

A SYSTEM OF SUBROUTINES FOR ITERATIVELY
REWEIGHTED LEAST SQUARES COMPUTATIONS

David Coleman*

Paul Holland**

Neil Kadent†

Virginia Klema*

Working Paper No. 189

National Bureau of Economic Research, Inc.
575 Technology Square
Cambridge, Massachusetts 02139

July 1977

This research was supported by the National Science Foundation under Grants DCR75-08802 and MCS76-11989.

*National Bureau of Economic Research, Inc., Computer Research Center,
575 Technology Square, Cambridge, Massachusetts 02139.

**Educational Testing Service, Rosedale Road, Princeton, N. J. 08540.

†Department of Computer Science, University of Toronto, Toronto,
Canada M5S 1 A7.

ABSTRACT

A description of a system of subroutines to compute solutions to the iteratively reweighted least squares problem is presented. The weights are determined from the data and linear fit and are computed as functions of the scaled residuals. Iteratively reweighted least squares is a part of robust statistics where "robustness" means relative insensitivity to moderate departures from assumptions. The software for iteratively reweighted least squares is cast as semi-portable Fortran code whose performance is unaffected (in the sense that performance will not be degraded) by the computer or operating-system environment in which it is used. An ℓ_1 start and an ℓ_2 start are provided. Eight weight functions, a numerical rank determination, convergence criterion, and a stem-and-leaf display are included.

Key Words and Phrases: least squares, data analysis, mathematical software, portability, linear algebra, curve fitting, robust estimation, weight functions.

CR Categories 5.14 and 5.5

TABLE OF CONTENTS

Introduction	1
Section 1	3
Section 2	6
Section 3	8
Acknowledgments.	26
References	27

Introduction

The purpose of this paper is to describe a system of Fortran subroutines written as modular mathematical software to solve the iteratively reweighted least squares problem. The software includes documentation for use and flow of control as comments in the subroutines. The specifications from which the software was written are contained in [12]. The collection of subroutines uses orthogonal factorizations by Householder transformations or the singular value decomposition from EISPACK II [7] to compute the ℓ_2 start and iterations for reweighted least squares. CL1 [2] computes the ℓ_1 start, an overdetermined solution in the ℓ_1 norm.

The computational tools that we provide include an interactive driver, eight weight functions, John Tukey's stem-and-leaf display [15, 9] the diagonal of the "hat" matrix [10] which is the projection matrix P_A effectively computed as $U\Sigma^+U^T$ from the singular value decomposition [7] or QQ^T from QR, the Householder transformations. Optionally, the ℓ_2 condition of the matrix, the weights, residuals, and the convergence criterion can be displayed. Two forms of equilibration are also provided [16].

The usual statistics information to display the number of observations, number of variables, maximum diagonal element of the "hat" matrix, condition number of the weighted data matrix, the maximum absolute value of the residuals, and the minimum weight is optionally available. There is an option to provide the (weighted) sum of squared residuals, and the sum of absolute residuals. Also available is the (weighted) R-squared statistic, the (weighted) standard error, and the (weighted) F statistic.

The software is presented in the form of a basis tape suitable for use by the Fortran converter [1] from IMSL to produce target Fortran code for CDC, Burroughs, Honeywell, PDP-10, and Univac machines. The source code is long precision IBM code acceptable to the Fortran converter. The PFORT verifier [14] was used to check the software.

We do not discuss the theoretical properties of iteratively reweighted least squares or the tuning constants for the weight functions in this paper. Rather we refer the reader to Holland and Welsch [11] for such information.

The organization of this paper is as follows. Section 1 defines the iteratively reweighted least squares problem. Section 2 describes the selection of rank for the data matrix and the re-weighted data matrix. Section 3 gives some numerical results. The weight functions are listed in Table 1 of Section 1. The subroutines for the computation are listed in Table 2 of Section 3. A copy of the subroutine to compute one of the weight functions, i.e., the Biweight weight function, is listed at the end of Section 3.

Section 1

The method of least squares has been the primary technique for fitting models to data for many years and is versatile and numerically stable when computationally stable methods are used [13]. Despite its central role in the past, much work has been done by statisticians to improve least squares in the sense of getting more information about the data than is available from just the least squares solution or from the matrix factorizations that are used to obtain it.

The area of work that our software addresses is robust regression which is aimed at analyzing and improving the behavior of least squares estimation when the disturbances are not well-behaved. We focus our attention on one of the computational procedures for robust linear regression, iteratively reweighted least squares.

Consider the model $b = Ax + r$ where b is an $m \times 1$ vector of observations, A is an $m \times n$ data or design matrix, x is an $n \times 1$ vector of parameters, and r is an $m \times 1$ vector. The notation $b = Ax + r$ corresponds to the statistical notation $y = X\beta + \epsilon$ where y is $n \times 1$, X is $n \times p$, β is $p \times 1$, and ϵ is $n \times 1$.

The ordinary least squares problem is $\min_x \sum_{i=1}^m ((r_i(x))/s)^2$ where r is a vector of residuals $b - Ax$, and s is a constant or fixed scale.

The weighted least squares problem is $\min_x \sum_{i=1}^m W_i ((r_i(x))/s)^2$ which is solved by using ordinary least squares with $W^{1/2}A$ and $W^{1/2}b$ where W is a diagonal matrix of weights that are functions of scaled residuals.

The iteratively reweighted least squares problem assumes a start

$\hat{x}^{(0)}$, which can be obtained from ℓ_2 , ordinary least squares, least absolute residuals, that is to say, the overdetermined solution in the ℓ_1 norm, previous iterations of iteratively reweighted least squares, or a start specified by the user. Given $\hat{x}^{(0)}$, the problem is iterated to obtain the least squares solution, $\hat{x}^{(k+1)} = (W^{(k+1)})^{1/2} A + (W^{(k+1)})^{1/2} b$ where the diagonal matrix W is computed as a function of scaled residuals. The residual scaling function we use is the maximum absolute deviation, i.e., the median of the absolute values of the non-zero residuals. The modularity of the software makes readily possible the inclusion of additional residual scaling functions such as the inner-quartile-range of the residuals. The software to compute the weights includes the eight weight functions listed in Table 1.

