



Calculation of Gauss Quadrature Rules

Author(s): Gene H. Golub and John H. Welsch

Source: *Mathematics of Computation*, Vol. 23, No. 106 (Apr., 1969), pp. 221-230+s1-s10

Published by: American Mathematical Society

Stable URL: <http://www.jstor.org/stable/2004418>

Accessed: 29/05/2010 10:26

Your use of the JSTOR archive indicates your acceptance of JSTOR's Terms and Conditions of Use, available at <http://www.jstor.org/page/info/about/policies/terms.jsp>. JSTOR's Terms and Conditions of Use provides, in part, that unless you have obtained prior permission, you may not download an entire issue of a journal or multiple copies of articles, and you may use content in the JSTOR archive only for your personal, non-commercial use.

Please contact the publisher regarding any further use of this work. Publisher contact information may be obtained at <http://www.jstor.org/action/showPublisher?publisherCode=ams>.

Each copy of any part of a JSTOR transmission must contain the same copyright notice that appears on the screen or printed page of such transmission.

JSTOR is a not-for-profit service that helps scholars, researchers, and students discover, use, and build upon a wide range of content in a trusted digital archive. We use information technology and tools to increase productivity and facilitate new forms of scholarship. For more information about JSTOR, please contact support@jstor.org.



American Mathematical Society is collaborating with JSTOR to digitize, preserve and extend access to *Mathematics of Computation*.

<http://www.jstor.org>

Calculation of Gauss Quadrature Rules*

By Gene H. Golub** and John H. Welsch

Abstract. Several algorithms are given and compared for computing Gauss quadrature rules. It is shown that given the three term recurrence relation for the orthogonal polynomials generated by the weight function, the quadrature rule may be generated by computing the eigenvalues and first component of the orthonormalized eigenvectors of a symmetric tridiagonal matrix. An algorithm is also presented for computing the three term recurrence relation from the moments of the weight function. ■

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

$$\int_a^b g(t)dt = \int_a^b \omega(t)f(t)dt \approx \sum_{j=1}^N w_j f(t_j).$$

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.

In [6], Gautschi presents an algorithm for calculating Gauss quadrature rules when neither the recurrence relationship nor the moments are known.

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

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

The polynomial $p_n(x) = k_n \prod_{i=1}^n (x - t_i)$, $k_n > 0$, has n real roots $a < t_1 < t_2 < \dots$

Received November 13, 1967, revised July 12, 1968.

* 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.

** Present address: Hewlett-Packard Company, Palo Alto, California 94304.

$\dots < t_n < b$. The roots of the orthogonal polynomials play an important role in Gaussian quadrature.

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

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

where

$$w_j = -\frac{k_{N+1}}{k_N} \frac{1}{p_{N+1}(t_j) p_N'(t_j)}, \quad \left(p_N'(t_j) = \left. \frac{dp_N(t)}{dt} \right|_{t=t_j} \right), \quad j = 1, 2, \dots, N.$$

Thus the Gauss quadrature rule is exact for all polynomials of degree $\leq 2N - 1$.

Proofs of the above statements and Theorem can be found in Davis and Rabinowitz [4, Chapter 2].

Several algorithms have been proposed for calculating $\{w_j, t_j\}_{j=1}^N$; cf. [10], [11]. 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 tridiagonal matrix.

2. Generating the Gauss Rule. Any set of orthogonal polynomials, $\{p_j(x)\}_{j=1}^N$, satisfies a three term recurrence relationship:

$$(2.1) \quad p_j(x) = (a_j x + b_j) p_{j-1}(x) - c_j p_{j-2}(x), \\ j = 1, 2, \dots, N; \quad p_{-1}(x) \equiv 0, \quad p_0(x) \equiv 1,$$

with $a_j > 0, c_j > 0$. The coefficients $\{a_j, b_j, c_j\}$ have been tabulated for a number of weight functions $\omega(x)$, cf. [8]. In Section 4 we shall give a simple method for generating $\{a_j, b_j, c_j\}$ for any weight function.

