

Algorithms Supplement

Previously published algorithms

The following Algorithms have been published in the *Communications of the Association for Computing Machinery* during the period October–December 1967.

312 ABSOLUTE VALUE AND SQUARE ROOT OF A COMPLEX NUMBER

313 MULTI-DIMENSIONAL PARTITION GENERATOR

314 FINDING A SOLUTION OF N FUNCTIONAL EQUATIONS IN N UNKNOWNs

Performs inverse interpolation in n dimensions, i.e. finds a set of values for n variables x_1, \dots, x_n such that n functions $f_i(x_1, \dots, x_n)$ are zero.

315 THE DAMPED TAYLOR'S SERIES METHOD FOR MINIMISING A SUM OF SQUARES AND FOR SOLVING SYSTEMS OF NONLINEAR EQUATIONS.

Finds the minimum of the sum of the squares of a set of m functions $f_i(x_1, x_2, \dots, x_n)$ ($m \geq n$).

316 SOLUTION OF SIMULTANEOUS NON-LINEAR EQUATIONS

Solves a system of n simultaneous non-linear equations using a quadratically convergent method.

317 PERMUTATION

Produces the k th permutation on n variables.

318 CHEBYSCHEV CURVE-FIT (REVISED)

Evaluates the coefficients of an m th order polynomial $P(x) = a_0 + a_1x + \dots + a_mx^m$ such that the maximum error $\text{abs}(P(x_i) - y_i)$ is a minimum over the n ($n > m+1$) sample points $(x_1, y_1), \dots, (x_n, y_n)$. This procedure is an extensive revision of Algorithm 91.

The following Algorithm has been published in *Nordisk Tidskrift for Informationsbehandling* in the July 1967 issue.

Contribution No. 21

ZEROS OF POLYNOMIALS

Algorithms

Algorithm 32.

CALCULATION OF EIGENVALUES OF REAL MATRICES BY THE QR METHOD USING DOUBLE QR STEP

J. Grad, K. A. Redish, M. A. Brebner,
Computer Services,
University of Birmingham,
Birmingham, 15.

Authors' Note:

The algorithm uses the QR method with double QR step (Wilkinson, 1965) to find the eigenvalues of a real matrix of

order n . This method is more complicated than the general QR algorithm used by the authors of algorithms (Businger, 1965; Ruhe, 1966), but it is particularly economical with regard to space and time for real non-symmetric matrices. It has the advantage that even if some of the eigenvalues are complex the use of complex arithmetic is avoided without additional calculations and storage requirement.

The general matrix A is reduced to an upper-Hessenberg form H by means of similarity transformations (Wilkinson, 1965). The QR iterative process using double QR step is performed on the Hessenberg matrix until all elements of the sub-diagonal that converge to zero are in modulus less than $2^{-t}||H||_E$, where t is the number of significant digits in the mantissa (for KDF9 computer with $t = 39$ this is equal to $1.82 \times 10^{-12}||H||_E$). This bound is defined in the subroutine by the statement

```
E = SQRT(X) * 1.82E-12.
```

After calling the subroutine the value in the store $A(1,1)$ indicates whether or not the convergence of the QR process has been achieved. If $A(1,1)$ is equal to 0.0, the process has converged. Otherwise $A(1,1)$ is equal to 1.0 and the message that no convergence has been obtained after $n*10$ iteration steps of the QR process is output on unit IUNIT. This bound is defined by the statement

```
MAXST = N*10.
```

This version of the subroutine does not include any detection or correction for the accumulation of rounding errors due to ill-conditioned matrices.