To test convergence of iteratively reweighted least squares we use the convergence criterion suggested by John Dennis [4]. After the k^{th} iteration, we compute a scale-independent measure of the gradient, $A^T r$, where r is the residuals $b - Ax$, which is

$$\left(\left(\left(W^{(k+1)} \right)^{1/2} A_j \right)^T \left(\left(W^{(k+1)} \right)^{1/2} r^{(k)} \right) \right) / \left\| \left(W^{(k+1)} \right)^{1/2} A_j \right\|_2 \left\| \left(W^{(k+1)} \right)^{1/2} r^{(k)} \right\|_2$$

where $\|\cdot\|$ is the Euclidean norm,

The problem of iteratively reweighted least squares is a problem in optimization in the sense that one is minimizing a function of scaled residuals,

Table 1

Weight functions (where u = scaled residual), range and default tuning constants.

<u>Name</u>	<u>$w(u)$</u>	<u>Range</u>	<u>Tuning Constant</u>
<u>ANDREWS</u>	$w_A(u) = \begin{cases} \sin(u/A)/(u/A) \\ 0 \end{cases}$	$ u \leq \pi A$ $ u > \pi A$	$A = 1.339$
<u>BIWEIGHT</u>	$w_B(u) = \begin{cases} [1 - (u/B)^2]^2 \\ 0 \end{cases}$	$ u \leq B$ $ u > B$	$B = 4.685$
<u>CAUCHY</u>	$w_C(u) = 1/(1 + (u/C)^2)$		$C = 2.385$
<u>FAIR</u>	$w_F(u) = 1/(1 + u/F)$		$F = 1.400$
<u>HUBER</u>	$w_H(u) = \begin{cases} 1 \\ H/ u \end{cases}$	$ u \leq H$ $ u > H$	$H = 1.345$
<u>LOGISTIC</u>	$w_L(u) = (\tanh(u/L))/(u/L)$		$L = 1.205$
<u>TALWAR</u>	$w_T(u) = \begin{cases} 1 \\ 0 \end{cases}$	$ u \leq T$ $ u > T$	$T = 2.795$
<u>WELSCH</u>	$w_R(u) = e^{-(u/R)^2}$		$R = 2.985$

Section 2

Starting points for the iterations include ℓ_2 , ordinary least squares and ℓ_1 , the overdetermined solution in the ℓ_1 norm [2] which corresponds to least-absolute-residuals regression. The ℓ_2 start and iterations subsequent to the ℓ_1 , ℓ_2 , or user-supplied start are computed by orthogonal factorizations, i.e., Householder transformations or a combination of Householder transformations and the singular value decomposition.

The way in which we decide to use the QR (Householder transformations) or a combination of QR and MINFIT [7] (least-squares solution by singular value decomposition) needs some explanation. Frequently the data matrix, A , in the statistical model $b = Ax + r$ has some variables (columns) that are close in the numerical sense to being linear combinations of other columns of A . Since such a situation may occur the numerical rank [8] of A must be determined before proceeding with the least-squares computation. The numerical rank should be determined by the user, and the determination of rank should be made with respect to the certainty of the data. Since the rank must be determined at every iteration (reweighting may down-weight rows, i.e., observations, to the extent that the effective deletion of observations creates rank degeneracy) it is necessary to estimate the condition of the weighted A as inexpensively as possible. For the ℓ_2 start and for all iterations after any start we default to a QR factorization with column pivoting. Unless A is exactly singular the completion of the QR factorization of A provides the upper triangular factor R whose condition is that of A . The condition estimate using R [3] is only $O(n^2)$ operations, gives a reliable measure of the ill-conditioning of A , and is used to determine whether QR is computationally sufficient or whether the computationally more expensive singular value decomposition, MINFIT, is necessary.

We strongly believe that the user should determine the rank of his data or design matrix by inspecting the singular values of A (which are the same as those of R). However we provide a conservative default determination of rank associated with the condition of the matrix at each iteration relative to the square-root of the precision of the computing machine that is used. Explicitly, the condition estimate of R as obtained by [3] is an estimate of the largest and smallest singular values, σ_{\max} and σ_{\min} , of A . If the ratio $(\sigma_{\min}/\sigma_{\max}) < \epsilon^{1/2}$ where ϵ is the relative precision of the arithmetic of the computing machine, the computation is continued by using the singular value decomposition. When the singular value decomposition is used the number, k , of singular values such that $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k > 0, \sigma_{k+1} = \dots = \sigma_n = 0$, is determined from the certainty of the data or from the square root of the precision of the computing machine, whichever is larger.

Section 3

The software to compute the solutions to the iteratively reweighted least squares problem consists of 17,500 lines of code, comments, and documentation for use. The 17,500 lines includes all of the software that is required for the interactive driver and its options for use plus a selection of test matrices. The software includes "help" commands to give on-line information to users. The interactive driver is designed to operate effectively on any computer system that permits the transmission of four characters to and from a terminal. The subroutines, however, can also be used in a batch environment. With the exception of CLL, all of the software is designed to be processed by the Fortran converter.

We have included a selection of test matrices including those from [5]. We chose this particular collection of matrices because of the widespread use of [5] as a reference and text. The matrices are cast in integer form and then assigned as floating point numbers to insure that there is uniform input to a variety of computing machines.

The name and a brief description of the subroutines that are needed for all options of the iteratively reweighted least squares problem are listed in Table 2 of this section.

Selected results from one of the weight functions, Biweight, and the terminal session used to compute the results applied to [6] follows Table 2.

The data matrix given in [6] is

$$A = \begin{bmatrix} 1 & .499 & 11.1 \\ 1 & .558 & 8.9 \\ 1 & .604 & 8.8 \\ 1 & .441 & 8.9 \\ 1 & .550 & 8.8 \\ 1 & .528 & 9.9 \\ 1 & .418 & 10.7 \\ 1 & .480 & 10.5 \\ 1 & .406 & 10.5 \\ 1 & .467 & 10.7 \end{bmatrix} \quad b = \begin{bmatrix} 11.14 \\ 12.74 \\ 13.13 \\ 11.51 \\ 12.38 \\ 12.60 \\ 11.13 \\ 11.70 \\ 11.02 \\ 11.41 \end{bmatrix}$$

and has singular values 31.6, .109, and .395.

