$0 \Rightarrow x_{i1} = x_{i2}$. Therefore, from the first part of (4), $x_{i1} = x_{i2} = y_i/2$. The factor $\frac{1}{2}$ need not be used because eigenvectors are obtained only up to a constant multiplier.

(ii) $\mu_i \in \lambda(Q) \Rightarrow \mu_i = \eta_j$, $(j \in I2)$, $\Rightarrow r_i = cz_j = x_{i1} - x_{i2}$, where $c$ is an arbitrary constant. Therefore, $cz_j = x_{i1} - x_{i2}$ and $y_i = x_{i1} + x_{i2}$ give $x_{i1} = (y_i + cz_j)/2$ and $x_{i2} = (y_i - cz_j)/2$. Again the constant factor need not be used.

A proof for the case $i \in I2$ can be similarly constructed.

### 3. A Numerical Example

As an illustration of the use of Algorithm 1 we consider a $4 \times 4$ matrix

$$S = \begin{bmatrix} 0.25 & 3.25 & -1.25 & -1.25 \\ -1.25 & 0.75 & -1.75 & 3.25 \\ -1.25 & -1.25 & 0.25 & 3.25 \\ -1.75 & 3.25 & -1.25 & 0.75 \end{bmatrix}. \quad (9)$$

From (9) we have

$$P = A + B = \begin{bmatrix} -1 & 2 \\ -3 & 4 \end{bmatrix},$$
$$Q = A - B = \begin{bmatrix} 1.5 & 4.5 \\ 0.5 & -2.5 \end{bmatrix}. \quad (10)$$

Solving for the eigenvalues and eigenvectors of $P$ and $Q$ gives $\lambda(P) = (\lambda_1, \lambda_2) = (\mu_1, \mu_2) = (1, 2)$; $\lambda(Q) = (\lambda_3, \lambda_4) = (\eta_3, \eta_4) = (2, -3)$, $y_1' = (1, 1)$, $y_2' = (\frac{2}{3}, 1)$, $z_3' = (9, 1)$, and $z_4' = (-1, 1)$.

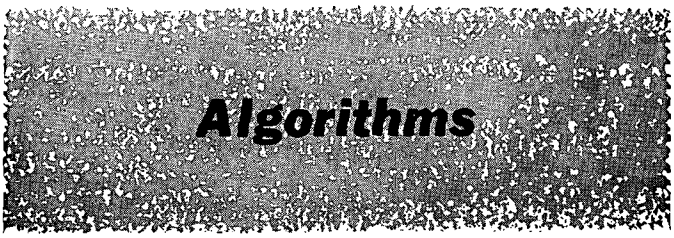
By use of Algorithm 1 the eigenvector of $S$ belonging to $\lambda_1 = 1$ can be formed by noting that $\lambda_1 \notin \lambda(Q)$. Therefore, $x_1' = (y_1', y_1') = (1, 1, 1, 1)$. Similarly, to form $x_2$ belonging to $\lambda_2 = 2$ we note that $\lambda_2 \in \lambda(Q)$. In particular $\lambda_2 = \lambda_3$. Therefore, $x_2' = (y_2' + cz_3', y_2 - cz_3') = (\frac{2}{3} + 9c, 1 + c, \frac{2}{3} - 9c, 1 - c)$, where $c$ is an arbitrary constant.

*Acknowledgment.* The preparation of this paper was supported by the National Research Council of Canada Grant No. A-3135.

RECEIVED DECEMBER, 1966; REVISED MAY, 1967

REFERENCES

1. FRIEDMAN, B. Eigenvalues of compound matrices. Research Rept. No. TW-16, Math. Res. Group. New York U., New York, 1951.
2. MARCUS, M. *Basic Theorems in Matrix Theory.* Appl. Math. Ser. 57, Nat. Bur. Standards, 1960, p. 17, Govt. Printing Office, Washington, D.C., p. 17
3. MONTGOMERY, C. G. ET AL. *Principles of Microwave Circuits.* Boston Technical Publishers, Inc., Lexington, Mass. 1964, pp. 437–452.
4. MONTROLL, E. W. Markoff chains and excluded volume effect in polymer chains. *J. Chem. Phys. 18* (1950), 734–743.




