# Impedance Estimation

by Jon Claerbout and Ted Madden

Certain statistical problems arise in the estimation of impedance functions. These problems are as yet unfamiliar in reflection seismology, but as we attempt to extract more information from reflection seismograms these problems will become more apparent. As a guide to the future in seismics we will take a look at the present in electromagnetic interpretation.

We first consider magnetotellurics. Ordinarily one measures three magnetic components ( $h_x$, $h_y$, $h_z$ ) and two electric components ( $e_x$, $e_y$ ) of the earth's natural fields. Physically these should be related by a matrix operator like

$$\begin{bmatrix} e_x \\ e_y \end{bmatrix} \approx \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \end{bmatrix} \begin{bmatrix} h_x \\ h_y \\ h_z \end{bmatrix} \quad \text{or} \quad E = R\,H \tag{1}$$

The matrix times vector operation in (1) is a complex multiplication for each fourier frequency component. Likewise there is an inverse relation

$$\begin{bmatrix} h_x \\ h_y \\ h_z \end{bmatrix} \approx \begin{bmatrix} s_{11} & s_{12} \\ s_{21} & s_{22} \\ s_{31} & s_{32} \end{bmatrix} \begin{bmatrix} e_x \\ e_y \end{bmatrix} \quad \text{or} \quad H = S\,E \tag{2}$$

Substituting (2) into (1) we get

$$E \approx R\,H \approx R\,S\,E$$

or

$$R\,S \approx I \tag{3}$$

Now since $R$ and $S$ are ordinarily estimated as $\hat{R}$ and $\hat{S}$ by different statistical procedures we may wonder about the quality of the approximation (3).

To simplify the discussion let us talk about vectors $x$ and $y$ where components of $x$ and $y$ should satisfy the relations

$$x = a\,y \tag{4a}$$

$$y = b\,x \tag{4b}$$

where $a = 1/b$ but they do not because $x$ has an independent random number added to each component and so does $y$. Hence instead of (4) we have

$$x \approx a\,y \tag{5a}$$

$$y \approx b\,x \tag{5b}$$

Suppose further that the corrupting noises are unstationary and unknown. The simple minded least squares estimates of $a$ and $b$ from (5) are

$$\hat{a} = (x \cdot y) / (y \cdot y) \tag{6a}$$

$$\hat{b} = (y \cdot x) / (x \cdot x) \tag{6b}$$

Multiplying (6a) and (6b) together we get the well known inequality

$$\hat{a}\,\hat{b} = \frac{(x \cdot y)\,(y \cdot x)}{(y \cdot y)\,(x \cdot x)} \leq 1 \tag{7}$$

which is to be compared to the desired product (3). The inequality arises because additive noise biases both $\hat{a}$ and $\hat{b}$ to have denominators which are too big. The more noise the worse the bias. Even with an infinite amount of data we are unable to average away the bias.

Next we noticed that estimates of $a$ and $b$ formed by medians are better behaved. Specifically, if

$$\hat{a} = \underset{i}{\text{median}} \left\{ x_i / y_i \right\} \tag{8a}$$

$$\hat{b} = \underset{i}{\text{median}} \left\{ y_i / x_i \right\} \tag{8b}$$

then it is easy to see that

$$\hat{a}\,\hat{b} = 1 \tag{9}$$

This led us to believe that we were in search of some sort of matrix median for the magnetotelluric problem. However, there is a pitfall here. For a while we assumed that if (9) was satisfied that $\hat{a}$ and $\hat{b}$ would not be biased and that averaging an infinite amount of data would always lead to the correct limits ( $\hat{a} \to a$ ) and ( $\hat{b} \to b$ ). To see that this is not so imagine that $a$ is positive, $x_i$ contains no random additive component, but that $y_i$ contains a massive amount of random noise. To compute (8a) we arrange the values of $x_i/y_i$ in a table of descending numerical value as in table 1. Then select the median as the middle element in the table. Clearly the answer is nearly zero regardless of the true value $a$ .

| value of $x_i/y_i$ | because $y_i$ happens to be | size of group |
|---|---|---|
| near $+\infty$ | near zero | tiny |
| near $a$ | nearly noise free | tiny |
| near $1/+\infty$ | noisy with $x_i/y_i$ positive | nearly 50% |
| near $1/(-\infty)$ | noisy with $x_i/y_i$ negative | nearly 50% |
| near $-\infty$ | near zero | tiny |

Table 1. Given $x_i/y_i$ has $y_i$ polluted by a massive amount of additive noise this table tabulates $x_i/y_i$ in numerical order. The middle value of $x_i/y_i$ on this table obviously takes a numerical value near zero.