Table 2

C	NAME	DESCRIPTION	HEA00370
C	****	*****	HEA00380
C			HEA00390
C	EQ01	MODIFIED ROW-INF-EQUILIBRATION	HEA00400
C	EQ02	COLUMN (MAX. ELEMENT) EQUILIBRATION	HEA00410
C	EQ03	ROW (MAX. ELEMENT) EQUILIBRATION	HEA00420
C	EQ04	COLUMN (SQRT. SUM OF SQUARES) EQUILIBRATION	HEA00430
C	EQ05	ROW (SQRT. SUM OF SQUARES) EQUILIBRATION	HEA00440
C	EUNORM	EUCLIDIAN (SQRT. SUM OF SQUARES) NORM	HEA00450
C	GETXL1	GET THE L1 START VECTOR	HEA00460
C	HMATSV	FORMS DIAG OF H-MAT=U*SIGMA*SIGMA-PSEUDO-INV*U-TRANS	HEA00470
C	HMATQR	FORMS DIAG OF H-MAT=Q*Q-TRANS	HEA00480
C	HMATRI	WRITES IERR PARAMETER ON DSET	HEA00490
C	IFLOOR	INTEGER FUNCTION FINDS FLOOR(REAL) \$\$	HEA00500
C	IRLSQR	SUBROUTINE TO DO ONE I R L S ITERATION (QRSOL)	HEA00510
C	IRLSSV	SUBROUTINE TO DO ONE I R L S ITERATION (MINSOL)	HEA00520
C	MATIO1	WRITES MATRIX ON DSET, OPTIONAL HEADING	HEA00530
C	MINFIT	SINGULAR VALUE DECOMPOSITION A=U*SIGMA*V-TRANSP	HEA00540
C	MINSOL	SOLVES AX=B GIVEN OUTPUT FROM MINFIT	HEA00550
C	MSG1	WRITES VARIABLE-LENGTH MESSAGE ON DSET	HEA00560
C	PERMUT	MATRIX COLUMN PERKUTATION	HEA00570
C	QR1	QR DECOMPOSITION, Q ORTHOGONAL TRANSFORMATIONS	HEA00580
C	QR1SOL	SOLVES AX=B USING QR1	HEA00590
C	RESTOR	RESTORE MATRIX WEIGHTS	HEA00600
C	RESIDL	COMPUTES RESIDUAL B-AX	HEA00610
C	SCLMAD	SCALE RESIDUALS BY SCALING FACTOR	HEA00620
C	SLDSPY	DOES STEM AND LEAF DISPLAY (CALLS OTHERS) \$\$	HEA00630
C	SLLEAF	DETERMINES STEMS AND LEAVES \$\$	HEA00640
C	SLPRNT	PRINTS STEM AND LEAF DISPLAY \$\$	HEA00650
C	SLSCAL	DETERMINES SCALE FACTOR AND UNIT FOR DISPLAY \$\$	HEA00660
C	SLSORT	SHELL SORT IN INCREASING ORDER \$\$	HEA00670
C	SMAD	DETERMINES MAD SCALING FACTOR	HEA00680
C	START0	USER START FOR I R L S	HEA00690
C	START1	L1 START (FROM CL1) FOR I R L S	HEA00700
C	START2	L2 START (FROM MINSOL) FOR I R L S	HEA00710
C	START3	L2 START (FROM QR1) FOR I R L S	HEA00720
C	SVMAX	ESTIMATES LARGEST S.V. OF UPPER TRIANGULAR MATRIX	HEA00730
C	SVMIN	ESTIMATES SMALLEST S.V. OF UPPER TRIANGULAR MATRIX	HEA00740
C	UNIF01	UNIFORM (0,1) RANDOM NUMBER GENERATOR FUNCTION	HEA00750
C	WANDPW	ANDREWS WEIGHTING FUNCTION \$	HEA00760
C	WBIWGT	BIWEIGHT (BISQUARE) WEIGHTING FUNCTION \$	HEA00770
C	WCAUCH	CAUCHY WEIGHTING FUNCTION \$	HEA00780
C	WELSCH	WELSCH WEIGHTING FUNCTION \$	HEA00790
C	WFAIR	FAIR WEIGHTING FUNCTION \$	HEA00800
C	WGRAD1	COMPUTES GRADIENT	HEA00810
C	WGRAD2	COMPUTES SCALE INDEPENDANT MEASURE OF GRADIENT	HEA00820
C	WHUBER	HUBER WEIGHTING FUNCTION \$	HEA00830
C	WLOGIS	LOGISTIC WEIGHTING FUNCTION \$	HEA00840
C	WTALWR	TALWAR (ZERO-ONE) WEIGHTING FUNCTION \$	HEA00850
C	WUSER	USER-DEFINED WEIGHTING FUNCTION \$	HEA00860
C			HEA00870
C			HEA00880
C			HEA00890
C	\$ -	WEIGHTING FUNCTIONS USED IN WEIGHTED LEAST SQUARES	HEA00900
C			HEA00910
C	\$\$ -	PART OF STEM AND LEAF	HEA00920
C			HEA00930

Table 2 con't.

C	NAME	DESCRIPTION	HEA00370
C	****	*****	HEA00380
C			HEA00390
C	CL1	DOES AN L1 START	HEA00400
C	HELP1	PRINTS HELP FOR I R L S	HEA00410
C	IRHELP	I R L S HELP COMMAND	HEA00420
C	IRLSDR	INTERACTIVE DRIVER FOR I R L S	HEA00430
C	IROPN	GETS I1 OPTION FOR I R L S	HEA00440
C	IROPNN	GETS I2 OPTION FOR I R L S	HEA00450
C	IROPFR	GETS REAL OPTION FOR I R L S	HEA00460
C	IRPRNT	I R L S PRINT COMMAND	HEA00470
C	IRPROP	I R L S OPTIONS COMMAND	HEA00480
C	IRSTAT	I R L S STATISTICS COMMAND	HEA00490
C	IRSTLF	I R L S STEM&LEAF COMMAND	HEA00500
C	MAT0A	GETS DATA MATRIX FOR I R L S	HEA00510
C	MAT0B	GETS RHS VECTOR FOR I R L S	HEA00520
C	MAT01	$A(I,J) = J * \text{SQRT}(R)$	HEA00530
C	MAT02	$A(I,J) = J / D$	HEA00540
C	MAT03	$A(I,J) = (I * J) / D$	HEA00550
C	MAT04	$A(I,J) = 1 / (1 + \text{ABS}(J-I))$	HEA00560
C	MAT05	TLONGLEY DATA	HEA00570
C	MAT06	TLONGLEY RHS	HEA00580
C	MAT07	SLONGLEY DATA	HEA00590
C	MAT08	LONGLEY RHS	HEA00600
C	MAT09	LONGLEY DATA	HEA00610
C	MAT10	UPPER TRIANGULAR A(N) MATRIX	HEA00620
C	MAT11	BAUER RHS	HEA00630
C	MAT12	BAUERTI DATA	HEA00640
C	MAT13	BAUER DATA	HEA00650
C	MAT14	IDENTITY MATRIX, CONSTANT ROWS OR COLS APPENDED	HEA00660
C	MAT15	$(I-2/N) * \text{SIGMA} * (I-2/N)$ MATRIX	HEA00670
C	OTHER MATXX ROUTINES (MAT16 - MAT47) INCLUDE DRAPER&SMITH PROBLEMS		HEA00680
C			HEA00690
C			HEA00700
C			HEA00710