Following Wilf [12], we may identify (2.1) with the matrix equation

$$x \begin{bmatrix} p_0(x) \\ p_1(x) \\ \cdot \\ \cdot \\ p_{N-1}(x) \end{bmatrix} = \begin{bmatrix} -b_1/a_1 & 1/a_1 & 0 & & \\ c_2/a_2 & -b_2/a_2 & 1/a_2 & & \\ 0 & \cdot & \cdot & \cdot & \\ & & \cdot & \cdot & \cdot \\ \bigcirc & & & \cdot & \cdot & 1/a_{N-1} \\ & c_N/a_N & -b_N/a_N & & \end{bmatrix} \begin{bmatrix} p_0(x) \\ p_1(x) \\ \cdot \\ \cdot \\ p_{N-1}(x) \end{bmatrix} \\ + \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ 0 \\ p_N(x)/a_N \end{bmatrix}$$

or, equivalently in matrix notation

$$x\mathbf{p}(x) = T\mathbf{p}(x) + (1/a_N)p_N(x)\mathbf{e}_N$$

where T is the tridiagonal matrix and $\mathbf{e}_N = (0, 0, \dots, 0, 1)^T$. Thus $p_N(t_j) = 0$ if and only if $t_j\mathbf{p}(t_j) = T\mathbf{p}(t_j)$ where t_j is an eigenvalue of the tridiagonal matrix T . In [12], 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 tridiagonal matrix J . Thus

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

where

$$(2.2) \quad \alpha_i = -\frac{b_i}{a_i}, \quad \beta_i = \left(\frac{c_{i+1}}{a_i a_{i+1}}\right)^{1/2}.$$

It is shown by Wilf [12] that as a consequence of the Christoffel-Darboux identity

$$(2.3) \quad w_j[\mathbf{p}(t_j)]^T[\mathbf{p}(t_j)] = 1, \quad j = 1, 2, \dots, N$$

where $\mathbf{p}(t_j)$ corresponds to the eigenvector associated with the eigenvalue t_j . Suppose that the eigenvectors of T are calculated so that

$$(2.4) \quad J\mathbf{q}_j = t_j\mathbf{q}_j, \quad j = 1, 2, \dots, N$$

with $\mathbf{q}_j^T \mathbf{q}_j = 1$. If

$$(2.5) \quad \mathbf{q}_j^T = (q_{1,j}, q_{2,j}, \dots, q_{N,j}),$$

then $q_{1,j}^2 = w_j(p_0(t_j))^2$ by (2.3). Thus from (1.1), we see

$$(2.6) \quad w_j = \frac{q_{1,j}^2}{p_0^2(t_j)} = \frac{q_{1,j}^2}{k_0^2} = q_{1,j}^2 \times \int_a^b \omega(x) dx \equiv q_{1,j}^2 \times \mu_0.$$

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 $J = J^{(0)}$, 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

$$(3.1) \quad \begin{aligned} J^{(i)} &= Q^{(i)}R^{(i)}, \\ J^{(i+1)} &= R^{(i)}Q^{(i)} = Q^{(i+1)}R^{(i+1)} \end{aligned}$$

so that

$$(3.2) \quad J^{(i+1)} = Q^{(i)T}J^{(i)}Q^{(i)} = Q^{(i)T}Q^{(i-1)T} \dots Q^{(0)T}JQ^{(0)}Q^{(1)} \dots Q^{(i)}.$$

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

$$\{S^{(i)}\}_{k,1} = \{Q^{(i)}\}_{k,1} \quad (k = 1, 2, \dots, N),$$

(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) $K^{(i+1)}$ is a tridiagonal matrix,
- (iii) $J^{(i)}$ is nonsingular,
- (iv) the subdiagonal elements of $K^{(i+1)}$ are positive, it follows that $K^{(i+1)} = J^{(i+1)}$.

For the tridiagonal matrices, the calculation is quite simple. Dropping the iteration counter i , let

$$Z_p = \begin{bmatrix} & & & & (p) & (p+1) \\ & & & & & \\ 1 & & & & \cdot & \cdot \\ & \cdot & & & \cdot & \cdot \\ & & \cdot & & \cdot & \cdot \\ & & & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cos \theta_p & \sin \theta_p & \cdot & \cdot & \cdot & \cdot & (p) \\ \cdot & \cdot & \cdot & \cdot & \sin \theta_p & -\cos \theta_p & \cdot & \cdot & \cdot & \cdot & (p+1) \\ & & & & \cdot & \cdot & 1 & & & & \\ \bigcirc & & & & \cdot & \cdot & & \cdot & & & \\ & & & & \cdot & \cdot & & & & & \\ & & & & \cdot & \cdot & & & & & 1 \end{bmatrix}.$$

Then $\cos \theta_1$ is chosen so that

$$\{Z_1 J\}_{k,1} = 0, \quad k = 2, 3, \dots, N.$$

Let

$$J = \begin{bmatrix} a_1 & b_1 & 0 & & & \\ b_1 & a_2 & b_2 & & \bigcirc & \\ 0 & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ \bigcirc & & & \cdot & \cdot & b_{N-1} \\ & & & & b_{N-1} & a_N \end{bmatrix}.$$