J. G. HERRIOT, Editor

## ALGORITHM 318
## CHEBYSCHEV CURVE-FIT (REVISED) [E2]

J. BOOTHROYD (Recd. 15 May 1967)
University of Tasmania, Hobart, Tas., Australia

```
procedure chebfit(x, y, n, a, m); value n, m;
  array x, y, a; integer n, m;
comment evaluates, in a[0] through a[m] of a[0:m+1], the co-
  efficients of an mth order polynomial P(x) = a0 + a1x + ··· amx^m
  such that the maximum error abs(P(xi)−yi)) is a minimum over
  the n(n>m+1) sample points x, y[1:n]. The x[i] must form a
  strictly monotonic sequence.
  This procedure is an extensive revision of Algorithm 91 (Albert
  Newhouse, Chebyshev Curve-Fit, Comm. ACM 5 (May 1961),
  281). The polynomial P(x) is a best-fit polynomial in the Cheby-
  shev sense as described by Stiefel (Numerical Methods of Tcheby-
  cheff Approximation), in Langer (Ed.), On Numerical Approxi-
  mation, U. of Wisconsin Press, 1959, pp. 217–232. Stiefel (p. 221)
  shows that the procedure must terminate after a finite number
  of steps. This is not always so with imperfect arithmetic, where
  roundoff errors may cause cycling of the chosen reference sets.
  This condition is detected by checking that the reference devia-
  tion is always raised monotonically. At exit the absolute value
  of a[m+1] yields the final reference deviation. Negative a[m+1]
  indicates that the procedure has been terminated following the
  detection of cycling;
begin
  integer i, j, k,mplus1, ri, i1, imax, rj, j1;
  real d, h, ai1, rhi1, denom, ai, rhi, xj, hmax, himax, xi, hi, abshi,
    nexthi, prevh;
  integer array r[0:m+1]; array rx, rh[0:m+1];
  mplus1 := m + 1; prevh := 0;
  comment index vector for initial reference set;
  r[0] := 1; r[mplus1] := n;
  d := (n−1)/mplus1; h := d;
  for i := 1 step 1 until m do
  begin r[i] := h + 1; h := h + d end;
start: h := −1.0;
  comment select m + 2 reference pairs and set alternating
    deviation vector;
  for i := 0 step 1 until mplus1 do
  begin
    ri := r[i];
    rx[i] := x[ri]; a[i] := y[ri];
    rh[i] := h := −h
  end i;
  comment compute m + 1 leading divided differences;
  for j := 0 step 1 until m do
  begin
    i1 := mplus1; ai1 := a[i1];
    rhi1 := rh[i1];
    for i := m step −1 until j do
    begin
      denom := rx[i1] − rx[i−j];
      ai := a[i]; rhi := rh[i];
      a[i1] := (ai1−ai)/denom;
```

(Continued on page 803)

# Index By Subject To Algorithms, 1967


A1 REAL ARITHMETIC, NUMBER THEORY
A1 35 SIEVE OF ERATOSTHENES 3-61(151),4-62(209),
A1 35 8-62(438),9-67(570)
A1 307 SYMMETRIC GROUP CHARACTERS 7-67(451)
A1 310 PRIME NUMBER GENERATOR 1 9-67(569),9-67(570)
A1 311 PRIME NUMBER GENERATOR 2 9-67(570),9-67(570)
A1 313 MULTI-DIMENSION PARTITION GEN. 10-67(666)
A1 SUM OF FACTORS OF N (SUMFAC) COMP.J.V9(416)

A2 COMPLEX ARITHMETIC
A2 ADD,SUB,MULT,DIVD-COMPLEX COMP.J.V10(112),
A2 V10(208)
A2 312 COMPLEX ABS,SQRT 10-67(665)

C1 OPERATIONS ON POLYNOMIALS AND POWER SERIES
C1 305 SYMMETRIC POLYNOMIALS 7-67(450)
C1 EVALUATION OF CONTINUED FRACTNS CHIFFRES V9(327)

