▼Journal of Research (Science) , Bahauddin Zakariya University, Multan, Pakistan, 2000, Vol. 11, No. 1, pp.33-41 ISSN 1021-1012

SIMULTANEOUS DETERMINATION OF ALL THE ZEROS OF A LINEAR COMBINATION OF LEGENDRE POLYNOMIAL

Nazir Ahmad Mir*, Aman Ullah Khan and Zahida Akram
Centre for Advanced Studies in Pure & Applied Mathematics,
Bahauddin Zakariya University, Multan , Pakistan.

Abstract: An algorithm based on modified improved Ehrlich method is developed which using real arithmetic could find all the zeros of Legendre-, Shifted Legendre- and Doubly Shifted Legendre polynomials, simultaneously, using a three term recurrence relation. The only information required is degree of the polynomial. The algorithm also finds simultaneously all the zeros of a linear combination of these polynomials, requiring information about the number of terms and the coefficients in the linear combination. The same algorithm works for finding simultaneously all the zeros of real polynomials requiring information about its degree and coefficients.

Keywords: *Zeros, Polynomials, Generalised Basis, Simultaneous Methods.*

1. INTRODUCTION

An algorithm[1,2] was designed for finding simultaneously all the zeros of a real as well as complex polynomials, expressed in monomial basis. The two algorithms required information only about degree and coefficients of the polynomial. Six methods belonging to the Durand-Kerner and Ehrlich families[3,4,5,6,7]were compared numerically in the above mentioned papers. It was found that the modified improved Ehrlich method was the best amongst the six methods considered. Consider a real polynomial of degree n

$$a_0 z^n + a_1 z^{n-1} + \ldots + a_{n-1} z + a_n \qquad (1.1)$$

, where $a_i, i = 0,1,\ldots n,$ are real. Occasionally, this explicit representation of the polynomial may not be immediately available. For example, it may be required to find the zeros of a determinant with polynomial entries. In this work, therefore, consideration is given to the ways in which an algorithm for finding the zeros of a real polynomial expressed in the monomial basis could be modified to find the zeros of a polynomial expressed in the ' Legendre basis' and their linear combination.

The linear combination of the Legendre basis polynomial is of the form

$$p_n(z) = \sum_{r=0}^{n} d_r \phi_r(z) \qquad (1.2)$$

where $\phi_r(z)$ is the rth degree Legendre or Shifted Legendre or Doubly Shifted Legendre polynomial and d_r is a real number.

We have tested the algorithm for finding simultaneously all the zeros of a real polynomial over 250 polynomials of varying degrees[1] .

The basis polynomial is discussed in Section 2 whereas, the modified improved Ehrlich method is given in algorithm form in Section 3. In Section 4, some ways of generating initial estimates considered whilst termination criterion for the implemented algorithm and polynomial evaluation is described in Sections 5 and 6. Sections 7 and 8 contains numerical results for Legendre-, Shifted Legendre- and Doubly Shifted Legendre polynomials and their linear combinations. Finally, Section 9 contains a description of the form of the implemented algorithm, which is given as a FORTRAN procedure in the appendix.

2. THE BASIS POLYNOMIAL

The basis polynomial[8] considered will form polynomials $\{\phi_r(z)\}$, where

* Author to receive correspondence

$\phi_r(z)$ is of exact degree r in $z, r = 0,1,2,...$ The family will be generated by

$$\phi_{-1}(z) = 0, \phi_0(z) = g_{1,0},$$
$$\phi_{k+1}(z) = (z g_{1,k+1} + g_{2,k+1})\phi_k(z) + g_{3,k+1}\phi_{k-1}(z), \quad (2.1)$$
$$k \geq 0, g_{1,k+1} \neq 0.$$

Then $\{\phi_r(z)\}, r = 0,1,...,n,$ form a linearly independent set and hence provides a basis for the representation of any polynomial of degree n.
A number of basis can be generated in this way, but we consider here the following three bases:

i) **The Legender Polynomials**

ii)
$$\phi_r(z) = p_r(z)$$
$$g_{1,0} = 1, g_{1,k} = \frac{2k-1}{k}, g_{2,k} = 0$$
and $g_{3,k} = -\frac{k-1}{k}$ for $k \geq 1$

iii) **The Shifted Legendre Polynomials**

$$\phi_r(z) = p_r^*(z)$$
$$g_{1,0} = 1, g_{1,k} = \frac{4k-2}{k}, g_{2,k} = -\frac{2k-1}{k}$$
and $g_{3,k} = -\frac{k-1}{k}$ for $k \geq 1$

iv) **The Doubly Shifted Polynomial**

$$\phi_r(z) = \tilde{p}_r^*(z)$$
$$g_{1,0} = 1, g_{1,k} = \frac{4(k-2)}{k}, g_{2,k} = -\frac{2k-1}{k}$$
and $g_{3,k} = -\frac{k-1}{k}$ for $k \geq 1$

