CALCULATION OF GAUSS QUADRATURE RULES

BY

GENE H. GOLUB AND JOHN H. WELSCH

TECHNICAL REPORT NO. CS 81
NOVEMBER 3, 1967

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY
CALCULATION OF GAUSS QUADRATURE RULES*

Gene H. Golub and John H. Welsch

Introduction

Most numerical integration techniques consist of approximating the integrand by a polynomial in a region or regions and then integrating the polynomial exactly. Often a complicated integrand can be factored into a non-negative "weight" function and another function better approximated by a polynomial, thus

\[ \int_a^b g(t)\,dt = \int_a^b \omega(t)f(t)\,dt \approx \sum_{i=1}^{N} w_i f(t_i). \]

Hopefully, the quadrature rule \( \{w_j, t_j\}_{j=1}^N \) corresponding to the weight function \( \omega(t) \) is available in tabulated form, but more likely it is not. We present here two algorithms for generating the Gaussian quadrature rule defined by the weight function when:

a) the three term recurrence relation is known for the orthogonal polynomials generated by \( \omega(t) \), and

b) the moments of the weight function are known or can be calculated.

*The work of the first author was in part supported by the Office of Naval Research and the National Science Foundation; the work of the second author was in part supported by the Atomic Energy Commission.
1. Definitions and Preliminaries

Let $w(x) > 0$ be a fixed weight function defined on $[a, b]$. For $w(x)$, it is possible to define a sequence of polynomials $p_0(x), p_1(x), \ldots$ which are orthonormal with respect to $w(x)$ and in which $p_n(x)$ is of exact degree $n$ so that

$$\int_a^b w(x)p_m(x)p_n(x)dx = \begin{cases} 1 & \text{when } m = n, \\ 0 & \text{when } m \neq n \end{cases} \quad (1.1)$$

The polynomial

$$p_n(x) = k_n \prod_{i=1}^{n} (x-t_i), \quad k_n > 0,$$

has $n$ real roots

$$a < t_1 < t_2 < \ldots < t_n < b.$$

The roots of the orthogonal polynomials play an important role in Gaussian quadrature.

**Theorem:** Let $f(x) \in C^{2N}[a, b]$, then

$$\int_a^b w(x)f(x)dx = \sum_{j=1}^{N} w_j f(t_j) + \frac{f^{(2N)}(\xi)}{(2N)!} k_N^2, \quad (a < \xi < b),$$

where
\[ w_j = -\frac{k_{N+1}}{k_N} \frac{1}{p_{N+1}(t_j)p'_N(t_j)}, \]

\[ \left( p'_N(t_j) = \frac{dp_N(t)}{dt} \right) \bigg|_{t = t_j}, \quad j = 1, 2, \ldots, N. \]

Thus the Gauss quadrature rule is exact for all polynomials of degree \(< 2N-1\). Proofs of the above statements and Theorem can be found in Davis and Rabinowitz [4], chapter 2.

Several algorithms have been proposed for calculating

\[ \{w_j, t_j\}_{j=1}^N; \text{ cf [8], [9].} \]

In this note, we shall give effective numerical algorithms which are based on determining the eigenvalues and the first component of the eigenvectors of a symmetric tri-diagonal matrix.

2. Generating the Gauss Rule

Any set of orthogonal polynomials, \(\{p_j(x)\}_{j=1}^N\), satisfies a three term recurrence relationship:

\[ p_{j+1}(x) = (a_{j+1}x + b_{j+1})p_j(x) - c_{j+1}p_{j-1}(x) \]  

\[ j = 0, 1, 2, \ldots, N-1; \quad p_{-1}(x) \equiv 0, \quad p_0(x) \equiv 1, \]

with

\[ a_j > 0, \quad c_j > 0. \]
The coefficients \( \{a_j, b_j, c_j\} \) have been tabulated for a number of weight functions \( \omega(x) \), cf[6]. In section 4 we shall give a simple method for generating \( \{a_j, b_j, c_j\} \) for any weight function.

Following Wilf [10], we may identify (2.1) with the matrix equation

\[
\begin{pmatrix}
 p_0(x)
 \vdots
 p_{N-1}(x)
\end{pmatrix}
= 
\begin{pmatrix}
 -b_1/a_1, & 1/a_1, & 0 \\
 c_2/a_2, & -b_2/a_2, & 1/a_2 \\
 \vdots & \ddots & \ddots \\
 0 & \ddots & \ddots & 1/a_N \\
 c_N/a_N, & -b_N/a_N & \ddots & \ddots & \ddots & 1/a_N \\
\end{pmatrix}
\begin{pmatrix}
 p_0(x) \\
 \vdots \\
 p_{N-1}(x) \\
 p_N(x)/a_1
\end{pmatrix}
\]

or, equivalently in matrix notation

\[
xp(x) = Tp(x) + \frac{1}{a_N} p_N(x)e_N
\]

where \( T \) is the tri-diagonal matrix and \( e_N = (0, 0, \ldots, 0, 1)^T \). Thus \( p_N(t_j) = 0 \) if and only if

\[
t_j p(t_j) = Tp(t_j)
\]

where \( t_j \) is an eigenvalue of the tri-diagonal matrix \( T \). In [10], it is shown that \( T \) is symmetric if the polynomials are orthonormal. If \( T \) is not symmetric, then we may perform a diagonal similarity
transformation which will yield a symmetric tri-diagonal matrix $J$.