Now we are faced with the realization that we do not have an averaging technique which, given an infinite amount of data, ensures convergence to the correct answer ( $\hat{a} \to a$ ). We concluded this discussion with the idea that by iterative techniques we could always hope to satisfy non-linear constraints like

$$\hat{a}\,\hat{b} = 1 \tag{10a}$$

or

$$\hat{R}\,\hat{S} = I \tag{10b}$$

whether we use $L_1$ or $L_2$ norms. Although such constraints seem like a good idea they do not ensure the convergence ( $\hat{a} \to a$ ) or ( $\hat{b} \to b$ ).

Despite the fact that the median won't eliminate bias even with the constraints (10) and because of the necessity of robust processing field data we decided to go ahead anyway in an effort to get a generalized median. We decided to define a scalar $E$ (like a mechanical potential energy function) to be minimized which is such that each data point contributes a unit vector to the error gradient (like a unit of physical force). In the regression

$$z_t \approx a\,x_t + b\,y_t \tag{11}$$

the potential energy $E$ where

$$E = \sum_t \left| \frac{z_t - a\,x_t - b\,y_t}{(x_t^2 + y_t^2)^{1/2}} \right| \tag{12}$$

has the desired gradient

$$\begin{bmatrix} \partial_a \\ \partial_b \end{bmatrix} E = \sum_t -(x^2 + y^2)^{-1/2} \begin{bmatrix} +x_t \\ +y_t \end{bmatrix} \operatorname{sgn}(z_t - a\,x_t - b\,y_t) \tag{13}$$

in which each time point contributes a unit vector. This puts all data points on an equal basis. To actually determine numerical parameters for a and b it is anticipated that iterative adjustment in the direction of the negative gradient would provide a useful technique in many applications. Unfortunately without having had any real practical experience with these estimation procedures we are unable to identify the important pitfalls which seem certain to arise.

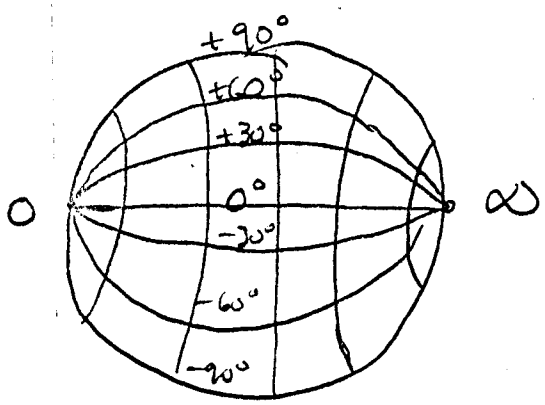
January, 1974

Dear Jon,

I received your typed discussion on estimation today, and after going in a few circles I think I found a mistake which changes the conclusions, although I haven't worked out the new conclusion. The mistake is the statement

$$\hat{a} = \text{median} \quad \{x_i/y_i\} = \frac{1}{\hat{b}} = \frac{1}{\text{median}\ \{\frac{y_i}{x_i}\}}$$

which became obvious when considering your example of $x$ noise free for in this case $\hat{b}$ should be correct (at least in the trivial case where all $x_i = x_o$). The problem comes in the ordering, since small positive $x_i/y_i$ are lumped in with small negative $x_i/y_i$ and the median is very biased, while large pos. and neg. $y_i/x_i$ are separated and they don't bias the result. If we use the same ordering but constrain $\hat{a}\,\hat{b} = 1$ our result is still biased (but less so), but I wonder if our log $x_i/y_i$ on a cylinder will help. I guess not. We need an averaging that dissipates the influence of ratio terms with equal magnitudes but opposite signs irrespective of the size of the ratio. How about $\text{sgn}(\frac{x}{y})\ \ell n\ \frac{x}{y}$ ( I think the use of $\ell n$ guarantees $\hat{a}\,\hat{b} = 1$ without using this as a constraint). When $x$ and $y$ are complex I am not sure of the strategy. We need a way of distinguishing between $\text{sgn}(\frac{x}{y})$ and $\text{sgn}(\ell n\ \frac{x}{y})$ and still be scale invariant. How about a spherical surface? No as $\pm 0$ or $\pm\infty$ are lumped together. The cylinder is better but the averaging law must allow cancellation for elements with equal $(\frac{x}{y})$ but a balanced distribution of angles.




