2

132838

A COMPUTER PROGRAM TO CALCULATE ZEROES, EXTREMA, AND INTERVAL INTEGRALS FOR THE ASSOCIATED LEGENDRE FUNCTIONS

Mary H. Payne

```
NASA-CR-132838)  A COMPUTER PROGRAM TO                    N74-10562
CALCULATE ZEROES, EXTREMA, AND INTERVAL
INTEGRALS FOR THE ASSOCIATED LEGENDRE
(Analytical Mechanics Associates, Inc.)                   Unclas
~~45~~ p HC $4.25                  CSCL 12A    G3/19      20263
44
```

Report No. 73-19
Contract No. NAS 5-21787
April 1973

ANALYTICAL MECHANICS ASSOCIATES, INC.
50 JERICHO TURNPIKE
JERICHO, N. Y. 11753

ABSTRACT

A computer program is described for the calculation of the zeroes of the associated Legendre functions, P_{nm}, and their derivatives, for the calculation of the extrema of P_{nm} and also the integral between pairs of successive zeroes. The program has been run for all n, m from $(0,0)$ to $(20,20)$ and selected cases beyond that for n up to 40. Up to $(20,20)$, the program (written in double precision) retains nearly full accuracy, and indications are that up to $(40,40)$ there is still sufficient precision (4-5 decimal digits for a 54-bit mantissa) for estimation of various bounds and errors involved in geopotential modelling, the purpose for which the program was written.

PRECEDING PAGE BLANK NOT FILMED

TABLE OF CONTENTS

I. INTRODUCTION

This report describes a computer program for the calculation of data on the associated Legendre functions of the first kind. These data are useful in the estimation of bounds for truncation error in the spherical harmonic expansion of the geopotential, and also for the estimation of bounds on the coefficients in such an expansion. The application of the results of this calculation to these estimation problems is discussed in References 1 and 2. The accuracy requirements for estimation purposes are not very stringent, a few significant digits should be adequate. The program can operate up to degree and order 100; this limitation is imposed by the dimensioning of various arrays and would be easy to change. The program has been run from 0, 0 through 20, 20 and appears to have accuracy of 8 or 9 significant digits for this range of degrees and orders. Runs for degrees 30 and 40 with order zero indicate that one can probably run it to 40, 40 with an accuracy of four significant digits. The accuracy can probably be significantly increased by implementing one or another of the suggested modifications to the subroutine for finding roots.

In constructing the program, two formulations for the associated Legendre functions were implemented. In one, $z = \cos\theta$, where θ is the polar angle of spherical coordinates, is the independent variable. In the other, $x = \sin^2\theta/2$ is the independent variable. These two variables are related by

$$z = 1 - 2x \tag{1.1}$$

and the corresponding associated Legendre functions are given by

$$P_{nm}(z) = \begin{cases} (1-z^2)^{m/2} \cdot \text{polynomial of degree } (n-m)/2 \text{ in } z^2 \\ \qquad \text{for } n-m \text{ even} \\ (1-z^2)^{m/2} \cdot z \cdot \text{polynomial of degree } (n-m-1)/2 \\ \qquad \text{in } z^2 \text{ for } n-m \text{ odd} \end{cases} \tag{1.2}$$

$$P_{nm}(x) = [x(1-x)]^{m/2} \cdot \text{polynomial of degree } (n-m) \text{ in } x \tag{1.2}$$

From Eqs. (1.2), it would at first appear that the calculation must accommodate three cases; actually there are six cases, since the extrema of P_{nm} are found from the zeroes of the derivative of P_{nm} with respect to its independent variable, and the derivative must be handled in different ways for $m = 0$ and $m > 0$. In addition, there are seven special cases that must be handled separately (e.g., one of these is P_{00} = constant, for which there are no zeroes, extrema, or interval integrals.

The "interval integrals" mentioned above and in the title, are the integrals, between successive zeroes of P_{nm}, with respect to its independent variable. From Eq. (1.1)

$$\begin{aligned} dz &= -2\,dx \\ x &= 0 \qquad \text{corresponds to } z = 1 \\ x &= 1 \qquad \text{corresponds to } z = -1 \end{aligned} \tag{1.3}$$

The final print-out of the full set of calculations lists the zeroes of P_{nm} and its derivative in adjacent columns and in increasing order relative to the variable used. The associated extrema and interval integrals appear in the third and fourth columns. Because of the correspondence of the endpoints of the interval of definition of P_{nm} indicated in Eq. (1.3), the results read from top to bottom in the z-formulation correspond to those read from bottom to top in the x-formulation. The magnitudes of the zeroes are related by Eq. (1.1). The extrema should be identical. The interval integrals are related by a factor 2 which comes from Eq. (1.3) (not -2, since the minus sign is compensated by an interchange in limits of integration as one transforms from one formulation to the other).

In any particular run of the program, one formulation or the other is selected by an input switch. The two formulations were implemented because it seemed likely that they might well complement one another and, as we shall see, this is indeed the case. In addition, check-out of the program was greatly facilitated.

Several output options are available through another input switch. The general flow of the main program is as follows:

1. Input and initialization, including selection of the formulation to be used.

2. Calculate and print the coefficients of the polynomial parts of P_{nm} and P'_{nm}.
   Option: Terminate the program at this point and go to more input at 1

3. Calculate the zeroes of P_{nm} and P'_{nm}.
   Option: Print these zeroes and go to 1
   Option: No print; bypass 4 and go to 5

4. Calculate extrema of P_{nm} by evaluation at the zeroes of P'_{nm}.
   Option: Print zeroes and extrema and go to 1

5. Calculate the interval integrals using the zeroes of P_{nm}.
   Option (only if 4 is bypassed):
   Print zeroes and interval integral and go to 1

6. Print zeroes of P_{nm} and P'_{nm}, extrema of P_{nm}, and interval integrals in tabular form.

7. Go to 1 (with exit if no more data available).

Listings of the main program and all the subroutines are provided in the appendices. The remaining sections of the report describe the steps listed above in greater detail, with references to line and statement numbers appearing in the listings.

Section II contains a list of the input parameters and a discussion of their various functions. The branching involved in the six cases mentioned earlier is also described in Section II. This is followed, in Section III, by the recursion formulas used to obtain the coefficients of the polynomial parts of P_{nm} and P'_{nm}, and a discussion of the subroutines in which they are implemented.

The zeroes of the polynomial parts of P_{nm} and P'_{nm} are calculated by Graeffe's root squaring method, implemented in subroutine GRAEFF. Some interesting problems were encountered, and these problems and their resolution are described in Section IV. This subroutine presently limits the accuracy of the program, and hence the size of degree and order to which it can be applied. The results of a few test runs are presented, and several possibilities for improvement of the accuracy are discussed briefly.

The extrema of P_{nm} are found by direct substitution of the zeroes of P'_{nm} into P_{nm} and this is accomplished by subroutines FUNCT and EVAL, which are straightforward and easily followed from the listing. The interval integrals are calculated by Gaussian quadrature in subroutine GAUSS, which is also straightforward. A few comments on these three subroutines appear in Section V.

II. INPUT, INITIALIZATION, AND OUTPUT

The major portion of the Main Program is taken up by input, initialization, and output. The calculations are all done in subroutines, called by the Main Program. A listing of the Main Program is given in Appendix A. The references to symbols, statement numbers, and line numbers in this section apply to the Main Program. The output section is located between Statements 600 and 800. It follows the flow indicated in the Introduction with the indicated options implemented in Lines 20800, 24600, 24800, 31300, and 31800.

A block of 20 integers, IN(20), is reserved for input parameters. A block of 100 integers, NUM(100), is also used for input under certain conditions. These blocks are in NAMELISTS IN1 and IN2. The output of the program is carried in the arrays

C(101)	coefficients for the polynomial part of P_{nm}
CP(102)	coefficients for the polynomial part of P'_{nm}
Z(102)	zeroes of P_{nm}
ZP(101)	zeroes of P'_{nm}
EX(101)	extrema of P_{nm}
FIN(101)	interval integrals

The first part of the initialization consists of identifying the input block, IN, with mnemonic names as follows:

IN(1) = IND = 0	independent variable is $z = \cos\theta$
1	independent variable is $x = \sin^2\theta/2$

IN(2) = NOPT = 0	a range of degrees equally spaced is desired; see IN(7), IN(8), and IN(9)
> 0	a list of NOPT degrees to be read into the block NUM, using NAMELIST IN2 for the input
IN(3) = MOPT = -1	process all orders consistent with each specified degree
≥ 0	process only order MOPT for the specified degrees

IN(4) = INC: Print Options:

0	compute and print only C and CP
1	compute and print only C, CP, Z, and ZP
2	compute and print only C, CP, Z, ZP, and FIN
3	compute and print only C, CP, Z, ZP, and EX
4	compute and print C, CP, Z, ZP, EX, and FIN

IN(5) = ITMAX	maximum number of iterations allowed in GRAEFF for the calculation of Z and ZP	
IN(6) = NI	use the zeroes and weight factors for $P_{(NI+1),0}$ in GAUSS	
IN(7) = IMIN, IN(8) = ISTEP, IN(9) = INX	process a range of INX degrees starting at IMIN and spaced at ISTEP intervals	
IN(10) = NTOL	convergence criterion	See Section IV on GRAEFF
IN(11), IN(12)	SCALE = IN(11)**IN(12)	See Section IV on GRAEFF
IN(13)	TOL = 10**IN(13)	See Section V on GAUSS
IN(14) - IN(20)	not used at present	

A single error return is provided for several input conditions which might result in poor functioning of the program.

