An Algorithm for Generating Interpolatory Quadrature Rules of the Highest Degree of Precision with Preassigned Nodes for General Weight Functions

T. N. L. PATTERSON
The Queen's University of Belfast

The construction of an algorithm is described for generating interpolatory quadrature rules of the highest degree of precision with arbitrarily preassigned nodes for general constant signed weight functions. It is of very wide application in that to operate, only the definition of the 3-term recurrence relation for the orthogonal polynomials associated with the weight function need be supplied. The algorithm can be used to produce specific individual quadrature rules or sequences of rules by iterative application.

Categories and Subject Descriptors: G.1.4 **[Numerical Analysis]**: Quadrature and Numerical Differentiation—*Gaussian quadrature*; G.4 **[Mathematics of Computing]**: Mathematical Software—*algorithm analysis*

General Terms: Algorithms

Additional Key Words and Phrases: Gaussian integration, integration rules, optimal rules, preassigned nodes, rule extension

1. INTRODUCTION

The problem of creating high precision interpolatory quadrature rules with preassigned nodes has been discussed in detail by Krylov [7]. The rules take the general form

$$\int_a^b w(x)f(x)\,dx \approx \mathscr{R}(n, m) = \sum_{k=1}^{n} A_k f(x_k) + \sum_{k=1}^{m} A_{k+n} f(x_{k+n}) \qquad (1.1)$$

where the n nodes $x_1, \ldots, x_n$ are preassigned. The m free nodes $x_{n+1}, \ldots, x_{n+m}$ are calculated to achieve the highest possible degree of precision and are said to be *optimally* added. The weight function $w(x)$ is of the constant sign in $[a, b]$.

Author's address: Department of Applied Mathematics and Theoretical Physics, The Queen's University of Belfast, Belfast, BT7 1NN, Northern Ireland.

In 1964, Kronrod [6] calculated tables of such rules, which had the effect of stimulating a fresh interest in this approach to numerical integration. Kronrod chose the n preassigned nodes as the zeros of the Legendre polynomial $P_n(x)$ corresponding to the well-known Gauss–Legendre rule and computed $n + 1$ additional nodes (the minimum possible) to provide a new $2n + 1$ node rule of an algebraic degree of precision of at least $3n + 1$. It may be noted that any arbitrary set of n preassigned nodes leads to exactly the same degree of precision in the new $2n + 1$ node rule (providing of course it exists), but by selecting the nodes of the Gauss rule, two high-precision results could be obtained without discarding work already invested in a Gaussian calculation.

The additional nodes calculated by Kronrod can be shown to be the zeros of the polynomial $E_{n+1}(x)$ which satisfies

$$\int_{-1}^{1} P_n(x)E_{n+1}(x)x^k\,dx = 0, \qquad \text{for} \quad k = 0, \ldots, n. \tag{1.2}$$

It is interesting to note that a study of the properties of the polynomial $E_{n+1}(x)$ can be traced back to 1894 as the subject of a correspondence [14] between no less mathematicians than Hermite and Stieltjes. The problem of existence was discussed further by Szegö [17] in 1934 for a particular class of weight functions. In fact the polynomial $E_{n+1}(x)$ is orthogonal with respect to a variable signed weight function [16] for which the results of standard orthogonality theory relating to the existence and distribution of its zeros are not applicable. An account of the historical background and the theoretical properties of the so-called Stieltjes polynomials can be found in an excellent review article by Monegato [9].

The ideas have been developed by a number of authors including Patterson [10], Piessens and Branders [12], and Monegato [8]. The calculations of Kronrod [6] did not proceed beyond the single extension based on preassigned Gaussian nodes. However, the paper of Patterson [10] exploited the idea of iteratively applying the Krylov procedure to obtain sequences of rules of increasing polynomial precision with interlacing nodes, the so-called *optimal* extensions, and resulted in the development of an efficient high precision quadrature algorithm [11]. Many scientific software libraries now incorporate the Kronrod scheme or its derivatives as the basis of general purpose adaptive integrators.

In this paper we discuss the construction of a stable and efficient algorithm for generating quadrature rules of the form (1.1). The algorithm is of very wide application in that, to operate, it need be supplied only with a definition of the recurrence relation for the orthogonal system of polynomials appropriate to the weight function. The value of the zero moment integral, h_0, for the domain of integration (defined in Section 2.1) is required as a normalizing factor in the calculation of the quadrature weights.

High precision integration rules with preassigned nodes are of importance in many situations where high accuracy is needed but certain nodes must be constrained. This can occur, for example, in solving some types of integral equations. Further, the properties of quadrature rules arising from variable signed weight functions are still poorly understood, and the algorithm can thus form a useful investigative tool.

2. THEORETICAL BACKGROUND

2.1 Basic Equations

Let $\phi_i(x)$, $i = 0, 1, \ldots$ be a set of polynomials, orthogonal on the interval $[a, b]$ with respect to the constant signed weight function $w(x)$, and let

$$\int_a^b w(x)\phi_i(x)\phi_j(x)\,dx = h_j\delta_{i,j} \tag{2.1}$$

define the jth moment integral.

