISN 0080          C
ISN 0081          C
ISN 0082          C
ISN 0083          C
ISN 0084          C
ISN 0085          C
ISN 0086          C
ISN 0087          C
ISN 0088          C
ISN 0089          C
ISN 0090          C
ISN 0091          C
ISN 0092          C
ISN 0093          C
ISN 0094          C
ISN 0095          C
ISN 0096          C
ISN 0097          C
ISN 0098          C
ISN 0099          C
ISN 0100          C
ISN 0101          C
ISN 0102          C
ISN 0103          C
ISN 0104          C
ISN 0105          C
ISN 0106          C
ISN 0107          C
ISN 0108          C
ISN 0109          C
ISN 0110          C
ISN 0111          C
ISN 0112          C
ISN 0113          C
ISN 0114          C
ISN 0115          C
ISN 0116          C
ISN 0117          C
ISN 0118          C
ISN 0119          C
ISN 0120          C
ISN 0121          C
ISN 0122          C
ISN 0123          C
ISN 0124          C
ISN 0125          C
ISN 0126          C
ISN 0127          C
ISN 0128          C
ISN 0129          C
ISN 0130          C
ISN 0131          C
ISN 0132          C
ISN 0133          C
ISN 0134          C
ISN 0135          C
ISN 0136          C
ISN 0137          C
ISN 0138          C
ISN 0139          C
ISN 0140          C
ISN 0141          C
ISN 0142          C
ISN 0143          C
ISN 0144          C
ISN 0145          C
ISN 0146          C
ISN 0147          C
ISN 0148          C
ISN 0149          C
ISN 0150          C
ISN 0151          C
ISN 0152          C
ISN 0153          C
ISN 0154          C
ISN 0155          C
ISN 0156          C
ISN 0157          C
ISN 0158          C
ISN 0159          C
ISN 0160          C
ISN 0161          C
ISN 0162          C
ISN 0163          C
ISN 0164          C
ISN 0165          C
ISN 0166          C
ISN 0167          C
ISN 0168          C
ISN 0169          C
ISN 0170          C
ISN 0171          C
ISN 0172          C
ISN 0173          C
ISN 0174          C
ISN 0175          C
ISN 0176          C
ISN 0177          C
ISN 0178          C
ISN 0179          C
ISN 0180          C
ISN 0181          C
ISN 0182          C
ISN 0183          C
ISN 0184          C
ISN 0185          C
ISN 0186          C
ISN 0187          C
ISN 0188          C
ISN 0189          C
ISN 0190          C
ISN 0191          C
ISN 0192          C
ISN 0193          C
ISN 0194          C
ISN 0195          C
ISN 0196          C
ISN 0197          C
ISN 0198          C
ISN 0199          C
ISN 0200          C
ISN 0201          C
ISN 0202          C
ISN 0203          C
ISN 0204          C
ISN 0205          C
ISN 0206          C
ISN 0207          C
ISN 0208          C
ISN 0209          C
ISN 0210          C
ISN 0211          C
ISN 0212          C
ISN 0213          C
ISN 0214          C
ISN 0215          C
ISN 0216          C
ISN 0217          C
ISN 0218          C
ISN 0219          C
ISN 0220          C
ISN 0221          C
ISN 0222          C
ISN 0223          C
ISN 0224          C
ISN 0225          C
ISN 0226          C
ISN 0227          C
ISN 0228          C
ISN 0229          C
ISN 0230          C
ISN 0231          C
ISN 0232          C
ISN 0233          C
ISN 0234          C
ISN 0235          C
ISN 0236          C
ISN 0237          C
ISN 0238          C
ISN 0239          C
ISN 0240          C
ISN 0241          C
ISN 0242          C
ISN 0243          C
ISN 0244          C
ISN 0245          C
ISN 0246          C
ISN 0247          C
ISN 0248          C
ISN 0249          C
ISN 0250          C
ISN 0251          C
ISN 0252          C
ISN 0253          C
ISN 0254          C
ISN 0255          C
ISN 0256          C
ISN 0257          C
ISN 0258          C
ISN 0259          C
ISN 0260          C
ISN 0261          C
ISN 0262          C
ISN 0263          C
ISN 0264          C
ISN 0265          C
ISN 0266          C
ISN 0267          C
ISN 0268          C
ISN 0269          C
ISN 0270          C
ISN 0271          C
ISN 0272          C
ISN 0273          C
ISN 0274          C
ISN 0275          C
ISN 0276          C
ISN 0277          C
ISN 0278          C
ISN 0279          C
ISN 0280          C
ISN 0281          C
ISN 0282          C
ISN 0283          C
ISN 0284          C
ISN 0285          C
ISN 0286          C
ISN 0287          C
ISN 0288          C
ISN 0289          C
ISN 0290          C
ISN 0291          C
ISN 0292          C
ISN 0293          C
ISN 0294          C
ISN 0295          C
ISN 0296          C
ISN 0297          C
ISN 0298          C
ISN 0299          C
ISN 0300          C
ISN 0301          C
ISN 0302          C
ISN 0303          C
ISN 0304          C
ISN 0305          C
ISN 0306          C
ISN 0307          C
ISN 0308          C
ISN 0309          C
ISN 0310          C
ISN 0311          C
ISN 0312          C
ISN 0313          C
ISN 0314          C
ISN 0315          C
ISN 0316          C
ISN 0317          C
ISN 0318          C
ISN 0319          C
ISN 0320          C
ISN 0321          C
ISN 0322          C
ISN 0323          C
ISN 0324          C
ISN 0325          C
ISN 0326          C
ISN 0327          C
ISN 0328          C
ISN 0329          C
ISN 0330          C
ISN 0331          C
ISN 0332          C
ISN 0333          C
ISN 0334          C
ISN 0335          C
ISN 0336          C
ISN 0337          C
ISN 0338          C
ISN 0339          C
ISN 0340          C
ISN 0341          C
ISN 0342          C
ISN 0343          C
ISN 0344          C
ISN 0345          C
ISN 0346          C
ISN 0347          C
ISN 0348          C
ISN 0349          C
ISN 0350          C
ISN 0351          C
ISN 0352          C
ISN 0353          C
ISN 0354          C
ISN 0355          C
ISN 0356          C
ISN 0357          C
ISN 0358          C
ISN 0359          C
ISN 0360          C
ISN 0361          C
ISN 0362          C
ISN 0363          C
ISN 0364          C
ISN 0365          C
ISN 0366          C
ISN 0367          C
ISN 0368          C
ISN 0369          C
ISN 0370          C
ISN 0371          C
ISN 0372          C
ISN 0373          C
ISN 0374          C
ISN 0375          C
ISN 0376          C
ISN 0377          C
ISN 0378          C
ISN 0379          C
ISN 0380          C
ISN 0381          C
ISN 0382          C
ISN 0383          C
ISN 0384          C
ISN 0385          C
ISN 0386          C
ISN 0387          C
ISN 0388          C
ISN 0389          C
ISN 0390          C
ISN 0391          C
ISN 0392          C
ISN 0393          C
ISN 0394          C
ISN 0395          C
ISN 0396          C
ISN 0397          C
ISN 0398          C
ISN 0399          C
ISN 0400          C
ISN 0401          C
ISN 0402          C