References

- BUSINGER, P. A. (1965). Eigenvalues and Eigenvectors of a Real Symmetric Matrix by the QR Method, Algorithm 254, *Communications of the Association for Computing Machinery*, Vol. 8, No. 4, p. 218.
FRANCIS, J. G. F. (1961). The QR Transformation, *The Computer Journal*, Vol. 4, No. 3, p. 265.
RUHE, A. (1966). Eigenvalues of a complex matrix by the QR-method, *Nordisk Tidskrift for Informationsbehandling*, Bind 6, Hefte Nr. 4, pp. 350–8.
WILKINSON, J. H. (1965). *The Algebraic Eigenvalue Problem*. Oxford: Clarendon Press, pp. 347–351, 528–537.

```
SUBROUTINE EIGEN (N, NM, A, EVR, EVI, IUNIT)
DIMENSION A (NM, 1), EVR(NM), EVI(NM)
C THIS SUBROUTINE FINDS ALL THE EIGEN-
C VALUES OF A REAL NON-SYMMETRIC
C OR SYMMETRIC MATRIX A.
C
C N IS THE ORDER OF THE MATRIX.
C NM DEFINES THE FIRST DIMENSION OF THE
C TWO DIMENSIONAL ARRAY A AND THE
C DIMENSION OF ONE DIMENSIONAL ARRAYS
C EVR, EVI.
C THEREFORE IN THE CALLING ROUTINE THERE
C SHOULD BE DIMENSION A(NM,M), EVR(NM),
C EVI(NM)
C WHERE NM AND M ARE EQUAL TO OR GREATER
C THAN N.
```

```

C THE UPPER LIMIT FOR NM DEPENDS ON THE
C SIZE OF STORE.
C THE ELEMENTS OF THE MATRIX ARE TO BE
C STORED IN THE FIRST N ROWS AND
C COLUMNS OF THE ARRAY A. THE ORIGINAL
C MATRIX IS DESTROYED BY THE ROUTINE.
C THE REAL COMPONENTS OF THE N EIGEN-
C VALUES WILL BE FOUND IN FIRST N PLACES
C OF EVR, AND THE IMAGINARY COMPONENTS
C IN THE FIRST N PLACES OF EVI.
C
C
C
C START THE REDUCTION OF MATRIX A TO
C UPPER HESSENBERG FORM BY MEANS OF
C SIMILARITY TRANSFORMATIONS (HOUSE-
C HOLDER METHOD).
C
IF (N .LT .3) GOTO 9
M = N - 2
C THERE ARE N-2 STEPS.
DO 8 K = 1, M
L = K + 1
SR2 = 0.0
DO 31 I = L, N
31 SR2 = SR2 + A(I, K) ** 2
IF (SR2 .NE. 0.0) GOTO 27
EVR(K) = 0.0
GOTO 8
C PREMULTIPLICATION.
27 SR = SQRT(SR2)
IF (A(L, K) .LT. 0.0) GOTO 33
SR = - SR
33 SR2 = SR2 - SR * A(L, K)
A(L, K) = A(L, K) - SR
EVR(K) = SR
DO 34 I = L, N
34 EVR(I) = A(I, K) / SR2
DO 36 J = L, N
SR = 0.0
DO 35 I = L, N
35 SR = SR + A(I, K) * A(I, J)
