BEBR

FACULTY WORKING
PAPER NO. 1141

Computing Regression Quantiles

Roger W. Koenker
Vasco D'Orey

College of Commerce and Business Administration
Bureau of Economic and Business Research
University of Illinois, Urbana-Champaign

BEBR

FACULTY WORKING PAPER NO. 1141

College of Commerce and Business Administration

University of Illinois at Urbana-Champaign

April, 1985

Computing Regression Quantiles

Roger W. Koenker, Professor
Department of Economics

Vasco D'Orey, Graduate Student
Department of Economics

Computing Regression Quantiles

Roger W. Koenker*

and

Vasco D'Orey*

*Department of Economics, University of Illinois, Champaign, IL, 61801, USA.

Key Words and Phrases: Linear Models, Robust Estimation, Regression Quantiles, Empirical Processes, Parametric Linear Programming.

Language

Fortran 66

Description and Purpose

Some slight modifications of the well-known Barrodale and Roberts (1974) algorithm for least absolute error estimation of the linear regression model are described. The modified algorithm computes the regression quantile statistics of Koenker and Bassett (1978) and the associated empirical quantile (and distribution) functions. These methods have applications to robust estimation and inference for the linear model.

Theory

The ℓ_1-estimator in the linear model,

$$Y_i = x_i\beta + u_i, \quad u_i \sim iid\ F_u, \tag{1.1}$$

which solves over $b \in \mathbb{R}^p$,

$$R(b) = \sum_{i=1}^{n} |y_i - x_i b| = min! \tag{1.2}$$

provides a natural generalization of the sample median to the general linear regression model. This observation raises the question: Are there equally natural analogues of the rest of the sample quantiles for the linear model?

An affirmative answer is offered in Koenker and Bassett(1978) where p-dimensional "regression quantiles" are defined as solutions to

$$R_\theta(b) = \sum_{i=1}^{n} \rho_\theta(y_i - x_i b) = min! \tag{1.3}$$

where $\theta \in (0,1)$ and

$$\rho_\theta(u) = \begin{cases} \theta u & u \geq 0 \\ (\theta-1)u & u < 0 \end{cases}$$

In the one-sample (location) problem, solutions to (1.3) are simply the θ^{th} sample quantiles from the sample $(y_1, \cdots, y_n)$.

The asymptotic theory of the ordinary sample quantiles extends in a straightforward way to the joint asymptotic behavior of finitely many regression quantiles. See Koenker and Bassett(1978), Ruppert and Carroll(1980) and the recent work of Jureckova (1983). Thus, the theory of linear combinations of sample quantiles, or L-estimators, is available to construct robust estimators of linear models based upon regression quantiles. Perhaps more significantly, it is possible to construct trimmed least squares estimators for the linear model whose asymptotic behavior mimics the theory of the trimmed mean, see Ruupert and Carroll (1980). Recently, Jureckova (1983) has demonstrated the close connection between these trimmed least squares estimators and Huber's M-estimators for the linear model. Dejongh and DeWet(1984a,b) have also investigated this approach.

Estimates of the conditional quantile, and distribution, functions of Y given x may also be constructed based on these methods. For the model (1.1) we may define the conditional quantile function of Y at x as

$$Q_Y(\theta|x) = x_i\beta + F_u^{-1}(\theta)$$

And the conditional distribution function of Y is simply,

$$F_Y(Y|x) = sup\{\theta | Q_Y(\theta|x) \leq y\}.$$

Clearly, $F_Y(\cdot)$ is simply a translation of $F_u(\cdot)$ under the iid error hypothesis.

Bassett and Koenker(1982) propose the estimate

$$\hat{Q}_Y(\theta) \equiv \inf\{xb \,|\, R_\theta(b) = \min!\}.$$

For reasons developed there, interest focuses on $\hat{Q}_Y$ at the mean of the design, that is, on $\hat{Q}_Y(\theta) \equiv \hat{Q}_Y(\theta|\bar{x})$. The corresponding estimate of the conditional distribution function is simply,

$$\hat{F}_Y(y\,|x) = \sup\{\theta\,|\,\hat{Q}_Y(\theta|x) \leq y\}$$

and we will write $\hat{F}_Y(y)$ for $\hat{F}_Y(y\,|\bar{x})$.