ISN 0403          C
ISN 0404          C
ISN 0405          C
ISN 0406          C
ISN 0407          C
ISN 0408          C
ISN 0409          C
ISN 0410          C
ISN 0411          C
ISN 0412          C
ISN 0413          C
ISN 0414          C
ISN 0415          C
ISN 0416          C
ISN 0417          C
ISN 0418          C
ISN 0419          C
ISN 0420          C
ISN 0421          C
ISN 0422          C
ISN 0423          C
ISN 0424          C
ISN 0425          C
ISN 0426          C
ISN 0427          C
ISN 0428          C
ISN 0429          C
ISN 0430          C
ISN 0431          C
ISN 0432          C
ISN 0433          C
ISN 0434          C
ISN 0435          C
ISN 0436          C
ISN 0437          C
ISN 0438          C
ISN 0439          C
ISN 0440          C
ISN 0441          C
ISN 0442          C
ISN 0443          C
ISN 0444          C
ISN 0445          C
ISN 0446          C
ISN 0447          C
ISN 0448          C
ISN 0449          C
ISN 0450          C
ISN 0451          C
ISN 0452          C
ISN 0453          C
ISN 0454          C
ISN 0455          C
ISN 0456          C
ISN 0457          C
ISN 0458          C
ISN 0459          C
ISN 0460          C
ISN 0461          C
ISN 0462          C
ISN 0463          C
ISN 0464          C
ISN 0465          C
ISN 0466          C
ISN 0467          C
ISN 0468          C
ISN 0469          C
ISN 0470          C
ISN 0471          C
ISN 0472          C
ISN 0473          C
ISN 0474          C
ISN 0475          C
ISN 0476          C
ISN 0477          C
ISN 0478          C
ISN 0479          C
ISN 0480          C
ISN 0481          C
ISN 0482          C
ISN 0483          C
ISN 0484          C
ISN 0485          C
ISN 0486          C
ISN 0487          C
ISN 0488          C
ISN 0489          C
ISN 0490          C
ISN 0491          C
ISN 0492          C
ISN 0493          C
ISN 0494          C
ISN 0495          C
ISN 0496          C
ISN 0497          C
ISN 0498          C
ISN 0499          C
ISN 0500          C
ISN 0501          C
ISN 0502          C
ISN 0503          C
ISN 0504          C
ISN 0505          C
ISN 0506          C
ISN 0507          C
ISN 0508          C
ISN 0509          C
ISN 0510          C
ISN 0511          C
ISN 0512          C
ISN 0513          C
ISN 0514          C
ISN 0515          C
ISN 0516          C
ISN 0517          C
ISN 0518          C
ISN 0519          C
ISN 0520          C
ISN 0521          C
ISN 0522          C
ISN 0523          C
ISN 0524          C
ISN 0525          C
ISN 0526          C
ISN 0527          C
ISN 0528          C
ISN 0529          C
ISN 0530          C
ISN 0531          C
ISN 0532          C
ISN 0533          C
ISN 0534          C
ISN 0535          C
ISN 0536          C
ISN 0537          C
ISN 0538          C
ISN 0539          C
ISN 0540          C
ISN 0541          C
ISN 0542          C
ISN 0543          C
ISN 0544          C
ISN 0545          C
ISN 0546          C
ISN 0547          C
ISN 0548          C
ISN 0549          C
ISN 0550          C
ISN 0551          C
ISN 0552          C
ISN 0553          C
ISN 0554          C
ISN 0555          C
ISN 0556          C
ISN 0557          C
ISN 0558          C
ISN 0559          C
ISN 0560          C
ISN 0561          C
ISN 0562          C
ISN 0563          C
ISN 0564          C
ISN 0565          C
ISN 0566          C
ISN 0567          C
ISN 0568          C
ISN 0569          C
ISN 0570          C
ISN 0571          C
ISN 0572          C
ISN 0573          C
ISN 0574          C
ISN 0575          C
ISN 0576          C
ISN 0577          C
ISN 0578          C
ISN 0579          C
ISN 0580          C
ISN 0581          C
ISN 0582          C
ISN 0583          C
ISN 0584          C
ISN 0585          C
ISN 0586          C
ISN 0587          C
ISN 0588          C
ISN 0589          C
ISN 0590          C
ISN 0591          C
ISN 0592          C
ISN 0593          C
ISN 0594          C
ISN 0595          C
ISN 0596          C
ISN 0597          C
ISN 0598          C
ISN 0599          C
ISN 0600          C
ISN 0601          C
ISN 0602          C
ISN 0603          C
ISN 0604          C
ISN 0605          C
ISN 0606          C
ISN 0607          C
ISN 0608          C
ISN 0609          C
ISN 0610          C
ISN 0611          C
ISN 0612          C
ISN 0613          C
ISN 0614          C
ISN 0615          C
ISN 0616          C
ISN 0617          C
ISN 0618          C
ISN 0619          C
ISN 0620          C
ISN 0621          C
ISN 0622          C
ISN 0623          C
ISN 0624          C
ISN 0625          C
ISN 0626          C
ISN 0627          C
ISN 0628          C
ISN 0629          C
ISN 0630          C
ISN 0631          C
ISN 0632          C
ISN 0633          C
ISN 0634          C
ISN 0635          C
ISN 0636          C
ISN 0637          C
ISN 0638          C
ISN 0639          C
ISN 0640          C
ISN 0641          C
ISN 0642          C
ISN 0643          C
ISN 0644          C
ISN 0645          C
ISN 0646          C
ISN 0647          C
ISN 0648          C
ISN 0649          C
ISN 0650          C
ISN 0651          C
ISN 0652          C
ISN 0653          C
ISN 0654          C
ISN 0655          C
ISN 0656          C
ISN 0657          C
ISN 0658          C
ISN 0659          C
ISN 0660          C
ISN 0661          C
ISN 0662          C
ISN 0663          C
ISN 0664          C
ISN 0665          C
ISN 0666          C
ISN 0667          C
ISN 0668          C
ISN 0669          C
ISN 0670          C
ISN 0671          C
ISN 0672          C
ISN 0673          C
ISN 0674          C
ISN 0675          C
ISN 0676          C
ISN 0677          C
ISN 0678          C
ISN 0679          C
ISN 0680          C
ISN 0681          C
ISN 0682          C
ISN 0683          C
ISN 0684          C
ISN 0685          C
ISN 0686          C
ISN 0687          C
ISN 0688          C
ISN 0689          C
ISN 0690          C
ISN 0691          C
ISN 0692          C
ISN 0693          C
ISN 0694          C
ISN 0695          C
ISN 0696          C
ISN 0697          C
ISN 0698          C
ISN 0699          C
ISN 0700          C
ISN 0701          C
ISN 0702          C
ISN 0703          C
ISN 0704          C
ISN 0705          C
ISN 0706          C
ISN 0707          C
ISN 0708          C
ISN 0709          C
ISN 0710          C
ISN 0711          C
ISN 0712          C
ISN 0713          C
ISN 0714          C
ISN 0715          C
ISN 0716          C
ISN 0717          C
ISN 0718          C
ISN 0719          C
ISN 0720          C
ISN 0721          C
ISN 0722          C
ISN 0723          C
ISN 0724          C