The matrix

$$Z_1 J Z_1 = \begin{bmatrix} a_1' & b_1' & d_1 & & & \\ b_1' & a_2' & b_2' & 0 & & \bigcirc \\ d_1 & b_2' & a_3 & b_3 & & \\ & 0 & b_3 & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ \bigcirc & & & & \cdot & \cdot & b_{N-1} \\ & & & & & b_{N-1} & a_N \end{bmatrix},$$

where the primes indicate altered elements of J ; then

$$K = Z_{N-1} Z_{N-2} \cdots Z_1 J Z_1 \cdots Z_{N-1}$$

and Z_2, \dots, Z_{N-1} are constructed so that K is tridiagonal. 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)

$$\mathbf{q}^T \equiv [q_{1,1}, q_{1,2}, \dots, q_{1,N}] = [1, 0, 0, \dots, 0] \times \prod_{i=0}^{\infty} (Z_1^{(i)} \times Z_2^{(i)} \times \cdots \times Z_{N-1}^{(i)})$$

and it is not necessary to compute the entire matrix of eigenvectors. More explicitly, for $j = 1, 2, \dots, N-1$

$$\begin{aligned} \sin \theta_j^{(i)} &= d_{j-1}^{(i)} / [(d_{j-1}^{(i)})^2 + (\bar{b}_{j-1}^{(i)})^2]^{1/2}, \\ \cos \theta_j^{(i)} &= \bar{b}_{j-1}^{(i)} / [(d_{j-1}^{(i)})^2 + (\bar{b}_{j-1}^{(i)})^2]^{1/2}, \\ a_j^{(i+1)} &= \bar{a}_j^{(i)} \cos^2 \theta_j^{(i)} + 2\bar{b}_j^{(i)} \cos \theta_j^{(i)} \sin \theta_j^{(i)} + a_{j+1}^{(i)} \sin^2 \theta_j^{(i)}, \\ \bar{a}_{j+1}^{(i)} &= \bar{a}_j^{(i)} \sin^2 \theta_j^{(i)} - 2\bar{b}_j^{(i)} \cos \theta_j^{(i)} \sin \theta_j^{(i)} + a_{j+1}^{(i)} \cos^2 \theta_j^{(i)}, \\ (3.3) \quad b_{j-1}^{(i+1)} &= \bar{b}_{j-1}^{(i)} \cos \theta_j^{(i)} + d_{j-1}^{(i)} \sin \theta_j^{(i)} = [(\bar{b}_{j-1}^{(i)})^2 + (d_{j-1}^{(i)})^2]^{1/2}, \\ \bar{b}_j^{(i)} &= (\bar{a}_j^{(i)} - a_{j+1}^{(i)}) \sin \theta_j^{(i)} \cos \theta_j^{(i)} + \bar{b}_j^{(i)} (\sin^2 \theta_j^{(i)} - \cos^2 \theta_j^{(i)}), \\ \bar{b}_{j+1}^{(i)} &= -b_{j+1}^{(i)} \cos \theta_j^{(i)}, \\ d_j^{(i)} &= b_{j+1}^{(i)} \sin \theta_j^{(i)}, \quad z_j^{(i+1)} = \bar{z}_j^{(i)} \cos \theta_j + z_{j+1}^{(i)} \sin \theta_j, \\ \bar{z}_{j+1}^{(i)} &= \bar{z}_j^{(i)} \sin \theta_j^{(i)} - z_{j+1}^{(i)} \cos \theta_j^{(i)}, \end{aligned}$$

with

$$\begin{aligned} d_0^{(i)} &= b_1^{(i)}, \quad \bar{b}_0^{(i)} = (a_1^{(i)} - \lambda^{(i)}), \\ \bar{a}_1^{(i)} &= a_1^{(i)}, \quad \bar{b}_1^{(i)} = b_1^{(i)}, \quad \bar{z}_1^{(i)} = z_1^{(i)}. \end{aligned}$$

Initially

$$z_1^{(0)} = 1, \quad z_j^{(0)} = 0 \quad \text{for } j = 2, \dots, N$$

so that $\mathbf{z}^{(i)T} \rightarrow \mathbf{q}^T$ as $i \rightarrow \infty$. In the actual computation, no additional storage is required for $\{\bar{a}_j^{(i)}, \bar{b}_j^{(i)}, \bar{c}_j^{(i)}, \bar{z}_j^{(i)}\}$ 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

$$\begin{bmatrix} a_{N-1}^{(i)} & b_{N-1}^{(i)} \\ b_{N-1}^{(i)} & a_N^{(i)} \end{bmatrix}.$$

When $b_{N-1}^{(i)}$ 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 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.

