SIAM J. SCI. STAT. COMPUT.
Vol. 1, No. 2, June 1980

0196-5204/80/0102-0010 $1.00/0

LEAST ABSOLUTE DEVIATIONS CURVE-FITTING*

PETER BLOOMFIELD† AND WILLIAM STEIGER‡

Abstract. A method is proposed for least absolute deviations curve fitting. It may be used to obtain least absolute deviations fits of general linear regressions. As a special case it includes a minor variant of a method for fitting straight lines by least absolute deviations that was previously thought to possess no generalization. The method has been tested on a computer and was found on a range of problems to execute in as little as 1/3 the CPU time required by a published algorithm based on linear programming. More important, this advantage appears to increase indefinitely with the number of data points

Key words. least absolute deviations, linear programming

1. Introduction. The least absolute deviations method of curve-fitting consists of fitting the model

$$y_i = \sum_{j=1}^{k} x_{ij}\theta_j + \varepsilon_i, \qquad i = 1, \ldots, n \tag{1}$$

to data $(x_{i1}, \ldots, x_{ik}, y_i, i = 1, \ldots, n)$ by choosing the parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)$ to minimize the sum of absolute deviations,

$$S(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left| y_i - \sum_{j=1}^{k} x_{ij}\theta_j \right|. \tag{2}$$

According to Eisenhart (1961), the minimization of a quantity like (2) was suggested by Boscovitch in the mid-eighteenth century for fitting lines well before the introduction of the method of least squares. Boscovitch added the condition that the sum of the signed residuals be zero, which constrains the line to pass through the centroid of the $(\mathbf{x}_i, y_i)$ points. He also described an algorithm for finding the slope of the minimizing line, which can of course be used in conjunction with different constraints such as that of a zero intercept.

More than one hundred years later, Edgeworth dropped the constraint and proposed the fitting of lines and of more complicated models by unconstrained minimization of (2). However, the computations are inherently more complex than the solution of the linear equations that arise in the method of least squares, and Edgeworth's method does not seem to have been used widely.

A new method was introduced by Rhodes (1930) and discussed by Singleton (1940), who also proposed an alternative. The development of linear programming and the observation of Harris (1950) that the least absolute deviations fitting problem could be turned into a linear programming problem was the next major advance. This

*Received by the editors October 22, 1979, and in final revised form May 6, 1980.

†Department of Statistics, Princeton University, Princeton, New Jersey 08540. The work of this author was supported in part by the U.S. Energy Research and Development Administration, under Contract EY76-SO2-2310, awarded to the Department of Statistics, Princeton University.

‡Department of Computer Science, Rutgers University, New Brunswick, New Jersey 08903. The work of this author was supported in part by the Rutgers University Research Council. Computing facilities were provided by the Rutgers University PDP 20 devoted to computer science research.

line was pursued by Wagner (1959) and many others, including Barrodale and Roberts (1973, 1974) and Narula and Wellington (1977). We comment on these algorithms in §§4 and 5.

The current resurgence of interest in least absolute deviations methods is associated with the development of robust and resistant methods (see Huber, 1973 or Andrews, 1974). That a least absolute deviations fit is less sensitive to extreme errors than is a least squares fit was noted by Bowditch in an English translation of a work by Laplace (see Eisenhart, 1961). Similar remarks have been made by Edgeworth and by Rhodes (1930), who exhibited an example to support the point. While there are techniques that are at the same time statistically more efficient in reasonable circumstances and even less affected by extreme errors, the conceptual simplicity of least absolute deviations estimates and their competitive computational cost makes them well worth considering. We note that these estimates are maximum likelihood and hence asymptotically efficient in the (perhaps uncommon) situation when the errors follow the double exponential Laplace distribution.

In addition to their use for robust estimation of regression equations, least absolute deviations estimates could also be used as starting points for iterative estimation schemes. Although this would be computationally more expensive than the use of least squares estimates as starting points, the resistance of the procedure to outliers would be improved.