Thus

$$DTD^{-1} = J = \begin{pmatrix}
\alpha_1 & \beta_1 & 0 \\
\beta_1 & \alpha_2 & \beta_2 \\
0 & \cdots & \cdots \\
\cdots & \cdots & \cdots \\
0 & \cdots & \beta_{N-1} \\
\beta_{N-1} & \alpha_N 
\end{pmatrix}$$

where

$$a_i = \frac{b_i}{a_i}, \quad \beta_i = \left( \frac{c_{i+1}}{a_i a_{i+1}} \right) \quad (2.2)$$

It is shown by Wilf [10] that as a consequence of the Christoffel-Darboux identity

$$w_j [\varphi(t_j)]^T [\varphi(t_j)] = 1 \quad j = 1, 2, \ldots, N \quad (2.3)$$

where $\varphi(t_j)$ corresponds to the normalized eigenvector associated with the eigenvalue $t_j$. Suppose that the eigenvectors of $T$ are calculated so that

$$J \varphi_j = t_j \varphi_j \quad j = 1, 2, \ldots, N \quad (2.4)$$
with
\[ a_j^T a_j = 1. \]

Then if
\[ a_j^T = (a_{0j}, a_{1j}, \ldots, a_{N-1,j}), \]
\[ q_j^2 = w_j (p_0(t_j))^2 \]

by (2.3). Thus from (1.1), we see
\[ w_j = \frac{q_j^2}{p_0(t_j)} = \frac{q_j^2}{k_0} = \int_a^b \omega(x) \, dx = a_{0j}^2 \times \mu_0. \]

Consequently, if one can compute the eigenvalues of \( T \) and the first component of the orthonormal eigenvectors, one is able to determine the Gauss quadrature rule.

3. The Q-R algorithm

One of the most effective methods of computing the eigenvalues and eigenvectors of a symmetric matrix is the Q-R algorithm of Francis [5]. The Q-R algorithm proceeds as follows:

Begin with the given matrix \( J = \int_0^j \), compute the factorization
\[ J(0) = Q(0)R(0) \]
where \( Q(0)^T Q(0) = I \) and \( R(0) \) is an upper triangular matrix, and then multiply the matrices in reverse order so that

\[
J(1) = R(0)Q(0) = Q(0)^T J(0)Q(0) .
\]

Now one treats the matrix \( J(1) \) in the same fashion as the matrix \( J(0) \), and a sequence of matrices is obtained by continuing ad infinitum. Thus

\[
J(i) = Q(i)R(i) ,
\]

\[
\vdots J(i+1) = R(i)Q(i) = Q(i+1)R(i+1) \]

so that

\[
J(i+1) = Q(i)^T J(i)Q(i)
= Q(i)^T Q(i-1) T \cdots Q(0)^T JQ(0)Q(1) \cdots Q(i) . \tag{3.2}
\]

Since the eigenvalues of \( J \) are distinct and real for orthogonal polynomials, a real translation parameter \( \lambda \) may be chosen so that the eigenvalues of \( J(1) - \lambda I \) are distinct in modulus. Under these conditions, it is well known [5] that \( J(1) - \lambda I \) converges to the diagonal matrix of eigenvalues of \( J - \lambda I \) as \( i \to \infty \) and that \( P(i) = Q(0) \times Q(1) \times \cdots \times Q(i) \) converges to the orthogonal matrix of eigenvectors of \( J \). The method has the advantage that the matrix \( J(i) - \lambda I \) remains tri-diagonal throughout the computation.

Francis has shown that it is not necessary to compute the decomposition (3.1) explicitly but it is possible to do the calculation (3.2) directly. Let
\[ S^{(i)}_{k,l} = Q^{(i)}_{k,l} \quad (k = 0, 1, \ldots, N-1), \]

(i.e., the elements of the first column of \( S^{(i)} \) are equal to the elements of the first column of \( Q^{(i)} \)). Then if

i) \( K^{(i+1)} = S^{(i)T} J^{(i)} S^{(i)} \),

ii) \( J^{(i)} \) is a tri-diagonal matrix,

iii) \( J^{(i)} \) is non-singular,

iv) the sub-diagonal elements of \( K^{(i+1)} \) are positive,

it follows that \( K^{(i+1)} = K^{(i+1)} \).