The $\phi_i(x)$ are taken to satisfy the 3-term recurrence relation

$$\phi_{k+1} = (c_k x + d_k)\phi_k + e_k\phi_{k-1} \tag{2.2}$$

with $\phi_{-1} = 0$ and $\phi_0 = 1$.

The coefficients c_k, e_k, and h_k are related according to

$$e_k c_{k-1}\, h_{k-1} = -c_k h_k \tag{2.3}$$

which gives immediately,

$$h_k/h_0 = (-1)^k(c_0/c_k)\prod_{i=1}^{k} e_i. \tag{2.4}$$

We wish to construct an $n + m$ node quadrature rule consisting of n preassigned nodes, together with m nodes determined to give the rule the highest possible algebraic degree of precision. Let the preassigned nodes be the zeros of the polynomial $H_n(x)$ expressed as

$$H_n(x) = \sum_{i=m_0}^{n} (\tau_i/h_i)\phi_i(x) \tag{2.5}$$

and let

$$E_m(x) = \sum_{i=0}^{m} \epsilon_i\phi_i(x) \tag{2.6}$$

be the polynomial to be determined whose m zeros are the required optimal nodes. The scaling of the coefficients of $H_n(x)$ with respect to h_i is beneficial. It postpones the need to compute the moments until the quadrature weights are required and generally makes the numbers appearing in the course of the calculations more manageable. It can be shown (Krylov [7]) that to achieve the highest possible precision, namely, $n + 2m - 1$, requires

$$\int_a^b w(x)H_n(x)E_m(x)x^k\,dx = 0, \qquad k = 0, \ldots, m - 1. \tag{2.7}$$

The polynomial $E_m(x)$ is orthogonal with respect to the variable signed weight $w(x)H_n(x)$. Thus, its zeros are not only the m optimal nodes for the $n + m$ point extended rule with respect to the *constant signed* weight $w(x)$, but are also the nodes of the Gaussian rule of degree $2m - 1$ appropriate to the *variable signed* weight $w(x)H_n(x)$.

2.2 Properties of the Extended Rules

In this section we briefly summarize some theoretical results on the minimum value of m and on the integrating degree of (1.1) which give guidance on the structure of the rules which may be usefully constructed.

We have noted in Section 1.1 that the situation is complicated by the fact that generally the polynomial orthogonality is with respect to a variable signed weight function, and the concept of degeneracy [16] must be introduced. We define degeneracy as follows.

Definition. Let $V(x)$ be a weight function which may change sign in the domain of integration $[a, b]$ and let $p_m(x)$ be an orthogonal polynomial of *exact* degree m such that

$$\begin{aligned} \int_a^b V(x)p_m(x)x^j\,dx &= 0, \qquad j = 0, \ldots, m + \mu - 1 \\ &\neq 0, \qquad j = m + \mu. \end{aligned} \tag{2.8}$$

If $\mu > 0$, we say that the orthogonality is μ-fold degenerate. If $\mu = 0$, we say that the orthogonality is nondegenerate.

The following two simple theorems can be readily established.

THEOREM 2.1. *Let $\mathscr{R}(n, m)$ be the extended interpolatory quadrature rule* (1.1) *resulting from adding m nodes to n preassigned nodes, and let $H_n(x)$ and $E_m(x)$ be the polynomials whose roots are the respective nodes. Let $E_m(x)$ be μ-fold degenerate with respect to the weight $w(x)H_n(x)$. Then $\mathscr{R}(n, m)$ has integrating degree $n + 2m + \mu - 1$.*

PROOF. A general polynomial of degree $n + 2m + \mu - 1$ can be expressed as

$$G_{n+2m+\mu-1}(x) = Q_{n+m-1}(x) + H_n(x)E_m(x)q_{m+\mu-1}(x) \tag{2.9}$$

where Q and q are arbitrary polynomials of respective degrees $n + m - 1$ and $m + \mu - 1$. Multiplying by $w(x)$ and integrating gives

$$\begin{aligned} &\int_a^b w(x)G_{n+2m+\mu-1}(x)\,dx \\ &\quad = \int_a^b w(x)Q_{n+m-1}(x)\,dx + \int_a^b w(x)H_n(x)E_m(x)q_{m+\mu-1}(x)\,dx. \end{aligned} \tag{2.10}$$

The second term on the right of (2.10) vanishes due to the degeneracy condition (2.8). Both terms are integrated exactly by $\mathscr{R}(n, m)$, the first because the rule is interpolatory, and the second due to the presence of the root factors $E_m(x)$ and $H_n(x)$. The result follows. □

THEOREM 2.2. *Let $\mathscr{R}(n, m)$, $H_n(x)$ and $E_m(x)$ be defined as in Theorem* 2.1. *Then the quadrature weights associated with the s optimal nodes of $\mathscr{R}(n + m, s)$ corresponding to the extension of $\mathscr{R}(n, m)$ will be zero unless $s > m + \mu$.*