Schlossmacher (1973) described a different relationship between iterative and least absolute deviations methods. He pointed out that least absolute deviations estimates could be found by iteratively reweighted least squares. However, numerical tests have shown that this is a computationally expensive way to find an approximate solution to a problem that admits exact solution. Abdelmalek (1971) suggested a different approximate solution, namely minimization of the l_p-norm with p approaching 1 from above. This approach was found by Barrodale and Roberts (1973) to be inefficient.

2. Minimizing the sum of absolute deviations. When there is only one degree of freedom in the fit, such as when $k=1$ or when constraints are imposed as by Boscovitch, the solution is straightforward. For

$$\sum_{i=1}^{n} |y_i-\theta x_i| = \sum_{i=1}^{n} |x_i||y_i/x_i-\theta|,$$

and the minimizing value of θ is thus the weighted median of the ratios y_i/x_i, with respect to weights $|x_i|$. This weighted median may be defined as any value θ such that

$$\sum_{i:\, y_i/x_i=\theta} |x_i| \geqq \left| \sum_{i:\, y_i/x_i<\theta} |x_i| - \sum_{i:\, y_i/x_i>\theta} |x_i| \right|$$

It may be found by ordering the ratios, and then summing the weights from one end until the partial sum first exceeds or equals one half the total of the weights. The corresponding ratio is the weighted median.

This shows that when $k=1$, θ may always be taken to be one of the ratios y_i/x_i, and that at least one of the residuals $y_i-\theta x_i$ vanishes. In the general case there is a solution $\boldsymbol{\theta}$ for which at least r of the residuals vanish, r being the rank of $\mathbf{X}=(x_{ij})$. This is easily seen by a linear programming formulation of (2) (e.g. Wagner (1959)) or

via a simple direct argument: Suppose $0 \leqq m < r$ residuals, $y_i - \sum_{j=1}^{k} x_{ij}\theta_j$, vanish, say for $i = i_1, \ldots, i_m$. Since $m < r$, there is a row, say the p^{th}, not in the space spanned by rows $i_1, \ldots, i_m$, and a vector $\boldsymbol{\delta}$ orthogonal to rows $i_1, \ldots, i_m$ but not row p. Thus the function $f(t) = \sum_{i=1}^{n} |y_i - \sum_{j=1}^{k} x_{ij}(\theta_j + t\delta_j)|$ is a sum with zero terms when $i = i_1, \ldots, i_m$, and $S(\boldsymbol{\theta}))) = f(0)$. Finally, write $f(t) = \sum_{i=1}^{n} |w_i - tv_i|$, where $w_i = y_i - \sum_{j=1}^{k} x_{ij}\theta_j$ and $v_i = \sum_{j=1}^{k} x_{ij}\delta_j$. From the $k = 1$ case, f is minimized for $t = \hat{t} = y_q / x_q$, and the q^{th} term of the sum is zero. Now $S(\boldsymbol{\theta} + \hat{t}\boldsymbol{\delta})$ has $m+1$ zero residuals at $i = i_1, \ldots, i_m$ or $i = q$, and $S(\boldsymbol{\theta} + \hat{t}\boldsymbol{\delta}) = f(\hat{t}) \leqq f(0) = S(\boldsymbol{\theta})$, an argument that holds as long as $m < r$.

The problem of minimizing (2) is thus a discrete search problem. One need only search the $\binom{n}{r}$ combinations of r zero residuals, find an appropriate $\boldsymbol{\theta}$ for each, and evaluate $S(\boldsymbol{\theta})$.

This observation motivates a simple method for solving the two-parameter problem of fitting a straight line $y = \theta_1 + \theta_2 x$. The method, described by Rhodes (1930) and Karst (1958), is the basis of a computer algorithm published by Sadovski (1974). First, a line is fitted to the data while constrained to pass through some arbitrary point, such as the origin. As noted above, the fitted line must pass through at least one data point. Next, a line is fitted while constrained to pass through the data point thus identified (an arbitrary one in the case of multiplicity). This identifies a new point to replace the previous one, and the algorithm continues. It terminates when the fitted line does not change.

It is easily seen that the sum of absolute deviations goes down at each step, and that no more than $n-1$ steps can be taken before termination. At some stages the problem may be degenerate in the sense that more than two residuals are zero. Some care must be taken so the algorithm does not cycle endlessly (see Sposito, 1976), or terminate prematurely. Karst (1958) recognized the difficulties introduced by degeneracy. The optimal line through P_1 may pass through P_2 and vice versa, and yet still not be the overall optimum.