For the tri-diagonal matrices, the calculation is quite simple. Dropping the iteration counter \( i \), let

\[
\begin{pmatrix}
1 & \cdots & \\
& \ddots & \cos P & \sin P & \cdots & \\
& & \ddots & \cos P & \sin P & \cdots & \\
& & & \ddots & \ddots & \ddots & \\
& & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & & & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & & & & & & & \ddots & \cos P & \sin P & \cdots \\
& & & & & & & & & & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & & & & & & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & & & & & & & & & & \ddots & \cos P & \sin P & \cdots \\
& & & & & & & & & & & & & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & & & & & & & & & & & & \ddots & \cos P & \sin P & \cdots & \\
& & & & & & & & & & & & & & & & & & \ddots & \cos P & \sin P & \cdots \\
\end{pmatrix}
\]

Then \( \cos \theta_{p} \) is chosen so that

\[ \{Z_{1}, J\}_{k,l} = 0 \quad k = 2, 3, \ldots, \]
Let

\[
J = \begin{bmatrix}
    a_1 & b_1 & 0 & & & \\
    b_1 & a_2 & b_2 & 0 & & \\
    & b_2 & a_3 & b_3 & \ddots & \\
    & & & \ddots & \ddots & 0 \\
    & & & \cdots & b_{N-1} & 0 \\
    & & & & & b_N
\end{bmatrix}
\]

The matrix

\[
Z_1 J Z_1 = \begin{bmatrix}
    a'_1 & b'_1 & d_1 & & & \\
    b'_1 & a'_2 & b'_2 & 0 & & \\
    & b'_2 & a'_3 & b'_3 & \ddots & \\
    & & & \ddots & \ddots & 0 \\
    & & & \cdots & b'_{N-1} & 0 \\
    & & & & & b'_N
\end{bmatrix}
\]

where the primes indicate altered elements of \( J \); then
and $Z_2, \ldots, Z_{N-1}$ are constructed so that $K$ is tri-diagonal. The product of all the orthogonal rotations yields the matrix of orthogonal eigenvectors. To determine $\{w_j\}_{j=1}^N$, however, we need only the first component of the orthonormal eigenvector. Thus, using (2.3)

$$q_j^T = [a_{q0}, a_{q2}, \ldots, a_{qN}] = [1, 0, 0, \ldots, 0] \times \prod_{i=0}^{\infty} (Z_1^{(i)} x Z_2^{(i)} \ldots \times Z_{N-1}^{(i)})$$

and it is not necessary to compute the entire matrix of eigenvectors.

More explicitly, for $j = 1, 2, \ldots, N-1$

$$\sin \theta_j^{(i)} = \frac{d_{j-1}^{(i)}}{[d_{j-1}^{(i)}]^2 + (b_{j-1}^{(i)})^2]^{\frac{1}{2}}}$$

$$\cos \theta_j^{(i)} = \frac{b_{j-1}^{(i)}}{[d_{j-1}^{(i)}]^2 + (b_{j-1}^{(i)})^2]^{\frac{1}{2}}}$$

$$a_j^{(i+1)} = -a_j^{(i)} \cos^2 \theta_j^{(i)} + 2b_j^{(i)} \cos \theta_j^{(i)} \sin \theta_j^{(i)} + a_j^{(i)} \sin^2 \theta_j^{(i)}$$

$$b_j^{(i+1)} = -b_j^{(i)} \cos^2 \theta_j^{(i)} + 2b_j^{(i)} \cos \theta_j^{(i)} \sin \theta_j^{(i)} + a_j^{(i)} \sin^2 \theta_j^{(i)}$$

$$b_{j-1}^{(i+1)} = b_{j-1}^{(i)} \cos \theta_j^{(i)} + d_{j-1}^{(i)} \sin \theta_j^{(i)} = [(b_{j-1}^{(i)})^2 + (d_{j-1}^{(i)})^2]^{\frac{1}{2}}$$

$$b_j^{(i+1)} = (a_j^{(i)} - a_j^{(i+1)} \sin \theta_j^{(i)} \cos \theta_j^{(i)} + b_j^{(i)} \sin \theta_j^{(i)} - \cos \theta_j^{(i)}) j$$

$$b_{j+1}^{(i)} = -b_{j+1}^{(i)} \cos \theta_j^{(i)}$$
\[
\begin{align*}
&d_j^{(i)} = b_j^{(i)} \sin \theta_j^{(i)}, \\
&z_j^{(i+1)} = z_j^{(i)} \cos \theta_j + z_{j+1}^{(i)} \sin \theta_j, \\
&z_j^{(i+1)} = z_j^{(i)} \sin \theta_j - z_{j+1}^{(i)} \cos \theta_j,
\end{align*}
\]

with
\[
\begin{align*}
d_0^{(i)} &= b_1^{(i)}, \quad \nu_0^{(i)} = (a_1^{(i)} - \lambda^{(i)}) .
\end{align*}
\]

Initially
\[
\begin{align*}
z_1^{(0)} &= 1, \quad z_j^{(0)} = 0 \text{ for } j = 2, \ldots, N.
\end{align*}
\]

so that
\[
\begin{align*}
z_i^{(i)} &\rightarrow z_i^{(\infty)} \text{ as } i \rightarrow \infty .
\end{align*}
\]

In the actual computation, no additional storage is required for
\[
\{a_j^{(i)}, b_j^{(i)}, \nu_j^{(i)}, \mu_j^{(i)} \}
\]

since they may overwrite
\[
\{a_j^{(i)}, b_j^{(i)}, z_j^{(i)} \}.
\]

We choose \(\lambda^{(i)}\) as an approximation to an eigenvalue; usually, it is related to the eigenvalues of the matrix.
When $b_N^{(i)}$ is sufficiently small, $a_N^{(i)}$ is taken as eigenvalue and $N$ is replaced by $N-1$.