C2 ZEROS OF POLYNOMIALS
C2 30 BAIRSTOW-NEWTON 12-60(643),
C2 30 5-61(238),5-67(293)
C2 BAIRSTOW COMP.J.V10(207)

C5 ZEROS OF ONE OR MORE TRANSCENDENTAL EQUATIONS
C5 314 N FUNCTIONAL EQNS.IN N UNKNOWNS 11-67(726)
C5 315 DAMPED TAYLR SERIES-NONLIN.SYS. 11-67(726)
C5 316 NON-LINEAR SYSTEM 11-67(728)

C6 SUMMATION OF SERIES, CONVERGENCE ACCELERATION
C6 EPSILON ALG.-CONTINUED FRACTNS CHIFFRES V9(327)

D1 QUADRATURE
D1 279 CHEBYSHEV QUADRATURE 4-66(270),6-66(434),
D1 279 5-67(294),10-67(666)
D1 281 ROMBERG QUADRATURE COEFFICENTS 4-66(271),3-67(188)
D1 303 ADAPTIVE QUAD.-RANDM PANEL SIZE 6-67(373)
D1 QUADRATURE BY EXTRAPOLATION NUM.MATH.V9(274)

E2 CURVE AND SURFACE FITTING
E2 28 LEAST SQUARES BY ORTHOG. POLYN. 11-60(604),
E2 28 12-61(544),5-67(293)
E2 91 CHEBYSHEV FIT 5-62(281),4-63(167),
E2 91 5-64(290),12-67(803)
E2 295 EXPONENTIAL CURVE FIT 2-67(87)
E2 296 LEAST SQ.FIT-ORTHOG.POLYS. 2-67(87),6-67(377)
E2 318 CHEBYSHEV CURVE FIT(REVISED) 12-67(801)

E4 MINIMIZING OR MAXIMIZING A FUNCTION
E4 315 MINIMIZING SUM OF SQUARES 11-67(726)
E4 FIBONACCI SEARCH COMP.BULL.V8(147),
E4 V9(105),COMP.J.V9(414),V9(416)
E4 MIN.OF UNIMODAL FCN.OF 1 VAR. COMP.BULL.V9(104),
E4 COMP.J.V9(414)
E4 MINIMIZING ARG.OF UNIMODAL FCN. COMP.J.V9(415)
E4 MIN.OF A UNIMODAL FCN.OF 1 VAR. COMP.J.V9(415)

F1 MATRIX OPERATIONS, INCLUDING INVERSION
F1 298 SQ.RT.OF A POS.DEFINITE MATRIX 3-67(182)
F1 SMITH NORMAL FORM BIT 1967(163)
F1 PERMUTATIONS OF ROWS AND COLS. COMP.J.V10(206)

F2 EIGENVALUES AND EIGENVECTORS OF MATRICES
F2 253 SYMMETRIC QR-EIGENVALUES 4-65(217),6-67(376)
F2 254 SYMMETRIC QR-EIVALUES,EIVECTORS 4-65(218),6-67(376)
F2 297 SYM.SYS.(A-LAM*B)X,EIVALS-VECS. 3-67(181)
F2 EIGENVECTORS OF BAND MATRICES NUM.MATH.V9(285)
F2 EIVALUES OF SYMM.TRIDIAG.MATRIX NUM.MATH.V9(388)

F4 SIMULTANEOUS LINEAR EQUATIONS
F4 SYMM.AND UNSYMM.BAND EQUATIONS NUM.MATH.V9(285)

G1 SIMPLE CALCULATIONS ON STATISTICAL DATA
G1 TAIL AREA PROB.FOR 2X2 TABLE COMP.BULL.V9(56),
G1 COMP.J.V9(212),V9(416)

G5 RANDOM NUMBER GENERATORS
G5 294 UNIFORM RANDOM 1-67(40)

G6 PERMUTATIONS AND COMBINATIONS
G6 87 PERMUTATION GENERATOR 4-62(209),8-62(440),
G6 87 10-62(514),7-67(452)
G6 102 PERMUTATIONS IN LEXIC. ORDER 6-62(346),
G6 102 10-62(514), 7-67(452)
G6 130 PERMUTE 11-62(551),7-67(452)
G6 202 PERMUTATIONS 9-63(517),9-65(556),
G6 202 7-67(452)
G6 306 PERMUTATIONS WITH REPETITIONS 7-67(450)
G6 308 PERMUT.IN PSEUDOLEXIC.ORDER 7-67(452)
G6 317 PERMUTATION 11-67(729)

