ISSN: 1121-0540

ISTITUTO
DI
ANALISI NUMERICA

del

CONSIGLIO NAZIONALE DELLE RICERCHE
Via Abbiategrasso, 209 – 27100 PAVIA (Italy)

PAVIA
1993

PUBBLICAZIONI

N. 891

D. Funaro

FORTRAN Routines for Spectral Methods

FORTRAN Routines for Spectral Methods

Daniele Funaro

Contents

Introduction

Spectral methods are an efficient tool to recover accurate approximated solutions to ordinary and partial differential equations, in terms of high-degree trigonometric or algebraic polynomials. A lot has been done to study the numerical implementation and the theoretical analysis of convergence of these techniques. A survey of the main results is given for instance in [1], [2], [3], [5], [7].

When building a numerical code using spectral methods, a preliminary part has to be devoted to a series of basic algorithms, not often readily available in the usual program libraries. This requires the user to be a little acquainted with the properties of orthogonal functions. The initialization is however quite standard and the collection of FORTRAN routines presented here is intended to provide the user with a ready software product, in order to allow a smooth start in the developement of a more extensive code.

The *double precision* subroutines are listed in this manual together with a detailed description. Style and notations are those adopted in [4], where *tau* and *collocation* approximations by algebraic polynomials are studied for one-dimensional differential equations. As a matter of fact, this text has been conceived to complement the material contained in [4].

The name of each routine consists of six letters. The third and the fourth denote the kind of polynomial basis considered: JA stands for Jacobi, LE for Legendre, CH for Chebyshev, LA for Laguerre, HE for Hermite. For the routines related with a certain set of collocation nodes, the last two letters denote the type of integration formula originating from these nodes: GA stands for Gauss, GL for Gauss-Lobatto, GR for Gauss-Radau.

In order to fit the mathematical notations, many vectors are dimensioned starting from the component zero. The user is invited to check the vector dimensioning of his main program with the help of the tables provided at the end of the manual.
Some of the routines related to the set of Chebyshev collocation nodes have a second version using the algorithm of the Fast Fourier Transform. In this case, external auxiliary subroutines of the NAG (*Numerical Algorithms Group*) FORTRAN Library are needed.
The routines have been tested when the various parameters fall in the range of standard applications. Of course, for high values of the polynomial degrees, round off errors may occur, and a special care is required in the computations involving Laguerre or Hermite polynomials. The user is suggested to double check the validity of the output results in those situations expected to be critical.

Purposes

GAMMAF: evaluates the real gamma function at a given point

VAJAPO: computes the value and the derivatives of the Jacobi polynomials at a given point

VALEPO: computes the value and the derivatives of the Legendre polynomials at a given point

VACHPO: computes the value and the derivatives of the Chebyshev polynomials at a given point

VALAPO: computes the value and the derivatives of the Laguerre polynomials at a given point

VAHEPO: computes the value and the derivatives of the Hermite polynomials at a given point

VALASF: computes the value and the derivative of the scaled Laguerre functions at a given point

VAHESF: computes the value and the derivative of the scaled Hermite functions at a given point

ZEJAGA: finds the zeroes of the Jacobi polynomials

ZELEGA: finds the zeroes of the Legendre polynomials

ZECHGA: finds the zeroes of the Chebyshev polynomials

ZELAGA: finds the zeroes of the Laguerre polynomials

ZEHEGA: finds the zeroes of the Hermite polynomials

ZEJAGL: finds the nodes of the Jacobi Gauss-Lobatto integration formula

ZELEGL: finds the nodes of the Legendre Gauss-Lobatto integration formula

ZECHGL: finds the nodes of the Chebyshev Gauss-Lobatto integration formula

ZELAGR: finds the nodes of the Laguerre Gauss-Radau integration formula

WEJAGA: finds the weights of the Jacobi Gauss integration formula

WELEGA: finds the weights of the Legendre Gauss integration formula

WECHGA: finds the weights of the Chebyshev Gauss integration formula

WELAGA: finds the weights of the Laguerre Gauss integration formula

WEHEGA: finds the weights of the Hermite Gauss integration formula

WEJAGL: finds the weights of the Jacobi Gauss-Lobatto integration formula

WELEGL: finds the weights of the Legendre Gauss-Lobatto integration formula

WECHGL: finds the weights of the Chebyshev Gauss-Lobatto integration formula

WELAGR: finds the weights of the Laguerre Gauss-Radau integration formula

WECHCC: finds the weights of the Clenshaw-Curtis integration formula

INJAGA: evaluates at a given point the value of a polynomial given at the Jacobi zeroes

INLEGA: evaluates at a given point the value of a polynomial given at the Legendre zeroes

INCHGA: evaluates at a given point the value of a polynomial given at the Chebyshev zeroes

INLAGA: evaluates at a given point the value of a polynomial given at the Laguerre zeroes

INHEGA: evaluates at a given point the value of a polynomial given at the Hermite zeroes

INJAGL: evaluates at a given point the value of a polynomial given at the Jacobi Gauss-Lobatto nodes

INLEGL: evaluates at a given point the value of a polynomial given at the Legendre Gauss-Lobatto nodes

INCHGL: evaluates at a given point the value of a polynomial given at the Chebyshev Gauss-Lobatto nodes

INLAGR: evaluates at a given point the value of a polynomial given at the Laguerre Gauss-Radau nodes

NOLEGA: evaluates the $L^2(-1,1)$ norm and the discrete maximum norm of a polynomial given at the Legendre zeroes

NOCHGA: evaluates the $L^2(-1,1)$ norm (with and without weight function) and the discrete maximum norm of a polynomial given at the Chebyshev zeroes

NOJAGL: evaluates the weighted $L^2(-1,1)$ norm, the quadrature norm, and the discrete maximum norm of a polynomial given at the Jacobi Gauss-Lobatto nodes

NOLEGL: evaluates the $L^2(-1,1)$ norm, the quadrature norm, and the discrete maximum norm of a polynomial given at the Legendre Gauss-Lobatto nodes

NOCHGL: evaluates the $L^2(-1,1)$ norm (with and without weight function), the quadrature norm, and the discrete maximum norm of a polynomial given at the Chebyshev Gauss-Lobatto nodes

COJAGA: evaluates the Jacobi Fourier coefficients of a polynomial given at the Jacobi zeroes

COLEGA: evaluates the Legendre Fourier coefficients of a polynomial given at the Legendre zeroes

COCHGA: evaluates the Chebyshev Fourier coefficients of a polynomial given at the Chebyshev zeroes (without using FFT)

COLAGA: evaluates the Laguerre Fourier coefficients of a polynomial given at the Laguerre zeroes

COHEGA: evaluates the Hermite Fourier coefficients of a polynomial given at the Hermite zeroes

COJAGL: evaluates the Jacobi Fourier coefficients of a polynomial given at the Jacobi Gauss-Lobatto nodes

COLEGL: evaluates the Legendre Fourier coefficients of a polynomial given at the Legendre Gauss-Lobatto nodes

COCHGL: evaluates the Chebyshev Fourier coefficients of a polynomial given at the Chebyshev Gauss-Lobatto nodes (without using FFT)

COLAGR: evaluates the Laguerre Fourier coefficients of a polynomial given at the Laguerre Gauss-Radau nodes

PVJAEX: computes the value and the derivatives at a given point of a polynomial, from its Jacobi Fourier coefficients

PVLEEX: computes the value and the derivatives at a given point of a polynomial, from its Legendre Fourier coefficients

PVCHEX: computes the value and the derivatives at a given point of a polynomial, from its Chebyshev Fourier coefficients

PVLAEX: computes the value and the derivatives at a given point of a polynomial, from its Laguerre Fourier coefficients

PVHEEX: computes the value and the derivatives at a given point of a polynomial, from its Hermite Fourier coefficients

NOJAEX: evaluates the weighted $L^2(-1,1)$ norm of a polynomial, from its Jacobi Fourier coefficients

NOLEEX: evaluates the $L^2(-1,1)$ norm of a polynomial, from its Legendre Fourier coefficients

NOCHEX: evaluates the $L^2(-1,1)$ norm (with and without weight function) of a polynomial, from its Chebyshev Fourier coefficients

NOLAEX: evaluates the weighted $L^2(0,+\infty)$ norm of a polynomial, from its Laguerre Fourier coefficients

NOHEEX: evaluates the weighted $L^2(\mathbf{R})$ norm of a polynomial, from its Hermite Fourier coefficients

COJADE: computes the Jacobi Fourier coefficients of the derivatives of a polynomial, from its Jacobi Fourier coefficients

COLEDE: computes the Legendre Fourier coefficients of the derivatives of a polynomial, from its Legendre Fourier coefficients

COCHDE: computes the Chebyshev Fourier coefficients of the derivatives of a polynomial, from its Chebyshev Fourier coefficients

COLADE: computes the Laguerre Fourier coefficients of the derivatives of a polynomial, from its Laguerre Fourier coefficients

COHEDE: computes the Hermite Fourier coefficients of the derivatives of a polynomial, from its Hermite Fourier coefficients

DEJAGA: computes the derivative at the Jacobi zeroes of a polynomial given at the Jacobi zeroes

DELAGA: computes the derivative at the Laguerre zeroes of a polynomial given at the Laguerre zeroes

DEHEGA: computes the derivative at the Hermite zeroes of a polynomial given at the Hermite zeroes

DEJAGL: computes the derivative at the Jacobi Gauss-Lobatto nodes of a polynomial given at the Jacobi Gauss-Lobatto nodes

DELEGL: computes the derivative at the Legendre Gauss-Lobatto nodes of a polynomial given at the Legendre Gauss-Lobatto nodes

DECHGL: computes the derivative at the Chebyshev Gauss-Lobatto nodes of a polynomial given at the Chebyshev Gauss-Lobatto nodes (without using FFT)

DELAGR: computes the derivative at the Laguerre Gauss-Radau nodes of a polynomial given at the Laguerre Gauss-Radau nodes

DMJAGL: gives the entries of the derivative matrix relative to the Jacobi Gauss-Lobatto nodes

DMLEGL: gives the entries of the derivative matrix relative to the Legendre Gauss-Lobatto nodes

DMCHGL: gives the entries of the derivative matrix relative to the Chebyshev Gauss-Lobatto nodes

DMLAGR: gives the entries of the derivative matrix relative to the Laguerre Gauss-Radau nodes

FCCHGA: evaluates the Chebyshev Fourier coefficients of a polynomial given at the Chebyshev zeroes (using FFT).

FCCHGL: evaluates the Chebyshev Fourier coefficients of a polynomial given at the Chebyshev Gauss-Lobatto nodes (using FFT).

FVCHGL: computes the values at the Chebyshev Gauss-Lobatto nodes of a polynomial, from its Chebyshev Fourier coefficients (using FFT).

FDCHGL: computes the derivative at the Chebyshev Gauss-Lobatto nodes of a polynomial given at the Chebyshev Gauss-Lobatto nodes (using FFT).

DESCRIPTIONS AND LISTINGS

GAMMAF

The routine computes the value, at a given point $x \in]10^{-75}, 57[$, of the Gamma function

$$\Gamma(x) = \int_0^{+\infty} t^{x-1} e^{-t} dt.$$

One is reduced to work with $x < 1$ by using the relations

$$\Gamma(x+1) \;=\; x\Gamma(x), \quad x > 0 \qquad \text{and} \qquad \Gamma(n+1) \;=\; n!, \quad n \in \mathbf{N}.$$

The computation is further restricted to the interval $].5, 1[$ by virtue of the formula

$$\Gamma(x)\Gamma(1-x) \;=\; \frac{\pi}{\sin \pi x}, \qquad 0 < x < 1.$$

Finally, the value of $\Gamma(x)$, $x \in].5, 1[$, is obtained by a truncation of a series expansion as suggested in [6], Vol.1, p.30.

Input variable	*Output variable*
X , the argument x	GX , the value of Γ in x

```
      SUBROUTINE GAMMAF(X,GX)
*************************************************************
*     COMPUTES THE GAMMA FUNCTION AT A GIVEN POINT
*     X = ARGUMENT GREATER THAN 1.E-75 AND SMALLER THAN 57.
*     GX= VALUE OF GAMMA IN X
*************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION C(11)

      IF (X .LE. 1.D-75 .OR. X .GE. 57.D0) THEN
         WRITE(*,*) 'ARGUMENT OUT OF RANGE IN SUBROUTINE GAMMAF'
         RETURN
      ENDIF
```

```
         PI  = 3.14159265358979323846D0
         EPS = 1.D-14
         XX  = X
         GX  = 1.0D0

1     IF (DABS(XX-1.D0) .LT. EPS) RETURN
      IF (XX .GE. 1.D0) THEN
         XX = XX-1.D0
         GX = GX*XX
         GOTO 1
      ENDIF
         IND = 0
      IF (XX .LT. .5D0) THEN
         IND = 1
         GX  = GX*PI/DSIN(PI*XX)
         XX  = 1.D0-XX
      ENDIF

         PR = 1.D0
         S  = 0.426401432711220868D0
         C(1)  = -0.524741987629368444D0
         C(2)  =  0.116154405493589130D0
         C(3)  = -0.765978624506602380D-2
         C(4)  =  0.899719449391378898D-4
         C(5)  = -0.194536980009534621D-7
         C(6)  =  0.199382839513630987D-10
         C(7)  = -0.204209590209541319D-11
         C(8)  =  0.863896817907000175D-13
         C(9)  =  0.152237501608472336D-13
         C(10) = -0.825725175277719950D-14
         C(11) =  0.299734782205224610D-14

      DO 2 K=1,11
         PR = PR*(XX-DFLOAT(K))/(XX+DFLOAT(K-1))
         S  = S+C(K)*PR
2     CONTINUE
         G  = S*DEXP(1.D0-XX)*(XX+4.5D0)**(XX-.5D0)
      IF (IND .EQ. 1) THEN
         GX = GX/G
      ELSE
         GX = GX*G
      ENDIF

      RETURN
      END
```

VAJAPO

The routine computes the value of the Jacobi polynomial $P_n^{(\alpha,\beta)}$ of degree $n \in \mathbf{N}$ ($\alpha > -1$ and $\beta > -1$ are real parameters) at the point x, according to the recursion formula

$$\begin{cases} P_0^{(\alpha,\beta)}(x) = 1, \\ P_1^{(\alpha,\beta)}(x) = \frac{1}{2}(\alpha+\beta+2)x + \frac{1}{2}(\alpha-\beta), \\ P_n^{(\alpha,\beta)}(x) = c_1^{-1}[(c_2 x + c_3)P_{n-1}^{(\alpha,\beta)}(x) - c_4 P_{n-2}^{(\alpha,\beta)}(x)], \quad n \geq 2, \end{cases}$$

where

$$c_1 = 2n(n+\alpha+\beta)(2n+\alpha+\beta-2),$$

$$c_2 = (2n+\alpha+\beta-1)(2n+\alpha+\beta-2)(2n+\alpha+\beta),$$

$$c_3 = (2n+\alpha+\beta-1)(\alpha^2-\beta^2),$$

$$c_4 = 2(n+\alpha-1)(2n+\alpha+\beta)(n+\beta-1).$$

Simultaneously, first and second derivatives of $P_n^{(\alpha,\beta)}$ at the point x are computed by taking the derivatives of the terms of the above recursion formula.

Input variables	*Output variables*
N , the degree n	Y , the value of $P_n^{(\alpha,\beta)}$ in x
A , the parameter α	DY , the value of $\frac{d}{dx}P_n^{(\alpha,\beta)}$ in x
B , the parameter β	$D2Y$, the value of $\frac{d^2}{dx^2}P_n^{(\alpha,\beta)}$ in x
X , the argument x	

```
      SUBROUTINE VAJAPO(N,A,B,X,Y,DY,D2Y)
*************************************************************
*   COMPUTES THE VALUE OF THE JACOBI POLYNOMIAL OF DEGREE N
*   AND ITS FIRST AND SECOND DERIVATIVES AT A GIVEN POINT
*   N  = DEGREE OF THE POLYNOMIAL
*   A  = PARAMETER >-1
*   B  = PARAMETER >-1
*   X  = POINT IN WHICH THE COMPUTATION IS PERFORMED
*   Y  = VALUE OF THE POLYNOMIAL IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*   D2Y= VALUE OF THE SECOND DERIVATIVE IN X
*************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
         Y   = 1.D0
         DY  = 0.D0
         D2Y = 0.D0
      IF (N .EQ. 0) RETURN

         AB  = A+B
         Y   = .5D0*(AB+2.D0)*X+.5D0*(A-B)
         DY  = .5D0*(AB+2.D0)
         D2Y = 0.D0
      IF(N .EQ. 1) RETURN

         YP   = 1.D0
         DYP  = 0.D0
         D2YP = 0.D0
      DO 1 I=2,N
         DI = DFLOAT(I)
         C0 = 2.D0*DI+AB
         C1 = 2.D0*DI*(DI+AB)*(C0-2.D0)
         C2 = (C0-1.D0)*(C0-2.D0)*C0
         C3 = (C0-1.D0)*(A-B)*AB
         C4 = 2.D0*(DI+A-1.D0)*C0*(DI+B-1.D0)
         YM = Y
         Y  = ((C2*X+C3)*Y-C4*YP)/C1
         YP = YM
         DYM  = DY
         DY   = ((C2*X+C3)*DY-C4*DYP+C2*YP)/C1
         DYP  = DYM
         D2YM = D2Y
         D2Y  = ((C2*X+C3)*D2Y-C4*D2YP+2.D0*C2*DYP)/C1
         D2YP = D2YM
1     CONTINUE

      RETURN
      END
```

VALEPO

The routine computes the value of the Legendre polynomial P_n of degree $n \in \mathbf{N}$ and its first and second derivatives at the point x, according to the recursion formulas

$$\begin{cases} P_0(x) = 1, \\ P_1(x) = x, \\ P_n(x) = \dfrac{1}{n}[(2n-1)xP_{n-1}(x) - (n-1)P_{n-2}(x)] \quad n \geq 2, \end{cases}$$

$$\begin{cases} P_0'(x) = 0, \\ P_1'(x) = 1, \\ P_n'(x) = \dfrac{1}{n}[(2n-1)(xP_{n-1}'(x) + P_{n-1}(x)) - (n-1)P_{n-2}'(x)] \quad n \geq 2, \end{cases}$$

$$\begin{cases} P_0''(x) = 0, \\ P_1''(x) = 0, \\ P_n''(x) = \dfrac{1}{n}[(2n-1)(xP_{n-1}''(x) + 2P_{n-1}'(x)) - (n-1)P_{n-2}''(x)] \quad n \geq 2. \end{cases}$$

Input variables	*Output variables*
N , the degree n X , the argument x	Y , the value of P_n in x DY , the value of P_n' in x $D2Y$, the value of P_n'' in x

```
      SUBROUTINE VALEPO(N,X,Y,DY,D2Y)
**************************************************************
*   COMPUTES THE VALUE OF THE LEGENDRE POLYNOMIAL OF DEGREE N
*   AND ITS FIRST AND SECOND DERIVATIVES AT A GIVEN POINT
*   N  = DEGREE OF THE POLYNOMIAL
*   X  = POINT IN WHICH THE COMPUTATION IS PERFORMED
*   Y  = VALUE OF THE POLYNOMIAL IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*   D2Y= VALUE OF THE SECOND DERIVATIVE IN X
**************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

         Y   = 1.D0
         DY  = 0.D0
         D2Y = 0.D0
      IF (N .EQ .0) RETURN

         Y   = X
         DY  = 1.D0
         D2Y = 0.D0
      IF(N .EQ. 1) RETURN

         YP   = 1.D0
         DYP  = 0.D0
         D2YP = 0.D0
      DO 1 I=2,N
         C1 = DFLOAT(I)
         C2 = 2.D0*C1-1.D0
         C4 = C1-1.D0
         YM = Y
         Y  = (C2*X*Y-C4*YP)/C1
         YP = YM
         DYM  = DY
         DY   = (C2*X*DY-C4*DYP+C2*YP)/C1
         DYP  = DYM
         D2YM = D2Y
         D2Y  = (C2*X*D2Y-C4*D2YP+2.D0*C2*DYP)/C1
         D2YP = D2YM
1     CONTINUE

      RETURN
      END
```

VACHPO

The routine computes the value of the Chebyshev polynomial T_n of degree $n \in \mathbf{N}$ and its first and second derivatives at the point x, according to the recursion formulas

$$\begin{cases} T_0(x) = 1, \\ T_1(x) = x, \\ T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x) \quad n \geq 2, \end{cases}$$

$$\begin{cases} T_0'(x) = 0, \\ T_1'(x) = 1, \\ T_n'(x) = 2xT_{n-1}'(x) + 2T_{n-1}(x) - T_{n-2}'(x) \quad n \geq 2, \end{cases}$$

$$\begin{cases} T_0''(x) = 0, \\ T_1''(x) = 0, \\ T_n''(x) = 2xT_{n-1}''(x) + 4T_{n-1}'(x) - T_{n-2}''(x) \quad n \geq 2. \end{cases}$$

Input variables	*Output variables*
N , the degree n X , the argument x	Y , the value of T_n in x DY , the value of T_n' in x $D2Y$, the value of T_n'' in x

```
      SUBROUTINE VACHPO(N,X,Y,DY,D2Y)
****************************************************************
*   COMPUTES THE VALUE OF THE CHEBYSHEV POLYNOMIAL OF DEGREE N
*   AND ITS FIRST AND SECOND DERIVATIVES AT A GIVEN POINT
*   N  = DEGREE OF THE POLYNOMIAL
*   X  = POINT IN WHICH THE COMPUTATION IS PERFORMED
*   Y  = VALUE OF THE POLYNOMIAL IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*   D2Y= VALUE OF THE SECOND DERIVATIVE IN X
****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

         Y   = 1.D0
         DY  = 0.D0
         D2Y = 0.D0
      IF (N .EQ. 0) RETURN

         Y   = X
         DY  = 1.D0
         D2Y = 0.D0
      IF (N .EQ. 1) RETURN

         YP   = 1.D0
         DYP  = 0.D0
         D2YP = 0.D0
      DO 1 K=2,N
         YM  = Y
         Y   = 2.D0*X*Y-YP
         YP  = YM
         DYM = DY
         DY  = 2.D0*X*DY+2.D0*YP-DYP
         DYP = DYM
         D2YM= D2Y
         D2Y = 2.D0*X*D2Y+4.D0*DYP-D2YP
         D2YP= D2YM
1     CONTINUE

      RETURN
      END
```

VALAPO

The routine computes the value of the Laguerre polynomial $L_n^{(\alpha)}$ of degree $n \in \mathbf{N}$ ($\alpha > -1$ is a real parameter) at the point x, according to the recursion formula

$$\begin{cases} L_0^{(\alpha)}(x) = 1, \\ L_1^{(\alpha)}(x) = 1 + \alpha - x, \\ L_n^{(\alpha)}(x) = \dfrac{1}{n}[(2n + \alpha - 1 - x)L_{n-1}^{(\alpha)}(x) - (n + \alpha - 1)L_{n-2}^{(\alpha)}(x)], \quad n \geq 2. \end{cases}$$

Simultaneously, first and second derivatives of $L_n^{(\alpha)}$ at the point x are computed by taking the derivatives of the terms of the above recursion formula.

Input variables	*Output variables*
N , the degree n	Y , the value of $L_n^{(\alpha)}$ in x
A , the parameter α	DY , the value of $\frac{d}{dx}L_n^{(\alpha)}$ in x
X , the argument x	$D2Y$, the value of $\frac{d^2}{dx^2}L_n^{(\alpha)}$ in x

```
      SUBROUTINE VALAPO(N,A,X,Y,DY,D2Y)
***************************************************************
*   COMPUTES THE VALUE OF THE LAGUERRE POLYNOMIAL OF DEGREE N
*   AND ITS FIRST AND SECOND DERIVATIVES AT A GIVEN POINT
*   N  = DEGREE OF THE POLYNOMIAL
*   A  = PARAMETER >-1
*   X  = POINT IN WHICH THE COMPUTATION IS PERFORMED
*   Y  = VALUE OF THE POLYNOMIAL IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*   D2Y= VALUE OF THE SECOND DERIVATIVE IN X
***************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

         Y   = 1.D0
         DY  = 0.D0
         D2Y = 0.D0
      IF (N .EQ. 0) RETURN

         Y   = 1.D0+A-X
         DY  = -1.D0
         D2Y = 0.D0
      IF (N .EQ. 1) RETURN

         YP  = 1.D0
         DYP = 0.D0
         D2YP= 0.D0
      DO 1 K=2,N
         DK = DFLOAT(K)
         B1 = (2.D0*DK+A-1.D0-X)/DK
         B2 = (DK+A-1.D0)/DK
         YM = Y
         Y  = B1*Y-B2*YP
         YP = YM
         DYM = DY
         DY  = B1*DY-YP/DK-B2*DYP
         DYP = DYM
         D2YM= D2Y
         D2Y = B1*D2Y-2.D0*DYP/DK-B2*D2YP
         D2YP= D2YM
 1    CONTINUE

      RETURN
      END
```

VAHEPO

The routine computes the value of the Hermite polynomial H_n, $n \in \mathbf{N}$, and its first and second derivatives at the point x, according to the recursion formulas

$$\begin{cases} H_0(x) = 1, \\ H_1(x) = 2x, \\ H_n(x) = 2xH_{n-1}(x) - 2(n-1)H_{n-2}(x) \qquad n \geq 2, \end{cases}$$

$$\begin{cases} H_0'(x) = 0, \\ H_n'(x) = 2nH_{n-1}(x) \qquad n \geq 1, \end{cases}$$

$$\begin{cases} H_0''(x) = 0, \\ H_1''(x) = 0, \\ H_n''(x) = 4n(n-1)H_{n-2}(x) \qquad n \geq 2. \end{cases}$$

Input variables	*Output variables*
N , the degree n	Y , the value of H_n in x
X , the argument x	DY , the value of H_n' in x
	$D2Y$, the value of H_n'' in x

```
      SUBROUTINE VAHEPO(N,X,Y,DY,D2Y)
*************************************************************
*   COMPUTES THE VALUE OF THE HERMITE POLYNOMIAL OF DEGREE N
*   AND ITS FIRST AND SECOND DERIVATIVES AT A GIVEN POINT
*   N  = DEGREE OF THE POLYNOMIAL
*   X  = POINT IN WHICH THE COMPUTATION IS PERFORMED
*   Y  = VALUE OF THE POLYNOMIAL IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*   D2Y= VALUE OF THE SECOND DERIVATIVE IN X
*************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

         Y   = 1.D0
         DY  = 0.D0
         D2Y = 0.D0
      IF (N .EQ. 0) RETURN

         Y   = 2.D0*X
         DY  = 2.D0
         D2Y = 0.D0
      IF (N .EQ. 1) RETURN

         YP=1.D0
      DO 1 K=2,N
         DK  = DFLOAT(K-1)
         YM  = Y
         Y   = 2.D0*X*Y-2.D0*DK*YP
         YPM = YP
         YP  = YM
1     CONTINUE
         DN  = 2.D0*DFLOAT(N)
         DNN = 2.D0*DFLOAT(N-1)
         DY  = DN*YP
         D2Y = DN*DNN*YPM

      RETURN
      END
```

VALASF

The routine computes the value at the point $x \geq 0$ of the *scaled Laguerre function* of degree $n \in \mathbf{N}$, defined for $\alpha > -1$ by

$$\begin{cases} \hat{L}_0^{(\alpha)}(x) &= L_0^{(\alpha)}(x), \\ \hat{L}_n^{(\alpha)}(x) &= \left[\dfrac{\Gamma(n+\alpha+1)}{n!\ \Gamma(\alpha+1)} \displaystyle\prod_{k=1}^{n} \left(1 + \frac{x}{4k}\right) \right]^{-1} L_n^{(\alpha)}(x) \qquad n \geq 1. \end{cases}$$

The following recursion formula is used:

$$\begin{cases} \hat{L}_0^{(\alpha)}(x) &= 1, \\ \hat{L}_1^{(\alpha)}(x) &= \dfrac{4(\alpha+1-x)}{(\alpha+1)(x+4)}, \\ \hat{L}_n^{(\alpha)}(x) &= \dfrac{4n}{(n+\alpha)(4n+x)} \left[(2n+\alpha-1-x)\hat{L}_{n-1}^{(\alpha)}(x) - \dfrac{4(n-1)^2}{4n+x-4} \hat{L}_{n-2}^{(\alpha)}(x) \right]. \end{cases}$$

Simultaneously, the first derivative of $\hat{L}_n^{(\alpha)}$ at the point x is computed by taking the derivative of the terms of the above recursion formula.

Scaled Laguerre functions will be used in place of Laguerre polynomials when possible, in order to improve the implementation (see the introduction).

Input variables	*Output variables*
N , the degree n	Y , the value of $\hat{L}_n^{(\alpha)}$ in x
A , the parameter α	DY , the value of $\frac{d}{dx}\hat{L}_n^{(\alpha)}$ in x
X , the argument x	

```
      SUBROUTINE VALASF(N,A,X,Y,DY)
*********************************************************************
*   COMPUTES THE VALUES OF THE SCALED LAGUERRE FUNCTION OF DEGREE N
*   AND ITS FIRST DERIVATIVE AT A GIVEN POINT
*   N  = DEGREE
*   A  = PARAMETER >-1
*   X  = POINT (NON NEGATIVE) IN WHICH THE COMPUTATION IS PERFORMED
*   Y  = VALUE OF THE FUNCTION IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

         Y  = 1.D0
         DY = 0.D0
      IF (N .EQ. 0) RETURN

         C0 = 1.D0/(4.D0+X)
         C1 = 4.D0*C0/(A+1.D0)
         Y  = C1*(A+1.D0-X)
         DY = -C0*C1*(A+5.D0)
      IF (N .EQ. 1) RETURN

         YP  = 1.D0
         DYP = 0.D0
      DO 1 K=2,N
         DK  = DFLOAT(K)
         DK4 = 4.D0*DK
         C0  = 1.D0/(DK4+X)
         C1  = DK4+X-2.D0
         C2  = 1.D0/(C1-2.D0)
         C3  = DK4*C0/(DK+A)
         C4  = 2.D0*DK+A-1.D0
         C5  = C4-X
         C6  = C4+DK4
         C7  = C2*DFLOAT(4*(K-1)**2)
         DYM = DY
         DY  = C3*(C5*DY-C0*C6*Y+C7*(2.D0*C0*C1*C2*YP-DYP))
         DYP = DYM
         YM  = Y
         Y   = C3*(C5*Y-C7*YP)
         YP  = YM
1     CONTINUE

      RETURN
      END
```

VAHESF

The routine computes the value and the first derivative at the point x of the *scaled Hermite function* of degree $n \in \mathbf{N}$ defined by

$$\hat{H}_n(x) = \hat{L}^{(-1/2)}_{n/2}(x^2) \qquad \text{if } n \text{ is even,}$$

$$\hat{H}_n(x) = x\hat{L}^{(1/2)}_{(n-1)/2}(x^2) \qquad \text{if } n \text{ is odd.}$$

Here, $\hat{L}^{(\alpha)}_n$, $k \in \mathbf{N}$, $\alpha > -1$, are the scaled Laguerre functions, defined in the description of subroutine VALASF.