In Bassett and Koenker(1982) it is shown that $\hat{Q}_Y(\cdot)$ is a proper quantile function -- a monotone jump function on the interval [0,1], and under mild regularity conditions, that the random function,

$$Z_n(\theta) = \sqrt{n}\,(F_Y(\hat{Q}_Y(\theta)) - \theta)$$

has finite dimensional distributions which converge to those of the Brownian Bridge. Portnoy (1983) has recently shown that the process $Z_n(\theta)$ is tight and consequently converges weakly to the Brownian Bridge.

Thus, $\hat{F}_Y(\cdot)$ provides a reasonable alternative to estimates based on residuals (from some preliminary estimate of the vector β) for diagnostic checking of distributional hypotheses and also perhaps for implementing recent proposals for bootstrapping and adaptive estimation of linear models which rely on estimates of the shape of the error distribution.

Method

Barrodale and Roberts(1973) proposed a modified simplex algorithm for the ℓ_1-estimation problem (1.1) which substantially improves upon earlier algorithms in speed and simplicity. Trivial modifications are required to adapt the Barrodale and Roberts algorithm to solve the "regression quantile" problem (1.3) for a fixed value of θ. One simply adds the scalar THETA to the calling sequence, declares it real, and replaces the statement immediately preceeding the statement labeled 50 with the statements:

```
WGT=SIGN(1.0,A(I,N2))
SUM=SUM + A(I,J) *(2.0 * THETA *  WGT + 1.0 - WGT)
```

However, to compute $\hat{Q}_Y(\cdot)$ and $\hat{F}_Y(\cdot)$ one must solve (1.3) for *all* values of $\theta \in [0,1]$. This is slightly more complex, requiring the solution to a *parametric* linear program. See Gal(1979) for comprehensive treatment of this general class of problems.

For any $\theta_0 \in (0,1)$, there exist solutions to the problem (1.3) of the form,

$$b_h = X_h^{-1} y_h \tag{2.1}$$

where the subscript h denotes a p-element subset of the first n integers, X_h is the $p \times p$ submatrix of X consisting of the rows indexed by h, and y_h denotes the corresponding subvector of y. Indeed the set of the solutions to (1.3) is a polytope with extreme points of this form. In the terminology of linear programming b_h is a "basic" solution.

Such a solution is optimal at θ_0 if and only if, it satisfies the subgradient condition,

$$(\theta_0-1)\mathbf{1}_p \le \sum_{i\in h}[\tfrac{1}{2}-\tfrac{1}{2}sgn(y_i-x_i b_h)-\theta_0]x_i b_h \le \theta_0\mathbf{1}_p, \tag{2.2}$$

where $\mathbf{1}_p$ denotes a p-vector of ones. Thus, for $\theta \neq \theta_0$, b_h remains optimal until these p double inequalities are violated. So, starting from θ_0, we have $2p$ inequalities in θ

$$(\theta-1) \le a_j + d_j\theta \le \theta \quad j=1,\ldots,p \tag{2.3}$$

with the a_j's and d_j's defined in the obvious way from (2.2). This decomposition of the "gradient" is stored in two new rows of the Barrodale and Roberts simplex tableau. To compute the next value of θ i.e. the value of θ at which b_h ceases to be optimal, we find

$$\theta_1 = \min_{\theta>\theta_0}\{a_j/(1-d_j),(a_j+1)/(1-d_j),\ \ j=1,\ldots,p\}. \tag{2.4}$$

At θ_1, we make one simplex pivot from b_h to a new basic solution b_h', which differs in only one element of h, recompute the a's and d's, and continue the iteration.

In practice we use instead,

$$\theta_1^* = \theta_1 + (\epsilon + \epsilon/|1+d^*|)||X||$$

where ϵ is a tolerance parameter specified below, d^* is the value the d_j at which the minimum occurs in (2.4) and $||X||$ is a norm of the design matrix. We use,

$$||X|| = \max_i \sum_{j=1}^{p} |x_{ij}|.$$

This insures a distinct new solution with $h' \neq h$. Also, the user may specify values θ_0 and θ_L at which to begin and end the iterations. The natural choice here is $\theta_0 = 1/n$ and $\theta_L = 1-1/n$. Koenker and Bassett (1978) note that the residuals $u_i(b) = y_i - x_i b$ from any solution $\hat{\beta}$, to the problem (1.3) satisfy the inequalities,

$$N = \#\{i|u_i(\hat{\beta}) < 0\} \le n\theta \le \#\{i|u_i(\hat{\beta}) \le 0\} = N+Z$$

Since $N = 0$ at $\theta = 0$, and $N = 1$ at the first jump, say θ_1, it follows that $\theta_1 \ge 1/n$. Similarly, the last jump $\theta_L \le 1 - 1/n$.