4. Determining the Three Term Relationship from the Moments

For many weight functions, the three term recursion relationship of the orthogonal polynomials have been determined. In some situations, however, the weight function is not known explicitly but one has a set of $2N+1$ moments, viz.

$$
\mu_k = \int_a^b \omega(x)x^k \, dx \quad k = 0, 1, \ldots, 2N.
$$

Let

$$
M = \begin{bmatrix}
\mu_0, & \mu_1, & \ldots, & \mu_N \\
\mu_1, & \ldots, & \mu_{N+1} \\
\vdots & \ddots & \vdots \\
\mu_N, & \ldots, & \mu_{2N}
\end{bmatrix},
$$

and

$$
D_j = \det \begin{bmatrix}
\mu_0, & \mu_2, & \ldots, & \mu_j \\
\mu_1, & \mu_3, & \ldots, & \mu_{j+1} \\
\vdots & \ddots & \vdots \\
\mu_j, & \mu_{j+2}, & \ldots, & \mu_{2j}
\end{bmatrix}, \quad j = 0, 1, \ldots, N-1.
$$
and

\[
F_j = \det \begin{bmatrix}
\mu_0 & \mu_1 & \cdots & \mu_{j-1} & \mu_j & \mu_{j+1} \\
\mu_1 & \mu_2 & \cdots & \mu_j & \mu_{j+1} & \mu_{j+2} \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
\mu_j & \mu_{j+1} & \cdots & \mu_{2j-1} & \mu_{2j} & \mu_{2j+1}
\end{bmatrix}, \quad j = 1, 2, \ldots, N-1.
\]

It is shown in [1] that

\[
x p_j(x) = \beta_{j-1} p_{j-1}(x) + \alpha_j p_j(x) + \beta_j p_{j+1}(x)
\]

(4.1)

for \(j = 1, 2, \ldots\),

where

\[
a_j = \frac{F_{j-1}}{D_{j-1}} - \frac{F_{j-2}}{D_{j-2}}, \quad -1 = 0, \quad F_0 = \mu_1, \quad j = 1, 2, \ldots, N
\]

\[
\beta_j = \frac{\sqrt{D_{j-2} D_j}}{D_{j-1}}, \quad (D_{-1} = 1, \ D_0 = \mu_0), \quad j = 1, 2, \ldots, N-1.
\]

Note that the tri-diagonal matrix so generated is symmetric.

In [7], Rutishauser gives explicit formulas in terms of determinants for the Gauss-Crout decomposition of a matrix. We may use these relationships to evaluate the coefficients

\[
\{a_j\}_{j=1}^{N}, \quad \{\beta_j\}_{j=1}^{N-1}
\]

Let \(R\) denote the Cholesky decomposition of \(M\) so that
and \( R \) is an upper triangular matrix whose elements are

\[
 r_{ii} = (m_{ii} - \sum_{k=1}^{i-1} r_{ki}) \frac{r_{ii}}{r_{ij}} \]

and

\[
 r_{ij} = (m_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj}) / r_{ii}, \quad i < j, \tag{4.2}
\]

for \( i \) and \( j \) between 1 and \( n \).

Then, from the formulas of Rutishauser

\[
 \alpha_j = \frac{r_{j,j+1}}{r_{j,j}} - \frac{r_{j-1,j}}{r_{j-1,j-1}}, \quad j = 1, 2, \ldots, N \tag{4.3}
\]

\[
 \beta_j = \frac{r_{j+1,j+1}}{r_{j,j}}, \quad j = 1, 2, \ldots, N-1 \}
\]

with \( r_{0,0} = 1, \quad r_{0,1} = 0 \).

There are other means for evaluating \( \{a_j\}, \{\beta_j\} \) but it is the opinion of the authors that the above method will lead to the most accurate formulas.

5. Description of Computational Procedures

In the following section there are three ALGOL 60 procedures for performing the algorithms presented above. We have tried to keep the identifiers as close to the notation of the equations as possible without
sacrificing storage or efficiency. The weights and abscissas of the quadrature rule are the result of the procedure GAUSSQUADRULE which must be supplied with the recurrence relation by either procedure GENORTHOPOLY or procedure CLASSICORTHOPOLY. The former requires the moments of the weight function and the latter the name of the particular orthogonal polynomial. A short description of each procedure follows.

CLASSICORTHOPOLY produces $\mu_0$ and the three term recurrence relationship $(a_j, b_j, c_j)$ for six well-known kinds of orthogonal polynomials:

KIND = 1, Legendre Polynomials $P_n(x)$ on $[-1.0, +1.0]$, $\omega(x) = 1.0$.

KIND = 2, Chebyshev Polynomials of the first kind $T_n(x)$ on $[-1.0, +1.0]$, $\omega(x) = (1-x^2)^{-1/2}$.