The second part of the initialization involves setting up the array NUM(I) in such a way that NUM(I) is the Ith degree to be processed, with a total of INX degrees. This information goes into the main DO loop starting at Statement 44; DO 1000, I=1,INX followed by N1=NUM(I), where N1 is the degree currently being processed. For NOPT>0, NUM is filled from the second READ statement (Line 4400). The DO loops to 6, 8, and 10 rearrange the degrees read and restore them to NUM so that

$$\text{NUM(I1)} > \text{NUM(I2)} \quad \text{if and only if} \quad \text{I1} > \text{I2}$$

This means that the degrees may be in any order in the data statement. For NOPT=0, Statements 20 and 30 construct NUM so that

$$\begin{aligned} \text{NUM(1)} &= \text{IMIN} \\ \text{NUM(I)} &= \text{NUM(I-1)} + \text{ISTEP} \\ \text{NUM(INX)} &= \text{IMIN} + \text{ISTEP}*\text{(INX-1)} \end{aligned}$$

Note that the dimensions of 101 and 102 for C and CP imply that the degree N1 must not exceed 100. For direct input (NOPT>0) no test is made, but for NOPT=0, NUM(I) is not permitted to exceed 100 (see DO loop 30).

The third part of the initialization calls subroutine FNORM0 (Line 8600); this step, together with the call to FNORM in Statement 58, is better discussed in the next section dealing with the calculation of the coefficients in the polynomial parts of P_{nm} and P'_{nm}.

The fourth part of the initialization sets up IMX and the array MUM, which do for orders what INX and NUM do for degrees. If MOPT>0, MUM(1)= MOPT and IMX, the number of orders to be processed is set to 1. If MOPT<0, MUM and IMX are defined (inside the DO 1000 I=1, INX loop) to include all orders consistent with the current value of N1 by the DO 45 loop.

The final step in the initialization is perhaps the most complex; it starts at Line 10300 near the beginning of the DO 999 loop (which processes all orders specified for the current N1 value) and extends to Line 20400, just before CALL COEF. This step sets up the branching procedure for the six cases mentioned in the Introduction. A basic reason for the large number of cases was the desire to make use of the symmetry involved in the $z = \cos\theta$ formulation to reduce computation time. In this formulation, the polynomial parts of P_{nm} and P'_{nm} are polynomials in $\cos^2\theta$, so that only their positive zeroes need be calculated and, from these, only the corresponding extrema and interval integrals need be calculated. The complete set is then obtained from multiplication of this set by + or -1. Further, there is little point in making GRAEFF find a zero root which is readily found by factoring.

The parameter KIND identifies the six cases, the special case for each, and the differences in their treatment. The various parameters listed with KIND are as follows:

NR	=	number of zeroes of P_{nm} to be found by GRAEFF
NRP	=	number of zeroes of P'_{nm} to be found by GRAEFF
NC	=	number of coefficients in the polynomial part of P_{nm}
NCP	=	number of coefficients in the polynomial part of P'_{nm}
NP	=	number of zeroes of P_{nm}, including endpoints and zero, if present
NPP	=	number of zeroes of P'_{nm}, including endpoints and zero, if present

Parameters starting with K are used in the rearranging and augmentation processes listed for each value of KIND below.

The case $n = m = 0$ is very special; there are no roots, extrema, or interval integrals. A special printout is provided as soon as this can be detected, Line number 9200.

For $m>1$, P'_{nm} has zeroes at ± 1 in the $\cos\theta$ formulation and at 0 and 1 in the $\sin^2\theta/2$ formulation; these points correspond to zeroes of P_{nm}, rather than to extrema, at least for the purposes of this report. These zeroes of P'_{nm} are ignored in the program and output.

For IND = 0 ($\cos\theta$ formulation), most of the zeroes of P_{nm} and P'_{nm} are obtained by taking $\pm$ the square root of the output of GRAEFF. This formulation consists of four cases, as follows:

KIND = 1: $m = 0$, n even, special case is $n = 2$

set of zeroes of P'_{nm} must be augmented by $ZP = 0$

extrema corresponding to zeroes of P'_{nm} are symmetric about $Z = 0$

interval integrals are also symmetric about $Z = 0$

set of interval integrals must be augmented by $\int_{-1}^{\text{1st zero}}$ and $\int_{\text{last zero}}^{1}$

KIND = 2: $m = 0$, n odd, special case is $n = 1$

set of zeroes of P_{nm} must be augmented by $Z = 0$

extrema and interval integrals are antisymmetric about $Z = 0$

set of interval integrals must be augmented by end-point integrals

KIND = 3: $m>0$, $n-m$ even, special case is $n = m$

set of zeroes for P_{nm} must be augmented by $Z = \pm 1$

set of zeroes for P'_{nm} must be augmented by $ZP = 0$

extrema and interval integrals are symmetric about $Z = 0$

KIND = 4: $m > 0$, $n-m$ odd, special case is $n = m+1$

set of zeroes for P_{nm} must be augmented by $Z = 0, \pm 1$

extrema and interval integrals are antisymmetric about $Z = 0$

For IND = 1 ($\sin^2 \theta/2$ formulation), subroutine GRAEFF gives all zeroes for P_{nm} and P'_{nm} except at the endpoints. The parity of $n-m$ is not significant and we do not exploit the symmetry properties of P_{nm} and P'_{nm} about the point $x = \frac{1}{2}$.

KIND = 5: $m = 0$, special case is $n = 1$

output of GRAEFF is used unchanged for zeroes and extrema

set of interval integrals must be augmented by the endpoint integrals

KIND = 6: $m > 0$, special case is $n = m$

set of zeroes of P_{nm} must be augmented by $x = 0, 1$

Although not properly a part of initialization, we mention here that in Statements 140-220 the positive square roots of the output of GRAEFF are taken (for KIND = 1, 2, 3, 4) and Z = 0, ZP = 0 are introduced where necessary. The remaining rearrangement of all roots, extrema, and interval integrals for output purposes is carried out in Statements 535-600.

The special cases, identified by ISP = 1, together with KIND, are given special treatment in Statements 800-910.

A word should be said about values to be used for some of the input parameters. The principal reason for including ITMAX in GRAEFF was to avoid being trapped in a loop, in case convergence fails. The test cases run indicate that a reasonable value for ITMAX is

$$ITMAX = 20$$

since iterations in excess of 20 appear to have no significance. NTOL and SCALE are defined by IN(10), IN(11), and IN(12). The values used in testing the program were

$$\begin{aligned} IN(10) &= NTOL = 14 \\ \left.\begin{aligned} IN(11) &= 10 \\ IN(12) &= 1 \end{aligned}\right\} & \text{ implying } SCALE = 10 \end{aligned}$$

Utilization of a hexadecimal basis for SCALE with proper adjustment of NTOL might have computational advantages on the IBM 360.

In all the tests carried out, we set

$$NI = 9$$

Some experimentation might show that a lower value could be used, particularly for small values of N, without sacrificing accuracy. Since there are NI+1 evaluations of the integrand for each entry into GAUSS, some saving of machine time could be achieved if lower values of NI yield acceptable results. In the tests on the program, we set

$$IN(13) = -12 \quad \text{implying} \quad TOL = 10^{-12}$$

This parameter is probably not significant for the analysis of P_{nm}; it was introduced so that GAUSS would be a self-contained subroutine, available for any program in which a Gaussian quadrature would be of use.