Our modified algorithm returns an array dimensioned $k\times 2$ whose first column contains a vector of quantiles and whose second column contains the mass associated with each quantile. Of course in the one sample problem, with $X = \mathbf{1}_n$, the second column is simply an n-vector with i^{th} element i/n. However, in general the mass associated with the quantiles is variable. The storage allocation for this array is somewhat problematic. For problems of modest size, say $p<10$ and $n<500$ we have found $2n<k<3n$ an adequate rule-of-thumb. However, for larger problems k may increase quite rapidly. Indeed, it is known, see Murty(1983), that there are worst-case parametric linear programs for which $p=n/2$ and $k=2^p$. Whether these examples can be adapted to the special structure of problem (1.3) is an open question, but their existence suggests that there may be no polynomial upper bound in p and n for k.

Implementation

The principle modification of the Barrodale and Roberts routine is the addition of three new rows of the array A which contains the simplex tableau. The three new rows of the tableau contain the decomposition of the marginal cost row: a's and b's appear in $A(M+2,\cdot)$ and $A(M+3,\cdot)$ respectively, and the vector $\bar{x}$ is stored in $A(M+4,\cdot)$ The only substantive change in the code is the addition of the section labeled "compute next theta". Further modifications along the lines suggested by Bloomfield and Steiger(1980) may improve the efficiency of the algorithm somewhat. The recent work of Karmarker(1984) may lead to further improvements especially for large problems. The tolerance parameter ϵ referred to above is chosen to be the smallest safely detectable value of $|x-y|/x$, see for example the routine R1MACH in Fox(1976).

Structure

CALL RQ(N,P,N5,P2,X,Y,T,TOLER,B,E,S,WX,WY,NSOL,SOLP,SOLQ,LSOL)

Formal Parameters

N	Integer	Input:	Number of observations.
P	Integer	Input:	Number of parameters.
N5	Integer	Input:	N+5
P2	Integer	Input:	P+2
X	Real(N,P)	Input:	The problem design matrix.
Y	Real(N)	Input:	The response variable.
T	Real	Input:	The desired quantile. If T is not in [0,1], the problem is solved for all T in [0,1].
TOLER	Real	Input:	A small positive constant.
NSOL	Integer	Input:	Dimension of the solution array.
S	Integer(N)	Work:	
WX	Real(N5,P2)	Work:	
WY	Real(N)	Work:	
B	Real(P)	Output:	Optimal parameters at last t.
E	Real(N)	Output:	Optimal residuals at last t.
WX(N+1,P+1)		Output:	Objective function at last t.
WX(N+1,P+2)		Output:	Rank of design matrix.
WX(N+2,P+1)		Output:	Exit code: 0 = Solution nonunique. 1 = Solution OK.

			2 = Premature end. 3 = N5 != N+5. 4 = P2 != P+2.
WX(N+2,P+2)		Output:	Number of simplex iterations.
SOLP	Real(NSOL)	Output:	A solution vector which contains the cumulative probabilit mass for each quantile.
SOLQ	Real(NSOL)	Output:	A solution vector of (monotone increasing) quantiles.
LSOL	Integer	Output:	Actual length of the solution vectors.

References

Barrodale, I. and Roberts, F.D.K. (1974). Solution of an overdetermined system of equations in the l1 norm. *Communications of the ACM,* 17, 319-329 .

Bassett, G., Koenker, R. (1978). The asymptotic distribution of the least absolute error estimator. *Journal of the American Statistical Association,* 73, 618-622 .

Bassett, G. and Koenker, R. (1982). An empirical quantile function for linear models with iid errors. *Journal of the American Statistical Association,* 77, 407-415 .

Bloomfield P. and Steiger, W. L. (1980) Least Absolute Deviations Curve-Fitting, *SIAM Journal of Scientific and Statistical Ccomputing,* 1, 290-301.

Dejongh, P.J. and DeWet, T. (1984). Adaptive Regression Trimmed Means, unpublished.

Fox, P. (1976) *PORT Mathematical Subroutine Library* Murray Hill: Bell Laboratories

Gal, T. (1979). *Postoptimal Analyses,Parametric Programming and Related Topics.* New York : McGraw-Hill.

Jureckova, J. and Sen, P.K. (1984) On Adaptive Scale-Equivariant M-Estimators in Linear Models, *Statistics and Decisions,* forthcoming.

Karmarker, N. (1984) A New Polynomial Time Algorithm for Linear Programming *Combinatorica,* forthcoming.