Sample Terminal Session with some Numerical Results

load irlsdr (start

EXECUTION BEGINS...

I R L S ENVIRONMENT INITIALIZED
FOR LISTING OF COMMANDS TYPE HELP

SPECIAL OPTION NUMBERS:

9 (I1) OR 99 (I2) PRINTS HELP
8 (I1) OR 98 (I2) KEEPS OLD OPTION

TYPE OPTIONS ON A NEW LINE

IRLS COMMAND (A4):

>ihma#1#conv#1#iwst#01#prco#1#prco#2#prco#3#star#1

OPTION NUMBER? (I1): (9 GETS HELP)

IRLS COMMAND (A4):

OPTION NUMBER? (I1): (9 GETS HELP)

IRLS COMMAND (A4):

OPTION NUMBER? (I2): (99 GETS HELP)

IRLS COMMAND (A4):

OPTION NUMBER? (I1): (9 GETS HELP)

IRLS COMMAND (A4):

OPTION NUMBER? (I1): (9 GETS HELP)

IRLS COMMAND (A4):

OPTION NUMBER? (I1): (9 GETS HELP)

IRLS COMMAND (A4):

OPTION NUMBER? (I1): (9 GETS HELP)

IERR = 1 OUTPUT FROM I.1 START

IRLS COMMAND (A4):

>iter

** ITERATION 1 DONE

IRLS COMMAND (A4):

>prco#0#prin

OPTION NUMBER? (I1): (9 GETS HELP)

IRLS COMMAND (A4):

AFTER ITERATION 1 X =

0.8992867D+01 0.9319223D+01 -0.1716523D+00

PREVIOUS X =

0.9083704D+01 0.9189189D+01 -0.1709062D+00

I	RESIDUAL(I)	WDIAB(I)	HDIAB(I)
1	-0.5978190D+00	0.5546860D+00	0.1894029D+00
2	0.7471191D-01	0.9972362D+00	0.2545601D+00
3	0.1886233D-01	0.1000000D+01	0.4461569D+00
4	-0.6493913D-01	0.9876936D+00	0.4807548D+00
5	-0.2278996D+00	0.9282057D+00	0.2310283D+00
6	0.3859408D+00	0.8584280D+00	0.1414427D+00
7	0.7837715D-01	0.9987181D+00	0.2807317D+00
8	0.3625488D-01	0.1000000D+01	0.2111730D+00
9	0.4587736D-01	0.1000000D+01	0.3210570D+00
10	-0.9826476D-01	0.9792732D+00	0.2436725D+00

GRADIENT (CONVERGENCE LEVEL) =

0.3589985D+00 0.2994557D+00 0.3893084D+00

IRLS COMMAND (A4):

>stop@n1

OPTION NUMBER? (I1): (9 GETS HELP)

=====
RESIDUALS
=====

```

1          LO I    -0.5978

          ( UNIT =  0.1000D-01 )

```

```

2          -2 I 2
2          -1. I
2          -1 I
4          -0. I 96
4          -0 I
3          0 I 134
3          0. I 77

```

```

1          HI I    0.3859

```

```

IERR =      0 FOR RESIDUALS
IRLS COMMAND (A4):
>stem#2
OPTION NUMBER? (11): (9 GETS HELP)

```

```

=====
W-MATRIX
=====

```

STEM-AND-LEAF DISPLAY, N = 10

```

1          LO I    0.5547

          ( UNIT =  0.1000D-01 )

```

```

2          F I 5
2          S I
2          8. I
2          9 I
3          T I 2
3          F I
4          S I 7
3          9. I 899
3          10 1 000

```

```

IERR =      0 FOR DIAGONAL ELEMENTS OF W-MATRIX
IRLS COMMAND (A4):
>stem#3
OPTION NUMBER? (11): (9 GETS HELP)

```

```

=====
H-MATRIX
=====

```

STEM-AND-LEAF DISPLAY, N = 10

(UNIT = 0.1000D-01)

```

1          1 I 4

```



```
2      1. I 8
5      2 I 134
5      2. I 58
3      3 I 2
2      3. I
2      4 I 4
```

```
1      HI I      0.6808
```

IERR = 0 FOR DIAGONAL ELEMENTS OF H-MATRIX

IRLS COMMAND (A4):

>step#09*iter

OPTION NUMBER? (12): (99 GETS HELP)

IRLS COMMAND (A4):

```
** ITERATION 2 DONE
** ITERATION 3 DONE
** ITERATION 4 DONE
** ITERATION 5 DONE
** ITERATION 6 DONE
** ITERATION 7 DONE
** ITERATION 8 DONE
** ITERATION 9 DONE
** ITERATION 10 DONE
```

AFTER ITERATION 10 X =

```
0.9483807D+01 0.8967400D+01 -0.2055357D+00
```

PREVIOUS X =

```
0.9480502D+01 0.8969719D+01 -0.2053066D+00
```

I	RESIDUAL(I)	WDIAG(I)	HDIAG(I)
1	-0.5370930D+00	0.6779665D+00	0.2601124D+00
2	0.8165179D-01	0.9925734D+00	0.2529899D+00
3	0.3859782D-01	0.9983490D+00	0.4461462D+00
4	-0.9916240D-01	0.9890925D+00	0.6597202D+00
5	-0.2271626D+00	0.9424843D+00	0.2377027D+00
6	0.4162095D+00	0.8070726D+00	0.1199851D+00
7	0.9705210D-01	0.9895256D+00	0.2720100D+00
8	0.6996615D-01	0.9945766D+00	0.1957716D+00
9	0.5355376D-01	0.9968081D+00	0.3197178D+00
10	-0.6235050D-01	0.9956344D+00	0.2355411D+00