Narula and Wellington (1977) described a method based on linear programming (in which, surprisingly, they readopted Boscovitch's constraint that the fit should pass through the centroid). They commented that the Karst/Sadovski technique does not lend itself to regression problems with more parameters. There is, however, a very natural extension that not only leads to an efficient numerical solution of the problem, but also sheds light on the relationship of the least absolute deviations problem to linear programming. This extension, based on the foregoing argument and related to the descent method of Usow (1967), is described in the next section.

3. The method. Assume that $n > k$, and that $\mathbf{X}$ is of full rank. The basis of the method is to search for a set of k data points such that the fit, when constrained to make the corresponding residuals vanish, is optimal. As in the method described above for $k = 2$, this set of data points is found iteratively, by successive improvements. In each iteration one point from the current set is identified as a good prospect for deletion. This point is then replaced by the best alternative. This clearly generalizes the 2-parameter technique. It is also related to the descent technique of Usow (1967). The novel features of our method are

* an efficient method for finding optimal replacement, and
* a heuristic method for identifying the point to be deleted.

(i) *Replacement*. Suppose that the rows of $\mathbf{X}=(x_{ij})$ that correspond to the current set of points are $\mathbf{x}_1^T,\ldots,\mathbf{x}_k^T$, and that $\mathbf{x}_k^T$ has been identified for replacement. There is a one-dimensional set of parameter values $\boldsymbol{\theta}$ that satisfy the remaining constraints

$$y_i=\mathbf{x}_i^T\boldsymbol{\theta}, \qquad i=1,\ldots,k-1, \tag{3}$$

which we may parametrize as

$$\boldsymbol{\theta}=\boldsymbol{\theta}_0+t\boldsymbol{\delta},$$

where $\boldsymbol{\theta}_0$ is any arbitrary member of the set, and $\boldsymbol{\delta}$ satisfies

$$\mathbf{x}_i^T\boldsymbol{\delta}=0, \qquad i=1,\ldots,k-1. \tag{4}$$

Within this set, the optimum may be found by minimizing

$$\sum_{i=1}^{n}|y_i-\mathbf{x}_i^T(\boldsymbol{\theta}_0+t\boldsymbol{\delta})| \tag{5}$$

with respect to the scalar t. Rewriting this objective function as

$$\sum_{i=1}^{n}|(y_i-\mathbf{x}_i^T\boldsymbol{\theta}_0)-t(\mathbf{x}_i^T\boldsymbol{\delta})|,$$

one can minimize (5) by solving the one-dimensional problem of regressing $y_i-\mathbf{x}_i^T\boldsymbol{\theta}_0$ on $\mathbf{x}_i^T\boldsymbol{\delta}$. As was remarked earlier, this minimizing value of t is the weighted median of

$$(y_i-\mathbf{x}_i^T\boldsymbol{\theta}_0)/(\mathbf{x}_i^T\boldsymbol{\delta}), \qquad i=1,\ldots,n,$$

with respect to the weights

$$|\mathbf{x}_i^T\boldsymbol{\delta}|, \qquad i=1,\ldots,n.$$

Thus the minimizing value of t may be found efficiently by using a weighted version of the partial quick-sort procedure (see Chambers, (1971)). The next set of parameter values may then be computed as $\boldsymbol{\theta}_0+t\boldsymbol{\delta}$. Finally, the data point that replaces $\mathbf{x}_k$ is the point corresponding to the weighted median.

For $\boldsymbol{\theta}_0$, we use the parameter values associated with $\mathbf{x}_1,\ldots,\mathbf{x}_k$, for these satisfy (3) and the additional equation with $i=k$. The vector $\boldsymbol{\delta}$ is determined up to scalar multiples by (4).

(ii) *Deletion*. A reasonable goal when deleting a point at some intermediate stage would be to select that point which, when optimally replaced, leads to the largest reduction in the objective function. However, we have found no reliable way to identify this point short of trying all deletions. In tests, this "look ahead" approach was more expensive than the heuristic method described below, and often did not lead to fewer steps before termination.