Koenker, R.W. and Bassett, G.W. (1978). Regression quantiles. *Econometrica,* 46 , 33-50 .

Murty, K.G. (1983) *Linear Programming,* Wiley.

Portnoy, S. (1984) Tightness of the Sequence of Empiric CDF Processes Defined from Regression Fractils, in Franke, J., Handle, W. and Martin, D. (eds.) *Robust and Nonlinear Time Series Analysis,* New York: Springer-Verlag.

Ruppert, D. and Carroll, R.J. (1980). Trimmed least squares estimation in the linear model. *Journal of the American Statistical Association,* 75, 828-838 .

EXAMPLE: THE STACKLOSS DATA

```
      PROGRAM MAIN
      REAL X(21,3),WX(26,6),Y(21),WY(21),E(21),B(4),SOLP(42),SOLQ(42)
      INTEGER S(21),LSOL
      DATA TOLER/1.2E-7/
      DO 1 I=1,21
      X(I,1)=1.0
    1 CONTINUE
      READ (5,*) ((X(I,J),J=2,4),Y(I),I=1,21)
      CALL RQ(21,4,26,6,X,Y,2.,TOLER,B,E,S,WX,WY,42,SOLP,SOLQ,LSOL)
      WRITE(6,10)WX(23,5)
10    FORMAT("EXIT CODE=",F5.0)
      WRITE(6,20)(SOLP(I),SOLQ(I),I=1,LSOL)
20    FORMAT(2F16.3)
      STOP
      END
```

Stackloss Data

x1	x2	x3	y
80	27	89	42
80	27	88	37
75	25	90	37
62	24	87	28
62	22	87	18
62	23	87	18
62	24	93	19
62	24	93	20
58	23	87	15
58	18	80	14
58	18	89	14
58	17	88	13
58	18	82	11
58	19	93	12
50	18	89	8
50	18	86	7
50	19	72	8
50	19	79	8
50	20	80	9
56	20	82	15
70	20	91	15

Output from a VAX-11/780

exit code= 1.