PROOF. From the Lagrangian interpolation formula, the weights associated with the s optimal nodes $v_1, \ldots, v_s$ are given by

$$A_k \propto \int_a^b w(x)H_n(x)E_m(x) \prod_{\substack{i=1 \\ i\neq k}}^{s} (x - v_i)\, dx, \tag{2.11}$$

which vanish due to the degeneracy condition (2.8) unless $s > m + \mu$. □

These results are easily demonstrated for the weight $w(x) = (1 - x^2)^{-1/2}$ in $[-1, 1]$ with the n roots of the Chebyshev polynomial $T_n(x)$ of the first kind of degree n being preassigned and $n + 1$ nodes being added optimally. It is simple to show that $E_{n+1}(x) = (1 - x^2)U_{n-1}(x)$ is $(n - 2)$-fold degenerate where $U_{n-1}(x)$ is the Chebyshev polynomial of the second kind of degree $n - 1$. That is,

$$\begin{aligned} \int_{-1}^{1} (1 - x^2)^{-1/2}T_n(x)E_{n+1}(x)x^j\, dx &= 0, \qquad j = 0, \ldots, 2n - 2 \\ &\neq 0, \qquad j = 2n - 1. \end{aligned} \tag{2.12}$$

Theorem 2.1 indicates that the degree of the resulting $2n + 1$ point rule will be $4n - 1$. Theorem 2.2 shows that the minimum number of nodes by which this new rule may be usefully extended is $2n$.

3. COMPUTATIONAL DETAILS

3.1 Calculation of the Polynomial $E_m(x)$

There are a number of techniques available for determining the coefficients of $E_m(x)$ from (2.7). A numerically satisfactory procedure can be obtained from the observation that if

$$H_{n+m}(x) = H_n(x)E_m(x) = \sum_{j=0}^{m} \epsilon_j \sum_{i=m_0}^{n} (\tau_i/h_i)\phi_i\phi_j = \sum_{j=0}^{n+m} (\rho_j/h_j)\phi_j, \tag{3.1}$$

then (2.7) implies

$$\rho_j = 0, \qquad j = 0, \ldots, m - 1, \tag{3.2}$$

and so,

$$H_n(x)E_m(x) = \sum_{j=m}^{n+m} (\rho_j/h_j)\phi_j. \tag{3.3}$$

To obtain the coefficients of $E_m(x)$, we multiply (3.1) by $w(x)\phi_s(x)$ and integrate over $[a, b]$. Then,

$$\rho_s = \sum_{j=0}^{m} \epsilon_j \sum_{i=m_0}^{n} (\tau_i/h_i) \int_a^b w(x)\phi_i\phi_j\phi_s\, dx = \sum_{j=0}^{m} \epsilon_j \sum_{i=m_0}^{n} \tau_i a_i^{(s,j)} \tag{3.4}$$

where

$$h_i a_i^{(s,j)} = \int_a^b w(x)\phi_i\phi_j\phi_s\, dx. \tag{3.5}$$

The coefficients $a_i^{(s,j)}$ are simply the coefficients of the expansion of $\phi_s\phi_j$, namely,

$$\phi_s\phi_j = \sum_{i=|s-j|}^{s+j} a_i^{(s,j)}\phi_i. \tag{3.6}$$

Expressions for $a_i^{(s,j)}$ for a number of orthogonal systems can be found in books such as that by Gradshteyn and Ryzhik [4], for example, but to ensure complete generality we show in Section 3.2 how they can be generated from the defining recurrence relation (2.2).

Equation (3.2) requires that

$$\sum_{j=0}^{m} \epsilon_j \sum_{i=m_0}^{n} \tau_i a_i^{(s,j)} = 0, \qquad \text{for} \quad s = 0, \ldots, m-1, \tag{3.7}$$

and arbitrarily taking $\epsilon_m = 1$, we obtain a symmetric system of m linear equations in the m unknowns $\epsilon_0, \epsilon_1, \ldots, \epsilon_{m-1}$ which completely determine $E_m(x)$. Once $E_m(x)$ is known, the coefficients $\rho_m, \ldots, \rho_{m+n}$ can be calculated from (2.4) and (3.4) by direct substitution. Thus, $H_{n+m}(x)$ is known, and another extension can be generated with $n + m$ replacing n and m replacing m_0. At each stage the zeros of $E_m(x)$ can be calculated and a complete optimal interpolatory quadrature rule obtained.

This procedure has the advantage over that discussed originally by Patterson [10] in that the roots of $E_m(x)$ play no direct part in the calculation of the next extension, thus minimizing the accumulation of errors. We note that the form of (3.3) also allows (2.5) to be interpreted as the polynomial whose zeros represent the nodes of a quadrature rule arising by the same process as (2.7) from optimally adding m_0 nodes to a set of $n - m_0$ preassigned nodes.