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

Based on an elegant paper of Mysovskih [9], Gautschi*** has given a simple derivation of the three term relationship which we give below. This result also follows from certain determinantal relations (cf. [7]).

Let $\Omega \subset E^n$ be a domain in n -dimensional Euclidean space and $\omega(\mathbf{x}) \geq 0$ be a weight function on Ω for which all "moments"

$$\mu_{\gamma_1, \gamma_2, \dots, \gamma_n} = \int_{\Omega} \omega(\mathbf{x}) x_1^{\gamma_1} x_2^{\gamma_2} \dots x_n^{\gamma_n} d\mathbf{x}$$

exist, and $\mu_{0,0,\dots,0} > 0$. Enumerate the monomials

$$x_1^{\gamma_1} x_2^{\gamma_2} \dots x_n^{\gamma_n}, \quad \gamma_1 \geq 0, \dots, \gamma_n \geq 0$$

as $\{\varphi_i(\mathbf{x})\}_{i=1}^L$, whereby $i < j$ if degree $\varphi_i <$ degree φ_j , the enumeration within the same degree being arbitrary. In particular, $\varphi_1(\mathbf{x}) = 1$. Let

$$M = [(\varphi_i, \varphi_j)]_{i,j=1}^L \equiv \{m_{ij}\}$$

denote the "Gram matrix" for the system $\{\varphi_i(\mathbf{x})\}_{i=1}^L$, where

$$(\varphi_i, \varphi_j) = \int_{\Omega} \omega(\mathbf{x}) \varphi_i(\mathbf{x}) \varphi_j(\mathbf{x}) d\mathbf{x}.$$

Note M is positive definite. Let $M = R^T R$ be the Cholesky decomposition of M with

$$r_{ii} = \left(m_{ii} - \sum_{k=1}^{i-1} r_{ki}^2 \right)^{1/2}$$

and

$$(4.1) \quad r_{ij} = \left(m_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj} \right) / r_{ii}, \quad i < j$$

*** Personal communication.

for i and j between 1 and L . Let

$$R^{-1} \equiv \begin{bmatrix} s_{11} & s_{12} & \cdot & \cdot & \cdot & s_{1L} \\ & s_{22} & \cdot & \cdot & \cdot & s_{2L} \\ & & \cdot & & & \cdot \\ & & & \cdot & & \cdot \\ \bigcirc & & & & \cdot & \cdot \\ & & & & & s_{LL} \end{bmatrix}.$$

THEOREM (MYSOVSKIY). *The polynomials*

$$F_j(\mathbf{x}) = s_{1j}\varphi_1(\mathbf{x}) + s_{2j}\varphi_2(\mathbf{x}) + \cdots + s_{jj}\varphi_j(\mathbf{x}) \quad (j = 1, 2, \dots, L)$$

form an orthonormal system.

Now in the special case $n = 1$, one has $\varphi_j(x) = x^{j-1}$ with $L = N + 1$, and M is just the "Hankel" matrix in the moments. Moreover, we know in this case $F_j(x) = p_{j-1}(x)$, a polynomial of degree $j - 1$, and $\{p_j(x)\}_{j=0}^N$ satisfy

$$(4.2) \quad xp_{j-1}(x) = \beta_{j-1}p_{j-2}(x) + \alpha_j p_{j-1}(x) + \beta_j p_j(x), \quad j = 1, \dots, N,$$

where $p_{-1}(x) \equiv 0$. Comparing the coefficients of x^j and x^{j-1} on either side of this identity, one gets

$$s_{jj} = \beta_j s_{j+1, j+1}$$

$$s_{j-1, j} = \alpha_j s_{jj} + \beta_j s_{j, j+1}$$

and so

$$\beta_j = \frac{s_{j, j}}{s_{j+1, j+1}}, \quad \alpha_j = \frac{s_{j-1, j}}{s_{j, j}} - \frac{s_{j, j+1}}{s_{j+1, j+1}}.$$

Further, if

$$R = \begin{bmatrix} r_{11} & r_{12} & \cdot & \cdot & \cdot & r_{1, N+1} \\ & r_{22} & \cdot & \cdot & \cdot & r_{2, N+1} \\ & & \cdot & & & \cdot \\ & & & \cdot & & \cdot \\ \bigcirc & & & & \cdot & \cdot \\ & & & & & r_{N+1, N+1} \end{bmatrix}$$

a straightforward computation shows that

$$s_{jj} = \frac{1}{r_{jj}}, \quad s_{j, j+1} = \frac{-r_{j, j+1}}{r_{j, j} r_{j+1, j+1}}.$$

Thus

$$(4.3) \quad \alpha_j = \frac{r_{j, j+1}}{r_{j, j}} - \frac{r_{j-1, j}}{r_{j-1, j-1}}, \quad j = 1, 2, \dots, N,$$

$$\beta_j = \frac{r_{j+1, j+1}}{r_{j, j}}, \quad j = 1, 2, \dots, N - 1,$$

with $r_{0,0} = 1, r_{0,1} = 0$.