Input variables	*Output variables*
N , the degree n	Y , the value of $\hat{H}_n$ in x
X , the argument x	DY , the value of $\hat{H}'_n$ in x

Auxiliary routine: **VALASF**

```
      SUBROUTINE VAHESF(N,X,Y,DY)
************************************************************
*   COMPUTES THE VALUES OF THE SCALED HERMITE FUNCTION
*   OF DEGREE N AND ITS FIRST DERIVATIVE AT A GIVEN POINT
*   N  = DEGREE
*   X  = POINT IN WHICH THE COMPUTATION IS PERFORMED
*   Y  = VALUE OF THE FUNCTION IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

         Y  = 1.D0
         DY = 0.D0
      IF (N .EQ. 0) RETURN

         XX = X*X
         M  = N/2
      IF(N .EQ. 2*M) THEN
         A  = -.5D0
      CALL VALASF(M,A,XX,Y,DY)
         DY = 2.D0*X*DY
      ELSE
         A  = .5D0
      CALL VALASF(M,A,XX,Y,DY)
         DY = Y+2.D0*XX*DY
         Y  = X*Y
      ENDIF

      RETURN
      END
```

ZEJAGA

For any $n \geq 1$, the routine computes the n zeroes $\xi_i^{(n)} \in]-1,1[$, $1 \leq i \leq n$, of the Jacobi polynomial $P_n^{(\alpha,\beta)}$, where the parameters satisfy $-\frac{1}{2} \leq \alpha \leq \frac{1}{2}$, $-\frac{1}{2} \leq \beta \leq \frac{1}{2}$.
The Newton method is used with the initial guess

$$\xi_i^{(n)} \approx -\cos\left(\frac{4i+\alpha+\beta-1}{2n+\alpha+\beta+1}\right)\frac{\pi}{2} \qquad 1 \leq i \leq n.$$

The values of the Jacobi polynomial and its derivative at a given point are obtained by the recursion formula given in the description of subroutine VAJAPO.
An output vector contains the zeroes in increasing order.
Another output vector contains the quantities $\frac{d}{dx}P_n^{(\alpha,\beta)}(\xi_i^{(n)})$, $1 \leq i \leq n$.
The zeroes of the Legendre polynomial P_n are obtained either by setting $\alpha = \beta = 0$ in this routine, or by calling subroutine ZELEGA. Similarly, the zeroes of the Chebyshev polynomial T_n are obtained either by setting $\alpha = \beta = -\frac{1}{2}$ in this routine, or by calling subroutine ZECHGA.

Input variables	*Output variables*
N , the degree n	CS , vector containing the zeroes $\xi_i^{(n)}$
A , the parameter α	DZ , vector of the values $\frac{d}{dx}P_n^{(\alpha,\beta)}(\xi_i^{(n)})$
B , the parameter β	

Auxiliary routine: **VAJAPO**

```
      SUBROUTINE ZEJAGA(N,A,B,CS,DZ)
****************************************************************
*   COMPUTES THE ZEROES OF THE JACOBI POLYNOMIAL OF DEGREE N
*   N  = THE NUMBER OF ZEROES
*   A  = PARAMETER BETWEEN -1/2 AND 1/2
*   B  = PARAMETER BETWEEN -1/2 AND 1/2
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   DZ = VECTOR OF THE DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1)
      IF (N .EQ .0) RETURN

         AB = A+B
         CS(1) = (B-A)/(AB+2.D0)
         DZ(1) = .5D0*AB+1.D0
      IF(N .EQ. 1) RETURN

         EPS= 1.E-17
         PH = 1.57079632679489661923D0
         C  = PH/(2.D0*DFLOAT(N)+AB+1.D0)
      DO 1 I=1,N
         DI  = DFLOAT(I)
         CSX = -DCOS(C*(4.D0*DI+AB-1.D0))
      DO 2 IT=1,8
      CALL VAJAPO(N,A,B,CSX,Y,DY,D2Y)
      IF(DABS(Y) .LT. EPS) GOTO 3
         CSX = CSX-Y/DY
2     CONTINUE
3     IF(DABS(CSX) .LT. EPS) CSX=0.D0
         CS(I) = CSX
         DZ(I) = DY
1     CONTINUE

      RETURN
      END
```

ZELEGA

For any $n \geq 1$, the routine computes the n zeroes $\xi_i^{(n)} \in]-1,1[,\ 1 \leq i \leq n$, of the Legendre polynomial P_n.

The Newton method is used with the initial guess

$$\xi_i^{(n)} \approx -\cos\left(\frac{4i-1}{2n+1}\right)\frac{\pi}{2} \qquad 1 \leq i \leq n.$$

The values of the Legendre polynomial and its derivative at a given point are obtained by the resursion formula given in the description of subroutine VALEPO.

An output vector contains the zeroes in increasing order.

Another output vector contains the quantities $P_n'(\xi_i^{(n)}),\ 1 \leq i \leq n$.

Input variable	*Output variables*
N , the degree n	CS , vector containing the zeroes $\xi_i^{(n)}$ DZ , vector of the values $P_n'(\xi_i^{(n)})$

Auxiliary routine: **VALEPO**

```
      SUBROUTINE ZELEGA(N,CS,DZ)
****************************************************************
*   COMPUTES THE ZEROES OF THE LEGENDRE POLYNOMIAL OF DEGREE N
*   N  = THE NUMBER OF ZEROES
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   DZ = VECTOR OF THE DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1)
      IF (N .EQ .0) RETURN

         CS(1) = 0.D0
         DZ(1) = 1.D0
      IF(N .EQ. 1) RETURN

         N2 = N/2
         IN = 2*N-4*N2-1
         PH = 1.57079632679489661923D0
         C  = PH/(2.D0*DFLOAT(N)+1.D0)
      DO 1 I=1,N2
         DI  = DFLOAT(I)
         CSX = DCOS(C*(4.D0*DI-1.D0))
      DO 2 IT=1,8
      CALL VALEPO(N,CSX,Y,DY,D2Y)
         CSX = CSX-Y/DY
2     CONTINUE
         CS(I) = -CSX
         CS(N-I+1) = CSX
         DZ(I) = DY*DFLOAT(IN)
         DZ(N-I+1) = DY
1     CONTINUE

      IF(IN .EQ. -1) RETURN
         CSX = 0.D0
         CS(N2+1) = CSX
      CALL VALEPO(N,CSX,Y,DY,D2Y)
         DZ(N2+1) = DY

      RETURN
      END
```

$\boxed{\textbf{ZECHGA}}$

For any $n \geq 1$, the routine gives the n zeroes

$$\xi_i^{(n)} = -\cos\frac{(2i-1)\pi}{2n} \in]-1,1[\qquad 1 \leq i \leq n,$$

of the Chebyshev polynomial T_n.

An output vector contains the zeroes in increasing order.

Another output vector contains the quantities

$$T_n'(\xi_i^{(n)}) = \frac{n(-1)^{n+i}}{\sqrt{1-\left[\xi_i^{(n)}\right]^2}} \qquad 1 \leq i \leq n.$$

Input variable	*Output variables*
N , the degree n	CS , vector containing the zeroes $\xi_i^{(n)}$ DZ , vector of the values $T_n'(\xi_i^{(n)})$

```
      SUBROUTINE ZECHGA(N,CS,DZ)
*****************************************************************
*   COMPUTES THE ZEROES OF THE CHEBYSHEV POLYNOMIAL OF DEGREE N
*   N  = THE NUMBER OF ZEROES
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   DZ = VECTOR OF THE DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
*****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1)
      IF (N .EQ. 0) RETURN

         CS(1) = 0.D0
         DZ(1) = 1.D0
      IF(N .EQ. 1) RETURN

         N2 = N/2
         IN = 1+4*N2-2*N
         PH = 1.57079632679489661923D0
         DN = DFLOAT(N)
         C  = PH/DN
         SI = -1.D0
      DO 1 I=1,N2
         DI = DFLOAT(I)
         CSX = DCOS(C*(2.D0*DI-1.D0))
         CS(I) = -CSX
         CS(N-I+1) = CSX
         QX = DN/DSQRT(1.D0-CSX*CSX)
         DZ(I) = QX*SI*DFLOAT(IN)
         DZ(N-I+1) = -QX*SI
         SI = -SI
1     CONTINUE

      IF(IN .EQ. 1) RETURN
         CS(N2+1) = 0.D0
         N4 = N2/2
         IN2 = 1+4*N4-2*N2
         DZ(N2+1) = DN*DFLOAT(IN2)

      RETURN
      END
```

ZELAGA

For any $n \geq 1$, the routine computes the n zeroes $\xi_i^{(n)} \in]0, +\infty[$, $1 \leq i \leq n$, of the Laguerre polynomial $L_n^{(\alpha)}$, $\alpha > -1$. These coincide with the zeroes of the scaled Laguerre function $\hat{L}_n^{(\alpha)}$, $\alpha > -1$, introduced in the description of subroutine VALASF. A better treatment of the rounding errors is realized by using scaled Laguerre functions in the computation.

The Newton method is used with the initial guess

$$\xi_i^{(n)} \approx 2(2n+\alpha+1)z_i - \frac{1}{6(2n+\alpha+1)}\left[\frac{5}{4(z_i-1)^2} + \frac{1}{z_i-1} - 1 + 3\alpha^2\right],$$

where

$$z_i = \left[\cos\left(\frac{y_i}{2}\right)\right]^2 \qquad 1 \leq i \leq n,$$

and

$$y_i - \sin y_i = 2\pi \frac{n-i+3/4}{2n+\alpha+1} \qquad 1 \leq i \leq n.$$

The values of the scaled Laguerre function $\hat{L}_n^{(\alpha)}$ and its derivative at a given point are obtained by the recursion formula given in the description of subroutine VALASF.

An output vector contains the zeroes in increasing order.

Another output vector contains the quantities $\frac{d}{dx}\hat{L}_n^{(\alpha)}(\xi_i^{(n)})$, $1 \leq i \leq n$.

Input variables	*Output variables*
N , the degree n	CS , vector containing the zeroes $\xi_i^{(n)}$
A , the parameter α	DZ , vector of the values $\frac{d}{dx}\hat{L}_n^{(\alpha)}(\xi_i^{(n)})$

Auxiliary routine: **VALASF**

```
      SUBROUTINE ZELAGA(N,A,CS,DZ)
**********************************************************************
*    COMPUTES THE ZEROES OF THE LAGUERRE POLYNOMIAL OF DEGREE N
*    N  = THE NUMBER OF ZEROES
*    A  = PARAMETER >-1
*    CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*    DZ = DERIVATIVES OF THE SCALED FUNCTIONS AT THE ZEROES, DZ(I), I=1,N
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1)
      IF (N .EQ. 0) RETURN

         A1 = A+1.D0
         CS(1) = A1
         DZ(1) = -4.D0/(A1*(A+5.D0))
      IF (N .EQ. 1) RETURN

         PI = 3.14159265358979323846D0
         DN = DFLOAT(N)
         C1 = 2.D0*DN+A1
      DO 1 M=1,N
          DM = DFLOAT(M)
          C2 = 2.D0*(DN+.75D0-DM)*PI/C1
          XN = (C2+PI)/2.D0
      DO 2 IT=1,8
          XP = XN
          XN = (DSIN(XP)-XP*DCOS(XP)+C2)/(1.D0-DCOS(XP))
2     CONTINUE
          Z  = (DCOS(XN/2.D0))**2
          ZD = 1.D0/(Z-1.D0)
          CSX= 2.D0*C1*Z-((1.25D0*ZD+1.D0)*ZD-1.D0+3.D0*A*A)/(6.D0*C1)
      DO 3 IT=1,6
      CALL VALASF(N,A,CSX,Y,DY)
          CSX = CSX-Y/DY
3     CONTINUE
          CS(M) = CSX
          DZ(M) = DY
1     CONTINUE
4         IN = 0
      DO 5 M=1,N-1
      IF(CS(M) .LE. CS(M+1)) GOTO 5
          CSM = CS(M)
          CS(M) = CS(M+1)
          CS(M+1) = CSM
          DZM = DZ(M)
          DZ(M) = DZ(M+1)
          DZ(M+1) = DZM
          IN = 1
5     CONTINUE
      IF(IN .EQ. 1) GOTO 4

      RETURN
      END
```

ZEHEGA

For any $n \geq 1$, the routine computes the n zeroes $\xi_i^{(n)} \in \mathbf{R}$, $1 \leq i \leq n$, of the Hermite polynomial H_n. These are related to the zeroes of the Laguerre polynomials by the formulas

$$L_{n/2}^{(-1/2)}([\xi_i^{(n)}]^2) = 0 \qquad 1 \leq i \leq n, \quad \text{if } n \text{ is even},$$

$$\xi_i^{(n)} L_{(n-1)/2}^{(1/2)}([\xi_i^{(n)}]^2) = 0 \quad 1 \leq i \leq n, \quad \text{if } n \text{ is odd},$$

Therefore, subroutine ZELAGA is called in this computation.
An output vector contains the zeroes in increasing order.
Another output vector contains the quantities $\hat{H}'_n(\xi_i^{(n)})$, $1 \leq i \leq n$, where $\hat{H}_n$ is the scaled Hermite functions of degree n, introduced in the description of subroutine VAHESF.

Input variable	*Output variables*
N , the degree n	CS , vector containing the zeroes $\xi_i^{(n)}$ DZ , vector of the values $\hat{H}'_n(\xi_i^{(n)})$

Auxiliary routines: **ZELAGA, VALASF**

```
      SUBROUTINE ZEHEGA(N,CS,DZ)
***************************************************************************
*   COMPUTES THE ZEROES OF THE HERMITE POLYNOMIAL OF DEGREE N
*   N  = THE NUMBER OF ZEROES
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   DZ = DERIVATIVES OF THE SCALED FUNCTIONS AT THE ZEROES, DZ(I), I=1,N
***************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1)
      IF (N .EQ. 0) RETURN

         CS(M+1) = 0.D0
         DZ(M+1) = 1.D0
      IF(N .EQ. 1) RETURN

         M = N/2
         IN = 2*N-4*M-1
      IF(IN .EQ. -1) THEN
         A = -.5D0
      CALL ZELAGA(M,A,CS,DZ)
      ELSE
         A = .5D0
      CALL ZELAGA(M,A,CS,DZ)
      ENDIF

      DO 1 I=1,M
         CSX = DSQRT(CS(M-I+1))
         CS(N-I+1) = CSX
      IF(IN .EQ. -1) THEN
         DZ(N-I+1) = 2.D0*CSX*DZ(M-I+1)
      ELSE
         DZ(N-I+1) = 2.D0*CSX*CSX*DZ(M-I+1)
      ENDIF
1     CONTINUE

      DO 2 I=1,M
         CS(I) = -CS(N-I+1)
         DZ(I) = DZ(N-I+1)*DFLOAT(IN)
2     CONTINUE

      RETURN
      END
```

ZEJAGL

For any $n \geq 1$, the routine computes the $n+1$ nodes $\eta_i^{(n)} \in [-1,1]$, $0 \leq i \leq n$, of the Jacobi Gauss-Lobatto formula

$$\sum_{i=0}^{n} f(\eta_i^{(n)})\ \tilde{w}_i^{(n)} \ \approx \ \int_{-1}^{1} f(x)(1-x)^{\alpha}(1+x)^{\beta} dx,$$

where the parameters satisfy $-\frac{1}{2} \leq \alpha \leq \frac{1}{2}$, $-\frac{1}{2} \leq \beta \leq \frac{1}{2}$. The quantities $\tilde{w}_i^{(n)}$, $0 \leq i \leq n$, are the weights of the formula.

One has $\eta_0^{(n)} = -1$, $\eta_n^{(n)} = 1$, and $\frac{d}{dx} P_n^{(\alpha,\beta)}(\eta_i^{(n)}) = 0$, $1 \leq i \leq n-1$. In particular, the nodes in $]-1,1[$ satisfy $P_{n-1}^{(\alpha+1,\beta+1)}(\eta_i^{(n)}) = 0$, $1 \leq i \leq n-1$. Hence, they are obtained with the same algorithm used for subroutine ZEJAGA.

An output vector contains the nodes in increasing order.

Another output vector contains the quantities $P_n^{(\alpha,\beta)}(\eta_i^{(n)})$, $0 \leq i \leq n$.

Input variables	*Output variables*
N , the degree n	ET , vector containing the nodes $\eta_i^{(n)}$
A , the parameter α	VN , vector of the values $P_n^{(\alpha,\beta)}(\eta_i^{(n)})$
B , the parameter β	

Auxiliary routine: **VAJAPO**

```
      SUBROUTINE ZEJAGL(N,A,B,ET,VN)
*********************************************************************
*   COMPUTES THE NODES RELATIVE TO THE JACOBI GAUSS-LOBATTO FORMULA
*   N  = ORDER OF THE FORMULA
*   A  = PARAMETER BETWEEN -1/2 AND 1/2
*   B  = PARAMETER BETWEEN -1/2 AND 1/2
*   ET = VECTOR OF THE NODES, ET(I), I=0,N
*   VN = VALUES OF THE JACOBI POLYNOMIAL AT THE NODES, VN(I), I=0,N
*********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*)
      IF (N .EQ. 0) RETURN

         ET(0) = -1.D0
         ET(N) = 1.D0
         X = -1.D0
      CALL VAJAPO(N,A,B,X,Y,DY,D2Y)
         VN(0) = Y
         X = 1.D0
      CALL VAJAPO(N,A,B,X,Y,DY,D2Y)
         VN(N) = Y
      IF (N .EQ. 1) RETURN

         EPS= 1.E-17
         PH = 1.57079632679489661923D0
         AB = A+B
         DN = DFLOAT(N)
         C  = PH/(2.D0*DN+AB+1.D0)
         N1 = N-1
         A1 = A+1.D0
         B1 = B+1.D0
      DO 1 I=1,N1
         DI  = DFLOAT(I)
         ETX = -DCOS(C*(4.D0*DI+AB+1.D0))
      DO 2 IT=1,8
      CALL VAJAPO(N1,A1,B1,ETX,Y,DY,D2Y)
      IF(DABS(Y) .LE. EPS) GOTO 3
         ETX = ETX-Y/DY
2     CONTINUE
3     IF(DABS(ETX) .LT. EPS) ETX=0.D0
         ET(I) = ETX
         VN(I) = -.5D0*DY*(1.D0-ETX*ETX)/DN
1     CONTINUE

      RETURN
      END
```

ZELEGL

For any $n \geq 1$, the routine computes the $n+1$ nodes $\eta_i^{(n)} \in [-1,1]$, $0 \leq i \leq n$, of the Legendre Gauss-Lobatto formula

$$\sum_{i=0}^{n} f(\eta_i^{(n)})\ \tilde{w}_i^{(n)} \ \approx \ \int_{-1}^{1} f(x)\ dx.$$

The quantities $\tilde{w}_i^{(n)}$, $0 \leq i \leq n$, are the weights of the formula.
One has $\eta_0^{(n)} = -1$, $\eta_n^{(n)} = 1$, and $\frac{d}{dx} P_n(\eta_i^{(n)}) = 0$, $1 \leq i \leq n-1$. For the nodes in $]-1,1[$, the Newton method is used with the initial guess $\eta_i^{(n)} \approx -\cos\frac{i\pi}{n}$, $1 \leq i \leq n-1$. The derivatives of the Legendre polynomial at a given point are obtained by the recursion formula given in the description of subroutine VALEPO.
An output vector contains the nodes in increasing order.
Another output vector contains the quantities $P_n(\eta_i^{(n)})$, $0 \leq i \leq n$.

Input variable	*Output variables*
N , the degree n	ET , vector containing the nodes $\eta_i^{(n)}$ VN , vector of the values $P_n(\eta_i^{(n)})$

<u>Auxiliary routine</u>: **VALEPO**

```
      SUBROUTINE ZELEGL(N,ET,VN)
***********************************************************************
*    COMPUTES THE NODES RELATIVE TO THE LEGENDRE GAUSS-LOBATTO FORMULA
*    N  = ORDER OF THE FORMULA
*    ET = VECTOR OF THE NODES, ET(I), I=0,N
*    VN = VALUES OF THE LEGENDRE POLYNOMIAL AT THE NODES, VN(I), I=0,N
***********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*)
      IF (N .EQ. 0) RETURN

         N2 = (N-1)/2
         SN = DFLOAT(2*N-4*N2-3)
         ET(0) = -1.D0
         ET(N) = 1.D0
         VN(0) = SN
         VN(N) = 1.D0
      IF (N .EQ. 1) RETURN

         ET(N2+1) = 0.D0
         X = 0.D0
      CALL VALEPO(N,X,Y,DY,D2Y)
         VN(N2+1) = Y
      IF(N .EQ. 2) RETURN

         PI = 3.14159265358979323846D0
         C  = PI/DFLOAT(N)
      DO 1 I=1,N2
         ETX = DCOS(C*DFLOAT(I))
      DO 2 IT=1,8
      CALL VALEPO(N,ETX,Y,DY,D2Y)
         ETX = ETX-DY/D2Y
2     CONTINUE
         ET(I) = -ETX
         ET(N-I) = ETX
         VN(I) = Y*SN
         VN(N-I) = Y
1     CONTINUE

      RETURN
      END
```

ZECHGL

For any $n \geq 1$, the routine computes the $n+1$ nodes $\eta_i^{(n)} \in [-1, 1]$, $0 \leq i \leq n$, of the Chebyshev Gauss-Lobatto formula

$$\sum_{i=0}^{n} f(\eta_i^{(n)})\ \tilde{w}_i^{(n)} \approx \int_{-1}^{1} f(x) \frac{dx}{\sqrt{1-x^2}}.$$

These, in increasing order, are given by

$$\eta_i^{(n)} = -\cos\frac{i\pi}{n} \qquad 0 \leq i \leq n.$$

Input variable	*Output variable*
N , the degree n	ET , vector containing the nodes $\eta_i^{(n)}$

```
      SUBROUTINE ZECHGL(N,ET)
************************************************************************
*   COMPUTES THE NODES RELATIVE TO THE CHEBYSHEV GAUSS-LOBATTO FORMULA
*   N  = ORDER OF THE FORMULA
*   ET = VECTOR OF THE NODES, ET(I), I=0,N
************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*)
      IF (N .EQ. 0) RETURN

         ET(0) = -1.D0
         ET(N) = 1.D0
      IF (N .EQ. 1) RETURN

         N2 = (N-1)/2
         ET(N2+1) = 0.D0
      IF(N .EQ. 2) RETURN

         PI = 3.14159265358979323846D0
         C  = PI/DFLOAT(N)
      DO 1 I=1,N2
         ETX = DCOS(C*DFLOAT(I))
         ET(I) = -ETX
         ET(N-I) = ETX
1     CONTINUE

      RETURN
      END
```

ZELAGR

For any $n \geq 1$, the routine computes the n nodes $\eta_i^{(n)} \in [0, +\infty[,\ 0 \leq i \leq n-1$, of the Laguerre Gauss-Radau formula

$$\sum_{i=0}^{n-1} f(\eta_i^{(n)})\ \tilde{w}_i^{(n)} \approx \int_0^{+\infty} f(x)\ x^\alpha e^{-x}\ dx,$$

where $\alpha > -1$. The quantities $\tilde{w}_i^{(n)}$, $0 \leq i \leq n-1$, are the weights of the formula. One has $\eta_0^{(n)} = 0$, and $\frac{d}{dx}L_n^{(\alpha)}(\eta_i^{(n)}) = 0,\ 1 \leq i \leq n-1$. The nodes in $]0, +\infty[$ satisfy $L_{n-1}^{(\alpha+1)}(\eta_i^{(n)}) = 0,\ 1 \leq i \leq n-1$. Hence, they are obtained with the same algorithm used for subroutine ZELAGA.

An output vector contains the nodes in increasing order.

Another output vector contains the quantities $\hat{L}_n^{(\alpha)}(\eta_i^{(n)}),\ 0 \leq i \leq n-1$, where $\hat{L}_n^{(\alpha)}$ is the scaled Laguerre functions of degree n, introduced in the description of subroutine VALASF.

Input variables	*Output variables*
N , the degree n	ET , vector containing the nodes $\eta_i^{(n)}$
A , the parameter α	VN , vector of the values $\hat{L}_n^{(\alpha)}(\eta_i^{(n)})$

Auxiliary routine: **VALASF**

```
      SUBROUTINE ZELAGR(N,A,ET,VN)
**********************************************************************
*   COMPUTES THE NODES RELATIVE TO THE LAGUERRE GAUSS-RADAU FORMULA
*   N  = ORDER OF THE FORMULA
*   A  = PARAMETER >-1
*   ET = VECTOR OF THE NODES, ET(I), O=1,N-1
*   VN = SCALED LAGUERRE FUNCTION AT THE NODES, VN(I), I=0,N-1
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*)
      IF (N .EQ. 0) RETURN
```

```
      ET(0) = 0.D0
      VN(0) = 1.D0
   IF (N .EQ. 1) RETURN

      A1 = A+1.D0
      PI = 3.14159265358979323846D0
      N1 = N-1
      DN = DFLOAT(N1)
      C1 = 2.D0*DN+A1+1.D0
   DO 1 M=1,N1
       DM = DFLOAT(M)
       C2 = 2.D0*(DN+.75D0-DM)*PI/C1
       XN = (C2+PI)/2.D0
   DO 2 IT=1,8
       XP = XN
       XN = (DSIN(XP)-XP*DCOS(XP)+C2)/(1.D0-DCOS(XP))
2      CONTINUE
       Z  = (DCOS(XN/2.D0))**2
       ZD = 1.D0/(Z-1.D0)
       ETX= 2.D0*C1*Z-((1.25D0*ZD+1.D0)*ZD-1.D0+3.D0*A1*A1)/(6.D0*C1)
   DO 3 IT=1,6
   CALL VALASF(N1,A1,ETX,Y,DY)
       ETX = ETX-Y/DY
3      CONTINUE
       ET(M) = ETX
   CALL VALASF(N,A,ETX,Y,DY)
       VN(M) = Y
1      CONTINUE

4          IN = 0
   DO 5 M=1,N-2
   IF(ET(M) .LE. ET(M+1)) GOTO 5
       ETM = ET(M)
       ET(M) = ET(M+1)
       ET(M+1) = ETM
       VNM = VN(M)
       VN(M) = VN(M+1)
       VN(M+1) = VNM
       IN = 1
5      CONTINUE
   IF(IN .EQ. 1) GOTO 4

   RETURN
   END
```

WEJAGA

For any $n \geq 1$, the routine computes the n weights $w_i^{(n)}$, $1 \leq i \leq n$, of the Jacobi Gauss integration formula

$$\sum_{i=1}^{n} f(\xi_i^{(n)})\, w_i^{(n)} \approx \int_{-1}^{1} f(x)(1-x)^{\alpha}(1+x)^{\beta} dx,$$

where $\alpha > -1$, $\beta > -1$, and $\xi_i^{(n)}$, $1 \leq i \leq n$, are the zeroes of the Jacobi polynomial $P_n^{(\alpha,\beta)}$. For $1 \leq i \leq n$, the following expression is used:

$$w_j^{(n)} = 2^{\alpha+\beta+1} \frac{\Gamma(n+\alpha+1)\,\Gamma(n+\beta+1)}{n!\ \Gamma(n+\alpha+\beta+1)} \frac{1}{1-[\xi_j^{(n)}]^2} \left[\frac{d}{dx} P_n^{(\alpha,\beta)}(\xi_j^{(n)})\right]^{-2}.$$

Both the zeroes and the quantities $\frac{d}{dx} P_n^{(\alpha,\beta)}(\xi_i^{(n)})$, $1 \leq i \leq n$, can be computed by calling subroutine ZEJAGA.

Input variables	*Output variable*
N , the degree n A , the parameter α B , the parameter β CS , vector of the zeroes $\xi_i^{(n)}$ DZ , the values $\frac{d}{dx}P_n^{(\alpha,\beta)}(\xi_i^{(n)})$	WE , vector containing the weights $w_i^{(n)}$

Auxiliary routine: **GAMMAF**

```
      SUBROUTINE WEJAGA(N,A,B,CS,DZ,WE)
****************************************************************
*   COMPUTES THE WEIGHTS RELATIVE TO THE JACOBI GAUSS FORMULA
*   N  = ORDER OF THE FORMULA
*   A  = PARAMETER > -1
*   B  = PARAMETER > -1
*   CS = ZEROES OF THE JACOBI POLYNOMIAL, CS(I), I=1,N
*   DZ = VECTOR OF THE DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
*   WE = VECTOR OF THE WEIGHTS, WE(I), I=1,N
****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1), WE(1)
      IF (N .EQ. 0) RETURN

          AB = A+B+2.D0
          A2 = A+2.D0
          B2 = B+2.D0
      CALL GAMMAF(A2,GA2)
      CALL GAMMAF(B2,GB2)
      CALL GAMMAF(AB,GAB)
          C  = .5D0*(2.D0**AB)*GA2*GB2/GAB
      DO 1 M=2,N
          DM = DFLOAT(M)
          C  = C*(DM+A)*(DM+B)/(DM*(DM+A+B))
1     CONTINUE

      DO 2 I=1,N
          X  = CS(I)
          DY = DZ(I)
          WE(I) = C/((1.D0-X*X)*DY*DY)
2     CONTINUE

      RETURN
      END
```

WELEGA

For any $n \geq 1$, the routine computes the n weights $w_i^{(n)}$, $1 \leq i \leq n$, of the Legendre Gauss integration formula

$$\sum_{i=1}^{n} f(\xi_i^{(n)})\, w_i^{(n)} \approx \int_{-1}^{1} f(x)\, dx,$$

where $\xi_i^{(n)}$, $1 \leq i \leq n$, are the zeroes of the Legendre polynomial P_n.

The following expression is used:

$$w_i^{(n)} = \frac{2}{1 - [\xi_i^{(n)}]^2} \left[P_n'(\xi_i^{(n)})\right]^{-2}, \qquad 1 \leq i \leq n.$$

Both the zeroes and the quantities $P_n'(\xi_i^{(n)})$, $1 \leq i \leq n$, can be computed by calling subroutine ZELEGA.

Input variables	*Output variable*
N , the degree n CS , vector of the zeroes of P_n DZ , values of P_n' at the zeroes	WE , vector of the weights $w_i^{(n)}$

```
      SUBROUTINE WELEGA(N,CS,DZ,WE)
*****************************************************************
*   COMPUTES THE WEIGHTS RELATIVE TO THE LEGENDRE GAUSS FORMULA
*   N  = ORDER OF THE FORMULA
*   CS = ZEROES OF THE LEGENDRE POLYNOMIAL, CS(I), I=1,N
*   DZ = VECTOR OF THE DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
*   WE = VECTOR OF THE WEIGHTS, WE(I), I=1,N
*****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1), WE(1)
      IF (N .EQ. 0) RETURN

          N2 = N/2
      DO 1 I=1,N2
          X  = CS(I)
          DY = DZ(I)
          WEX = 2.D0/((1.D0-X*X)*DY*DY)
          WE(I) = WEX
          WE(N-I+1) = WEX
1     CONTINUE

      IF(N .EQ. 2*N2) RETURN
          DY = DZ(N2+1)
          WE(N2+1) = 2.D0/(DY*DY)

      RETURN
      END
```

WECHGA

For any $n \geq 1$, the routine computes the n weights $w_i^{(n)}$, $1 \leq i \leq n$, of the Chebyshev Gauss integration formula

$$\sum_{i=1}^{n} f(\xi_i^{(n)})\; w_i^{(n)} \;\approx\; \int_{-1}^{1} f(x) \frac{dx}{\sqrt{1-x^2}},$$

where $\xi_i^{(n)}$, $1 \leq i \leq n$, are the zeroes of the Chebyshev polynomial T_n.

