1977

Fast leastsquares algorithms

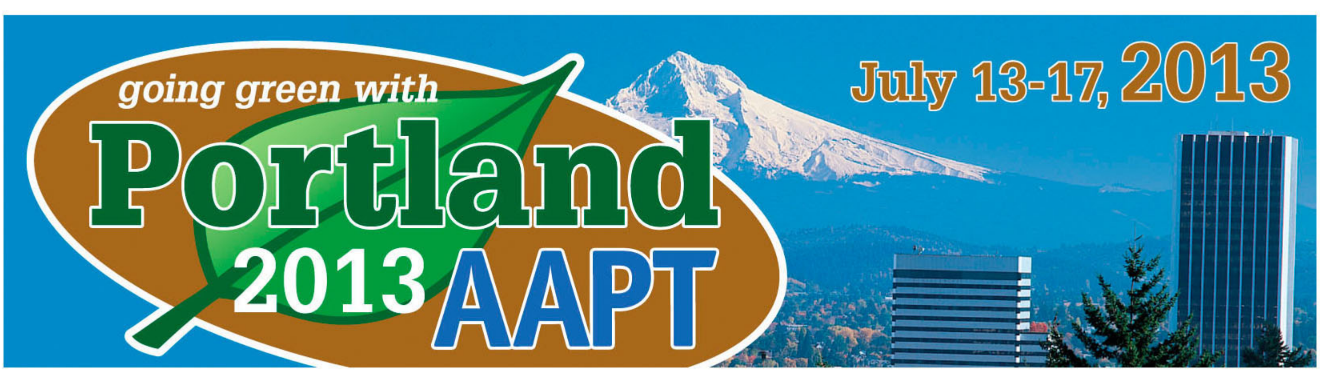
William C. Davidon
Haverford College

AAPT PHYSICS EDUCATION
AMERICAN JOURNAL of PHYSICS

Fast leastsquares algorithms

William C. Davidon

Citation: Am. J. Phys. **45**, 260 (1977); doi: 10.1119/1.11004
View online: http://dx.doi.org/10.1119/1.11004
View Table of Contents: http://ajp.aapt.org/resource/1/AJPIAS/v45/i3
Published by the American Association of Physics Teachers



Fast least-squares algorithms

William C. Davidon
Department of Physics, Haverford College, Haverford, Pennsylvania 19041
(Received 3 June 1976; revised 21 September 1976)

New least-squares algorithms are introduced. Instead of waiting for all data to be in before making a fit, these algorithms update a fit after each point is entered so trends can be detected promptly as an experiment proceeds. Coupled linear equations are not solved numerically, reducing rounding errors, calculation time, and memory requirements. When used for fitting degree-N polynomials to equally weighted data points whose abscissas are equally spaced, these algorithms need just one multiplication by an integer constant and one division to update each of the $N+1$ polynomial coefficients. Pocket calculator programs are available for polynomial fits to data points whose abscissas are equally spaced; one of these gives equal weight to all points while another gives more weight to recent points.

1. INTRODUCTION

Fast algorithms are defined as those which reduce by orders of magnitude the number of operations needed to solve certain types of problems. For example, the fast Fourier transform introduced by Cooley and Tukey in 1965 needs only a fixed multiple of $N \log N$ rather than the N^2 operations previously used when approximating the Fourier transform of a function, given its value at N equally spaced points.[1] Similar fast algorithms reduce the number of operations needed for matrix multiplication or for solving coupled sets of linear equations, though these are advantageous only for very large problems.[2,3] The fast least-squares algorithms introduced in this paper need only a fixed multiple of MN rather than the usual MN^2 operations to fit M data points with N parameters.

Among the least-squares algorithms previously described in this Journal, some introduce shortcuts specific to the important special case of straight-line fits.[4,5] More general algorithms for curve-fitting by higher degree polynomials, or by linear combinations of other functions, have either numerically solved a coupled set of linear equations or else used an appropriate set of orthogonal functions to eliminate the coupling.[6,7] Only the equation solvers can be used in the general case when the abscissas of the data points or their relative weights are not all specified in advance. However, these not only need more time and memory space than algorithms using orthogonal functions, but the coupled equations they solve can be so ill-conditioned that rounding errors make their output inaccurate, if not useless.

When the abscissas of all the data points and their relative weights have a preset pattern, algorithms using orthogonal functions have been preferable. These evaluate all of the orthogonal functions at the abscissa of each data point, and they determine a fit only after all the data are in, so they provide no intermediate output to help guide an experiment as it proceeds.