The form of (3.5) allows certain terms to be eliminated from the summation. Due to the orthogonality property (2.1), $a_i^{(s,j)}$ is zero if $j + s < i$, $i + s < j$, or $i + j < s$, and (3.4) becomes

$$\rho_s = \sum_{j=0}^{m} \epsilon_j \sum_{i=\max(m_0,|s-j|)}^{\min(n,s+j)} \tau_i a_i^{(s,j)}. \tag{3.8}$$

Consequently, the symmetric linear system (3.7) becomes

$$\sum_{j=0}^{m} \epsilon_j \sum_{i=\max(m_0,|s-j|)}^{\min(n,s+j)} \tau_i a_i^{(s,j)} = 0 \qquad \text{for} \quad s = 0, \ldots, m-1. \tag{3.9}$$

If the domain of integration and the weight function are symmetric, the coefficients $a_i^{(s,j)}$ are zero when $i + s + j$ is odd. Taking account of this halves the size of the linear system (3.9). Note that due to the definition adopted for the coefficients in (2.5), h_i does not appear in (3.7) or (3.9), giving a natural scaling to the system of equations.

3.2 Calculation of the Orthogonal Product Expansion

To obtain values of the coefficients $a_i^{(s,j)}$ for use in (3.8) and (3.9), we must calculate the expansion of $\phi_s\phi_j$. Assume without loss of generality that $s \geq j$.

Using (2.2) and (3.6) we obtain

$$\begin{aligned}\phi_s\phi_j &= (c_{j-1}x + d_{j-1})\phi_s\phi_{j-1} + e_{j-1}\phi_s\phi_{j-2} \\ &= c_{j-1}\sum_{i=s-j+1}^{s+j-1} a_i^{(s,j-1)} x\phi_i\phi_s + d_{j-1}\phi_s\phi_{j-1} + e_{j-1}\phi_s\phi_{j-2}.\end{aligned} \tag{3.10}$$

Again using (2.2), substituting for $x\phi_i$ gives

$$\begin{aligned}\phi_s\phi_j &= c_{j-1}\sum_{i=s-j+1}^{s+j-1}\frac{a_i^{(s,j-1)}}{c_i}(\phi_{i+1} - d_i\phi_i - e_i\phi_{i-1}) + d_{j-1}\phi_s\phi_{j-1} + e_{j-1}\phi_s\phi_{j-2} \\ &= c_{j-1}\sum_{i=s-j+2}^{s+j}\frac{a_{i-1}^{(s,j-1)}}{c_{i-1}}\phi_i + \sum_{i=s-j+1}^{s+j-1}\left(d_{j-1} - c_{j-1}\frac{d_i}{c_i}\right)a_i^{(s,j-1)}\phi_i \\ &\quad - c_{j-1}\sum_{i=s-j}^{s+j-2}\frac{e_{i+1}}{c_{i+1}}a_{i+1}^{(s,j-1)}\phi_i + e_{j-1}\sum_{i=s-j+2}^{s+j-2}a_i^{(s,j-2)}\phi_i \\ &= \sum_{i=s-j}^{s+j} a_i^{(s,j)}\phi_i.\end{aligned} \tag{3.11}$$

Clearly, from (3.11) it is a simple matter to calculate $a_i^{(s,j)}$ by recurrence using the values of $a_i^{(s,j-1)}$ and $a_i^{(s,j-2)}$. To begin the recurrence requires the expansion of $\phi_s\phi_0$ and $\phi_s\phi_1$. We have trivially,

$$\phi_s\phi_0 = \phi_s \qquad \text{(i.e., } a_s^{(s,0)} = 1\text{)}.$$

Using (2.2) gives immediately,

$$\phi_s\phi_1 = -\frac{e_s c_0}{c_s}\phi_{s-1} + \left(d_0 - \frac{d_s c_0}{c_s}\right)\phi_s + \frac{c_0}{c_s}\phi_{s+1}. \tag{3.12}$$

3.3 Calculation of the Roots of $E_m(x)$

This can be accomplished using a generalization of the Bairstow process due to Golub and Robertson [1]. Let us consider the general problem of finding the roots of

$$S_N(x) = \sum_{i=0}^{N}\gamma_i\phi_i(x). \tag{3.13}$$

We require to compute the decomposition

$$\begin{aligned}\sum_{i=0}^{N}\gamma_i\phi_i &= (\phi_2 - \alpha\phi_1 - \beta\phi_0)\sum_{i=0}^{N-2}\delta_i\phi_i + A\phi_1 + B\phi_0 \\ &= (\phi_2 - \alpha\phi_1 - \beta\phi_0)Q_{N-2}(x) + A\phi_1 + B\phi_0.\end{aligned} \tag{3.14}$$