H OPERATIONS RESEARCH, GRAPH STRUCTURES
H 258 TRANSPORT 6-65(381),7-65(445),
H 258 7-67(453)
H 285 MUTUAL PRIMAL-DUAL METHOD 5-66(326),7-67(453)
H 293 TRANSPORTATION PROBLEM 12-66(869),7-67(453)

K2 RELOCATION
K2 302 TRANSPOSE VECTOR STORED ARRAY 5-67(292)

S APPROXIMATION OF SPECIAL FUNCTIONS...
S FUNCTIONS ARE CLASSIFIED S01 TO S22, FOLLOWING
S FLETCHER-MILLER-ROSENHEAD, INDEX OF MATH. TABLES
S13 SIN INTEGRAL SI(X) NUM.MATH.V9(381)
S13 COS INTEGRAL CI(X) NUM.MATH.V9(382)
S14 179 BETA RATIO 6-63(314),6-67(375)
S14 309 GAMMA FCN.-ARBITRARY PRECISION 8-67(511)
S15 123 REAL ERROR FUNCTION, ERF(X) 9-62(483),6-63(316),
S15 123 10-63(618),3-64(145),6-67(377)
S15 180 ERROR FUNCTION-LARGE X 6-63(314),6-67(377)
S15 181 COMPLEMENTARY ERR.FCN.-LARGE X 6-63(315),
S15 181 12-64(702), 6-67(377)
S15 209 GAUSS 10-63(616),
S15 209 3-64(143), 8-64(482),6-67(377)
S15 226 NORMAL DISTRIBUTION FUNCTION 5-64(295),6-67(377)
S15 272 NORMAL DISTRIBUTION FUNCTION 12-65(789),6-67(377)
S15 299 CHI-SQUARED INTEGRAL 4-67(243)
S15 304 NORMAL CURVE INTEGRAL 6-67(374),6-67(377)
S15 NORMAL DISTRIBUTION CURVE COMP.J.V9(322),
S15 V10(113)
S20 301 AIRY FUNCTIONS 5-67(291),7-67(453)
S20 FRESNEL INTEGRALS S(X),C(X) NUM.MATH.V9(382)
S22 300 COULOMB WAVE FUNCTIONS 4-67(244)


*Key*—1st column: A1, B1, B3, etc. is the key to the underlined Modified Share Classification heading each group of algorithms; 2d column: number of the algorithm in *CACM*; 3d column: title of algorithm; 4th column: month, year and page (in parens) in *CACM*, or reference elsewhere. This index supplements the previously published indexes: Index by Subject to Algorithms: 1960–1963 [*CACM 7* (Mar. 1964), 146–149]; 1964 [*CACM 7* (Dec. 1964), 703]; 1965 [*CACM 8* (Dec. 1965), 791]; and 1966 [*CACM 9* (Dec. 1966), 872].

```
      rh[i1] := (rhi1−rhi)/denom;
      i1 := i; ai1 := ai; rhi1 := rhi
    end i
  end j;
  comment equate (m+1)th difference to zero to determine h;
  h := −a[mplus1]/rh[mplus1];
  comment with h known, combine the function and deviation
    differences;
  for i := 0 step 1 until mplus1 do
    a[i] := a[i] + rh[i] × h;
  comment compute polynomial coefficients;
  for j := m − 1 step −1 until 0 do
  begin
    xj := rx[j]; i := j; ai := a[i];
    for i1 := j + 1 step 1 until m do
    begin
      ai1 := a[i1];
      a[i] := ai − xj × ai1;
      ai := ai1; i := i1
    end i1
  end j;
  comment if the reference deviation is not increasing mono-
    tonically then exit;
  hmax := abs(h);
  if hmax ≤ prevh then
  begin a[mplus1] := −hmax; go to fit end;
  comment find the index, imax, and value, himax, of the largest
    absolute error for all sample points;
  a[mplus1] := prevh := hmax; imax := r[0]; himax:= h;
  j := 0; rj := r[j];
  for i := 1 step 1 until n do
    if i ≠ rj then
  begin
    xi := x[i]; hi := a[m];
    for k := m − 1 step −1 until 0 do
    hi := hi × xi + a[k];
    hi := hi − y[i]; abshi := abs(hi);
    if abshi > hmax then
    begin hmax := abshi; himax := hi; imax := i end
  end
  else
  if j < mplus1 then
  begin j := j + 1; rj := r[j] end;
  comment if the maximum error occurs at a nonreference
    point, exchange this point with the nearest reference point
    having an error of the same sign and repeat;
  if imax ≠ r[0] then
  begin
    for i := 0 step 1 until mplus1 do
    if imax < r[i] then go to swap;
    i := mplus1;
swap: nexthi := if i − i ÷ 2 × 2 = 0 then h else −h;
    if himax × nexthi ≥ 0 then r[i] := imax
    else
    if imax < r[0] then
    begin
      j1 := mplus1;
      for j := m step −1 until 0 do
      begin r[j1] := r[j]; j1 := j end;
      r[0] := imax
    end
    else
    if imax > r[mplus1] then
    begin
      j := 0;
      for j1 := 1 step 1 until mplus1 do
      begin r[j] := r[j1]; j := j1 end;
      r[mplus1] := imax
    end
    else r[i−1] := imax;
    go to start
  end;
fit:
end chebfit
```