The new algorithms to be introduced, like those using orthogonal functions, require that the abscissas of all the data points and their relative weights have some preset pattern. But unlike the older algorithms, these new ones evaluate only one function at each data point, and they update a fit as each data point is entered so trends can be detected and acted upon promptly. In their general form, these algorithms fit data by functions from any function space F [such as the $(N+1)$-dimensional space of all polynomials of degree at most N, or the $2N$-dimensional space of Fourier series with N nonzero frequencies] which is closed under real linear combinations; i.e., if f and g are any two functions in the space and α and β are any two real numbers, then $\alpha f + \beta g$ must also be in F. Given such a function space F and a sequence of data points (x_k, y_k) with weights $w_k \geqslant 0$ for each integer $k \geqslant 1$, these algorithms find a corresponding sequence of functions f_k from the function space F, each of which minimizes the weighted sum of squares

$$(\chi^2)_k = \sum_{1 \leqslant i \leqslant k} [y_i - f_k(x_i)]^2 w_i. \tag{1.1}$$

The weights w_k and abscissas x_k, which may be common to many data sets, are used to determine basis functions u_k from F, each of which minimizes the weighted sum of squares

$$[1 - u_k(x_k)]^2 w_k + \sum_{1 \leqslant i < k} u_k{}^2(x_i) w_i. \tag{1.2}$$

Updates from f_{k-1} to f_k and from $(\chi^2)_{k-1}$ to $(\chi^2)_k$ are made using these basis functions u_k and the differences $y_k - f_{k-1}(x_k)$ between the actual kth ordinate y_k and a predicted one $f_{k-1}(x_k)$ based on the least-squares fit to the $k-1$ previous points; specifically,

$$f_k = f_{k-1} + [y_k - f_{k-1}(x_k)] u_k \tag{1.3}$$

and

$$(\chi^2)_k = (\chi^2)_{k-1} + [y_k - f_{k-1}(x_k)]^2 [1 - u_k(x_k)] w_k. \tag{1.4}$$

The starting function f_0 is arbitrary, since for $k = 0$ there are no points to fit and $(\chi^2)_0 = 0$ for any f_0.

These algorithms are particularly simple and fast when F is the space of all polynomial functions of degree at most N, when all the abscissas x_k are equally spaced, and when the weights w_k are either all equal or else form an increasing geometric sequence giving more weight to recent points. The simplest of these is for $N = 0$, when the algorithm reduces to one for simple averaging. This is described in Sec. 2 to provide a most familiar context for comparing certain features of the new algorithms with others. Section 3 gives a more typical and useful example for fits by fourth-degree polynomials. Precise statements of the basic mathematical properties of the general algorithm are given in the Appendix.

2. AVERAGING

Simple averaging can be viewed as a least-squares fit to data by polynomials of degree 0, and this was the first use of least-squares by Gauss, who originated the method in 1795 when he was eighteen.[8] In this special case, no difference remains between algorithms solving coupled linear equations and ones using orthogonal polynomials. Both find the average a_k of the first k ordinates $y_1, y_2, \ldots, y_k$ in a sequence by

$$a_k = \sum_{1 \leqslant i \leqslant k} \frac{y_i}{k}, \tag{2.1}$$

and the chi-squared measure of how well these fit the data by

$$(\chi^2)_k = \sum_{1 \leqslant i \leqslant k} (y_i - a_k)^2 = \sum_{1 \leqslant i \leqslant k} y_i{}^2 - \left(\frac{1}{k}\right)\left(\sum_{1 \leqslant i \leqslant k} y_i\right)^2. \tag{2.2}$$

In this case, the functions f in the function space F are just constants $f(x) = a$ and the weights w_k all equal 1, so the basis functions u_k minimizing expression (1.2) are the constants $1/k$. The initial a_0 is arbitrary, $(\chi^2)_0 = 0$, Eq. (1.3) for updating f_k simplifies to

$$a_k = a_{k-1} + (y_k - a_{k-1})/k, \tag{2.3}$$

and Eq. (1.4) for updating $(\chi^2)_k$ simplifies to

$$(\chi^2)_k = (\chi^2)_{k-1} + (y_k - a_{k-1})^2(1 - 1/k). \tag{2.4}$$

While Eqs. (2.3) and (2.4) offer little computational advantage over their more familiar counterparts, Eqs. (2.1) and (2.2), they still may throw some light on certain aspects of this family of least-squares algorithms. Proving that Eqs. (2.1) and (2.3) give the same a_k and that Eqs. (2.2) and (2.4) give the same $(\chi^2)_k$ is an exercise in mathematical induction which may be useful, particularly for those who may not wish to study the more general mathematical results given in the Appendix.