III. CALCULATION OF THE COEFFICIENTS

The coefficients for the polynomial parts of P_{nm} and P'_{nm} are calculated in three steps for the $z = \cos\theta$ formulation, using subroutines FNORM0, FNORM, and COEF (listings given in Appendix B). For the $x = \sin^2\theta/2$ formulation, only FNORM and COEF are required. We start by writing P_{nm} in the two formulations as

$$P_{nm}(z) = (1-z^2)^{m/2}\left[\sum_{k=0}^{[(n-m)/2]} C_{nm}(k)\ z^{(n-m-2k)}\right]; \quad \text{IND} = 0$$

$$P_{nm}(x) = (x(1-x))^{m/2}\sum_{k=0}^{n-m} \bar{C}_{nm}(k)\ x^k \quad ; \quad \text{IND} = 1 \tag{3.1}$$

with

$$C_{nm}(0) = A_{nm} \quad , \quad \bar{C}_{nm}(0) = \bar{A}_{nm}$$

$$C_{nm}(k+1) = -\frac{(n-m-2k)(n-m-2k-1)}{2(k+1)(2n-2k-1)}\ C_{nm}(k) \quad , \quad k=1,2,\ldots,\left[\frac{n-m-2}{2}\right]$$

$$\bar{C}_{nm}(k+1) = -\frac{(n+m+k+1)(n-m-k)}{(k+1)(m+k+1)}\ \bar{C}_{nm}(k) \quad , \quad k=1,2,\ldots,(n-m) \tag{3.2}$$

and

$$A_{nm} = \frac{(2n)!}{2^n \cdot n!}\sqrt{\frac{(2-\delta_{m0})(2n+1)}{(n-m)!\,(n+m)!}}$$

$$\bar{A}_{nm} = \frac{1}{m!}\sqrt{(2-\delta_{m0})(2n+1)\frac{(n+m)!}{(n-m)!}} \tag{3.3}$$

$$\delta_{m0} = \begin{matrix} 1 & m=0 \\ 0 & m>0 \end{matrix} \quad , \quad [p] = \text{largest integer} \le p$$

The $C_{nm}(k)$ appearing here are, of course, not geopotential coefficients. The factors A_{nm} and $\bar{A}_{nm}$ include the factor

$$\sqrt{(2-\delta_{m0})(2n+1)\frac{(n-m)!}{(n+m)!}} \tag{3.4}$$

which converts conventional associated Legendre functions into the fully normalized form used by geodesists. The derivatives for the two formulations of P_{nm} take the form, for $m > 0$:

$$\frac{dP_{nm}}{dz} = (1-z^2)^{((m/2)-1)} \left[\sum_{k=0}^{\left[\frac{n-m+1}{2}\right]} CP_{nm}(k)\ z^{(n-m+1-2k)}\right]$$

$$\frac{dP_{nm}}{dx} = (x(1-x))^{((m/2)-1)} \sum_{k=0}^{n-m+1} \overline{CP}_{nm}(k)\ x^k \tag{3.5}$$

with

$$CP_{nm}(0) = B_{nm} = -n\,A_{nm}$$

$$\overline{CP}_{nm}(0) = \bar{B}_{nm} = \frac{m}{2}\,\bar{A}_{nm}$$

$$CP_{nm}(k) = C_{nm}(k) - (n^2-m^2)C_{n-1,m}(k-1)/(n(2n-1)) \qquad k = 1, 2, \ldots, \left[\frac{n-m+1}{2}\right] \tag{3.6}$$

$$\overline{CP}_{nm}(k) = -\frac{\bar{C}_{nm}(k-1)}{mk(m+k)}\left[(m+2k)(n^2+n) - m(m+k)(m+k-1)\right] \qquad k = 1, 2, \ldots, (n-m+1)$$

Verification of these formulas is tedious, but straightforward. For $m = 0$, things are simpler:

$$\frac{dP_{n0}}{dz} = \sum_{k=0}^{\left[\frac{n-m-1}{2}\right]} CP_{n0}(k)\ z^{[n-m-1-2k]} \tag{3.7}$$

$$\frac{dP_{n0}}{dx} = \sum_{k=0}^{n-m-1} \overline{CP}_{n0}(k)\ z^{k}$$

with

$$C_{n0}(0) = B_{n0} = n\,A_{n0}$$

$$\bar{C}_{n0}(0) = \bar{B}_{n0} = \bar{A}_{n0} = \sqrt{2n+1} \tag{3.8}$$

$$CP_{n0}(k) = \frac{n-2k}{n}\,C_{n0}(k)$$

$$\overline{CP}_{n0}(k) = (k+1)\,C_{n0}(k+1)$$

The first step in the calculation of the C's and CP's for the $z = \cos\theta$ formulation is to calculate B_{n0}. This is done in subroutine FNORM0 by setting

$$B_{00} = 0 \tag{3.9}$$

$$B_{10} = \sqrt{3}$$

and using the recursion relationship

$$B_{k0} = \frac{\sqrt{4k^2-1}}{k-1}\,B_{k-1,0} \tag{3.10}$$

FNORM0 is called only once during a run, and computes and stores B_{n0} up to and including the maximum value of n to be processed. $\bar{B}_{n0}$ is so simple that it is calculated when needed in subroutine FNORM.

The second step in calculating the coefficients is carried out in subroutine FNORM, which computes A_{nm}, B_{nm} or $\bar{A}_{nm}$, $\bar{B}_{nm}$, depending upon the formulation selected. For $m = 0$, of course, the calculation is trivial. For $m>0$, the following recursion formulas are implemented in FNORM.

$$
\begin{aligned}
A_{nm} &= \sqrt{\frac{n-m+1}{n+m}}\; A_{n,m-1} \qquad m>1 \\
A_{n1} &= \sqrt{\frac{2n}{n+1}}\; A_{n0} \quad ; \quad A_{n0} = B_{n0}/n \\
B_{nm} &= -n\, A_{nm} \\
\bar{A}_{n,m+1} &= \frac{\sqrt{(n+m+1)(n-m)}}{m+1}\; \bar{A}_{nm} \qquad m>1 \\
\bar{A}_{n1} &= \sqrt{2(2n+1)(n^2+n)} \quad ; \quad \bar{A}_{n0} = \sqrt{2n+1} \\
\bar{B}_{nm} &= \frac{m}{2}\, \bar{A}_{nm}
\end{aligned}
\tag{3.11}
$$

The factor $(2-\delta_{m0})$ in A_{nm} and $\bar{A}_{nm}$ necessitates starting the recursion from A_{n1} and $\bar{A}_{n1}$, rather than from A_{n0} and $\bar{A}_{n0}$.

Finally, subroutine COEF, using the output of FNORM, implements the recursion formulas given in Eqs. (3.2), (3.6), and (3.8) to obtain the C's and CP's or $\bar{C}$'s and $\overline{CP}$'s, depending upon the formulation desired.

No study of the growth of error with the number of passes through these recursion formulas has been made. It has been noted by S. Pines (Ref. 3) that care must be exercised in the use of recursion formulas. It is possible that inaccuracies in the coefficients are responsible for the lack of precision in the determination of the zeroes of P_{nm} and P'_{nm}, although the way in which this occurs suggests that other effects dominate any inaccuracy in the coefficients. This matter is discussed further in the next section.

Note that slight variations appear between the formulas given in this section and their implementation in subroutines FNORM0, FNORM, and COEF, because DO loops cannot start from zero.

IV. THE GRAEFFE ROOT SQUARING METHOD

The subroutine for finding the zeroes of P_{nm} and P'_{nm} is GRAEFF (AA, N, Z, SCALE, NTOL, ITMAX, IND). The listing is in Appendix C. It calculates the zeroes of a polynomial of degree N-1, with a coefficient array AA of M elements, associated with increasing or decreasing powers of the variable according as IND is 1 or 0. The Graeffe root squaring method is implemented in less than full generality: An implicit assumption is that the roots are real, positive, and distinct, a condition fulfilled by the polynomial parts of P_{nm} and P'_{nm}, if z is factored from those of odd degree in the $\cos\theta$ formulation. The zeroes are stored in the array Z. The remaining entries in the calling sequence, SCALE, NTOL, and ITMAX will be discussed later.