GRADIENT (CONVERGENCE LEVEL) =

```
-0.8818888D-03 -0.7873092D-03 -0.1026208D-02
```

```
IRLS COMMAND (A4):
>maxi#20#step#10#iter
OPTION NUMBER? (I2): (99 GETS HELP)
IRLS COMMAND (A4):
OPTION NUMBER? (I2): (99 GETS HELP)
IRLS COMMAND (A4):
** ITERATION 11 DONE
** ITERATION 12 DONE
** ITERATION 13 DONE
** ITERATION 14 DONE
** ITERATION 15 DONE
** ITERATION 16 DONE
** ITERATION 17 DONE
** ITERATION 18 DONE
** ITERATION 19 DONE
** ITERATION 20 DONE
AFTER ITERATION 20 X =
 0.9488481D+01 0.8964120D+01 -0.2058597D+00
PREVIOUS X =
 0.9488465D+01 0.8964131D+01 -0.2058586D+00
```

I	RESIDUAL(I)	WDIAG(I)	HDIAG(I)
1	-0.5365333D+00	0.6791081D+00	0.2607970D+00
2	0.8169217D-01	0.9925609D+00	0.2530100D+00
3	0.3875669D-01	0.9983257D+00	0.4461520D+00
4	-0.9950583D-01	0.9889630D+00	0.6594680D+00
5	-0.2271808D+00	0.9424685D+00	0.2377160D+00
6	0.4164755D+00	0.8066522D+00	0.1196153D+00
7	0.9721647D-01	0.9894649D+00	0.2719496D+00
8	0.7026910D-01	0.9944960D+00	0.1956213D+00
9	0.5361395D-01	0.9967958D+00	0.3197342D+00
10	-0.6202540D-01	0.9957114D+00	0.2357365D+00

```
GRADIENT (CONVERGENCE LEVEL) =
-0.4263579D-05 -0.3806967D-05 -0.4960356D-05
```

```
IRLS COMMAND (A4):
>stem#1
OPTION NUMBER? (I1): (9 GETS HELP)
```

```
=====
RESIDUALS
=====
```

STEM-AND-LEAF DISPLAY, N = 10

1 LO I -0.5365

(UNIT = 0.1000D-01)

```
2 -2 I 2
2 -1. I
2 -1 I
4 -0. I 96
4 -0 I
5 0 I 3
5 0. I 5789
```

1 HI I 0.4165

IERR = 0 FOR RESIDUALS
IRLS COMMAND (A4):
>stem#2
OPTION NUMBER? (I1): (9 GETS HELP)

=====
W-MATRIX
=====

STEM-AND-LEAF DISPLAY, N = 10

2 LO I 0.6791 0.8067 :

(UNIT = 0.1000D-02)

3 94 I 2
3 95 I
3 96 I
3 97 I
5 98 I 89
5 99 I 24568

IERR = 0 FOR DIAGONAL ELEMENTS OF W-MATRIX
IRLS COMMAND (A4):

>stem#3
OPTION NUMBER? (I1): (9 GETS HELP)

=====
H-MATRIX
=====

STEM-AND-LEAF DISPLAY, N = 10

(UNIT = 0.1000D-01)

1 1 I 1
2 1. I 9
4 2 I 33
3 2. I 567
3 3 I 1

2 HI I 0.4462 0.6595

IERR = 0 FOR DIAGONAL ELEMENTS OF H-MATRIX
IRLS COMMAND (A4):

>step#01#arc#0#star#3
OPTION NUMBER? (I2): (99 GETS HELP)
IRLS COMMAND (A4):
OPTION NUMBER? (I1): (9 GETS HELP)
IRLS COMMAND (A4):
OPTION NUMBER? (I1): (9 GETS HELP)

IERR = 0 FROM QR START AND RANK TEST
IRLS COMMAND (A4):

>iter

** ITERATION 1 DONE

IRLS COMMAND (A4):

>prco#0#prin

OPTION NUMBER? (I1): (9 GETS HELP)

IRLS COMMAND (A4):

AFTER ITERATION 1 X =

0.9807929D+01 0.8728491D+01 -0.2274461D+00

PREVIOUS X =

0.1030152D+02 0.8494711D+01 -0.2663214D+00

I	RESIDUAL(I)	WDIAG(I)	HDIAG(I)
1	-0.4987939D+00	0.7268122D+00	0.2969445D+00
2	0.8584364D-01	0.9934683D+00	0.2609452D+00
3	0.5158846D-01	0.9976387D+00	0.4560911D+00
4	-0.1229230D+00	0.9611901D+00	0.6419458D+00
5	-0.2270730D+00	0.9134825D+00	0.2301443D+00
6	0.4351445D+00	0.7198397D+00	0.9643418D-01
7	0.1072354D+00	0.9775559D+00	0.2678986D+00
8	0.9057973D-01	0.9809222D+00	0.1892309D+00
9	0.5648804D-01	0.9939678D+00	0.3245671D+00
10	-0.4046067D-01	0.9998900D+00	0.2357983D+00

GRADIENT (CONVERGENCE LEVEL) =

0.3861253D-01 0.2554913D-01 0.6280523D-01

IRLS COMMAND (A4):

>stem#1

OPTION NUMBER? (I1): (9 GETS HELP)

=====
RESIDUALS
=====

STEM-AND-LEAF DISPLAY, N = 10

1 LO I -0.4989

(UNIT = 0.1000D-01)

```

2      -2 I 2
2      -1. I
3      -1 I 2
3      -0. I
4      -0 I 4
4      0 I
4      0. I 5589
2      1 I 0

```

1 HI I 0.4351

IERR = 0 FOR RESIDUALS

IRLS COMMAND (A4):

>stem#2

OPTION NUMBER? (I1): (9 GETS HELP)

=====
W-MATRIX
=====

STEM-AND-LEAF DISPLAY, N = 10

2 LO I 0.7198 0.7268

(UNIT = 0.1000D-02)

3 91 I 3
3 92 I
3 93 I
3 94 I
3 95 I
4 96 I 1
5 97 I 7
5 98 I 0
4 99 I 3379

IERR = 0 FOR DIAGONAL ELEMENTS OF W-MATRIX
IRLS COMMAND (A4):