If we use $\mathrm{sgn}(\frac{x}{y})\,\ell n(\frac{x}{Ky})$ where $K$ is chosen such that almost all $Ky_i > x_i$ we avoid a sign ambiguity except for a small number of cases. Taking as an example $x_i = 1$, $a = 1$ which is a sort of gaussian looking distribution with $|N| > y$, and taking $K > 1$ and throwing out $y_i + N_i = 0$ (or putting half in $+0$, half in $-0$) the median $\mathrm{sgn}(\frac{x}{y})\,\ell n(\frac{x}{Ky})$ falls just inside the set associated with $a = 1$ and thus is unbiased.

| $y_i+N_i$ | No. i's |
|---|---|
| 1 | 10 |
| 0, 2 | 9 |
| -1, 3 | 9 |
| -2, 4 | 9 |
| -3, 5 | 9 |
| -4, 6 | 8 |
| -5, 7 | 8 |
| -6, 8 | 8 |
| -7, 9 | 7 |
| -8, 10 | 7 |
| -9, 11 | 6 |
| -10, 12 | 5 |
| -11, 13 | 4 |
| -12, 14 | 3 |
| -13, 15 | 2 |
| -14, 16 | 1 |
| -15, 17 | 1 |
| -16, 18 | 1 |
| -17, 19 | 1 |
| -18, 20 | 1 |

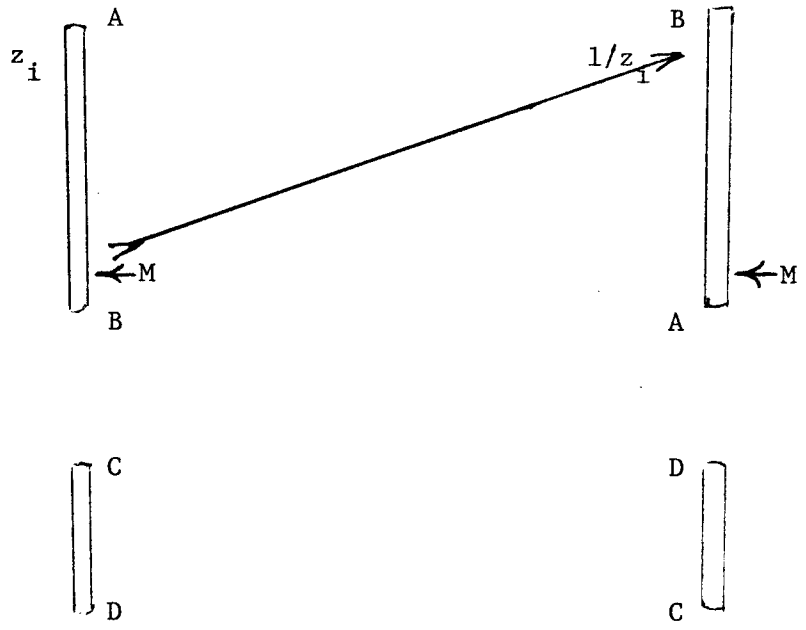
I guess this same trick will work for complex ratios $e^{i\theta}\,\ell n(\frac{x}{Ky})$, $\theta = \mathrm{phase}(\frac{x}{y})$.

It certainly was a good stay for us all. Thanks for all your help. I hope you all have a good holiday season and best wishes for the New Year.

Ted Madden *

* Hand written letter re-typed at Stanford(se)

Diagram to illustrate that for signed numbers $z_i$ , median $(z_i) \neq 1/\text{median}\ (1/z_i)$




Madden's approach is to put $\theta_i = \arctan\ (x_i/y_i)$ on a cylinder. Seems like a good idea, but we don't know anything about uniqueness or computation of such medians.