5. Description of Computational Procedures. In the microfiche section of this issue 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 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 μ_0 and the normalized three term recurrence relationship (a_j, b_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 MU[0] through MU[2 \otimes N] to compute the α_j 's and β_j 's of formula (4.2). First, the Cholesky decomposition (formula 4.1) 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 α_j 's and β_j 's which are placed in the arrays A and B respectively.

GAUSSQUADRULE has two modes of operation controlled by the Boolean parameter SYMM which indicates whether the tridiagonal matrix is symmetric or not. When the recurrence relation is produced by GENORTHOPOLY or by CLASSICORTHOPOLY, SYMM is *true*. If SYMM is *false*, the matrix is symmetricized using the formulas (2.2). The diagonal elements α_i are stored in A[I] and the off diagonal elements β_i are stored in B[I].

Beginning at label SETUP, several calculations and initializations are done: the l_1 norm of the tridiagonal matrix and the relative zero tolerance are computed; the first component of each eigenvector W[I] and the Q-R iteration are initialized. 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 (= NORM) outside and to the right of the interval containing the abscissas 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 SORT).

The maximum (EIGMAX) of the eigenvalues (LAMBDA1 and LAMBDA2) of the lower 2×2 submatrix is compared to the maximum (RHO) from the last iteration. If they are close, LAMBDA is replaced by EIGMAX. 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 EPS (see Section 6). 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). These equations have been rearranged to reduce the number of computer operations as suggested by W. Kahan in a report by Corneil [2].

TABLE
*A Comparison of the Abscissas and Weights of the
Gauss-Laguerre Quadrature Rule with $\alpha = -0.75$ and $N = 10$*

	Analytic Recurrence Relationship + QR	Moment Matrix + QR	Concus <u>et al</u> [1].
ABSCISSAS			
1	2.766655867080153 ₁₀ ⁻²	2.766655862878470 ₁₀ ⁻²	2.76665586707972 ₁₀ ⁻²
2	4.547844226059642 ₁₀ ⁻¹	4.547844219368714 ₁₀ ⁻¹	4.54784422605949 ₁₀ ⁻¹
3	1.382425761158619 ₁₀	1.382425759256314 ₁₀	1.382425761158599 ₁₀
4	2.833980012092734 ₁₀	2.833980008561162 ₁₀	2.833980012092697 ₁₀
5	4.850971448764668 ₁₀	4.850971443442301 ₁₀	4.850971448764914 ₁₀
6	7.500010942642896 ₁₀	7.500010935563904 ₁₀	7.500010942642825 ₁₀
7	1.088840802383446 ₁₀ ⁺¹	1.088840801516104 ₁₀ ⁺¹	1.0888408023834404 ₁₀ ⁺¹
8	1.519947804423765 ₁₀ ⁺¹	1.519947803419274 ₁₀ ⁺¹	1.5199478044237603 ₁₀ ⁺¹
9	2.078921462107018 ₁₀ ⁺¹	2.078921460989977 ₁₀ ⁺¹	2.0789214621070107 ₁₀ ⁺¹
10	2.857306016492222 ₁₀ ⁺¹	2.857306015294401 ₁₀ ⁺¹	2.8573060164922106 ₁₀ ⁺¹
WEIGHTS			
1	2.566765557790853 ₁₀	2.566765556932285 ₁₀	2.566765557790772 ₁₀
2	7.733479703443168 ₁₀ ⁻¹	7.733479706154000 ₁₀ ⁻¹	7.73347970344341 ₁₀ ⁻¹
3	2.331328349732182 ₁₀ ⁻¹	2.331328353678223 ₁₀ ⁻¹	2.33132834973219 ₁₀ ⁻¹
4	4.643674708956677 ₁₀ ⁻²	4.643674724992909 ₁₀ ⁻²	4.64367470895670 ₁₀ ⁻²
5	5.549123502036256 ₁₀ ⁻³	5.549123531829512 ₁₀ ⁻³	5.54912350203625 ₁₀ ⁻³
6	3.656466626776441 ₁₀ ⁻⁴	3.656466653186007 ₁₀ ⁻⁴	3.65646662677638 ₁₀ ⁻⁴
7	1.186879857102525 ₁₀ ⁻⁵	1.186879867642139 ₁₀ ⁻⁵	1.18687985710245 ₁₀ ⁻⁵
8	1.584410942056844 ₁₀ ⁻⁷	1.584410958350144 ₁₀ ⁻⁷	1.58441094205678 ₁₀ ⁻⁷
9	6.193266726796867 ₁₀ ⁻¹⁰	6.193266727518338 ₁₀ ⁻¹⁰	6.19326672679684 ₁₀ ⁻¹⁰
10	3.037759926517691 ₁₀ ⁻¹³	3.037759963698451 ₁₀ ⁻¹³	3.03775992651750 ₁₀ ⁻¹³