>stem53

OPTION NUMBER? (11): (9 GETS HELP)

=====
H-MATRIX
=====

STEM-AND-LEAF DISPLAY, N = 10

(UNIT = 0.1000D-01)

1 0. I 9
1 1 I
2 1. I 8
4 2 I 33
3 2. I 669
3 3 I 2
2 3. I
2 4 I
2 4. I 5

1 HI I 0.6419

IERR = 0 FOR DIAGONAL ELEMENTS OF H-MATRIX

IRLS COMMAND (A4):

>step#09#iter

OPTION NUMBER? (I2): (99 GETS HELP)

IRLS COMMAND (A4):

** ITERATION 2 DONE
** ITERATION 3 DONE
** ITERATION 4 DONE
** ITERATION 5 DONE
** ITERATION 6 DONE
** ITERATION 7 DONE
** ITERATION 8 DONE
** ITERATION 9 DONE
** ITERATION 10 DONE

AFTER ITERATION 10 X =

0.8800965D+01 0.9419934D+01 -0.1570752D+00

PREVIOUS X =

0.8827831D+01 0.9401409D+01 -0.1589556D+00

I	RESIDUAL(I)	WDIAG(I)	HDIAG(I)
1	-0.6179779D+00	0.4748387D+00	0.1482804D+00
2	0.8068059D-01	0.9909421D+00	0.2533161D+00
3	0.2165610D-01	0.9992975D+00	0.4526957D+00
4	-0.4718711D-01	0.9966559D+00	0.6990606D+00
5	-0.2196675D+00	0.9331172D+00	0.2343576D+00
6	0.3803538D+00	0.7981005D+00	0.1286704D+00
7	0.7220671D-01	0.9925810D+00	0.2815455D+00
8	0.2675576D-01	0.9908735D+00	0.2227332D+00
9	0.4383087D-01	0.9772916D+00	0.3200089D+00
10	-0.1093701D+00	0.9840117D+00	0.2593112D+00

GRADIENT (CONVERGENCE LEVEL) =

0.7703324D-02 0.6769983D-02 0.9167030D-02

```
IRLS COMMAND (A4):
>step#10#iter
OPTION NUMBER? (I2): (99 GETS HELP)
IRLS COMMAND (A4):
** ITERATION 11 DONE
** ITERATION 12 DONE
** ITERATION 13 DONE
** ITERATION 14 DONE
** ITERATION 15 DONE
** ITERATION 16 DONE
** ITERATION 17 DONE
** ITERATION 18 DONE
** ITERATION 19 DONE
** ITERATION 20 DONE
AFTER ITERATION 20 X =
0.8720285D+01 0.9475467D+01 -0.1514232D+00
PREVIOUS X =
0.8722084D+01 0.9474230D+01 -0.1515493D+00
```

I	RESIDUAL(I)	WDIAG(I)	HDIAG(I)
1	-0.6277457D+00	0.4548247D+00	0.1365807D+00
2	0.8007077D-01	0.9911210D+00	0.2530704D+00
3	0.1905696D-01	0.9994942D+00	0.4525619D+00
4	-0.4129958D-01	0.9976236D+00	0.7029458D+00
5	-0.2192678D+00	0.9534335D+00	0.2343049D+00
6	0.3757580D+00	0.8044204D+00	0.1316020D+00
7	0.6919790D-01	0.9933580D+00	0.2827640D+00
8	0.2143430D-01	0.9993569D+00	0.2252873D+00
9	0.4261887D-01	0.9974822D+00	0.3200544D+00
10	-0.1151000D+00	0.9816998D+00	0.2608286D+00

```
GRADIENT (CONVERGENCE LEVEL) =
0.5199742D-03 0.4561387D-03 0.6202752D-03
```

```
IRLS COMMAND (A4):
>stem#1
OPTION NUMBER? (I1): (9 GETS HELP)
```

```
=====
RESIDUALS
=====
```

STEM-AND-LEAF DISPLAY: N = 10

1 LO I -0.6277

(UNIT = 0.1000D-01)

```
2 -2 I 1
2 -1. I
3 -1 I 1
3 -0. I
4 -0 I 4
3 0 I 124
3 0. I 68
```

1 HI I 0.3758

IERR = 0 FOR RESIDUALS

IRLS COMMAND (A4):

>stem#2

OPTION NUMBER? (I1): (9 GETS HELP)

=====
W-MATRIX
=====

STEM-AND-LEAF DISPLAY, N = 10

2 LO I 0.4548 0.8044

(UNIT = 0.1000D-02)

3 93 I 3
3 94 I
3 95 I
3 96 I
3 97 I
4 98 I 1
6 99 I 137799

IERR = 0 FOR DIAGONAL ELEMENTS OF W-MATRIX

IRLS COMMAND (A4):

>stem#3

OPTION NUMBER? (I1): (9 GETS HELP)

=====
H-MATRIX
=====

STEM-AND-LEAF DISPLAY, N = 10

(UNIT = 0.1000D-01)