KIND = 3, Chebyshev Polynomials of the second kind $U_n(x)$ on $[1.0, +1.0]$, $\omega(x) = (1-x^2)^{+1/2}$.

KIND = 4, Jacobi Polynomials $P_n^{(\alpha, \beta)}(x)$ on $[-1.0, +1.0]$, $\omega(x) = (1-x)^{\alpha}(1+x)^{\beta}$ for $\alpha > -1$ and $\beta > -1$.

KIND = 5, Laguerre Polynomials $L_n^{(\alpha)}(x)$ on $[0, +\infty]$, $\omega(x) = e^{-x}x^{\alpha}$ for $\alpha > -1$.

KIND = 6, Hermite Polynomials $H_n(x)$ on $[-\infty, +\infty]$, $\omega(x) = e^{-x^2}$.

Notice that this procedure requires a real procedure to evaluate the gamma function $\Gamma(x)$. 
GENORTHOPOLY uses the $2N+1$ moments of the weight function which are supplied in $\text{MU}[0]$ through $\text{MU}[2 \otimes N]$ to compute the $\alpha_j'$s and $\beta_j'$s of formula (4.1). First, the Cholesky decomposition (formula 4.2) of the moment matrix is placed in the upper right triangular part of the array $R$, then the formulas (4.3) are used to compute the $\alpha_j'$s and $\beta_j'$s which are placed in the arrays $A$ and $B$ respectively.

GAUSSQUADRULE has two modes of operation controlled by the Boolean parameter $\text{SYMM}$ which indicates whether the tri-diagonal matrix is symmetric or not. When the recursion relation is produced by GENORTHOPOLY, $\text{SYMM}$ is true; when produced by CLASSICORTHOPOLY, $\text{SYMM}$ is false. If $\text{SYMM}$ is false, the matrix is symmetricized using the formulas (2.2). The diagonal elements $\alpha_1$ are stored in $A[I]$ and the off diagonal elements $\beta_1$ are stored in $B[I]$.

Beginning at label SETUP, several calculations and initializations are done: the $l_1$ norm of the tri-diagonal matrix and the relative zero tolerance are computed; the first component of each eigenvector $W[I]$ and the Q-R iteration are initialized. $\text{LAMBDA}$ is a variable subtracted off the diagonal elements to accelerate convergence of the Q-R iteration and control to some extent in what order the eigenvalues (abscissas) are found. It begins with a value outside and to the right of the interval containing the abscissas ($=\text{NORM}$) and moves to the left as the abscissas are found; thus the abscissas will be in ascending order in the array $T$ (just to be sure an exchange sort is used at label $\text{SORT}$).

The maximum (EIGMAX) of the eigenvalues ($\text{LAMBDA1}$ and $\text{LAMBDA2}$) of the lower $2 \times 2$ submatrix is compared to the maximum (RHO) from the last iteration. If they are close, $\text{LAMBDA}$ is replaced by $\text{EIGMAX}$. 

16
This scheme seems to stabilize LAMBDA and speed convergence immediately after deflation.

An eigenvalue has been found when the last off diagonal element falls below $\text{EPS}$ (see section 7). Its value is placed in $T[I]$ and the corresponding weight $W[I]$ is computed from formula (2.5). This convergence test and the test for matrix splitting are done following label INSPECT. Only the Lower block (from $K$ to $M$) needs to be transformed by the $Q$-$R$ equation given in formulas (3.3).
6. The ALGOL 60 Procedures
procedure classicorthopoly(kind, alfa, beta, n, a, b, c, muzero);
    value kind, n, alfa, beta;
    integer kind, n;  real alfa, beta, muzero;
    real array a, b, c;

begin comment This procedure supplies the coefficients of the three term
    recurrence relationship for various classical orthogonal polynomials.;
    integer i;  real abl, pi;
    switch swt := legendre, chebyshevl, chebyshev2, jacobi,
        laguerre, hermite;
    pi := 3.14159265359;
    go to swt[kind];
legendre: muzero := 2.0;
    comment P(x) in [-1, 1], w(x) = 1.0 ;
    for i:=1 step 1 until n do
        begin a[i] := (2*i-1)/i; b[i]:=0; c[i]:=(i-1)/i end;
    go to return;
chebyshevl: muzero := pi;
    comment T (x) in [-1, 1], w(x) = (1-x^2)^(-.5);
    for i:=1 step 1 until n do
        begin a[i] := 2; b[i]:=0; c[i]:=1 end;
    a[i] := 1;  go to return;
chebyshev2: muzero := pi/2.0;
    comment u(x) in [-1, 1], W(x) = (1-x^2)^.5;
    for i:=1 step 1 until n do
        begin a[i] := 2; b[i]:=0; c[i]:=1 end;
    go to return;
jacobi:  \[ m\text{uzero} := 2^\gamma (\alpha + \beta + 1) x \gamma(\alpha+1) x \gamma(\beta+1) / \gamma(\alpha+\beta+2) ; \]