(Underlined figures are those which disagree with Concus et al [1].)

6. Test Program and Results. The procedures in the microfiche section 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 from 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 driver program given in the microfiche section of this issue 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 [11] and Gauss-Laguerre quadrature in [1]. In the Table, we compare the abscissas and weights of the Gauss-Laguerre quadrature rule with $\alpha = -0.50$ and $N = 10$ computed by: (A) the analytic recurrence relationship and the Q - R algorithm; (B) the moment matrix and the Q - R algorithm; (C) Concus et al. [1]. The calculations for (A) and (B) were performed on the IBM 360.

Acknowledgment. The authors are very pleased to acknowledge Professor Walter Gautschi for his many helpful comments and for his derivation given in Section 4.

Computer Science Department
Stanford University
Stanford, California 94305

Stanford Linear Accelerator Center
Stanford University
Stanford, California 94305

1. P. CONCUS, D. CASSATT, G. JAEHNIG & E. MELBY, "Tables for the evaluation of $\int_0^\infty x^\beta e^{-x} f(x) dx$ by Gauss-Laguerre quadrature," *Math. Comp.*, v. 17, 1963, pp. 245–256. MR 28 #1757.
2. D. CORNEIL, *Eigenvalues and Orthogonal Eigenvectors of Real Symmetric Matrices*, Institute of Computer Science Report, Univ. of Toronto, 1965.
3. P. DAVIS & I. POLONSKY, "Numerical interpolation, differentiation, and integration," in *Handbook of Mathematical Functions, with Formulas, Graphs, and Mathematical Tables*, Nat. Bur. Standards Appl. Math. Ser., 55, Superintendent of Documents, U. S. Government Printing Office, Washington, D. C., 1964; 3rd printing with corrections, 1965. MR 29 #4914; MR 31 #1400.
4. P. J. DAVIS & P. RABINOWITZ, *Numerical Integration*, Blaisdell, Waltham, Mass., 1967. MR 35 #2482.
5. J. G. F. FRANCIS, "The Q - R transformation: A unitary analogue to the L - R transformation. I, II," *Comput. J.*, v. 4, 1961/62, pp. 265–271; 332–345. MR 23 #B3143; MR 25 #744.
6. W. GAUTSCHI, "Construction of Gauss-Christoffel quadrature formulas," *Math. Comp.*, v. 22, 1968, pp. 251–270.
7. G. GOLUB & J. WELSCH, *Calculation of Gauss Quadrature Rules*, Technical Report No. CS 81, Stanford University, 1967.
8. U. HOCHSTRASSER, "Orthogonal polynomials," in *Handbook of Mathematical Functions, with Formulas, Graphs, and Mathematical Tables*, Nat. Bur. Standards Appl. Math. Ser., 55, Superintendent of Documents, U. S. Government Printing Office, Washington, D. C., 1964; 3rd printing with corrections, 1965. MR 29 #4914; MR 31 #1400.
9. I. P. MYSOVSKII, "On the construction of cubature formulas with the smallest number of nodes," *Dokl. Akad. Nauk. SSSR*, v. 178, 1968, pp. 1252–1254 = *Soviet Math. Dokl.*, v. 9, 1968, pp. 277–280. MR 36 #7328.
10. H. RUTISHAUSER, "On a modification of the Q - D algorithm with Graeffe-type convergence," *Z. Angew. Math. Phys.*, v. 13, 1963, pp. 493–496.
11. A. STROUD & D. SECREST, *Gaussian Quadrature Formulas*, Prentice-Hall, Englewood Cliffs, N. J., 1966. MR 34 #2185.
12. H. WILF, *Mathematics for the Physical Sciences*, Wiley, New York, 1962.