ISN 0725          C
ISN 0726          C
ISN 0727          C
ISN 0728          C
ISN 0729          C
ISN 0730          C
ISN 0731          C
ISN 0732          C
ISN 0733          C
ISN 0734          C
ISN 0735          C
ISN 0736          C
ISN 0737          C
ISN 0738          C
ISN 0739          C
ISN 0740          C
ISN 0741          C
ISN 0742          C
ISN 0743          C
ISN 0744          C
ISN 0745          C
ISN 0746          C
ISN 0747          C
ISN 0748          C
ISN 0749          C
ISN 0750          C
ISN 0751          C
ISN 0752          C
ISN 0753          C
ISN 0754          C
ISN 0755          C
ISN 0756          C
ISN 0757          C
ISN 0758          C
ISN 0759          C
ISN 0760          C
ISN 0761          C
ISN 0762          C
ISN 0763          C
ISN 0764          C
ISN 0765          C
ISN 0766          C
ISN 0767          C
ISN 0768          C
ISN 0769          C
ISN 0770          C
ISN 0771          C
ISN 0772          C
ISN 0773          C
ISN 0774          C
ISN 0775          C
ISN 0776          C
ISN 0777          C
ISN 0778          C
ISN 0779          C
ISN 0780          C
ISN 0781          C
ISN 0782          C
ISN 0783          C
ISN 0784          C
ISN 0785          C
ISN 0786          C
ISN 0787          C
ISN 0788          C
ISN 0789          C
ISN 0790          C
ISN 0791          C
ISN 0792          C
ISN 0793          C
ISN 0794          C
ISN 0795          C
ISN 0796          C
ISN 0797          C
ISN 0798          C
ISN 0799          C
ISN 0800          C
ISN 0801          C
ISN 0802          C
ISN 0803          C
ISN 0804          C
ISN 0805          C
ISN 0806          C
ISN 0807          C
ISN 0808          C
ISN 0809          C
ISN 0810          C
ISN 0811          C
ISN 0812          C
ISN 0813          C
ISN 0814          C
ISN 0815          C
ISN 0816          C
ISN 0817          C
ISN 0818          C
ISN 0819          C
ISN 0820          C
ISN 0821          C
ISN 0822          C
ISN 0823          C
ISN 0824          C
ISN 0825          C
ISN 0826          C
ISN 0827          C
ISN 0828          C
ISN 0829          C
ISN 0830          C
ISN 0831          C
ISN 0832          C
ISN 0833          C
ISN 0834          C
ISN 0835          C
ISN 0836          C
ISN 0837          C
ISN 0838          C
ISN 0839          C
ISN 0840          C
ISN 0841          C
ISN 0842          C
ISN 0843          C
ISN 0844          C
ISN 0845          C
ISN 0846          C
ISN 0847          C
ISN 0848          C
ISN 0849          C
ISN 0850          C
ISN 0851          C
ISN 0852          C
ISN 0853          C
ISN 0854          C
ISN 0855          C
ISN 0856          C
ISN 0857          C
ISN 0858          C
ISN 0859          C
ISN 0860          C
ISN 0861          C
ISN 0862          C
ISN 0863          C
ISN 0864          C
ISN 0865          C
ISN 0866          C
ISN 0867          C
ISN 0868          C
ISN 0869          C
ISN 0870          C
ISN 0871          C
ISN 0872          C
ISN 0873          C
ISN 0874          C
ISN 0875          C
ISN 0876          C
ISN 0877          C
ISN 0878          C
ISN 0879          C
ISN 0880          C
ISN 0881          C
ISN 0882          C
ISN 0883          C
ISN 0884          C
ISN 0885          C
ISN 0886          C
ISN 0887          C
ISN 0888          C
ISN 0889          C
ISN 0890          C
ISN 0891          C
ISN 0892          C
ISN 0893          C
ISN 0894          C
ISN 0895          C
ISN 0896          C
ISN 0897          C
ISN 0898          C
ISN 0899          C
ISN 0900          C
ISN 0901          C
ISN 0902          C
ISN 0903          C
ISN 0904          C
ISN 0905          C
ISN 0906          C
ISN 0907          C
ISN 0908          C
ISN 0909          C
ISN 0910          C
ISN 0911          C
ISN 0912          C
ISN 0913          C
ISN 0914          C
ISN 0915          C
ISN 0916          C
ISN 0917          C
ISN 0918          C
ISN 0919          C
ISN 0920          C
ISN 0921          C
ISN 0922          C
ISN 0923          C
ISN 0924          C
ISN 0925          C
ISN 0926          C
ISN 0927          C
ISN 0928          C
ISN 0929          C
ISN 0930          C
ISN 0931          C
ISN 0932          C
ISN 0933          C
ISN 0934          C
ISN 0935          C
ISN 0936          C
ISN 0937          C
ISN 0938          C
ISN 0939          C
ISN 0940          C
ISN 0941          C
ISN 0942          C
ISN 0943          C
ISN 0944          C
ISN 0945          C
ISN 0946          C
ISN 0947          C
ISN 0948          C
ISN 0949          C
ISN 0950          C
ISN 0951          C
ISN 0952          C
ISN 0953          C
ISN 0954          C
ISN 0955          C
ISN 0956          C
ISN 0957          C
ISN 0958          C
ISN 0959          C
ISN 0960          C
ISN 0961          C
ISN 0962          C
ISN 0963          C
ISN 0964          C
ISN 0965          C
ISN 0966          C
ISN 0967          C
ISN 0968          C
ISN 0969          C
ISN 0970          C
ISN 0971          C
ISN 0972          C
ISN 0973          C
ISN 0974          C
ISN 0975          C
ISN 0976          C
ISN 0977          C
ISN 0978          C
ISN 0979          C
ISN 0980          C
ISN 0981          C
ISN 0982          C
ISN 0983          C
ISN 0984          C
ISN 0985          C
ISN 0986          C
ISN 0987          C
ISN 0988          C
ISN 0989          C
ISN 0990          C
ISN 0991          C
ISN 0992          C
ISN 0993          C
ISN 0994          C
ISN 0995          C
ISN 0996          C
ISN 0997          C
ISN 0998          C
ISN 0999          C
ISN 1000          C
ISN 1001          C
ISN 1002          C
ISN 1003          C
ISN 1004          C
ISN 1005          C
ISN 1006          C
ISN 1007          C
ISN 1008          C
ISN 1009          C
ISN 1010          C
ISN 1011          C
ISN 1012          C
ISN 1013          C
ISN 1014          C
ISN 1015          C
ISN 1016          C
ISN 1017          C
ISN 1018          C
ISN 1019          C
ISN 1020          C
ISN 1021          C
ISN 1022          C
ISN 1023          C
ISN 1024          C
ISN 1025          C
ISN 1026          C
ISN 1027          C
ISN 1028          C
ISN 1029          C
ISN 1030          C
ISN 1031          C
ISN 1032          C
ISN 1033          C
ISN 1034          C
ISN 1035          C
ISN 1036          C
ISN 1037          C
ISN 1038          C
ISN 1039          C
ISN 1040          C
ISN 1041          C
ISN 1042          C
ISN 1043          C
ISN 1044          C
ISN 1045          C
ISN 1046          C
ISN 10
```