First we outline the basic idea of the method; an excellent discussion is given by Lanczos (Ref. 4). We suppose that

$$x_1 > x_2 > \dots > x_n > 0 \tag{4.1}$$

are the zeroes in descending order of magnitude of the polynomial

$$\sum_{k=0}^{n} A_i x^i \tag{4.2}$$

Then

$$y_1 = x_1^2 > y_2 = x_2^2 > \dots > y_n = x_n^2 \tag{4.3}$$

are the zeroes in descending order of magnitude of the polynomial

$$\sum_{i=0}^{n} B_i y^i \tag{4.4}$$

where

$$
\begin{aligned}
B_0 &= A_0^2 \\
B_1 &= A_1^2 + 2A_0 A_2 \\
B_2 &= A_2^2 - 2A_1 A_3 + 2A_0 A_4 \\
&\vdots \\
B_{n-1} &= (-1)^{n-1}(A_{n-1}^2 - 2A_{n-2} A_n) \\
B_n &= (-1)^n A_n^2
\end{aligned}
\tag{4.5}
$$

As this process is iterated one obtains, on the K^{th} iterate, a polynomial with coefficients $B^{[K]}$ and zeroes

$$x_1^{(2^K)} > x_2^{(2^K)} > \ldots > x_n^{(2^K)} \tag{4.6}$$

such that the ratio of the i^{th} to the $(i-1)^{st}$ zero becomes arbitrarily small for all i and sufficiently large K. Using this fact, and the relationship between the coefficients $B^{[K]}$ and sums of products of roots, it is easy to verify that

$$\left(B_{i+1}^{[K]} \Big/ B_i^{[K]}\right)^{(2^{-K})} \tag{4.7}$$

(or its reciprocal, depending on IND) converges to the zeroes of the given polynomial. As the iterates of the coefficients B_i are constructed, it becomes apparent that they become more and more widely separated in order of magnitude. Numerically, the method terminates when the separation of the coefficients becomes such that

$$B_i^{[K+1]} = \left[B_i^{[K]}\right]^2 (-1)^i \tag{4.8}$$

because the remaining cross-product terms [see Eq. (4.5)] are beyond the word length of $\left[B_i^{[K]}\right]^2$. If the word length for the calculation is L decimal digits, the criterion for termination is thus

$$2\,B_{i+j}^{[K]} \cdot B_{i-j}^{[K]} < \left[B_i^{[K]}\right]^2 \cdot 10^{-L} \qquad (4.9)$$

for all relevant values of j. This is essentially the criterion used in GRAEFF, and L is given the name NTOL, an input quantity.

In this subroutine, the terms contributing to each B_i are added on one at a time from left to right, as shown in Eq. (4.5). An array K1(I) is defined to give the number of terms making up $B_i^{[K]}$ from the previous set of coefficients $B_i^{[K-1]}$. When the last term in this sum is beyond the word length of the K1(I)-1 terms already summed, K1(I) is diminished by 1. When K1(I) = 0 for all I, the iteration terminates.

Since both round-off error and machine time can be expected to increase with the number of iterations, ITMAX, another input quantity, is also allowed to terminate the iteration, in which case the calculation of the zeroes proceeds on the basis of the B's so far obtained. In this case, a message is written together with the array K1(I), which indicates which of the B's have failed to converge. An error message is written if any zero is negative, and the calculation proceeds with the absolute value of such a zero. A standard print states the number of iterations used on the current entry to the subroutine.

A significant problem in the implementation of Graeffe's method arose because the iterates of the coefficients grow very rapidly, and soon produce overflows. To avoid this problem, the parameter SCALE is used to convert all coefficients and their iterates to values less than SCALE and greater than or equal to 1. Then additional arrays are introduced to carry the powers associated with the coefficients; i.e., for each I

$$1 \le B(I) < SCALE \qquad NEXB(I) = \text{power} \tag{4.10}$$

and the actual corresponding coefficient is given by

$$B(I) * SCALE ** NEXB(I) \tag{4.11}$$

The program has been run (in double precision) using SCALE = 10, NTOL = 14 for all orders and degrees of P_{nm} from 0,0 to 20,20, on the DEC KA10, which has a mantissa of 54 bits. The indications are that the zeroes near zero hold 15 decimal digit precision for polynomials at least up to degree 20. The polynomial parts of P_{nm} and P'_{nm} have their largest zeroes near unity and for such a polynomial part of degree 10, the largest zeroes have 10-11 digit precision; for one of degree 20, the precision of the largest zeroes is only three or four digits. These data on precision were obtained by comparison of the zeroes of P_{n0} tabulated by the National Bureau of Standards (Ref. 5), and by comparison of the output from the two formulations. In fact, the availability of the two formulations probably enables one to go to 40,40 with 7-8 digits of precision. This is so because the small zeroes of the $\sin^2 \theta/2$ formulation can be transformed into the zeroes near unity of the $\cos\theta$ formulation, while the small zeroes of the $\cos\theta$ formulation are transformed into those near $x = \frac{1}{2}$ in the $\sin^2 \theta/2$ formulation. Thus, using the "good" zeroes from each of the two formulations and the symmetry properties, a set of zeroes good to 12 or 13 digits may easily be constructed for 20, 20. Selected cases up to 40, 40 have been run. The user of the program is cautioned that an ITMAX of 20 will be exceeded and that overflows may occur for IND = 1, if n-m is appreciably greater than 20. Both conditions may be ignored since they affect only those zeroes for which significance is already lost; they must be found by running IND = 0.

One would like, of course, to account for the lack of precision of the "large" zeroes and, if possible, improve the accuracy. An immediate thought might be that errors in the input coefficients (recall that these are computed by recursion formulas) are the primary cause. This does not seem likely, however, because for IND = 0 (cos θ formulation) the most important coefficients for large zeroes are those which start the recursion calculation. Still, all coefficients do ultimately enter the iterates of $B_i^{[K]}$ for the large zeroes, and the possibility cannot be eliminated without further testing. Another thought is that the round-off error produced by scaling is the culprit. This possibility has been tested and round-off, while present, is several orders of magnitude less than the discrepancies observed. The most plausible, but as yet untested, explanation is loss of significance in the subtractions implied by Eq. (4.5). It is possible that combining these terms starting with the smallest and ending with the largest (in magnitude) might help, at the expense of machine time spent in the sort. Probably the most practical method to improve the situation is to use the output of GRAEFF as the initial guess to a Newton procedure.

V. CALCULATION OF THE EXTREMA AND INTERVAL INTEGRALS

The subroutines for these calculations are straightforward and require little comment. To obtain the extrema, P_{nm} must be evaluated at the zeroes of P'_{nm}. This calculation is carried out by subroutines FUNCT (X, F, L) and EVAL (A, N, M, X, P, IND) (listed in Appendix D). Subroutine EVAL simply evaluates a polynomial of degree N-1 with a coefficient array A (associated with ascending or descending powers of the variable, according as IND = 1 or 0) at an array X of M points. These evaluations are returned in the array P. Subroutine FUNCT, which accepts the array of L evaluation points X, supplies whichever of the factors $(1-z^2)^{m/2}$, $z(1-z^2)^{m/2}$, or $[x(1-x)]^{m/2}$ is applicable and returns the values of P_{nm} in the array F. It appears that the extrema are relatively insensitive to errors in the zeroes of P'_{nm}. This supports the opinion given in the previous section that errors in the coefficients C and CP are relatively unimportant.

To obtain the interval integrals, subroutine GAUSS(A, B, NI, ABINT, TOL) implements the Gaussian quadrature procedure, which is well described by Lanczos (Ref. 4). Two input options are provided: The zeroes and weight factors for P_{k0}, $k = 2, 3, \ldots, 10$, are stored in data statements. The parameter NI selects those for $k = NI+1$. A parameter TOL is introduced to avoid difficulties with small differences: if the limits A, B of the integral to be evaluated satisfy

$$|A-B| < TOL \tag{5.1}$$

the subroutine returns the value zero in the output parameter ABINT, and prints out a message to this effect. Subroutine GAUSS calls FUNCT and then EVAL to evaluate the integrand where necessary.

The interval integrals are quite sensitive to inaccuracies in the zeroes of P_{nm}, as one might expect, since these inaccuracies will destroy the non-negative character of the integrand. However, it is felt that, using both formulations and symmetry considerations, the interval integrals have 8-10 digits of accuracy up to 20 , 20 and will probably retain 3-4 digits perhaps up to 40 , 40 , which should be adequate for the estimation purposes discussed in Reference 2.

It should be mentioned that the program does <u>not</u> implement the construction of a single table for the zeroes, extrema, and interval integrals utilizing the output of the two formulations in such a way as to maximize accuracy. The necessary additions to the program would be easy to insert. Time, however, did not permit sufficiently detailed examination of the output to determine the points at which the switch between formulations should be made. These switch points are very likely functions of m and n, though perhaps sensitive only to the difference n-m.

REFERENCES

1. Payne, M. H.; "An Analytic Study on Bounds for the Associated Legendre Functions," Analytical Mechanics Associates, Inc. Report No. 73-35, July 1973.

2. Payne, M. H.; "Estimation of Bounds for the Geopotential Coefficients," Analytical Mechanics Associates, Inc. Report No. 73-28, May 1973.