0.12411893	13.45404339
0.13007915	13.99367046
0.13038845	15.30951786
0.14944933	15.30952358
0.16074213	15.30952358
0.22314128	15.30952072
0.25399303	15.30952168
0.27513024	15.30952549
0.33102346	16.16141319
0.37501332	16.44413567
0.39190131	16.80134010
0.40951341	16.95934868
0.48986971	17.42450523
0.56481242	17.43436623
0.59239787	17.44517708
0.60424811	17.45659256
0.62001455	19.13624954
0.65115529	19.13750839
0.68975174	19.14842606
0.76212549	19.15640259
0.76345610	19.19264221
0.77394605	19.71523857
0.77770203	19.98903847
0.81431276	20.12132454
0.83394426	20.16070366
0.91308522	20.20633698
1.00000000	21.70072937

```
      SUBROUTINE RQ(M,N,M5,N2,A,B,T,TOLER,X,E,S,WA,WB,NSOL,
     *SOLP,SOLQ,LSOL)
      DOUBLE PRECISION SUM
      REAL MIN,MAX,A(M,N),X(N),WA(M5,N2),WB(M),E(M)
      REAL B(M),SOLP(NSOL),SOLQ(NSOL)
      INTEGER OUT, S(M)
      LOGICAL STAGE, TEST,INIT,IEND
      DATA BIG/1.E37/
C
C  INITIALIZATION
C
      M1 = M+1
      N1 = N+1
      M2 = M+2
      M3 = M+3
      M4 = M+4
      DO 2 I=1,M
      SUM = 0.0
      WB(I)=B(I)
      DO 1 J=1,N
      WA(I,J)=A(I,J)
      SUM = SUM + ABS(A(I,J))
  1   CONTINUE
      IF(SUM .GT. AMG)AMG = SUM
  2   CONTINUE
      IF(M5 .NE. M+5)WA(M2,N1) = 3.
      IF(N2 .NE. N+2)WA(M2,N1) = 4.
      IF(WA(M+2,N+1) .GT. 2.)RETURN
      DIF = 0.0
      IEND = .TRUE.
      IF(T .GE. 0.0 .AND. T .LE. 1.0)GOTO 3
      T0 = 1./FLOAT(M)-TOLER
      T1 = 1. - T0
      T = T0
      IEND = .FALSE.
  3   CONTINUE
      INIT = .FALSE.
      LSOL = 1
      KOUNT = 0
      DO 9 K=1,N
      WA(M5,K) = 0.0
      DO 8 I=1,M
      WA(M5,K) = WA(M5,K) + WA(I,K)
  8   CONTINUE
      WA(M5,K) = WA(M5,K)/FLOAT(M)
  9   CONTINUE
      DO 10 J=1,N
      WA(M4,J) = J
      X(J) = 0.
 10   CONTINUE
      DO 40 I=1,M
      WA(I,N2) = N+I
      WA(I,N1) = WB(I)
      IF(WB(I).GE.0.)GOTO 30
      DO 20 J=1,N2
      WA(I,J) = -WA(I,J)
 20   CONTINUE
 30   E(I) = 0.
 40   CONTINUE
      DO 42 J=1,N
      WA(M2,J) = 0.0
      WA(M3,J) = 0.0
      DO 41 I=1,M
      AUX = SIGN(1.0,WA(M4,J)) * WA(I,J)
```

```
      WA(M2,J) = WA(M2,J) + AUX * (1.0 - SIGN(1.0,WA(I,N2)))
      WA(M3,J) = WA(M3,J) + AUX * SIGN(1.0,WA(I,N2))
 41   CONTINUE
      WA(M3,J) = 2.0 * WA(M3,J)
 42   CONTINUE
      GO TO 48
 43   CONTINUE
      LSOL = LSOL + 1
      DO 44 I=1,M
      S(I) = 0.0
 44   CONTINUE
      DO 45 J=1,N
      X(J) = 0.0
 45   CONTINUE
C
C  COMPUTE NEXT T
C
      SMAX = 2.0
      DO 47 J=1,N
      B1 =  WA(M3,J)
      A1 = (-2.00 - WA(M2,J))/B1
      B1 = -WA(M2,J)/B1
      IF(A1 .LT. T)GO TO 46
      IF(A1 .GE. SMAX) GO TO 46
      SMAX = A1
      DIF = (B1 - A1 )/2.00
 46   IF(B1 .LE. T) GO TO 47
      IF(B1 .GE. SMAX)GO TO 47
      SMAX = B1
      DIF = (B1 - A1)/2.00
 47   CONTINUE
      TNT = SMAX + TOLER * (1.00 + ABS(DIF)) * AMG
      IF(TNT .GE. T1 + TOLER)IEND = .TRUE.
      T = TNT
      IF(IEND)T = T1
 48   CONTINUE
C
C COMPUTE NEW MARGINAL COSTS
C
      DO 49 J=1,N
      WA(M1,J) = WA(M2,J) + WA(M3,J) * T
 49   CONTINUE
      IF(INIT) GO TO 265
C
C STAGE 1
C
C DETERMINE THE VECTOR TO ENTER THE BASIS
C
      STAGE=.TRUE.
      KR=1
      KL=1
 70   MAX=-1.
      DO 80 J=KR,N
      IF(ABS(WA(M4,J)).GT.N)GOTO 80
      D=ABS(WA(M1,J))
      IF(D.LE.MAX)GOTO 80
      MAX=D
      IN=J
 80   CONTINUE
      IF(WA(M1,IN).GE.0.)GOTO 100
      DO 90 I=1,M4