3. QUARTIC FITS

Input to the algorithm specified in this section again consists of just the ordinates y_k from a sequence of data points (x_k, y_k) whose abscissas x_k are equally spaced. Its output is a sequence of fourth-degree polynomial functions p_k, scaled so that $p_k(i)$ is an estimate for the ith ordinate y_i based on the least-squares fit to the first k points. Hence, each p_k minimizes the chi-squared measure $(\chi^2)_k$ of the fit to the first k data points:

$$(\chi^2)_k = \sum_{1 \leqslant i \leqslant k} [y_i - p_k(i)]^2. \tag{3.1}$$

Minimizing $(\chi^2)_k$ determines the quartic uniquely if and only if the number k of points is at least 5. For $k \leqslant 5$, $(\chi^2)_k = 0$ and there is a $(5 - k)$-dimensional subspace of quartics which all fit the data exactly.

Of the many ways to use five real numbers to specify the quartics p_k, one which offers several computational advantages is

$$p_k(t) = a_k + \left\{b_k + \left[c_k + \left(d_k + e_k \frac{t-k-4}{4}\right) \times \frac{t-k-3}{3}\right] \frac{t-k-2}{2}\right\} (t-k-1). \tag{3.2}$$

With this expansion, no further computation is needed to evaluate the prediction $p_{k-1}(k) = a_{k-1}$ for the kth ordinate based on the least-squares fit to the first $k - 1$ points.

The five coefficients of the starting quartic are arbitrary, for again there are not yet any points to fit. However, rounding errors are usually minimized with $a_0 = b_0 = c_0 = d_0 = e_0 = 0$. Equation (1.3) for updating f_k now reduces to

$$a_k = a_{k-1} + b_{k-1} + 25 \frac{y_k - a_{k-1}}{k}, \tag{3.3a}$$

$$b_k = b_{k-1} + c_{k-1} + 300 \frac{y_k - a_{k-1}}{k(k+1)}, \tag{3.3b}$$

$$c_k = c_{k-1} + d_{k-1} + 2100 \frac{y_k - a_{k-1}}{k(k+1)(k+2)}, \tag{3.3c}$$

$$d_k = d_{k-1} + e_{k-1} + 8400 \frac{y_k - a_{k-1}}{k(k+1)(k+2)(k+3)}, \tag{3.3d}$$

and

$$e_k = e_{k-1} + 15\,120 \frac{y_k - a_{k-1}}{k(k+1)(k+2)(k+3)(k+4)}. \tag{3.3e}$$

When fitting by polynomials of degree N, the $N + 1$ integers replacing 25, 300, 2100, 8400, and 15 210 in the generalization of Eqs. (3.3) are

$$(N+1) \frac{(N+n+1)!}{(N-n)!\,(n+1)!}$$

for n from 0 through N.

Equation (1.4) for updating $(\chi^2)_k$ reduces in this case to

$$(\chi^2)_k = (\chi^2)_{k-1} + (y_k - a_{k-1})^2 \times \frac{(k-1)(k-2)(k-3)(k-4)(k-5)}{k(k+1)(k+2)(k+3)(k+4)}. \tag{3.4}$$

The updates of Eqs. (3.3) take just one multiplication by an integer constant and one division for each coefficient of the least-squares polynomial. The update of Eq. (3.4) takes $N + 2$ additional multiplications when fitting polynomials of degree N, for a total of $3N + 4$ multiplications or divisions to update both the least-squares polynomial as well as the chi-squared measure of how well it fits all past data. Only $N + 3$ quantities need be stored from one iteration to the next: the current k, the $N + 1$ polynomial coefficients, and $(\chi^2)_k$.

One measure of the simplicity of this algorithm for fitting quartics to equally spaced and equally weighted data is that a program of 99 steps is available for an HP-65 pocket calculator that updates the quartic p_k and its $(\chi^2)_k$ using Eqs. (3.3) and (3.4), and that also evaluates p_k for any t using Eq. (3.2).[9] A similar 100-step program is also available for sixth-degree fits which give more weight to recent points, but without the calculation of the weighted chi-squares.[10]

APPENDIX

Here, while generalizing the specific algorithms of Secs. 2 and 3, we continue to consider only the field **R** of real numbers, though complex or other number fields could be used as well. Our first theorem follows directly from linearity or the superposition principle: a least-squares fit to the sum of two sequences is the sum of the least-squares fits to each.