3. Pines, S.; "A Uniform Representation of the Gravitational Potential and its Derivatives for a Rotating Non-Spherical Body," Analytical Mechanics Associates, Inc. Report No. 71-7, February 1971.

4. Lanczos, C.; Applied Analysis, Prentice Hall, Englewood Cliffs, New Jersey, 1956.

5. Abramowitz, M. and Stegun, I. A.; Handbook of Mathematical Functions, National Bureau of Standards, Applied Mathematics Series 55, U.S. Chamber of Commerce, 1964.

APPENDIX A - Listing of the Main Program

```
      IMPLICIT REAL*8(A-H,O-Z)
C  MAIN PROGRAM FOR CALCULATING THE ZEROES OF PNM AND
C      PNM PRIME, AND EXTREMA AND INTERVAL INTEGRALS
C      FOR PNM,
C  C IS THE COEFFICIENT ARRAY FOR PNM,
C  CP IS THE COEFFICIENT ARRAY FOR PNM PRIME,
C  Z IS THE ARRAY OF ZEROES FOR PNM,
C  ZP IS THE ARRAY OF ZEROES FOR PNM PRIME,
C  EX IS THE ARRAY OF EXTREMA FOR PNM,
C  FIN IS THE ARRAY OF INTERVAL INTEGRALS FOR PNM,
C  NUM(I) IS THE ITH DEGREE TO BE PROCESSED,
C  MUM(J) IS THE JTH ORDER TO BE PROCESSED FOR THE
C      CURRENT DEGREE,
      DIMENSION IN(20),NUM(100),MUM(102),EX(101),
     CZ(102),ZP(101),FIN(101)
      COMMON/NORM0/B0(101)
      COMMON/NORM/A(101),B(101)
      COMMON/COEFF/C(101),CP(102),NC,NCP,N1,M1,IND
      NAMELIST/IN1/IN
      NAMELIST/IN2/NUM
    5   READ (5,IN1,END=3000)
      WRITE(6,IN1)
      IND=IN(1)
      NOPT=IN(2)
      MOPT=IN(3)
      INC=IN(4)
      ITMAX=IN(5)
      NI=IN(6)
      NTOL=IN(10)
      TOL=10,D0**IN(13)
      SCALE =IN(11)
      SCALE=SCALE**IN(12)
      IF (IND,LT,0) GO TO 2000
      IF (IND,GT,1) GO TO 2000
      IF (INC,LT,0) GO TO 2000
      IF (INC,GT,4) GO TO 2000
      IF (ITMAX,LE,0) GO TO 2000
      IF (ITMAX,GT,100) GO TO 2000
      IF (NI,GT,9) GO TO 2000
      IF (NI,LE,0) GO TO 2000
      IF (NTOL,LE,0) GO TO 2000
      IF (NOPT,LT,0) GO TO 2000
      IF (NOPT,EQ,0) GO TO 20
      READ(5,IN2)
      WRITE(6,IN2)
      DO 6 I=1,102
      MUM(I)=0
    6 CONTINUE
      DO 8 I =1,NOPT
      MUM(NUM(I))=1
    8 CONTINUE
      INX=1
      DO 10 I=1,100
      IF (MUM(I),EQ,0) GO TO 10
      NUM(INX)=I
      INX=INX+1
   10 CONTINUE
      IF (INX,GT,1) GO TO 15
      WRITE (6,12)
   12 FORMAT(' LIST IN2 BAD; DO LOOP TO 10 FAILED')
```

```
     CGO TO NEXT CASE')
      GO TO 5
   15 INX=INX-1
      GO TO 40
   20 IMIN=IN(7)
      ISTEP=IN(8)
      INX=IN(9)
      IF (IMIN.LT.0) GO TO 2000
      IF (ISTEP.LE.0) GO TO 2000
      IF (INX.LE.0) GO TO 2000
      DO 30 I=1,INX
      I1=I-1
      NUM(I)=IMIN+I1*ISTEP
      IF (NUM(I).LE.100) GO TO 30
      WRITE (6,25) I,NUM(I),I1
   25 FORMAT(' NUM(',I3,') =',I6,' GREATER THAN 100
     C' INX SET TO ',I3)
      INX=I1
      GO TO 40
   30 CONTINUE
   40 MUM(1)=MOPT
      IMX=1
      N=NUM(INX)
      IF (IND.EQ.1) GO TO 44
      N1=N+1
      CALL FNORM0(N,IND)
      WRITE (6,42)  N ,(B0(I),I=1, N1)
   42 FORMAT (' NORMALIZATION FACTORS FOR P00 TO PN0
     C WITH N =',I3,'  ARE'/(10X,1P3D25.14)/)
   44 DO 1000 I=1,INX
      N1=NUM(I)
      IF (N1.EQ.0) GO TO 765
      IF (MOPT.GE.0) GO TO 50
      IMX=N1+1
      DO 45 J=1,IMX
      MUM(J)=J-1
   45 CONTINUE
   50 CALL FNORM (N1,MOPT,IND)
      DO 999 J=1,IMX
      M1=MUM(J)
      WRITE (6,510) N1,M1,IND,A(J),B(J)
      N1MM1=N1-M1
      MODNM=MOD(N1MM1,2)
      IF (IND.EQ.1) MODNM=-1
      ISP=0
      IF(M1.GT.0) GO TO 90
      IF (MODNM) 60,70,80
   60 NR=N1
      NRP=NR-1
      NP=NR
      KIND=5
      IF (N1MM1.EQ.1)ISP=1
      GO TO 130
   70 NR=N1MM1/2
      NRP=NR-1
      NP=2*NR
      KIND =1
      IF (N1MM1.EQ.2) ISP=1
      K7=2*NR
      K1=K7+1
      K2=NR+1
      K3=NR
      K12=NR-1
      K4=K2
      K5=K3
      K6=K3
      K8=K2
      K9=K3
      K10=K3
      K11=K3
      KF3=K8
      SI=1.D0
      GO TO 130
   80 NR=(N1MM1-1)/2
      NRP=NR
      NP=2*NR+1
      KIND=2
      IF (N1MM1.EQ.1) ISP=1
      K7=2*NR+1
      K1=K7+1
      K5=NR+1
      K2=K5+1
      K3=K5
      K4=K5
      K8=K5
      K6=NR
      K9=K6
      K10=K5
      K11=K6
      K12=K6
      KF3=K8+1
      SI=-1.D0
      GO TO 130
   90 IF (MODNM) 120,100,110
  100  NR=N1MM1/2
      NRP=NR
      NP=2*NR+2
      KIND=3
      IF (N1MM1.EQ.0) ISP=1
      K1=2*NR+2
      K7=K1
      K3=NR
      K12=K3
      K2=NR+1
      K5=K2
      K6=K2
      K9=K2
      K10=K2
      K11=K2
      K4=K2+1
      K8=K4
      Z(K1)=1.D0
      SI=1.D0
      GO TO 130
  110  NR=(N1MM1-1)/2
      NRP=NR+1
      NP=2*NR+3
      KIND=4
      IF (N1MM1.EQ.1) ISP=1
      K1=2*NR+3
```

```
          K7=K1
          K3=NR+1
          K6=K3
          K9=K3
          K11=K3
          K12=K3
          K2=K3+1
          K4=K2
          K5=K2
          K8=K2
          K10=K2
          Z(K1)=1.D0
          SI=-1.D0
          GO TO 130
    120    NR=N1MM1
          NRP=NR+1
          NP=NR+2
          KIND=6
          IF (N1MM1.EQ.0) ISP=1
    130    NC=NR+1
          NCP=NRP+1
          NPP=NP-1
          IF (M1.EQ.0) NFP=NP+1
          IF (M1.GT.0) NFP=NP-1
          CALL COEF
          WRITE (6,520)(C(K),K=1,NC)
          WRITE (6,530)(CP(K),K=1,NCP)
          IF (INC.EQ.0) GO TO 999
          IF (ISP.EQ.1) GO TO 800
          CALL GRAEFF(C,NC,Z,SCALE,NTOL,ITMAX,IND)
          CALL GRAEFF(CP,NCP,ZP,SCALE,NTOL,ITMAX,IND)
          GO TO (140,160,180,200,220,220) KIND
    140       DO 150 K=1,NRP
          Z(K)=DSQRT(Z(K))
          KK=NRP+1-K