Writing

$$\phi_k\phi_1 = l_{k+1}\phi_{k+1} + m_k\phi_k + r_{k-1}\phi_{k-1} \tag{3.15}$$

and

$$\phi_k\phi_2 = s_{k+2}\phi_{k+2} + t_{k+1}\phi_{k+1} + u_k\phi_k + v_{k-1}\phi_{k-1} + w_{k-2}\phi_{k-2}, \tag{3.16}$$

we can compute δ_i, A, and B from

$$\delta_k = \frac{1}{s_{k+2}} \{\gamma_{k+2} + (\alpha l_{k+2} - t_{k+2})\delta_{k+1} + (\beta + \alpha m_{k+2} - u_{k+2})\delta_{k+2} + (\alpha r_{k+2} - v_{k+2})\delta_{k+3} - w_{k+2}\delta_{k+4}\} \tag{3.17}$$

for $k = N-2, N-1, \ldots, 0$ with

$$\delta_{N+2} = \delta_{N+1} = \delta_N = \delta_{N-1} = 0 \tag{3.18}$$

and

$$A = \gamma_1 + \alpha\delta_0 + (\beta + \alpha m_1 - r_1)\delta_1 + (\alpha r_1 - v_1)\delta_2 - w_1\delta_3 \tag{3.19}$$

$$B = \gamma_0 + \beta\delta_0 + \alpha r_0\delta_1 - w_0\delta_2. \tag{3.20}$$

The quantities l_k, m_k, r_k, s_k, t_k, u_k, v_k, and w_k in (3.16) and (3.17) can be expressed in terms of the coefficients of the recurrence relation (2.2). Equation (3.12) immediately gives expressions for l_k, m_k, and r_k; thus

$$l_k = c_0/c_{k-1} \tag{3.21}$$

$$m_k = d_0 - c_0 d_k/c_k = d_0 - d_k l_{k+1} \tag{3.22}$$

$$r_k = -c_0 e_{k+1}/c_{k+1} = -e_{k+1} l_{k+2}. \tag{3.23}$$

A little algebra using (2.2) gives

$$s_{k+2} = c_1 l_{k+1}/c_{k+1}, \tag{3.24}$$

$$t_{k+2} = d_1 l_{k+2} - c_1 l_{k+2} d_{k+2}/c_{k+2} + c_1 m_{k+1}/c_{k+1}, \tag{3.25}$$

$$u_{k+2} = d_1 m_{k+2} + e_1 - c_1 l_{k+3} e_{k+3}/c_{k+3} - c_1 m_{k+2} d_{k+2}/c_{k+2} + c_1 r_{k+1}/c_{k+1}, \tag{3.26}$$

$$v_{k+2} = d_1 r_{k+2} - c_1 m_{k+3} e_{k+3}/c_{k+3} - c_1 r_{k+2} d_{k+2}/c_{k+2}, \tag{3.27}$$

$$w_{k+2} = -c_1 r_{k+3} e_{k+3}/c_{k+3}. \tag{3.28}$$

The calculation now proceeds in the standard Bairstow manner. The quantities A and B depend on α and β. We seek to find α^* and β^* such that

$$A(\alpha^*, \beta^*) = B(\alpha^*, \beta^*) = 0. \tag{3.29}$$

Given an initial approximation $(\alpha^{(0)}, \beta^{(0)})$, we calculate the sequence

$$\alpha^{(j+1)} = \alpha^{(j)} + \Delta\alpha, \qquad \beta^{(j+1)} = \beta^{(j)} + \Delta\beta, \qquad \text{for } j = 0, 1, \ldots \tag{3.30}$$

where

$$\Delta\alpha = (BA_\beta - AB_\beta)/(A_\alpha B_\beta - A_\beta A_\alpha) \tag{3.31}$$

$$\Delta\beta = (AB_\alpha - BA_\alpha)/(A_\alpha B_\beta - A_\beta A_\alpha) \tag{3.32}$$

with

$$A_\alpha = \frac{\partial A}{\partial \alpha}, \qquad A_\beta = \frac{\partial A}{\partial \beta}, \qquad B_\alpha = \frac{\partial B}{\partial \alpha}, \qquad B_\beta = \frac{\partial B}{\partial \beta}. \tag{3.33}$$

The various quantities in (3.31) and (3.32) are evaluated at $(\alpha^{(j)}, \beta^{(j)})$. The partial derivatives of A and B are calculated from (3.19) and (3.20); thus,

$$\frac{\partial A}{\partial \alpha} = \delta_0 + m_1\delta_1 + r_1\delta_2 + \alpha\frac{\partial \delta_0}{\partial \alpha} + (\beta + \alpha m_1 - r_1)\frac{\partial \delta_1}{\partial \alpha} + (\alpha r_1 - v_1)\frac{\partial \delta_2}{\partial \alpha} - w_1\frac{\partial \delta_3}{\partial \alpha} \tag{3.34}$$

$$\frac{\partial A}{\partial \beta} = \delta_1 + \alpha\frac{\partial \delta_0}{\partial \beta} + (\beta + \alpha m_1 - r_1)\frac{\partial \delta_1}{\partial \beta} + (\alpha r_1 - v_1)\frac{\partial \delta_2}{\partial \beta} - w_1\frac{\partial \delta_3}{\partial \beta} \tag{3.35}$$

$$\frac{\partial B}{\partial \alpha} = r_0\delta_1 + \beta\frac{\partial \delta_0}{\partial \alpha} + \alpha r_0\frac{\partial \delta_1}{\partial \alpha} - w_0\frac{\partial \delta_2}{\partial \alpha} \tag{3.36}$$

$$\frac{\partial B}{\partial \beta} = \delta_0 + \beta\frac{\partial \delta_0}{\partial \beta} + \alpha r_0\frac{\partial \delta_1}{\partial \beta} - w_0\frac{\partial \delta_2}{\partial \beta}. \tag{3.37}$$