These basis are the examples of orthogonal basis and of course, any orthogonal basis is generated by a three-term recurrence of the form (2.1). The intervals of orthogonality of Legendre-, Shifted Legendre- and Double Shifted Legendre polynomials are [-1,1], [0,1] and [0,1/2] respectively.

3. ALGORITHM FOR MODIFIED IMPROVED EHRLICH METHOD

We give the algorithm for updating a set of zero estimates $z_1, z_2, \cdots, z_n$ to obtain improved estimates $\tilde{z}_1, \tilde{z}_2, \cdots, \tilde{z}_n$. Within the algorithm, the polynomial is denoted by $p(z)$ and its derivative by $p'(z)$.

We assume some initial ordering of the indices, say (1,2…n) and each stage of algorithm is extended for i running through these values.

(i) $\Delta_i = -p(z_i)/p'(z_i)$

(ii) $z_i^* = z_i + \Delta z_i$

(iii) $\Delta z_i = \Delta_i / \left[1 + \Delta_i \sum_{k=1, k\neq i}^{n} \frac{1}{z_i - z_k^*}\right]$

with $z_k^* = \begin{cases} z_k + \Delta z_k, & k < i \\ z_k + \Delta z_k, & k > i \end{cases}$

(iv) $\tilde{z}_i = z_i + \Delta z_i$

(v) The ordering of the indices is reversed before the next iteration and this alternation continues until convergence.

4. GENERATION OF INITIAL ESTIMATES

Consider the polynomial (1.1) having zeros $z_1, z_2, \cdots, z_n$. Henrici [9], states that these roots all lie inside a circle of radius Beta

$$2 \max_{1 \leq k \leq n} \left| a_k / a_0 \right|^{1/k} \quad (4.1)$$

where Beta is a bound on the largest zero of a polynomial (1.1). It is a well known result [13] that if $p(z)$ has a zero inside the circle $|z| \leq \rho$ $p(\rho z)$ has a zero inside the unit circle. This result is true for both real and complex polynomials. The polynomial (1.1) was therefore scaled using the bound (4.1) to bring all of its zeros into the unit circle. The arbitrary initial estimates for the

zeros of the polynomial (1.1) were taken to be the points uniformly spaced r ound the unit circle having the centre at the origin, that is, the points $\exp[2\pi(k-1)/n\ i]$ in the complex plane. 0.05 was added to the exponent argument to avoid symmetric distributions which may cause numerical difficulties.

5. CONVERGENCE CRITERION

This is provided by computing a bound on the accumulated round off error in the computed value of the polynomial. When this indicates that the later value can be fully accounted for by rounding error, the iterative process is terminated, as there may be no useful information available from which to determine an improved estimate. Bounds of this type have been given by Adams [10],Peters and Wilkinson [11] and Grant and Hitchins [12]. In practice, they have been found to be extremely reliable and accurate, p articularly, if one or two extra iterations are performed to allow for the conservative nature of the bound. Their major disadvantage is the cost of evaluating them.

6. POLNOMIAL EVALUATION

As ultimately, we are getting polynomials with real coefficients only, they and their derivatives can be evaluated at $\alpha + i\beta$ by executing the algorithm

$$p = -2\alpha,\ q = \alpha^2 + \beta^2,$$

$$b_0 = a_0, b_1 = a_1 - p b_0,$$

$$c_0 = b_0, c_1 = b_1 - p c_0,$$

$$b_k = a_k - p b_{k-1} - q b_{k-2},\ k = 2(1)n - 1,$$

$$c_k = b_k - p c_{k-1} - q c_{k-2},\ k = 2(1)n - 3,$$

$$b_n = a_n + b_{n-1} - q b_{n-2}$$

$$c_{n-2} = b_{n-2} - q c_{n-4}$$

when $p_n(\alpha + i\beta) = b_n + i\beta b_{n-1}$

and $p'_n(\alpha + i\beta) = (-2\beta^2 c_{n-3} + b_{n-1}) + i\,2\beta(\alpha c_{n-3} + c_{n-2})$.