JACOBI: MUZERO := $2^{\dagger}(\text{ALFA}+\text{BETA}+1) \times \text{GAMMA}(\text{ALFA}+1) \times \text{GAMMA}(\text{BETA}+1)$
 $/\text{GAMMA}(\text{ALFA}+\text{BETA}+2)$;

comment P(ALFA,BETA)(x) on [-1, +1],

$\omega(x) = (1-x)^{\dagger}\text{ALFA} \times (1+x)^{\dagger}\text{BETA}$, ALFA AND BETA > -1 ;

ABI := 2+ALFA+BETA; A[1] := (BETA-ALFA)/ABI;

B[1] := $\text{sqrt}(4 \times (1+\text{ALFA}) \times (1+\text{BETA}) / ((\text{ABI}+1) \times \text{ABI}^{\dagger}2))$;

A2B2 := $\text{BETA}^{\dagger}2 - \text{ALFA}^{\dagger}2$;

for I := 2 step 1 until N-1 do

begin ABI := 2×I+ALFA+BETA;

A[I] := $\text{A2B2} / ((\text{ABI}-2) \times \text{ABI})$;

B[I] := $\text{sqrt}(4 \times \text{I} \times (\text{I}+\text{ALFA}) \times (\text{I}+\text{BETA}) \times (\text{I}+\text{ALFA}+\text{BETA}) /$
 $((\text{ABI}^{\dagger}2-1) \times \text{ABI}^{\dagger}2))$;

end;

ABI := 2×N+ALFA+BETA;

A[N] := $\text{A2B2} / ((\text{ABI}-2) \times \text{ABI})$;

go to RETURN;

LAGUERRE: MUZERO := $\text{GAMMA}(\text{ALFA}+1.0)$;

comment L(ALFA)(x) on [0, INFINITY),

$\omega(x) = \text{EXP}(-x) \times x^{\dagger}\text{ALFA}$, ALFA > -1 ;

for I := 1 step 1 until N-1 do

begin A[I] := 2×I-1+ALFA;

B[I] := $-\text{sqrt}(\text{I} \times (\text{I}+\text{ALFA}))$;

end;

A[N] := 2×N-1+ALFA; go to RETURN;

HERMITE: MUZERO := sqrt(PI);

comment $H(x)$ on $(-\text{INFINITY}, +\text{INFINITY})$, $w(x) = \text{EXP}(-x^2)$;

for I := 1 step 1 until N-1 do

begin A[I] := 0; B[I] := sqrt(I/2) end;

A[N] := 0;

RETURN: end CLASSICORTHOPOLY ;

```

procedure GENORTHOPOLY(N, MU, A, B);

  value N; integer N;

  real array MU, A, B;

begin comment Given the  $2N+1$  moments (MU) of the weight function,
      generate the recursion coefficients (A, B) of the normalized
      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 then 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;

  A[N] := R[N,N+1]/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 N; real MUZERO; boolean SYMM;
    real array A, B, C, T, W ;
    begin comment Given the coefficients (A, B, C) of the three term
        recurrence relation:  $P(K) = (A(K)X+B(K))P(K-1)-C(K)P(K-2)$ ,
        this procedure computes the abscissas T and the weights W
        of the Gaussian type quadrature rule associated with the ortho-
        gonal polynomial by QR type iteration with origin shifting.;
    integer I, J, K, M, M1;
    real NORM, EPS, CT, ST, F, Q, 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]/(AI*A[I+1]));
            end;
            A[N] := -B[N]/A[N];
    comment Find the maximum row sum norm and initialize W[ ];
    SETUP: B[0] := 0; NORM := 0;
        for I := 1 step 1 until N-1 do
            begin NORM := MAX(NORM, abs(B[I-1])`+abs(A[I])`+abs(B[I]))`;
                W[I] := 0;
            end;

```

```

NORM := MAX(NORM, abs(A[N])+abs(B[N-1]));

EPS := NORM*16.0*(-14); comment Relative zero tolerance ;

W[I] := 1.0; W[N] := 0; M := N;

LAMBDA := LAMB1 := LAMB2 := RHO := NORM;

comment Look for convergence of lower diagonal element ;

INSPECT: if M = 0 then go to SORT else I := K := M1 := M-1;

    if abs(B[M1]) ≤ EPS then

        begin T[M] := A[M]; W[M] := MUZERO*W[M]^2;

            RHO := (if LAMB1 < LAMB2 then LAMB1 else LAMB2);

            M := M1; go to INSPECT ;

        end;

comment Small off diagonal element means matrix can be split ;

    for I := I-1 while abs(B[I]) > EPS do K := I;

comment Find eigenvalues of lower 2X2 and select accelerating shift ;

    B2 := B[M1]^2; DET := sqrt((A[M1]-A[M])^2+4.0*B2) ;

    AA := A[M1]+A[M] ;

    LAMB2 := 0.5*(if AA ≥ 0 then AA+DET else AA-DET);

    LAMB1 := (A[M1]*A[M]-B2)/LAMB2;

    EIGMAX := MAX(LAMB1, LAMB2);

    if abs(EIGMAX-RHO) ≤ 0.125*abs(EIGMAX) then

        LAMBDA := RHO := EIGMAX else RHO := EIGMAX;

comment Transform block from K to M ;

    CJ := B[K]; B[K-1] := A[K]-LAMBDA;

    for J := K step 1 until M1 do

        begin R := sqrt(CJ^2+B[J-1]^2) ;

            ST := CJ/R; CT := B[J-1]/R; AJ := A[J];

            B[J-1] := R; CJ := B[J+1]*ST;