## CERTIFICATION OF ALGORITHM 91 [E2] CHEBYSHEV CURVE-FIT [Albert Newhouse *Comm. ACM 5* (May 1962), 281; *6* (April 1963), 167; *7* (May 1964), 296]

J. Boothroyd (Recd. 15 May 1967 and 5 Sept. 1967)
University of Tasmania, Hobart, Tasmania, Australia.

In addition to the corrections noted by R. P. Hale [op. cit., April 1963] and P. Naur [op. cit., May 1964], the following changes are necessary:

1. The first statement should be $k := entier((m-1)/(n+1))$
2. A semi-colon should precede label *L*1.

With these changes the procedure ran successfully using Elliott 503 Algol.

Although this procedure is an implementation of a finite algorithm, roundoff errors may give rise to cyclic changes of the reference set causing the procedure to fail to terminate.

Algorithm 318 [J. Boothroyd, Chebyshev Curve-Fit(Revised), *Comm.ACM 10* (Dec. 1967), 801] avoids this cycling difficulty, uses less than half the auxiliary array space of Algorithm 91 and, on test, appears to be at least four times as fast.

## Deutsch and Lampson—Cont'd from p. 799

be consciously aware of them. Many small details throughout the system have been arranged to permit smooth and rapid operation.

*Acknowledgment.* The basic framework of QED was designed and the program written by Mr. Deutsch. Many people have contributed suggestions for its improvement; we are especially indebted to M. W. Pirtle and R. Morris in this respect.

Received August, 1966; revised August, 1967

### REFERENCES

1. Crisman, P. A. (Ed.). *The Compatible Time-Sharing System: A Programmer's Guide*, 2nd ed. MIT Press, Cambridge, Mass., 1965, Section AH.9.01.
2. Aranda, S. M. Q-32 Time-sharing system user's guide, executive service: context editing (EDTXT). SDC-TM-2708/204/00, Sys. Devel. Corp., Santa Monica, Calif., Mar. 1966.
3. Lampson, B. W., Lichtenberger, W. W., and Pirtle, M. W. A user machine in a time-sharing system. *Proc. IEEE 54*, 12 (Dec. 1966), 1766–1774.
4. Angluin, D. C., and Deutsch, L. P. Reference manual: QED Time-sharing editor. Project Genie Doc. R-15, U. of California, Berkeley, Calif., Jan. 1967.
5. Farber, D. J., Griswold, R. E., and Polonsky, I. P. The SNOBOL3 programming language. *Bell Sys. Tech. J. 45*, 6 (July 1966), 845–944.
6. Murphy, D. TECO. Memorandum, Bolt, Beranek and Newman, Cambridge, Mass. (Nov. 1966).