The partial derivative in (3.34)–(3.37) is obtained from (3.17); thus,

$$\frac{\partial \delta_k}{\partial \alpha} = \frac{1}{s_{k+2}}\left\{l_{k+2}\delta_{k+1} + m_{k+2}\delta_{k+2} + r_{k+2}\delta_{k+3} + (\alpha l_{k+2} - t_{k+2})\frac{\partial \delta_{k+1}}{\partial \alpha} + (\beta + \alpha m_{k+2} - u_{k+2})\frac{\partial \delta_{k+2}}{\partial \alpha} + (\alpha r_{k+2} - v_{k+2})\frac{\partial \delta_{k+3}}{\partial \alpha} - w_{k+2}\frac{\partial \delta_{k+4}}{\partial \alpha}\right\} \tag{3.38}$$

$$\frac{\partial \delta_k}{\partial \beta} = \frac{1}{s_{k+2}}\left\{\delta_{k+2} + (\alpha l_{k+2} - t_{k+2})\frac{\partial \delta_{k+1}}{\partial \beta} + (\beta + m_{k+2} - u_{k+2})\frac{\partial \delta_{k+2}}{\partial \beta} + (\alpha r_{k+2} - v_{k+2})\frac{\partial \delta_{k+3}}{\partial \beta} - w_{k+2}\frac{\partial \delta_{k+4}}{\partial \beta}\right\} \tag{3.39}$$

for $k = N - 2, N - 1, \ldots, 0$ with

$$\frac{\partial \delta_k}{\partial \alpha} = \frac{\partial \delta_k}{\partial \beta} = 0, \qquad \text{for} \quad k = N - 1, N, N + 1, N + 2. \tag{3.40}$$

The recurrences (3.38) and (3.39) are calculated simultaneously with (3.17). The partial derivatives of δ_0, δ_1, δ_2, and δ_3 are then used in (3.34)–(3.37). Golub and Robertson [1] show that the process converges quadratically, provided the roots of the quadratic factor are not also roots of the quotient $Q_{N-2}(x)$. In practice, the numbers can be kept more manageable by calculating the recurrences (3.17), (3.38), and (3.39) in terms of the scaled quantities $s_{k+2}\delta_k$, $s_{k+2}\partial\delta_k/\partial\alpha$, and $s_{k+2}\partial\delta_k/\partial\beta$.

The algorithm used to find the roots of $E_m(x)$ is now described. Let ϵ be a small positive real number (typically $\mathcal{O}(10^{-7})$). We define the *real* seed approximation as the values of α and β which make ϵ and zero (or ϵ and $-\epsilon$ when symmetry is present) roots of the quadratic factor $(\phi_2 - \alpha\phi_1 - \beta\phi_0)$. The *complex*

seed approximation is defined as the values of α and β which make $\epsilon \pm i\epsilon$ roots of the quadratic factor. If m is odd, a single real root is first extracted, and the polynomial is deflated using Newton iterations beginning with ϵ as the initial approximation. Details of that are covered in the next section. The Bairstow process is used to find the remaining roots.

Initially, set $Q_m(x) = E_m(x)$ (or equal to the Newton-deflated polynomial if m is odd) and choose $(\alpha_{m+2}, \beta_{m+2})$ using the *real* seed approximation. For $j = m, m-2, m-4, \ldots$, carry out the following operations:

(1) *Root Location.* Use the Bairstow process to find (α_j^*, β_j^*) so that

$$Q_j(x) = (\phi_2 - \alpha_j^* \phi_1 - \beta_j^* \phi_0) Q_{j-2}(x) \tag{3.41}$$

where $Q_j(x)$ is the appropriate quotient of degree j with earlier roots eliminated. If convergence for the previous quadratic factor (iteration $j+2$) was satisfactory, use the initial approximation $(\alpha_{j+2}, \beta_{j+2})$; otherwise, use the *real* seed approximation. If the convergence on the present step is unsatisfactory, repeat using the *complex* seed approximation. If this is still unsatisfactory, the failure is recorded and the latest quadratic factor accepted.

(2) *Root Refinement.* With initial approximation (α_j^*, β_j^*) from step 1, use the Bairstow process on the full expansion $E_m(x)$ to find (α_j, β_j) so that

$$E_m(x) = (\phi_2 - \alpha_j \phi_1 - \beta_j \phi_0) S_{m-2}(x) \tag{3.42}$$

where $S_{m-2}(x)$ is the appropriate quotient of degree $m-2$.