Following Adams[10], abound on the error can be found using

$$e_0 = 0.8|b_0|,$$

$$e_k = \sqrt{q}\, e_{k-1} + |b_k|,\ k = 1(1)n,$$

when the process is terminated if

$$|p_n(\alpha + i\beta)| \le (2|\alpha b_{n-1}| - 8(|b_n| + \sqrt{q}|b_{n-1}|) + 10e_n)\rho$$

which is a machine constant, the smallest number which when added to one produces a change. In the numerical result ρ was taken to be 2^{-53}.

7. NUMERICAL RESULTS

Some results of Legendre Polynomials of varying degrees are discussed here.

7.1 Legendre Polynomials

The zeros of Legendre polynomial are real and always lie in the interval [-1,1]. Estimated zeros of Legendre Polynomials of degree 15, using implemented algorithm are given below:

POLYNOMIAL OF DEGREE 15

Real Part	imaginary Part
0.9879925180E+00	0.7177876729E-38
0.8482065834E+00	-0.5262173930E-37
0.5709721726E+00	0.0000000000E+00
0.2011940940E+00	0.0000000000E+00
-0.3941513471E+00	0.0000000000E+00
-0.7244177314E+00	0.0000000000E+00
-0.9372733924E+00	0.0000000000E+00
-0.9879925180E+00	0.2888047299E-36
-0.8482065834E+00	0.0000000000E+00
-0.5709721726E+00	0.0000000000E+00
-0.2011940940E+00	0.0000000000E+00
0.3941513471e+00	0.0000000000E+00
0.7244177314E+00	-0.1128474577E-35
0.9372733924E+00	0.0000000000E+00
0.0000000000E+00	0.0000000000E+00

We observe the eleven zeros are correct to the last decimal place and the largest error in the remaining zeros is -0.1×10^{-35}.

7.2 Shifted Legendre Polynomials

The zeros of Shifted Legendre polynomial are real and always lie in the interval [0,1]. Estimated zeros of Shifted Legendre Polynomials of degree 15, using implemented algorithm are given below:

POLYNOMIAL OF DEGREE 15

Real Part	imaginary Part
0.9939962629E+00	-0.161587134E-26
0.9241033056E+00	-0.5835834238E-31
0.7854860889E+00	-0.5547422732E-27
0.6055970472E+00	-0.3118639581E-27
0.3994029530E+00	-0.1724419894E-28
0.2145139137E+00	0.2310514080E-31
0.7589670829E-01	0.1880790961E-36
0.6003740990E-02	0.0000000000E+00
0.3136330380E-01	0.0000000000E+00
0.1377911343E+00	0.0000000000E+00
0.3029243265E+00	-0.1764593345E-33
0.50000000000E+00	0.0000000000E+00
0.6970756723E+00	0.1371926612E-24
0.8622088582E+00	0.1461664596E-25
0.9686366869E+00	-0.5844907863E-32

7.3 Doubly Shifted Legendre Polynomials

The zeros of Doubly Shifted Legendre polynomial are real and always lie in the interval [0,1/2]. Estimated zeros of Shifted Legendre Polynomials of degree 15, using implemented algorithm are given below:

POLYNOMIAL OF DEGREE 15

Real Part	imaginary Part
0.4969981313E+00	-0.1222581212E-21
0.4620516524E+00	0.2626612176E+18
0.3927430450E+00	-0.4086428332E-19
0.3002985237E+00	0.5125362133E-21
0.1997014765E+00	-0.1802783196E-21
0.1072569568E+00	0.8028836967E-24
0.3794835415E-01	-0.3559734835E-27
0.3001870495E-02	-0.1194076566E-31
0.1568165190E-01	-0.5926454613E-17
0.6889556716E-01	-0.1625053465E-27
0.1514621632E+00	0.3283302196E-27
0.2500000000E+00	0.0000000000E+00
0.3485378360E+00	0.4114130118E-19
0.4311044290E+00	-0.1692409061E-17
0.4843183407E+00	0.3353233703E-17

We observe that one of the zeros are correct to last decimal place and the largest error in the remaining zeros is -0.5×10^{-17}.