Our heuristic method is based on gradients. The quantity (5) is a convex, piecewise linear function of t. We examine the larger of its left-hand derivative at 0 and the negative of its right-hand derivative, which may be expressed as

$$\left|\sum_{i:\,r_i<0}w_i-\sum_{i:\,r_i>0}w_i\right|-\sum_{i:\,r_i=0}w_i, \tag{6}$$

where

$$r_i=(y_i-\mathbf{x}_i^T\boldsymbol{\theta}_0)/\mathbf{x}_i^T\boldsymbol{\delta}$$

and

$$w_i = |\mathbf{x}_i^T \boldsymbol{\delta}|.$$

If (6) is negative, then the unique minimum of (5) is at $t=0$, and thus if $\mathbf{x}_k$ were deleted, the same point would immediately reenter (or possibly a different point, but still leading to no improvement in the objective function nor any change in the parameter values). If (6) is zero, $t=0$ is still a minimum, but there is an interval of values all of which also minimize (5). Thus the objective function still cannot be improved, although there are other sets of parameter values that give the same value.

We therefore avoid nonpositive values of (6). To convert the gradient into an estimate of the amount by which (5) may be reduced, we multiply by a rough estimate of the scale of the ratios r_i. Since the numerators are the same in all the possible cases at a given stage, we use the reciprocal of the sum of the denominators as this rough estimate. Thus we use the quantity

$$\rho = \frac{\left| \sum_{r_i<0} w_i - \sum_{r_i>0} w_i \right| - \sum_{r_i=0} w_i}{\sum w_i} \tag{7}$$

to measure the merit of deleting the given point, and we delete the point for which ρ is largest.

We note that $\max(\rho,0)$ has some of the properties of an absolute correlation coefficient between the residuals $y_i - \mathbf{x}_i^T \boldsymbol{\theta}_0$ and the linear compound $\mathbf{x}_i^T \boldsymbol{\delta}$:

* it lies between 0 and 1,
* it is 0 iff the (least absolute deviations) regression of the residuals on the compound is null,
* it is 1 iff the signs of the residuals and the compound are all the same or all opposite.

We make the calculation of (7) for each candidate for deletion economical by the choice of $\boldsymbol{\theta}_0$. For this is the same for each candidate, and hence the residuals need only be computed once. For $\boldsymbol{\delta}$ we use the appropriate column of the inverse of the $k \times k$ submatrix

$$\mathbf{Z} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_k^T \end{bmatrix}. \tag{8}$$

Thus, in evaluating the merit of deleting $\mathbf{x}_j$ from the current set, we compute (7) using the j^{th} column of $\mathbf{Z}^{-1}$ for $\boldsymbol{\delta}$. Clearly this choice satisfies the requirement embodied in (4).

In addition, the partitioning of the residuals into negative, zero, and positive values can be used as the first of the partitioning steps involved in the weighted median calculation. This reduces the series length on the average by almost one half, and leads to a further useful economy.

To start the iteration, any set of k independent rows of $\mathbf{X}$ may be chosen, with the appropriate $\boldsymbol{\theta}_0$. However the following procedure is usually more efficient. Take $\boldsymbol{\theta}=\mathbf{0}$, which corresponds to an empty set of data points. We then add variables in a stepwise fashion, until we have a fit $\boldsymbol{\theta}_0$ and a corresponding set of k data points. At each intermediate stage, the fit involves m variables, $0 \leqq m < k$, and a corresponding set of m data points with zero residuals. We then have the option of including another variable, thus increasing m to $m+1$, or improving the fit with m variables, using the principles described earlier. Both of these alternatives may be carried out by regressing the current residuals on some appropriate linear compound of the variables in the problem.