# $L_1$ Norm Program and Test Case - Jon F. Claerbout

2/73

```
$JOB            CLAERBOUT       BIN=DO3
C       DEAR GANG,
C       EARLY ON IN THIS RESEARCH I WAS HOPING TO COME UP WITH AN
C       ALGORITHM WHICH WOULD BE NEARLY OPTIMAL IN MOST APPLICATIONS
C       AND AT LEAST COULD ALWAYS BE EXPECTED TO WORK IN ANY
C       APPLICATION.  I DIDN'T ACHIEVE EITHER GOAL.  THE PRESENT
C       ALGORITHM IS FAR FROM OPTIMAL FOR THE LARGE  M  WHICH OCCUR
C       IN APPLICATIONS WHERE UNDERDETERMINED SOLUTIONS ARE TO BE
C       SMOOTHED.   I AM CONFIDENT OF THE EXISTANCE
C       OF MUCH BETTER ALGORITHMS  FOR SUCH SPARSE MATRIX CASES.
C       FURTHERMORE I DISCOVERED SOME PATHOLOGICAL CASES IN WHICH
C       MY PRESENT ALGORITHM DOESN'T WORK.  IN LINEAR PROGRAMMING THESE
C       CASES ARE CALLED DEGENERATE.  THESE CASES ARISE ONLY WHEN MORE
C       THAN PRECISELY  M  OF THE  N  EQUATIONS TURN OUT TO BE EXACTLY
C       SATISFIED.  OFTEN THIS DOESN'T HURT BUT SOMETIMES AN EXIT THEN
C       OCCURS BEFORE YOU HIT THE TRUE MINIMUM.  THIS HAPPENED SEVERAL
C       TIMES IN TEST CASES WHERE THE DATA VECTOR CONTAINED INTEGERS.
C       USUALLY YOU ONLY HAVE TO LOOK IN M DIRECTIONS TO SEE IF
C       DESCENT IS POSSIBLE.  THESE ARE THE DIRECTIONS YOU GET FROM
C       CASTING OUT ONE EQUATION AT A TIME FROM A BASIS OF M EQUATIONS.
C       IN THE DEGENERATE CASE (MORE THAN M EXACTLY SATISFIED) THERE
C       ARE MORE DIRECTIONS (SETS OF M-1 EQUATIONS).  THE NUMBER OF
C       DIRECTIONS IS SOME PREPOSTEROUS FACTORIAL FUNCTION OF THE
C       AMOUNT OF DEGENERACY.
C       PRESENTLY THE ONLY TWO WAYS I KNOW TO GET AROUND THIS ARE
C       EITHER TO ADD A LITTLE NOISE TO THE DATA, OR TO DO WHAT THEY
C       DO IN LINEAR PROGRAMMING, WHICH IS SOMEWHAT HAIRY AND AMOUNTS
C       TO PRETENDING YOU HAVE ADDED A LITTLE NOISE.  IT DIDN'T SEEM
C       WORTH THE TROUBLE.
C       BEST REGARDS,
C       JON F. CLAERBOUT
C        THIS TEST CASE FITS SINUSOIDS TO A STEP.
C       BELIEVE IT OR NOT THIS IS A VERY ODD PATHOLOGICAL TEST CASE.
C       IT RUNS IN ABOUT .5 SEC ON PRINCETON'S IBM 36091 IN WATFIV.
 1          DIMENSION A(41,14),X(14)
 2          DIMENSION D(41),E(41),GU(41),GD(41)
 3          DIMENSION NOW(14)
 4          DATA D/20*-1.,20*+1.,0./
C       NOW=POINTERS TO PRESENT BASIS EQUATIONS, INITIALLY NULL.
 5          DATA NOW/14*0/
C       SMALL=ABOUT (10**-5)* TYPICAL D
 6          SMALL=1.E-5
 7          N=40
 8          ND=41
C       HERE WE SET UP WEIGHTS FOR  L1  NORM FIT.
 9          DATA GU,GD/41*1.,41*-1./
10          M=4
C       FILL IN COEFFICIENT MATRIX.
11          DO 10 I=1,N
12          ARG=(I-N/2-.5)*3.14159265/N
13          DO 10 J=1,M,2
14          A(I,J)=COS(ARG*(J-1))
15    10    A(I,J+1)=SIN(ARG*J)
16          CALL        ELSKEW(ND,N,M,A,D,GU,GD,SMALL,NOW,X)
17          DO 20 I=1,N
18          E(I)=0.
19          DO 20 J=1,M
20    20    E(I)=E(I)+A(I,J)*X(J)
21          CALL SPLOT(N,E,D)
22          STOP
```

```
      END

      SUBROUTINE ELSKEW(ND,N,M,A,D,GU,GD,SMALL,NOW,X)
C FIND X(I) TO MINIMIZE
C                N                    M
C       ESUM = SUM SKEWNORM(K, SUM (D(K)-A(K,I)*X(I)) )