8. Numerical Results for Linear Combination.

Consider (1.2) whose zeros are to be determined. Although the zeros of Legender-, Shifted Legendre- or Doubly Shifted Legendre Polynomials are real and lie in a specific interval, the zeros of their linear combination need not to be real and may be complex. Sum results of these linear combinations of varying number of terms are given below:

8.1 Linear Combination of Legender Polynomial

$$\phi_{10} + 2\phi_9 + 3\phi_8 + 4\phi_7 + 5\phi_6 + 6\phi_5 + 7\phi_4 + 8\phi_3 + 9\phi_2 + 10\phi_1 + 11\phi_0 = 0$$

The computed estimates are:

Real Part	imaginary Part
0.8951466537E+00	0.1824543521E+00
0.4602537989E+00	0.3801561966E+00
-0.1375577237E+00	0.4504238943E+00
-0.7007230756E+00	0.3561127939E+00
-0.1043435443E+01	0.1333524611E+00
-0.1043435443E+01	0.1333524611E+00
-0.7007230756E+00	0.3561127939E+00
-0.1375577237E+00	0.4504238943E+00
0.4602537989E+00	0.3801561966E+00
0.8951466537E+00	0.1824543521E+00

We observe that all the zeros are complex.

8.2 Linear Combination of Shifted Legendre Polynomial

$$3\phi_{10} + 4\phi_9 + 5\phi_8 + 6\phi_7 + 7\phi_6 + 6\phi_5 + 7\phi_4 + 5\phi_3 + 4\phi_2 + 3\phi_1 + 7\phi_0 = 0$$

The computed estimates are

Real Part	imaginary Part
0.9324297345E+00	0.3806003375E-01
0.7457879250E+00	0.1226397926E+00
0.4632757747E+00	0.1577534282E+00
0.1731724819E+00	0.1175716178E+00
0.9895487309E-02	-0.5488715206E-01
0.9895487309E-02	-0.5488715206E-01
0.1731724819E+00	-0.1175716178E+00
0.4632757747E+00	-0.1577534282E+00
0.9324297345E+00	-0.3806003375E-01

All the zeros are complex.

8.3 Linear Combination of Doubly Shifted Legendre Polynomial

$$5\phi_{10} + 6\phi_9 + 7\phi_8 + 8\phi_7 + 9\phi_6 + 7\phi_5 + 8\phi_4 + 6\phi_3 + 5\phi_2 + 4\phi_1 + 3\phi_0 = 0$$

The computed estimates are

Real Part	imaginary Part
0.4667574331E+00	-0.2159165371E-34
0.3661671901E+00	0.3118736756E-01
0.2340185073E+00	0.3648278874E-01
0.9009704126E-01	-0.1154488685E-01
0.1768478607E-01	0.1266778523E-01
0.1768478607E-01	-0.1266778523E-01
0.9009704126E-01	0.1154488685E-01
0.2340185073E+00	-0.3648278874E-01
0.3661671901E+00	-0.3118736756E-01
0.4594127807E+00	-0.2779679007E-33

Here only two zeros are real with the largest error -0.2×10^{-33}.

8. THE IMPLEMENTATION

The FORTRAN subroutine given in the Appendix is a direct implementation of the algorithm given in 3. Double-length real arithmetic is used throughout to allow for a termination criterion based on the use of Adams type error bound given in 6. If required, starting estimates are taken in the form $\exp[2\pi(k-1)/n + 0.05\,i], k = 1(1)n$.

Polynomial is scaled using an upper bound on the largest zero of (1.1). After the zeros of scaled polynomial have been calculated, they are transferred back to give zero estimates for original polynomial evaluation.

'Select' is an integer variable and has value 1 if we want to find zeros of Legendre basis polynomial equation, 2 if zeros of their linear combination equation, and 3 if the zeros of monomial basis polynomial equation.

Similarly, 'Kind ' is also an integer variable and has value 1 for Legendre, 2 for Shifted Legendre, and 3 for Doubly Shifted Legendre polynomial. If 'Select' has value 2, then for any value of ' Kind ', subroutine Lcomb transforms (1.2) into the form (1.1).

Several different modes of entry are possible. Initial estimates may or may not be specified. If they are given as rlz(k) + i cmz (k), k =1(1)n, the character parameter ans should be 'y' or 'Y'. The logical parameter con should be 'true' , if up-dating of an estimate is to cease once it has been detected as having converged; a value 'false' means that up-dating will continue until all estimates are indicated as having converged on the same sweep .

