1 SUMMARY

To provide **weights and zeros for Gaussian type quadratures** of integrals of the form

$$\int_0^\infty e^{-x} f(x)\, dx$$

where, if w_i and x_i for i = 1, 2,..., k are the weights and zeros of the quadrature of order k, an approximation to the integral is given by

$$\sum_{i=1}^{k} w_i f(x_i)$$

The weights and zeros are calculated to give exact results when $f(x)$ is a polynomial of degree $2k$–1 in $z=\dfrac{x}{a+x}$ where a is chosen by the user. A range of quadratures may be requested based upon polynomials $P_k(z)$ for orders k=n, n+1,..., m. The calculations are relatively expensive and the subroutine should only be used when the quadrature is to be applied to many integrals. The weights and zeros produced by the subroutine will in many cases be found to be superior to those of Gauss-Laguerre for functions $f(x)$ which tend to a finite limit as $x \rightarrow \infty$, provided the parameter a is chosen appropriately.

ATTRIBUTES — **Version:** 1.0.0. **Remark:** Use of this subroutine is likely to give better results than Gauss-Laguerre for bounded $f(x)$. **Types:** QC02A; QC02AD. **Calls:** FD05, NB01 and QA05. **Original date:** 1965. **Origin:** A.R.Curtis, Harwell.

2 HOW TO USE THE PACKAGE

2.1 Argument list

The single precision version

```
CALL QC02A(EPS,A,M,N,X,W)
```

The double precision version

```
CALL QC02AD(EPS,A,M,N,X,W)
```

EPS is a REAL (DOUBLE PRECISION in the D version) variable set by the user to the absolute accuracy required in the zeros and weights.

A is a REAL (DOUBLE PRECISION in the D version) variable which must be set by the user to the value of the parameter a used in the transformation of the variable $z=\dfrac{x}{a+x}$.

M,N are INTEGER variables which must be set by the user to define the range of quadratures to be computed, i.e. the weights and zeros of orders k = n, n+1,..., m are computed. In the extremely unlikely event that NB01 fails at some stage to find a root, M will be reset by QC02 to the highest order actually computed.

X is a REAL (DOUBLE PRECISION in the D version) two-dimensional array of dimensions 21 by 21 in which the subroutine puts the zeros. These are stored such that the k zeros for the quadrature of degree $2k$–1 are returned in X(k,1), X(k,2),..., X(k,k). The zeros are arranged in decreasing order of magnitude.

W is a REAL (DOUBLE PRECISION in the D version) two-dimensional array of dimensions 21 by 21 in which the subroutine puts the weights. These are stored so as to correspond to the zeros returned in the array X.

Note that the subroutine only computes the weights w_i and zeros x_i for i = 1, 2,..., k. The user is expected to obtain the integral value by writing code to evaluate the approximation

$$\sum_{i=1}^{k} w_i f(x_i)$$

3 GENERAL INFORMATION

Use of common: the first 532 bytes of unnamed common is used.

Workspace: see use of common.

Other routines called directly: `FD05`, `NB01`, `QA05` and two private subroutines to the `QC02` package `QC02B/BD` and `QC02C/CD`.

Input/output: the values of the quadrature weights and zeros are always printed. An error diagnostic is possible, but unlikely.

4 METHOD

Orthogonal polynomials $P_r\left(\frac{x}{a+x}\right)$ such that

$$\int_0^\infty e^{-x} P_r P_s \, dx = \begin{cases} 0 & \text{for } r \neq s \\ C_r \neq 0 & \text{for } r = s \end{cases}$$

for r=1, 2,..., m are derived, and putting $z = \frac{x}{a+x}$ their zeros are calculated, i.e. for $P_r(z)$, z_1, z_2,..., z_r say, from which the required Gaussian zeros are obtained $x_j = \frac{az_j}{1-z_j}$. The corresponding weights are found by

$$w_j = \alpha C_{r-1} / P_{r-1}(z_j) P'_r(z_j)$$

where α is a known normalising constant.

The program has been used successfully with `M=20` and `EPS=1.0D-14`, giving zeros and weights to this accuracy.

5 EXAMPLE OF USE

Suppose it is required to calculate many values of

$$e^y Ei(y) = \int_0^\infty \frac{e^{-x}}{y+x} \, dx$$

and `QC02` is going to be used to obtain approximations Q to this. We specify an accuracy in the zeros and weights of 10^{-10}, in an attempt to get this accuracy in the integral value. We shall try a range of quadratures and compare the results, say from order 2 to 7. We choose $a = 1$ to get good results for values of y not too far from unity. The code might go as follows:

```
C      arrays for zeros, weights and f(x)
      DOUBLE PRECISION X(21,21),W(21,21),F(21)
C      y parameter and integral value
      DOUBLE PRECISION Y,Q
C      get quadratures for degrees 3 to 15 (2n-1 to 2m-1)
      N=2
      M=7
C      obtain zeros and weights
      CALL QC02AD(1D-10,1D0,M,N,X,W)
C      read in the y value
   10 READ(5,20,END=70) Y
   20 FORMAT(D20.15)
C      compute and print out the six integral values
      DO 60 K=N,M
        DO 30 J=1,K
          F(J)=1D0/(Y+X(K,J))
   30   CONTINUE
C        evaluate the approximation to the integral
        Q=0D0
        DO 40 J=1,K
          Q=Q+W(K,J)*F(J)
   40   CONTINUE
C        print out the integral value
        WRITE(6,50) K,Q,Y
   50   FORMAT(' DEGREE ',I2,' INTEGRAL VALUE ',D24.16,
     +         ' FOR Y = ',D24.16)
   60 CONTINUE
C      go back and do next integral
      GO TO 10
C      end of data, no more to do
   70 STOP
      END
```