```



```

    B[J+1] := -B[J+1]*XT;
    F := AJ*XT + B[J]*ST;
    Q := B[J]*XT + A[J+1]*ST;
    A[J] := F*XT + Q*ST;  B[J] := F*ST - Q*XT;
    WJ := W[J];
    A[J+1] := AJ+A[J+1]-A[J];
    W[J] := WJ*XT+W[J+1]*ST;  W[J+1] := WJ*ST-W[J+1]*XT;

  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
          WJ := T[I-1];  T[I-1] := T[I];  T[I] := WJ;
          WJ := W[I-1];  W[I-1] := W[I];  W[I] := WJ;
          EX := true
        end;
      if ~ EX then go to RETURN;
    end;
  end;
RETURN:  end GAUSSQUADRULE ;

```

```

begin

comment Driver program for GAUSSQUADRULE;

    real array A, C, T, W[1:10], MU[0:20], B[0:10];

    real MUZERO; integer I, N;

        N := 10;

comment Legendre polynomials. ;

    outstring (1, 'Legendre Quadrature. ');

    CLASSICORTHOPOLY(1, 0, 0, N, A, B, MUZERO);

    GAUSSQUADRULE(N, A, B, C, MUZERO, true, T, W);

    outstring (1, 'abscissas:'); outarray (1, T);

    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, MUZERO);

    GAUSSQUADRULE (N, A, B, C, MUZERO, true T, W);

    outstring (1, 'abscissas:'); outarray (1, T);

    outstring (1, 'weights:'); outarray (1, W);

    MU[0] := MUZERO := 1.772453850905516; 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;
```

COMPUTATION OF GALOIS GROUP ELEMENTS

OF A POLYNOMIAL EQUATION

E. J. COCKAYNE

FORTRAN PROGRAM

C PRINTS IRREDUCIBLE FACTORS MOD P-OF INTEGER POLYNOMIAL DEGREE D
 C (LESS THAN 10).COEFFICIENTS OF POLYNOMIAL ARE A(1)...A(D+1) IN
 C ASCENDING POWERS OF X.

C
 C

```

    INTEGER A(10),A1(10),B(5),IREM(5),IQUO(10),P,D
    MOD(N1,N2)=N1-N2*(N1/N2)
    READ (2,38) A,D,P
    L=D
    N=1
    CALL VECTO (IREM,5)
    CALL VECTO (IQUO,10)
  
```

C

C GENERATES MONIC POLYS B DEGREE N.LE.4

C

```

    28 CALL VECTO(B,5)
      B(1)=-1
      B(N+1)=1
    33 B(1)=B(1)+1
      DO 20 I=1,N
        IF (B(I)-P) 34,19,34
    19 B(I)=0
      IF (I-N) 20,48,20
    20 B(I+1)=B(I+1)+1
  
```

C

C TEST FOR FACTOR B

C

```

    34 DO 39 I3=1,10
    39 A1(I3)=A(I3)
      CALL VECTO (IQUO,10)
      CALL 10DIV (A1,L+1,B,N+1,IQUO,IREM)
      ISUM=0
      DO 54 I9=1,N
        IREM(I9)=MOD(IREM(I9),P)
        IF (IREM(I9)) 111,54,54
    111 IREM(I9)=IREM(I9)+P
      54 ISUM=ISUM+IREM(I9)
      CALL VECTO (IREM,5)
      IF (ISUM) 33,112,33
  
```

C

C FACTOR OF DEGREE N

C

```

    112 DO 24 I7=1,10
      IQUO(I7)=MOD(IQUO(I7),P)
      IF (IQUO(I7)) 113,24,24
    113 IQUO(I7)=IQUO(I7)+P
    24 CONTINUE
      WRITE (3,127) N,B
      L=L-N
      IF (L-(2*N-1)) 30,30,114
    114 DO 58 I4=1,10
      58 A(I4)=IQUO(I4)
      GO TO 34
  
```

C