The Legendre Polynomials of odd degree have zero root which cause numerical difficulties. Tgerefore a logical parameter zflag was introduced to separate zero root from the polynomial. This parameter is set to ' false ', if it has zero root, i.e. if constant term of the polynomial is zero.

On exit, the coefficients are unaltered . A successful conclusion is indicated by icode having the value 1, when the computed root estimates are available as rlz(k) + i cmz(k) , k=1(1)n. The integer array itusd gives the number of iterations to convergence for the individual estimates. A value of -1 for icode indicates that not all the roots have converged within the permitted number of iterations ; those that have

not converged being shown by the corresponding itusd entry 0.

There are some error exists from the subroutine indicated by the parameter iex, which normally has value 0. A value of –2 indicates that either the leading coefficient in the polynomial is 0 or the degree is less than 1. A value of-1 indicates that division by the complex number 0.0 + i 0.0 has been attempted within the subroutine. There are ten internally used arrays of fixed length allowing for the solution of polynomials of degrees not greater than 50.

REFERENCES

1. J. A., Grant, and N. A., Mir, "A numerical comparison of methods for finding simultaneously all the zeros of a real polynomial", Report No. 93-36, University of Bradford, U.K., 1993.
2. N.A. Mir and Faisal Ali, "A comparison of methods for finding simultaneously all the zeros of a complex polynomial", *Sci. Int.* (accepted)
3. O. Aberth, "Iteration methods for finding all the zeros of a polynomial simultaneously", *Math. Comp.* ,1973, **27,** 339-344.
4. E., Durand, "Solutions Numeriques des Equations Algebriques", Tome I, Masson, Paris ,1960.
5. L.W.,Ehrilich,"A modified Newton method for Polynomials", *Comm., ACM,* 1967, **10**, 107-108.
6. I. O., Kerner, "Ein Gesamschrittverfahren zur Berechnung der Nullstellen eines Polynoms",*Num.Math*., 1966, **8**, 290-294.
7. A. W. M., Nourein, "An improvement on two iteration methods for simultaneous determination of the zeros of a polynomial", *Intern. J. Computer Math*., **6**, 1977, 241-252.
8. J.A. Grant, and A.A., J.A. Rahman, "Determination of the zeros of a linear combination of generalised polynomials", *Journal of Computational and Applied Mathematics,* 1992, **42**, 269-278.
9. P. Henrici, and B.O Watkins,." Finding zeros of a polynomial by the Q-D algorithm", *Comm. ACM,* 1965,**8,** 570-574.
10. D. A Adams, D. A "A stopping criterion for polynomial root finding", *Comm. ACM,* 1967, **10**, 655-658.
11. G. Peters, and J. H., Wilkinson, "Practical problems arising in the solution of polynomial equations", *J. Inst.Math Appl.,* 1971, **6**, 16-35.
12. J. A. Grant, and G. D Hitchins," Two algorithms for the solution of polynomial equations to limiting machine precision", *Comp. J*., 1975, **18**, 258-264.