DO 36 I = L, N
36 A(I,J) = A(I, J) - EVR(I) * SR
C POSTMULTIPLICATION.
DO 38 J = 1, N
SR = 0.0
DO 37 I = L, N
37 SR = SR + A(J, I) * A(I, K)
DO 38 I = L, N
38 A(J, I) = A(J, I) - EVR(I) * SR
8 CONTINUE
DO 32 K = 1, M
32 A(K + 1, K) = EVR(K)
C
C
C QR ITERATIVE PROCESS. REDUCTION OF UPPER
C HESSENBERG MATRIX TO AN UPPER MODIFIED
C TRIANGULAR FORM.
C
C DETERMINATION OF THE SHIFT OF ORIGIN
C FOR FIRST STEP OF THE QR PROCESS IF THE
C ELEMENTS A(N, N), A(N - 1, N - 1), A(N - 1, N)
C ARE EQUAL TO ZERO.
9 SHIFT = 0.0
MAXST = N * 10
IF (A(N, N) .NE. 0.0) GOTO 1
IF (N .LE. 2) GOTO 1
IF (A(N - 1, N - 1) .NE. 0.0) GOTO 1
IF (A(N - 1, N) .NE. 0.0) GOTO 1
SHIFT = A(N, N - 1)
C
C CALCULATION OF EUCLIDIAN NORM OF THE
C UPPER HESSENBERG MATRIX.
1 X = 0.0
DO 5 K = 1, N
IF (K .NE. 1) GOTO 2
M = 1
GOTO 3
2 M = K - 1
3 DO 5 I = M, N
5 X = X + A(K, I) ** 2
E = SQRT(X) * 1.82E-12
C
M = N
NS = 0
C START MAIN LOOP OF QR TRANSFORMATION
C OF THE UPPER HESSENBERG MATRIX.
10 K = M - 1
M1 = K
I = K
IF (K) 99, 11, 12
C FIND ANY DECOMPOSITIONS OF THE MATRIX.
C JUMP TO 11 IF THE LAST SUBMATRIX OF THE
C DECOMPOSITION IS OF ORDER ONE.
C JUMP TO 13 IF THE LAST SUBMATRIX OF THE
C DECOMPOSITION IS OF ORDER TWO.
12 IF (M - 2 .EQ. 0) GOTO 13
IF (ABS(A(M, K)) .LE. E) GOTO 11
16 I = I - 1
IF (ABS(A(K, I)) .LE. E) GOTO 17
K = I
IF (K .GT. 1) GOTO 16
17 IF (K .EQ. M1) GOTO 13
C START THE TRANSFORMATION FOR MATRIX
C OF ORDER GREATER THAN TWO.
S = A(M, M) + A(M1, M1) + SHIFT
R = A(M, M) * A(M1, M1) - A(M, M1) *
1A(M1, M) + SHIFT ** 2 * 0.25
A(K + 2, K) = 0.0
I = K + 1
C CALCULATE X1, Y1, Z1, FOR SUBMATRIX
C OBTAINED BY THE DECOMPOSITION.
X = A(K, K) * (A(K, K) - S) + A(K, I) * A(I, K) + R
Y = A(I, K) * (A(K, K) + A(I, I) - S)
Z = A(K + 2, I) * A(I, K)
SHIFT = 0.0
NS = NS + 1
C MAIN LOOP FOR ONE STEP OF QR PROCESS.
DO 29 I = K, M1
I1 = I + 1
I2 = I + 2
I3 = I + 3
IF (I .EQ. K) GOTO 18
C CALCULATE XR,YR,ZR.
X = A(I, I - 1)
Y = A(I1, I - 1)
IF (I2 .LE. M) GOTO 19
Z = 0.0