```

ISN 0017      C STEP 3 - CALCULATE DISCRETE HESSIAN (OPTIONAL)
C           C CALL DHESS ( NITER, WKA(IH), WKA(IWK), WKA(IGL),
C           1 WKA(IGL2), DELF, XLAM, WKA(IICR), WKA(IY), N, X )
C
ISN 0018      C STEPS 4 & 5 - CALCULATE MULTIPLIER & Y-VARIABLE CORRECTIONS
C           C CALL DELAMY (WKA(IGY), WKA(IH), WKA(IGL), N, WKA(IHGL), WKA(IHGY),
C           1 DLAM, WKA(IDLY), G, WKA(IY), WKA(IICR) )
C
ISN 0019      C STEP 6 - CALCULATE NEW Y BASED ON (OPTIONAL ) DESCENT
C           C CALL NEWY (DNORM, XLAM, WKA(IY), N, DLAM, WKA(IDLY), G, NITER, WKA(IICR),
C           1 WKA(IYSAV), X, WKA(IDF), DELF, WKA(IDY), WKA(IGL) )
C
ISN 0020      C STEP 7 - UPDATE APPROXIMATE INVERSE HESSIAN
C           C CALL UPDTINITER, WKA(IYSAV), N, WKA(IGL), XLAM, WKA(IICR),
C           1 WKA(IY), WKA(IH), WKA(IGL2), WKA(IDY), WKA(IDS), WKA(IWK) )
C
ISN 0021      C STEP 8 - UPDATE CONSTRAINT REJECTION INDICATORS
C           C CALL CONSTR(WKA(IDF), X, N, WKA(IICR), XLAM, DELF, WKA(IGL), WKA(IY),
C           1 WKA(IYSAV), DNORM, A)
C
ISN 0022      C           1 GO TO 100
ISN 0023      C           200 RETURN
ISN 0024      C           END

```

LEVEL 21.8 ( JUN 74 )

OS/360 FORTRAN H

DATE 7

COMPILER OPTIONS = NAME= MAIN.OPT=00,LINECNT=60,SIZE=0000K,  
ISN 0002 SOURCE=EBCDIC,NOLIST,NOECK,LOAD,MAP,NOEDIT, ID,XREF  
SUBROUTINE DIMS(N)

C COMPUTE STARTING POINTS OF WORKING ARRAYS USED. THE ORDER IS ICR,  
C Y,H, DF,GL2,DY, & WK

C

ISN 0003 INTEGER\*4 IDM(8)  
ISN 0004 COMMON /ARANGE/ IDM,IGY,IYSAV,IHGL,IHGY,IDLX,IDS  
ISN 0005 IDM(1)=1  
ISN 0006 IDM(2)=(N+1)/2  
ISN 0007 IDM(3)=N  
ISN 0008 IDM(4)=N\*(N+1)/2  
ISN 0009 IDM(5)=N  
ISN 0010 IDM(6)=N  
ISN 0011 IDM(7)=N  
ISN 0012 IDM(8)=N  
ISN 0013 DO 10 I=2,8  
ISN 0014 IDM(I)=IDM(I)+IDM(I-1)  
ISN 0015 IGY=IDM(8)  
ISN 0016 IYSAV=IGY+N  
ISN 0017 IHGL=IYSAV+N  
ISN 0018 IHGY=IHGL+N  
ISN 0019 IDLY=IHGY+N  
ISN 0020 IDS=IDLY+N  
ISN 0021 RETURN  
ISN 0022 END

(3)

LEVEL 21.8 ( JUN 74 )

05/360 FORTRAN H

DATE 7

COMPILER OPTIONS - NAME= MAIN,OPT=00,LINECNT=60,SIZE=0000K,  
SOURCE=EBCDIC,NODECK,LOAD,MAP,NOEDIT,IDL,XREF  
SUBROUTINE INPT (TMITER,TSTC,TCRC,TIDN,TDH,TOUT,N,A,RSTRT)  
ISN 0002 C SET DEFAULT VALUES FOR PARAMETERS USED.  
C  
ISN 0003 LOGICAL#1 DSNT,DSNT,RSTRT  
ISN 0004 INTEGER#4 TMITER,TION,TIOUT  
ISN 0005 REAL#4 A(2)  
ISN 0006 REAL#4 TSTC,TCRC,DH  
ISN 0007 REAL#4 STC,CRC,DH  
ISN 0008 COMMON /INPUT/ MITER,STC,CRC,IDN,DH,IOUT,DSNT  
ISN 0009 NAMELIST /IN/ MITER,STC,CRC,IDN,DH,IOUT,DSNT  
ISN 0010 EPS#1.E-20  
ISN 0011 MITER=MAX0(.30,.2\*N)  
ISN 0012 IF (TMITER.GT.#0) MITER=TMITER  
ISN 0013 STC=1.E-8  
ISN 0014 IF (TSTC.GT.EPS) STC=TSTC  
ISN 0015 CRC=.01  
ISN 0016 IF (TCRC.GT.EPS) CRC=TCRC  
ISN 0017 IDN=HINO(10,N)  
ISN 0018 IF (TIDN.GE.0) IDN=Y1R<sup>1</sup>  
ISN 0019 DH=.01  
ISN 0020 IF (TDH.GT.EPS) DH=TDH  
ISN 0021 IOUT=1  
ISN 0022 IF (TIOUT.GE.0) IOUT=TIOUT  
ISN 0023 DSNT=DSNT  
ISN 0024 IF ((A(2).LE.EPS) A(2)=.2  
ISN 0025 IF ((A(1).LT.0.) A(2)=0.  
ISN 0026 A(1)=ABS(A(1))  
ISN 0027 IF ((A(1).LE.EPS) A(1)=1.  
ISN 0028 WRITE ('6.1')  
ISN 0029 11 FORMAT ('1 PARAMETERS USED')  
ISN 0030 ISN 0031 WRITE ('6.1')  
ISN 0032 ISN 0033 WRITE ('6.1')  
ISN 0033 ISN 0034 WRITE ('6.1')  
ISN 0034 ISN 0035 WRITE ('6.1')  
ISN 0035 ISN 0036 WRITE ('6.1')  
ISN 0036 ISN 0037 WRITE ('6.1')  
ISN 0037 ISN 0038 WRITE ('6.1')  
ISN 0038 ISN 0039 WRITE ('6.1')  
ISN 0039 ISN 0040 WRITE ('6.1')  
ISN 0040 ISN 0041 12 FORMAT (' A(1)=' ,G16.8, ' A(2)=' ,G16.8)  
ISN 0042 ISN 0043 WRITE ('6.1') RSTRT  
ISN 0043 ISN 0044 RETURN  
ISN 0044 ISN 0045  
ISN 0045

(4)

LEVEL 21.8 ( JUN 74 )

OS/360 FORTRAN H

DATE 7

COMPILER OPTIONS - NAME= MAIN,OPT=00,LINECNT=60,SIZE=0000K,  
ISN 0002 SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,IO,XREF  
ISN 0003 SUBROUTINE INIT(X,DF,Y,H,GL,ICR,XLAM,NITER,N,DELFL,RSRT,USELAM)  
IMPLICIT REAL\*8 (A=H,O=Z)

THIS ROUTINE INITIALIZES SEVERAL VARIABLES USED IN THE OPTIMIZATION

```
C LOGICAL*1 USELAM
C LOGICAL*1 RSRT
C INTEGER*2 ICR(N)
C REAL*8 X(N),DF(N),Y(N),GL(N),H(1)
C IF (RSRT) GO TO 5
C SN=SORT(2.*N)
C DO 10 I=1,N
C ICR(I)=2
C Y(I)=SN
C 10 X(I)=1./N
C GO TO 15
C 5 DO 40 I=1,N
C Y(I)=X(I)
C IF ((ICR(I).EQ.1) .AND. (ICR(I).GE.2))
C Y(I)=DSORT(2.*X(I))
C 40 CONTINUE
C 15 CONTINUE
C DF IS THE GRADIENT OF F
C CALL DELF(DF,X,N)
C SET H TO THE IDENTITY MATRIX
C N2=N*(N+1)/2
C DO 20 K=1,N2
C H(K)=0.
C 20 I=0
C DO 30 K=1,N
C I=I+K
C H(I)=1.
C 30 IF (*.NOT.USELAM) XLAM=DOT(DF,X,N)
C NITER=0
C GL IS THE GRADIENT OF THE LAGRANGIAN
C CALL GRADL(GL,DF,XLAM,ICR,Y,N)
C RETURN
C END
```

(5)

LEVEL 21.0 ( JUN 74 )

OS/360 FORTRAN H

DATE 7

