ALGORITHM 28
LEAST SQUARES FIT BY ORTHOGONAL POLYNOMIALS

JOHN G. MACKINNEY
General Kinetics Incorporated, Arlington 6, Virginia

```
procedure LSFIT (f, x1, xm, m, k, alpha, beta, sigma, s, p) ;
          value x1, xm, m, k ; real x1, xm ; integer
            m, k ;
          real array f, alpha, beta, sigma, s, p ;
comment LSFIT accepts m values of the function f at equal
   intervals of the abscissa from x1 through xm, and obtains in
   p[0] through p[k] the coefficients of the best polynomial ap-
   proximation of degree k or less (least squares) as programmed
   by George E. Forsythe, Journal SIAM 5, no. 2, June 1957,
   with only minor variations. The output values alpha [1:k],
   beta [0:k], and s [0:k] enable the user to make final adjust-
   ments to the results, according to the statistic sigma [0:k].
   LSFIT uses the procedure POLYX (a, b, c, d, n) to trans-
   form its results from the interval (−2, 2) to the interval (x1,
   xm) ;
begin integer i, j ; real dummy, x, xone, deltax, delsq,
            omega, lastw, thisw ;
          real array cthisp, cpoly [0:k], clastp [−1:k],
            lastp, thisp [1:m] ;
          Boolean swx ;
comment Initialization ;
   swx := true ; beta [0] := clastp [0] := clastp [−1] :=
     delsq := omega := 0 ;
   cthisp [0] := 1 ; thisw := m ;
   for i := 1 step 1 until m do
   begin delsq := delsq + f[i]↑2 ;
       thisp [i] := 1 ; lastp [i] := 0 ;
       omega := omega + f[i] end ;
   s [0] := cpoly [0] := omega/thisw ;
   delsq := delsq − s [0] × omega ;
   sigma [0] := delsq/(m−1) ;
comment Transformation of abscissa ; i := m + 2 ;
         if 2×i = m then deltax := 4/(m − 1) else deltax :=
           4/m ; xone := −2 ;
comment Main Computation loop ;
         for i := 0 step 1 until k−1 do
         begin dummy := 0 ; x := xone ;
         1: for j := 1 step 1 until m do
         begin dummy := dummy + x × thisp [j] ↑ 2 ;
         x := x + deltax end ;
         2: alpha [i + 1] := dummy/thisw ;
             lastw := thisw ;
             thisw := omega := 0 ;
            x := xone ;
         3: for j := 1 step 1 until m do
         begin dummy := beta [i] × lastp [j] ;
             lastp [j] := thisp [j] ;
             thisp [j] := (x − alpha [i + 1]) × thisp [j]
               −dummy ;
             thisw := thisw + thisp [j] ↑ 2 ;
             omega := omega + f [j] × thisp [j] ;
             x := x + deltax end ;
         4: beta [i + 1] := thisw / lastw ;
         s[i + 1] := omega / thisw ;
            delsq := delsq − s[i + 1] × omega ;
         sigma [i + 1] := delsq / (m − i − 1) ;
         if swx then go to 6 ;
         5: cpoly [i + 1] := 0 ; go to 9 ;
comment Termination of main loop when higher power will
           not improve fit ;
         6: if sigma [i + 1] < sigma [i] then go to 7 ;
            swx := false ; go to 5 ;
comment Recursion for polynomial coefficients ;
         7: for j := 0 step 1 until i do
            begin dummy := clastp [j] × beta [i] ;
            clastp [j] := cthisp [j] ;
cthisp [j] := clastp [j − 1] − alpha [i + 1] × cthisp [j] − dummy ;
cpoly [j] := cpoly [j] + s [i + 1] × cthisp [j] end ;
    8: cpoly [i + 1] := s [i + 1]
    cthisp [i + 1] := 1 ;
    9: clastp [i + 1] := 0 end of main
computation loop, transformation of polynomial follows ;
    begin real a, b ;
    a := deltax × (m − 1) / (xm − x1) ;
    b := xone − a × x1 ;
    POLYX (a, b, cpoly, p, k) end
end of LSFIT
```

REMARK ON ALGORITHM 28
LEAST-SQUARES FIT BY ORTHOGONAL POLYNOMIALS (John G. MacKinney, *Comm. ACM 3* (Nov. 1960))

D. B. MACMILLAN
Knolls Atomic Power Laboratory, General Electric Co., Schenectady, N. Y.

The algorithm obtains the coefficients of the fitted polynomial of lowest degree such that an increase in the degree would cause an increase in the statistic sigma (sigma squared in Forsythe's notation). A significant decrease in sigma, as one goes from a fitted polynomial to one of higher degree, indicates that the increase in degree causes an improvement in the fit to the function underlying the data, rather than merely following more closely the random variations about that function introduced by the physical measurement process.

If one of the orthogonal polynomials, say the one of ith degree, is missing from the underlying function, and some of the orthogonal polynomials of higher degree are present, then the fitted polynomial of ith degree will not be a real improvement over that of $(i - 1)$-th degree, but higher order fitted polynomials will be a real improvement. For example, in one of our recent routine problems the coefficient of the second degree orthogonal polynomial was quite small, and the first few values of sigma, starting with sigma (1), were .255, .264, .062, .046, .048. The algorithm would have chosen the first degree fitted polynomial as "best", but the third and fourth degree fitted polynomials were clearly better than it.

This loophole may be plugged by modifying the algorithm so it computes the coefficients of the polynomial of lowest degree i for which it is true that

$$\text{sigma }(i + 1) \geq \text{sigma }(i)$$

and that

$$\text{sigma }(j) \geq .6 \text{ sigma }(i) \qquad j = i+2, i+3, \cdots, k,$$

(.6 was chosen arbitrarily).

REMARK ON ALGORITHM 28 [E2] LEAST SQUARES FIT BY ORTHOGONAL POLYNOMIALS [John G. MacKinney, *Comm. ACM 3* (Nov. 1960), 604]

G. J. Makinson (Recd. 30 Sept. 1965, 29 Aug. 1966 and 7 Nov. 1966)
University of Liverpool, Liverpool 3, England

There are three errors in the published procedure.

Line 32 $i := m + 2$; should read $i := m \div 2$;

Line 56 $delsq/(m-i-1)$; should read $delsq/(m-i-2)$;

Line 69 ; is missing from end of statement $cpoly[i+1] := s[i+1]$;

Three improvements can be made to the procedure. In the case of equally spaced points, it is possible to center them about the origin; all alphas are then zero. This is achieved by replacing the statements on lines 32, 33, and 34 by $deltax := 4/(m-1)$; $xone := -2$; All statements involving alphas can then be revised.

Another improvement can be made by deleting the two statements on line 37 and all of lines 38, 39, and 40. These statements are completely redundant.

The third improvement is to rewrite line 71 to read

clastp[*i*+1] := 0; 9: **end** *of main*

instead of

9: *clastp*[*i*+1] := 0 **end** *of main*