```

```

GOTO 18
19 Z = A(I2, I - 1)
18 S = SQRT(X ** 2 + Y ** 2 + Z ** 2)
  SR = S
  IF (X .LT. 0.0) GOTO 20
  S = - S
20 IF (I .EQ. K) GOTO 21
  A(I, I - 1) = S
21 IF (SR .GE. E) GOTO 30
  IF (I3 .GT. M) GOTO 29
  GOTO 28
30 AL = 1.0 - X/S
  S = X - S
  X = Y / S
  Y = Z / S

```

C PREMULTIPLICATION.

```

DO 23 J = I, M
  S = A(I, J) + A(I1, J) * X
  IF (I2 .GT. M) GOTO 22
  S = S + A(I2, J) * Y
22 S = S * AL
  A(I, J) = A(I, J) - S
  A(I1, J) = A(I1, J) - S * X
  IF (I2 .GT. M) GOTO 23
  A(I2, J) = A(I2, J) - S * Y
23 CONTINUE

```

C POSTMULTIPLICATION.

```

L = I2
IF (I .LT. M1) GOTO 24
L = M
24 DO 26 J = K, L
  S = A(J, I) + A(J, I1) * X
  IF (I2 .GT. M) GOTO 25
  S = S + A(J, I2) * Y
25 S = S * AL
  A(J, I) = A(J, I) - S
  A(J, I1) = A(J, I1) - S * X
  IF (I2 .GT. M) GOTO 26
  A(J, I2) = A(J, I2) - S * Y
26 CONTINUE
  IF (I3 .GT. M) GOTO 29
  S = - A(I3, I2) * Y * AL
28 A(I3, I) = S
  A(I3, I1) = S * X
  A(I3, I2) = S * Y + A(I3, I2)
29 CONTINUE
  IF (NS .GT. MAXST) GOTO 6
  GOTO 10

```

C COMPUTE THE LAST EIGENVALUE.

```

11 EVR(M) = A(M, M)
  EVI(M) = 0.0
  M = K
  GOTO 10

```

C COMPUTE THE EIGENVALUES OF THE LAST 2X2

C MATRIX ON THE DIAGONAL.

```

13 R = 0.5 * (A(K, K) + A(M, M))
  S = 0.5 * (A(M, M) - A(K, K))
  S = S * S + A(K, M) * A(M, K)
  IF (S .LT. 0.0) GOTO 14
  S = SQRT(S)
  EVR(K) = R - S
  EVR(M) = R + S
  EVI(K) = 0.0
  EVI(M) = 0.0
15 M = M - 2

```

```

GOTO 10
14 S = SQRT(- S)
  EVR(K) = R
  EVR(M) = R
  EVI(K) = S
  EVI(M) = - S
  GOTO 15
6 WRITE (IUNIT, 7) MAXST
7 FORMAT (48H1 NO CONVERGENCE AFTER
1NUMBER OF QR ITER.STEPS=, I6)
  A(1, 1) = 0.0
  GOTO 100
99 A(1, 1) = 0.0
100 RETURN
END

```

Algorithm 33.

FITTING DATA TO AN EXPONENTIAL WITH A STRAIGHT LINE AS BACKGROUND

H. Späth,
Institut für Neutronenphysik
und Reaktortechnik,
Kernforschungszentrum Karlsruhe,
Germany.

Author's Note:

The Algorithm finds the best fit, in the least squares sense, of the set of data points $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ with associated weights p_1, p_2, \dots, p_n to the curve

$$y = f(x) = a + bx + ce^{-dx}.$$

In this method the problem reduces to that of finding values a, b, c, d such that

$$s(a, b, c, d) = \sum_{i=1}^n p_i (y_i - f(x_i))^2$$

is a minimum.

From the necessary conditions for this

$$\frac{\partial s}{\partial a} = \frac{\partial s}{\partial b} = \frac{\partial s}{\partial c} = \frac{\partial s}{\partial d} = 0$$

$a = a(d)$, $b = b(d)$ and $c = c(d)$ are eliminated by substitution into $\frac{\partial s}{\partial d} = 0$. This results in a single equation in one variable

$$F(d) = \frac{\partial s}{\partial d}(a(d), b(d), c(d), d) = 0.$$

If a value d' is obtained with $F(d') = 0$ then the corresponding values for a, b and c are obtained from $a' = a(d')$, $b' = b(d')$ and $c' = c(d')$. By tabulating F and the corresponding value of s as a function of d this method enables one to find all the stationary points of s and, if it exists, the desired absolute minimum.

This method is basically the same as the one used for $g(x) = a + be^{-cx}$ (Späth, 1967), but as the fit to the model f is often needed in chemistry and radiation chemistry, i.e. within reactions which show no saturation so long as material exists, it seems worthwhile to give the corresponding algorithm explicitly. The algorithm could be used to fit data of radiated Thymine-6-T to a kinetic model (Heitkamp *et al.*, 1968).

[The author wishes to acknowledge valuable suggestions for improvement made by the referee.]