The weights are explicity given by

$$w_i^{(n)} \;=\; \frac{\pi}{n} \qquad 1 \leq i \leq n.$$

Input variable	*Output variable*
N , the degree n	WE , vector containing the weights $w_i^{(n)}$

```
      SUBROUTINE WECHGA(N,WE)
******************************************************************
*   COMPUTES THE WEIGHTS RELATIVE TO THE CHEBYSHEV GAUSS FORMULA
*   N  = ORDER OF THE FORMULA
*   WE = VECTOR OF THE WEIGHTS, WE(I), I=1,N
******************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION  WE(1)
      IF (N .EQ. 0) RETURN

         PI = 3.14159265358979323846D0
         C  = PI/DFLOAT(N)
      DO 1 I=1,N
         WE(I) = C
1     CONTINUE

      RETURN
      END
```

WELAGA

For any $n \geq 1$, the routine computes the n weights $w_i^{(n)}$, $1 \leq i \leq n$, of the Laguerre Gauss integration formula

$$\sum_{i=1}^{n} f(\xi_i^{(n)})\ w_i^{(n)} \ \approx \ \int_0^{+\infty} f(x)\ x^\alpha e^{-x}\ dx,$$

where $\alpha > -1$, and $\xi_i^{(n)}$, $1 \leq i \leq n$, are the zeroes of the Laguerre polynomial $L_n^{(\alpha)}$, which can be computed by subroutine ZELAGA.

The following expression is used:

$$w_i^{(n)} \ = \ \frac{\Gamma(n+\alpha+1)\ \xi_i^{(n)}}{(n+1)!\ \ (n+1)} \left[L_{n+1}^{(\alpha)}(\xi_i^{(n)}) \right]^{-2}, \qquad 1 \leq i \leq n.$$

Input variables	*Output variable*
N , the degree n A , the parameter α CS , vector of the zeroes of $L_n^{(\alpha)}$	WE , vector of the weights $w_i^{(n)}$

Auxiliary routines: **GAMMAF, VALAPO**

```
      SUBROUTINE WELAGA(N,A,CS,WE)
*****************************************************************
*   COMPUTES THE WEIGHTS RELATIVE TO THE LAGUERRE GAUSS FORMULA
*   N  = ORDER OF THE FORMULA
*   A  = PARAMETER > -1
*   CS = ZEROES OF THE LAGUERRE POLYNOMIAL, CS(I), I=1,N
*   WE = VECTOR OF THE WEIGHTS, WE(I), I=1,N
*****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), WE(1)
      IF (N .EQ. 0) RETURN

          A1 = A+1.D0
      CALL GAMMAF(A1,GA1)
          N1 = N+1
          DN = DFLOAT(N1)
          C = GA1/DN
      DO 1 M=1,N
          DM = DFLOAT(M)
          C = C*(DM+A)/(DM+1.D0)
1     CONTINUE

      DO 2 I=1,N
          X =CS(I)
      CALL VALAPO(N1,A,X,Y,DY,D2Y)
          WE(I) = C*X/(Y*Y)
2     CONTINUE

      RETURN
      END
```

WEHEGA

For any $n \geq 1$, the routine computes the n weights $w_i^{(n)}$, $1 \leq i \leq n$, of the Hermite Gauss integration formula

$$\sum_{i=1}^{n} f(\xi_i^{(n)})\, w_i^{(n)} \;\approx\; \int_{-\infty}^{+\infty} f(x)\, e^{-x^2} dx,$$

where $\xi_i^{(n)}$, $1 \leq i \leq n$, are the zeroes of the Hermite polynomial H_n, which can be computed by subroutine ZEHEGA.

For $1 \leq i \leq n$, the following expression is used:

$$w_i^{(n)} \;=\; \frac{\sqrt{\pi}}{n}\, 2^{n-1}\, (n-1)!\, \left[H_{n-1}(\xi_i^{(n)})\right]^{-2} \;=\; \frac{\sqrt{\pi}}{n} \left[\tilde{H}_{n-1}(\xi_i^{(n)})\right]^{-2},$$

where $\tilde{H}_k = (2^k k!)^{-1/2} H_k$, $k \in \mathbf{N}$, is obtained by recursion

$$\begin{cases} \tilde{H}_0(x) \;=\; 1, \\ \tilde{H}_1(x) \;=\; \sqrt{2}x, \\ \tilde{H}_k(x) \;=\; \sqrt{\frac{2}{k}}x\tilde{H}_{k-1}(x) - \sqrt{\frac{k-1}{k}}\tilde{H}_{k-2}(x), \quad k \geq 2. \end{cases}$$

Input variables	*Output variable*
N , the degree n CS , vector of the zeroes of H_n	WE , vector of the weights $w_i^{(n)}$

```
      SUBROUTINE WEHEGA(N,CS,WE)
*****************************************************************
*   COMPUTES THE WEIGHTS RELATIVE TO THE HERMITE GAUSS FORMULA
*   N  = ORDER OF THE FORMULA
*   CS = ZEROES OF THE HERMITE POLYNOMIAL, CS(I), I=1,N
*   WE = VECTOR OF THE WEIGHTS, WE(I), I=1,N
*****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), WE(1)
      IF (N .EQ. 0) RETURN

          PR = 1.77245385090551588D0
          R2 = 1.41421356237309515D0
          C  = PR/DFLOAT(N)
          N2 = N/2
      DO 1 I=1,N2
          X  = CS(I)
          YP = 1.D0
          Y  = R2*X
      DO 2 K=2,N-1
          DK = DFLOAT(K)
          RK = DSQRT(DK)
          QK = DSQRT(DK-1.D0)
          YM = Y
          Y  = (R2*X*Y-QK*YP)/RK
          YP = YM
2     CONTINUE
          WEX = C/(Y*Y)
          WE(I) = WEX
          WE(N-I+1) = WEX
1     CONTINUE

      IF(N .EQ. 2*N2) RETURN
          Y = 1.D0
      DO 3 K=2,N-1,2
          DK = DFLOAT(K)
          Y  = Y*DSQRT((DK-1.D0)/DK)
3     CONTINUE
          WE(N2+1) = C/(Y*Y)

      RETURN
      END
```

WEJAGL

For any $n \geq 1$, the routine computes the $n+1$ weights $\tilde{w}_j^{(n)}$, $0 \leq j \leq n$, of the Jacobi Gauss-Lobatto formula of order n, introduced in the description of subroutine ZEJAGL.

For $n = 1$, one has $\tilde{w}_0^{(n)} = 2^{\alpha+\beta+1} \frac{\Gamma(\alpha+2)\,\Gamma(\beta+1)}{\Gamma(\alpha+\beta+3)}$, $\tilde{w}_n^{(n)} = 2^{\alpha+\beta+1} \frac{\Gamma(\alpha+1)\,\Gamma(\beta+2)}{\Gamma(\alpha+\beta+3)}$.

For $n \geq 2$, the following expression is used:

$$\tilde{w}_j^{(n)} = \begin{cases} K_n(\alpha,\beta) \sum_{m=1}^{n-1} (1 - \eta_m^{(n)}) & \text{if } j = 0, \\ \tilde{K}_n(\alpha,\beta) \left[P_n^{(\alpha,\beta)}(\eta_j^{(n)}) \, \frac{d}{dx} P_{n-1}^{(\alpha,\beta)}(\eta_j^{(n)}) \right]^{-1} & \text{if } 1 \leq j \leq n-1, \\ K_n(\beta,\alpha) \sum_{m=1}^{n-1} (1 + \eta_m^{(n)}) & \text{if } j = n, \end{cases}$$

with

$$K_n(\alpha,\beta) = 2^{\alpha+\beta} \, \frac{\Gamma(\beta+1)\,\Gamma(\beta+2)\,(2n+\alpha+\beta)\,(n-2)!\,\Gamma(n+\alpha)}{\Gamma(n+\alpha+\beta+2)\,\Gamma(n+\beta+1)},$$

$$\tilde{K}_n(\alpha,\beta) = -2^{\alpha+\beta} \, \frac{(2n+\alpha+\beta)\,\Gamma(n+\alpha)\,\Gamma(n+\beta)}{(n+\alpha+\beta+1)\,n!\,\Gamma(n+\alpha+\beta)},$$

where $\alpha > -1$, $\beta > -1$, and $\eta_j^{(n)}$, $0 \leq j \leq n$, are the Gauss-Lobatto nodes, i.e., $\eta_0^{(n)} = -1$, $\eta_n^{(n)} = 1$, and $\frac{d}{dx} P_n^{(\alpha,\beta)}(\eta_j^{(n)}) = 0$, $1 \leq j \leq n-1$, which can be obtained by calling subroutine ZEJAGL.

Input variables	*Output variable*
N , the degree n A , the parameter α B , the parameter β ET , vector of the nodes $\eta_j^{(n)}$	WT , vector containing the weights $\tilde{w}_i^{(n)}$

Auxiliary routines: **GAMMAF, VAJAPO**

```
      SUBROUTINE WEJAGL(N,A,B,ET,WT)
*********************************************************************
*   COMPUTES THE WEIGHTS RELATIVE TO THE JACOBI GAUSS-LOBATTO FORMULA
*   N  = ORDER OF THE FORMULA
*   A  = PARAMETER > -1
*   B  = PARAMETER > -1
*   ET = JACOBI GAUSS-LOBATTO NODES, ET(I), I=0,N
*   WT = VECTOR OF THE WEIGHTS, WT(I), I=0,N
*********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), WT(0:*)
      IF (N .EQ. 0) RETURN

          A1  = A+1.D0
          B1  = B+1.D0
          AB  = A+B
          AB2 = A1+B1
      CALL GAMMAF(A1,GA1)
      CALL GAMMAF(B1,GB1)
      CALL GAMMAF(AB2,GAB2)
          C = (2.D0**AB)*GA1*GB1/GAB2
          WT(0) = 2.D0*C*A1/AB2
          WT(N) = 2.D0*C*B1/AB2
      IF (N .EQ. 1) RETURN

          N1 = N-1
          DN = DFLOAT(N)
          C  = C*(2.D0*DN+AB)/(DN+AB+1.D0)
          C1 = C*A1/((B+2.D0)*AB2)
          C2 = C*B1/((A+2.D0)*AB2)
          C3 = .5D0*C*A1*B1
      DO 1 K=1,N-2
          DK = DFLOAT(K)
          C1 = C1*(DK+A1)*DK/((DK+AB2)*(DK+B+2.D0))
          C2 = C2*(DK+B1)*DK/((DK+AB2)*(DK+A+2.D0))
          C3 = C3*(DK+A1)*(DK+B1)/((DK+2.D0)*(DK+AB+1.D0))
1     CONTINUE

          SU = 0.D0
      DO 2 M=1,N1
          SU = SU+ET(M)
2     CONTINUE
          WT(0) = C1*(DN-1.D0-SU)
          WT(N) = C2*(DN-1.D0+SU)
      DO 3 I=1,N1
          X = ET(I)
      CALL VAJAPO(N,A,B,X,Y,DY,D2Y)
          C4 = -C3/Y
      CALL VAJAPO(N1,A,B,X,Y,DY,D2Y)
          WT(I) = C4/DY
3     CONTINUE
      RETURN
      END
```

WELEGL

For any $n \geq 1$, the routine computes the $n+1$ weights $\tilde{w}_j^{(n)}$, $0 \leq j \leq n$, of the Legendre Gauss-Lobatto formula of order n, introduced in the description of subroutine ZELEGL.

The following expression is used:

$$\tilde{w}_j^{(n)} = \frac{2}{n(n+1)}[P_n(\eta_j^{(n)})]^{-2}, \qquad 0 \leq j \leq n,$$

where $\eta_j^{(n)}$, $0 \leq j \leq n$, are the Gauss-Lobatto nodes, i.e., $\eta_0^{(n)} = -1$, $\eta_n^{(n)} = 1$, and $\frac{d}{dx}P_n(\eta_j^{(n)}) = 0$, $1 \leq j \leq n-1$. Both the nodes and the quantities $P_n(\eta_j^{(n)})$, $0 \leq j \leq n$, can be computed by calling subroutine ZELEGL.

Input variables	*Output variable*
N , the degree n ET , vector of the nodes $\eta_j^{(n)}$ VN , values of P_n at the nodes	WT , vector containing the weights $\tilde{w}_j^{(n)}$

```
      SUBROUTINE WELEGL(N,ET,VN,WT)
*************************************************************************
*   COMPUTES THE WEIGHTS RELATIVE TO THE LEGENDRE GAUSS-LOBATTO FORMULA
*   N  = ORDER OF THE FORMULA
*   ET = LEGENDRE GAUSS-LOBATTO NODES, ET(I), I=0,N
*   VN = VALUES OF THE LEGENDRE POLYNOMIAL AT THE NODES, VN(I), I=0,N
*   WT = VECTOR OF THE WEIGHTS, WT(I), I=0,N
*************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*), WT(0:*)
      IF (N .EQ. 0) RETURN

          N2 = (N-1)/2
          DN = DFLOAT(N)
          C  = 2.D0/(DN*(DN+1.D0))
      DO 1 I=0,N2
          X = ET(I)
          Y = VN(I)
          WTX = C/(Y*Y)
          WT(I) = WTX
          WT(N-I) = WTX
1     CONTINUE

      IF(N-1 .EQ. 2*N2) RETURN
          X = 0.D0
          Y = VN(N2+1)
          WT(N2+1) = C/(Y*Y)

      RETURN
      END
```

WECHGL

For any $n \geq 1$, the routine computes the $n+1$ weights $\tilde{w}_j^{(n)}$, $0 \leq j \leq n$, of the Chebyshev Gauss-Lobatto formula of order n, introduced in the description of subroutine ZECHGL.

These are given by

$$\tilde{w}_0^{(n)} = \tilde{w}_n^{(n)} = \frac{\pi}{2n}, \qquad \tilde{w}_j^{(n)} = \frac{\pi}{n} \quad 1 \leq j \leq n-1.$$

Input variable	*Output variable*
N , the degree n	WT , vector containing the weights $\tilde{w}_j^{(n)}$

```
      SUBROUTINE WECHGL(N,WT)
*************************************************************************
*   COMPUTES THE WEIGHTS RELATIVE TO THE CHEBYSHEV GAUSS-LOBATTO FORMULA
*   N  = ORDER OF THE FORMULA
*   WT = VECTOR OF THE WEIGHTS, WT(I), I=0,N
*************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION  WT(0:*)
      IF (N .EQ. 0) RETURN

          PI = 3.14159265358979323846D0
          C  = PI/DFLOAT(N)
          C2 = .5D0*C
          WT(0) =C2
          WT(N) =C2
      IF (N .EQ. 1) RETURN

      DO 1 I=1,N-1
          WT(I) = C
1     CONTINUE

      RETURN
      END
```

WELAGR

For any $n \geq 1$, the routine computes the n weights $\tilde{w}_j^{(n)}$, $0 \leq j \leq n-1$, of the Laguerre Gauss-Radau formula of order n, introduced in the description of subroutine ZELAGR.

The following expression is used:

$$\tilde{w}_j^{(n)} = \begin{cases} \frac{(\alpha+1)\ \Gamma^2(\alpha+1)\ (n-1)!}{\Gamma(n+\alpha+1)} & \text{if } j = 0, \\ \frac{\Gamma(n+\alpha)}{n!} \left[L_n^{(\alpha)}(\eta_j^{(n)})\ \frac{d}{dx} L_{n-1}^{(\alpha)}(\eta_j^{(n)}) \right]^{-1} & \text{if } 1 \leq j \leq n-1, \end{cases}$$

where $\alpha > -1$, and $\eta_j^{(n)}$, $0 \leq j \leq n-1$, are the Gauss-Radau nodes, i.e., $\eta_0^{(n)} = 0$ and $\frac{d}{dx} L_n^{(\alpha)}(\eta_j^{(n)}) = 0$, $1 \leq j \leq n-1$, which can be computed by calling subroutine ZELAGR.

Input variables	*Output variable*
N , the degree n A , the parameter α ET , vector of the nodes $\eta_j^{(n)}$	WT , vector containing the weights $\tilde{w}_j^{(n)}$

Auxiliary routines: **GAMMAF, VALAPO**

```
      SUBROUTINE WELAGR(N,A,ET,WT)
***********************************************************************
*   COMPUTES THE WEIGHTS RELATIVE TO THE LAGUERRE GAUSS-RADAU FORMULA
*   N  = ORDER OF THE FORMULA
*   A  = PARAMETER > -1
*   ET = LAGUERRE GAUSS-RADAU NODES, ET(I), I=0,N-1
*   WT = VECTOR OF THE WEIGHTS, WT(I), I=0,N-1
***********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), WT(0:*)
      IF (N .EQ. 0) RETURN

          A1 = A+1.D0
      CALL GAMMAF(A1,GA1)
          C1 = GA1
          WT(0) = C1
      IF (N .EQ. 1) RETURN

          N1 = N-1
          C2 = GA1
      DO 1 K=1,N1
          DK = DFLOAT(K)
          C1 = C1*DK/(DK+A1)
          C2 = C2*(DK+A)/(DK+1.D0)
1     CONTINUE
          WT(0) = C1
      DO 2 I=1,N1
          X = ET(I)
      CALL VALAPO(N,A,X,Y,DY,D2Y)
          C3 = C2/Y
      CALL VALAPO(N1,A,X,Y,DY,D2Y)
          WT(I) = C3/DY
2     CONTINUE

      RETURN
      END
```

WECHCC

For any $n \geq 1$, the routine computes the $2n+1$ weights $\chi_j^{(2n)}$, $0 \leq j \leq 2n$, of the Clenshaw-Curtis formula of order $2n$

$$\int_{-1}^{1} f \, dx \approx \sum_{j=0}^{2n} f\left(\cos \frac{j\pi}{2n}\right) \chi_j^{(2n)}.$$

The following expression is used:

$$\chi_j^{(2n)} = \begin{cases} \dfrac{1}{4n^2-1} & \text{if } j=0, \\[2ex] \dfrac{1}{n}\left[1 - \dfrac{(-1)^j}{4n^2-1} + \displaystyle\sum_{k=1}^{n-1} \dfrac{2}{1-4k^2} \cos \dfrac{kj\pi}{n}\right] & \text{if } 1 \leq j \leq n, \\[2ex] \chi_{2n-j}^{(2n)} & \text{if } n+1 \leq j \leq 2n. \end{cases}$$

Input variable	*Output variable*
N , the parameter n	WK , vector containing the weights $\chi_j^{(2n)}$

```
      SUBROUTINE WECHCC(N,WK)
**********************************************************************
*   COMPUTES THE WEIGHTS OF THE CLENSHAW-CURTIS FORMULA OF ORDER 2*N
*   N  = INTEGER PARAMETER
*   WK = VECTOR OF THE WEIGHTS, WK(I), I=0,2*N
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION WK(0:*)
      IF (N .EQ. 0) RETURN

         PI = 3.14159265358979323846D0
         M  = 2*N
         DN = DFLOAT(N)
         DM = DFLOAT(M)
         C  = 1.D0/(DM*DM-1.D0)
         WK(0) = C
         WK(M) = C
         WK(N) = 1.33333333333333333333D0
      IF (N .EQ. 1) RETURN

      DO 1 J=1,N-1
          DJ = DFLOAT(J)
          SU = 1.D0-((-1.D0)**J)*C
      DO 2 K=1,N-1
          DK = 2.D0*DFLOAT(K)
          SU = SU+2.D0*DCOS(DJ*DK*PI/DM)/(1.D0-DK*DK)
2     CONTINUE
          WK(J) = SU/DN
          WK(M-J) = SU/DN
1     CONTINUE

          SU = 1.D0-((-1.D0)**N)*C
      DO 3 K=1,N-1
          DK = 2.D0*DFLOAT(K)
          SU = SU+2.D0*((-1.D0)**K)/(1.D0-DK*DK)
3     CONTINUE
          WK(N) = SU/DN

      RETURN
      END
```

INJAGA

For any $n \geq 1$, the routine computes the value at the point x of a polynomial q of degree at most $n-1$, determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the Jacobi polynomial $P_n^{(\alpha,\beta)}$.

One has

$$q(x) \;=\; \sum_{j=1}^{n} q(\xi_j^{(n)})\; l_j^{(n)}(x),$$

where for $1 \leq j \leq n$

$$l_j^{(n)}(x) \;=\; \begin{cases} \left[\dfrac{d}{dx} P_n^{(\alpha,\beta)}(\xi_j^{(n)})\right]^{-1} \dfrac{P_n^{(\alpha,\beta)}(x)}{x - \xi_j^{(n)}} & \text{if } x \neq \xi_j^{(n)}, \\ \\ 1 & \text{if } x = \xi_j^{(n)}. \end{cases}$$

The zeroes and the values $\frac{d}{dx} P_n^{(\alpha,\beta)}(\xi_j^{(n)})$, $1 \leq j \leq n$, can be obtained by calling subroutine ZEJAGA.

Input variables	*Output variable*
N , the degree n A , the parameter α B , the parameter β CS , vector of the zeroes $\xi_j^{(n)}$ DZ , the values $\frac{d}{dx}P_n^{(\alpha,\beta)}(\xi_j^{(n)})$ QZ , values of q at the zeroes X , the point x	QX , the value of q in x

Auxiliary routine: **VAJAPO**

```
      SUBROUTINE INJAGA(N,A,B,CS,DZ,QZ,X,QX)
*********************************************************************
*   COMPUTES THE VALUE AT A GIVEN POINT OF A POLYNOMIAL INDIVIDUATED
*   BY THE VALUES ATTAINED AT THE ZEROES OF THE JACOBI POLYNOMIAL
*   N  = THE NUMBER OF ZEROES
*   A  = PARAMETER > -1
*   B  = PARAMETER > -1
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   DZ = JACOBI POLYNOMIAL DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   QX = VALUE OF THE POLYNOMIAL IN X
*********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1), QZ(1)
      IF (N .EQ. 0) RETURN

          EPS = 1.D-14
      CALL VAJAPO(N,A,B,X,Y,DY,D2Y)
          QX = 0.D0
      DO 1 J=1,N
          ED = X-CS(J)
      IF(DABS(ED) .LT. EPS) THEN
          QX = QZ(J)
      RETURN
      ELSE
          QX = QX+QZ(J)*Y/(DZ(J)*ED)
      ENDIF
1     CONTINUE

      RETURN
      END
```

INLEGA

For any $n \geq 1$, the routine computes the value at the point x of a polynomial q of degree at most $n-1$, determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the Legendre polynomial P_n.

One has

$$q(x) \ = \ \sum_{j=1}^{n} q(\xi_j^{(n)}) \ l_j^{(n)}(x),$$

where for $1 \leq j \leq n$

$$l_j^{(n)}(x) \ = \begin{cases} \left[P_n'(\xi_j^{(n)})\right]^{-1} \dfrac{P_n(x)}{x - \xi_j^{(n)}} & \text{if } x \neq \xi_j^{(n)}, \\ 1 & \text{if } x = \xi_j^{(n)}. \end{cases}$$

The zeroes and the values $P_n'(\xi_j^{(n)})$, $1 \leq j \leq n$, can be obtained by calling subroutine ZELEGA.

Input variables	*Output variable*
N , the degree n CS , vector of the zeroes $\xi_j^{(n)}$ DZ , the values $P_n'(\xi_j^{(n)})$ QZ , values of q at the zeroes X , the point x	QX , the value of q in x

Auxiliary routine: **VALEPO**

```
      SUBROUTINE INLEGA(N,CS,DZ,QZ,X,QX)
**********************************************************************
*   COMPUTES THE VALUE AT A GIVEN POINT OF A POLYNOMIAL INDIVIDUATED
*   BY THE VALUES ATTAINED AT THE ZEROES OF THE LEGENDRE POLYNOMIAL
*   N  = THE NUMBER OF ZEROES
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   DZ = LEGENDRE POLYNOMIAL DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   QX = VALUE OF THE POLYNOMIAL IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1), QZ(1)
      IF (N .EQ. 0) RETURN

          EPS = 1.D-14
      CALL VALEPO(N,X,Y,DY,D2Y)
          QX = 0.D0
      DO 1 J=1,N
          ED = X-CS(J)
      IF(DABS(ED) .LT. EPS) THEN
          QX = QZ(J)
      RETURN
      ELSE
          QX = QX+QZ(J)*Y/(DZ(J)*ED)
      ENDIF
1     CONTINUE

      RETURN
      END
```

INCHGA

For any $n \geq 1$, the routine computes the value at the point x of a polynomial q of degree at most $n-1$, determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the Chebyshev polynomial T_n.

One has

$$q(x) \;=\; \sum_{j=1}^{n} q(\xi_j^{(n)})\; l_j^{(n)}(x),$$

where for $1 \leq j \leq n$

$$l_j^{(n)}(x) \;=\; \begin{cases} \left[T_n'(\xi_j^{(n)})\right]^{-1} \dfrac{T_n(x)}{x-\xi_j^{(n)}} & \text{if } x \neq \xi_j^{(n)}, \\[2ex] 1 & \text{if } x = \xi_j^{(n)}. \end{cases}$$

The values $T_n'(\xi_j^{(n)})$, $1 \leq j \leq n$, can be obtained by calling subroutine ZECHGA.

Input variables	*Output variable*
N , the degree n DZ , the values $T_n'(\xi_j^{(n)})$ QZ , values of q at the zeroes X , the point x	QX , the value of q in x

Auxiliary routine: **VACHPO**

```
      SUBROUTINE INCHGA(N,DZ,QZ,X,QX)
*********************************************************************
*   COMPUTES THE VALUE AT A GIVEN POINT OF A POLYNOMIAL INDIVIDUATED
*   BY THE VALUES ATTAINED AT THE ZEROES OF THE CHEBYSHEV POLYNOMIAL
*   N  = THE NUMBER OF ZEROES
*   DZ = CHEBYSHEV POLYNOMIAL DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   QX = VALUE OF THE POLYNOMIAL IN X
*********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION  DZ(1), QZ(1)
      IF (N .EQ. 0) RETURN

          EPS = 1.D-14
          PH = 1.57079632679489661923D0
          DN = DFLOAT(N)
          C  = PH/DN
      CALL VACHPO(N,X,Y,DY,D2Y)
          QX = 0.D0
      DO 1 J=1,N
          ED = X+DCOS(C*(2.D0*DFLOAT(J)-1.D0))
      IF(DABS(ED) .LT. EPS) THEN
          QX = QZ(J)
      RETURN
      ELSE
          QX = QX+QZ(J)*Y/(DZ(J)*ED)
      ENDIF
1     CONTINUE

      RETURN
      END
```

INLAGA

For any $n \geq 1$, the routine computes the value at the point x of a polynomial q of degree at most $n-1$ determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the Laguerre polynomial $L_n^{(\alpha)}$.

One has

$$q(x) = \sum_{j=1}^{n} q(\xi_j^{(n)}) \; l_j^{(n)}(x),$$

where for $1 \leq j \leq n$

$$l_j^{(n)}(x) = \begin{cases} \left[\dfrac{d}{dx}\hat{L}_n^{(\alpha)}(\xi_j^{(n)})\right]^{-1} \dfrac{\hat{L}_n^{(\alpha)}(x)}{x - \xi_j^{(n)}} \displaystyle\prod_{k=1}^{n} \dfrac{4k + x}{4k + \xi_j^{(n)}} & \text{if } x \neq \xi_j^{(n)}, \\ 1 & \text{if } x = \xi_j^{(n)}. \end{cases}$$

Here, $\hat{L}_n^{(\alpha)}$ is the scaled Laguerre function of degree n, introduced in the description of subroutine VALASF. The zeroes and the values $\frac{d}{dx}\hat{L}_n^{(\alpha)}(\xi_j^{(n)})$, $1 \leq j \leq n$, can be obtained by calling subroutine ZELAGA.

Input variables	*Output variable*
N , the degree n A , the parameter α CS , vector of the zeroes $\xi_j^{(n)}$ DZ , the values $\frac{d}{dx}\hat{L}_n^{(\alpha)}(\xi_j^{(n)})$ QZ , values of q at the zeroes X , the point x	QX , the value of q in x

Auxiliary routine: **VALASF**

```
      SUBROUTINE INLAGA(N,A,CS,DZ,QZ,X,QX)
**********************************************************************
*   COMPUTES THE VALUE AT A GIVEN POINT OF A POLYNOMIAL INDIVIDUATED
*   BY THE VALUES ATTAINED AT THE ZEROES OF THE LAGUERRE POLYNOMIAL
*   N  = THE NUMBER OF ZEROES
*   A  = PARAMETER > -1
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   DZ = SCALED LAGUERRE DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   QX = VALUE OF THE POLYNOMIAL IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1), QZ(1)
      IF (N .EQ. 0) RETURN

          EPS = 1.D-14
      CALL VALASF(N,A,X,Y,DY)
          QX = 0.D0
      DO 1 J=1,N
          ED = X-CS(J)
      IF(DABS(ED) .LT. EPS) THEN
          QX = QZ(J)
      RETURN
      ELSE
          PR = 1.D0
      DO 2 K=1,N
          DK = 4.D0*DFLOAT(K)
          PR = PR*(DK+X)/(DK+CS(J))
2     CONTINUE
          QX = QX+QZ(J)*Y*PR/(DZ(J)*ED)
      ENDIF
1     CONTINUE

      RETURN
      END
```

INHEGA

For any $n \geq 1$, the routine computes the value at the point x of a polynomial q of degree at most $n-1$, determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the Hermite polynomial H_n.

One has

$$q(x) \; = \; \sum_{j=1}^{n} q(\xi_j^{(n)}) \; l_j^{(n)}(x),$$

where for $1 \leq j \leq n$

$$l_j^{(n)}(x) \; = \begin{cases} \left[\hat{H}_n'(\xi_j^{(n)})\right]^{-1} \dfrac{\hat{H}_n(x)}{x-\xi_j^{(n)}} \displaystyle\prod_{k=1}^{m} \frac{4k+x^2}{4k+[\xi_j^{(n)}]^2} & \text{if } x \neq \xi_j^{(n)}, \\ 1 & \text{if } x = \xi_j^{(n)}. \end{cases}$$

Here, $\hat{H}_n$ is the scaled Hermite function of degree n introduced in the description of subroutine VAHESF, and $m = n/2$ if n is even, $m = (n-1)/2$ if n is odd.

The zeroes and the values $\hat{H}_n'(\xi_j^{(n)})$, $1 \leq j \leq n$, can be obtained by calling subroutine ZEHEGA.

Input variables	*Output variable*
N , the degree n CS , vector of the zeroes $\xi_j^{(n)}$ DZ , the values $\hat{H}_n'(\xi_j^{(n)})$ QZ , values of q at the zeroes X , the point x	QX , the value of q in x

Auxiliary routine: **VAHESF**

```
      SUBROUTINE INHEGA(N,CS,DZ,QZ,X,QX)
**********************************************************************
*   COMPUTES THE VALUE AT A GIVEN POINT OF A POLYNOMIAL INDIVIDUATED
*   BY THE VALUES ATTAINED AT THE ZEROES OF THE HERMITE POLYNOMIAL
*   N  = THE NUMBER OF ZEROES
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   DZ = SCALED HERMITE DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   QX = VALUE OF THE POLYNOMIAL IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1), QZ(1)
      IF (N .EQ. 0) RETURN

          QX = QZ(1)
      IF (N .EQ. 1) RETURN

          EPS = 1.D-14
      CALL VAHESF(N,X,Y,DY)
          QX = 0.D0
      DO 1 J=1,N
          ED = X-CS(J)
      IF(DABS(ED) .LT. EPS) THEN
          QX = QZ(J)
      RETURN
      ELSE
          PR = 1.D0
      DO 2 K=1,N/2
          DK = 4.D0*DFLOAT(K)
          PR = PR*(DK+X*X)/(DK+CS(J)**2)
2     CONTINUE
          QX = QX+QZ(J)*Y*PR/(DZ(J)*ED)
      ENDIF
1     CONTINUE

      RETURN
      END
```

INJAGL

For any $n \geq 1$, the routine computes the value at the point x of a polynomial q of degree at most n, determined by the values attained at the Jacobi Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \leq j \leq n$.