C               K=1                  I=1
C WHERE                     ( GU(K)*(ER-SMALL) IF ER.GT.+SMALL      GU.GT.0
C        SKEWNORM(K,ER) =   ( GD(K)*(ER+SMALL) IF ER.LT.-SMALL      GD.LT.0
C                           (  0.              IF ABS(ER).LE.SMALL.GE.0.
      DIMENSION A(ND,M),X(M),D(ND),GU(ND),GD(ND),NOW(M)
      DIMENSION W(41),F(41),K(41)
      DIMENSION B(14,14),COL(14)
      CALL       ENIT(ND,N,M,K,A,D,X,B,NOW,F,GU,GD,SMALL)
      LOOP=0
C     IF ONLY GU AND GD CHANGED YOU MAY REENTER HERE.
      ENTRY AGAIN
 50   LOOP=LOOP+1
      CALL       HUGO(ND,N,M,A,F,GU,GD,SMALL,B,KICK,COL,NOW)
      IF(KICK.EQ.0) RETURN
C     FIND SCALAR T WHERE X=X0+(COL OF B)*T
      DO 60  I=1,N
      W(I)=0.
      DO 60 J=1,M
 60   W(I)=W(I)+A(I,J)*COL(J)
      CALL SKEWER(ND,N,W,F,GU,GD,SMALL,K,T,ML,MH)
      WRITE(6,77) (K(I),I=ML,MH)
 77   FORMAT(40I3)
C     PICK OUT A NEW BASIS EQUATION
      NEW=K(ML)
      DO 70 L=ML,MH
 70   IF(ABS(W(NEW)).LT.ABS(W(K(L)))) NEW=K(L)
      NOW(KICK)=NEW
      CALL       REBASE(ND,N,M,A,B,KICK,NEW)
      T=F(NEW)/W(NEW)
      IF(T.EQ.0..AND.LOOP.GT.M) RETURN
      DO 75 J=1,M
 75   X(J)=X(J)+COL(J)*T
      ESUM=0.
      DO 80 I=1,N
      F(I)=F(I)-W(I)*T
      IF(F(I).GT.SMALL) ESUM=ESUM+GU(I)*F(I)
 80   IF(F(I).LT.-SMALL)ESUM=ESUM+GD(I)*F(I)
      WRITE(6,71) T,(X(J),J=1,M),ESUM
 71   FORMAT(1X,10E12.5)
      GO TO 50
      END

      SUBROUTINE ENIT(ND,N,M,K,A,D,X,B,NOW,F,GU,GD,SMALL)
C     INITIALIZATION  ('INIT' HAPPENS TO BE A SYSTEM ENTRY AT PRINCETON)
      DIMENSION A(ND,M),D(ND),X(M),NOW(M),F(ND),K(ND)
      DIMENSION GU(N),GD(N)
      DIMENSION B(14,14),COL(14)
      DO 30 J=1,M
C     INITIALIZE SOLUTION  X  TO ZERO.
      X(J)=0.
      DO 10 I=1,M
 10   B(I,J)=0.
      SC=0.
      DO 20 I=1,N
```

```
 69     20   SC=SC+ABS(A(I,J))
        C    INITIALIZE BASIS INVERSE MATRIX  B  TO DIAG.
 70     30   B(J,J)=N/SC
 71          DO 40 I=1,N
        C    INITIALIZE RESIDUAL VECTOR F(I) TO D(I)
 72          F(I)=D(I)
        C    INITIALIZE EQUATION POINTERS TO  1  THRU  N .
 73     40   K(I)=I
        C    GET BASIS EQNS WHICH WERE OUTPUT FROM PREVIOUS CALL, IF ANY.
        C    THIS IS VERY GOOD IF OVERALL PROBLEM IS A SEQUENCE OF SIMILAR ONES
 74          DO 80 KICK=1,M
 75          NEW=NOW(KICK)
 76          IF(NEW.LE.0.OR.NEW.GT.N) GO TO 80
 77          CALL       REBASE(ND,N,M,A,B,KICK,NEW)
 78     80   CONTINUE
        C    FINALLY UPDATE SOLUTION AND RESIDUAL FOR THE SELECTED BASIS.
 79          DO 65 I=1,M
 80          DO 60 J=1,M
 81          NEW=NOW(J)
 82          IF(NEW.LE.0.OR.NEW.GT.N) GO TO 60
 83          X(I)=X(I)+B(I,J)*D(NEW)
 84     60   CONTINUE
 85     65   CONTINUE
 86          DO 50 I=1,N
 87          DO 50 J=1,M
 88     50   F(I)=F(I)-A(I,J)*X(J)
        C    COMPUTE AND PRINT ERROR SUM=ESUM.
 89          ESUM=0.
 90          DO 90 I=1,N
 91          IF(F(I).GT.SMALL)  ESUM=ESUM+GU(I)*F(I)
 92     90   IF(F(I).LT.-SMALL) ESUM=ESUM+GD(I)*F(I)
 93          WRITE(6,71) SC,(X(J),J=1,M),ESUM
 94     71   FORMAT(1X,10E12.5)
 95          RETURN
 96          END

 97          SUBROUTINE HUGO(ND,N,M,A,F,GU,GD,SMALL,B,KICK,COL,NOW)