2 1 I 33
2 1. I
4 2 I 23
3 2. I 568
3 3 I 2
2 3. I
2 4 I
2 4. I 5

1 HI I 0.7029

IERR = 0 FOR DIAGONAL ELEMENTS OF H-MATRIX

In conclusion we show the listing of one subroutine - the Biweight weight function. The software on tape for the iteratively reweighted least squares problem is available from the Algorithms Distribution Service. The authors of this paper are responsible for any modifications that subsequent use may show to be necessary.

```

SUBROUTINE WBIWGT(N,U,CONST,SQW)
C *****PARAMETERS:
    INTEGER N
    REAL*8 U(N),CONST,SQW(N)
C *****LOCAL VARIABLES:
    INTEGER I
    REAL*8 OFLIM,UFETA,U1,PROD
C *****FUNCTIONS:
    REAL*8 DABS
C
C
C
C
C
C
C *****PURPOSE:
C     THIS SUBROUTINE PRODUCES THE SQUARE ROOTS OF THE WEIGHTS
C     DETERMINED BY THE INPUT VECTOR U OF PREVIOUSLY COMPUTED
C     SCALED RESIDUALS AND THE BIWEIGHT (BISQUARE) WEIGHT FUNCTION.(1)
C *****PARAMETER DESCRIPTION:
C
C     ON INPUT:
C
C         N MUST BE SET TO THE NUMBER OF ELEMENTS IN THE VECTORS U AND
C         SQW;
C
C         U CONTAINS THE STANDARDIZED RESIDUALS FROM A PREVIOUS LINEAR
C         FIT. THAT IS,  $U(I) = R(I) / S$  WHERE  $R(I)$  IS THE I-TH
C         RESIDUAL FROM A LINEAR FIT,  $R(I) = Y(I) - YFITTED(I)$ ,
C         AND  $S = S(R)$  IS A RESIDUAL SCALING FNCTION (E.G. S COULD
C         BE THE OUTPUT OF THE FORTRAN SUBROUTINE SMAP).
C
C         CONST IS THE 'TUNING CONSTANT' FOR THE WEIGHT FUNCTION
C         W(U). CONST MUST BE POSITIVE (SEE APPLICATION
C         AND USAGE RESTRICTIONS).
C
C     ON OUTPUT:
C
C         SQW CONTAINS A VECTOR OF THE SQUARE ROOTS OF THE WEIGHTS
C         DETERMINED BY THE SCALED RESIDUALS AND THE WEIGHTING
C         FUNCTION.
C *****APPLICATION AND USAGE RESTRICTIONS:
C     THE ROOT-WEIGHTS ARE NEEDED FOR THE COMPUTATION OF THE
C     ITERATIVELY REWEIGHTED LEAST SQUARES ESTIMATES USING THE
C     FORTRAN SUBROUTINES MIFIT AND MINSOL. IN THIS COMPUTATION
C     SQW(I) MULTIPLIES THE CORRESPONDING ROWS OF THE X-MATRIX
C     AND THE Y-VECTOR. (1)
C
C     THE LARGER THE VALUE OF CONST, THE MORE NEARLY ALL THE VALUES
C     OF W(U) WILL EQUAL UNITY.
C
C     IF CONST IS TAKEN TO BE VERY SMALL IT IS POSSIBLE TO PRODUCE A
C     VECTOR OF ROOT-WEIGHTS ALL OF WHICH EQUAL OR NEARLY EQUAL
C     ZERO, AND THIS WILL BE USELESS AS INPUT TO THE WEIGHTED LEAST
C     SQUARES COMPUTATIONS.
C
C     IF A TUNING CONSTANT VALUE OF 4.305 IS USED, UNDER THE
C     ASSUMPTION OF GAUSSIAN ERRORS, THE RESULTING ESTIMATOR
C     WILL HAVE 95 PERCENT ASYMPTOTIC EFFICIENCY.

```

WBI00010
WBI00020
WBI00030
WBI00040
WBI00050
WBI00060
WBI00070
WBI00080
WBI00090
WBI00100
WBI00110
WBI00120
WBI00130
WBI00140
WBI00150
WBI00160
WBI00170
WBI00180
WBI00190
WBI00200
WBI00210
WBI00220
WBI00230
WBI00240
WBI00250
WBI00260
WBI00270
WBI00280
WBI00290
WBI00300
WBI00310
WBI00320
WBI00330
WBI00340
WBI00350
WBI00360
WBI00370
WBI00380
WBI00390
WBI00400
WBI00410
WBI00420
WBI00430
WBI00440
WBI00450
WBI00460
WBI00470
WBI00480
WBI00490
WBI00500
WBI00510
WBI00520
WBI00530
WBI00540
WBI00550
WBI00560
WBI00570
WBI00580
WBI00590
WBI00600
WBI00610
WBI00620


```

C# ELSE, 1 CARD
CC ***** DATA STATEMENT *****
C# DATA OFLIM /SINFP/
DATA OFLIM /Z7FFFFFFFFFFFFFFF/
C
C ::::::::::: UFETA IS THE SMALLEST POSITIVE FLOATING POINT NUMBER
C S.T. UFETA AND -UFETA CAN BOTH BE REPRESENTED.
C# IF (IBM) THEN
CC IBM 360/370: UFETA = 16.**-65 :::::::::::
C# ELSE IF (XEROX) THEN
CC XEROX: UFETA = 16.**-65 :::::::::::
C# ELSE IF (UNIVAC) THEN
CC UNIVAC: UFETA = 2.**-129 :::::::::::
C# ELSE IF (HIS) THEN
CC HONEYWELL: UFETA = (2.**-128) :::::::::::
C# ELSE IF (DEC) THEN
CC PDP 10: UFETA = 2.**-129 :::::::::::
C# ELSE IF (CDC) THEN
CC CONTROL DATA: UFETA = 2.**-975 :::::::::::
C# ELSE IF (BGH) THEN
CC BURROUGHS: UFETA = 8.**-51 :::::::::::
C# ELSE, 1 CARD
CC ***** DATA STATEMENT *****
C# DATA UFETA /SETA/
DATA UFETA /Z0010000000000000/
C
C *****BODY OF PROGRAM:
C IF (CONST .LE. 1.000) PROD = OFLIM * CONST
C IF (CONST .GT. 1.000) PROD = UFETA * CONST
C
C DO 100 I=1,N
C U1 = DABS( U(I) )
C IF (U1 .LE. CONST) GO TO 10
C ::::::::::: DABS(U(I)) .GT. CONST :::::::::::
C SQW(I) = 0.000
C GO TO 100
10 CONTINUE
C IF (CONST .GT. 1.000) GO TO 20
C IF (U1 .LE. PROD) GO TO 20
C ::::::::::: DIVISION WOULD OVERFLOW :::::::::::
C SQW(I) = 0.000
C GO TO 100
20 CONTINUE
C IF (CONST .LE. 1.000) GO TO 30
C IF (U1 .GE. PROD) GO TO 30
C ::::::::::: DIVISION WOULD UNDERFLOW :::::::::::
C SQW(I) = 1.000
C GO TO 100
30 CONTINUE
C ::::::::::: FUNCTION CAN BE COMPUTED NORMALLY :::::::::::
C U1 = U(I) / CONST
C SQW(I) = ((0.500 + U1) + 0.500) * ((0.500 - U1) + 0.500)
100 CONTINUE
C
C RETURN
C ::::::::::: LAST CARD OF WBIKOT :::::::::::
C ENB