Suppose variables $v_1, \ldots, v_m$ are in the model corresponding to points $i_1, \ldots, i_m$. Thus the current m variable model has residuals $y_i - \sum_{j=1}^{m} x_{iv_j}\theta_j$ equal to zero when $i = i_1, \ldots, i_m$. If the criterion (7) is largest at variable $p \neq i_1, \ldots, i_m$, p may be added to the model as follows. Choose $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_m, \delta_{m+1})$ orthogonal to $(x_{iv_1}, \ldots, x_{iv_m}, x_{ip})$ for $i = i_1, \ldots, i_m$, which, since there are only m of them, can clearly be done. The function $f(t) = \Sigma | y_i - \sum_{j=1}^{m} x_{iv_j}(\theta_j + t\delta_j) - t x_{ip}\delta_{m+1} |$ is a sum with zero terms when $i = i_1, \ldots, i_m$, and $f(0)$ is the sum of absolute deviations at the current fit. If we write $f(t) = \Sigma | w_i - t u_i |$, where $w_i = y_i - \sum_{j=1}^{m} x_{iv_j}\theta_j$ and $u_i = \sum_{j=1}^{m} x_{iv_j}\delta_j + x_{ip}\delta_{m+1}$, it is clear that f is minimized at a value of $t = \hat{t} = w_q / u_q$ and that the q^{th} term of the sum is then equal to zero. Hence we have an $m+1$ variable model $(\boldsymbol{\theta}, 0) + \hat{t}\boldsymbol{\delta}$ based on variables $v_1, \ldots, v_m$, p and points $i_1, \ldots, i_m, q$ corresponding to zero residuals.

On the other hand if (7) is largest for variable v_p, say, the corresponding point i_p is deleted and replaced in the manner already described; an improved m variable model results. In any case it is the criterion (7) that determines whether we step up to a larger model or improve the current one without stepping up.

The mechanism described above for identifying a point to be dropped may give false indications of convergence in the presence of degeneracy. Degeneracy occurs when $k' > k$ residuals vanish at a given stage. In this case, there are $\binom{k'}{k}$ sets of points that all give the same fit. However, it is possible that not all of them can be improved by replacing a single point. To guard against the possibility of arriving at such a set of points and incorrectly concluding that the minimum has been reached, we examine all the sets before allowing the procedure to terminate.

4. Relationship to linear programming. From a computational standpoint two sets of quantities are required for the deletion phase, the residuals $y_i - \mathbf{x}_i^T\boldsymbol{\theta}$ and the weights $|\mathbf{x}_i^T\boldsymbol{\delta}|$, $i = 1, \ldots, n$. There are k different sets of weights, one for the value of $\boldsymbol{\delta}$ associated with the point in the current set being tested for deletion. Further, the k different $\boldsymbol{\delta}$ values are the columns of $\mathbf{Z}^{-1}$ in (8). The needed weights are thus the inner products of the rows of $\mathbf{X}$ with the columns of $\mathbf{Z}^{-1}$, the inverse of the submatrix of $\mathbf{X}$ corresponding to the data points in the current set. These quantities may be computed and updated easily by using the *pivot* operation of linear programming.

We begin with the bordered matrix $(\mathbf{X}:\mathbf{y})$. By *pivoting* on an entry in the $\mathbf{X}$ portion of this array, we mean first scaling the column in which that entry appears to make its value one, and then subtracting multiples of that column from each other column, to make the remaining entries in that row vanish. If we pivot on one entry in each row of the desired submatrix, with one entry also in each column, we obtain an array in which $\mathbf{X}$ has been replaced by the necessary inner products, and y has been

replaced by the residuals from the corresponding fit. For $\mathbf{X}$ has effectively been post-multiplied by a matrix that has reduced the relevant submatrix to the identity matrix, which must therefore be the desired inverse. Also, y has had multiples of the columns of $\mathbf{X}$ subtracted from it to make certain entries vanish, and hence has been replaced by the residuals from the corresponding fit.

When one point is replaced by another, we can adjust the array by a single additional pivot. If we pivot on that entry in the new row that falls in the same column as the unit entry in the old row, then the rows corresponding to the other points in the current set are unaffected, and hence the resulting array again contains all the values needed in the next step of the solution.

This pivoting operation is, of course, precisely the operation used in updating bases in the simplex method of linear programming. The question arises as to how well one can do using linear programming techniques. Initially, solutions by linear programming required the introduction of additional variables and constraints, as in the formulation:

$$\text{minimize} \quad \sum_{i=1}^{n} e_i$$

$$\text{subject to} \quad e_i \geqq y_i - \sum_{j=1}^{k} x_{ij}\theta_j,$$

$$e_i \geqq -y_i + \sum_{j=1}^{k} x_{ij}\theta_j, \qquad i = 1, \ldots, n.$$

Such solutions must inevitably operate with larger arrays than our method, and are not of comparable efficiency.