One has

$$q(x) \;=\; \sum_{j=0}^{n} q(\eta_j^{(n)})\ \tilde{l}_j^{(n)}(x),$$

where

$$\tilde{l}_j^{(n)}(x) \;=\; \begin{cases} \dfrac{(\beta+1)(x-1)}{n(n+\alpha+\beta+1)\ P_n^{(\alpha,\beta)}(\eta_0^{(n)})}\ \dfrac{d}{dx}P_n^{(\alpha,\beta)}(x) & j=0, \\[3ex] \dfrac{(x^2-1)}{n(n+\alpha+\beta+1)\ P_n^{(\alpha,\beta)}(\eta_j^{(n)})\ (x-\eta_j^{(n)})}\ \dfrac{d}{dx}P_n^{(\alpha,\beta)}(x) & 1 \leq j \leq n-1, \\[3ex] \dfrac{(\alpha+1)(x+1)}{n(n+\alpha+\beta+1)\ P_n^{(\alpha,\beta)}(\eta_n^{(n)})}\ \dfrac{d}{dx}P_n^{(\alpha,\beta)}(x) & j=n. \end{cases}$$

The nodes and the values $P_n^{(\alpha,\beta)}(\eta_j^{(n)})$, $0 \leq j \leq n$, can be obtained by calling subroutine ZEJAGL.

Input variables	*Output variable*
N , the degree n A , the parameter α B , the parameter β ET , vector of the nodes $\eta_j^{(n)}$ VN , the values $P_n^{(\alpha,\beta)}(\eta_j^{(n)})$ QN , values of q at the nodes X , the point x	QX , the value of q in x

<u>Auxiliary routine</u>: **VAJAPO**

```
      SUBROUTINE INJAGL(N,A,B,ET,VN,QN,X,QX)
**********************************************************************
*   COMPUTES THE VALUE AT A GIVEN POINT OF A POLYNOMIAL INDIVIDUATED
*   BY THE VALUES ATTAINED AT THE JACOBI GAUSS-LOBATTO NODES
*   N  = THE DEGREE OF THE POLYNOMIAL
*   A  = PARAMETER > -1
*   B  = PARAMETER > -1
*   ET = VECTOR OF THE NODES, ET(I), I=0,N
*   VN = VALUES OF THE JACOBI POLYNOMIAL AT THE NODES, VN(I), I=0,N
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   QX = VALUE OF THE POLYNOMIAL IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*), QN(0:*)
      IF (N .EQ. 0) RETURN

          EPS = 1.D-14
      CALL VAJAPO(N,A,B,X,Y,DY,D2Y)
          DN = DFLOAT(N)
          C  = 1.D0/(DN*(DN+A+B+1.D0))
          QX = QN(0)*C*DY*(B+1.D0)*(X-1.D0)/VN(0)
          QX = QX+QN(N)*C*DY*(A+1.D0)*(X+1.D0)/VN(N)
      IF (N .EQ. 1) RETURN

      DO 1 J=1,N-1
          ED = X-ET(J)
      IF(DABS(ED) .LT. EPS) THEN
          QX = QN(J)
      RETURN
      ELSE
          QX = QX+QN(J)*C*DY*(X*X-1.D0)/(VN(J)*ED)
      ENDIF
1     CONTINUE

      RETURN
      END
```

INLEGL

For any $n \geq 1$, the routine computes the value at the point x of a polynomial q of degree at most n, determined by the values attained at the Legendre Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \leq j \leq n$.

One has

$$q(x) = \sum_{j=0}^{n} q(\eta_j^{(n)}) \, \tilde{l}_j^{(n)}(x),$$

where

$$\tilde{l}_j^{(n)}(x) = \begin{cases} \dfrac{(x-1)\ P_n'(x)}{n(n+1)\ P_n(\eta_0^{(n)})} & j = 0, \\[2ex] \dfrac{(x^2-1)\ P_n'(x)}{n(n+1)\ P_n(\eta_j^{(n)})\ (x-\eta_j^{(n)})} & 1 \leq j \leq n-1, \\[2ex] \dfrac{(x+1)\ P_n'(x)}{n(n+1)\ P_n(\eta_n^{(n)})} & j = n. \end{cases}$$

The nodes and the values $P_n(\eta_j^{(n)})$, $0 \leq j \leq n$, can be obtained by calling subroutine ZELEGL.

Input variables	*Output variable*
N , the degree n ET , vector of the nodes $\eta_j^{(n)}$ VN , the values $P_n(\eta_j^{(n)})$ QN , values of q at the nodes X , the point x	QX , the value of q in x

Auxiliary routine: **VALEPO**

```
      SUBROUTINE INLEGL(N,ET,VN,QN,X,QX)
**********************************************************************
*   COMPUTES THE VALUE AT A GIVEN POINT OF A POLYNOMIAL INDIVIDUATED
*   BY THE VALUES ATTAINED AT THE LEGENDRE GAUSS-LOBATTO NODES
*   N  = THE DEGREE OF THE POLYNOMIAL
*   ET = VECTOR OF THE NODES, ET(I), I=0,N
*   VN = VALUES OF THE LEGENDRE POLYNOMIAL AT THE NODES, VN(I), I=0,N
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   QX = VALUE OF THE POLYNOMIAL IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*), QN(0:*)
      IF (N .EQ. 0) RETURN

          EPS = 1.D-14
      CALL VALEPO(N,X,Y,DY,D2Y)
          DN = DFLOAT(N)
          C  = 1.D0/(DN*(DN+1.D0))
          QX = QN(0)*C*DY*(X-1.D0)/VN(0)
          QX = QX+QN(N)*C*DY*(X+1.D0)/VN(N)
      IF (N .EQ. 1) RETURN

      DO 1 J=1,N-1
          ED = X-ET(J)
      IF(DABS(ED) .LT. EPS) THEN
          QX = QN(J)
      RETURN
      ELSE
          QX = QX+QN(J)*C*DY*(X*X-1.D0)/(VN(J)*ED)
      ENDIF
1     CONTINUE

      RETURN
      END
```

INCHGL

For any $n \geq 1$, the routine computes the value at the point x of a polynomial q of degree at most n, determined by the values attained at the Chebyshev Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \leq j \leq n$.

One has

$$q(x) = \sum_{j=0}^{n} q(\eta_j^{(n)}) \, \tilde{l}_j^{(n)}(x),$$

where

$$\tilde{l}_j^{(n)}(x) = \begin{cases} \dfrac{(-1)^n}{2n^2}(x-1)\, T_n'(x) & j = 0, \\[2ex] \dfrac{(-1)^{n+j}}{n^2} \, \dfrac{(x^2-1)\, T_n'(x)}{x - \eta_j^{(n)}} & 1 \leq j \leq n-1, \\[2ex] \dfrac{1}{2n^2}(x+1)\, T_n'(x) & j = n. \end{cases}$$

Input variables	*Output variable*
N , the degree n QN , values of q at the nodes X , the point x	QX , the value of q in x

Auxiliary routine: **VACHPO**

```
      SUBROUTINE INCHGL(N,QN,X,QX)
**********************************************************************
*   COMPUTES THE VALUE AT A GIVEN POINT OF A POLYNOMIAL INDIVIDUATED
*   BY THE VALUES ATTAINED AT THE CHEBYSHEV GAUSS-LOBATTO NODES
*   N  = THE DEGREE OF THE POLYNOMIAL
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   QX = VALUE OF THE POLYNOMIAL IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION  QN(0:*)
      IF (N .EQ. 0) RETURN

          EPS = 1.D-14
          PI = 3.14159265358979323846D0
      CALL VACHPO(N,X,Y,DY,D2Y)
          DN = DFLOAT(N)
          SN = DFLOAT(1+4*(N/2)-2*N)
          PN = PI/DN
          C  = 1.D0/(DN*DN)
          QX = .5D0*SN*QN(0)*C*DY*(X-1.D0)
          QX = QX+.5D0*QN(N)*C*DY*(X+1.D0)
      IF (N .EQ. 1) RETURN

          SJ = -1.D0
      DO 1 J=1,N-1
          ED = X+DCOS(PN*DFLOAT(J))
      IF(DABS(ED) .LT. EPS) THEN
          QX = QN(J)
      RETURN
      ELSE
          QX = QX+QN(J)*SN*SJ*C*DY*(X*X-1.D0)/ED
      ENDIF
          SJ = -SJ
1     CONTINUE

      RETURN
      END
```

INLAGR

For any $n \geq 1$, the routine computes the value at the point x of a polynomial q of degree at most $n-1$, determined by the values attained at the Laguerre Gauss-Radau nodes $\eta_j^{(n)}$, $0 \leq j \leq n-1$.

One has

$$q(x) \ = \ \sum_{j=0}^{n-1} q(\eta_j^{(n)}) \ \tilde{l}_j^{(n)}(x),$$

where the polynomial $\tilde{l}_j^{(n)}$ is given by

$$-\frac{\alpha+1}{n} \left[\hat{L}_n^{(\alpha)}(\eta_0^{(n)})\right]^{-1} \left[\frac{d}{dx}\hat{L}_n^{(\alpha)}(x) + \left(\sum_{m=1}^{n} \frac{1}{4m+x}\right) \hat{L}_n^{(\alpha)}(x)\right] \prod_{k=1}^{n} \frac{4k+x}{4k} \qquad \text{if } j=0,$$

$$-\frac{x}{n(x-\eta_j^{(n)})} \left[\hat{L}_n^{(\alpha)}(\eta_j^{(n)})\right]^{-1} \left[\frac{d}{dx}\hat{L}_n^{(\alpha)}(x) + \left(\sum_{m=1}^{n} \frac{1}{4m+x}\right) \hat{L}_n^{(\alpha)}(x)\right] \prod_{k=1}^{n} \frac{4k+x}{4k+\eta_j^{(n)}}$$
$$\text{if } \ 1 \leq j \leq n-1.$$

Here, $\hat{L}_n^{(\alpha)}$ is the scaled Laguerre function of degree n, introduced in the description of subroutine VALASF. The nodes and the values $\hat{L}_n^{(\alpha)}(\eta_j^{(n)})$, $0 \leq j \leq n$, can be obtained by calling subroutine ZELAGR.

Input variables	*Output variable*
N , the degree n A , the parameter α ET , vector of the nodes $\eta_j^{(n)}$ VN , the values $\hat{L}_n^{(\alpha)}(\eta_j^{(n)})$ QN , values of q at the nodes X , the point x	QX , the value of q in x

Auxiliary routine: **VALASF**

```
      SUBROUTINE INLAGR(N,A,ET,VN,QN,X,QX)
**********************************************************************
*    COMPUTES THE VALUE AT A GIVEN POINT OF A POLYNOMIAL INDIVIDUATED
*    BY THE VALUES ATTAINED AT THE LAGUERRE GAUSS-RADAU NODES
*    N  = THE NUMBER OF NODES
*    A  = PARAMETER > -1
*    ET = VECTOR OF THE NODES, ET(I), I=0,N-1
*    VN = SCALED LAGUERRE FUNCTION AT THE NODES, VN(I), I=0,N-1
*    QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N-1
*    X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*    QX = VALUE OF THE POLYNOMIAL IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*), QN(0:*)
      IF (N .EQ. 0) RETURN
          QX = QN(0)
      IF (N .EQ. 1) RETURN
          EPS = 1.D-14
          DN = DFLOAT(N)
          C  = -1.D0/DN
          PR = 1.D0
          SU = 0.D0
      DO 1 M=1,N
          DM = 4.D0*DFLOAT(M)
          PR = PR*(DM+X)/DM
          SU = SU+1.D0/(DM+X)
1     CONTINUE
      CALL VALASF(N,A,X,Y,DY)
          QX = QN(0)*C*(A+1.D0)*PR*(DY+SU*Y)/VN(0)
      DO 2 J=1,N-1
          ED = X-ET(J)
      IF(DABS(ED) .LT. EPS) THEN
          QX = QN(J)
      RETURN
      ELSE
          PR = 1.D0
      DO 3 K=1,N
          DK = 4.D0*DFLOAT(K)
          PR = PR*(DK+X)/(DK+ET(J))
3     CONTINUE
          QX = QX+QN(J)*C*PR*(DY+SU*Y)*X/(VN(J)*ED)
      ENDIF
2     CONTINUE
      RETURN
      END
```

NOLEGA

For any $n \geq 1$, the routine computes the quantities

$$\mathcal{I} \;=\; \left(\int_{-1}^{1} q^2(x)\; dx\right)^{\frac{1}{2}} \;=\; \left(\sum_{j=1}^{n} q^2(\xi_j^{(n)})\; w_j^{(n)}\right)^{\frac{1}{2}},$$

$$\mathcal{M} \;=\; \max_{1\leq j\leq n}\; |q(\xi_j^{(n)})|,$$

where q is a polynomial of degree at most $n-1$ individuated by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the Legendre polynomial P_n.
Here, $w_j^{(n)}$, $1 \leq j \leq n$, are the weights of the Legendre Gauss integration formula, which can be determined calling subroutine WELEGA.

Input variables	*Output variables*
N , the number of zeroes	QI , the norm $\mathcal{I}$
QZ , the values of q at the zeroes	QM , the norm $\mathcal{M}$
WE , Gauss-Legendre weights $w_j^{(n)}$	

```
      SUBROUTINE NOLEGA(N,QZ,WE,QI,QM)
***********************************************************************
*   COMPUTES THE NORMS OF A POLYNOMIAL DEFINED AT THE LEGENDRE ZEROES
*   N  = THE NUMBER OF ZEROES
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   WE = VECTOR OF THE LEGENDRE GAUSS WEIGHTS, WE(I), I=1,N
*   QI = INTEGRAL NORM OF THE POLYNOMIAL
*   QM = MAXIMUM VALUE OF THE POLYNOMIAL AT THE ZEROES
***********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION QZ(1), WE(1)
      IF (N .EQ. 0) RETURN

          EPS = 1.D-14
          SU  = 0.D0
          QM  = 0.D0
      DO 1 J=1,N
          Y  = DABS(QZ(J))
      IF(Y .GT. QM) QM=Y
      IF(Y .LT. EPS) GOTO 1
          SU = SU+Y*Y*WE(J)
1     CONTINUE
          QI = DSQRT(SU)

      RETURN
      END
```

NOCHGA

For any $n \geq 1$, the routine computes the quantities

$$\mathcal{W} = \left(\int_{-1}^{1} q^2(x) \frac{dx}{\sqrt{1-x^2}} \right)^{\frac{1}{2}} = \left(\frac{\pi}{n} \sum_{j=1}^{n} q^2(\xi_j^{(n)}) \right)^{\frac{1}{2}},$$

$$\mathcal{I} = \left(\int_{-1}^{1} q^2(x)\, dx \right)^{\frac{1}{2}} = \left(\sum_{j=0}^{2n} q^2 \left(\cos \frac{j\pi}{2n} \right) \chi_j^{(2n)} \right)^{\frac{1}{2}},$$

$$\mathcal{M} = \max_{1 \leq j \leq n} |q(\xi_j^{(n)})|,$$

where q is a polynomial of degree at most $n-1$ individuated by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the Chebyshev polynomial T_n.
Here, $\chi_j^{(2n)}$, $0 \leq j \leq 2n$, are the weights of the Clenshaw-Curtis integration formula, which can be determined by calling subroutine WECHCC. The zeroes and the values $T_n'(\xi_j^{(n)})$, $1 \leq j \leq n$, can be obtained by calling subroutine ZECHGA.

Input variables	*Output variables*
N , the number of zeroes	QW , the norm $\mathcal{W}$
DZ , the values $T_n'(\xi_j^{(n)})$	QI , the norm $\mathcal{I}$
QZ , the values of q at the zeroes	QM , the norm $\mathcal{M}$
WK , Clenshaw-Curtis weights $\chi_j^{(2n)}$	

Auxiliary routines: **INCHGA, VACHPO**

```
      SUBROUTINE NOCHGA(N,DZ,QZ,WK,QW,QI,QM)
*************************************************************************
*   COMPUTES THE NORMS OF A POLYNOMIAL DEFINED AT THE CHEBYSHEV ZEROES
*   N  = THE NUMBER OF ZEROES
*   DZ = CHEBYSHEV POLYNOMIAL DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   WK = VECTOR OF THE CLENSHAW-CURTIS WEIGHTS, WE(I), I=0,2*N
*   QW = WEIGHTED INTEGRAL NORM OF THE POLYNOMIAL
*   QI = INTEGRAL NORM OF THE POLYNOMIAL
*   QM = MAXIMUM VALUE OF THE POLYNOMIAL AT THE ZEROES
*************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION DZ(1), QZ(1), WK(0:*)
      IF (N .EQ. 0) RETURN

          PI = 3.14159265358979323846D0
          DN = DFLOAT(N)
          X  = -1.D0
      CALL INCHGA(N,DZ,QZ,X,QX)
          S1 = 0.D0
          S2 = QX*QX*WK(0)
          QM = 0.D0
      DO 1 J=1,N
          DJ = DFLOAT(J)
          J2 = 2*J
          X  = -DCOS(PI*DJ/DN)
          Y  = DABS(QZ(J))
      IF(Y .GT. QM) QM=Y
      CALL INCHGA(N,DZ,QZ,X,QX)
          S1 = S1+Y*Y
          S2 = S2+Y*Y*WK(J2-1)+QX*QX*WK(J2)
1     CONTINUE
          QW = DSQRT(S1*PI/DN)
          QI = DSQRT(S2)

      RETURN
      END
```

NOJAGL

For any $n \geq 1$, the routine computes the quantities

$$\mathcal{W} = \left(\int_{-1}^{1} q^2(x)\ (1-x)^\alpha (1+x)^\beta\ dx \right)^{\frac{1}{2}},$$

$$\mathcal{S} = \left(\sum_{j=0}^{n} q^2(\eta_j^{(n)})\ \tilde{w}_j^{(n)} \right)^{\frac{1}{2}},$$

$$\mathcal{M} = \max_{0 \leq j \leq n} |q(\eta_j^{(n)})|,$$

where q is a polynomial of degree at most n individuated by the values attained at the Jacobi Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \leq j \leq n$.
Here, $\tilde{w}_j^{(n)}$, $0 \leq j \leq n$, are the weights of the Jacobi Gauss-Lobatto formula, which can be determined by calling subroutine WEJAGL.
The nodes and the values $P_n^{(\alpha,\beta)}(\eta_j^{(n)})$, $0 \leq j \leq n$, can be obtained by calling subroutine ZEJAGL.

Input variables	*Output variables*
N , the degree n	QW , the norm $\mathcal{W}$
A , the parameter α	QS , the norm $\mathcal{S}$
B , the parameter β	QM , the norm $\mathcal{M}$
VN , the values $P_n^{(\alpha,\beta)}(\eta_j^{(n)})$	
QN , the values of q at the nodes	
WT , the weights $\tilde{w}_j^{(n)}$	

Auxiliary routine: **GAMMAF**

```
      SUBROUTINE NOJAGL(N,A,B,VN,QN,WT,QW,QS,QM)
*********************************************************************
*   COMPUTES THE NORMS OF A POLYNOMIAL DEFINED AT THE JACOBI GAUSS-
*   LOBATTO NODES
*   N  = THE DEGREE OF THE POLYNOMIAL
*   A  = PARAMETER > -1
*   B  = PARAMETER > -1
*   VN = VALUES OF THE JACOBI POLYNOMIAL AT THE NODES, VN(I), I=0,N
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   WT = VECTOR OF THE JACOBI GAUSS-LOBATTO WEIGHTS, WT(I), I=0,N
*   QW = WEIGHTED INTEGRAL NORM OF THE POLYNOMIAL
*   QS = QUADRATURE NORM OF THE POLYNOMIAL
*   QM = MAXIMUM VALUE OF THE POLYNOMIAL AT THE NODES
*********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION VN(0:*), QN(0:*), WT(0:*)
      IF (N .EQ. 0) RETURN

          EPS = 1.D-14
          A1  = A+1.D0
          B1  = B+1.D0
          AB  = A+B
          AB2 = AB+2.D0
          DN = DFLOAT(N)
          C  = ((2.D0)**(AB+1.D0))*(DN+AB+1.D0)/(2.D0*DN+AB+1.D0)
      CALL GAMMAF(A1,GA1)
      CALL GAMMAF(B1,GB1)
      CALL GAMMAF(AB2,GAB2)
          C  = C*GA1*GB1/GAB2
      DO 1 K=1,N
          DK = DFLOAT(K)
          C  = C*(DK+A)*(DK+B)/(DK*(DK+AB+1.D0))
1     CONTINUE

          S1 = 0.D0
          S2 = 0.D0
          S3 = 0.D0
          QM = 0.D0
      DO 2 J=0,N
          Y1 = QN(J)
          YM = DABS(Y1)
      IF(YM .GT. QM) QM=YM
          Y2 = VN(J)
          S2 = S2+Y1*Y2*WT(J)
          S3 = S3+Y2*Y2*WT(J)
      IF(YM .LT. EPS) GOTO 2
          S1 = S1+Y1*Y1*WT(J)
2     CONTINUE
          QS = DSQRT(S1)
          QW = DSQRT(DABS(S1-(S3-C)*S2*S2/(S3*S3)))

      RETURN
      END
```

NOLEGL

For any $n \geq 1$, the routine computes the quantities

$$\mathcal{I} \;=\; \left(\int_{-1}^{1} q^2(x)\;dx\right)^{\frac{1}{2}} \;=\; \left[\mathcal{S}^2 - \frac{n(n+1)}{2(2n+1)}\left(\sum_{j=0}^{n} (qP_n)(\eta_j^{(n)})\;\tilde{w}_j^{(n)}\right)^2\right]^{\frac{1}{2}},$$

$$\mathcal{S} \;=\; \left(\sum_{j=0}^{n} q^2(\eta_j^{(n)})\;\tilde{w}_j^{(n)}\right)^{\frac{1}{2}},$$

$$\mathcal{M} \;=\; \max_{0\leq j\leq n} |q(\eta_j^{(n)})|,$$

where q is a polynomial of degree at most n individuated by the values attained at the Legendre Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \leq j \leq n$.
Here, $\tilde{w}_j^{(n)}$, $0 \leq j \leq n$, are the weights of the Legendre Gauss-Lobatto formula, which can be determined by calling subroutine WELEGL. The nodes and the values $P_n(\eta_j^{(n)})$, $0 \leq j \leq n$, can be obtained by calling subroutine ZELEGL.

Input variables	*Output variables*
N , the degree n VN , the values $P_n(\eta_j^{(n)})$ QN , the values of q at the nodes WT , the weights $\tilde{w}_j^{(n)}$	QI , the norm $\mathcal{I}$ QS , the norm $\mathcal{S}$ QM , the norm $\mathcal{M}$

```
      SUBROUTINE NOLEGL(N,VN,QN,WT,QI,QS,QM)
**********************************************************************
*   COMPUTES THE NORMS OF A POLYNOMIAL DEFINED AT THE LEGENDRE GAUSS-
*   LOBATTO NODES
*   N  = THE DEGREE OF THE POLYNOMIAL
*   VN = VALUES OF THE LEGENDRE POLYNOMIAL AT THE NODES, VN(I), I=0,N
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   WT = VECTOR OF THE LEGENDRE GAUSS-LOBATTO WEIGHTS, WT(I), I=0,N
*   QW = INTEGRAL NORM OF THE POLYNOMIAL
*   QS = QUADRATURE NORM OF THE POLYNOMIAL
*   QM = MAXIMUM VALUE OF THE POLYNOMIAL AT THE NODES
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION VN(0:*), QN(0:*), WT(0:*)
      IF (N .EQ. 0) RETURN

          EPS = 1.D-14
          DN = DFLOAT(N)
          C  = .5D0*DN*(DN+1.D0)/(2.D0*DN+1.D0)

          S1 = 0.D0
          S2 = 0.D0
          QM = 0.D0
      DO 1 J=0,N
          Y1 = QN(J)
          YM = DABS(Y1)
      IF(YM .GT. QM) QM=YM
          Y2 = VN(J)
          S2 = S2+Y1*Y2*WT(J)
      IF(YM .LT. EPS) GOTO 1
          S1 = S1+Y1*Y1*WT(J)
1     CONTINUE
          QS = DSQRT(S1)
          QI = DSQRT(DABS(S1-C*S2*S2))

      RETURN
      END
```

NOCHGL

For any $n \geq 1$, the routine computes the quantities

$$\mathcal{W} = \left(\int_{-1}^{1} q^2(x) \frac{dx}{\sqrt{1-x^2}} \right)^{\frac{1}{2}} = \left[\mathcal{S}^2 - \frac{1}{2\pi} \left(\sum_{j=0}^{n} (-1)^{n+j} \, q(\eta_j^{(n)}) \, \tilde{w}_j^{(n)} \right)^2 \right]^{\frac{1}{2}},$$

$$\mathcal{I} = \left(\int_{-1}^{1} q^2(x) \, dx \right)^{\frac{1}{2}} = \left[\sum_{j=0}^{2n} q^2 \left(\cos \frac{j\pi}{2n} \right) \chi_j^{(2n)} \right]^{\frac{1}{2}},$$

$$\mathcal{S} = \left(\sum_{j=0}^{n} q^2(\eta_j^{(n)}) \, \tilde{w}_j^{(n)} \right)^{\frac{1}{2}}, \quad \mathcal{M} = \max_{0 \leq j \leq n} |q(\eta_j^{(n)})|,$$

where q is a polynomial of degree at most n individuated by the values attained at the Chebyshev Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \leq j \leq n$.
Here, $\tilde{w}_j^{(n)}$, $0 \leq j \leq n$, are the weights of the Chebyshev Gauss-Lobatto formula and $\chi_j^{(2n)}$, $0 \leq j \leq 2n$, are the Clenshaw-Curtis weights. The first can be determined by calling subroutine WECHGL, the last by calling subroutine WECHCC.

Input variables	*Output variables*
N , the degree n	QW , the norm $\mathcal{W}$
QN , the values of q at the nodes	QI , the norm $\mathcal{I}$
WK , Clenshaw-Curtis weights $\chi_j^{(2n)}$	QS , the norm $\mathcal{S}$
	QM , the norm $\mathcal{M}$

Auxiliary routines: **INCHGL, VACHPO**

```
      SUBROUTINE NOCHGL(N,QN,WK,QW,QI,QS,QM)
*********************************************************************
*   COMPUTES THE NORMS OF A POLYNOMIAL DEFINED AT THE CHEBYSHEV GAUSS-
*   LOBATTO NODES
```

```
*   N  = THE DEGREE OF THE POLYNOMIAL
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   WK = VECTOR OF THE CLENSHAW-CURTIS WEIGHTS, WE(I), I=0,2*N
*   QW = WEIGHTED INTEGRAL NORM OF THE POLYNOMIAL
*   QI = INTEGRAL NORM OF THE POLYNOMIAL
*   QS = QUADRATURE NORM OF THE POLYNOMIAL
*   QM = MAXIMUM VALUE OF THE POLYNOMIAL AT THE NODES
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION QN(0:*), WK(0:*)
      IF (N .EQ. 0) RETURN

          PI = 3.14159265358979323846D0
          PH = 1.57079632679489661923D0
          DN = DFLOAT(N)
          SN = DFLOAT(1+4*(N/2)-2*N)
          Y  = QN(0)
          S1 = .5D0*Y*Y
          S2 = .5D0*Y*SN
          S3 = Y*Y*WK(0)
          QM = DABS(Y)

          SJ = -1.D0
      DO 1 J=1,N-1
          J2 = 2*J
          DJ = DFLOAT(J2-1)
          X  = -DCOS(PH*DJ/DN)
          Y  = QN(J)
          YM = DABS(Y)
      IF(YM .GT. QM) QM=YM
      CALL INCHGL(N,QN,X,QX)
          S1 = S1+Y*Y
          S2 = S2+Y*SN*SJ
          S3 = S3+QX*QX*WK(J2-1)+Y*Y*WK(J2)
          SJ = -SJ
1     CONTINUE
          N2 = 2*N
          DD = DFLOAT(N2-1)
          X  = -DCOS(PH*DD/DN)
          Y  = QN(N)
          YM = DABS(Y)
      IF(YM .GT. QM) QM=YM
      CALL INCHGL(N,QN,X,QX)
          S1 = S1+.5D0*Y*Y
          S2 = S2+.5D0*Y
          S3 = S3+QX*QX*WK(N2-1)+Y*Y*WK(N2)

          QW = DSQRT(DABS(PI*S1/DN-PH*S2*S2/(DN*DN)))
          QI = DSQRT(S3)
          QS = DSQRT(PI*S1/DN)

      RETURN
      END
```

COJAGA

For any $n \geq 1$, the routine computes the n Fourier coefficients c_k, $0 \leq k \leq n-1$, with respect to the Jacobi polynomial basis, of a polynomial q of degree at most $n-1$ determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of $P_n^{(\alpha,\beta)}$.
One has

$$\begin{aligned} c_k &= \gamma_k \int_{-1}^{1} q(x)\ P_k^{(\alpha,\beta)}(x)\ (1-x)^\alpha (1+x)^\beta dx \\ &= \gamma_k \sum_{j=1}^{n} q(\xi_j^{(n)})\ P_k^{(\alpha,\beta)}(\xi_j^{(n)})\ w_j^{(n)} \qquad 0 \leq k \leq n-1, \end{aligned}$$

where $\gamma_k = \left(\int_{-1}^{1} [P_k^{(\alpha,\beta)}(x)]^2 (1-x)^\alpha (1+x)^\beta dx\right)^{-1}$, $0 \leq k \leq n-1$.
Here, $w_j^{(n)}$, $1 \leq j \leq n$, are the weights of the Jacobi Gauss integration formula, which can be determined by calling subroutine WEJAGA. The zeroes can be obtained by calling subroutine ZEJAGA.