(3) *Quotient Refinement.* With (α_j, β_j) from step 2, update the quotient $Q_{j-2}(x)$ using the division operation (3.17) to give

$$Q_j(x) = (\phi_2 - \alpha_j \phi_1 - \beta_j \phi_0) Q_{j-2}(x). \tag{3.43}$$

The successive quotients $Q_j(x)$ decrease in degree by two at each stage, reducing the computational work. Since the approximation to the quadratic factor at the start of step 2 is normally good, convergence for the full expansion is rapid (usually only one iteration is required). This process of root location on the reduced expansion followed by root refinement on the full expansion is effective in minimizing the accumulation of errors.

3.4 Division of the Orthogonal Expansion by a Linear Factor

This procedure is required to evaluate the orthogonal expansion and its first derivative for application to the Newton iteration (Section 3.3) and the calculation of the quadrature weights (Section 3.5).

We wish to divide the expansion $S_N(x) = \sum_{k=0}^{N} \gamma_k \phi_k$ by $(x - \alpha)$ to give a quotient $Q_{N-1}(x)$ and remainder R. Thus,

$$S_N(x) = \sum_{k=0}^{N} \gamma_k \phi_k = (x - \alpha) Q_{N-1}(x) + R = (x - \alpha) \sum_{i=0}^{N-1} \delta_i \phi_i + R. \tag{3.44}$$

Using the recurrence (2.2) and (3.15), it is easily established that

$$c_0 \sum_{k=0}^{N} \gamma_k \phi_k = \sum_{k=1}^{N} \delta_{k-1} l_k \phi_k - \sum_{k=0}^{N-1} \delta_k l_{k+1} (d_k + \alpha c_k) \phi_k - \sum_{k=0}^{N-2} \delta_{k+1} e_{k+1} l_{k+2} \phi_k + c_0 R \tag{3.45}$$

and

$$c_0 R = c_0 \gamma_0 + \delta_0 l_1 (d_0 + \alpha c_0) + \delta_1 e_1 l_2. \tag{3.46}$$

Equation (3.45) reduces to

$$l_k \delta_{k-1} = c_0 \gamma_k + \delta_k l_{k+1} (d_k + \alpha c_k) + e_{k+1} l_{k+2} \delta_{k+1}, \tag{3.47}$$

and since $l_0 = 1$ it is clear that

$$c_0 R = \delta_{-1}. \tag{3.48}$$

To obtain the solution we apply (3.47) for $k = N, N - 1, \ldots, 0$, giving $\delta_{N-1}, \ldots, \delta_0, \delta_{-1}$ starting from $\delta_N = \delta_{N+1} = 0$. The relations can be simplified if we let

$$b_k = l_k \delta_{k-1} / c_0 = \delta_{k-1} / c_{k-1}. \tag{3.49}$$

Then (3.47) becomes

$$b_k = \gamma_k + (d_k + \alpha c_k) b_{k+1} + e_{k+1} b_{k+2} \tag{3.50}$$

for $k = N, N - 1, \ldots, 0$, with $b_{N+1} = b_{N+2} = 0$, and the required quotient coefficients are given by

$$\delta_k = c_k b_{k+1}, \qquad \text{for} \quad k = 0, \ldots, N - 1. \tag{3.51}$$

It is trivial to show that the remainder is

$$R = b_0. \tag{3.52}$$

We note that

$$S_N(\alpha) = R, \tag{3.53}$$

giving a convenient method of calculating the value of the expansion at $x = \alpha$. The recurrence (3.50) is a particularly stable one and has been discussed in a different context by Smith [13].

The derivative of $S_N(x)$ at $x = \alpha$ can also be evaluated as $dR/d\alpha$. Differentiating (3.50) gives

$$\frac{db_k}{d\alpha} = (d_k + \alpha c_k) \frac{db_{k+1}}{d\alpha} + e_{k+1} \frac{db_{k+2}}{d\alpha} + c_k b_{k+1}. \tag{3.54}$$

Application of (3.54), simultaneously with (3.50), for $k = N, N - 1, \ldots, 0$ with $db_{N+1}/d\alpha = db_{N+2}/d\alpha = 0$ gives

$$\frac{dR}{d\alpha} = \frac{db_0}{d\alpha}. \tag{3.55}$$

These results can be used to carry out stably a Newton iteration of $S_N(x)$ for a single root as referred to in Section 3.3. If $x^{(j)}$ is an approximation to the root, then the iteration sequence is

$$x^{(j+1)} = x^{(j)} - \frac{R}{dR/d\alpha} = x^{(j)} - \frac{b_0}{db_0/d\alpha}. \tag{3.56}$$

3.5 Calculation of the Quadrature Weights

Once the expansion $H_{n+m}(x)$ as defined by (3.1) has been calculated together with the optimally extended nodes (the roots of $E_m(x)$), the process of constructing the quadrature rule (1.1) can be completed in a simple manner. To calculate the weight A_i corresponding to any node x_i, we let

$$g(x) = \frac{H_{n+m}(x)}{x - x_i} = \sum_{i=0}^{n+m-1} \delta_i \phi_i \tag{3.57}$$

where the coefficients δ_i are computed using the procedure of Section 3.4 applied to $H_{n+m}(x)$. Since the rule must be interpolatory, we have

$$A_i = \int_a^b w(x) \frac{g(x)}{g(x_i)}\, dx = \frac{\delta_0 h_0}{g(x_i)} \tag{3.58}$$

due to orthogonality. The procedure of Section 3.4 can be repeated to divide $g(x)$, as given by (3.57), by $(x - x_i)$, yielding $g(x_i)$ as the remainder (see Eq. (3.52)). The weights A_i are now fully determined, assuming that the zero moment integral h_0 has been specified.

