



Algorithm AS 132: Least Absolute Value Estimates for a Simple Linear Regression Problem

Ronald D. Armstrong; Mabel Tam Kung

Applied Statistics, Vol. 27, No. 3 (1978), 363-366.

Stable URL:

<http://links.jstor.org/sici?sici=0035-9254%281978%2927%3A3%3C363%3AAA1LAV%3E2.0.CO%3B2-R>

Applied Statistics is currently published by Royal Statistical Society.

Your use of the JSTOR archive indicates your acceptance of JSTOR's Terms and Conditions of Use, available at <http://www.jstor.org/about/terms.html>. JSTOR's Terms and Conditions of Use provides, in part, that unless you have obtained prior permission, you may not download an entire issue of a journal or multiple copies of articles, and you may use content in the JSTOR archive only for your personal, non-commercial use.

Please contact the publisher regarding any further use of this work. Publisher contact information may be obtained at <http://www.jstor.org/journals/rss.html>.

Each copy of any part of a JSTOR transmission must contain the same copyright notice that appears on the screen or printed page of such transmission.

JSTOR is an independent not-for-profit organization dedicated to creating and preserving a digital archive of scholarly journals. For more information regarding JSTOR, please contact support@jstor.org.

Algorithm AS 132

Least Absolute Value Estimates for a Simple Linear Regression Problem

By RONALD D. ARMSTRONG and MABEL TAM KUNG

University of Texas at Austin, Austin, Texas

Keywords: LINEAR PROGRAMMING; REGRESSION; LEAST ABSOLUTE VALUES; L1-NORM LANGUAGE

ISO Fortran

DESCRIPTION AND PURPOSE

Let (x_i, y_i) , $i = 1, 2, \dots, n$, be given. The least absolute curve fitting problem is to find (α, β) to

$$\text{minimize } \sum_{i=1}^n |y_i - \alpha - x_i \beta|. \tag{1}$$

It has been known for some time (see Charnes *et al.*, 1955) that (1) is equivalent to the following linear programming (LP) problem:

$$\text{minimize } \sum_{i=1}^n (P_i + N_i), \tag{2}$$

subject to $\alpha + x_i \beta + P_i - N_i = Y_i$, $P_i \geq 0$, $N_i \geq 0$, $i = 1, 2, \dots, n$, where P_i and N_i are, respectively, the positive and negative deviation associated with the i th observation.

Sadovski (1974) presents an algorithm to solve (1) based on a method proposed by Edgeworth (1888). The algorithm given here is a specialization of the LP procedure of Barrodale and Roberts (1973) for the more general linear curve fitting problem. This latter specialization has the following beneficial features:

- (a) Our study (see Time and Accuracy section) has indicated that the specialized LP approach is uniformly faster than the Sadovski code on all problem sizes.
- (b) No accumulative round-off error is present since all necessary values are recalculated from the original data at each iteration.
- (c) Considerably less storage is required when compared to the Barrodale and Roberts code and approximately the same amount when compared to the Sadovski code.
- (d) The convergent properties of LP theory are available while Sposito (1976) has demonstrated that the Sadovski algorithm need not converge in general.

STRUCTURE

SUBROUTINE SIMLP (N, X, Y, SAD, ALPHA, BETA, D, ITER, INEXT, IFAULT)

Formal parameters

<i>N</i>	Integer	input: number of observations
<i>X</i>	Real array (<i>N</i>)	input: the observed values (x_i) , $i = 1, 2, \dots, n$
<i>Y</i>	Real array (<i>N</i>)	input: the observed values (y_i) , $i = 1, 2, \dots, n$
<i>SAD</i>	Real	output: sum of the absolute deviations
<i>ALPHA</i>	Real	output: estimate of α
<i>BETA</i>	Real	output: estimate of β
<i>D</i>	Real array (<i>N</i>)	output: vector of the signed residuals
<i>ITER</i>	Integer	output: number of iterative steps

INEXT Integer array (N) working array
IFAILT Integer output: see Failure indications below

Failure indications

IFAILT = 1 if all x_i , $i = 1, 2, \dots, n$, values are equal
 = 0 otherwise

Restrictions

The local constants are *ACU* and *BIG* which have the values 10^{-6} and 10^{19} respectively. *ACU* is used to check if the rows in the initial basis are independent, and later in the algorithm to test for optimality. *BIG* is used as an initial value when determining the minimum ratio in the LP algorithm.