References

- SPÄTH, H. (1967). Exponential Curve Fit, Algorithm 295, *Communications of the Association for Computing Machinery*, Vol. 10, No. 2, p. 87.
- HEITKAMP, D., MERWITZ, O., and SPÄTH, H. (1968). Zur Kinetik der strahleninduzierten Wasserstoffabspaltung aus Thymin, *Zeitschrift für Naturforschung*, Serie B. (In Print.)
- ```

procedure ABCD(x, y, p, n, Root, ad, ed, eps, a, b, c, d, s,
fx, ex); value n, ad, ed, eps; integer n; real ad, ed, eps, a, b,
c, d, s; array x, y, p, fx; label ex; procedure Root;
comment The procedure finds a quadruple (a, b, c, d) which
is a stationary point of s. The formal procedure Root (e.g. a
bisection method) must find a zero, d, of a function Fofd = F(d)
in the interval [ad, ed] with relative accuracy eps. If sign
(F(ad)) = sign (F(ed)) the procedure should exit to label ex.
This label is further used when n < 4. The array fx finally
contains the values f(xi);
begin integer i; real t, u, v, w, h1, h2, h3, h4, h5, h6, h7, h8,
h9, h10, h11, h12, h13, h14, h15, h16, p1, p2, p3, p4, p5,
p6, det;
procedure Fofd(d, fd); value d; real d, fd;
begin h1 := h2 := h3 := h4 := h5 := h6 := h7 :=
h8 := h9 := h10 := h11 := h12 := h13 := 0;
for i := 1 step 1 until n do
begin v := x[i]; u := exp (- d × v);
w := p[i]; t := y[i];
h10 := w × v; h14 := v × u;
h15 := u × w; h16 := w × t;
h1 := h1 + w; h2 := h2 + h10;
h3 := h3 + h15; h4 := h4 + h10 × v;
h5 := h5 + h10 × u; h6 := h6 + h15 × u;
h7 := h7 + h16; h8 := h8 + h10 × t;
h9 := h9 + h16 × u; h11 := h11 + h10 × h14;
h12 := h12 + h14 × h15; h13 := h13 + h14 × h16
end;
p1 := h4 × h6 - h5 × h5; p2 := h6 × h8 - h5 × h9;
p3 := h5 × h8 - h4 × h9; p4 := h5 × h3 - h2 × h6;
p5 := h2 × h9 - h3 × h8; p6 := h2 × h5 - h4 × h3;
det := 1.0 / (h1 × p1 + h2 × p4 + h3 × p6);
a := det × (h7 × p1 - h2 × p2 + h3 × p3);
b := det × (h1 × p2 + h7 × p4 + h3 × p5);
c := det × (h7 × p6 - h1 × p3 - h2 × p5);
fd := a × h5 + b × h11 + c × h12 - h13
end Fofd;

```

```

if n < 4 then goto ex;
Root (Fofd, ad, ed, eps, d, ex); t := 0;
for i := 1 step 1 until n do
begin v := fx[i] := a + b × x[i] + c × exp (- d × x[i]);
v := v - y[i]; t := t + p[i] × v × v
end;
s := t
end ABCD

```

## Special Note on Algorithm 31

## COMPLEX FOURIER ANALYSIS

J. Boothroyd,  
Hydro-University Computing Centre,  
University of Tasmania.

The editor regrets a typographical error in this algorithm.  
The last three lines of page 415 should read

```

comment main program;
wtk := nn := if n < 0 then -n else n;
primefactors (nn, b, nb);

```

## Applied Statistics Algorithm Section

The Royal Statistical Society in co-operation with the Science Research Council's Working Party on Statistical Computing intends to publish, in *Applied Statistics*, algorithms relevant to statistics. Algorithms written in ALGOL-60, ASA Standard FORTRAN, or PL/1 accompanied by an external specification may be sent to

The Editor, Applied Statistics  
Royal Statistical Society  
21 Bentinck Street  
LONDON W.1  
England

A policy statement describing the editorial policy appears in *Applied Statistics*, Vol. 17, No. 1 (1968). A support paper describing the expected contents of the external specification, and making recommendations for the layout of algorithms and for programming strategy will appear in the following issue.

Contributions for the Algorithms Supplement should be sent to

P. Hammersley  
University Mathematical Laboratory  
Corn Exchange Street  
Cambridge