\[ \text{comment } P(\alpha, \beta)(x) \text{ in } [-1, 1], w(x) = (1-x)^\gamma(\alpha+1) / \gamma(\alpha+\beta+2) \]

\[ \alpha > -1 \text{ and } \beta > -1 ; \]

\[ a[1] := 0.5 x (\alpha + \beta + 2) ; b[1] := 0.5 x (\alpha - \beta) ; \]

\[ \text{for } i := 2 \text{ step 1 until } n \text{ do} \]

\[ \text{begin } \]

\[ a[i] := (2 i + \alpha + \beta + 1) x (2 i + \alpha + \beta) / \gamma(2 i + \alpha + \beta) ; \]

\[ b[i] := (2 i + \alpha + \beta - 1) x (\alpha + 2 - \beta + 2) / \gamma(2 i + \alpha + \beta) ; \]

\[ c[i] := 2 x (i + \alpha) x (i + \beta) x (2 i + \alpha + \beta) / \gamma(2 i + \alpha + \beta) ; \]

\[ \text{end} ; \]

\[ \text{go to return } ; \]

laguerre:  \[ m\text{uzero} := \gamma(\alpha + 1) ; \]

\[ \text{comment } L(\alpha)(x) \text{ in } [0, \infty) , w(x) = \exp(-x)^\gamma(\alpha+1) , \alpha > -1 ; \]

\[ \text{for } i := 1 \text{ step 1 until } n \text{ do} \]

\[ \text{begin } \]

\[ a[i] := -1 / i ; b[i] := (2 i - 1 + \alpha) / i ; \]

\[ c[i] := (i - 1 + \alpha) / i ; \]

\[ \text{end} ; \]

\[ \text{go to return } ; \]

hermite:  \[ m\text{uzero} := \sqrt{\pi} ; \]

\[ \text{comment } H(x) \text{ in } (-\infty, +\infty), w(x) = \exp(-x^2) ; \]

\[ \text{for } i := 1 \text{ step 1 until } n \text{ do} \]

\[ \text{begin } a[i] := 2 ; b[i] := 0 ; c[i] := 2 x (i - 1) \text{ end ;} \]

return:  \[ \text{end classicorthopoly } ; \]
procedure genorthopoly(n, mu, a, b);
  value n; integer n;
  real array mu, a, b;
begin comment Given the 2n+1 moments of the weight function,
  generate the recursion coefficients of the orthogonal
  polynomials. ;
  real array r[0:n+1,0:n+1]; real sum;
  integer i, j, k;
comment Place the Cholesky decomposition of the moment matrix in r[];
  for i:=1 step 1 until n+1 do
    for j:=i step 1 until n+1 do
      begin
        sum:= mu[i+j-2];
        for k:=i-1 step -1 until 1 do
          sum:= sum*r[k,i]*r[k,j];
        r[i,j]:= (if i=j e n sqrt(sum) else sum/r[i,i]);
      end;
comment Compute the recursion coefficients'from the decomposition r[];
  r[0,0] := 1.0; r[0,1] := 0;
  for i:=0 step 1 until n+1 do
    a[i] := r[n,n+i]/r[n,n]-r[n-1,n]/r[n-1,n-1];
  for j:=n-1 step -1 until 1 do
    begin
      b[j] := r[j+1,j+1]/r[j,j];
      a[j] := r[j,j+1]/r[j,j]-r[j-1,j]/r[j-1,j-1];
    end;
end genorthopoly;
procedure gaussquadrule(n, a, b, c, muzero, symm, t, w);

value n, muzero, symm;

integer real muzero; boolean symm;

real array a, b, c, t, w;

begin m e n t  Given the coefficients a, b, c of the three term recurrence

relation : p[k+l] = (a[k+l]x+b[k+l])p[k]-c[k+l]p[k+1], this procedure

computes the abscissas t and the weights w of the gaussian type

quadrature rule associated with the orthogonal polynomial by QR

type iteration with origin shifting;

integer i, j, k, m, ml;

real norm, eps, ct, st, ct2, st2, sc, aa, ai, aj, a2, eigmax, lambda,

lambda1, lambda2, rho, r, det, bi, bj, b2, wj, cj;

boolean ex;

real procedure max(x,y); value x, y; real x, y;

max := if x > y then x else y;

if symm then go to setup;

comment Symmetrize the matrix, if required.;

for i:=1 step 1 until n-1 do

begin ai := a[i]; a[i] := -b[i]/ai;

b[i] := sqrt(c[i+1]/(a[i]*a[i+1]));

end;

a[n] := -b[n]/a[n];

comment Find the maximum row sum norm and initialize w[1];

setup : b[0] := 0; norm := 0;

for i:=1 step 1 until n-1 do

begin norm := max(norm, abs(ai) + abs(b[i-1]) + abs(b[i-1]));

w[i] := 0;
\text{end;}

\text{norm} := \max(\text{norm}, \text{abs(a[n]}+\text{abs(b[n-l])))};

w[1] := 1.0; w[n] := 0; m := n;

\text{eps} := \text{norm}\times1.0 \times 10^{13}; \quad \text{comment} \quad \text{Relative zero tolerance:}