```
COMPLIER OPTIONS - NAME= MAIN.OPT=00•LINECNT=60•SIZE=0000K*
 SOURCE=EBCDIC•NOLIST•NODECK•LOAD•HAP•NOED•IT•ID•XREF
 SUBROUTINE CHKSTP(X,N,G,DNORM,NITER,XLAM,GL,FUNCT)
1*)
      IMPLICIT REAL*8 (A=H,0=Z)
      REAL*8 SUM,SUM2
      REAL*8 X(N),GL(N)
      REAL*4 STC,CRC,DH
      COMMON /INPUT/ NITER,STC,CRC,IDL,DH,IOUT
      C EVALUATE THE NORM OF GRAD(L) + THE NORM OF THE EQUALITY CONSTRAINT, G.
      C
      G=1•DO=SUM(X,N)
      DNORM=G*G+SUM2(GL,N)
      IF (DNORM.GE.STC.AND.NITER.LT.NITER) GO TO 15
      IF (IOUT.EQ.0) RETURN 1
      F=FUNCT(X,N)
      WRITE(6,11) NITER, DNORM,G,XLAM,F,(X(I),I=1,N)
11 FORMAT(//,NITER, DNORM,G,XLAM,F,X,I10,4G16.8,(/8G16.8))
      RETURN 1
      C
      C INTERMEDIATE OUTPUT IF DESIRED
      C
      15 IF (IOUT.EQ.0) GO TO 25
      IF (NITER.IOUT*IOUT*NE.NITER) GO TO 25
      F=FUNCT(X,N)
      WRITE(6,11) NITER, DNORM,G,XLAM,F,(X(I),I=1,N)
25 NITER=NITER+1
      RETURN
      END
      ISN 0018
      ISN 0020
      ISN 0022
      ISN 0023
      ISN 0024
      ISN 0025
      ISN 0026
```

LEVEL 21.8 ( JUN 74 )

OS/360 FORTRAN H

DATE ?

COMPILER OPTIONS - NAME= MAIN,OPT=00,LINECNT=60,SIZE=0000K.  
SOURCE,EBCDIC,NOLIST,NOECK,LOAD,MAP,NOEDIT,IO,XREF  
SUBROUTINE DHESS(NITER,H,  
IMPLICIT REAL\*8 ( A=H,C=Z )  
ISN 0002 C  
ISN 0003 C  
ISN 0004 C  
ISN 0005 C  
ISN 0006 C  
ISN 0007 C  
ISN 0008 C  
ISN 0010 C  
ISN 0012 C  
ISN 0013 C  
ISN 0014 C  
ISN 0015 C  
ISN 0016 C  
ISN 0017 C  
ISN 0018 C  
ISN 0019 C  
ISN 0020 C  
ISN 0021 C  
ISN 0022 C  
ISN 0023 C  
ISN 0024 C  
ISN 0025 C  
ISN 0026 C  
ISN 0027 C  
ISN 0028 C  
C CALCULATE DISCRETE APPROX. TO THE HESSIAN IF DESIRED  
REAL\*8 WK(1),H(1), GL(1),GL2(1), Y(1),X(1)  
INTEGER\*2 ICR(1)  
REAL\*4 STC,CRC,DH  
CGMNDN /INPUT/ MITER,STC,CRC,JDN,DH  
IF (JDN.EQ.0) RETURN  
IF ((NITER-1)/IDN\*JDN.NITER-1) RETURN  
C COMPUTE THE HESSIAN  
K=0  
DO 10 J=1,N  
YT=Y(J)  
Y(J)=YC(J)+DH  
XT=X(J)  
X(J)=YT  
CALL DELF(WK,X,N)  
X(J)=XT  
CALL GRADL(GL2,WK,XLM,ICR,Y,N)  
Y(J)=YT  
DO 10 I=1,J  
K=K+1  
H(K)=(GL2(I)-GL(I))/DH  
10 CONTINUE  
C COMPUTE THE MODIFIED CHOLESKY DECOMP. OF H & STORE IN H  
CALL MCHLSK(H,N,WK)  
RETURN  
END

(7)

LEVEL 21.8 ( JUN 74 )

OS/360 FORTRAN H

DATE 7

```
COMPILER OPTIONS = NAME= MAIN,OPT=00,LINECNT=60,SIZE=0000K.
                  SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,IDL,XREF
ISN 0002          SUBROUTINE DELAMY(GY,H,GL,N,HGL,HGY,DLAM,DLY,G,Y,ICR)
ISN 0003          IMPLICIT REAL*8 (A-H,O-Z)

C           COMPUTE DELTA-LAMBDA & DELTA-Y

ISN 0004          REAL*8 GY(N),H(1), GL(N),HGL(N),HGY(N),DLY(N),Y(N)
ISN 0005          INTEGER*2 ICR(N)
ISN 0006          DO 10 I=1,N
ISN 0007          GY(I)=Y(I)
ISN 0008          10 IF (ICR(I)*NE*2) GY(I)=-1.
ISN 0010          CALL PDSOLV(H,HGL,GL,N)
ISN 0011          CALL PDSOLV(H,HGY,GY,N)
ISN 0012          DLAY=G-DOT(GY,HGL,N)/DOT(GY,HGY,N)
ISN 0013          DO 20 I=1,N
ISN 0014          20 DLY(I)=-HGL(I)-DLAM*HGY(I)
ISN 0015          RETURN
ISN 0016          END
```

(8)

LEVEL 21.8 ( JUN 74 )

05/360 FORTRAN H

DATE 7

COMPILER OPTIONS - NAME= MAIN,OPT=00,LINECNT=50,SIZE=0000X,  
SOURCE=EBCDIC,NOLIST,NOECK,LOAD,MAP,NOEDIT, ID,XREF  
SUBROUTINE NEWY (DNORM,XLAM,Y,N,DLAM,DEL,Y,ITER,ICR,YSAV,X,  
DF,DELF,DFSAV,GL)  
IMPLICIT REAL\*8 (A=H,B=Z)

C COMPUTE A NEW VALUE OF Y BASED (OPTIONAL Y) ON THE DESCENT OF THE  
NORM OF GRAD(L) + NORM OF G.

ISN 0004  
ISN 0005  
ISN 0006  
ISN 0007  
ISN 0008  
ISN 0009  
ISN 0010  
ISN 0011  
ISN 0012  
ISN 0013  
ISN 0014  
ISN 0015  
ISN 0016  
ISN 0017  
ISN 0018  
ISN 0019  
ISN 0020  
ISN 0021  
ISN 0022  
ISN 0023  
ISN 0024  
ISN 0025  
ISN 0026  
ISN 0027  
ISN 0028  
ISN 0029  
ISN 0030  
ISN 0032  
ISN 0033  
ISN 0035  
ISN 0036

REAL\*8 SUM2  
REAL\*8 SUM  
LOGICAL\*1 DSNT,DSNT2  
REAL\*8 GL(N),DFSAV(N)  
REAL\*8 Y(N),DELY(N),YSAV(N),X(N),DF(N)  
REAL\*4 STC,CRC,DH  
INTEGER\*2 ICR(N)  
COMMON /INPUT/ MITER,STC,CRC,IDN,DH,IOUT,DSNT  
DSNT2=DSNT  
ALP=1.  
DNSAVE=DNORM  
XLASAV=XLAM  
DO 10 I=1,N  
DFSAV(I)=DF(I)  
10 YSAV(I)=Y(I)  
15 DO 20 L=1,10  
XLAM=XLASAV+ALP\*DLAM  
DO 30 J=1,N  
Y(J)=YSAV(J)+ALP\*DELY(J)  
IF (ICR(J).EQ.1.AND.Y(J).LT.0.) Y(J)=0.  
X(J)=Y(J)\*ICR(J)/ICR(J)  
30 CONTINUE  
CALL DELF(DF,X,N)  
C DESCEND IF DESIRED  
C  
G=1.D0-SUM(X,N)  
CALL GRADL(GL,DF,XLAM,ICR,Y,N)  
IF (.NOT.DSNT2) RETURN  
DNORM=G\*G+SUM2(CL,N)  
IF (DNORM.LT.DNSAVE) RETURN  
20 ALP=ALP/2.  
IF (ALP.LT.0.) GO TO 25  
C TRY OTHER DIRECTION  
C  
ALP=-1.  
GO TO 15  
25 WRITE (6,11) MITER,ISN  
11 FORMAT ("\*ON ITER",ISN," DESCENT COULD NOT BE IMPLEMENTED. STEP LE  
NGTH SET TO 1.")  
ALP=1.  
DSNT2=.FALSE.  
GO TO 15  
RETURN  
END

(9)

LEVEL 21.8 ( JUN 74 )

OS/360 FORTRAN H

DATE