Input variables	*Output variable*
N , the number of zeroes A , the parameter α B , the parameter β CS , vector of the zeroes $\xi_j^{(n)}$ QZ , the values of q at the zeroes WE , the weights $w_j^{(n)}$	CO , the Fourier coefficients of q

Auxiliary routine: **GAMMAF**

```
      SUBROUTINE COJAGA(N,A,B,CS,QZ,WE,CO)
*****************************************************************
*   COMPUTES THE JACOBI FOURIER COEFFICIENTS OF A POLYNOMIAL
*   INDIVIDUATED BY THE VALUES ATTAINED AT THE JACOBI ZEROES
*   N  = THE NUMBER OF ZEROES
*   A  = PARAMETER >-1
*   B  = PARAMETER >-1
```

```
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   WE = VECTOR OF THE WEIGHTS, WE(I), I=1,N
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N-1
*****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), QZ(1), WE(1), CO(0:*)
      IF (N .EQ. 0) RETURN

          A1  = A+1.D0
          B1  = B+1.D0
          AB  = A+B
          AB2 = AB+2.D0
      CALL GAMMAF(A1,GA1)
      CALL GAMMAF(B1,GB1)
      CALL GAMMAF(AB2,GAB2)
          C   = ((2.D0)**(AB+1.D0))*GA1*GB1/GAB2

          SU = 0.D0
      DO 1 J=1,N
          SU = SU+QZ(J)*WE(J)
          CO(J-1) = 0.D0
1     CONTINUE
          CO(0) = SU/C
      IF (N .EQ. 1) RETURN

      DO 2 J=1,N
          X  = CS(J)
          YP = QZ(J)*WE(J)
          Y  = .5D0*YP*(AB2*X+A-B)
      DO 3 K=1,N-1
          CO(K) = CO(K)+Y
          DK = DFLOAT(K+1)
          CC = 2.D0*DK+AB
          C1 = 2.D0*DK*(DK+AB)*(CC-2.D0)
          C2 = (CC-1.D0)*(CC-2.D0)*CC
          C3 = (CC-1.D0)*(A-B)*AB
          C4 = 2.D0*(DK+A-1.D0)*CC*(DK+B-1.D0)
          YM = Y
          Y  = ((C2*X+C3)*Y-C4*YP)/C1
          YP = YM
3     CONTINUE
2     CONTINUE

      DO 4 K=1,N-1
          DK = DFLOAT(K)
          C  = C*(DK+A)*(DK+B)/DK
          CO(K) = CO(K)*(2.D0*DK+AB+1.D0)/C
          C  = C/(DK+AB+1.D0)
4     CONTINUE

      RETURN
      END
```

COLEGA

For any $n \geq 1$, the routine computes the n Fourier coefficients c_k, $0 \leq k \leq n-1$, with respect to the Legendre polynomial basis, of a polynomial q of degree at most $n-1$ determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of P_n.
One has

$$c_k = \frac{2k+1}{2}\int_{-1}^{1} q(x)\, P_k(x)\, dx = \frac{2k+1}{2}\sum_{j=1}^{n} q(\xi_j^{(n)})\, P_k(\xi_j^{(n)})\, w_j^{(n)} \qquad 0 \leq k \leq n-1.$$

Here, $w_j^{(n)}$, $1 \leq j \leq n$, are the weights of the Legendre Gauss integration formula, which can be determined by calling subroutine WELEGA.
The zeroes can be obtained by calling subroutine ZELEGA.

Input variables	*Output variable*
N , the number of zeroes CS , vector of the zeroes $\xi_j^{(n)}$ QZ , the values of q at the zeroes WE , the weights $w_j^{(n)}$	CO , the Fourier coefficients of q

```
      SUBROUTINE COLEGA(N,CS,QZ,WE,CO)
*****************************************************************
*   COMPUTES THE LEGENDRE FOURIER COEFFICIENTS OF A POLYNOMIAL
*   INDIVIDUATED BY THE VALUES ATTAINED AT THE LEGENDRE ZEROES
*   N  = THE NUMBER OF ZEROES
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   WE = VECTOR OF THE WEIGHTS, WE(I), I=1,N
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N-1
*****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), QZ(1), WE(1), CO(0:*)
      IF (N .EQ. 0) RETURN

          SU = 0.D0
      DO 1 J=1,N
          SU = SU+QZ(J)*WE(J)
          CO(J-1) = 0.D0
1     CONTINUE
          CO(0) = .5D0*SU
      IF (N .EQ. 1) RETURN

      DO 2 J=1,N
          X  = CS(J)
          YP = QZ(J)*WE(J)
          Y  = X*YP
      DO 3 K=1,N-1
          CO(K) = CO(K)+Y
          DK = DFLOAT(K+1)
          C1 = 2.D0*DK-1.D0
          C2 = DK-1.D0
          YM = Y
          Y  = (C1*X*Y-C2*YP)/DK
          YP = YM
3     CONTINUE
2     CONTINUE

      DO 4 K=1,N-1
          CO(K) = .5D0*CO(K)*(2.D0*DFLOAT(K)+1.D0)
4     CONTINUE

      RETURN
      END
```

COCHGA

For any $n \geq 1$, the routine computes the n Fourier coefficients c_k, $0 \leq k \leq n-1$, with respect to the Chebyshev polynomial basis, of a polynomial q of degree at most $n-1$ determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of T_n.
One has for $0 \leq k \leq n-1$

$$c_k \;=\; \gamma_k \int_{-1}^{1} q(x)\; T_k(x)\; \frac{dx}{\sqrt{1-x^2}} \;=\; \frac{(-1)^k \gamma_k \pi}{n} \sum_{j=1}^{n} q(\xi_j^{(n)}) \;\cos \frac{k(2j-1)\pi}{2n},$$

where $\gamma_0 = 1/\pi$ and $\gamma_k = 2/\pi$, $1 \leq k \leq n-1$.

Input variables	*Output variable*
N , the number of zeroes QZ , the values of q at the zeroes	CO , the Fourier coefficients of q

```
      SUBROUTINE COCHGA(N,QZ,CO)
*****************************************************************
*   COMPUTES THE CHEBYSHEV FOURIER COEFFICIENTS OF A POLYNOMIAL
*   INDIVIDUATED BY THE VALUES ATTAINED AT THE CHEBYSHEV ZEROES
*   N  = THE NUMBER OF ZEROES
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N-1
*****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION  QZ(1), CO(0:*)
      IF (N .EQ. 0) RETURN

          PH = 1.57079632679489661923D0
          DN = DFLOAT(N)
          SU = 0.D0
      DO 1 J=1,N
          SU = SU+QZ(J)
1     CONTINUE
          CO(0) = SU/DN
      IF (N .EQ. 1) RETURN

          SK = -2.D0
      DO 2 K=1,N-1
          DK = DFLOAT(K)
          SU = 0.D0
      DO 3 J=1,N
          DJ = 2.D0*DFLOAT(J)-1.D0
          SU = SU+QZ(J)*DCOS(DK*DJ*PH/DN)
3     CONTINUE
          CO(K) = SK*SU/DN
          SK = -SK
2     CONTINUE

      RETURN
      END
```

COLAGA

For any $n \geq 1$, the routine computes the n Fourier coefficients c_k, $0 \leq k \leq n-1$, with respect to the Laguerre polynomial basis, of a polynomial q of degree at most $n-1$ determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of $L_n^{(\alpha)}$.
One has

$$c_k = \frac{k!}{\Gamma(k+\alpha+1)} \int_0^{+\infty} q(x)\ L_k^{(\alpha)}(x)\ x^\alpha e^{-x} dx$$
$$= \frac{k!}{\Gamma(k+\alpha+1)} \sum_{j=1}^{n} q(\xi_j^{(n)})\ L_k^{(\alpha)}(\xi_j^{(n)})\ w_j^{(n)} \qquad 0 \leq k \leq n-1.$$

Here, $w_j^{(n)}$, $1 \leq j \leq n$, are the weights of the Laguerre Gauss integration formula, which can be determined by calling subroutine WELAGA.
The zeroes can be obtained by calling subroutine ZELAGA.

Input variables	*Output variable*
N , the number of zeroes A , the parameter α CS , vector of the zeroes $\xi_j^{(n)}$ QZ , the values of q at the zeroes WE , the weights $w_j^{(n)}$	CO , the Fourier coefficients of q

Auxiliary routine: **GAMMAF**

```
      SUBROUTINE COLAGA(N,A,CS,QZ,WE,CO)
*****************************************************************
*   COMPUTES THE LAGUERRE FOURIER COEFFICIENTS OF A POLYNOMIAL
*   INDIVIDUATED BY THE VALUES ATTAINED AT THE LAGUERRE ZEROES
*   N  = THE NUMBER OF ZEROES
*   A  = PARAMETER >-1
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   WE = VECTOR OF THE WEIGHTS, WE(I), I=1,N
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N-1
*****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), QZ(1), WE(1), CO(0:*)
      IF (N .EQ. 0) RETURN

          A1  = A+1.D0
      CALL GAMMAF(A1,C)
          SU = 0.D0
      DO 1 J=1,N
          SU = SU+QZ(J)*WE(J)
          CO(J-1) = 0.D0
1     CONTINUE
          CO(0) = SU/C
      IF (N .EQ. 1) RETURN

      DO 2 J=1,N
          X  = CS(J)
          YP = QZ(J)*WE(J)
          Y  = (A1-X)*YP
      DO 3 K=1,N-1
          CO(K) = CO(K)+Y
          DK = DFLOAT(K+1)
          B1 = (2.D0*DK+A-1.D0-X)/DK
          B2 = (DK+A-1.D0)/DK
          YM = Y
          Y  = B1*Y-B2*YP
          YP = YM
3     CONTINUE
2     CONTINUE
      DO 4 K=1,N-1
          DK = DFLOAT(K)
          C  = C*(DK+A)/DK
          CO(K) = CO(K)/C
4     CONTINUE

      RETURN
      END
```

COHEGA

For any $n \geq 1$, the routine computes the n Fourier coefficients c_k, $0 \leq k \leq n-1$, with respect to the Hermite polynomial basis, of a polynomial q of degree at most $n-1$ determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of H_n.

One has for $0 \leq k \leq n-1$

$$c_k = \frac{1}{2^k\ k!\ \sqrt{\pi}} \int_{-\infty}^{+\infty} q(x) H_k(x)\ e^{-x^2} dx$$

$$= \frac{1}{2^k\ k!\ \sqrt{\pi}} \sum_{j=1}^{n}\ q(\xi_j^{(n)}) H_k(\xi_j^{(n)})\ w_j^{(n)} = \frac{1}{\sqrt{\pi}} \sum_{j=1}^{n}\ q(\xi_j^{(n)}) \tilde{H}_k(\xi_j^{(n)})\ w_j^{(n)},$$

where $\tilde{H}_k = (2^k k!)^{-1} H_k,\ k \in \mathbf{N}$, is given by the recursion formula

$$\begin{cases} \tilde{H}_0(x) = 1 \\ \tilde{H}_1(x) = x \\ \tilde{H}_k(x) = \frac{x}{k}\tilde{H}_{k-1}(x) - \frac{1}{2k}\tilde{H}_{k-2}(x), \quad k \geq 2. \end{cases}$$

Here, $w_j^{(n)}$, $1 \leq j \leq n$, are the weights of the Hermite Gauss integration formula, which can be determined by calling subroutine WEHEGA.

The zeroes can be obtained by calling subroutine ZEHEGA.

Input variables	*Output variable*
N , the number of zeroes CS , vector of the zeroes $\xi_j^{(n)}$ QZ , the values of q at the zeroes WE , the weights $w_j^{(n)}$	CO , the Fourier coefficients of q

```
      SUBROUTINE COHEGA(N,CS,QZ,WE,CO)
*****************************************************************
*   COMPUTES THE HERMITE FOURIER COEFFICIENTS OF A POLYNOMIAL
*   INDIVIDUATED BY THE VALUES ATTAINED AT THE HERMITE ZEROES
*   N  = THE NUMBER OF ZEROES
*   CS = VECTOR OF THE ZEROES, CS(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   WE = VECTOR OF THE WEIGHTS, WE(I), I=1,N
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N-1
*****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), QZ(1), WE(1), CO(0:*)
      IF (N .EQ. 0) RETURN

          PR = 1.7724538509055158800
          SU = 0.D0
      DO 1 J=1,N
          SU = SU+QZ(J)*WE(J)
          CO(J-1) = 0.D0
1     CONTINUE
          CO(0) = SU/PR
      IF (N .EQ. 1) RETURN

      DO 2 J=1,N
          X  = CS(J)
          YP = QZ(J)*WE(J)/PR
          Y  = X*YP
      DO 3 K=1,N-1
          CO(K) = CO(K)+Y
          DK = DFLOAT(K+1)
          YM = Y
          Y  = (X*Y-.5D0*YP)/DK
          YP = YM
3     CONTINUE
2     CONTINUE

      RETURN
      END
```

COJAGL

For any $n \in \mathbf{N}$, the routine computes the $n+1$ Fourier coefficients c_k, $0 \le k \le n$, with respect to the Jacobi polynomial basis, of a polynomial q of degree at most n determined by the values attained at the Jacobi Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \le j \le n$. One has

$$\begin{aligned} c_k &= \gamma_k \int_{-1}^{1} q(x)\ P_k^{(\alpha,\beta)}(x)\ (1-x)^\alpha (1+x)^\beta dx \\ &= \delta_k \sum_{j=0}^{n} q(\eta_j^{(n)})\ P_k^{(\alpha,\beta)}(\eta_j^{(n)})\ \tilde{w}_j^{(n)} \qquad 0 \le k \le n, \end{aligned}$$

where $\gamma_k = \left(\int_{-1}^{1} [P_k^{(\alpha,\beta)}(x)]^2 (1-x)^\alpha (1+x)^\beta dx\right)^{-1}$, $0 \le k \le n$, and $\delta_k = \gamma_k$ if $0 \le k \le n-1$, while $\delta_n = \left(\sum_{j=0}^{n} [P_n^{(\alpha,\beta)}(\eta_j^{(n)})]^2 \tilde{w}_j^{(n)}\right)^{-1}$.

Here, $\tilde{w}_j^{(n)}$, $0 \le j \le n$, are the weights of the Jacobi Gauss-Lobatto formula, which can be determined by calling subroutine WEJAGL. The nodes and the values $P_n^{(\alpha,\beta)}(\eta_j^{(n)})$, $0 \le j \le n$, can be obtained by calling subroutine ZEJAGL.

Input variables	*Output variable*
N , the degree n A , the parameter α B , the parameter β ET , vector of the nodes $\eta_j^{(n)}$ VN , the values $P_n^{(\alpha,\beta)}(\eta_j^{(n)})$ QN , the values of q at the nodes WT , the weights $\tilde{w}_j^{(n)}$	CO , the Fourier coefficients of q

Auxiliary routine: **GAMMAF**

```
      SUBROUTINE COJAGL(N,A,B,ET,VN,QN,WT,CO)
*********************************************************************
*   COMPUTES THE JACOBI FOURIER COEFFICIENTS OF A POLYNOMIAL
*   INDIVIDUATED BY ITS VALUES AT THE JACOBI GAUSS-LOBATTO NODES
*   N  = THE DEGREE OF THE POLYNOMIAL
*   A  = PARAMETER >-1
*   B  = PARAMETER >-1
*   ET = VECTOR OF THE NODES, ET(I), I=0,N
*   VN = VALUES OF THE JACOBI POLYNOMIAL AT THE NODES, VN(I), I=0,N
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   WT = VECTOR OF THE WEIGHTS, WT(I), I=0,N
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*), QN(0:*), WT(0:*), CO(0:*)
          CO(0) = QN(0)
      IF (N .EQ. 0) RETURN

          A1  = A+1.D0
          B1  = B+1.D0
          AB  = A+B
          AB2 = AB+2.D0
          CO(1) = (QN(1)-QN(0))/AB2
          CO(0) = .5D0*(QN(0)+QN(1)-(A-B)*CO(1))
      IF (N .EQ. 1) RETURN

      CALL GAMMAF(A1,GA1)
      CALL GAMMAF(B1,GB1)
      CALL GAMMAF(AB2,GAB2)
          C  = ((2.D0)**(AB+1.D0))*GA1*GB1/GAB2
          SU = 0.D0
      DO 1 J=0,N
          SU = SU+QN(J)*WT(J)
          CO(J) = 0.D0
1     CONTINUE
          CO(0) = SU/C
          CN = 0.D0
      DO 2 J=0,N
          X  = ET(J)
          YP = QN(J)*WT(J)
          Y  = .5D0*YP*(AB2*X+A-B)
          CN = CN+VN(J)*VN(J)*WT(J)
      DO 3 K=1,N
          CO(K) = CO(K)+Y
          DK = DFLOAT(K+1)
          CC = 2.D0*DK+AB
          C1 = 2.D0*DK*(DK+AB)*(CC-2.D0)
```

```
        C2 = (CC-1.D0)*(CC-2.D0)*CC
        C3 = (CC-1.D0)*(A-B)*AB
        C4 = 2.D0*(DK+A-1.D0)*CC*(DK+B-1.D0)
        YM = Y
        Y  = ((C2*X+C3)*Y-C4*YP)/C1
        YP = YM
3     CONTINUE
2     CONTINUE

      DO 4 K=1,N-1
        DK = DFLOAT(K)
        C  = C*(DK+A)*(DK+B)/DK
        CO(K) = CO(K)*(2.D0*DK+AB+1.D0)/C
        C  = C/(DK+AB+1.D0)
4     CONTINUE
        CO(N) = CO(N)/CN

      RETURN
      END
```

COLEGL

For any $n \in \mathbf{N}$, the routine computes the $n+1$ Fourier coefficients c_k, $0 \le k \le n$, with respect to the Legendre polynomial basis, of a polynomial q of degree at most n determined by the values attained at the Legendre Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \le j \le n$.

One has

$$c_k \;=\; \gamma_k \int_{-1}^{1} q(x)\; P_k(x)\; dx \;=\; \gamma_k \sum_{j=0}^{n} q(\eta_j^{(n)})\; P_k(\eta_j^{(n)})\; \tilde{w}_j^{(n)} \qquad 0 \le k \le n,$$

where $\gamma_k = (2k+1)/2$ if $0 \le k \le n-1$, while $\gamma_n = n/2$.

Here, $\tilde{w}_j^{(n)}$, $0 \le j \le n$, are the weights of the Legendre Gauss-Lobatto formula, which can be determined by calling subroutine WELEGL.

The nodes can be obtained by calling subroutine ZELEGL.

Input variables	*Output variable*
N , the degree n ET , vector of the nodes $\eta_j^{(n)}$ QN , the values of q at the nodes WT , the weights $\tilde{w}_j^{(n)}$	CO , the Fourier coefficients of q

```
      SUBROUTINE COLEGL(N,ET,QN,WT,CO)
*******************************************************************
*   COMPUTES THE LEGENDRE FOURIER COEFFICIENTS OF A POLYNOMIAL
*   INDIVIDUATED BY ITS VALUES AT THE LEGENDRE GAUSS-LOBATTO NODES
*   N  = THE DEGREE OF THE POLYNOMIAL
*   ET = VECTOR OF THE NODES, ET(I), I=0,N
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   WT = VECTOR OF THE WEIGHTS, WT(I), I=0,N
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*******************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), QN(0:*), WT(0:*), CO(0:*)
          CO(0) = QN(0)
      IF (N .EQ. 0) RETURN

          CO(0) = .5D0*(QN(0)+QN(1))
          CO(1) = .5D0*(QN(1)-QN(0))
      IF (N .EQ. 1) RETURN

          SU = 0.D0
      DO 1 J=0,N
          SU = SU+QN(J)*WT(J)
          CO(J) = 0.D0
1     CONTINUE
          CO(0) = .5D0*SU

      DO 2 J=0,N
          X  = ET(J)
          YP = QN(J)*WT(J)
          Y  = X*YP
      DO 3 K=1,N
          CO(K) = CO(K)+Y
          DK = DFLOAT(K+1)
          C1 = 2.D0*DK-1.D0
          C2 = DK-1.D0
          YM = Y
          Y  = (C1*X*Y-C2*YP)/DK
          YP = YM
3     CONTINUE
2     CONTINUE

          DN = DFLOAT(N)
          CO(N) = .5D0*DN*CO(N)
      IF (N .EQ. 1) RETURN

      DO 4 K=1,N-1
          CO(K) = .5D0*CO(K)*(2.D0*DFLOAT(K)+1.D0)
4     CONTINUE

      RETURN
      END
```

COCHGL

For any $n \in \mathbf{N}$, the routine computes the $n+1$ Fourier coefficients $c_k,\ 0 \leq k \leq n$, with respect to the Chebyshev polynomial basis, of a polynomial q of degree at most n determined by the values attained at the Chebyshev Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \leq j \leq n$.

One has

$$c_k = \gamma_k \int_{-1}^{1} q(x)\ T_k(x) \frac{dx}{\sqrt{1-x^2}}$$

$$= \frac{(-1)^k \gamma_k \pi}{n} \left(\tfrac{1}{2} q(\eta_0^{(n)}) + \tfrac{1}{2}(-1)^k q(\eta_n^{(n)}) + \sum_{j=1}^{n-1} q(\eta_j^{(n)})\ \cos \frac{kj\pi}{n} \right) \qquad 0 \leq k \leq n,$$

where $\gamma_0 = \gamma_n = 1/\pi$ and $\gamma_k = 2/\pi,\ 1 \leq k \leq n-1$.

Input variables	*Output variable*
N , the degree n QN , the values of q at the nodes	CO , the Fourier coefficients of q

```
      SUBROUTINE COCHGL(N,QN,CO)
**********************************************************************
*   COMPUTES THE CHEBYSHEV FOURIER COEFFICIENTS OF A POLYNOMIAL
*   INDIVIDUATED BY ITS VALUES AT THE CHEBYSHEV GAUSS-LOBATTO NODES
*   N  = THE DEGREE OF THE POLYNOMIAL
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION  QN(0:*), CO(0:*)
          CO(0) = QN(0)
      IF (N .EQ. 0) RETURN

          PI = 3.14159265358979323846D0
          DN = DFLOAT(N)
          DD = DFLOAT(1+4*(N/2)-2*N)
          CO(0) = .5D0*(QN(0)+QN(N))
          CO(N) = .5D0*(QN(0)+DD*QN(N))
      IF (N .EQ. 1) RETURN

          SO = CO(0)
          SN = CO(N)
          SJ = -1.D0
      DO 1 J=1,N-1
          SO = SO+QN(J)
          SN = SN+QN(J)*SJ
          SJ = -SJ
1     CONTINUE
          CO(0) = SO/DN
          CO(N) = DD*SN/DN

          SK = -1.D0
      DO 2 K=1,N-1
          DK = DFLOAT(K)
          SU = .5D0*(QN(0)+QN(N)*SK)
      DO 3 J=1,N-1
          DJ = DFLOAT(J)
          SU = SU+QN(J)*DCOS(DK*DJ*PI/DN)
3     CONTINUE
          CO(K) = 2.D0*SK*SU/DN
          SK = -SK
2     CONTINUE

      RETURN
      END
```

COLAGR

For any $n \in \mathbf{N}$, the routine computes the n Fourier coefficients c_k, $0 \leq k \leq n-1$, with respect to the Laguerre polynomial basis, of a polynomial q of degree at most $n-1$ determined by the values attained at the Laguerre Gauss-Radau nodes $\eta_j^{(n)}$, $0 \leq j \leq n-1$.

One has

$$c_k = \frac{k!}{\Gamma(k+\alpha+1)} \int_0^{+\infty} q(x)\ L_k^{(\alpha)}(x)\ x^\alpha e^{-x} dx$$

$$= \frac{k!}{\Gamma(k+\alpha+1)} \sum_{j=0}^{n-1} q(\eta_j^{(n)})\ L_k^{(\alpha)}(\eta_j^{(n)})\ \tilde{w}_j^{(n)} \qquad 0 \leq k \leq n-1.$$

Here, $\tilde{w}_j^{(n)}$, $0 \leq j \leq n-1$, are the weights of the Laguerre Gauss-Radau formula, which can be determined by calling subroutine WELAGR.

The nodes can be obtained by calling subroutine ZELAGR.

Input variables	*Output variable*
N , the degree n A , the parameter α B , the parameter β ET , vector of the nodes $\eta_j^{(n)}$ QN , the values of q at the nodes WT , the weights $\tilde{w}_j^{(n)}$	CO , the Fourier coefficients of q

Auxiliary routine: **GAMMAF**

```
      SUBROUTINE COLAGR(N,A,ET,QN,WT,CO)
*********************************************************************
*   COMPUTES THE LAGUERRE FOURIER COEFFICIENTS OF A POLYNOMIAL
*   INDIVIDUATED BY ITS VALUES AT THE LAGUERRE GAUSS-RADAU NODES
*   N  = THE NUMBER OF NODES
*   A  = PARAMETER >-1
*   ET = VECTOR OF THE NODES, ET(I), I=0,N-1
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N-1
*   WT = VECTOR OF THE WEIGHTS, WT(I), I=0,N-1
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N-1
*********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), QN(0:*), WT(0:*), CO(0:*)
      IF (N .EQ. 0) RETURN
          A1  = A+1.D0
      CALL GAMMAF(A1,C)
          SU = 0.D0
      DO 1 J=0,N-1
          SU = SU+QN(J)*WT(J)
          CO(J) = 0.D0
1     CONTINUE
          CO(0) = SU/C
      IF (N .EQ. 1) RETURN

      DO 2 J=0,N-1
          X  = ET(J)
          YP = QN(J)*WT(J)
          Y  = (A1-X)*YP
      DO 3 K=1,N-1
          CO(K) = CO(K)+Y
          DK = DFLOAT(K+1)
          B1 = (2.D0*DK+A-1.D0-X)/DK
          B2 = (DK+A-1.D0)/DK
          YM = Y
          Y  = B1*Y-B2*YP
          YP = YM
3     CONTINUE
2     CONTINUE

      DO 4 K=1,N-1
          DK = DFLOAT(K)
          C  = C*(DK+A)/DK
          CO(K) = CO(K)/C
4     CONTINUE

      RETURN
      END
```

PVJAEX

For any $n \in \mathbf{N}$, the routine computes the value of a polynomial q of degree n and its first and second derivatives at a point x, from the Fourier coefficients of the expansion of q with respect to the Jacobi polynomial basis. More precisely, one has

$$q(x) \;=\; \sum_{k=0}^{n} c_k \; P_k^{(\alpha,\beta)}(x).$$

The values $q'(x)$ and $q''(x)$ are obtained by the recursion formula relating the Jacobi polynomials (see subroutine VAJAPO).

Input variables	*Output variables*
N , the degree n	Y , the value of q in x
A , the parameter α	DY , the value of q' in x
B , the parameter β	$D2Y$, the value of q'' in x
X , the argument x	
CO , the Fourier coefficients of q	

```
      SUBROUTINE PVJAEX(N,A,B,X,CO,Y,DY,D2Y)
**********************************************************************
*   COMPUTES THE VALUE OF A POLYNOMIAL OF DEGREE N AND ITS FIRST AND
*   SECOND DERIVATIVES BY KNOWING THE JACOBI FOURIER COEFFICIENTS
*   N  = THE DEGREE OF THE POLYNOMIAL
*   A  = PARAMETER >-1
*   B  = PARAMETER >-1
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   Y  = VALUE OF THE POLYNOMIAL IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*   D2Y= VALUE OF THE SECOND DERIVATIVE IN X
**********************************************************************
```

```
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*)
          Y   = CO(0)
          DY  = 0.D0
          D2Y = 0.D0
      IF (N .EQ. 0) RETURN

          AB  = A+B
          P   = .5D0*((AB+2.D0)*X+A-B)
          DP  = .5D0*(AB+2.D0)
          D2P = 0.D0
          Y   = CO(0)+P*CO(1)
          DY  = DP*CO(1)
          D2Y = 0.D0
      IF (N .EQ. 1) RETURN

          PP  = 1.D0
          DPP = 0.D0
          D2PP = 0.D0
      DO 1 K=2,N
          DK = DFLOAT(K)
          CC = 2.D0*DK+AB
          C1 = 2.D0*DK*(DK+AB)*(CC-2.D0)
          C2 = (CC-1.D0)*(CC-2.D0)*CC
          C3 = (CC-1.D0)*(A-B)*AB
          C4 = 2.D0*(DK+A-1.D0)*CC*(DK+B-1.D0)
          PM = P
          P  = ((C2*X+C3)*P-C4*PP)/C1
          Y  = Y+P*CO(K)
          PP = PM
          DPM = DP
          DP  = ((C2*X+C3)*DP-C4*DPP+C2*PP)/C1
          DY  = DY+DP*CO(K)
          DPP  = DPM
          D2PM = D2P
          D2P  = ((C2*X+C3)*D2P-C4*D2PP+2.D0*C2*DPP)/C1
          D2Y  = D2Y+D2P*CO(K)
          D2PP = D2PM
1     CONTINUE

      RETURN
      END
```

PVLEEX

For any $n \in \mathbf{N}$, the routine computes the value of a polynomial q of degree n and its first and second derivatives at a point x, from the Fourier coefficients of the expansion of q with respect to the Legendre polynomial basis. More precisely, one has

$$q(x) = \sum_{k=0}^{n} c_k \, P_k(x).$$

The values $q'(x)$ and $q''(x)$ are obtained by the recursion formula relating the Legendre polynomials (see subroutine VALEPO).

Input variables	*Output variables*
N , the degree n	Y , the value of q in x
X , the argument x	DY , the value of q' in x
CO , the Fourier coefficients of q	$D2Y$, the value of q'' in x

```
      SUBROUTINE PVLEEX(N,X,CO,Y,DY,D2Y)
**********************************************************************
*   COMPUTES THE VALUE OF A POLYNOMIAL OF DEGREE N AND ITS FIRST AND
*   SECOND DERIVATIVES BY KNOWING THE LEGENDRE FOURIER COEFFICIENTS
*   N  = THE DEGREE OF THE POLYNOMIAL
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   Y  = VALUE OF THE POLYNOMIAL IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*   D2Y= VALUE OF THE SECOND DERIVATIVE IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*)
          Y   = CO(0)
          DY  = 0.D0
          D2Y = 0.D0
      IF (N .EQ. 0) RETURN

          P   = X
          DP  = 1.D0
          D2P = 0.D0
          Y   = CO(0)+P*CO(1)
          DY  = DP*CO(1)
          D2Y = 0.D0
      IF (N .EQ. 1) RETURN

          PP  = 1.D0
          DPP = 0.D0
          D2PP = 0.D0
      DO 1 K=2,N
          DK = DFLOAT(K)
          C2 = 2.D0*DK-1.D0
          C4 = DK-1.D0
          PM = P
          P  = (C2*X*P-C4*PP)/DK
          Y  = Y+P*CO(K)
          PP = PM
          DPM = DP
          DP  = (C2*X*DP-C4*DPP+C2*PP)/DK
          DY  = DY+DP*CO(K)
          DPP  = DPM
          D2PM = D2P
          D2P  = (C2*X*D2P-C4*D2PP+2.D0*C2*DPP)/DK
          D2Y  = D2Y+D2P*CO(K)
          D2PP = D2PM
1     CONTINUE

      RETURN
      END
```

PVCHEX

For any $n \in \mathbf{N}$, the routine computes the value of a polynomial q of degree n and its first and second derivatives at a point x, from the Fourier coefficients of the expansion of q with respect to the Chebyshev polynomial basis. More precisely, one has

$$q(x) \;=\; \sum_{k=0}^{n} c_k \; T_k(x).$$

The values $q'(x)$ and $q''(x)$ are obtained by the recursion formula relating the Chebyshev polynomials (see subroutine VACHPO).