      WA(I,IN)=-WA(I,IN)
 90   CONTINUE
C
```

```
C DETERMINE THE VECTOR TO LEAVE THE BASIS
C
 100   K=0
       DO 110 I=KL,M
       D=WA(I,IN)
       IF(D.LE.TOLER)GOTO 110
       K=K+1
       WB(K)=WA(I,N1)/D
       S(K)=I
       TEST=.TRUE.
 110   CONTINUE
 120   IF(K.GT.0)GOTO 130
       TEST=.FALSE.
       GOTO 150
 130   MIN=BIG
       DO 140 I=1,K
       IF(WB(I).GE.MIN)GOTO 140
       J=I
       MIN=WB(I)
       OUT=S(I)
 140   CONTINUE
       WB(J)=WB(K)
       S(J)=S(K)
       K=K-1
C
C CHECK FOR LINEAR DEPENDENCE IN STAGE 1
C
 150   IF(TEST.OR..NOT.STAGE)GOTO 170
       DO 160 I=1,M4
       D=WA(I,KR)
       WA(I,KR)=WA(I,IN)
       WA(I,IN)=D
 160   CONTINUE
       KR=KR+1
       GOTO 260
 170   IF(TEST)GOTO 180
       WA(M2,N1)=2.
       GOTO 390
 180   PIVOT=WA(OUT,IN)
       IF(WA(M1,IN)-PIVOT-PIVOT.LE.TOLER)GOTO 200
       DO 190 J=KR,N1
       D=WA(OUT,J)
       WA(M1,J)=WA(M1,J)-D-D
       WA(M2,J)=WA(M2,J)-D-D
       WA(OUT,J)=-D
 190   CONTINUE
       WA(OUT,N2)=-WA(OUT,N2)
       GOTO 120
C
C PIVOT ON WA(OUT,IN)
C
 200   DO 210 J=KR,N1
       IF(J.EQ.IN)GOTO 210
       WA(OUT,J)=WA(OUT,J)/PIVOT
 210   CONTINUE
       DO 230 I=1,M3
       IF(I.EQ.OUT)GOTO 230
       D=WA(I,IN)
       DO 220 J=KR,N1
       IF(J.EQ.IN)GOTO 220
       WA(I,J)=WA(I,J)-D*WA(OUT,J)
 220   CONTINUE
 230   CONTINUE
       DO 240 I=1,M3
```

```
      IF(I.EQ.OUT)GOTO 240
      WA(I,IN)=-WA(I,IN)/PIVOT
 240  CONTINUE
      WA(OUT,IN)=1./PIVOT
      D=WA(OUT,N2)
      WA(OUT,N2)=WA(M4,IN)
      WA(M4,IN)=D
      KOUNT=KOUNT+1
      IF(.NOT.STAGE)GOTO 270
C
C INTERCHANGE ROWS IN STAGE 1
C
      KL=KL+1
      DO 250 J=KR,N2
      D=WA(OUT,J)
      WA(OUT,J)=WA(KOUNT,J)
      WA(KOUNT,J)=D
 250  CONTINUE
 260  IF(KOUNT+KR.NE.N1)GOTO 70
C
C STAGE 2
C
 265  STAGE=.FALSE.
C
C DETERMINE THE VECTOR TO ENTER THE BASIS
C
 270  MAX=-BIG
      DO 290 J=KR,N
      D=WA(M1,J)
      IF(D.GE.0.)GOTO 280
      IF(D.GT.(-2.))GOTO 290
      D=-D-2.
 280  IF(D.LE.MAX)GOTO 290
      MAX=D
      IN=J
 290  CONTINUE
      IF(MAX.LE.TOLER)GOTO 310
      IF(WA(M1,IN).GT.0.)GOTO 100
      DO 300 I=1,M4
      WA(I,IN)=-WA(I,IN)
 300  CONTINUE
      WA(M1,IN)=WA(M1,IN)-2.
      WA(M2,IN)=WA(M2,IN)-2.
      GOTO 100
C
C COMPUTE QUANTILES
C
 310  CONTINUE
      DO 320 I=1.KL-1
      K=WA(I,N2)*SIGN(1.0,WA(I.N2))
      X(K) = WA(I,N1) * SIGN(1.0,WA(I,N2))
 320  CONTINUE
      SUM=0.0
      DO 330 I=1,N
      SUM=SUM + X(I) * WA(M5,I)
 330  CONTINUE
      SOLP(LSOL) = T
      SOLQ(LSOL) = SUM
      IF(IEND)GO TO 340
      INIT = .TRUE.
      GO TO 43
 340  CONTINUE
      DO 350 I = 2,LSOL
      SOLP(I-1)=SOLP(I)
```

```
350   CONTINUE
      LSOL=LSOL-1
      SOLP(LSOL)=1.
      L =KL-1
      DO 370 I=1,L
      IF(WA(I,N1).GE.0.)GOTO 370
      DO 360 J=KR,N2
      WA(I,J)=-WA(I,J)
360   CONTINUE
370   CONTINUE
      WA(M2,N1)=0.
      IF(KR.NE.1)GOTO 390
      DO 380 J=1,N
      D=ABS(WA(M1,J))
      IF(D.LE.TOLER.OR.2.-D.LE.TOLER)GOTO 390
380   CONTINUE
      WA(M2,N1)=1.
390   DO 400 I=KL,M
      K = WA(I,N2) * SIGN(1.0,WA(I,N2))
      D = WA(I,N1) * SIGN(1.0,WA(I,N2))
      K=K-N
      E(K)=D
400   CONTINUE
      WA(M2,N2)=KOUNT
      WA(M1,N2)=N1-KR
      SUM = 0.0
      DO 410 I=1,M
      SUM = SUM + E(I)*(.5 + SIGN(1.0,E(I))*(T-.5))
410   CONTINUE
      WA(M1,N1) = SUM
      RETURN
      END
```