ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
XIII, 2 (1972)

ALGORITHM 19

A. RAKUS (Warszawa)

CALCULATION OF ALL EIGENVALUES OF A REAL MATRIX

1. Procedure declaration. The procedure QR finds all (also complex) eigenvalues of real matrices.

Data:

n — dimension of matrix a,
$a[1:n, 1:n]$ — matrix whose eigenvalues are to be calculated,
p — number of significant bits representing the mantissa in actual implementation of Algol.

Results:

On exit from procedure QR the real parts of the eigenvalues are placed in the first column of a ($a[i,1]$ for $i = 1, 2, \ldots, n$) and the imaginary parts of the eigenvalues are placed in the second column of that array ($a[i,2]$ for $i = 1, 2, \ldots, n$). All initial elements of a are destroyed.

2. Method used. This procedure is written on the basis of Francis' iterative QR method (1961) [1]:

$$\begin{array}{rlll} A = A_1 &= Q_1 R_1 & R_1 Q_1 &= A_2 \\ A_2 &= Q_2 R_2 & R_2 Q_2 &= A_3 \\ \cdot & \cdot\ \cdot & \cdot\ \cdot\ \cdot & \\ \cdot & \cdot\ \cdot & \cdot\ \cdot\ \cdot & \\ A_s &= Q_s R_s & R_s Q_s &= A_{s+1}, \end{array}$$

where Q_k are orthogonal matrices, and R_k are upper triangle matrices.

After entry to procedure QR the matrix A is converted to the form of the upper triangular matrix of Hessenberg and practically the next iterations are processed on matrices of that form.

The transformation of matrix A to the form of the upper triangular matrix of Hessenberg and further iterations in this procedure are based on Householder's method.

```
procedure QR(n,a,p);
 value n;
 integer n,p;
 array a;
 begin
  Boolean ghes;
  integer v,q,k,m,r,u,t,w;
  real s,z,eps,eps1,p1,p2,p3,r1,r2,r3;
  array c[1:n];
  eps:=eps1:=0;
  for k:=1 step 1 until n do
   begin
    z:=0;
    for m:=1 step 1 until n do
     begin
      s:=abs(a[k,m]);
      eps1:=eps1+s×s;
      z:=z+s
     end m;
    if z>eps
     then eps:=z
   end k;
  z:=2↑(-p);
  eps:=eps×z;
  eps1:=sqrt(eps1)×z;
  ghes:=false;
  q:=t:=n;
  u:=w:=1;
IT:
  for v:=w step 1 until n-2 do
```

```
begin
 r:=v+1;
 if ghes
  then
  begin
   q:=if v=n-2 then n else v+3;
   if v<n-3
    then
    begin
     t:=v+4;
     a[t,r]:=a[t,r+1]:=.0
    end v<n-3
    else t:=n
  end ghes;
 if ¬(v=w∧ghes)
  then
  begin
   z:=.0;
   for k:=q step -1 until r do
    begin
     r1:=c[k]:=a[k,v];
     z:=z+r1×r1
    end k
  end ¬(v=w∧ghes);
 s:=sqrt(z);
 if ¬(s>eps∨ghes)
  then go to dv;
 if r1<.0
  then s:=-s:
```

```
    r1:=1.0/(r1+s);
z:=1.0;
for k:=r+1 step 1 until q do
 begin
  r2:=c[k]:=c[k]×r1;
  z:=z+r2×r2
 end k;
z:=2.0/z;
for m:=r step 1 until n do
 begin
  r1:=a[r,m];
  for k:=r+1 step 1 until q do
   r1:=r1+c[k]×a[k,m];
  r1:=r1×z;
  a[r,m]:=a[r,m]-r1;
  for k:=r+1 step 1 until q do
   a[k,m]:=a[k,m]-c[k]×r1
 end m;
for m:=u step 1 until t do
 begin
  r1:=a[m,r];
  for k:=r+1 step 1 until q do
   r1:=r1+a[m,k]×c[k];
  r1:=r1×z;
  a[m,r]:=a[m,r]-r1;
  for k:=r+1 step 1 until q do
   a[m,k]:=a[m,k]-c[k]×r1
 end m;
if ¬(v=w∧ghes)
 then a[r,v]:=-s;
```

```
dv:end v;
  ghes:=true;
dz:
  w:=0;
  k:=n-1;
dy:
  if k=w
   then go to dw;
  if abs(a[k+1,k])>eps
   then
   begin
    k:=k-1;
    go to dy
   end abs(a[k+1,k])>eps;
  w:=k;
dw:
  v:=n-w;
  if v=1
   then go to d1;
  k:=n-1;
  r1:=a[k,k];
  r2:=a[n,n];
  s:=-r1-r2;
  z:=r1×r2-a[n,k]×a[k,n];
  z:=s×s-4.0×z;
  if z>.0
   then
   begin
    z:=sqrt(z);
    p1:=-.5×(s+z);
```

```
        p2:=-.5×(s-z);
        p3:=.0
      end z>.0
      else
      begin
        z:=sqrt(-z);
        p1:=p2:=-.5×s;
        p3:=.5×z
      end z≤.0;
    if v=2
      then go to d2;
    m:=n-2;
red2:
    u:=m;
    m:=m-1;
    k:=m+2;
    r:=m+3;
    r1:=a[u,u];
    z:=r1-p1-p2;
    s:=a[k,u];
    r3:=p1×p2+p3×p3;
    r1:=r1×z+a[u,k]×s+r3;
    r2:=c[k]:=s×(z+a[k,k]);
    r3:=c[r]:=a[r,k]×s;
    if m=w
      then
defl:
      begin
        z:=r1×r1+r2×r2+r3×r3;
        a[r,u]:=.0;
```

```
w:=m;
go to IT
end m=w;
if abs(r1)<eps
then go to red2
else
if abs(a[u,m]×(abs(r2)+abs(r3))/r1)≤eps1
then go to defl
else go to red2;
d2;
a[n,1]:=p1;
a[n,2]:=p3;
a[k,1]:=p2;
a[k,2]:=-p3;
n:=n-2;
go to d3;
d1:
a[n,1]:=a[n,n];
a[n,2]:=.0;
n:=n-1;
d3:
if n≥1
then go to dz
end QR
```