Input variables	*Output variables*
N , the degree n	Y , the value of q in x
X , the argument x	DY , the value of q' in x
CO , the Fourier coefficients of q	$D2Y$, the value of q'' in x

```
      SUBROUTINE PVCHEX(N,X,CO,Y,DY,D2Y)
**********************************************************************
*   COMPUTES THE VALUE OF A POLYNOMIAL OF DEGREE N AND ITS FIRST AND
*   SECOND DERIVATIVES BY KNOWING THE CHEBYSHEV FOURIER COEFFICIENTS
*   N  = THE DEGREE OF THE POLYNOMIAL
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   Y  = VALUE OF THE POLYNOMIAL IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*   D2Y= VALUE OF THE SECOND DERIVATIVE IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*)
          Y   = CO(0)
          DY  = 0.D0
          D2Y = 0.D0
      IF (N .EQ. 0) RETURN

          P   = X
          DP  = 1.D0
          D2P = 0.D0
          Y   = CO(0)+P*CO(1)
          DY  = DP*CO(1)
          D2Y = 0.D0
      IF (N .EQ. 1) RETURN

          PP  = 1.D0
          DPP = 0.D0
          D2PP = 0.D0
      DO 1 K=2,N
          PM = P
          P  = 2.D0*X*P-PP
          Y  = Y+P*CO(K)
          PP = PM
          DPM = DP
          DP  = 2.D0*X*DP+2.D0*PP-DPP
          DY  = DY+DP*CO(K)
          DPP  = DPM
          D2PM = D2P
          D2P  = 2.D0*X*D2P+4.D0*DPP-D2PP
          D2Y  = D2Y+D2P*CO(K)
          D2PP = D2PM
1     CONTINUE

      RETURN
      END
```

PVLAEX

For any $n \in \mathbf{N}$, the routine computes the value of a polynomial q of degree n and its first and second derivatives at a point x, from the Fourier coefficients of the expansion of q with respect to the Laguerre polynomial basis. More precisely, one has

$$q(x) = \sum_{k=0}^{n} c_k \, L_k^{(\alpha)}(x).$$

The values $q'(x)$ and $q''(x)$ are obtained by the recursion formula relating the Laguerre polynomials (see subroutine VALAPO).

Input variables	*Output variables*
N , the degree n	Y , the value of q in x
A , the parameter α	DY , the value of q' in x
X , the argument x	$D2Y$, the value of q'' in x
CO , the Fourier coefficients of q	

```
      SUBROUTINE PVLAEX(N,A,X,CO,Y,DY,D2Y)
**********************************************************************
*   COMPUTES THE VALUE OF A POLYNOMIAL OF DEGREE N AND ITS FIRST AND
*   SECOND DERIVATIVES BY KNOWING THE LAGUERRE FOURIER COEFFICIENTS
*   N  = THE DEGREE OF THE POLYNOMIAL
*   A  = PARAMETER >-1
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   Y  = VALUE OF THE POLYNOMIAL IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*   D2Y= VALUE OF THE SECOND DERIVATIVE IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*)
          Y   = CO(0)
          DY  = 0.D0
          D2Y = 0.D0
      IF (N .EQ. 0) RETURN

          P   = 1.D0+A-X
          DP  = -1.D0
          D2P = 0.D0
          Y   = CO(0)+P*CO(1)
          DY  = DP*CO(1)
          D2Y = 0.D0
      IF (N .EQ. 1) RETURN

          PP  = 1.D0
          DPP = 0.D0
          D2PP = 0.D0
      DO 1 K=2,N
          DK = DFLOAT(K)
          B1 = (2.D0*DK+A-1.D0-X)/DK
          B2 = (DK+A-1.D0)/DK
          PM = P
          P  = B1*P-B2*PP
          Y  = Y+P*CO(K)
          PP = PM
          DPM = DP
          DP  = B1*DP-PP/DK-B2*DPP
          DY  = DY+DP*CO(K)
          DPP  = DPM
          D2PM = D2P
          D2P  = B1*D2P-2.D0*DPP/DK-B2*D2PP
          D2Y  = D2Y+D2P*CO(K)
          D2PP = D2PM
1     CONTINUE

      RETURN
      END
```

PVHEEX

For any $n \in \mathbf{N}$, the routine computes the value of a polynomial q of degree n and its first and second derivatives at a point x, from the Fourier coefficients of the expansion of q with respect to the Hermite polynomial basis. More precisely, one has

$$q(x) = \sum_{k=0}^{n} c_k \; H_k(x).$$

The values $q'(x)$ and $q''(x)$ are obtained by the recursion formula relating the Hermite polynomials (see subroutine VAHEPO).

Input variables	*Output variables*
N , the degree n	Y , the value of q in x
X , the argument x	DY , the value of q' in x
CO , the Fourier coefficients of q	$D2Y$, the value of q'' in x

```
      SUBROUTINE PVHEEX(N,X,CO,Y,DY,D2Y)
**********************************************************************
*   COMPUTES THE VALUE OF A POLYNOMIAL OF DEGREE N AND ITS FIRST AND
*   SECOND DERIVATIVES BY KNOWING THE HERMITE FOURIER COEFFICIENTS
*   N  = THE DEGREE OF THE POLYNOMIAL
*   X  = THE POINT IN WHICH THE COMPUTATION IS PERFORMED
*   CO = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   Y  = VALUE OF THE POLYNOMIAL IN X
*   DY = VALUE OF THE FIRST DERIVATIVE IN X
*   D2Y= VALUE OF THE SECOND DERIVATIVE IN X
**********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*)
          Y   = CO(0)
          DY  = 0.D0
          D2Y = 0.D0
      IF (N .EQ. 0) RETURN

          P   = 2.D0*X
          DP  = 2.D0
          D2P = 0.D0
          Y   = CO(0)+P*CO(1)
          DY  = DP*CO(1)
          D2Y = 0.D0
      IF (N .EQ. 1) RETURN

          PP  = 1.D0
          DPP = 0.D0
          D2PP = 0.D0
      DO 1 K=2,N
          DK = DFLOAT(K)
          PM = P
          P  = 2.D0*X*P-2.D0*PP*(DK-1.D0)
          Y  = Y+P*CO(K)
          DY = DY+2.D0*DK*PM*CO(K)
          D2Y = D2Y+4.D0*DK*(DK-1.D0)*PP*CO(K)
          PP  = PM
1     CONTINUE

      RETURN
      END
```

NOJAEX

For any $n \in \mathbf{N}$, the routine computes the quantity

$$\mathcal{W} = \left(\int_{-1}^{1} q^2(x) \; (1-x)^\alpha (1+x)^\beta dx \right)^{\frac{1}{2}}$$

$$= \left(\sum_{k=0}^{n} c_k^2 \int_{-1}^{1} [P_n^{(\alpha,\beta)}(x)]^2 (1-x)^\alpha (1+x)^\beta dx \right)^{\frac{1}{2}},$$

where q is a polynomial of degree at most n individuated by the Fourier coefficients c_k, $0 \le k \le n$, with respect to the Jacobi polynomial basis.

Input variables	*Output variable*
N , the degree n A , the parameter α B , the parameter β CO , the Fourier coefficients of q	QW , the norm $\mathcal{W}$

Auxiliary routine: **GAMMAF**

```
      SUBROUTINE NOJAEX(N,A,B,CO,QW)
***************************************************************
*   COMPUTES THE INTEGRAL NORM OF A POLYNOMIAL BY KNOWING THE
*   FOURIER COEFFICIENTS WITH RESPECT TO THE JACOBI BASIS
*   N  = THE DEGREE OF THE POLYNOMIAL
*   A  = PARAMETER >-1
*   B  = PARAMETER >-1
*   CO = COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   QW = WEIGHTED INTEGRAL NORM OF THE POLYNOMIAL
***************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*)

          EPS = 1.D-14
          A1  = A+1.D0
          B1  = B+1.D0
          AB  = A+B
          AB2 = AB+2.D0
          DN = DFLOAT(N)
      CALL GAMMAF(A1,GA1)
      CALL GAMMAF(B1,GB1)
      CALL GAMMAF(AB2,GAB2)
          C  = ((2.D0)**(AB+1.D0))*GA1*GB1/GAB2
          V  = DABS(CO(0))
          QW = V*DSQRT(C)
      IF (N .EQ. 0) RETURN

          SU = 0.D0
      IF (V .LT. EPS) GOTO 1
          SU = C*V*V
1     DO 2 K=1,N
          DK = DFLOAT(K)
          C  = C*(DK+A)*(DK+B)/DK
          V  = DABS(CO(K))
      IF (V .LT. EPS) GOTO 3
          SU = SU+C*V*V/(2.D0*DK+AB+1.D0)
3         C  = C/(DK+AB+1.D0)
2     CONTINUE
          QW = DSQRT(SU)

      RETURN
      END
```

$\boxed{\textbf{NOLEEX}}$

For any $n \in \mathbf{N}$, the routine computes the quantity

$$\mathcal{I} \;=\; \left(\int_{-1}^{1} q^2(x)\,dx\right)^{\frac{1}{2}} \;=\; \left(\sum_{k=0}^{n} \frac{2c_k^2}{2k+1}\right)^{\frac{1}{2}},$$

where q is a polynomial of degree at most n individuated by the Fourier coefficients c_k, $0 \le k \le n$, with respect to the Legendre polynomial basis.

Input variables	*Output variable*
N , the degree n CO , the Fourier coefficients of q	QI , the norm $\mathcal{I}$

```
      SUBROUTINE NOLEEX(N,CO,QI)
***************************************************************
*   COMPUTES THE INTEGRAL NORM OF A POLYNOMIAL BY KNOWING THE
*   FOURIER COEFFICIENTS WITH RESPECT TO THE LEGENDRE BASIS
*   N  = THE DEGREE OF THE POLYNOMIAL
*   CO = COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   QI = INTEGRAL NORM OF THE POLYNOMIAL
***************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*)

          EPS = 1.D-14
          SU = 0.D0
      DO 1 K=0,N
          DK = DFLOAT(K)
          V  = DABS(CO(K))
      IF (V .LT. EPS) GOTO 1
          SU = SU+V*V/(2.D0*DK+1.D0)
1     CONTINUE
          QI = DSQRT(2.D0*SU)

      RETURN
      END
```

NOCHEX

For any $n \in \mathbf{N}$, the routine computes the quantities

$$\mathcal{W} \; = \; \left(\int_{-1}^{1} q^2(x) \, \frac{dx}{\sqrt{1-x^2}} \right)^{\frac{1}{2}} \; = \; \left(\pi c_0^2 \; + \; \frac{\pi}{2} \sum_{k=1}^{n} \, c_k^2 \right)^{\frac{1}{2}},$$

$$\mathcal{I} \; = \; \left(\int_{-1}^{1} q^2(x) \, dx \right)^{\frac{1}{2}} \; = \; \left(\sum_{k=0}^{n} \sum_{m=0}^{n} \, c_k c_m I_{km} \right)^{\frac{1}{2}},$$

$$\text{with} \qquad I_{km} \; = \; \begin{cases} 0 & \text{if } \; k+m \; \text{ is odd,} \\ \frac{1}{1-(k+m)^2} + \frac{1}{1-(k-m)^2} & \text{if } \; k+m \; \text{ is even,} \end{cases}$$

where q is a polynomial of degree at most n individuated by the Fourier coefficients c_k, $0 \le k \le n$, with respect to the Chebyshev polynomial basis.

Input variables	*Output variables*
N , the degree n	QW , the norm $\mathcal{W}$
CO , the Fourier coefficients of q	QI , the norm $\mathcal{I}$

```
      SUBROUTINE NOCHEX(N,CO,QW,QI)
****************************************************************
*    COMPUTES THE INTEGRAL NORMS OF A POLYNOMIAL BY KNOWING THE
*    FOURIER COEFFICIENTS WITH RESPECT TO THE CHEBYSHEV BASIS
*    N  = THE DEGREE OF THE POLYNOMIAL
*    CO = COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*    QW = WEIGHTED INTEGRAL NORM OF THE POLYNOMIAL
*    QI = INTEGRAL NORM OF THE POLYNOMIAL
****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*)

          EPS = 1.D-14
          PR = 1.7724538509055160D0
          R2 = 1.41421356237309515D0
          V  = DABS(CO(0))
          QW = PR*V
          QI = R2*V
      IF (N .EQ. 0) RETURN

          SU = 0.D0
      IF (V .LT. EPS) GOTO 1
          SU = 2.D0*V*V
1     DO 2 K=1,N
          V  = DABS(CO(K))
      IF (V .LT. EPS) GOTO 2
          SU = SU+V*V
2     CONTINUE
          QW = PR*DSQRT(.5D0*SU)

          SU = 0.D0
      DO 3 K=0,N,2
          V  = CO(K)
      DO 3 M=0,N,2
          D1 = 1.D0-DFLOAT((K+M)*(K+M))
          D2 = 1.D0-DFLOAT((K-M)*(K-M))
          C  =1.D0/D1+1.D0/D2
          SU = SU+C*V*CO(M)
3     CONTINUE
      DO 4 K=1,N,2
          V  = CO(K)
      DO 4 M=1,N,2
          D1 = 1.D0-DFLOAT((K+M)*(K+M))
          D2 = 1.D0-DFLOAT((K-M)*(K-M))
          C  =1.D0/D1+1.D0/D2
          SU = SU+C*V*CO(M)
4     CONTINUE
          QI = DSQRT(SU)

      RETURN
      END
```

NOLAEX

For any $n \in \mathbf{N}$, the routine computes the quantity

$$\mathcal{W} = \left(\int_0^{+\infty} q^2(x)\, x^\alpha e^{-x}\, dx \right)^{\frac{1}{2}} = \left(\sum_{k=0}^{n} c_k^2 \, \frac{\Gamma(k+\alpha+1)}{k!} \right)^{\frac{1}{2}},$$

where q is a polynomial of degree at most n individuated by the Fourier coefficients c_k, $0 \leq k \leq n$, with respect to the Laguerre polynomial basis.

Input variables	*Output variable*
N , the degree n A , the parameter α CO , the Fourier coefficients of q	QW , the norm $\mathcal{W}$

Auxiliary routine: **GAMMAF**

```
      SUBROUTINE NOLAEX(N,A,CO,QW)
****************************************************************
*   COMPUTES THE INTEGRAL NORM OF A POLYNOMIAL BY KNOWING THE
*   FOURIER COEFFICIENTS WITH RESPECT TO THE LAGUERRE BASIS
*   N  = THE DEGREE OF THE POLYNOMIAL
*   A  = PARAMETER >-1
*   CO = COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   QW = WEIGHTED INTEGRAL NORM OF THE POLYNOMIAL
****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*)

          EPS = 1.D-14
          A1  = A+1.D0
      CALL GAMMAF(A1,C)
          V  = DABS(CO(0))
          QW = V*DSQRT(C)
      IF (N .EQ. 0) RETURN

          SU = 0.D0
      IF (V .LT. EPS) GOTO 1
          SU = C*V*V
1     DO 2 K=1,N
          DK = DFLOAT(K)
          C  = C*(DK+A)/DK
          V  = DABS(CO(K))
      IF (V .LT. EPS) GOTO 2
          SU = SU+C*V*V
2     CONTINUE
          QW = DSQRT(SU)

      RETURN
      END
```

NOHEEX

For any $n \in \mathbf{N}$, the routine computes the quantity

$$\mathcal{W} \;=\; \left(\int_{-\infty}^{+\infty} q^2(x)\, e^{-x^2}\, dx\right)^{\frac{1}{2}} \;=\; \left(\sqrt{\pi}\sum_{k=0}^{n} c_k^2\, 2^k k!\right)^{\frac{1}{2}},$$

where q is a polynomial of degree at most n individuated by the Fourier coefficients c_k, $0 \le k \le n$, with respect to the Hermite polynomial basis.

Input variables	*Output variable*
N , the degree n CO , the Fourier coefficients of q	QW , the norm $\mathcal{W}$

```
      SUBROUTINE NOHEEX(N,CO,QW)
****************************************************************
*   COMPUTES THE INTEGRAL NORM OF A POLYNOMIAL BY KNOWING THE
*   FOURIER COEFFICIENTS WITH RESPECT TO THE HERMITE BASIS
*   N  = THE DEGREE OF THE POLYNOMIAL
*   CO = COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   QW = WEIGHTED INTEGRAL NORM OF THE POLYNOMIAL
****************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*)

          EPS = 1.D-14
          PR = 1.3313353638003895D0
          R2 = 1.4142135623730951D0
          V  = DABS(CO(0))
          QW = V*PR
      IF (N .EQ. 0) RETURN

          SU = 0.D0
      IF (V .LT. EPS) GOTO 1
          SU = V*V
1         C  = 1.D0
      DO 2 K=1,N
          DK = DFLOAT(K)
          C  = C*R2*DSQRT(DK)
          V  = DABS(CO(K))
      IF (V .LT. EPS) GOTO 2
          SU = SU+(C*V)*(C*V)
2     CONTINUE
          QW = PR*DSQRT(SU)

      RETURN
      END
```

COJADE

For any $n \in \mathbf{N}$, the routine computes the Fourier coefficients of the first derivative $(c_k^{(1)},\ 0 \le k \le n)$ and the second derivative $(c_k^{(2)},\ 0 \le k \le n)$ of a polynomial q of degree at most n determined by the Fourier coefficients $c_k,\ 0 \le k \le n$, with respect to the Jacobi polynomial basis in the case $\alpha = \beta = \gamma$.

The first derivative coefficients are given by

$$c_k^{(1)} = \begin{cases} 0 & \text{if } k = n, \\ \frac{(2n+2\gamma-1)(n+\gamma)}{n+2\gamma} c_n & \text{if } k = n-1, \\ \frac{(2k+2\gamma+1)(k+\gamma+1)}{k+2\gamma+1} \left[\frac{k+\gamma+2}{(2k+2\gamma+5)(k+2\gamma+2)} c_{k+2}^{(1)} + c_{k+1} \right] & \text{if } 1 \le k \le n-2, \\ \frac{\gamma+2}{4\gamma+10} c_2^{(1)} + (\gamma+1) c_1 & \text{if } k = 0. \end{cases}$$

The second derivative coefficients are obtained by assuming $c_k^{(1)}$ in place of c_k in the above formula.

Input variables	*Output variables*
N , the degree n	CD , the Fourier coefficients of q'
G , the parameter γ	$CD2$, the Fourier coefficients of q''
CO , the Fourier coefficients of q	

```
      SUBROUTINE COJADE(N,G,CO,CD,CD2)
**************************************************************************
*   COMPUTES THE FOURIER COEFFICIENTS OF THE DERIVATIVES OF A POLYNOMIAL
*   FROM ITS FOURIER COEFFICIENTS WITH RESPECT TO THE JACOBI BASIS
*   N = THE DEGREE OF THE POLYNOMIAL
*   G = PARAMETER >-1
*   CO = COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   CD = COEFFICIENTS OF THE FIRST DERIVATIVE, CD(I), I=0,N
*   CD2 = COEFFICIENTS OF THE SECOND DERIVATIVE, CD2(I), I=0,N
**************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*), CD(0:*), CD2(0:*)

  CD(N)  = 0.D0
  CD2(N) = 0.D0
      IF (N .EQ. 0) RETURN

  CD(0) = (G+1.D0)*CO(1)
  CD2(N-1) = 0.D0
      IF (N .EQ. 1) RETURN

  DN = DFLOAT(N)
  G2 = 2.D0*G
  CD(N-1) = (2.D0*DN+G2-1.D0)*(DN+G)*CO(N)/(DN+G2)
      DO 1 K=0,N-2
  KR = N-K-2
      IF (KR .NE. 0) THEN
  DK = DFLOAT(KR)
  C1 = (2.D0*DK+G2+1.D0)*(DK+G+1.D0)/(DK+G2+1.D0)
  C2 = (DK+G+2.D0)/((2.D0*DK+G2+5.D0)*(DK+G2+2.D0))
  CD(KR)  = C1*(C2*CD(KR+2)+CO(KR+1))
  CD2(KR) = C1*(C2*CD2(KR+2)+CD(KR+1))
      ELSE
  CD(0) = .25D0*(G+2.D0)*CD(2)/(G+2.5D0)+(G+1.D0)*CO(1)
  CD2(0) = .25D0*(G+2.D0)*CD2(2)/(G+2.5D0)+(G+1.D0)*CD(1)
      ENDIF
1     CONTINUE

      RETURN
      END
```

COLEDE

For any $n \in \mathbf{N}$, the routine computes the Fourier coefficients of the first derivative ($c_k^{(1)}$, $0 \leq k \leq n$) and the second derivative ($c_k^{(2)}$, $0 \leq k \leq n$) of a polynomial q of degree at most n determined by the Fourier coefficients c_k, $0 \leq k \leq n$, with respect to the Legendre polynomial basis.

The first derivative coefficients are given by

$$c_k^{(1)} = \begin{cases} 0 & \text{if } k = n, \\ (2n-1)c_n & \text{if } k = n-1, \\ (2k+1)\left(\frac{1}{2k+5}c_{k+2}^{(1)} + c_{k+1}\right) & \text{if } 0 \leq k \leq n-2. \end{cases}$$

The second derivative coefficients are obtained by assuming $c_k^{(1)}$ in place of c_k in the above formula.

Input variables	*Output variables*
N , the degree n CO , the Fourier coefficients of q	CD , the Fourier coefficients of q' $CD2$, the Fourier coefficients of q''

```
      SUBROUTINE COLEDE(N,CO,CD,CD2)
**************************************************************************
*   COMPUTES THE FOURIER COEFFICIENTS OF THE DERIVATIVES OF A POLYNOMIAL
*   FROM ITS FOURIER COEFFICIENTS WITH RESPECT TO THE LEGENDRE BASIS
*   N = THE DEGREE OF THE POLYNOMIAL
*   CO = COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   CD = COEFFICIENTS OF THE FIRST DERIVATIVE, CD(I), I=0,N
*   CD2 = COEFFICIENTS OF THE SECOND DERIVATIVE, CD2(I), I=0,N
**************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*), CD(0:*), CD2(0:*)

  CD(N)  = 0.D0
  CD2(N) = 0.D0
      IF (N .EQ. 0) RETURN

  DN = DFLOAT(N)
  CD(N-1)  = (2.D0*DN-1.D0)*CO(N)
  CD2(N-1) = 0.D0
      IF (N .EQ. 1) RETURN

      DO 1 K=0,N-2
  KR = N-K-2
  DK = 2.D0*DFLOAT(KR)+1.D0
  CD(KR)  = DK*(CD(KR+2)/(DK+4.D0)+CO(KR+1))
  CD2(KR) = DK*(CD2(KR+2)/(DK+4.D0)+CD(KR+1))
1     CONTINUE

      RETURN
      END
```

COCHDE

For any $n \in \mathbf{N}$, the routine computes the Fourier coefficients of the first derivative $(c_k^{(1)},\ 0 \le k \le n)$ and the second derivative $(c_k^{(2)},\ 0 \le k \le n)$ of a polynomial q of degree at most n determined by the Fourier coefficients $c_k,\ 0 \le k \le n$, with respect to the Chebyshev polynomial basis.

The first derivative coefficients are given by

$$c_k^{(1)} = \begin{cases} 0 & \text{if } k = n, \\ 2nc_n & \text{if } k = n-1, \\ c_{k+2}^{(1)} + 2(k+1)c_{k+1} & \text{if } 1 \le k \le n-2, \\ \frac{1}{2}c_2^{(1)} + c_1 & \text{if } k = 0. \end{cases}$$

The second derivative coefficients are obtained by assuming $c_k^{(1)}$ in place of c_k in the above formula.

Input variables	*Output variables*
N , the degree n	CD , the Fourier coefficients of q'
CO , the Fourier coefficients of q	$CD2$, the Fourier coefficients of q''

```
      SUBROUTINE COCHDE(N,CO,CD,CD2)
************************************************************************
*   COMPUTES THE FOURIER COEFFICIENTS OF THE DERIVATIVES OF A POLYNOMIAL
*   FROM ITS FOURIER COEFFICIENTS WITH RESPECT TO THE CHEBYSHEV BASIS
*   N = THE DEGREE OF THE POLYNOMIAL
*   CO = COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   CD = COEFFICIENTS OF THE FIRST DERIVATIVE, CD(I), I=0,N
*   CD2 = COEFFICIENTS OF THE SECOND DERIVATIVE, CD2(I), I=0,N
************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*), CD(0:*), CD2(0:*)

  CD(N)  = 0.D0
  CD2(N) = 0.D0
      IF (N .EQ. 0) RETURN

  CD(0) = CO(1)
  CD2(N-1) = 0.D0
      IF (N .EQ. 1) RETURN

  DN = DFLOAT(N)
  CD(N-1) = 2.D0*DN*CO(N)
      DO 1 K=0,N-2
  KR = N-K-2
      IF (KR .NE. 0) THEN
  DK = 2.D0*(DFLOAT(KR)+1.D0)
  CD(KR)  = CD(KR+2)+DK*CO(KR+1)
  CD2(KR) = CD2(KR+2)+DK*CD(KR+1)
      ELSE
  CD(0)  = .5D0*CD(2)+CO(1)
  CD2(0) = .5D0*CD2(2)+CD(1)
      ENDIF
1     CONTINUE

      RETURN
      END
```

COLADE

For any $n \in \mathbf{N}$, the routine computes the Fourier coefficients of the first derivative $(c_k^{(1)},\ 0 \le k \le n)$ and the second derivative $(c_k^{(2)},\ 0 \le k \le n)$ of a polynomial q of degree at most n determined by the Fourier coefficients c_k, $0 \le k \le n$, with respect to the Laguerre polynomial basis.

The first derivative coefficients are given by

$$c_k^{(1)} = \begin{cases} 0 & \text{if } k = n, \\ c_{k+1}^{(1)} - c_{k+1} & \text{if } 0 \le k \le n-1. \end{cases}$$

The second derivative coefficients are obtained by assuming $c_k^{(1)}$ in place of c_k in the above formula.

Input variables	*Output variables*
N , the degree n	CD , the Fourier coefficients of q'
CO , the Fourier coefficients of q	$CD2$, the Fourier coefficients of q''

```
      SUBROUTINE COLADE(N,CO,CD,CD2)
***************************************************************************
*   COMPUTES THE FOURIER COEFFICIENTS OF THE DERIVATIVES OF A POLYNOMIAL
*   FROM ITS FOURIER COEFFICIENTS WITH RESPECT TO THE LAGUERRE BASIS
*   N = THE DEGREE OF THE POLYNOMIAL
*   CO = COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*   CD = COEFFICIENTS OF THE FIRST DERIVATIVE, CD(I), I=0,N
*   CD2 = COEFFICIENTS OF THE SECOND DERIVATIVE, CD2(I), I=0,N
***************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*), CD(0:*), CD2(0:*)

  CD(N)  = 0.D0
  CD2(N) = 0.D0
      IF (N .EQ. 0) RETURN

  CD(N-1)  = -CO(N)
  CD2(N-1) = 0.D0
      IF (N .EQ. 1) RETURN

      DO 1 K=0,N-2
  KR = N-K-2
  CD(KR)  = CD(KR+1)-CO(KR+1)
  CD2(KR) = CD2(KR+2)-CD(KR+1)
1     CONTINUE

      RETURN
      END
```

COHEDE

For any $n \in \mathbf{N}$, the routine computes the Fourier coefficients of the first derivative $(c_k^{(1)},\ 0 \leq k \leq n)$ and the second derivative $(c_k^{(2)},\ 0 \leq k \leq n)$ of a polynomial q of degree at most n determined by the Fourier coefficients $c_k,\ 0 \leq k \leq n$, with respect to the Hermite polynomial basis.

The first derivative coefficients are given by

$$c_k^{(1)} = \begin{cases} 0 & \text{if } k = n, \\ 2(k+1)c_{k+1} & \text{if } 0 \leq k \leq n-1. \end{cases}$$

The second derivative coefficients are obtained by assuming $c_k^{(1)}$ in place of c_k in the above formula.

Input variables	*Output variables*
N , the degree n CO , the Fourier coefficients of q	CD , the Fourier coefficients of q' $CD2$, the Fourier coefficients of q''

```
      SUBROUTINE COHEDE(N,CO,CD,CD2)
**************************************************************************
*    COMPUTES THE FOURIER COEFFICIENTS OF THE DERIVATIVES OF A POLYNOMIAL
*    FROM ITS FOURIER COEFFICIENTS WITH RESPECT TO THE HERMITE BASIS
*    N = THE DEGREE OF THE POLYNOMIAL
*    CO = COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*    CD = COEFFICIENTS OF THE FIRST DERIVATIVE, CD(I), I=0,N
*    CD2 = COEFFICIENTS OF THE SECOND DERIVATIVE, CD2(I), I=0,N
**************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CO(0:*), CD(0:*), CD2(0:*)

  CD(N)  = 0.D0
  CD2(N) = 0.D0
      IF (N .EQ. 0) RETURN

  DN = DFLOAT(N)
  CD(N-1)  = 2.D0*DN*CO(N)
  CD2(N-1) = 0.D0
      IF (N .EQ. 1) RETURN

      DO 1 K=0,N-2
  KR = N-K-2
  DK = 2.D0*DFLOAT(KR)+2.D0
  CD(KR)  = DK*CO(KR+1)
  CD2(KR) = DK*CD(KR+1)
1     CONTINUE

      RETURN
      END
```

DEJAGA

For any $n \geq 1$, the routine computes the first derivative of a polynomial q of degree at most $n-1$ at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the Jacobi polynomial of degree n. The polynomial q is individuated by the values attained at the zeroes.

We have the formula

$$q'(\xi_i^{(n)}) \ = \ \sum_{j=1}^{n} d_{ij}^{(1)} \ q(\xi_j^{(n)}),$$

where

$$d_{ij}^{(1)} \ = \ \begin{cases} \dfrac{\frac{d}{dx}P_n^{(\alpha,\beta)}(\xi_i^{(n)})}{\frac{d}{dx}P_n^{(\alpha,\beta)}(\xi_j^{(n)})} \ \dfrac{1}{\xi_i^{(n)} - \xi_j^{(n)}} & \text{if } i \neq j, \\[2ex] \dfrac{(\alpha+\beta+2)\xi_i^{(n)} + \alpha - \beta}{2(1-[\xi_i^{(n)}]^2)} & \text{if } i = j. \end{cases}$$

The Legendre case is obtained by setting $\alpha = \beta = 0$. Similarly, the Chebyshev case is obtained by setting $\alpha = \beta = -\frac{1}{2}$. The zeroes and the values $\frac{d}{dx}P_n^{\alpha,\beta}(\xi_j^{(n)})$, $1 \leq j \leq n$, can be determined by calling subroutine ZEJAGA.

Input variables	*Output variable*
N , the number of zeroes A , the parameter α B , the parameter β CZ , vector of the zeroes $\xi_j^{(n)}$ DZ , the values $\frac{d}{dx}P_n^{(\alpha,\beta)}(\xi_j^{(n)})$ QZ , the values of q at the zeroes	DQZ , the values of q' at the zeroes

```
      SUBROUTINE DEJAGA(N,A,B,CS,DZ,QZ,DQZ)
***********************************************************************
*    COMPUTES THE DERIVATIVE OF A POLYNOMIAL AT THE JACOBI ZEROES FROM
*    THE VALUES OF THE POLYNOMIAL ATTAINED AT THE SAME POINTS
*    N = THE NUMBER OF ZEROES
*    A = PARAMETER >-1
*    B = PARAMETER >-1
*    CS = ZEROES OF THE JACOBI POLYNOMIAL, CS(I), I=1,N
*    DZ = IACOBI DERIVATIVES AT THE ZEROES, DZ(I), I=1,N
*    QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*    DQZ = DERIVATIVES OF THE POLYNOMIAL AT THE ZEROES, DQZ(I), I=1,N
***********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), DZ(1), QZ(1), DQZ(1)
      IF (N .EQ. 0) RETURN

      DO 1 I=1,N
  SU = 0.D0
  CI = CS(I)
  DI = DZ(I)
      DO 2 J=1,N
      IF (I .NE. J) THEN
  CJ = CS(J)
  DJ = DZ(J)
  SU = SU+QZ(J)/(DJ*(CI-CJ))
      ELSE
  SU = SU+.5D0*QZ(I)*((A+B+2.D0)*CI+A-B)/(DI*(1.D0-CI*CI))
      ENDIF
2     CONTINUE
  DQZ(I) = DI*SU
1     CONTINUE

      RETURN
      END
```

DELAGA

For any $n \geq 1$, the routine computes the first derivative of a polynomial q of degree at most $n-1$ at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the Laguerre polynomial of degree n. The polynomial q is individuated by the values attained at the zeroes.