        C    SELECT A DIRECTION BY TAKING SOME COLUMN OUT OF INVERSE BASIS.
 98          DIMENSION A(ND,M),F(ND),GU(ND),GD(ND)
 99          DIMENSION NOW(M)
100          DIMENSION B(14,14),COL(14)
101          DIMENSION G(14),GP(14),GM(14)
102          DO 110 I=1,M
103          GP(I)=0.
104          GM(I)=0.
105     110  G(I)=0.
106          DO 135 L=1,N
107          IF(ABS(F(L)).LT.SMALL) GO TO 120
108          IF(F(L).GT.0.)HIT=GU(L)
109          IF(F(L).LT.0.)HIT=GD(L)
110          DO 115 J=1,M
111     115  G(J)=G(J)-A(L,J)*HIT
112          GO TO 135
113     120  DO 130 I=1,M
114          WT=0.
        C    IF YOU ARE SURE THAT YOU WON'T HAVE DEGENERACY YOU CAN SAVE
        C     TIME BY REPLACING NEXT DO LOOP BY   WT=1.
115          DO 125 J=1,M
116     125  WT=WT+A(L,J)*B(J,I)
117          IF(WT.LT.0.)GP(I)=GP(I)-GU(L)*WT
```

```
118           IF(WT.GT.0.)GP(I)=GP(I)-GD(L)*WT
119           IF(WT.GT.0.)GM(I)=GM(I)-GU(L)*WT
120      130  IF(WT.LT.0.)GM(I)=GM(I)-GD(L)*WT
121      135  CONTINUE
122           KICK=0
123           OLDK=0.
124           DO 150 I=1,M
125           GR=0.
126           DO 140 J=1,M
127      140  GR=GR+G(J)*B(J,I)
128           GP(I)=GR+GP(I)
129           GM(I)=GR+GM(I)
130           IF(GP(I)*GM(I).LT.0.) GO TO 150
131           TK=AMIN1(ABS(GP(I)),ABS(GM(I)))
132           IF(TK.GT.OLDK)KICK=I
133           IF(TK.GT.OLDK)OLDK=TK
134      150  CONTINUE
      C     PRINT LEFT AND RIGHT ERROR GRADIENTS.
135           WRITE(6,71) OLDK,(GP(I),I=1,M)
136           WRITE(6,71) OLDK,(GM(I),I=1,M)
137      71   FORMAT(1X,10E12.5)
138           IF(KICK.EQ.0) RETURN
139           DO 170 I=1,M
140      170  COL(I)=B(I,KICK)
141           RETURN
142           END

143           SUBROUTINE REBASE(ND,N,M,A,B,KICK,NEW)
      C SCALE THE COLUM B(I,KICK) TO HAVE UNIT PROJECTION ON ROW A(NEW,I)
      C REMOVE FROM OTHER COLUMNS B(I,J) THEIR PROJECTIONS ONTO A(NEW,I)
144           DIMENSION A(ND,M)
145           DIMENSION B(14,14),ROW(14),COL(14)
146           DO 10 J=1,M
147           ROW(J)=0.
148           DO 10 I=1,M
149      10   ROW(J)=ROW(J)+A(NEW,I)*B(I,J)
150           DO 20 I=1,M
151      20   COL(I)=B(I,KICK)/ROW(KICK)
152           DO 30 I=1,M
153           DO 30 J=1,M
154      30   B(I,J)=B(I,J)-COL(I)*ROW(J)
155           DO 40 I=1,M
156      40   B(I,KICK)=COL(I)
157           RETURN
158           END

159           SUBROUTINE SKEWER(ND,N,W,F,GU,GD,SMALL,K,T,ML,MH)
      C     SOLVE RANK 1 OVERDETERMINED EQUATIONS WITH SKEW NORM
      C     INPUTS- N,W,F,GU,GD,SMALL,K.  OUTPUTS- K,T,ML,MH.
      C FIND  T  TO MINIMIZE
      C                N
      C  LS  =     SUM    SKEWNORM(K,F(K)-W(K)*T)
      C                K=1
      C WHERE                      ( GU(K)*(ER-SMALL)  IF ER.GT.+SMALL      GU.GT.0
      C       SKEWNORM(K,ER)  =    ( GD(K)*(ER+SMALL)  IF ER.LT.-SMALL      GD.LT.0
      C                            (  0.               IF ABS(ER).LE.SMALL.GE.0.
      C     GU,GD,W,AND F ARE REFERENCED INDIRECTLY AS W(K(I)),I=1,N  ETC
      C     MINIMA WILL BE AT EQUATIONS K(ML),K(ML+1),...K(MH).
160           DIMENSION W(ND),F(ND),K(ND),GU(ND),GD(ND)