3. Certification. The algorithm QR has been verified on the Odra 1204 computer. Results of the control calculations are given in the table.

With the matrix given

$$\begin{matrix} 0 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 \end{matrix}$$

one obtains the following results:

TABLE

The eigenvalues exact	Calculated eigenvalues
-1	-0.9999999998
$0.5+0.86602540378\times i$	$0.5+0.8660254039\times i$
$0.5-0.86602540378\times i$	$0.5-0.8660254039\times i$
$-i$	$0.0-1.0000000000\times i$
i	$0.0+1.0000000000\times i$

References

[1] J. G. F. Francis, *The QR transformation, Parts 1 and 2,* Computer J. 4 (1961), p. 256 and p. 332.

CENTRALNE LABORATORIUM GAZOWNICTWA, WARSZAWA

Received on 24. 6. 1970;
revised version on 29. 1. 1972

ALGORYTM 19

A. RAKUS (Warszawa)

OBLICZANIE WSZYSTKICH WARTOŚCI WŁASNYCH MACIERZY RZECZYWISTEJ

STRESZCZENIE

Procedura *QR* wyznacza wszystkie wartości własne macierzy rzeczywistej.

Dane:

n – wymiar macierzy a,
$a[1:n, 1:n]$ – macierz, której wartości własne mają być obliczone,
p – ilość bitów przeznaczonych na mantysę liczby zmiennoprzecinkowej w maszynie, na której przeprowadzamy obliczenia.

Wyniki:

Po zakończeniu obliczeń, wartości własne są umieszczone na miejscach $a[i, 1]$ oraz $a[i, 2]$ ($i = 1, 2, \ldots, n$), gdzie $a[i, 1]$ jest częścią rzeczywistą, natomiast $a[i, 2]$ częścią urojoną i-tej wartości własnej.

Cała macierz a jest zniszczona.

Procedura napisana jest na podstawie metody *QR* (Francis [1]). Algorytm podany w tej pracy sprowadza macierz do postaci górnej Hessenberga i dalsze iteracje przebiegają na macierzy tej postaci.

Wstępne sprowadzenie do postaci górnej Hessenberga, jak i postępowanie iteracyjne, oparte są na metodzie Householdera. Obliczenia kontrolne, wykonane na maszynie cyfrowej Odra 1204, wykazały poprawność procedury.