\lambda := \lambda_1 := \lambda_2 := \rho := \text{norm};

\text{comment} \quad \text{Look for convergence of lower diagonal element;}

\text{inspect: } \text{if } m=0 \text{ then go to sort else } i := k := m1 := m-1;

\text{if } \text{abs(b[m1] < eps then}

\begin{align*}
\text{begin } t[m] &:= a[m]; w[m] := \text{mu}	ext{zero}\times w[m] \times 2; \\
\rho &:= (\text{if } \lambda_1 < \lambda_2 \text{ then } \lambda_1 \text{ else } \lambda_2); \\
\text{m} &:= m1; \quad \text{go to inspect;}
\end{align*}

\text{end;

comment} \quad \text{Small of diagonal element means matrix can be split;}

\text{for } i:=i-1 \text{ while } \text{abs(b[i] > eps do } k := i;

\text{comment} \quad \text{Find eigenvalues of lower 2x2 and select accelerating shift;}

b2 := b[m1] \times 2; \quad \text{det} := \sqrt{(a[m]\times a[m] \times 2+4.0\times b2)}; \\
\text{aa} := a[m1]+a[m];

\lambda_2 := 0.5 \times (\text{if } \text{aa > 0 then } \text{aa +det else } \text{aa-det});

\lambda_1 := (a[m1]\times a[m]-b2)/\lambda_2;

\text{eigmax} := \max(\lambda_1, \lambda_2);

\text{if } \text{abs(eigmax - rho) < 0.125\times abs(eigmax) then } \lambda := \rho := \text{eigmax}

\text{else } \rho := \text{eigmax};

\text{comment} \quad \text{Transform block from k to m;}

cj := b[k]; b[k-1] := a[k]-\lambda;

\text{for } j:=k \text{ step 1 until } ml \text{ do
\begin{verbatim}
begin r := sqrt(c_{j+2}^2+b_{[j-1]}^2) ;
st := c_{j}/r; st2 := st^2;
ct := b_{[j-1]}/r; ct2 := ct^2;
scc := st*ct; aj := a_{[j]};
b_{[j]} := b_{[j]}; w_{[j]} := w_{[j]};
a_{[j]} := a_{[j]}*ct+2.0*b_{[j])*sc+b_{[j+1]}*st2;
b_{[j]} := (a_{[j]-a_{[j+1])})*sc+b_{[j]}*st*(st2-ct2);
a_{[j+1]} := a_{[j]}*st^2-2.0*b_{[j]}*sc+a_{[j+1]}*ct^2;
c_{j} := b_{[j+1]}*st; b_{[j+1]} := -b_{[j+1]}*ct; b_{[j-1]} := r
w_{[j]} := w_{[j]}*ct+w_{[j+1]}*st; w_{[j+1]} := w_{[j]}*st-w_{[j+1]}*ct;
end;

b_{[k-1]} := 0 go to inspect;

comment Arrange abscissas in ascending order;

sort: for m:=n step -1 until 2 do

begin ex := false;

    for i:=2 step 1 until m do

        if t_{[i-1]} > t_{[i]} then

            begin r := t_{[i-1]}; t_{[i-1]} := t_{[i]};
                      t_{[i]} := r; r := w_{[i-1]};
                      w_{[i-1]} := w_{[i]}; w_{[i]} := r;

            ex := true;

        end;

        if ~ ex then go to return ;

    end;

return : end gaussquadrule;
\end{verbatim}
7. Test Program and Results

The procedures in section 6 have been extensively tested in Burroughs B5500 Algol and IBM OS/360 Algol. There are two machine dependent items which must be mentioned. First, the constant used to define the "relative zero tolerance" EPS in procedure GAUSSQUADRULE is dependent on the length of the fraction part of the floating-point number representation (\(= 8^{-13}\) for the 13 octal digit fraction on the B5500, and \(= 16^{-14}\) for a 14 hexadecimal digit long-precision fraction on the IBM 360). Second, the moment matrix \(M\) defined in section 4 usually becomes increasingly ill conditioned with increasing \(N\). Thus the round-off errors generated during Cholesky decomposition in GENORTHOPOLY cause an ill conditioned \(M\) to appear no longer positive definite and the procedure fails on taking the square root of a negative number.

The procedure GAUSSQUADRULE proves to be quite stable and when the recursion coefficients are known or supplied by the procedure CLASSICORTHOPOLY it loses only several digits off of full-word accuracy even for \(N = 50\). Procedure GENORTHOPOLY usually failed to produce the recursion coefficients from the moments when \(N\) was about 20 for the IBM 360.