Theorem 1. Let F be any real vector space of functions from **R** to **R**, and for each integer $k \geqslant 1$, let w_k, x_k, and y_k be real numbers with $w_k \geqslant 0$. Define f_0 as any function in F. For each integer $k \geqslant 1$, define u_k as a function in F that minimizes expression (1.2) and define f_k by Eq. (1.3). Then each f_k minimizes the weighted sum of squares $(\chi^2)_k$ of Eq. (1.1), and these minimal $(\chi^2)_k$ satisfy Eq. (1.4).

This theorem is computationally useful because the functions u_k depend only on the function space F, the weights w_k, and the abscissas x_k, but not on the ordinates y_k of the data points to be fitted. Hence, the same set of functions u_k can be used in many different experimental runs, reducing the computation needed for each. In particular, if F is the space of all polynomial functions of degree at most N; if all the weights w_k are equal, and if the abscissas x_k are just $x_k = k$ for all $k \geqslant 1$, then we can express the u_k as follows.

Theorem 2. For each integer $k \geqslant 1$, a polynomial u_k of degree N which minimizes

$$\chi^2 = [1 - u_k(k)]^2 + \sum_{1 \leqslant i < k} u_k{}^2(i)$$

is

$$u_k(t) = \sum_{0 \leqslant n \leqslant N} \frac{(N+1)(N+n+1)!\,(k-1)!}{(N-n)!\,(n+1)!\,(n+k)!} \times \binom{t-k-1}{n},$$

for which

$$\chi^2 = 1 - u_k(k) = \binom{k-1}{N+1}\binom{N+k}{N+1}^{-1},$$

where $\binom{x}{n}$ is the binomial coefficient, defined for any real x and integer $n \geqslant 0$ by

$$\binom{x}{0} = 1 \qquad \text{and} \qquad \binom{x}{n+1} = \binom{x}{n}\frac{x-n}{n+1}.$$

The polynomial u_k is the only one minimizing χ^2 if and only if $k \geqslant N + 1$.

To prove this theorem, and its counterparts for other abscissas and weightings, use orthogonal polynomials for minimizing the appropriate sum of squares.[11] Once this is done and the functions u_k are determined, then only the u_k and not the orthogonal polynomials are needed for the least-squares algorithm itself. The integer coefficients $(N+1)(N+n+1)!/(N-n)!(n+1)!$ appearing in the sum for each u_k and, hence, in Eqs. (3.3) are just $(-1)^n n!$ times the corresponding element in the first column of the inverse of the $(N+1) \times (N+1)$ Hilbert matrix.[12]

[1]J. W. Cooley and J. W. Tukey, Math. Comp. **19**, 297–301 (1965).

[2]V. Strassen, Numer. Math. **13**, 354–356 (1969).

[3]E. Isaacson and H. B. Keller, *Analysis of Numerical Methods* (Wiley, New York, 1966), pp. 34–37.

[4]M. A. Heald, Am. J. Phys. **37**, 655–662 (1969).

[5]C. James and R. G. Chambers, Am. J. Phys. **39**, 686–687 (1971).

[6]J. Mathews and R. L. Walker, *Mathematical Methods of Physics* (Benjamin, New York, 1970), pp. 387–396.

[7]P. R. Bevington, *Data Reduction and Error Analysis for the Physical Sciences* (McGraw-Hill, New York, 1969), pp. 92–163.

[8]E. T. Bell, *The Development of Mathematics*, 2nd ed. (McGraw-Hill, New York, 1945), p. 586.

[9]W. C. Davidon, Program 4654R (HP-65 Users' Library, Hewlett-Packard, Cupertino, CA 95014; 1976).

[10]W. C. Davidon, Program 4681A (HP-65 Users' Library, Hewlett-Packard, Cupertino, CA 95014; 1976).

[11]M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions* (Dover, New York, 1965), pp. 788–792.

[12]M. Marcus, *Basic Theorems in Matrix Theory: Applied Mathematics Series, No. 57* (Natl. Bur. Stand., Washington, DC, 1960), p. 23.