```

R; T=0.00/0.60 08149105

>

WBI01290
 WBI01300
 WBI01310
 WBI01320
 WBI01330
 WBI01340
 WBI01350
 WBI01360
 WBI01370
 WBI01380
 WBI01390
 WBI01400
 WBI01410
 WBI01420
 WBI01430
 WBI01440
 WBI01450
 WBI01460
 WBI01470
 WBI01480
 WBI01490
 WBI01500
 WBI01510
 WBI01520
 WBI01530
 WBI01540
 WBI01550
 WBI01560
 WBI01570
 WBI01580
 WBI01590
 WBI01600
 WBI01610
 WBI01620
 WBI01630
 WBI01640
 WBI01650
 WBI01660
 WBI01670
 WBI01680
 WBI01690
 WBI01700
 WBI01710
 WBI01720
 WBI01730
 WBI01740
 WBI01750
 WBI01760
 WBI01770
 WBI01780
 WBI01790
 WBI01800
 WBI01810
 WBI01820
 WBI01830
 WBI01840
 WBI01850
 WBI01860

Acknowledgments

The authors are grateful to many people for their assistance in doing this work. Especially we thank John Dennis, Gene Golub, and Roy Welsch for their helpful suggestions. David Hoaglin and Stan Wasserman provided the software for the stem-and-leaf display which gives a useful summary of computational results. Richard Bartels made available the software for the λ_1 start. Alan Cline, Cleve Moler, G. W. Stewart, and J. H. Wilkinson shared with us their technique for estimating the condition number of a matrix. David Gay, with valuable advice from David Hoaglin on random number generators, wrote the code for the condition estimate.

Michael Sutherland and Richard Becker helped to check the software on CDC and Honeywell computers. Douglas Raynor ran several versions of the code on the PDP 10. T. J. Aird and Ed Battiste supplied the Fortran converter from IMSL.

Paul Velleman, after using the interactive driver from its own documentation, made valuable suggestions concerning documentation for use. Maurice Herlihy and Steve Peters independently used the software, checked parts of the program and documentation for use. Sandra Moriarty has provided technical assistance throughout the period in which this work was done.

References

1. Aird, T. J., "The Fortran Converter User's Guide," IMSL, 1975.
2. Bartels, R., and Conn, A., "Linearly Constrained Discrete ℓ_1 Problems," Johns Hopkins University, Technical Report #248, June 1976.
3. Cline, A., Moler, C., Stewart, G. W., and Wilkinson, J. H., "On an Estimate for the Condition Number of a Matrix," informal manuscript, 1977.
4. Dennis, J., private communication, June 1976.
5. Draper, N. R., and Smith, H., Applied Regression Analysis, John Wiley and Sons, Inc., 1966.
6. Draper, N. R., and Stoneman, D., "Residuals and Their Variance Patterns," *Technometrics*, 8, 1966, p. 695-699.
7. Garbow, B. S., Boyle, J. M., Dongarra, J. J., and Moler, C. B., Matrix Eigensystem Routines - EISPACK Guide Extension, Springer-Verlag, Lecture Notes in Computer Science, 51, 1977.
8. Golub, G., Klema, V., and Stewart, G. W., "Rank Degeneracy and Least Squares Problems," University of Maryland, TR-456, 1976, Stanford University, STAN-C5-76-559, 1976, National Bureau of Economic Research, Inc., Working Paper 165, 1977.
9. Hoaglin, D. C., and Wasserman, S., "Automating Stem-and-Leaf Displays," National Bureau of Economic Research, Inc., Working Paper 109, 1975.
10. Hoaglin, D. C., and Welsch, R. E., "The Hat Matrix in Regression and ANOVA," Harvard University, Department of Statistics, Memo. NS 341, December 1976.

11. Holland, P., and Welsch, R., "Robust Regression Using Iteratively Reweighted Least Squares," *Communications in Statistics: Theory and Methods*, 1977.
12. Kaden, N., and Klema, V., "Guidelines for Writing Semi-portable Fortran," National Bureau of Economic Research, Inc., Working Paper 130, 1976.
13. Lawson, C. L., and Hanson, R. J., Solving Least Squares Problems, Prentice-Hall, Inc., 1974.
14. Ryder, B. G., *The Fortran Verifier: User's Guide*, Computing Science Tech. Report 12, Bell Telephone Labs., 1975.
15. Tukey, J. W., Exploratory Data Analysis, Addison-Wesley, 1977.
16. Van Der Sluis, A., "Condition, Equilibration and Pivoting in Linear Algebraic Systems," *Number. Math.* 15 (1970).

Errata Sheet to Working Paper #189.

- Table of Contents: replace lines 2, 3, and 4 with
Section 1, Iteratively Reweighted Least Squares . . . 3
Section 2, Selecting the Rank of the Data Matrix. . . 6
Section 3, Some Numerical Results 8
- page 1, Introduction: paragraph 2, line 5: replace
"Optionally" with "Displaying".
paragraph 2, line 6, 7: replace "can be displayed" with
"is an option".
paragraph 3, line 1: "The usual statistical information -
the number"
paragraph 3, line 4: Insert a dash (-) between "weight"
and "is".
- page 2, paragraph 2, line 3: replace "Rather" with "For such
information".
paragraph 2, line 4: "and the references therein".
paragraph 3, line 3: remove the hyphen.
- page 3, insert subheading: "Iteratively Reweighted Least Squares"
line 2: "...years. It is..."
- page 4, line 5 from top: equation should read
$$= ((W^{(k+1)})^{1/2} A)^+ (W^{(k+1)})^{1/2} b .$$

line 4 from bottom: add a subscript 2, to $||\cdot||$.
Should read: $||\cdot||_2$.
Insert after last line: "The function that is being
minimized determines the formula for the weight function
used. In general, we minimize $\sum_{i=1}^m \rho(r_i(x)/s)$ so that the
weight function is given by a $W(u) = \rho'(u)$."
- page 5, line 2: Tuning Constant* (should have elevated asterisk).
line 6 from top: function is $1/(1+ |u/F|)$.

page 5,

insert as footnote:

*
These are default values for the tuning constants which are designed to have 95% asymptotic efficiency with respect to ordinary least squares when the disturbances from the normal or Gaussian distribution and a scaling function converge to the standard deviation of that disturbance distribution.

page 6,

insert subheading: "Selecting the Rank of the Data Matrix"

paragraph 2, line 6: insert a comma after "occur".

page 8,

insert subheading: "Some Numerical Results"

page 28,

reference 4: replace "private communication, June 1976" with "Non-linear Least Squares and Equations, The State of the Art in Numerical Analysis, Academic Press, 1977."