Another algorithm by Barrodale and Roberts (1973) seems to have been the best available method for solving (2). It appears to be closely related to linear programming but avoids the extra variables and constraints necessary in the direct formulation, above. However, as the next section strongly suggests, the present algorithm is much more efficient. It seems to be faster than the Barrodale-Roberts algorithm, and the relative advantage increases indefinitely with the size of the problem (2).

5. Computational aspects. In assessing the computational cost of the proposed least absolute deviations algorithm, it will be useful to compare it to ordinary least squares. For this purpose, we take the number of multiplication or division operations as a measure of complexity.

To find $\boldsymbol{\theta}$ in (1) with ordinary least squares, the k normal equations can be obtained at a cost of $(k-1)(k+2)n/2$ operations and solved in $k^3/3 + O(k^2)$ operations. By way of comparison, the computations involved in a single step of our algorithm are

* $2n(k-1)$ comparisons and $n(k-1)$ additions, to find the pivot column,
* finding the weighted median of the n ratios, which takes a number of comparisons and exchanges of the order of $n \log n$ and not more than n additions, and
* $n(k+1)$ multiplications and nk additions in the updating operation.

TABLE 1
An illustrative set of data from Karst (1958).

	1	2	3	4	5	6	7
x	12	18	24	30	36	42	48
y	5.27	5.68	6.25	7.21	8.02	8.71	8.42

If the least absolute deviations algorithm terminates at step N, the computational cost would be roughly

$$N(nk + \text{effort for weighted median}), \tag{9}$$

while ordinary least squares would require

$$n(k-1)(k+2)/2 + k^3/3 + O(k^2). \tag{10}$$

It is difficult to carry out the comparison further because we do not know how N depends on n, k, and $\mathbf{X}$. However, it is clear that as long as N is comparable to $k/2$ and $k^2/(2 \log n)$, the costs of ordinary least squares and our least absolute deviations algorithm will be comparable.

The remainder of this section is devoted to the question of how (9) might grow with n, k for certain specific or randomly presented curve fitting problems $\mathbf{X}$. However, instead of counting operations as in (9), we will measure complexity by the CPU time for executing the algorithm and compare this with other procedures for computing or approximating least absolute deviations fits.

To compare our "exact" procedure with that of Schlossmacher, consider the problems of obtaining a $k=2$ dimensional fit for the data of Table 1. In this problem the iterated least squares technique converged to a good approximation in 7 steps, while our least absolute deviations algorithm required 2. Although each of our iterations may be more work than a single step of the iterated least squares procedure, in view of (9) and (10) one expects our algorithm to find the exact fit in less time than that needed for convergence of the iterated least squares approximation.

Finally we performed Monte-Carlo experiments to compare our algorithm with the best available competition, that of Barrodale and Roberts (1974). The results are interesting and bear careful study.

For the model $Y=\theta_0+\theta_1 X_1+\cdots+\theta_k X_k+U, \theta_i=\sqrt{i}$, a sample $\mathbf{X}$ of size n was generated in the following way. For each $i=1,\ldots,n$, successive random numbers[1] $X_{i1}, X_{i2},\ldots, X_{ik}, U_i$ were generated and $Y_i=\theta_0+\theta_1 X_{i1}+,\ldots,+\theta_k X_{ik}+U_i$ formed. From this sample $\mathbf{X}$, LAD estimators $\hat{\theta}$ were computed with the Barrodale-Roberts algorithm and the present one, and the CPU times and numbers of iterations recorded. Since the Barrodale-Roberts iteration also exchanges one set of k zero residuals for another, it is sensible to compare iteration counts. In addition it may help to explain differences in the CPU times.

The above process was repeated for a total of 10 samples of size n. So that the comparisons do not depend too strongly on a particular sequence produced by the random number generator, the total CPU times and total iteration counts are compared.