```

COMPILER OPTIONS = NAME= MAIN,OPT=00,LINECNT=60,SIZE=0000K,
      SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT, ID,XREF,
      SUBROUTINE UPDTH(NITER,YSAV,N, GL,XLAM,ICR,Y,H, GL2,DY,DS,T)
      IMPLICIT REAL*8 (A-H,O-Z)

      UPDATE APPROX. INVERSE HESSIAN

      ISN 0002      C
      ISN 0003      C
      ISN 0004      C
      ISN 0005      C
      ISN 0006      C
      ISN 0007      C
      ISN 0008      C
      ISN 0009      C
      ISN 0010      C
      ISN 0012      C

      ISN 0014      C
      ISN 0015      C
      ISN 0016      C
      ISN 0017      C
      ISN 0018      C
      ISN 0019      C
      ISN 0020      C
      ISN 0021      C
      ISN 0022      C
      ISN 0023      C
      ISN 0025      C
      ISN 0026      C
      ISN 0027      C
      ISN 0028      C

      ISN 0029      C
      ISN 0030      C
      ISN 0031      C
      ISN 0032      C
      ISN 0033      C
      ISN 0034      C

      ISN 0035      C
      ISN 0036      C
      ISN 0037      C
      ISN 0038      C
      ISN 0040      C
      ISN 0041      C
      ISN 0042      C

      NAME= MAIN,OPT=00,LINECNT=60,SIZE=0000K,
      SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT, ID,XREF,
      SUBROUTINE UPDTH(NITER,YSAV,N, GL,XLAM,ICR,Y,H, GL2,DY,DS,T)
      IMPLICIT REAL*8 (A-H,O-Z)

      UPDATE APPROX. INVERSE HESSIAN

      REAL*8 H(1),YSAV(N),
      REAL*8 GL2(N),DY(N),DS(N)
      REAL*8 T(1)
      REAL*4 STC,CRC
      INTEGER*2 ICR(N)
      COMMON /INPUT/ MITER,STC,CRC, IDN
      IF ( IDN.EQ.0) GO TO 15
      IF (NITER*IDN*EQ.NITER) RETURN

      CALCULATE INTERMEDIATE QUANTITIES

      15 CONTINUE
      E1=1.E-16
      E2=.1
      CALL GRADL(GL2,DY,XLAM,ICR,YSAV,N)
      DO 10 I=1,N
      DY(I)=GL(I)-GL2(I)
      DS(I)=Y(I)-YSAV(I)
      10 CONTINUE
      D=DOT(DY,DS,N)
      IF (DABS(D).GT.E1) GO TO 25
      WRITE (6,12)
      12 FORMAT (' D NEAR 0 IN UPDTH')
      RETURN
      25 CONTINUE

      COMPUTE THE PRODUCT H*DS & STORE IN GL2
      CALL LDML(H,DS,GL2, T,N)
      UPDATE H

      TX=DOT(DS,GL2,N)
      T(1)=D
      CALL COMPT(H,T,DY,N,T(N+2),T(2*N+2))
      T(1)=TX
      CALL COMPT(H,T,GL2,N,T(N+2),T(2*N+2))

      INSURE THAT H IS POS. DEFN.

      II=0
      DO 20 I=1,N
      II=II+I
      IF (H(II).LE.E1) H(II)=E2
      20 CONTINUE
      RETURN
      END

```

(10)

LEVEL 21.8 ( JUN 74 )

OS/360 FORTRAN H

DATE

```
COMPILER OPTIONS - NAME= MAIN,OPT=00,LINECNT=60,SIZE=00000K,
 SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT, ID,XREF
 SUBROUTINE CONSTR(DF,X,N,ICR,XLAN,DELFL,Y.
 IMPLICIT REAL*8 (A=H,O=Z)
 ISN 0002      C
 ISN 0003      C
               C CHECK FOR REDUNDANT CONSTRAINTS & RESET CONSTRAINT INDICATORS
 ISN 0004      C
 ISN 0005      C
 ISN 0006      C
 ISN 0007      C
 ISN 0008      C
 ISN 0009      C
 ISN 0010      C
 ISN 0011      C
 ISN 0012      C
 ISN 0013      C
 ISN 0014      C
 ISN 0015      C
               C TESTING & RESETTING LOOP
 ISN 0016      C
 ISN 0017      C
 ISN 0018      C
 ISN 0019      C
 ISN 0020      C
 ISN 0021      C
 ISN 0023      C
 ISN 0024      C
 ISN 0026      C
 ISN 0028      C
 ISN 0029      C
 ISN 0030      C
 ISN 0031      C
 ISN 0033      C
 ISN 0034      C
 ISN 0035      C
 ISN 0036      C
 ISN 0038      C
 ISN 0040      C
 ISN 0041      C
 ISN 0042      C
 ISN 0044      C
 ISN 0045      C
 ISN 0046      C
 ISN 0048      C
 ISN 0049      C
 ISN 0051      C
 ISN 0052      C
 ISN 0053      C
 ISN 0054      C
               DO 10 I=1,N
               T=DF(I)-XLAN
               TS=A(1)*DNORM**A(2)
               CR=EPS/N
               CR=DMIN1(CR,TS)
               IF (DABS(X(I))-GE.CR) GO TO 12
               CR=DMIN1(CRC/SN,TS)
               IF (DABS(T)-GE.CR) GO TO 12
               IF ((ICR(I).NE.1) IND=.TRUE.
               ICR(I)=1
               GO TO 10
               12 CONTINUE
               IF ((ICR(I).NE.2) IND=.TRUE.
               ICR(I)=2
               10 CONTINUE
               15 IF ((IOUT.EQ.0) GO TO 25
               IF ((IOUT+(NITER/IOUT).EQ.NITER) WRITE (6,21) (ICR(I),I=1,N)
               21 FORMAT (' ',ICR',',20I5)
               25 CONTINUE
               IF (.NOT.IND) RETURN
               DO 30 I=1,N
               Y(I)=X(I)
               IF ((ICR(I).NE.2) GO TO 30
               Y(I)=0.
               IF ((X(I).GT.0.) Y(I)=DSQRT(2.*X(I))
               CALL GRADL(GL,DF,XLAN,ICR,Y,N)
               RETURN
               END
END
```

(11)

LEVEL 21.8 ( JUN 74 )

DS/360 FORTRAN H DATE