          ZP(KK+1)=DSQRT(ZP(KK))
    150       CONTINUE
          Z(NR)=DSQRT(Z(NR))
          ZP(1)=0.D0
          NRP=NRP+1
          GO TO 220
    160       DO 170 K=1,NR
          KK=NR+1-K
          Z(KK+1)=DSQRT(Z(KK))
          ZP(K)=DSQRT(ZP(K))
    170       CONTINUE
          Z(1)=0.D0
          NR=NR+1
          GO TO 220
    180       DO 190 K=1,NR
          KK=NR+1-K
          Z(K)=DSQRT(Z(K))
          ZP(KK+1)=DSQRT(ZP(KK))
    190       CONTINUE
          ZP(1)=0.D0
          NRP=NRP+1
          GO TO 220
    200       DO 210 K=1,NR
          KK=NR+1-K
          Z(KK+1)=DSQRT(Z(KK))
```

```
      ZP(K)=DSQRT(ZP(K))
210      CONTINUE
      Z(1)=0.D0
      ZP(NRP)=DSQRT(ZP(NRP))
      NR =NR+1
220    IF(INC-2) 535,240,230
230    CALL  FUNCT(ZP,EX,NRP)
      IF (INC.EQ.3) GO TO 535
240    X1=0.D0
      IF (MODNM) 250,250,260
250    KK1=1
      KK2=0
      GO TO 270
260    KK1=2
      KK2=-1
270    DO 300 K=KK1,NR
      X2=Z(K)
      CALL GAUSS(X1,X2,NI,ABINT,TOL)
      FIN(K+KK2)=ABINT
      X1=X2
300   CONTINUE
      X2=1.D0
      CALL GAUSS(X1,X2,NI,ABINT,TOL)
      FIN(NR+1+KK2)=ABINT
      GO TO 535
510   FORMAT (' N = ',I3,6X,'M = ',I3,6X,
     C'IND = ',I2/' NORM FACTORS: A(N,M) = '
     D,1PD21.14,5X,'B(N,M) = ',D21.14)
520    FORMAT (' COEFFICIENTS FOR PNM ARE:'/(6X,1P3D25.14)/)
530   FORMAT (' COEFFICIENTS OF PNM PRIME ARE:'
     C/(6X,1P3D25.14))
535      IF (KIND.EQ.5) GO TO 600
      IF (KIND.EQ.6) GO TO 580
      IF (M1.GT.0) GO TO 537
      KF1=K7+2
      KF2=K8+1
      KF4=K10+1
      KF5=K11+1
      GO TO 538
537      KF1=K7
      KF2=K8
      KF4=K10
      KF5=K11
538      IF (SI.GT.0.D0) FIN(1)=2.D0*FIN(1)
       DO 540 K=1,K3
      Z(K1-K)=Z(K2-K)
540      CONTINUE
      DO 550 K=1,K6
      Z(K4-K)=-Z(K5+K)
550      CONTINUE
      DO 560 K=1,K9
      K1=K7-K
      K2=K8-K
      ZP(K1)=ZP(K2)
      EX(K1)=EX(K2)
555      FIN(KF1-K)=FIN(KF2-K)
560     CONTINUE
      IF (M1.EQ.0) FIN(KF3)=FIN(1)
      DO 570 K=1,K12
      K1=K10-K
      K2=K11+K
      ZP(K1)=-ZP(K2)
      EX(K1)=SI*EX(K2)
      FIN(KF4-K)=SI*FIN(KF5+K)
  570     CONTINUE
      IF (M1,EQ,0) FIN(1)=SI*FIN(K7+1)
      GO TO 600
  580     DO 590 K=1,NR
      Z(NR-K+2)=Z(NR-K+1)
  590     CONTINUE
      Z(1)=0,D0
      Z(NR+2)=1
  600 IF (INC,EQ,4) GO TO 700
      WRITE (6,610) (Z(K),K=1,NP)
  610 FORMAT (' ZEROES OF PNM ARE'/(10X,1P3D25,14)/)
      WRITE (6,620) (ZP(K),K=1,NPP)
  620 FORMAT (' ZEROES OF PNM PRIME ARE'/(10X,1P3D25,14)/)
      IF (INC - 2) 999,650,630
  630 WRITE (6,640) (EX(K),K=1,NPP)
  640 FORMAT (' EXTREMA OF PNM ARE'/(10X,1P3D25,14)/)
      IF (INC,EQ,3) GO TO 999
  650 WRITE (6,660) (FIN(K),K=1,NFP)
  660 FORMAT (' INTERVAL INTEGRALS FOR PNM ARE'/
     C(10X,1P3D25,14)/)
      GO TO 999
  700 WRITE (6,710)
  710 FORMAT (' ZEROES OF PNM AND PNM PRIME,
     CEXTREMA OF PNM AND INTERVAL INTEGRALS FOR PNM FOLLOW'/
     D9X,'ZEROES OF PNM',9X,'ZEROES OF PNM PRIME',8X,'EX
     ETREMA OF PNM',9X,'INTERVAL INTEGRALS'/)
      IF (M1,GT,0) GO TO 715
      IP=IND-1
      WRITE (6,712) IP,FIN(1)
  712     FORMAT(40X,'INTEGRAL FROM ',I2,' TO FIRST ZERO
     CIS',4X,1PD25,14)
  715     WRITE (6,720) Z(1)
  720 FORMAT(3X,1PD25,14)
      K1=NP-1
      DO 750 K=1,K1
      KK=K
      IF(M1,EQ,0) KK=KK+1
      WRITE (6,730) ZP(K),EX(K),FIN(KK)
  730 FORMAT (28X,1P3D25,14)
      K2=K+1
      WRITE (6,740) Z(K2)
  740 FORMAT (3X,1PD25,14)
  750 CONTINUE
      IF (M1,GT,0) GO TO 999
      WRITE (6,760) FIN(NP+1)
  760     FORMAT (43X,'INTEGRAL FROM LAST ZERO TO 1 IS
     C',4X,1PD25,14)
      GO TO 999
  765     WRITE (6,770)
  770     FORMAT (' P00=1,0; P00 PRIME =0; NO ROOTS,
     AAND EXTREMA,NO INTERVAL INTEGRALS')
      GO TO 1000
  800     GO TO (810,820,830,840,850,860) KIND
  810     Z(2)=DSQRT(-C(2)/C(1))
      Z(1)=-Z(2)
      ZP(1)=0,D0
      EX(1)=C(2)
      CALL GAUSS(Z(2),1.D0,NI,FIN(3),TOL)
      CALL GAUSS(Z(1),Z(2),NI,FIN(2),TOL)
      FIN(1)=FIN(3)
      GO TO 600
820      Z(1)=0.D0
      CALL GAUSS(0.D0,1.D0,NI,FIN(2),TOL)
      FIN(1)=-FIN(2)
      GO TO 900
830      Z(1)=-1.D0
      Z(2)=1.D0
      ZP(1)=0.D0
      EX(1)=C(1)
      CALL GAUSS(-1.D0,1.D0,NI,FIN(1),TOL)
      GO TO 600
840      Z(1)=-1.D0
      Z(2)=0.D0
      Z(3)=1.D0
      ZP(2)=DSQRT(-CP(2)/CP(1))
      ZP(1)=-ZP(2)
      CALL FUNCT(ZP,EX,2)
      CALL GAUSS(0.D0,1.D0,NI,FIN(2),TOL)
      FIN(1)=-FIN(2)
      GO TO 600
850      Z(1)=-C(1)/C(2)
      CALL GAUSS(0.D0,Z(1),NI,FIN(1),TOL)
      FIN(2)=-FIN(1)
      GO TO 900
860      Z(1)=0.D0
      Z(2)=1.D0
      ZP(1)=1/2.D0
      EX(1)=C(1)/2.D0**M1
      CALL GAUSS(0.D0,1.D0,NI,FIN(1),TOL)
      GO TO 600
900      IF (KIND.EQ.2)NX1=-1
      IF (KIND.EQ.5)NX1=0
      WRITE (6,910)Z(1),NX1,FIN(1),FIN(2)
910      FORMAT (' P10 HAS ONE ZERO AT',1PD13.2,', AND NO
     A EXTREMA. THE INTERVAL INTEGRALS ARE!'/3X,'FROM
     B,I3,' TO Z(1)!',1PD25.14,'!',5X,'FROM Z(1) TO
     C1!',1PD25.14)
      GO TO 999
999   CONTINUE
1000  CONTINUE
      GO TO 5
2000  WRITE (6,2010)
2010  FORMAT (' INPUT IN1 DEFECTIVE; GO TO NEXT CASE')
      GO TO 5
3000  CONTINUE
      END
      SUBROUTINE FNORMØ (N,IND)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /NORMØ/B(1Ø1)
      IF (IND -1)1Ø,1ØØ,2ØØ
1Ø    B(1)=Ø,ØDØ
      B(2)=DSQRT(3,ØDØ)
      DO 2Ø I=2,N
      EI=I
      EISQ=EI*EI