APPENDIX

```
c  subroutine miehrl calculates zero
c  estimates using modified improved
c  Ehrlich method
   subroutine miehrl(a,n,rlz,cmz,maxit,
  +itusd,icode,iex,ans,con,kind,
  +zflag,select)



c  attempts to find the zeros of a real
c  monomial basis polynomial,legendre
c  basis polynomial and their linear
c  combination  equations
c  a -double precision one-dimensional
c  array of the coefficients
c  rlz,cmz - double precision array of
c  initial estimates of realandimaginary
c  parts of zeros on entry, computed
c  estimates on exit
c  maxit-maximum number of iterations
c  allowed
c  itusd - i-dimensional integer array
c  giving no. of iterations to
convergence
c  for the individual zero estimates
c  icode - on successful conclusion has
c  value 1, otherwise -1
c  iex - an integer indicating error
c  conditions, -2 if leading
c  coefficient is zero or n<2, -1 for
c  overflow etc
c  ans- character, if 'y' or'Y' then
c  initial estimates are available
c  otherwise they are generated
c  internally
```

```
c  con - logical variable which
c  determines whether estimates are
c  up-dated after convergence or not,
c  true means not
c  zflag-logical variable which
c  separates the zero root from the
c  legendre basis polynomial equation
c  kind -integer,1 if legendre
polynomial
c  ,2 if their linear combination
c  and 3 if monomial basis polynomial

   implicit double precision (a-h,o-z)
   dimension a(n+1),b(51),rdelta(50),
  +cdelta(50),rzstar(50),czstar(50),
  +rp(50),cp(50),rlz(n),cmz(n),iflag(50)
  +,itusd(n),rpd(51),cpd(51),d(50)
   character ans
   logical con,sat,zflag

 if(select.e.2.or.select.e.3)goto8
 if(.not.(ans.eq.'Y'.or.ans.eq.'y'))then
      call legpol(a,n,kind)
 endif

8     if(a(n+1).eq.0.0) then
        n=n-1
        zflag = .true.
      endif

      iex=0
cleading coeff zero or n<2force exit
  if (a(1).eq.0.0.or.n.lt.2) then
              iex=-2
              return
  endif
c initialize parameters
      np1=n+1
      icode=1
      iroot=0
      do i=1,n
          itusd(i)=0
      enddo
      k1=1
      k2=n
      k3=1
c  copy the coefficients
      do i=1,np1
          b(i)=a(i)
      enddo
      if (b(1).eq.1.0) goto 1
c  making leading coefficient unity
      const=1.0/b(1)
      do i=1,np1
          b(i)=b(i)*const
      enddo
1     if (ans.eq.'Y'.or.ans.eq.'y') goto
2
c  find bound on the largest zero
      call bnd(b,np1,beta)
c  scale the polynomial to bring zeros
in unit circle
      call scale(b,np1,beta)
c  generate inital estimates round the
unit circle
      call gstval(n,rlz,cmz)
c  set convergence flags
2     do i=1,n
          iflag(i)=1
      enddo
      do l=1,maxit

c  calculate poly and derive values for
c  non-converged roots and store
c  in arrays rp,cp rpd and cpd
   if (.not.con) iroot=0
    do 20 i=1,n
 if (con .and. iflag(i).eq.0) goto 20
 call evaluate(b,n,rlz(i),cmz(i),rp(i),
+cp(i),rpd(i),cpd(i),sat)
          if (sat .and. l.gt.1) then
              iflag(i)=0
              iroot=iroot+1
              itusd(i)=l
          endif
          bf=rpd(i)**2+cpd(i)**2
if (bf.eq.0.0) goto 12
rdelta(i)=(-rp(i)*rpd(i)-
cp(i)*cpd(i))/bf
cdelta(i)=(-
cp(i)*rpd(i)+rp(i)*cpd(i))/bf
c calculate zstar and store inrzstar
c and czstar
          rzstar(i)=rlz(i)+rdelta(i)
          czstar(i)=cmz(i)+cdelta(i)
20   continue
     do 500 k=k1,k2,k3
if (con .and. iflag(k).eq.0) goto 500
          sr=0.0
          sc=0.0
          do 50 j=1,n
              if (j.eq.k) goto 50
              ar=rlz(k)-rzstar(j)
              ac=cmz(k)-czstar(j)
              bf=ar**2+ac**2
              if (bf.eq.0.0) goto 12
              sr=sr+ar/bf
              sc=sc-ac/bf
50          continue
          ar=rdelta(k)
          ac=cdelta(k)
          br=ar*sr-ac*sc+1.0
          bc=ar*sc+ac*sr