```
COMPILER OPTIONS - NAME= MAIN.OPT=00.LINECNT=60.SIZE=0000K.  
SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT.ID,XREF  
SUBROUTINE GRADL(GL,DF,XLAM,ICR,Y,N)  
IMPLICIT REAL*8 (A-H,O-Z)  
  
C COMPUTE THE GRADIENT OF THE LAGRANGIAN  
  
C  
ISN 0004      REAL*8 GL(N),DF(N),Y(N)  
ISN 0005      INTEGER*2 ICR(N)  
ISN 0006      DO 10 I=1,N  
ISN 0007      GY=Y(I)  
ISN 0008      IF (ICR(I)*NE.*2) GY=-1.  
ISN 0010      10 GL(I)=(DF(I)-XLAM)*GY  
ISN 0011      RETURN  
ISN 0012      END
```

(12)

LEVEL 21.8 ( JUN 74 )

OS/360 FORTRAN H

DATE

COMPILER OPTIONS = NAME=MAIN,OPT=00,LINECNT=60,SIZE=0000K  
SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT, ID,XREF  
REAL FUNCTION SUM#8 (X,N)  
IMPLICIT REAL#8 (A=H,O=Z)

C PERFORM UTILITY FUNCTIONS

C

ISN 0004 C REAL#8 X(N),Y(N)  
ISN 0005 C REAL#8 S,SUM2  
ISN 0006 C S=0,DO  
ISN 0007 C DO 20 I=1,N  
ISN 0008 C S=S+X(I)  
ISN 0009 C SUM=S  
ISN 0010 C RETURN  
ISN 0011 C ENTRY SUM2(X,N)  
ISN 0012 C S=0,DO  
ISN 0013 C DO 30 I=1,N  
ISN 0014 C S=S+X(I)\*X(I)  
ISN 0015 C SUM2=S  
ISN 0016 C RETURN  
ISN 0017 C ENTRY DOT(X,Y,N)  
ISN 0018 C S=0,DO  
ISN 0019 C DO 40 I=1,N  
ISN 0020 C S=S+X(I)\*Y(I)  
ISN 0021 C DOT=S  
ISN 0022 C RETURN  
ISN 0023 C END

(13)

LEVEL 21:8 ( JUN 74 )

OS/360 FORTRAN H

DATE

COMPILER OPT

= MAIN,OPT=00,LINECNT=60,SIZE=0000K.

ISBN 0002  
0003  
0004

SOURCE EBCDIC, NOLIST, NODECK, LOAD, MAP, NOEDIT, ID, XREF  
SUBROUTINE MICHL SK (KK, NY, DUM)

(14)

```

ISN 0030      IF K IS NOT POS. DEF. , MAKE IT SO
ISN 0032      IF (TF.LT.0.) TF=EPS
ISN 0033      KK(J1)=TF
ISN 0034      IF (L.GT.NV) GO TO 10
ISN 0035      IRD=J1=L+1
ISN 0036      COMPUTE THE R, J-TH ELEMENT OF L USING T1
ISN 0037      DO 20 IR=L,NV
ISN 0038      IRD=IRD+IR-1
ISN 0039      T1=0.D0
ISN 0040      IF (JE1) GO TO 16
ISN 0041      DO 25 I=1,KL
ISN 0042      T1=T1-DUM(I)*KK(IRD+I)
ISN 0043      25 CONTINUE
ISN 0044      16 KK(IRD+J)=(T1+KK(IRD+J))/TF
ISN 0045      20 CONTINUE
ISN 0046      JE1=.FALSE.
ISN 0047      10 CONTINUE
ISN 0048      RETURN
ISN 0049      END
ISN 0050      DVR05180
ISN 0051      DVR05210
ISN 0052      DVR05220
ISN 0053      DVR05230
ISN 0054      DVR05240
ISN 0055      DVR05250
ISN 0056      DVR05260
ISN 0057      DVR05270
ISN 0058      DVR05290
ISN 0059      DVR05300
ISN 0060      DVR05310
ISN 0061      DVR05320
ISN 0062      DVR05340
ISN 0063      DVR05350
ISN 0064      DVR05360
ISN 0065      DVR05460

```

LEVEL 21.8 ( JUN 74 )

DS/360 FORTRAN H

DATE

```
COMPILER OPTIONS - NAME= MAIN.CPT=00•LINECNT=60•SIZE=0000K,
SOURCE=EBCDIC,NOLIST,NODECK,LOAD,MAP,NOED,IT, ID,XREF
SUBROUTINE POSOLV(A,X,Y,N)
IMPLICIT REAL*8 (A-H,O-Z)
REAL*8 A(1),X(N),Y(N)
REAL*8 S

C          SOLVE A*X=Y WHERE A IS A SYMMETRICALLY STORED MATRIX CONTAINING
C          THE MODIFIED CHOLESKY DECOMPOSITION OF ANOTHER MATRIX.

ISN 0006      C      ISB(J,I)=J*(J-1)/2+I

C          SOLVE L*X=Y & PUT RESULT IN X

ISN 0007      C      K=0
ISN 0008      DO 10 I=1,N
ISN 0009      S=Y(I)
ISN 0010      IF (I.EQ.1) GO TO 15
ISN 0012      I1=I-1
ISN 0013      DO 20 J=1,I1
ISN 0014      K=K+1
ISN 0015      S=S-A(K)*X(J)
ISN 0016      15 K=K+1
ISN 0017      10 X(I)=S

C          SOLVE D*B=Z FOLLOWDD BY L=TRANSPOSE * X=B & OVERWRITE RESULT ON X

ISN 0018      LL=N*(N+1)/2
ISN 0019      X(N)=X(N)/A(LL)
ISN 0020      IF (N.LE.1) RETURN
ISN 0022      DO 30 I1=2,N
ISN 0023      I=N-I1+1
ISN 0024      LL=LL-I1+1
ISN 0025      X(I)=X(I)/A(LL)
ISN 0026      I1=I+1
ISN 0027      S=X(I)
ISN 0028      DO 40 J=I1,N
ISN 0029      L=ISB(J,I)
ISN 0030      40 S=S-A(L)*X(J)
ISN 0031      X(I)=S
ISN 0032      RETURN
ISN 0033      END
```

(16)

LEVEL 21.8 ( JUN 74 )

OS/360 FORTRAN H

DATE /

```

COMPILER OPTIONS = NAME= MAIN,OPT=00,LINECNT=60,SIZE=0000K*
SUBROUTINE EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT, ID,XREF
IMPLICIT REAL*8 (A-H,O-Z)

ISN 0002          C THIS ROUTINE IS AN IMPLEMENTATION OF THE COMPOSITE - T ALGORITHM
ISN 0003          C TO PERFORM A RANK 1 UPDATE OF THE MCD STORED IN ARRAY LD(I,E*
C K=L*D*L*TRANS & WE WISH TO COMPUTE L* ED* S.T.K*=K+Z*Z-TRANS/T(1)
C E*K*=L*D*L*TRANS)
C THE COMPOSITE-T ALGORITHM WAS DEVELOPED BY FLETCHER & POWELL
C SEE "ON THE COMPUTATION & UPDATING OF THE MODIFIED CHOLESKY
C DECOMPOSITION OF A COVARIANCE MATRIX" BY D.L. VAN ROOY, ICSCA
C TECH.RPT. 275-025-024, RICE UNIV., HOUSTON, TX, MAY, 1976.