      ENUM=DSQRT(4,ØDØ*EISQ-1,ØDØ)
      B(I+1)=ENUM*B(I)/(EI-1,DØ)
2Ø    CONTINUE
      RETURN
1ØØ   RETURN
2ØØ   RETURN
      END
      SUBROUTINE FNORM(N,M,IND)
      IMPLICIT REAL *8 (A-H,O-Z)
      COMMON/NORM0/ B0(101)
      COMMON/NORM/A(101),B(101)
      EN =N
      IF (IND-1) 10,100,200
10    A(1)=B0(N+1)/EN
      B(1)=B0(N+1)
      IF (M,EQ,0) RETURN
      IF(M,LT,0) NM=N
      IF(M,GT,0) NM=M
      C = 2,D0*EN/(EN+1,D0)
      IF (M,EQ,1) GO TO 90
      B(2)=B(1)*B(1)*C
      DO 30 I = 2,NM
      EI = I
      D = (EN-EI+1,D0)/(EN+EI)
      B(I+1)=B(I)*D
 30   CONTINUE
      IF(M,GT,0) GO TO 60
      A(1)=B(1)/EN
      NN=N+1
      DO 50 I=3,NN
      B(I)=-DSQRT(B(I))
      A(I)=-B(I)/EN
50    CONTINUE
      GO TO 90
60    B(M+1)=-DSQRT(B(M+1))
      A(M+1)=-B(M+1)/EN
      RETURN
 90   C = DSQRT(C)
      B(2)=-B(1)*C
      A(2)=-B(2)/EN
      RETURN
100   EN2PN = EN*(EN+1,D0)
      C = (2,D0*EN+1,D0)
110   A(1)=DSQRT(C)
      B(1)=-EN2PN*A(1)
      IF (M,EQ,0) RETURN
120   A(2)=C*EN2PN*2,D0
      IF (M,LT,0) MM= N
      IF(M,GT,0) MM=M
      DO 130 I = 2,MM
      EI = I
      EI2MI = EI*EI-EI
      A(I+1)=A(I)*(EN2PN-EI2MI)/(EI*EI)
130   CONTINUE
150   EM = M
      IF (MM,EQ,M) GO TO 180
      DO 160 I=1,MM
      EI=I
      A(I+1)=DSQRT(A(I+1))
      B(I+1)=(A(I+1))*EI/2,D0
160   CONTINUE
      RETURN
180   A(M+1)=DSQRT(A(M+1))
      B(M+1)=A(M+1)*EM/2,D0
200   RETURN
      END
      SUBROUTINE COEF
      IMPLICIT REAL*8(A-H,O-Z)
C  COMPUTE COEFFICIENTS OF POLYNOMIAL PARTS OF PNM, PNM PRIME
C  INPUT: ORDER N,DEGREE M AND IND TO GIVE FORMULATION DESIRED
C  OUTPUT: NUMBER NC OF COEFFICIENTS C OF PNM
C              NUMBER NCP OF COEFFICIENTS CP OF PNM PRIME
C              COEFICIENTS C(I), CP(I)
C  IND=0: PNM GIVEN IN POWERS OF (COS(THETA))**2 WITH
C             NC=[(N-M+2)/2], NCP= NC+1
C  IND=1: PNM GIVEN IN POWERS OF (SIN(THETA/2))**2 WITH
C             NC=N-M, NCP=N-M+1
C  IF M=0, NCP=NC-1 FOR BOTH FORMULATIONS
C  IND.GT.1 MAY BE USED FOR OTHER FORMULATIONS
      COMMON/NORM/A(101),B(101)
      COMMON/COEFF/C(101),CP(102),NC,NCP,N,M,IND
      NMM=N-M
      ENMM=NMM
      EN=N
      EM=M
      C(1)=A(M+1)
      CP(1)=B(M+1)
      EN2PN=EN*EN+EN
      IF(IND-1)100,400,700
100      ENMO2=ENMM/2.D0
      NMMO2=ENMO2
      K1=NC-1
      TWONP1=2.D0*EN+1.D0
      TWONM1=TWONP1-2.D0
      TWOM=2.D0*EM
      ENMP1=ENMM+1.D0
      T=-ENMO2*CP(1)/(EN*TWONM1)
      S=EN2PN-EN*EM+TWOM
      DO 150 K=1,K1
      EK=K
      TWOK=2.D0*EK
      C(K+1)=-C(K)*(ENMM+2.D0-TWOK)*(ENMP1-TWOK)
     1/(TWOK*(TWONP1-TWOK))
      CP(K+1)=T*S
      T=-T*(ENMM-TWOK)*(ENMP1-TWOK)/((TWOK
     1+2.D0)*(TWONM1-TWOK))
      S=S+TWOM
150      CONTINUE
      IF(M.EQ.0) GO TO 200
      IF(NC.GE.NCP)RETURN
      ENC=NC
      CP(NCP)=C(NC)
      RETURN
 200       DO 210 I=2,NC
      EI=I
      CP(I)=(ENMM-2.D0*(EI-1.D0))*C(I)
 210       CONTINUE
      RETURN
400      DO 420 K=1,NMM
      EK=K
      TWOK=2.D0*EK
      EMPK=EM+EK
      EMPKM1=EMPK-1.D0
      EKMPK=EK*EMPK
      T=(EN2PN-EMPKM1*EMPK)/EKMPK
      C(K+1)=-T*C(K)
      S=(EM+TWOK)*EN2PN-EM*EMPKM1*EMPK
      CP(K+1)=-S*C(K)/(2.D0*EKMPK)
420      CONTINUE
      CP(NCP)=-EN*C(NC)
      IF(M.GT.0)RETURN
      DO 450 K=2,NCP
      EK=K
      CP(K)=EK*C(K+1)
450      CONTINUE
      RETURN
700      RETURN
      END
      SUBROUTINE GRAEFF (AA,N,Z,SCALE,NTOL,ITMAX,IND)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION A(1Ø2),AA(1Ø2),Z(1Ø2),B(1Ø2),K1(1Ø2),
     CNEXA(1Ø2),NEXB(1Ø2)
C ROOTS Z(I) OF A POLYNOMIAL OF DEGREE N-1 BY GRAEFFE'S METHOD
C A(I) IS COEFFICIENT OF X**(N-I), I=1,2,..,N FOR IND = Ø
C "    "     "              OF X**(I-1),     "      FOR IND.GT.Ø
C ITMAX IS THE MAXIMUM NUMBER OF ITERATIONS
C NTOL TERMINATES ITERATION ON A CONVERGENCE CRITERION
      IF (N.EQ.1) GO TO 27Ø
      IF (N.EQ.2) GO TO 28Ø
      ITER = 1
      EN=N
      ENO2 = EN/2.DØ
      NO2=ENO2
      DO 1Ø I = 1,N
      A(I)=AA(I)
      SIG1=1.ØDØ
      IF (I.LE.NO2) K1(I)=I-1
      IF (I.GT.NO2) K1(I)=N-I
      NEX=Ø
      TEST=A(I)
      IF (TEST.LT.Ø.DØ) SIG1=-1.DØ
      TEST=DABS(TEST)
    2 IF (TEST.LT.SCALE) GO TO 4
      TEST=TEST/SCALE
      NEX=NEX+1
      GO TO 2
    4 IF (TEST.GE.1.DØ) GO TO 6
      TEST = TEST*SCALE
      NEX=NEX - 1
      GO TO 4
    6 NEXA(I)=NEX
      A(I)=SIG1*TEST
   1Ø CONTINUE
   2Ø DO 1ØØ I=1,N
      SIG=-1.DØ
      C=A(I)*A(I)
      NEXC=NEXA(I)*2
      KSUM=K1(I)
      IF (K1(I).EQ.Ø) KSUM = 1
   3Ø DO 95 K=1,KSUM
      IF (K1(I).EQ.Ø) GO TO 75
      TERM=A(I+K)*A(I-K)*2.DØ
      NEXT=NEXA(I+K)+NEXA(I-K)
            NEXD=NEXC-NEXT
      IF(IABS(NEXD).LT.NTOL) GO TO 45
      IF (K.EQ.KSUM) K1(I)=KSUM-1
      GO TO 94
   45 IF (NEXD.LT.Ø) GO TO 5Ø
      TERM=TERM*SCALE**(-NEXD)
      C=C+TERM*SIG
      GO TO 75
   5Ø C=C*SCALE**(NEXD)
      C=C+TERM*SIG
      NEXC=NEXT
   75    SIG1=1
      IF(C.LT.Ø.DØ)SIG1=-1
      C=DABS(C)
   8Ø    IF (C.LT.SCALE) GO TO 85
```

```
          C=C/SCALE
          NEXC=NEXC+1
          GO TO 80
 85          IF (C.GE.1.D0) GO TO 90
          C=C*SCALE
          NEXC=NEXC-1
          GO TO 85
 90          C=C*SIG1
          IF (K1(I).EQ.0) GO TO 96
 94          SIG=-SIG
 95          CONTINUE
96          B(I)=C
          NEXB(I)=NEXC
 100      CONTINUE
          DO 110 I=1,N
          IF (K1(I).GT.0) GO TO 120
 110      CONTINUE