[1]The FORTRAN function RAN of the DEC system 20 was used in this task.

Finally to explore the effect of the distribution of the point cloud $\mathbf{X}$ on the complexity of obtaining the LAD fits, three distributions for the X's and U were used. The first two are from the family of Pareto densities $f(t)=1/(1+t)^{1+\alpha}$, $t \geqq 0, \alpha > 0$. For $\alpha > 1$ the mean exists and equals $\alpha/(\alpha-1)$, and thus the density

$$f(t)=\alpha/[1+(t-c)]^{1+\alpha}, \qquad t \geqq c=\alpha/(\alpha-1),$$

is Pareto and centered at the mean.

We generated X's and U using this density, first when $\alpha=1.2$, and second when $\alpha=2.2$. In the former case the variance is infinite, while in the latter, though the density is long-tailed, the variance is finite. Finally, in a third set of experiments we used X's and U from the unit normal distribution.

The results are in Tables 2, 3, and 4. In each table, each cell, (n, k) has the total CPU time for the 10 samples and the total iteration count used by both the present algorithm, labeled LAD, and the Barrodale-Roberts algorithm, labeled BAR. Incidentally, the experiment reported in each cell of each table employed a different seed for the random number generator.

The results are rather striking. As the size n of the point cloud increases, our algorithm gains relative advantage over the Barrodale-Roberts method, for each k and with all underlying distributions. More specifically, the tables suggest that LAD(n), the CPU time of our algorithm, grows linearly with n, while BAR(n), the CPU time for Barrodale-Roberts grows faster than linearly, perhaps like $n \log n$ or n^2. The three tables would support

$$\text{LAD}(n)=C_k(U)\cdot n,$$

$$\text{BAR}(n)=d_k(U)\cdot n\log n \quad \text{or} \quad d_k(U)\cdot n^2,$$

TABLE 2
Total CPU *time and iteration counts for* 10 *sets of* LAD *estimates, Pareto distribution,* $\alpha=1.2$.

n \ k	2		3		6	
	BAR	LAD	BAR	LAD	BAR	LAD
100	.76	.79	1.20	1.10	2.77	3.19
	34	42	58	34	119	71
300	3.08	2.34	4.74	4.10	11.20	10.79
	38	45	68	60	148	83
600	8.42	4.51	12.04	7.44	25.10	20.07
	44	40	65	49	136	78
1200	24.33	10.18	34.89	16.27	71.72	44.69
	45	52	71	61	167	109
1800	47.22	14.34	72.00	22.91	136.14	68.57
	41	48	77	53	163	108

TABLE 3
Total CPU *time and iteration counts for* 10 *sets of* LAD *estimates, Pareto distribution,* $\alpha = 2.2$.

		k = 2		3		6	
		BAR	LAD	BAR	LAD	BAR	LAD
n	100	.81	.83	1.21	1.31	2.91	3.41
		43	51	63	51	140	84
	300	2.83	2.51	4.53	4.15	10.41	11.33
		43	54	71	64	141	112
	600	7.35	5.64	11.18	8.08	25.91	23.87
		43	66	67	61	159	125
	1200	22.57	11.07	33.47	18.17	79.89	56.05
		52	60	87	78	231	165
	1800	40.41	17.41	63.99	24.87	139.81	89.77
		38	69	80	65	214	183

TABLE 4
Total CPU *time and iteration counts for* 10 *sets of* LAD *estimates, Gaussian distribution.*

		k = 2		3		6	
		BAR	LAD	BAR	LAD	BAR	LAD
n	100	.75	.98	1.40	1.55	3.83	4.41
		29	57	75	61	185	124
	300	2.59	3.17	5.21	5.61	15.28	16.04
		33	67	85	96	245	184
	600	6.62	6.50	12.65	10.89	41.49	35.15
		42	70	93	92	322	222
	1200	17.33	13.05	37.06	23.62	105.92	70.27
		47	67	121	104	341	219
	1800	29.63	19.52	68.93	34.66	182.80	121.88
		46	71	120	102	328	274

where $C_k(f)$, $d_k(f)$ are constants that both increase with k and each depends on f, the distribution of the X's and U (but in different ways to be mentioned presently). In fact, it seems reasonable to assert that

$$\frac{\text{BAR}(n)}{\text{LAD}(n)} = a(f) \text{ times an increasing function of } n,$$

where $a(f)$ is a constant depending only on the point cloud's density, f; thus, both $C_k(f)$ and $d_k(f)$ increase with k in the same way.