161           DIMENSION G(1000)
```

```
162           LOW=1
163           LARGE=N
164           ML=N
165           MH=1
166           GN=0.
167           GP=0.
168           DO 50 ITRY=1,N
169           L=K(LOW+MOD((LARGE-LOW)/3+ITRY,LARGE-LOW+1))
170           IF(ABS(W(L)).EQ.0.) GO TO 50
171           T=F(L)/(W(L))
172           F(L)=W(L)*T
173           DO 10 I=LOW,LARGE
174           L=K(I)
175           ER=F(L)-W(L)*T
176           G(L)=0.
177           IF(ER.GT.SMALL) G(L)=-W(L)*GU(L)
178    10     IF(ER.LT.-SMALL)G(L)=-W(L)*GD(L)
179           CALL SPLIT(LOW,LARGE,K,G,MLT,MHT)
180           GNT=GN
181           DO 20 I=LOW,MLT
182    20     GNT=GNT+G(K(I))
183           GPT=GP
184           DO 30 I=MHT,LARGE
185    30     GPT=GPT+G(K(I))
186           GPLX=0.
187           GMIX=0.
188           DO 40 I=MLT,MHT
189           L=K(I)
190           IF(W(L).LT.0.)GPLX=GPLX-W(L)*GU(L)
191           IF(W(L).GT.0.)GPLX=GPLX-W(L)*GD(L)
192           IF(W(L).GT.0.)GMIX=GMIX-W(L)*GU(L)
193    40     IF(W(L).LT.0.)GMIX=GMIX-W(L)*GD(L)
194           GRAD=GNT+GPT
195           IF((GRAD+GPLX)*(GRAD+GMIX).LT.0.) GO TO 60
196           IF(GRAD.GE.0.)LOW=MHT+1
197           IF(GRAD.LE.0.)LARGE=MLT-1
198           IF(LOW.GT.LARGE) GO TO 60
199           IF(GRAD.GE.0.)GN=GNT+GMIX
200           IF(GRAD.LE.0.)GP=GPT+GPLX
201           IF((GRAD+GPLX).EQ.0.)ML=MLT
202           IF((GRAD+GMIX).EQ.0.)MH=MHT
203    50     CONTINUE
204    60     ML=MINO(ML,MLT)
205           MH=MAXO(MH,MHT)
206           RETURN
207           END

208           SUBROUTINE SPLIT(LOW,LARGE,K,G,ML,MH)
      C       GIVEN G(K(I)),I=LOW,LARGE
      C       THEN REARRANGE K(I),I=LOW,LARGE AND FIND ML,MH SO THAT
      C       (G(K(I)),I=LOW,(ML-1)) .LT. 0 AND
      C       (G(K(I)),I=ML,MH)=0.   AND
      C       (G(K(I)),I=(MH+1),LARGE) .GT. 0.
209           DIMENSION K(LARGE),G(41)
210           ML=LOW
211           MH=LARGE
212    10     ML=ML-1
213    20     ML=ML+1
214           IF(G(K(ML)))20,30,30
215    30     MH=MH+1
```

```
 216     40    MH=MH-1
 217           IF(G(K(MH)))50,50,40
 218     50    KEEP=K(MH)
 219           K(MH)=K(ML)
 220           K(ML)=KEEP
 221           IF(G(K(ML)).NE.G(K(MH)))GO TO 10
 222           DO 60 I=ML,MH
 223           II=I
 224           IF(G(K(I)).NE.0.0) GO TO 70
 225     60    CONTINUE
 226           RETURN
 227     70    KEEP=K(MH)
 228           K(MH)=K(II)
 229           K(II)=KEEP
 230           GO TO 30
 231           END

 232           SUBROUTINE SPLOT(N,Y,D)
       C       THIS IS JUST A PRINTER PLOT SUBROUTINE.
 233           DIMENSION LINE(130),Y(N),D(N)
 234           DATA IBLANK,IX,IO/'    ','XXXX','====='/
 235           LA=121
 236           LA=75
 237           B=0.
 238           DO 10 I=1,N
 239           IF(ABS(Y(I)).GT.B) B=ABS(Y(I))
 240     10    IF(ABS(D(I)).GT.B) B=ABS(D(I))
 241           ALF=(LA+2)*.5
 242           BET=(LA-1)*.5/B
 243           DO 40 IT=1,N
 244           ID=ALF+BET*D(IT)
 245           IY=ALF+BET*Y(IT)
 246           I1=MIN0(ID,IY)
 247           I2=MAX0(ID,IY)
 248           DO 20 I=1,LA
 249     20    LINE(I)=IBLANK
 250           DO 30 I=I1,I2