We have the formula

$$q'(\xi_i^{(n)}) \;=\; \sum_{j=1}^{n} d_{ij}^{(1)}\; q(\xi_j^{(n)}),$$

where

$$d_{ij}^{(1)} \;=\; \begin{cases} \dfrac{\frac{d}{dx}L_n^{(\alpha)}(\xi_i^{(n)})}{\frac{d}{dx}L_n^{(\alpha)}(\xi_j^{(n)})}\,\dfrac{1}{\xi_i^{(n)}-\xi_j^{(n)}} & \text{if } i \neq j, \\[3ex] \dfrac{\xi_i^{(n)}-\alpha-1}{2\xi_i^{(n)}} & \text{if } i = j. \end{cases}$$

The zeroes can be determined by calling subroutine ZELAGA.

Input variables	*Output variable*
N , the number of zeroes A , the parameter α CZ , vector of the zeroes $\xi_j^{(n)}$ QZ , the values of q at the zeroes	DQZ , the values of q' at the zeroes

Auxiliary routine: **VALAPO**

```
      SUBROUTINE DELAGA(N,A,CS,QZ,DQZ)
************************************************************************
*   COMPUTES THE DERIVATIVE OF A POLYNOMIAL AT THE LAGUERRE ZEROES FROM
*   THE VALUES OF THE POLYNOMIAL ATTAINED AT THE SAME POINTS
*   N = THE NUMBER OF ZEROES
*   A = PARAMETER >-1
*   CS = ZEROES OF THE LAGUERRE POLYNOMIAL, CS(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   DQZ = DERIVATIVES OF THE POLYNOMIAL AT THE ZEROES, DQZ(I), I=1,N
************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), QZ(1), DQZ(1)
      IF (N .EQ. 0) RETURN

      DO 1 J=1,N
  CJ = CS(J)
      CALL VALAPO(N,A,CJ,Y,DY,D2Y)
  QZ(J) = QZ(J)/DY
1     CONTINUE

      DO 2 I=1,N
  SU = 0.D0
  CI = CS(I)
      CALL VALAPO(N,A,CI,Y,DI,D2Y)
      DO 3 J=1,N
      IF (I .NE. J) THEN
  CJ = CS(J)
  SU = SU+QZ(J)/(CI-CJ)
      ELSE
  SU = SU+.5D0*QZ(I)*(CI-A-1.D0)/CI
      ENDIF
3     CONTINUE
  DQZ(I) = DI*SU
2     CONTINUE

      DO 4 I=1,N
  CI = CS(I)
      CALL VALAPO(N,A,CI,Y,DY,D2Y)
  QZ(I) = DY*QZ(I)
4     CONTINUE

      RETURN
      END
```

DEHEGA

For any $n \geq 1$, the routine computes the first derivative of a polynomial q of degree at most $n-1$ at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the Hermite polynomial of degree n. The polynomial q is individuated by the values attained at the zeroes.

We have the formula

$$q'(\xi_i^{(n)}) = \sum_{j=1}^{n} d_{ij}^{(1)} \; q(\xi_j^{(n)}),$$

where

$$d_{ij}^{(1)} = \begin{cases} \dfrac{H_n'(\xi_i^{(n)})}{H_n'(\xi_j^{(n)})} \dfrac{1}{\xi_i^{(n)} - \xi_j^{(n)}} & \text{if } i \neq j, \\ \\ \xi_i^{(n)} & \text{if } i = j. \end{cases}$$

The zeroes can be determined by calling subroutine ZEHEGA.

Input variables	*Output variable*
N , the number of zeroes CZ , vector of the zeroes $\xi_j^{(n)}$ QZ , the values of q at the zeroes	DQZ , the values of q' at the zeroes

Auxiliary routine: **VAHEPO**

```
      SUBROUTINE DEHEGA(N,CS,QZ,DQZ)
*********************************************************************
*   COMPUTES THE DERIVATIVE OF A POLYNOMIAL AT THE HERMITE ZEROES FROM
*   THE VALUES OF THE POLYNOMIAL ATTAINED AT THE SAME POINTS
*   N = THE NUMBER OF ZEROES
*   CS = ZEROES OF THE HERMITE POLYNOMIAL, CS(I), I=1,N
*   QZ = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*   DQZ = DERIVATIVES OF THE POLYNOMIAL AT THE ZEROES, DQZ(I), I=1,N
*********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CS(1), QZ(1), DQZ(1)
      IF (N .EQ. 0) RETURN

      DO 1 J=1,N
  CJ = CS(J)
      CALL VAHEPO(N,CJ,Y,DY,D2Y)
  QZ(J) = QZ(J)/DY
1     CONTINUE

      DO 2 I=1,N
  SU = 0.D0
  CI = CS(I)
      CALL VAHEPO(N,CI,Y,DI,D2Y)
      DO 3 J=1,N
      IF (I .NE. J) THEN
  CJ = CS(J)
  SU = SU+QZ(J)/(CI-CJ)
      ELSE
  SU = SU+CI*QZ(I)
      ENDIF
3     CONTINUE
  DQZ(I) = DI*SU
2     CONTINUE

      DO 4 I=1,N
  CI = CS(I)
      CALL VAHEPO(N,CI,Y,DY,D2Y)
  QZ(I) = DY*QZ(I)
4     CONTINUE

      RETURN
      END
```

DEJAGL

For any $n \in \mathbf{N}$, the routine computes the first derivative of a polynomial q of degree at most n at the Jacobi Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \le j \le n$. The polynomial q is individuated by the values attained at these nodes.

We have the formula

$$q'(\eta_i^{(n)}) \;=\; \sum_{j=0}^{n} \tilde{d}_{ij}^{(1)} \; q(\eta_j^{(n)}),$$

where

$$\tilde{d}_{ij}^{(1)} \;=\; \begin{cases} \frac{\alpha - n(n+\alpha+\beta+1)}{2(\beta+2)} & \text{if } i = j = 0, \\[2mm] \frac{(\beta+1)P_n^{(\alpha,\beta)}(\eta_i^{(n)})}{(1+\eta_i^{(n)})P_n^{(\alpha,\beta)}(\eta_0^{(n)})} & \text{if } 1 \le i \le n-1, \ j = 0, \\[2mm] \frac{(\beta+1)P_n^{(\alpha,\beta)}(\eta_n^{(n)})}{2(\alpha+1)P_n^{(\alpha,\beta)}(\eta_0^{(n)})} & \text{if } i = n, \ j = 0, \\[2mm] \frac{-P_n^{(\alpha,\beta)}(\eta_0^{(n)})}{(\beta+1)(1+\eta_j^{(n)})P_n^{(\alpha,\beta)}(\eta_j^{(n)})} & \text{if } i = 0, \ 1 \le j \le n-1, \\[2mm] \frac{P_n^{(\alpha,\beta)}(\eta_i^{(n)})}{P_n^{(\alpha,\beta)}(\eta_j^{(n)})} \; \frac{1}{\eta_i^{(n)} - \eta_j^{(n)}} & \text{if } 1 \le i \le n-1, \ 1 \le j \le n-1, \ i \ne j, \\[2mm] \frac{(\alpha+\beta)\eta_i^{(n)} + \alpha - \beta}{2(1-[\eta_i^{(n)}]^2)} & \text{if } 1 \le i = j \le n-1, \\[2mm] \frac{P_n^{(\alpha,\beta)}(\eta_n^{(n)})}{(\alpha+1)(1-\eta_j^{(n)})P_n^{(\alpha,\beta)}(\eta_j^{(n)})} & \text{if } i = n, \ 1 \le j \le n-1, \\[2mm] \frac{-(\alpha+1)P_n^{(\alpha,\beta)}(\eta_0^{(n)})}{2(\beta+1)P_n^{(\alpha,\beta)}(\eta_n^{(n)})} & \text{if } i = 0, \ j = n, \\[2mm] \frac{-(\alpha+1)P_n^{(\alpha,\beta)}(\eta_i^{(n)})}{(1-\eta_i^{(n)})P_n^{(\alpha,\beta)}(\eta_n^{(n)})} & \text{if } 1 \le i \le n-1, \ j = n, \\[2mm] \frac{n(n+\alpha+\beta+1) - \beta}{2(\alpha+2)} & \text{if } i = j = n. \end{cases}$$

The nodes and the values $P_n^{(\alpha,\beta)}(\eta_j^{(n)})$, $0 \le j \le n$, can be determined by calling subroutine ZEJAGL.

Input variables	*Output variable*
N , the degree n A , the parameter α B , the parameter β ET , vector of the nodes $\eta_j^{(n)}$ VN , the values $P_n^{(\alpha,\beta)}(\eta_j^{(n)})$ QN , the values of q at the nodes	DQN , the values of q' at the nodes

```
      SUBROUTINE DEJAGL(N,A,B,ET,VN,QN,DQN)
************************************************************************
*   COMPUTES THE DERIVATIVE OF A POLYNOMIAL AT THE JACOBI GAUSS-LOBATTO
*   NODES FROM THE VALUES OF THE POLYNOMIAL ATTAINED AT THE SAME POINTS
*   N = THE DEGREE OF THE POLYNOMIAL
*   A = PARAMETER >-1
*   B = PARAMETER >-1
*   ET = VECTOR OF THE NODES, ET(I), I=0,N
*   VN = VALUES OF THE IACOBI POLYNOMIAL AT THE NODES, VN(I), I=0,N
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   DQN = DERIVATIVES OF THE POLYNOMIAL AT THE NODES, DQZ(I), I=0,N
************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*), QN(0:*), DQN(0:*)
  DQN(0) = 0.D0
      IF (N .EQ. 0) RETURN

  A1 = A+1.D0
  B1 = B+1.D0
  AB = A+B
  DN = DFLOAT(N)
  C1 = DN*(DN+AB+1.D0)
  C2 = A1*VN(0)/(B1*VN(N))
  DQN(0) = .5D0*((A-C1)*QN(0)/(B+2.D0)-C2*QN(N))
  DQN(N) = .5D0*(QN(0)/C2+(C1-B)*QN(N)/(A+2.D0))
      IF (N .EQ. 1) RETURN

  S1 = DQN(0)
  S2 = DQN(N)
```

```
  C3 = -VN(0)/B1
  C4 = VN(N)/A1
      DO 1 J=1,N-1
  VJ = QN(J)/VN(J)
  EJ = ET(J)
  S1 = S1+C3*VJ/(1.D0+EJ)
  S2 = S2+C4*VJ/(1.D0-EJ)
1     CONTINUE
  DQN(0) = S1
  DQN(N) = S2

      DO 2 I=1,N-1
  VI = VN(I)
  EI = ET(I)
  SU = B1*QN(0)/((1.D0+EI)*VN(0))-A1*QN(N)/((1.D0-EI)*VN(N))
      DO 3 J=1,N-1
      IF (I .NE. J) THEN
  VJ = VN(J)
  EJ = ET(J)
  SU = SU+QN(J)/(VJ*(EI-EJ))
      ELSE
  SU = SU+.5D0*QN(I)*(AB*EI+A-B)/(VI*(1.D0-EI*EI))
      ENDIF
3     CONTINUE
  DQN(I) = VI*SU
2     CONTINUE

      RETURN
      END
```

.

DELEGL

For any $n \in \mathbf{N}$, the routine computes the first derivative of a polynomial q of degree at most n at the Legendre Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \le j \le n$. The polynomial q is individuated by the values attained at these nodes.

We have the formula

$$q'(\eta_i^{(n)}) \;=\; \sum_{j=0}^{n} \tilde{d}_{ij}^{(1)} \; q(\eta_j^{(n)}),$$

where

$$\tilde{d}_{ij}^{(1)} \;=\; \begin{cases} -\frac{1}{4}n(n+1) & \text{if } i=j=0, \\[2ex] \dfrac{P_n(\eta_i^{(n)})}{P_n(\eta_j^{(n)})} \, \dfrac{1}{\eta_i^{(n)} - \eta_j^{(n)}} & \text{if } 0 \le i \le n,\; 0 \le j \le n,\; i \ne j, \\[2ex] 0 & \text{if } 1 \le i = j \le n-1, \\[1ex] \frac{1}{4}n(n+1) & \text{if } i=j=n. \end{cases}$$

The nodes and the values $P_n(\eta_j^{(n)})$, $0 \le j \le n$, can be determined by calling subroutine ZELEGL.

Input variables	*Output variable*
N , the degree n ET , vector of the nodes $\eta_j^{(n)}$ VN , the values $P_n(\eta_j^{(n)})$ QN , the values of q at the nodes	DQN , the values of q' at the nodes

```
      SUBROUTINE DELEGL(N,ET,VN,QN,DQN)
*************************************************************************
*  COMPUTES THE DERIVATIVE OF A POLYNOMIAL AT THE LEGENDRE GAUSS-LOBATTO
*  NODES FROM THE VALUES OF THE POLYNOMIAL ATTAINED AT THE SAME POINTS
*   N = THE DEGREE OF THE POLYNOMIAL
*   ET = VECTOR OF THE NODES, ET(I), I=0,N
*   VN = VALUES OF THE LEGENDRE POLYNOMIAL AT THE NODES, VN(I), I=0,N
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   DQN = DERIVATIVES OF THE POLYNOMIAL AT THE NODES, DQZ(I), I=0,N
*************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*), QN(0:*), DQN(0:*)
  DQN(0) = 0.D0
      IF (N .EQ. 0) RETURN

      DO 1 I=0,N
  SU = 0.D0
  VI = VN(I)
  EI = ET(I)
      DO 2 J=0,N
      IF (I .EQ. J) GOTO 2
  VJ = VN(J)
  EJ = ET(J)
  SU = SU+QN(J)/(VJ*(EI-EJ))
2     CONTINUE
  DQN(I) = VI*SU
1     CONTINUE

  DN = DFLOAT(N)
  C  = .25D0*DN*(DN+1.D0)
  DQN(0) = DQN(0)-C*QN(0)
  DQN(N) = DQN(N)+C*QN(N)

      RETURN
      END
```

DECHGL

For any $n \in \mathbf{N}$, the routine computes the first derivative of a polynomial q of degree at most n at the Chebyshev Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \le j \le n$. The polynomial q is individuated by the values attained at these nodes.
We have the formula

$$q'(\eta_i^{(n)}) \;=\; \sum_{j=0}^{n} \tilde{d}_{ij}^{(1)}\; q(\eta_j^{(n)}),$$

where

$$\tilde{d}_{ij}^{(1)} \;=\; \begin{cases} -\frac{1}{6}(2n^2+1) & \text{if } i=j=0, \\[2mm] \frac{1}{2}(-1)^i/(1+\eta_i^{(n)}) & \text{if } 1 \le i \le n-1,\; j=0, \\[2mm] \frac{1}{2}(-1)^n & \text{if } i=n,\; j=0, \\[2mm] -2(-1)^j/(1+\eta_j^{(n)}) & \text{if } i=0,\; 1 \le j \le n-1, \\[2mm] \dfrac{(-1)^{i+j}}{\eta_i^{(n)}-\eta_j^{(n)}} & \text{if } 1 \le i \le n-1,\; 1 \le j \le n-1,\; i \ne j, \\[4mm] \dfrac{-\eta_i^{(n)}}{2(1-[\eta_i^{(n)}]^2)} & \text{if } 1 \le i = j \le n-1, \\[4mm] 2(-1)^{n+j}/(1-\eta_j^{(n)}) & \text{if } i=n,\; 1 \le j \le n-1, \\[2mm] -\frac{1}{2}(-1)^n & \text{if } i=0,\; j=n, \\[2mm] -\frac{1}{2}(-1)^{n+i}/(1-\eta_i^{(n)}) & \text{if } 1 \le i \le n-1,\; j=n, \\[2mm] \frac{1}{6}(2n^2+1) & \text{if } i=j=n. \end{cases}$$

The nodes can be determined by calling subroutine ZECHGL.

Input variables	*Output variable*
N , the degree n ET , vector of the nodes $\eta_j^{(n)}$ QN , the values of q at the nodes	DQN , the values of q' at the nodes

```
      SUBROUTINE DECHGL(N,ET,QN,DQN)
**************************************************************************
* COMPUTES THE DERIVATIVE OF A POLYNOMIAL AT THE CHEBYSHEV GAUSS-LOBATTO
* NODES FROM THE VALUES OF THE POLYNOMIAL ATTAINED AT THE SAME POINTS
*   N = THE DEGREE OF THE POLYNOMIAL
*   ET = VECTOR OF THE NODES, ET(I), I=0,N
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*   DQN = DERIVATIVES OF THE POLYNOMIAL AT THE NODES, DQZ(I), I=0,N
**************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), QN(0:*), DQN(0:*)
  DQN(0) = 0.D0
      IF (N .EQ. 0) RETURN

  DN = DFLOAT(N)
  CN = (2.D0*DN*DN+1.D0)/6.D0
  SN = DFLOAT(1+4*(N/2)-2*N)
  DQN(0) = -CN*QN(0)-.5D0*SN*QN(N)
  DQN(N) = .5D0*SN*QN(0)+CN*QN(N)
      IF (N .EQ. 1) RETURN

  S1 = DQN(0)
  S2 = DQN(N)
  SGN = -1.D0
      DO 1 J=1,N-1
  EJ = ET(J)
  QJ = 2.D0*SGN*QN(J)
  S1 = S1-QJ/(1.D0+EJ)
  S2 = S2+QJ*SN/(1.D0-EJ)
  SGN = -SGN
1     CONTINUE
  DQN(0) = S1
  DQN(N) = S2
```

```
  SGNI = -1.D0
      DO 2 I=1,N-1
  EI = ET(I)
  SU = .5D0*SGNI*(QN(0)/(1.D0+EI)-SN*QN(N)/(1.D0-EI))
  SGNJ = -1.D0
      DO 3 J=1,N-1
      IF (I .NE. J) THEN
  EJ = ET(J)
  SU = SU+SGNI*SGNJ*QN(J)/(EI-EJ)
      ELSE
  SU = SU-.5D0*EI*QN(I)/(1.D0-EI*EI)
      ENDIF
  SGNJ = -SGNJ
3     CONTINUE
  DQN(I) = SU
  SGNI = -SGNI
2     CONTINUE

      RETURN
      END
```

DELAGR

For any $n \geq 1$, the routine computes the first derivative of a polynomial q of degree at most $n-1$ at the Laguerre Gauss-Radau nodes $\eta_j^{(n)}$, $0 \leq j \leq n-1$. The polynomial q is individuated by the values attained at these nodes.

We have the formula

$$q'(\eta_i^{(n)}) \ = \ \sum_{j=0}^{n-1} \tilde{d}_{ij}^{(1)} \ q(\eta_j^{(n)}),$$

where

$$\tilde{d}_{ij}^{(1)} \ = \ \begin{cases} -(n+1)/(\alpha+2) & \text{if } i=j=0, \\[2ex] \dfrac{(\alpha+1)L_n^{(\alpha)}(\eta_i^{(n)})}{\eta_i^{(n)} \ L_n^{(\alpha)}(\eta_0^{(n)})} & \text{if } 1 \leq i \leq n-1, \ j=0, \\[3ex] \dfrac{-L_n^{(\alpha)}(\eta_0^{(n)})}{(\alpha+1) \ \eta_j^{(n)} \ L_n^{(\alpha)}(\eta_j^{(n)})} & \text{if } i=0, \ 1 \leq j \leq n-1, \\[3ex] \dfrac{L_n^{(\alpha)}(\eta_i^{(n)})}{L_n^{(\alpha)}(\eta_j^{(n)})} \ \dfrac{1}{\eta_i^{(n)} - \eta_j^{(n)}} & \text{if } 1 \leq i \leq n-1, \ 1 \leq j \leq n-1, \ i \neq j, \\[3ex] (\eta_i^{(n)} - \alpha)/2\eta_i^{(n)} & \text{if } 1 \leq i = j \leq n-1. \end{cases}$$

The nodes can be determined by calling subroutine ZELAGR.

Input variables	*Output variable*
N , the number of nodes A , the parameter α ET , vector of the nodes $\eta_j^{(n)}$ QN , the values of q at the nodes	DQN , the values of q' at the nodes

Auxiliary routine: **VALAPO**

```
      SUBROUTINE DELAGR(N,A,ET,QN,DQN)
**************************************************************************
*   COMPUTES THE DERIVATIVE OF A POLYNOMIAL AT THE LAGUERRE GAUSS-RADAU
*   NODES FROM THE VALUES OF THE POLYNOMIAL ATTAINED AT THE SAME POINTS
*   N = THE NUMBER OF NODES
*   A = PARAMETER >-1
*   ET = VECTOR OF THE NODES, ET(I), I=0,N-1
*   QN = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N-1
*   DQN = DERIVATIVES OF THE POLYNOMIAL AT THE NODES, DQZ(I), I=0,N-1
**************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), QN(0:*), DQN(0:*)
  DN = DFLOAT(N)
  DQN(0) = (1.D0-DN)*QN(0)/(A+2.D0)
      IF (N .EQ. 1) RETURN

  A1 = A+1.D0
  SU = DQN(0)
  X  = 0.D0
      CALL VALAPO(N,A,X,Y,DY,D2Y)
  C = Y
      DO 1 J=1,N-1
  EJ = ET(J)
      CALL VALAPO(N,A,EJ,Y,DY,D2Y)
  QN(J) = QN(J)/Y
  SU = SU-C*QN(J)/(A1*EJ)
1     CONTINUE
  DQN(0) = SU

      DO 2 I=1,N-1
  EI = ET(I)
      CALL VALAPO(N,A,EI,Y,DY,D2Y)
  SU = A1*QN(0)/(C*EI)
      DO 3 J=1,N-1
      IF (I .NE. J) THEN
  EJ = ET(J)
  SU = SU+QN(J)/(EI-EJ)
      ELSE
  SU = SU+.5D0*QN(I)*(EI-A)/EI
      ENDIF
3     CONTINUE
  DQN(I) = Y*SU
2     CONTINUE

      DO 4 J=1,N-1
  EJ = ET(J)
      CALL VALAPO(N,A,EJ,Y,DY,D2Y)
  QN(J) = Y*QN(J)
4     CONTINUE

      RETURN
      END
```

DMJAGL

For any $n \in \mathbf{N}$, the routine gives the entries of the $(n+1) \times (n+1)$ matrix $\tilde{d}_{ij}^{(1)}$, $0 \le i \le n$, $0 \le j \le n$, relative to the derivative operator in the space of polynomials of degree at most n, with respect to the Jacobi Gauss-Lobatto nodes. The expression of the entries is provided in the description of subroutine DEJAGL. The nodes and the values $P_n^{(\alpha,\beta)}(\eta_j^{(n)})$, $0 \le j \le n$, can be determined by calling subroutine ZEJAGL.

Input variables	*Output variable*
N , the degree n NM , actual dimension of the matrix A , the parameter α B , the parameter β ET , vector of the nodes $\eta_j^{(n)}$ VN , the values $P_n^{(\alpha,\beta)}(\eta_j^{(n)})$	DMA , derivative matrix

```
      SUBROUTINE DMJAGL(N,NM,A,B,ET,VN,DMA)
*************************************************************************
*  COMPUTES THE ENTRIES OF THE DERIVATIVE MATRIX RELATIVE TO THE
*  IACOBI GAUSS-LOBATTO NODES
*  N   = PARAMETER RELATIVE TO THE DIMENSION OF THE MATRIX
*  NM  = ORDER OF THE MATRIX AS DECLARED IN THE MAIN DIMENSION STATEMENT
*  A   = PARAMETER >-1
*  B   = PARAMETER >-1
*  ET  = VECTOR OF THE NODES, ET(I), I=0,N
*  VN  = VALUES OF THE IACOBI POLYNOMIAL AT THE NODES, VN(I), I=0,N
*  DMA = DERIVATIVE MATRIX, DMA(I,J), I=0,N  J=0,N
*************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*), DMA(0:NM,0:*)
  DMA(0,0) = 0.D0
      IF (N .EQ. 0) RETURN
```

```
      A1 = A+1.D0
      B1 = B+1.D0
      AB = A+B
      DN = DFLOAT(N)
      C1 = DN*(DN+AB+1.D0)
      C2 = A1*VN(0)/(B1*VN(N))
      DMA(0,0) = .5D0*(A-C1)/(B+2.D0)
      DMA(N,N) = .5D0*(C1-B)/(A+2.D0)
      DMA(0,N) = -.5D0*C2
      DMA(N,0) = .5D0/C2
          IF (N .EQ. 1) RETURN

      C3 = VN(0)/B1
      C4 = VN(N)/A1
          DO 1 J=1,N-1
      VJ = VN(J)
      EJ = ET(J)
      DMA(0,J) = -C3/(VJ*(1.D0+EJ))
      DMA(N,J) = C4/(VJ*(1.D0-EJ))
      DMA(J,0) = VJ/(C3*(1.D0+EJ))
      DMA(J,N) = -VJ/(C4*(1.D0-EJ))
1         CONTINUE

          DO 2 I=1,N-1
      VI = VN(I)
      EI = ET(I)
          DO 3 J=1,N-1
          IF (I .NE. J) THEN
      VJ = VN(J)
      EJ = ET(J)
      DMA(I,J) = VI/(VJ*(EI-EJ))
          ELSE
      DMA(I,I) = .5D0*(AB*EI+A-B)/(1.D0-EI*EI)
          ENDIF
3         CONTINUE
2         CONTINUE

          RETURN
          END
```

DMLEGL

For any $n \in \mathbf{N}$, the routine gives the entries of the $(n+1) \times (n+1)$ matrix $\tilde{d}_{ij}^{(1)}$, $0 \leq i \leq n,\ 0 \leq j \leq n$, relative to the derivative operator in the space of polynomials of degree at most n, with respect to the Legendre Gauss-Lobatto nodes. The expression of the entries is provided in the description of subroutine DELEGL. The nodes and the values $P_n(\eta_j^{(n)}),\ 0 \leq j \leq n$, can be determined by calling subroutine ZELEGL.

Input variables	*Output variable*
N , the degree n NM , actual dimension of the matrix ET , vector of the nodes $\eta_j^{(n)}$ VN , the values $P_n(\eta_j^{(n)})$	DMA , derivative matrix

```
      SUBROUTINE DMLEGL(N,NM,ET,VN,DMA)
**************************************************************************
*  COMPUTES THE ENTRIES OF THE DERIVATIVE MATRIX RELATIVE TO THE
*  LEGENDRE GAUSS-LOBATTO NODES
*  N   = PARAMETER RELATIVE TO THE DIMENSION OF THE MATRIX
*  NM  = ORDER OF THE MATRIX AS DECLARED IN THE MAIN DIMENSION STATEMENT
*  ET  = VECTOR OF THE NODES, ET(I), I=0,N
*  VN  = VALUES OF THE LEGENDRE POLYNOMIAL AT THE NODES, VN(I), I=0,N
*  DMA = DERIVATIVE MATRIX, DMA(I,J), I=0,N  J=0,N
**************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), VN(0:*), DMA(0:NM,0:*)
  DMA(0,0) = 0.D0
      IF (N .EQ. 0) RETURN

      DO 1 I=0,N
  VI = VN(I)
  EI = ET(I)
      DO 2 J=0,N
      IF (I .NE. J) THEN
  VJ = VN(J)
  EJ = ET(J)
  DMA(I,J) = VI/(VJ*(EI-EJ))
      ELSE
  DMA(I,I) = 0.D0
      ENDIF
2     CONTINUE
1     CONTINUE

  DN = DFLOAT(N)
  C  = .25D0*DN*(DN+1.D0)
  DMA(0,0) = -C
  DMA(N,N) = C

      RETURN
      END
```

DMCHGL

For any $n \in \mathbf{N}$, the routine gives the entries of the $(n+1) \times (n+1)$ matrix $\tilde{d}_{ij}^{(1)}$, $0 \leq i \leq n$, $0 \leq j \leq n$, relative to the derivative operator in the space of polynomials of degree at most n, with respect to the Chebyshev Gauss-Lobatto nodes. The expression of the entries is provided in the description of subroutine DECHGL. The nodes can be determined by calling subroutine ZECHGL.

Input variables	*Output variable*
N , the degree n NM , actual dimension of the matrix ET , vector of the nodes $\eta_j^{(n)}$	DMA , derivative matrix

```
      SUBROUTINE DMCHGL(N,NM,ET,DMA)
**************************************************************************
*  COMPUTES THE ENTRIES OF THE DERIVATIVE MATRIX RELATIVE TO THE
*  CHEBYSHEV GAUSS-LOBATTO NODES
*  N   = PARAMETER RELATIVE TO THE DIMENSION OF THE MATRIX
*  NM  = ORDER OF THE MATRIX AS DECLARED IN THE MAIN DIMENSION STATEMENT
*  ET  = VECTOR OF THE NODES, ET(I), I=0,N
*  DMA = DERIVATIVE MATRIX, DMA(I,J), I=0,N  J=0,N
**************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), DMA(0:NM,0:*)
  DMA(0,0) = 0.D0
      IF (N .EQ. 0) RETURN
```

```
      DN = DFLOAT(N)
      CN = (2.D0*DN*DN+1.D0)/6.D0
      SN = DFLOAT(1+4*(N/2)-2*N)
      DMA(0,0) = -CN
      DMA(N,N) = CN
      DMA(0,N) = -.5D0*SN
      DMA(N,0) = .5D0*SN
          IF (N .EQ. 1) RETURN

      SGN = -1.D0
          DO 1 J=1,N-1
      EJ = ET(J)
      DMA(0,J) = -2.D0*SGN/(1.D0+EJ)
      DMA(N,J) = 2.D0*SGN*SN/(1.D0-EJ)
      DMA(J,0) = .5D0*SGN/(1.D0+EJ)
      DMA(J,N) = -.5D0*SGN*SN/(1.D0-EJ)
      SGN = -SGN
1         CONTINUE

      SGNI = -1.D0
          DO 2 I=1,N-1
      EI = ET(I)
      SGNJ = -1.D0
          DO 3 J=1,N-1
          IF (I .NE. J) THEN
      EJ = ET(J)
      DMA(I,J) = SGNI*SGNJ/(EI-EJ)
          ELSE
      DMA(I,I) = -.5D0*EI/(1.D0-EI*EI)
          ENDIF
      SGNJ = -SGNJ
3         CONTINUE
      SGNI = -SGNI
2         CONTINUE

          RETURN
          END
```

DMLAGR

For any $n \geq 1$, the routine gives the entries of the $n \times n$ matrix $\tilde{d}_{ij}^{(1)}$, $0 \leq i \leq n-1$, $0 \leq j \leq n-1$, relative to the derivative operator in the space of polynomials of degree at most n, with respect to the Laguerre Gauss-Radau nodes. The expression of the entries is provided in the description of subroutine DELAGR. The nodes can be determined by calling subroutine ZELAGR.