Finally, it is interesting to observe that f affects the algorithms in different ways:

(1) LAD is best for "spread out" point clouds, $\alpha = 1.2$, and worst for the normal data;

(2) BAR is best for finite variance but long-tailed point clouds, $\alpha = 2.2$, and worst, sometimes in the normal case, sometimes in the infinite variance case, $\alpha = 1.2$. In any event, the relative advantage of LAD is greatest when $\alpha = 1.2$, when the data are most heavy-tailed.

These observations are tentative, as there is little evidence to base such exact assertions upon. Nevertheless, it seems safe to claim that our algorithm has an increasing asymptotic advantage over Barrodale and Roberts as n, the number of points being fit, increases. More strongly,

$$\frac{\text{BAR}(n)}{\text{LAD}(n)} \to \infty$$

as $n \to \infty$. It would be interesting to study the ratio for more values of k and other types of distributions.

REFERENCES

N. N. ABDELMALEK (1971), *Linear L_1 approximation for a discrete point set and L_1 solutions of overdetermined linear equations*. J. Assoc. Comput. Mach., 18, pp. 41-47.

DAVID ANDREWS (1974), *A robust method for multiple linear regression*, Technometrics, 16, pp. 523–532.

I. BARRODALE AND F. D. K. ROBERTS (1973), *An impoved algorithm for discrete l_1 linear approximation*, SIAM J. Numer. Anal., 10, pp. 839–848.

_____ (1974), *Algorithm* 478: *solution of an overdetermined system of equations in the l_1 norm*, Comm. ACM, 17, pp. 319–320.

J. CHAMBERS (1971), *Algorithm* 410: *partial sorting*, Comm. ACM, 14, pp. 357–358.

F. Y. EDGEWORTH (1887), *A new method of reducing observations relating to several quantities*, Phil. Mag. (Fifth Series), 24, pp. 222–223.

_____ (1888), *On a new method of reducing observations relating to several quantities*. Phil. Mag. (Fifth Series), 25, pp. 184–191.

C. EISENHART (1961), *Boscovitch and the combination of observations* in Roger Joseph Boscovitch, L. L. Whyte, eds., Fordham University Press, New York.

T. HARRIS (1950), *Regression using minimum absolute deviations*, Amer. Statistician, 4, pp. 14–15.

PETER J. HUBER (1973), *Robust regression: asymptotics, conjectures, and Monte Carlo*, Ann. Statist. 1, pp. 799–821.

J. OTTO KARST (1958), *Linear curve fitting using least deviations*. J. Amer. Statist. Assoc., 53, pp. 118–132.

S. C. NARULA AND J. F. WELLINGTON (1977), *Algorithm AS*108. *Multiple linear regression with minimum sum of absolute errors*. Applied Stat., 26, pp. 106–111.

E. C. RHODES (1930), *Reducing observations by the method of minimum deviations*, Phil Mag. (Seventh Series), 9, pp. 974–992.

A. N. SADOVSKI (1974), *Algorithm AS*74. *L_1-norm fit of a straight line*. Applied Stat., 23, pp. 244–248.

E. J. SCHLOSSMACHER (1973), *An iterative technique for absolute deviations curve fitting*. J. Amer. Statist. Assoc., 68, pp. 857–865.

R. R. SINGLETON (1940), *A method for minimizing the sum of absolute values of deviations*, Ann. Math. Statist., 11, pp. 301–310.

V. SPOSITO (1976), *Remarks on Algorithm AS*74. Applied Stat., 25, pp. 96–97.

K. H. USOW (1967), *On L_1 approximation II: Computation for discrete functions and discretization effects*, SIAM J. Numer. Anal., 4, pp. 233–224.

HARVEY M. WAGNER (1959), *Linear programming techniques for regressions analysis*, J. Amer. Statist. Assoc., 54, pp. 206–212.