ISN 0004          C LD = ARRAY CONTAINING L & D STORED IN SYM. STORAGE MODE
ISN 0005          C T = AN N+1 VECTOR WHOSE FIRST ELEMENT IS AS ABOVE
ISN 0006          C Z = VECTOR OF THE UPDATE AS ABOVE
ISN 0007          C N = THE DIMENSION
ISN 0008          C V = WORKING STORAGE OF LENGTH *GE* N
                  C TMP = DOUBLE PRECISION WORKING STORAGE OF LENGTH .GE. N
                  C LOGICAL*1 TPOS,RNDERR,LALP

ISN 0009          C TPOS=T(1)*GT*0*
ISN 0010          C IF (TPOS) GO TO 35

ISN 0012          C A POINT IS TO BE DELETED
                  C EPS=5.97E-8
                  C SOLVE L*V=Z FOR V

ISN 0013          C K=1
ISN 0014          C V(1)=Z(1)
ISN 0015          C DO 10 I=2,N
ISN 0016          C 1J=I-1
ISN 0017          C S=0=DO
ISN 0018          C DD 15 J=1*IJ
ISN 0019          C K=K+1
ISN 0020          C S=S+LD(K)*V(J)
ISN 0021          C K=K+1
ISN 0022          C V(I)=Z(I)-S
ISN 0023          C 10 CONTINUE
                  C COMPUTE THE T(I*S)
ISN 0024          C K=0
ISN 0025          C RNDERR=.FALSE.
ISN 0026          C DO 20 I=1*N
ISN 0027          C K=K+1
ISN 0028          C TMP(I)=V(I)*V(I)/LD(K)
ISN 0029          C T(I+1)=T(I)+TMP(I)
ISN 0030          C IF (T(I+1).GE.0.) RNDERR=.TRUE.
ISN 0032          C 20 CONTINUE
                  C (17)

```

```

ISN 0033      C IF (.NOT.RNDERR) GO TO 35
C   C ROUNDING ERROR HAS MADE A  $\tau(i+1) \geq 0$ . SO CORRECT FOR THIS
ISN 0035      T(N+1)=EPS*T(1)
ISN 0036      DO 30 J=1,N
ISN 0037      I=N-J+1
ISN 0038      T(I)=T(I+1)-TMP(I)
ISN 0039      30 CONTINUE
ISN 0040      ISN 0041      35 CONTINUE
ISN 0042      LJ=0
ISN 0043      DO 40 I=1,N
ISN 0044      LJ=I+1
ISN 0045      V(I)=Z(I)
ISN 0046      DI=LD(I,J)
ISN 0047      IF (DI.GT.0.) GO TO 44
C   C D(I) =0. SO RANK OF D WILL EITHER INCREASE OR REMAIN UNCHANGED.
ISN 0049      C IF (DABS(V(I)).GT.1.E-30) GO TO 42
C   C RANK OF D WILL REMAIN UNCHANGED
ISN 0051      T(I+1)=T(I)
ISN 0052      GO TO 40
C   C RANK OF D WILL INCREASE BY 1
ISN 0053      42 LD(IJ)=V(I)*V(I)/T(I)
ISN 0054      IF (I.EQ.N) RETURN
ISN 0055      K=IJ
ISN 0056      DO 45 J=11,N
ISN 0057      K=K+J-1
ISN 0058      LD(K)=Z(J)/V(I)
ISN 0059      LD(IJ)=Z(J)/V(I)
ISN 0060      45 CONTINUE
ISN 0061      RETURN
ISN 0062      44 CONTINUE
C   C UPDATE D
ISN 0063      C IF (TPOS) T(I+1)=T(I)+V(I)*V(I)/DI
ISN 0065      ALP=T(I+1)/T(I)
ISN 0066      LD(IJ)=DI*ALP
ISN 0067      IF (I.EQ.N) RETURN
C   C UPDATE L & MODIFY Z ACCORDINGLY
ISN 0069      C BETA=(V(I)/DI)/T(I+1)
ISN 0070      LALP=.FALSE.
ISN 0071      C LD(IJ)=DI*ALP
IF (ALP.LE.4.) GO TO 52
C   C THIS METHOD USED TO INSURE STABILITY IF ALPHA GT. 4
LALP=.TRUE.
GAM=T(I)/T(I+1)
K=IJ
DO 50 J=11,N
K=K+J-1
ISN 0073
ISN 0074
ISN 0075
ISN 0076
ISN 0077
(18)

```

```
0078      XX=GAM*LD(K)+BETA*Z(J)
0079      Z(J)=Z(J)-V(I)*LD(K)
0080      LD(K)=XX
0081      50 CONTINUE
0082          GO TO 40
0083      52      K=LJ
0084      DO 60 J=11•N
0085          K=K+J-1
0086          Z(J)=Z(J)-V(I)*LD(K)
0087          LD(K)=LD(K)+BETA*Z(J)
0088      60 CONTINUE
0089      40 CONTINUE
0090      RETURN
0091      END
```

LEVEL 21.8 ( JUN 74 )

OS/360 FORTRAN H

DATE

COMPILER OPTIONS - NAME= MAIN.OPT=00.LINECNT=60.SIZE=00000K.  
SOURCE=EBCDIC.NCOLIST,NODECK,LOAD,MAP,NOEDIT, ID,XREF  
ISN 0002  
ISN 0003 C SUBROUTINE LDmul ( H,S,R,T,N)  
IMPLICIT REAL\*8 (A=H,D=Z)

C MULTIPLY THE MATRIX WHOSE MCD IS STORED IN R  
MODE) BY S & STORE THE RESULT IN R  
T IS WORKING STORAGE OF LENGTH .GE.N=1

ISN 0004  
ISN 0005  
ISN 0006  
ISN 0007 C REAL\*8 H(1),S(N),R(N),T(N)  
REAL\*8 SK  
DO 10 I=1,N

C MULTIPLY BY L=TRANSPOSE

ISN 0008  
ISN 0009  
ISN 0010  
ISN 0012  
ISN 0013  
ISN 0014  
ISN 0015 C IS=II  
SK=S(I)  
IF (I.GE.N) GO TO 25  
IP1=I+1  
DO 20 J=IP1,N  
IS=IS+J-1  
SK=SK+S(J)\*H(IS)  
20

C MULTIPLY BY D

ISN 0016  
ISN 0017 C 25 SK=SK\*H(II)  
T(I)=SK

C MULTIPLY BY L

ISN 0018  
ISN 0020  
ISN 0021  
ISN 0022  
ISN 0023  
ISN 0024  
ISN 0025  
ISN 0026  
ISN 0027  
ISN 0028  
ISN 0029 C IF (II.LE.1) GO TO 35  
IM1=I-1  
IJ=II-I  
DO 30 J=1, IM1  
IJ=IJ+1  
SK=SK+T(J)\*H(IJ)  
30 R(I)=SK  
II=II+I+1  
10 CONTINUE  
RETURN  
END

(20)