          GO TO 200
 120      IF (ITER.GE.ITMAX) GO TO 180
          ITER=ITER+1
          SIG=1.D00
          DO 130 K=1,N
          A(K)=B(K)*SIG
          NEXA(K)=NEXB(K)
          SIG=-SIG
 130      CONTINUE
          GO TO 20
 180      WRITE (6,190) (K1(I),I=1,N)
 190      FORMAT (' ITMAX EXCEEDED; K IS',10I5/(3X,15I5/))
 200      EXP=2.D0**(-ITER)
          N3=N-1
          DO 250 I=1,N3
          IF (IND) 210,210,220
 210      N4=N-I
          Z1   =B(I+1)/B(I)
          NEXZ=NEXB(I+1)-NEXB(I)
          GO TO 230
 220      Z1   = B(I)/B(I+1)
          NEXZ=NEXB(I)-NEXB(I+1)
          N4=I
 230      IF (Z1   .LT.0) WRITE (6,240) N4,Z1
 240      FORMAT (' Z(',I5,') NEGATIVE AND EQUAL TO',E18.8)
          Z1   =DABS(Z1   )**EXP
          EXZ=NEXZ
          EXZ=EXZ*EXP
          Z(N4)=Z1 *SCALE**EXZ
 250      CONTINUE
          WRITE (6,260) ITER
 260          FORMAT(' GRAEFF USED',I3,'ITERATIONS')
          RETURN
 270          WRITE (6,275)
 275          FORMAT (' POLYNOMIAL IS OF DEGREE 0; NO ROOTS')
          RETURN
 280          IF (IND.EQ.0) Z(1)=-AA(2)/AA(1)
          IF (IND.EQ.1) Z(1)=-AA(1)/AA(2)
          WRITE (6,285) Z(1)
 285          FORMAT (' POLYNOMIAL IS LINEAR; Z(1) =',1PD25.14)
          RETURN
          END
```

```
      SUBROUTINE FUNCT (X,P,L)
      IMPLICIT REAL*8(A-H,O-Z)
C  TO EVALUATE ASSOCIATED LEGENDRE FUNCTIONS PNM AT
C  L POINTS X, OUTPUT IS L VALUES OF PNM, IND
C  INDICATES THE FORMULATION USED.
      COMMON/COEFF/C(101),CP(102),NC,NCP,N,M,IND
      DIMENSION X(102),P(102),Y(102)
      EM=M
      EMO2=EM/2.0D0
      IF(IND-1) 10,100,200
   10 DO 20 I=1,L
      Y(I)=X(I)*X(I)
   20 CONTINUE
      CALL EVAL(C,NC,L,Y,P,IND)
      IF (MOD((N-M),2).EQ.0) GO TO 40
      DO 30 I =1,L
      P(I)=P(I)*X(I)*(1.D0-X(I)*X(I))**EMO2
   30 CONTINUE
      RETURN
   40 DO 50 I=1,L
      P(I)=P(I)*(1.D0-X(I)*X(I))**EMO2
   50 CONTINUE
      RETURN
  100 CALL EVAL(C,NC,L,X,P,IND)
      DO 110 I=1,L
      P(I)=P(I)*(X(I)*(1.D0-X(I)))**EMO2
  110 CONTINUE
      RETURN
  200 RETURN
      END
      SUBROUTINE EVAL(A,N,M,X,P,IND)
C  EVALUATE A POLYNOMIAL OF DEGREE N-1 AT M POINTS X(I)
C  RETURN M VALUES P(I)
C  A(I) IS THE COEFFICIENT OF X**(N-I) FOR IND = 0
C   "         "  "           "              " X**(I-1) FOR IND.NE.0
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(102),X(102),P(102)
      IF (N.GT.1) GO TO 10
      DO 5 K=1,M
      P(K)=A(1)
    5 CONTINUE
      RETURN
   10 IF (IND.EQ.1) GO TO 50
      DO 40 I=1,M
      T=X(I)
      Y=A(1)*T+A(2)
      IF (N.EQ.2) GO TO 35
      DO 30 K=3,N
      Y=T*Y+A(K)
   30 CONTINUE
   35    P(I)=Y
   40 CONTINUE
      RETURN
   50 DO 80 I=1,M
      T=X(I)
      Y=A(N)*T+A(N-1)
      IF (N.EQ.2) GO TO 75
      DO 70 K=3,N
      Y=Y*T+A(N+1-K)
   70 CONTINUE
   75    P(I)=Y
   80 CONTINUE
      RETURN
      END
      SUBROUTINE GAUSS(A,B,N,ABINT,TOL)
C  SUBROUTINE FOR THE INTEGRAL FROM A TO B BY GAUSSIAN QUADRATURE
C  Z(J,L) ARE THE ZEROES OF THE (L+1)ST LEGENDRE POLYNOMIAL
C  W(J,L) ARE THE CORRESPONDING WEIGHTS
C  CALLS SUBROUTINE FUNCT WHICH DEFINES THE INTEGRAND
C  ABINT IS THE OUTPUT
      IMPLICIT REAL *8 (A-H,O-Z)
      DIMENSION Z(5,9), W(5,9),X(10),F(10)
      DATA Z/.577350269189626D0, 4*0.D0,
     1 0.D0, .774596669241483D0, 3*0.D0,
     2 .339981043584856D0, .861136311594053D0, 3*0.D0,
     3 0.D0, .538469310105683D0, .906179845938664D0, 2*0.D0,
     4 .238619186083197D0, .661209386466265D0, .932469514203152D0, 3*
     5 0.D0, .405845151377397D0, .741531185599394D0,
     6 .949107912342759D0, 0.D0,
     7 .183434642495650D0, .525532409916329D0, .796666477413627D0,
     8 .960289856497536D0, 0.D0,
     9 0.D0, .324253423403809D0, .613371432700590D0,
     A .836031107326636D0, .968160239507626D0,
     B .148874338981631D0, .433395394129247D0, .679409568299024D0,
     C .865063366688985D0, .973906528317172D0/
      DATA W/1.D0, 4*0.D0,
     1 .888888888888889D0,.555555555555556D0,3*0.D0,
     2 .652145154862546D0, .347854845137454D0, 3*0.D0,
     3 .568888888888889D0, .478628670499366D0,
     4 .236926885056189D0, 2*0.D0,
     5 .467913934572691D0, .360761573048139D0,
     6 .171324492379170D0, 2*0.D0,
     7 .417959183673469D0, .381830050505119D0,
     8 .279705391489277D0, .129484966168870D0, 0.D0,
     9 .362683783378362D0, .313706645877887D0,
     A .222381034453374D0, .101228536290376D0, 0.D0,
     B .330239355001260D0, .312347077040003D0, .260610696402935D0,
     C .180648160694857D0, .081274388361574D0, .295524224714753D0,
     D .269266719309996D0, .219086362515982D0,
     E .149451349150581D0, .066671344308688D0/
      FACM=(B-A)/2.D0
      IF (DABS(FACM).GT.TOL) GO TO 5
      ABINT = 0.D0
      WRITE (6,2)
    2 FORMAT (' (B-A).LT.TOL; ABINT SET TO ZERO')
      RETURN
    5 FACP=(B+A)/2.D0
      ABINT=0.D0
      EN=N
      ENO2=EN/2.D0
      NO2=ENO2
      IF((N-2*NO2).EQ.0) K1=2
      IF((N-2*NO2).GT.0) K1=1
      K2=NO2+1
      K3=2*K2+1
      DO 10 K=K1,K2
      TERM=FACM*Z(K,N)
      X(K)=FACP-TERM
      X(K3 -K)= FACP+TERM
   10 CONTINUE
      IF (K1.EQ.1) GO TO 15
      X(1)=FACP
   15 L=N+1
      CALL FUNCT(X,F,L)
```

```
          DO 22 K=K1,K2
          ABINT=ABINT+(F(K)+F(K3-K  ))*W(K,N)
    22    CONTINUE
          IF(K1.EQ.1)  GO TO 25
          ABINT=ABINT+F(1)*W(1,N)
    25    ABINT=ABINT*FACM
          RETURN
```