 251     30    LINE(I)=IX
 252           SMALL =1.E-4
 253           IF(ABS(Y(IT)-D(IT)).LT.SMALL) LINE(I1)=IO
 254           WRITE(6,71) IT,(LINE(I),I=1,LA)
 255     71    FORMAT(I4,1X,129A1)
 256     40    CONTINUE
 257           RETURN
 258           END
       $END

       $ENTRY
 0.25471E 02 0.00000E 00 0.00000E 00 0.00000E 00 0.00000E 00 0.40000E 02
 0.40000E 02 0.00000E 00-0.40000E 02 0.20577E-04-0.13361E 02
 0.40000E 02 0.00000E 00-0.40000E 02 0.20577E-04-0.13361E 02
34  7
 0.72984E 00 0.00000E 00 0.11461E 01 0.00000E 00 0.00000E 00 0.13314E 02
 0.39919E 02 0.26864E 01 0.13136E 01 0.10770E 01-0.39919E 02
 0.39919E 02-0.13136E 01-0.26864E 01-0.22026E 01-0.39919E 02
  12  7 34
 0.19759E 00 0.00000E 00 0.11601E 01 0.00000E 00 0.31029E 00 0.83756E 01
 0.10517E 02 0.15624E 02-0.10517E 02-0.82177E 01 0.28931E 01
 0.10517E 02 0.76243E 01-0.14517E 02-0.12963E 02-0.11069E 01
4 37
```

```
-0.59364E-01 0.00000E 00 0.12261E 01 0.00000E 00 0.26529E 00 0.80000E 01
  00000E 00 0.80000E 01 0.28610E-05 0.14653E 01 0.57220E-05
 v.00000E 00-0.47684E-05-0.40000E 01-0.53518E 01-0.40000E 01
  1     XX
  2     XX
  3     X
  4     =
  5    XX
  6   XXX
  7   XXX
  8  XXXX
  9  XXXX
 10  XXXX
 11   XXX
 12     =
 13     XXX
 14     XXXXX
 15     XXXXXXXXX
 16     XXXXXXXXXXX
 17     XXXXXXXXXXXXXXX
 18     XXXXXXXXXXXXXXXXXXXX
 19     XXXXXXXXXXXXXXXXXXXXXXX
 20     XXXXXXXXXXXXXXXXXXXXXXXXXXX
 21                                                XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
 22                                                   XXXXXXXXXXXXXXXXXXXXXXXXX
 23                                                      XXXXXXXXXXXXXXXXXXXXX
-24                                                         XXXXXXXXXXXXXXXXX
                                                               XXXXXXXXXXXXX
 26                                                               XXXXXXXXX
 27                                                                   XXXXX
 28                                                                     XXX
 29                                                                       =
 30                                                                      XXX
 31                                                                      XXXX
 32                                                                      XXXX
 33                                                                      XXXX
 34                                                                      XXX
 35                                                                      XXX
 36                                                                      XX
 37                                                                      =
 38                                                                      X
 39                                                                     XX
 40                                                                     XX

CORE USAGE       OBJECT CODE=    13800 BYTES,ARRAY AREA=     9252 BYTES,TOTAL AI

COMPILE TIME=      0.39 SEC,EXECUTION TIME=      0.43 SEC,  WATFIV - VERSION 1 1
```