```

```
          bf=br**2+bc**2
          if (bf.eq.0.0) goto 12
          rdelta(k)=(ar*br+ac*bc)/bf
          cdelta(k)=(ac*br-ar*bc)/bf
          rzstar(k)=rlz(k)+rdelta(k)
          czstar(k)=cmz(k)+cdelta(k)
500   continue
      do i=1,n
          rlz(i)=rzstar(i)
          cmz(i)=czstar(i)
      enddo
c     reverse the order of updating
      k4=k1
      k1=k2
      k2=k4
      k3=-k3
      if (iroot.eq.n) goto 13
      enddo

      icode=-1
13if(.not.(ans.eq.'Y'.or.ans.eq.'y'))the
n
          do i=1,n
              rlz(i)=rlz(i)*beta
              cmz(i)=cmz(i)*beta
          enddo
      endif
      if (zflag) then
      n = n + 1
      rlz(n) = 0.
      cmz(n) = 0.
      itusd(n)=0
      endif
      return

c     abnormal exit - overflow
12    iex=-3
      return
      end
csubroutine to evaluate poly and deriv
   subroutine evaluate(a,n,x,y,rp,cp,rdp
  +,cdp,sat)
   double precision a(n+1),x,y,rp,cp,
  +rdp,cdp,b1,b2,b3,a1,a2,a3,p,q,c,
  + t,tol
      logical sat
      integer n,i
      tol=2.0**(-53)
      sat=.false.
      p=-2.0*x
      q=x*x+y*y
      t=dsqrt(q)
      b2=0.0
      a2=0.0
      b1=1.0
      a1=1.0
      c=0.8
      do i=1,n-2
          a3=a2
          a2=a1
          a1=a(i+1)-p*a2-q*a3
          c=t*c+dabs(a1)
          b3=b2
          b2=b1
          b1=a1-p*b2-q*b3
      enddo
      a3=a2
      a2=a1
      a1=a(n)-p*a2-q*a3
      rp=a(n+1)+x*a1-q*a2
      cp=a1*y
      rdp=a1-2.0*b2*y*y
      cdp=2.0*y*(b1-x*b2)
      c=t*(t*c+dabs(a1))+dabs(rp)
      sat=dsqrt(rp*rp+cp*cp).lt.
     +(2.0*dabs(x*a1)-8.0*(dabs(rp)+
     +dabs(a1)*t)+10.0*c)*tol
return
      end
csubroutineto find bound on largest zero
      subroutine bnd(b,n,beta)
      double precision b(n),beta,xm,xm1
      integer n,i
      xm=abs(b(1))
      do i=2,n
          xm1=abs(b(i))**(1.0/i)
          xm=dmax1(xm,xm1)
      enddo
      beta=2.0*xm
      return
      end
csubrout tobring zeros within unit
circle
      subroutine scale(b,n,beta)
      double precision b(n),beta,t,t1
      integer n,i
      t=1.0/beta
      t1=1.0
      do i=2,n
          t1=t1*t
          b(i)=b(i)*t1
      enddo
      return
      end
csubrout to generate initial estimates
      subroutine gstval(n,r,c)
      double precision r(n),c(n),x,a
      integer n,i
      a=4.0*atan(1.0)/n
      do i=1,n
          x=2*(i-1)*a+0.05
          r(i)=cos(x)
          c(i)=sin(x)
      enddo
      return
      end
```

```
subroutine legpol(a,n,kind)
implicit double precision(a-h,o-z)
dimension a(n+1),g(3,50),p(51,50)
do m=2,n
 x=float(m)
 g(3,m)=(1.0-x)/x
 if(kind.eq.2.or.kind.eq.3) g(2,m)
+=(1.0-2.0*x)/x
if(kind.eq.1)then
 p(1,1)=1.0
 p(2,1)=0.0
 p(1,2)=3.0/2.0
 p(2,2)=0.0
 p(3,2)=-1.0/2.0
 g(1,m)=(2.0*x-1.0)/x
 g(2,m)=0.0
elseif(kind.eq.2) then
 p(1,1)=2.0
 p(2,1)=-1.0
 p(1,2)=6.0
 p(2,2)=-6.0
 p(3,2)=1.0
 g(1,m)=(4.0*x-2.0)/x
 elseif(kind.eq.3)then
 p(1,1)=4.0
 p(2,1)=-1.0
 p(1,2)=24.0
 p(2,2)=-12.0
 p(3,2)=1.0
 g(1,m)=4.0*(2.0*x-1.0)/x
 endif
    p(1,m)=g(1,m)*p(1,m-1)
    p(2,m)=g(1,m)*p(2,m-1)+g(2,m)
  + *p(1,m-1)
 do i=3,m
    p(i,m)=g(1,m)*p(i,m-1)+g(2,m)
  +*p(i-1,m-1)+g(3,m)*p(i-2,m-2)
  enddo
    p(m+1,m)=g(2,m)*p(m,m-
    1)+g(3,m)
 + *p(m-1,m-2)
  enddo
  do k=1,n+1
    a(k)=p(k,n)
  enddo
  print*,'coeffts are',(a(i),
 + i=1,n+1)
  return
  end
csubroutine cto cdetermine the coefficients
cof cmonomial basis polynomial corresponding
cto a linear combination of legendre
cbasis polynomials
    subroutine lcomb(a,d,n,kind)
    implicit double precision(a-h,o-z)
    dimension a(n+1),d(n+1),p(51),c(51)
      do I=1,n+1
            a(I)=0.0
      do j=1,I
            k1=I-j+1
            k2=n-j+1
            call legpol(p,k2,kind)
            c(j)=d(j)*p(k1)
            a(j)=a(i)+c(j)
      enddo
      endo
      return
      end
```