Input variables	*Output variable*
N , the degree n NM , actual dimension of the matrix A , the parameter α ET , vector of the nodes $\eta_j^{(n)}$	DMA , derivative matrix

Auxiliary routine: **VALAPO**

```
      SUBROUTINE DMLAGR(N,NM,A,ET,DMA)
***********************************************************************
*  COMPUTES THE ENTRIES OF THE DERIVATIVE MATRIX RELATIVE TO THE
*  LAGUERRE GAUSS-RADAU NODES
*  N   = DIMENSION OF THE MATRIX
*  NM  = ORDER OF THE MATRIX AS DECLARED IN THE MAIN DIMENSION STATEMENT
*  A   = PARAMETER >-1
*  ET  = VECTOR OF THE NODES, ET(I), I=0,N-1
*  DMA = DERIVATIVE MATRIX, DMA(I,J), I=0,N-1  J=0,N-1
**********************************************************************
```

```
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:*), DMA(0:NM,0:*)
  DN = DFLOAT(N)
  DMA(0,0) = (1.D0-DN)/(A+2.D0)
      IF (N .EQ. 1) RETURN

  A1 = A+1.D0
  X  = 0.D0
      CALL VALAPO(N,A,X,Y,DY,D2Y)
  C = Y
      DO 1 J=1,N-1
  EJ = ET(J)
      CALL VALAPO(N,A,EJ,Y,DY,D2Y)
  DMA(0,J) = -C/(A1*EJ*Y)
  DMA(J,0) = A1*Y/(C*EJ)
1     CONTINUE

      DO 2 I=1,N-1
  EI = ET(I)
      CALL VALAPO(N,A,EI,Y,DY,D2Y)
      DO 3 J=1,N-1
      IF (I .NE. J) THEN
  EJ = ET(J)
  DMA(I,J) = Y/(EI-EJ)
      ELSE
  DMA(I,I) = .5D0*(EI-A)/EI
      ENDIF
3     CONTINUE
2     CONTINUE

      DO 4 J=1,N-1
  EJ = ET(J)
      CALL VALAPO(N,A,EJ,Y,DY,D2Y)
      DO 5 I=1,N-1
      IF (I .EQ. J) GOTO 5
  DMA(I,J) = DMA(I,J)/Y
5     CONTINUE
4     CONTINUE

      RETURN
      END
```

USING THE FAST FOURIER TRANSFORM

FCCHGA

For any $n \geq 1$, the routine computes the n Fourier coefficients c_k, $0 \leq k \leq n-1$, with respect to the Chebyshev polynomial basis, of a polynomial q of degree at most $n-1$ determined by the values attained at the zeroes $\xi_j^{(n)}$, $1 \leq j \leq n$, of the polynomial T_n. One has

$$c_0 = \frac{\gamma_0}{n}, \qquad c_k = \frac{(-1)^k}{n}\left[\gamma_k \ e^{\mathbf{i}k\pi/2n} + \gamma_{n-k} \ e^{-\mathbf{i}k\pi/2n}\right] \quad 1 \leq k \leq n-1,$$

where $\mathbf{i} = \sqrt{-1}$ is the imaginary unity, and

$$\gamma_k = \sum_{j=0}^{n-1} \delta_j \ e^{2\mathbf{i}kj\pi/n}, \quad 0 \leq k \leq n-1,$$

with

$$\delta_j = \begin{cases} q(\xi_{2j+1}^{(n)}) & 0 \leq j \leq n - [\frac{n}{2}] - 1, \\ q(\xi_{2n-2j}^{(n)}) & n - [\frac{n}{2}] \leq j \leq n-1. \end{cases}$$

The NAG FORTRAN Library routine C06EAF is used for the computation of the coefficients γ_k, $0 \leq k \leq n-1$, with the help of the Fast Fourier Transform algorithm.

Input variables	*Output variable*
N , the number of zeroes QZ , the values of q at the zeroes	CO , the Fourier coefficients of q

Auxiliary NAG Library routines are called by C06EAF.

```
      SUBROUTINE FCCHGA(N,QZ,CO)
************************************************************************
*  COMPUTES USING FFT THE CHEBYSHEV FOURIER COEFFICIENTS OF A POLYNOMIAL
*  INDIVIDUATED BY THE VALUES ATTAINED AT THE CHEBYSHEV ZEROES
*  N   = THE NUMBER OF ZEROES
*  QZ  = VALUES OF THE POLYNOMIAL AT THE ZEROES, QZ(I), I=1,N
*  CO  = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N-1
************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION QZ(1), CO(0:*)
      IF(N .EQ. 0) RETURN

          PH = 1.57079632679489661923D0
          R2 = 1.41421356237309515D0
          DN = DFLOAT(N)
          CO(0) = QZ(1)/DN
      IF(N .EQ. 1) RETURN

          N2 = N/2
          C  = 2.D0/DSQRT(DN)
          SN = DFLOAT(1+4*N2-2*N)
          CO(N-N2-1) = QZ(N)
      DO 1 I=1,N2
          CO(I-1) = QZ(2*I-1)
          CO(N-I) = QZ(2*I)
1     CONTINUE

      CALL C06EAF(CO,N,IFAIL)
      IF (IFAIL .NE. 0) THEN
      WRITE(*,*) 'IFAIL IS NOT ZERO IN SUBROUTINE C06EAF'
      ENDIF

          CO(0) = .5D0*C*CO(0)
      IF (2*N2 .EQ. N) THEN
          CO(N2) = C*((-1.D0)**N2)*CO(N2)/R2
      ENDIF
      IF (N .EQ. 2) RETURN
          SM = -1.D0
      DO 2 M=1,N-N2-1
          AR = PH*DFLOAT(M)/DN
          CS = DCOS(AR)
          SI = DSIN(AR)
          V1 = C*SM*(CO(M)*CS+CO(N-M)*SI)
          V2 = C*SM*SN*(CO(M)*SI-CO(N-M)*CS)
          CO(M) = V1
          CO(N-M) = V2
          SM = -SM
2     CONTINUE

      RETURN
      END
```

FCCHGL

For any $n \geq 1$, the routine computes the $n+1$ Fourier coefficients $c_k,\ 0 \leq k \leq n$, with respect to the Chebyshev polynomial basis, of a polynomial q of degree at most n determined by the values attained at the Chebyshev Gauss-Lobatto nodes $\eta_j^{(n)},\ 0 \leq j \leq n$. One has

$$c_0 \;=\; \frac{1}{n}\left[\frac{1}{2}\,q(\eta_0^{(n)}) \;+\; \frac{1}{2}\,q(\eta_n^{(n)}) \;+\; \sum_{j=1}^{n-1} q(\eta_j^{(n)})\right],$$

$$c_n \;=\; \frac{(-1)^n}{n}\left[\frac{1}{2}\,q(\eta_0^{(n)}) \;+\; \frac{(-1)^n}{2}\,q(\eta_n^{(n)}) \;+\; \sum_{j=1}^{n-1}(-1)^j q(\eta_j^{(n)})\right],$$

$$c_k \;=\; \frac{(-1)^k}{2n}\left[\gamma_k\left(1 \;+\; \frac{1}{2\sin(\pi k/n)}\right) \;+\; \overline{\gamma}_{n-k}\left(1 \;-\; \frac{1}{2\sin(\pi k/n)}\right)\right], \quad 1 \leq k \leq n-1,$$

where the overbar denotes complex conjugate, and

$$\gamma_k \;=\; \sum_{j=0}^{n-1} \delta_j\; e^{2\mathbf{i}kj\pi/n}, \quad 1 \leq k \leq n-1,$$

with

$$\delta_j \;=\; \begin{cases} q(\eta_0^{(n)}) & j = 0, \\ q(\eta_{2j}^{(n)}) + \mathbf{i}[q(\eta_{2j+1}^{(n)}) - q(\eta_{2j-1}^{(n)})] & 1 \leq j \leq n - [\frac{n}{2}] - 1, \\ q(\eta_n^{(n)}) & j = \frac{n}{2} \text{ if } n \text{ is even}, \\ q(\eta_{2n-2j}^{(n)}) + \mathbf{i}[q(\eta_{2n-2j-1}^{(n)}) - q(\eta_{2n-2j+1}^{(n)})] & [\frac{n}{2}] + 1 \leq j \leq n-1, \end{cases}$$

being $\mathbf{i} = \sqrt{-1}$ the imaginary unity.

The NAG FORTRAN Library routine C06EBF is used for the computation of the coefficients $\gamma_k,\ 1 \leq k \leq n-1$, with the help of the Fast Fourier Transform algorithm.

Input variables	*Output variable*
N , the degree n QZ , the values of q at the nodes	CO , the Fourier coefficients of q

Auxiliary NAG Library routines are called by C06EBF.

```
      SUBROUTINE FCCHGL(N,QN,CO)
************************************************************************
*  COMPUTES USING FFT THE CHEBYSHEV FOURIER COEFFICIENTS OF A POLYNOMIAL
*  INDIVIDUATED BY ITS VALUES AT THE CHEBYSHEV GAUSS-LOBATTO NODES
*  N   = THE DEGREE OF THE POLYNOMIAL
*  QN  = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*  CO  = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION QN(0:*), CO(0:*)
      IF(N .EQ. 0) RETURN

          PI = 3.14159265358979323846D0
          DN = DFLOAT(N)
          N2 = N/2
          SN = DFLOAT(1+4*N2-2*N)
          S1 = .5D0*(QN(0)+QN(N))
          S2 = .5D0*(SN*QN(0)+QN(N))
          CO(0) = S1
          CO(N) = S2
      IF(N .EQ. 1) RETURN

          CO(0) = QN(0)
      IF (2*N2 .EQ. N) THEN
          CO(N2) = QN(N)
      ENDIF
      IF(N .EQ. 2) GOTO 2

      DO 1 I=1,N-N2-1
          I2 = 2*I
          CO(I) = QN(I2)
          CO(N-I) = QN(I2-1)-QN(I2+1)
1     CONTINUE
```

```
2     CALL C06EBF(CO,N,IFAIL)
      IF (IFAIL .NE. 0) THEN
      WRITE(*,*) 'IFAIL IS NOT ZERO IN SUBROUTINE C06EBF'
      ENDIF

          SJ = -1.D0
      DO 3 J=1,N-1
          S1 = S1+QN(J)
          S2 = S2+SJ*SN*QN(J)
          SJ = -SJ
3     CONTINUE
          CO(0) = S1/DN
          CO(N) = S2/DN
          C  = .5D0/DSQRT(DN)
      IF (2*N2 .EQ. N) THEN
          CO(N2) = 2.D0*C*((-1.D0)**N2)*CO(N2)
      ENDIF
      IF(N .EQ. 2) RETURN

          SM = -1.D0
      DO 4 M=1,N-N2-1
          AR = PI*DFLOAT(M)/DN
          SI = .5D0/DSIN(AR)
          V1 = CO(M)*(1.D0+SI)+CO(N-M)*(1.D0-SI)
          V2 = CO(M)*(1.D0-SI)+CO(N-M)*(1.D0+SI)
          CO(M) = C*SM*V1
          CO(N-M) = C*SM*SN*V2
          SM = -SM
4     CONTINUE

      RETURN
      END
```

FVCHGL

For any $n \geq 1$, the routine computes the values of a polynomial q of degree at most n at the Chebyshev Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \leq j \leq n$, by knowing the $n+1$ Fourier coefficients c_k, $0 \leq k \leq n$, of q with respect to the Chebyshev polynomial basis. One has

$$q(\eta_0^{(n)}) = \sum_{k=0}^{n} (-1)^k c_k, \qquad q(\eta_n^{(n)}) = \sum_{k=0}^{n} c_k,$$

$$q(\eta_j^{(n)}) = \frac{1}{4}\left[\gamma_j\left(1 + \frac{1}{2\sin(\pi j/n)}\right) + \overline{\gamma}_{n-j}\left(1 - \frac{1}{2\sin(\pi j/n)}\right)\right], \quad 1 \leq j \leq n-1,$$

where the overbar denotes complex conjugate, and

$$\gamma_j = \sum_{k=0}^{n-1} \delta_k \, e^{2\mathbf{i}kj\pi/n}, \quad 1 \leq j \leq n-1,$$

with

$$\delta_k = \begin{cases} 2c_0 & k = 0 \\ c_{2k} + \mathbf{i}(c_{2k-1} - c_{2k+1}) & 1 \leq k \leq \frac{n}{2} - 1 \\ 2c_n & k = \frac{n}{2} \\ c_{2n-2k} + \mathbf{i}(c_{2n-2k-1} - c_{2n-2k+1}) & \frac{n}{2} + 1 \leq k \leq n-1 \end{cases} \qquad \text{for } n \text{ even;}$$

$$\delta_k = \begin{cases} 2c_0 & k = 0 \\ c_{2k} + \mathbf{i}(c_{2k-1} - c_{2k+1}) & 1 \leq k \leq \frac{n-3}{2} \\ c_{n-1} + \mathbf{i}(c_{n-2} - 2c_n) & k = \frac{n-1}{2} \\ c_{n-1} + \mathbf{i}(2c_n - c_{n-2}) & k = \frac{n+1}{2} \\ c_{2n-2k} + \mathbf{i}(c_{2n-2k-1} - c_{2n-2k+1}) & \frac{n+3}{2} \leq k \leq n-1 \end{cases} \qquad \text{for } n \text{ odd;}$$

being $\mathbf{i} = \sqrt{-1}$ the imaginary unity.

The NAG FORTRAN Library routine C06EBF is used for the computation of the coefficients γ_j, $1 \leq j \leq n-1$, with the help of the Fast Fourier Transform algorithm.

Input variables	*Output variable*
N , the degree n CO , the Fourier coefficients of q	QN , the values of q at the nodes

Auxiliary NAG Library routines are called by C06EBF.

```
      SUBROUTINE FVCHGL(N,CO,QN)
*******************************************************************
*  COMPUTES USING FFT THE VALUES OF A POLYNOMIAL AT THE CHEBYSHEV
*  GAUSS-LOBATTO NODES FROM ITS CHEBYSHEV FOURIER COEFFICIENTS
*  N   = THE DEGREE OF THE POLYNOMIAL
*  CO  = FOURIER COEFFICIENTS OF THE POLYNOMIAL, CO(I), I=0,N
*  QN  = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*******************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION QN(0:*), CO(0:*)
      IF(N .EQ. 0) RETURN

          PI = 3.14159265358979323846D0
          DN = DFLOAT(N)
          N2 = N/2
          SN = DFLOAT(1+4*N2-2*N)
          S1 = CO(0)+SN*CO(N)
          S2 = CO(0)+CO(N)
          QN(0) = S1
          QN(N) = S2
      IF(N .EQ. 1) RETURN

          QN(0) = 2.D0*CO(0)
      IF (2*N2 .EQ. N) THEN
          QN(N2) = 2.D0*CO(N)
      ENDIF
      IF(N .EQ. 2) GOTO 2
```

```
      DO 1 I=1,N-N2-1
          I2 = 2*I
          QN(N-I) = CO(I2+1)-CO(I2-1)
          QN(I) = CO(I2)
1     CONTINUE
      IF (2*N2 .NE. N) THEN
          QN(N2+1) = 2.DO*CO(N)-CO(N-2)
      ENDIF

2     CALL CO6EBF(QN,N,IFAIL)
      IF (IFAIL .NE. 0) THEN
      WRITE(*,*) 'IFAIL IS NOT ZERO IN SUBROUTINE CO6EBF'
      ENDIF

          SJ = -1.DO
      DO 3 J=1,N-1
          S1 = S1+SJ*CO(J)
          S2 = S2+CO(J)
          SJ = -SJ
3     CONTINUE
          QN(0) = S1
          QN(N) = S2
          C  = .25DO*DSQRT(DN)
      IF (2*N2 .EQ. N) THEN
          QN(N2) = 2.DO*C*QN(N2)
      ENDIF
      IF(N .EQ. 2) RETURN

      DO 4 M=1,N-N2-1
          AR = PI*DFLOAT(M)/DN
          SI = .5DO/DSIN(AR)
          V1 = QN(M)*(1.DO+SI)+QN(N-M)*(1.DO-SI)
          V2 = QN(M)*(1.DO-SI)+QN(N-M)*(1.DO+SI)
          QN(M) = C*V1
          QN(N-M) = C*V2
4     CONTINUE

      RETURN
      END
```

FDCHGL

For any $n \geq 1$, the routine computes the Fourier coefficients $c_k^{(1)}$, $0 \leq k \leq n$, and the values at the Chebyshev Gauss-Lobatto nodes $\eta_j^{(n)}$, $0 \leq j \leq n$, of the derivative of a polynomial q of degree at most n, by knowing the values of q at the nodes.
The computation is performed by first computing the Fourier coefficients c_k, $0 \leq k \leq n$, of q using FCCHGL. Then, with the recursion formula given in the description of subroutine COCHDE, one obtains the coefficients $c_k^{(1)}$, $0 \leq k \leq n$. Finally, the quantities $q'(\eta_j^{(n)})$, $0 \leq j \leq n$, are evaluated using subroutine FVCHGL.
Note that the NAG FORTRAN Library routine C06EBF is called both in subroutines FCCHGL and FVCHGL.

Input variables	*Output variables*
N , the degree n QN , the values of q at the nodes	CD , the Fourier coefficients of q' DQN , the values of q' at the nodes

Auxiliary routines: **FCCHGL, FVCHGL**

```
      SUBROUTINE FDCHGL(N,QN,CD,DQN)
*********************************************************************
*  COMPUTES USING FFT THE FOURIER COEFFICIENTS AND THE VALUES AT THE
*  CHEBYSHEV GAUSS-LOBATTO NODES OF THE DERIVATIVE OF A POLYNOMIAL
*  FROM THE VALUES ATTAINED BY THE POLYNOMIAL AT THE NODES
*  N   = THE DEGREE OF THE POLYNOMIAL
*  QN  = VALUES OF THE POLYNOMIAL AT THE NODES, QN(I), I=0,N
*  CD  = FOURIER COEFFICIENTS OF THE DERIVATIVE, CD(I), I=0,N
*  DQN = VALUES OF THE DERIVATIVE AT THE NODES, DQN(I), I=0,N
*********************************************************************
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION QN(0:*), CD(0:*), DQN(0:*)
      IF(N .EQ. 0) RETURN

      CALL FCCHGL(N,QN,DQN)
          CD(N) = 0.D0
          CD(0) = DQN(1)
      IF(N .EQ. 1) GOTO 2

          DN = DFLOAT(N)
          CD(N-1) = 2.D0*DN*DQN(N)
      DO 1 K=0,N-2
          KR = N-K-2
      IF(KR .NE. 0) THEN
          DK = 2.D0*(DFLOAT(KR)+1.D0)
          CD(KR) = CD(KR+2)+DK*DQN(KR+1)
      ELSE
          CD(0) = .5D0*CD(2)+DQN(1)
      ENDIF
1     CONTINUE

2     CALL FVCHGL(N,CD,DQN)

      RETURN
      END
```

AN EXAMPLE

For any $n \geq 1$, we would like to compute the approximated solution to the boundary-value problem

$$\begin{cases} -U'' + \tau U = f & \text{in }]-1,1[, \quad \tau > 0, \\ U(-1) = \sigma_1, \\ U(1) = \sigma_2, \end{cases}$$

by the collocation method at the Legendre Gauss-Lobatto points.

Thus, the approximating polynomial s_n of degree n satisfies the set of equations

$$\begin{cases} -s_n''(\eta_i^{(n)}) + \tau s_n(\eta_i^{(n)}) = f(\eta_i^{(n)}) & 1 \leq i \leq n-1, \\ s_n(\eta_0^{(n)}) = \sigma_1, \\ s_n(\eta_n^{(n)}) = \sigma_2. \end{cases}$$

In order to show how some of the routines described in this book can be called by a main program, we provide a simple example. The program SAMPLE evaluates the approximating polynomial s_n for $2 \leq n \leq 10$, when for instance $\tau = 1$, $\sigma_1 = \sigma_2 = 0$, and $f(x) \equiv 1$, $x \in]-1,1[$.

```
      PROGRAM SAMPLE
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ET(0:10), VN(0:10), WT(0:10)
      DIMENSION SO(0:10), FU(0:10), DMA(0:10,0:10)
      DIMENSION DMA2(0:10,0:10), QN(0:10), WKSPCE(0:10)

C  SET THE PARAMETERS
      SIGMA1 = 0.D0
      SIGMA2 = 0.D0
      TAU = 1.D0
      EF  = 1.D0
      EXP   = 2.718281828459045O9D0
      COST  = EXP/(1.D0+EXP*EXP)

      DO 1 N=2,10

C  COMPUTATION OF THE NODES, WEIGHTS AND DERIVATIVE MATRIX
      CALL ZELEGL(N,ET,VN)
      CALL WELEGL(N,ET,VN,WT)
      CALL DMLEGL(N,10,ET,VN,DMA)
```

```
C  CONSTRUCTION OF THE MATRIX CORRESPONDING TO THE
C  DIFFERENTIAL OPERATOR
      DO 2 I=0,N
      DO 2 J=0,N
      SUM = 0.D0
      DO 3 K=0,N
      SUM = SUM + DMA(I,K)*DMA(K,J)
3     CONTINUE
      OPER = -SUM
      IF(I.EQ.J) OPER = -SUM + TAU
      DMA2(I,J) = OPER
2     CONTINUE

C  CHANGE OF THE ENTRIES OF THE MATRIX ACCORDING TO THE
C  BOUNDARY CONDITIONS
      DO 4 J=0,N
      DMA2(0,J) = 0.D0
      DMA2(N,J) = 0.D0
4     CONTINUE
      DMA2(0,0) = 1.D0
      DMA2(N,N) = 1.D0

C  CONSTRUCTION OF THE RIGHT-HAND SIDE VECTOR
      DO 5 I=1,N-1
      FU(I) = EF
5     CONTINUE
      FU(0) = SIGMA1
      FU(N) = SIGMA2

C  SOLUTION OF THE LINEAR SYSTEM
      N1 = N + 1
      CALL F04AAF(DMA2,11,FU,11,N1,1,SO,11,WKSPCE,IFAIL)

C  COMPARISON WITH THE EXACT SOLUTION
      DO 6 I=0,N
      EI = DEXP(ET(I))
      QN(I) = SO(I)-1.D0+COST*(EI+1.D0/EI)
6     CONTINUE

C  NORM OF THE ERROR VECTOR
      CALL NOLEGL(N,VN,QN,WT,QI,QS,QM)
      WRITE(*,*) N, QI

1     CONTINUE
      RETURN
      END
```

The program allows the computation of the error

$$E_n \;=\; \left(\int_{-1}^{1} (s_n - I_n U)(x)\; dx\right)^{\frac{1}{2}},$$

where $I_n U$ is the n-degree interpolant polynomial at the nodes, of the exact solution given by

$$U(x) \;=\; 1 \;-\; \frac{e}{1+e^2}(e^x \;+\; e^{-x}).$$

The output results are shown in the following table.

n	E_n
2	$0.19222796835423791 2E-01$
3	$0.437255201408398857E-03$
4	$0.544146318231265948E-04$
5	$0.585754215000575047E-06$
6	$0.142928575377323905E-06$
7	$0.908588864973060870E-09$
8	$0.278096559701610084E-09$
9	$0.117001702046232747E-11$
10	$0.403084195089551988E-12$

To solve the system of $n+1$ linear equations we used the NAG routine F04AAF.

TIMING

In this section, we provide an asymptotic estimate, with respect to the parameter n, of the CPU time needed to run each routine. The results, measured in microseconds, are only qualitative since they strongly depend on the computer architecture. The times of the routines using the Fast Fourier Transform are related to the case in which n is a power of two.

VAJAPO	$104\ n$
VALEPO	$66\ n$
VACHPO	$39\ n$
VALAPO	$68\ n$
VAHEPO	$24\ n$
VALASF	$94\ n$
VAHESF	$50\ n$
ZEJAGA	$790\ n^2$
ZELEGA	$260\ n^2$
ZECHGA	$83\ n$
ZELAGA	$580\ n^2$
ZEHEGA	$150\ n^2$

ZEJAGL	$740\ n^2$
ZELEGL	$250\ n^2$
ZECHGL	$22\ n$
ZELAGR	$630\ n^2$
WEJAGA	$70\ n$
WELEGA	$15\ n$
WECHGA	$6\ n$
WELAGA	$75\ n^2$
WEHEGA	$47\ n^2$
WEJAGL	$200\ n^2$
WELEGL	$15\ n$
WECHGL	$8\ n$

WELAGR	130 n^2
WECHCC	57 n^2
INJAGA	130 n
INLEGA	95 n
INCHGA	99 n
INLAGA	22 n^2
INHEGA	12 n^2
INJAGL	140 n
INLEGL	100 n
INCHGL	130 n
INLAGR	23 n^2
NOLEGA	13 n
NOCHGA	100 n^2
NOJAGL	100 n
NOLEGL	30 n
NOCHGL	130 n^2
COJAGA	71 n^2
COLEGA	33 n^2
COCHGA	48 n^2
COLAGA	40 n^2

COHEGA	26 n^2
COJAGL	80 n^2
COLEGL	35 n^2
COCHGL	50 n^2
COLAGR	40 n^2
PVJAEX	120 n
PVLEEX	82 n
PVCHEX	53 n
PVLAEX	86 n
PVHEEX	52 n
NOJAEX	50 n
NOLEEX	13 n
NOCHEX	20 n^2
NOLAEX	36 n
NOHEEX	57 n
COJADE	73 n
COLEDE	46 n
COCHDE	40 n
COLADE	27 n
COHEDE	27 n

DEJAGA	23 n^2
DELAGA	220 n^2
DEHEGA	92 n^2
DEJAGL	25 n^2
DELEGL	23 n^2
DECHGL	27 n^2
DELAGR	220 n^2
DMJAGL	29 n^2

DMLEGL	30 n^2
DMCHGL	30 n^2
DMLAGR	230 n^2
FCCHGA	13 $n\log_2 n$
FCCHGL	13 $n\log_2 n$
FVCHGL	13 $n\log_2 n$
FDCHGL	30 $n\log_2 n$

DIMENSIONING

Zeroes of the orthogonal polynomials of degree n	$\xi_i^{(n)} \quad 1 \le i \le n$	`CS(I), I=1,N`
Derivatives of the Jacobi polynomial of degree n at the zeroes	$\frac{d}{dx} P_n^{(\alpha,\beta)}(\xi_i^{(n)}) \quad 1 \le i \le n$	`DZ(I), I=1,N`
Derivatives of the Legendre polynomial of degree n at the zeroes	$P_n'(\xi_i^{(n)}) \quad 1 \le i \le n$	`DZ(I), I=1,N`
Derivatives of the Chebyshev polynomial of degree n at the zeroes	$T_n'(\xi_i^{(n)}) \quad 1 \le i \le n$	`DZ(I), I=1,N`
Derivatives of the scaled Laguerre function of degree n at the zeroes	$\frac{d}{dx} \hat{L}_n^{(\alpha)}(\xi_i^{(n)}) \quad 1 \le i \le n$	`DZ(I), I=1,N`
Derivatives of the scaled Hermite function of degree n at the zeroes	$\hat{H}_n'(\xi_i^{(n)}) \quad 1 \le i \le n$	`DZ(I), I=1,N`
Nodes of the Gauss-Lobatto formula of order n	$\eta_i^{(n)} \quad 0 \le i \le n$	`ET(I), I=0,N`
Nodes of the Laguerre Gauss-Radau formula of order n	$\eta_i^{(n)} \quad 0 \le i \le n-1$	`ET(I), I=0,N-1`
Values of the Jacobi polynomial of degree n at the nodes	$P_n^{(\alpha,\beta)}(\eta_i^{(n)}) \quad 0 \le i \le n$	`VN(I), I=0,N`
Values of the Legendre polynomial of degree n at the nodes	$P_n(\eta_i^{(n)}) \quad 0 \le i \le n$	`VN(I), I=0,N`
Values of the scaled Laguerre function of degree n at the nodes	$\hat{L}_n^{(\alpha)}(\eta_i^{(n)}) \quad 0 \le i \le n-1$	`VN(I), I=0,N-1`
Weights of the Gauss formula of order n	$w_i^{(n)} \quad 1 \le i \le n$	`WE(I), I=1,N`
Weights of the Gauss-Lobatto formula of order n	$\tilde{w}_i^{(n)} \quad 0 \le i \le n$	`WT(I), I=0,N`
Weights of the Laguerre Gauss-Radau formula of order n	$\tilde{w}_i^{(n)} \quad 0 \le i \le n-1$	`WT(I), I=0,N-1`
Weights of the Clenshaw-Curtis formula of order $2n$	$\chi_i^{(2n)} \quad 0 \le i \le 2n$	`WK(I), I=0,2*N`

Values of a polynomial of degree $n-1$ at the zeroes	$q(\xi_i^{(n)})\quad 1 \le i \le n$	`QZ(I), I=1,N`
Values of a polynomial of degree n at the Gauss-Lobatto nodes	$q(\eta_i^{(n)})\quad 0 \le i \le n$	`QN(I), I=0,N`
Values of a polynomial of degree $n-1$ at the Laguerre Gauss-Radau nodes	$q(\eta_i^{(n)})\quad 0 \le i \le n-1$	`QN(I), I=0,N-1`
Fourier coefficients of a polynomial of degree n	$c_k\quad 0 \le k \le n$	`CO(K), K=0,N`
Fourier coefficients of the derivative of a polynomial of degree n	$c_k^{(1)}\quad 0 \le k \le n$	`CD(K), K=0,N`
Fourier coefficients of the second derivative of a polynomial of degree n	$c_k^{(2)}\quad 0 \le k \le n$	`CD2(K), K=0,N`
Derivative of a polynomial of degree $n-1$ at the zeroes	$q'(\xi_i^{(n)})\quad 1 \le i \le n$	`DQZ(I), I=1,N`
Derivative of a polynomial of degree n at the Gauss-Lobatto nodes	$q'(\eta_i^{(n)})\quad 0 \le i \le n$	`DQN(I), I=0,N`
Derivative of a polynomial of degree $n-1$ at the Laguerre Gauss-Radau nodes	$q'(\eta_i^{(n)})\quad 0 \le i \le n-1$	`DQN(I), I=0,N-1`
Derivative matrix relative to the Gauss-Lobatto nodes	$\tilde{d}_{ij}^{(1)}\quad 0 \le i \le n,\ 0 \le j \le n$	`DMA(I,J), I=0,N J=0,N`
Derivative matrix relative to the Laguerre Gauss-Radau nodes	$\tilde{d}_{ij}^{(1)}\quad 0 \le i \le n-1,\ 0 \le j \le n-1$	`DMA(I,J), I=0,N-1 J=0,N-1`

Bibliography

[1] C. Bernardi & Y. Maday, *Approximations Spectrales de Problèmes aux Limites Elliptiques*, Mathematiques & Applications, n.10, Springer, Paris (1992).

[2] J.P. Boyd, *Chebyshev & Fourier Spectral Methods*, Lecture Notes in Engineering, Springer, New York (1989).

[3] C. Canuto, M.Y. Hussaini, A. Quarteroni & T.A. Zang, *Spectral Methods in Fluid Dynamics*, Springer Series in Computational Physics, Springer, New York (1988).

[4] D. Funaro, *Polynomial Approximation of Differential Equations*, Lecture Notes in Physics, m8, Springer, Heidelberg (1992).

[5] D. Gottlieb & S.A. Orszag, *Numerical Analysis of Spectral Methods, Theory and Applications*, CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM, Philadelphia (1977).

[6] Y.L. Luke, *The Special Functions and their Applications*, Academic Press, New York (1969).

[7] B. Mercier, *An Introduction to the Numerical Analysis of Spectral Methods*, Springer, Berlin (1989).