TIME AND ACCURACY

A computational study was carried out to compare *LONESL* (Sadovski, 1974) and *SIMLP*. Both codes provide least absolute value estimates for the simple linear regression model $y = \alpha + \beta x$. The University of Texas CDC 6600 was used in this study and sample results are given in Table 1. Times are in CPU milliseconds. Numerous sample problems were generated and it was clearly shown that the LP code runs considerably faster.

TABLE 1
A comparison between SIMLP and LONESL

Number of observations	<i>SIMLP</i>		<i>LONESL</i>	
	Time (CPU milliseconds)	Number of iterations	Time (CPU milliseconds)	Number of iterations
50	7	3	57	2
100	32	6	407	4
150	46	5	953	4
200	102	5	1647	4
250	84	2	3230	5
300	70	3	8373	9
350	262	3	6367	5
400	265	3	16296	10
450	212	5	8341	4
500	184	3	15374	6

The accuracy of this subroutine is dependent on the maximum real number accuracy of the machine used subject to the setting of *ACU* described above.

ACKNOWLEDGEMENT

This work was supported by NSF Grant MCS 77-00100.

REFERENCES

- BARRODALE, I. and ROBERTS, F. D. K. (1973). An improved algorithm for discrete L_1 linear approximation. *SIAM J. Numer. Anal.*, **10**, 833-848.
 CHARNES, A., COOPER, W. W. and FERGUSON, R. (1955). Optimal estimation of executive compensation by linear programming. *Management Sci.*, **2**, 138-151.
 EDGEWORTH, F. Y. (1888). On a new method of reducing observations relating to several quantities. *Phil. Mag.* (5th Ser.), **25**, 184-191.
 SADOVSKI, A. N. (1974). Algorithm AS 74. L_1 -norm fit of a straight line. *Appl. Statist.*, **23**, 244-248.
 SPOSITO, V. A. (1976). A remark on Algorithm AS 74. L_1 -norm fit of a straight line. *Appl. Statist.*, **25**, 96-97.

```

SUBROUTINE SIMPL(N, X, Y, SAD, ALPHA, BETA, D, ITER, INEXT,
* IFAULT)
C
C      ALGORITHM AS 132 APPL. STATIST. (1978) VOL.27, NO.3
C
C      SIMPL -  $Y = ALPHA + BETA * X + ERROR$ 
C
C      DIMENSION X(N), Y(N), D(N), INEXT(N)
C      DATA ACU, BIG /1.0E-6, 1.0E19/
C
C      INITIAL SETTINGS
C
C      IFAULT = 0
C      ITER = 0
C      AHALF = 0.5 + ACU
C      AONE = AHALF + AHALF
C
C      DETERMINE INITIAL BASIS
C
C      D(1) = 0.0
C      Y1 = Y(1)
C      IBAS1 = 1
C      A1 = X(1)
C      DO 10 I = 2, N
C      IF(ABS(A1 - X(I)) .LT. ACU) GOTO 10
C      A2 = X(I)
C      IBAS2 = I
C      Y2 = Y(I)
C      GOTO 20
10 CONTINUE
C      IFAULT = 1
C      RETURN
C
C      CALCULATE INITIAL BETA VALUE
C
C      20 DET = 1.0 / (A2 - A1)
C      AAAA = (A2 - Y1 - A1 * Y2) * DET
C      BBBB = (Y2 - Y1) * DET
C
C      CALCULATE INITIAL D VECTOR
C
C      DO 30 I = 2, N
C      DDD = Y(I) - (AAAA + BBBB * X(I))
C      D(I) = SIGN(1.0, DDD)
30 CONTINUE
C      TOT1 = 1.0
C      TOT2 = X(IBAS2)
C      D(IBAS2) = -1.0
C      DO 40 I = 2, N
C      TOT1 = TOT1 + D(I)
C      TOT2 = TOT2 + D(I) * X(I)
40 CONTINUE
C      T = (A2 * TOT1 - TOT2) * DET
C      IF(ABS(T) .LT. AONE) GOTO 50
C      DET = -DET
C      GOTO 70
C
C      MAIN ITERATIVE LOOP BEGINS
C
C      50 T = (TOT2 - A1 * TOT1) * DET
C      IF(ABS(T) .LT. AONE) GOTO 160
C      IFLAG = 2
C      ICUT = IBAS2
C      X(ICUT) = A1
C      AAA = A1
C      BBB = A2
C      GOTO 80
60 T = (TOT2 - A2 * TOT1) * DET
C      IF(ABS(T) .LT. AONE) GOTO 160
70 IFLAG = 1
C      BBB = A1
C      AAA = A2
C      ICUT = IBAS1
80 RHO = SIGN(1.0, T)
C      T = 0.5 * ABS(T)
C      DET = DET * RHO
C
C      PERFORM PARTIAL SORT OF RATIOS