The test program given below is designed to compare the two methods of generating the quadrature rules--from the moments or the recursion coefficients. \(N\) can be increased until GENORTHOPOLY fails. Numerical results may be checked against tables for Gauss-Legendre quadrature in [9] and Gauss-Laguerre quadrature in [2]. In the Table, we compare the abscissas and weights of the Gauss-Laguerre quadrature rule with \(\alpha = -0.75\) and \(N = 10\) computed by (1) the analytic recurrence
relationship and the Q-R algorithm; (2) the moment matrix and the Q-R algorithm; (3) Concus et. al. [2]. The calculations for (1) and (2) were performed on the IBM 360.
begin

comment Driver program for gaussrule;

real array a, b, c, mu, t, w[0:10];
real muzero; integer i, n;

n := 10;

comment Legendre polynomials.

outstring(1, 'legendre quadrature.');
classicorthopoly(1, 0, 0, n, a, b, c, muzero);
gaussquadrule(n, a, b, c, muzero, false, t, w);
.
outstring(1, 'abscissas:'); outarray(1);
outstring(1, 'weights:'); outarray(1, w);

for i:=0 step 1 until 2*n do mu[i] := 0;
for i:=0 step 2 until 2*n do mu[i] := 2.0/(i+1);

genorthopoly(n, mu, a, b);
muzero := mu[0];

 gaussquadrule(n, a, b, c, muzero, true, t, w);
outstring(1, 'abscissas:'); outarray(1, t);
outstring(1, 'weights:'); outarray(1, w);

comment Laguerre polynomials.

outstring(1, 'laguerre quadrature. alpha =-0.5');
classicorthopoly(5, -0.5, 0, n, a, b, c, muzero);
gaussquadrule(n, a, b, c, muzero, false, t, w);
outstring(1, 'abscissas:'); outarray(1, t);
outstring(1, 'weights:'); outarray(1, w);
mu[0] := muzero := 1.7724538509; comment gamma (0.5);

for i:=1 step 1 until 2*n do
mu[i] := (i-0.5)*mu[i-1];
genorthopoly(n, mu, a, b);
gaussquadrule(n, a, b, c, muzero, true, t, w);
outstring(1, ' abscissas:'); outarray(1, t);
outstring(1, ' weights:'); outarray(1, w);
end;


Table

A Comparison of the Abscissas and Weights of the Gauss-Laguerre Quadrature Rule

with $\alpha = -0.75$ and $N = 10$

<table>
<thead>
<tr>
<th>Analytic Recurrence Relationship + QR</th>
<th>Moment Matrix + QR</th>
<th>Concus et al. [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ABSCISSAS</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.766655867080153</td>
<td>2.76665586784701</td>
<td>2.7666558707972</td>
</tr>
<tr>
<td>2</td>
<td>4.54784422605942</td>
<td>4.54784422605949</td>
<td>4.54784422605949</td>
</tr>
<tr>
<td>3</td>
<td>1.392425761158619</td>
<td>1.38245761158599</td>
<td>1.38245761158599</td>
</tr>
<tr>
<td>4</td>
<td>2.839568012092734</td>
<td>2.839568012092734</td>
<td>2.839568012092734</td>
</tr>
<tr>
<td>5</td>
<td>4.8597148754628</td>
<td>4.8597148754628</td>
<td>4.8597148754628</td>
</tr>
<tr>
<td>6</td>
<td>7.50011998241283</td>
<td>7.50011998241283</td>
<td>7.50011998241283</td>
</tr>
<tr>
<td>7</td>
<td>1.08890623823446</td>
<td>1.08890623823446</td>
<td>1.08890623823446</td>
</tr>
<tr>
<td>8</td>
<td>1.51997806437974</td>
<td>1.51997806437974</td>
<td>1.51997806437974</td>
</tr>
<tr>
<td>9</td>
<td>2.07892146210701</td>
<td>2.07892146210701</td>
<td>2.07892146210701</td>
</tr>
<tr>
<td>10</td>
<td>2.85730601649221</td>
<td>2.85730601649221</td>
<td>2.85730601649221</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>WEIGHTS</strong></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.566755557790853</td>
<td>2.56675555728235</td>
</tr>
<tr>
<td>2</td>
<td>7.73347970340221</td>
<td>7.73347970340221</td>
</tr>
<tr>
<td>3</td>
<td>2.35132839732162</td>
<td>2.35132839732162</td>
</tr>
<tr>
<td>4</td>
<td>4.64374708956672</td>
<td>4.64374708956672</td>
</tr>
<tr>
<td>5</td>
<td>5.54912353032852</td>
<td>5.54912353032852</td>
</tr>
<tr>
<td>6</td>
<td>3.65466266776141</td>
<td>3.65466266776141</td>
</tr>
<tr>
<td>7</td>
<td>1.18687965702585</td>
<td>1.18687965702585</td>
</tr>
<tr>
<td>8</td>
<td>1.58441095835014</td>
<td>1.58441095835014</td>
</tr>
<tr>
<td>9</td>
<td>6.19326672679582</td>
<td>6.19326672679582</td>
</tr>
<tr>
<td>10</td>
<td>3.07775996368175</td>
<td>3.07775996368175</td>
</tr>
</tbody>
</table>

(Underlined figures are those which disagree with Concus et al. [2].)
8. References


[2] P. Concus, D. Cassatt, G. Jaehnig and E. Melky, "Tables for the evaluation of
\[ \int_0^\infty x^\beta e^{-x} f(x) dx \]