```

APPLIED STATISTICS

```

INEXT(IBAS1) = IBAS2
RATIO = DIG
SUM = AHALF
DO 120 I = 1, N
DDD = (X(I) - AAA) * DET
IF(DDD * D(I) .LE. ACU) GOTO 120
TEST = (Y(I) - AAAA - BBBB * X(I)) / DDD
IF(TEGT .GE. RATIO) GOTO 120
J = IBAS1
SUM = SUM + ABS(DDD)
90 ISAVE = IABS(INEXT(J))
IF(TEGT .GE. D(ISAVE)) GOTO 110
IF(SUM .LT. T) GOTO 100
SUBT = ABS((X(ISAVE) - AAA) * DET)
IF(SUM - SUBT .LT. T) GOTO 100
SUM = SUM - SUBT
D(ISAVE) = ISIGN(1, INEXT(J))
INEXT(J) = INEXT(ISAVE)
GOTO 90
100 J = ISAVE
ISAVE = IABS(INEXT(J))
IF(TEGT .LT. D(ISAVE)) GOTO 100
110 INEXT(I) = INEXT(J)
INEXT(J) = ISIGN(1, INT(D(I)))
D(I) = TEST
IF(SUM .LT. T) GOTO 120
IIN = IABS(INEXT(IBAS1))
RATIO = D(IIN)
120 CONTINUE
C
C      UPDATE BASIC INDICATORS
C
IIN = IABS(INEXT(IBAS1))
J = IIN
130 ISAVE = IABS(INEXT(J))
IF(ISAVE .EQ. IBAS2) GOTO 140
ZZZ = ISIGN(1, INEXT(J))
TOT1 = TOT1 - ZZZ - ZZZ
TOT2 = TOT2 - 2.0 * ZZZ * X(ISAVE)
D(ISAVE) = -ZZZ
J = ISAVE
GOTO 130
140 ZZZ = ISIGN(1, INEXT(IBAS1))
TOT1 = TOT1 - RHO - ZZZ
TOT2 = TOT2 - RHO * BBB - ZZZ * X(IIN)
D(IOUT) = -RHO
ITER = ITER + 1
IF(IFLAG .EQ. 1) GOTO 150
X(IBAS2) = A1
IBAS2 = IIN
D(IBAS2) = -1.0
A2 = X(IIN)
Y2 = Y(IIN)
DET = 1.0 / (A1 - A2)
AAAA = (A1 * Y2 - A2 * Y1) * DET
BBBB = (Y1 - Y2) * DET
GOTO 160
150 IBAS1 = IIN
A1 = X(IIN)
D(IBAS1) = 0.0
Y1 = Y(IIN)
DET = 1.0 / (A2 - A1)
AAAA = (A2 * Y1 - A1 * Y2) * DET
BBBB = (Y2 - Y1) * DET
GOTO 30
C
C      CALCULATE OPTIMAL SUM OF ABSOLUTE DEVIATIONS
C
160 SAD = 0.0
DO 170 I = 1, N
D(I) = Y(I) - (AAAA + BBBB * X(I))
SAD = SAD + ABS(D(I))
170 CONTINUE
ALPHA = AAAA
BETA = BBBB
RETURN
END

```