Although this computational procedure is numerically stable, there is an inherent potential for cancellation in extreme situations when some of the weights are effectively zero. The high order Gauss–Laguerre rules are a typical example. When this occurs, the computed small weights will be approximated by a number close to the smallest possible relative machine precision. In practical terms, the resulting rule will be identical within machine precision to that using the exact weights. The difficulty is caused by the absence of a Christoffel–Darboux identity (which would implicitly remove terms which cancel exactly) in the case of orthogonal systems for variable signed weight functions. To minimize the difficulty, the standard Christoffel–Darboux result is applied in the algorithmic implementation when $H_{n+m}(x)$ has only one term (which occurs, for example, when $n = 0$); thus

$$H_{n+m}(x) = \phi_{n+m}(x). \tag{3.59}$$

In this case the standard result gives immediately

$$A_i = \frac{c_{n+m-1} h_{n+m-1}}{\phi_{n+m-1}(x_i)\phi'_{n+m}(x_i)}. \tag{3.60}$$

The quantity $\phi'_{n+m}(x_i)$ is calculated using the recurrence (2.2) and its derivative. Careful scaling is required in calculating (3.60) for some weight functions to avoid overflow in the computer representation of the numbers.

3.6 Incorporating the Preassigned Nodes

The algorithm as described requires specification of the polynomial (2.5), whose roots are the preassigned nodes. This can be done either explicitly or by generating the coefficients τ_i indirectly from the values of the nodes. In the latter case we simply solve the linear system

$$\sum_{i=0}^{n-1} \left(\frac{\tau_i}{h_i}\right)\phi_i(x_k) = -\phi_n(x_k), \qquad \text{for} \quad k = 1, \ldots, n \tag{3.61}$$

with $\tau_n = 1$. In the algorithmic implementation this procedure can be carried out automatically if desired.

4. TESTS AND GENERAL COMMENTS

The algorithm has been tested on a number of classical weight functions including

$$\begin{array}{ll} (1)\ w(x) = 1 & x \in [-1, 1] \\ (2)\ w(x) = \sqrt{x} & x \in [0, 1] \\ (3)\ w(x) = e^{-x} & x \in [0, \infty) \\ (4)\ w(x) = e^{-x^2} & x \in (-\infty, \infty). \end{array} \tag{4.1}$$

A formal initial test was done to generate the corresponding classical quadrature rule by preassigning zero nodes and calculating m optimal nodes to give a rule of degree $2m - 1$. The Gauss–Radau rules were formed by choosing -1 as a preassigned node. For the Gauss–Lobatto rules, -1 and 1 were preassigned. In these cases the polynomial (2.5) was generated as described in Section 3.6. Further tests involved the generation of sequences of rules by iteratively extending particular sets of classical nodes. As is well known, some of these sequences are short lived in that imaginary nodes soon appear. The Gauss–Legendre, Gauss–Lobatto, and Gauss–Radau generally produce many extended sequences. The Gauss–Laguerre and Gauss–Hermite are unfruitful in this respect, as can be verified theoretically [5].

In most of the tests carried out, the algorithm performed to expectations and produced accurate results. Of course, many excellent algorithms exist for generating *single* Gaussian type rules (e.g., [3]) which would be expected to perform more efficiently and robustly. However, the present algorithm is designed to generate sequences of extended rules for *any* weight function for which a 3-term recurrence relation can be specified, and thus it would be surprising if difficulties did not appear on occasion. The most critical section is the calculation of the roots of the extended polynomial $E_m(x)$. No general expression based on the recurrence relation (2.2) is available for accurately approximating the zeros, and polynomial deflation must be used to prevent spurious convergence to those already computed. Failures must ultimately occur when high-order rules are being constructed (the limiting order depending on the particular weight function) and is generally attributable to an accumulation of errors in the calculation of the deflated polynomials.

For example, in the case of Gaussian rules based on (1)–(4), the maximum number of nodes in the rules which could be reliably generated using double-precision arithmetic were, respectively, 94, 46, 51, and 66. For the Lobatto and

Radau rules, the corresponding respective numbers were 94 and 91. (The values of the machine parameters (β, t, L, U), using the notation of Golub and Van Loan [2], were (2, 56, −127, 127).) In such circumstances it is frequently found that the direct application of the root solver to the full expansion, with careful user guidance on good initial root approximations, allows highly accurate nodes to be obtained. Although the user must then become more intimately involved with the orchestration of the calculations, and the convenience of automatic calculation is forgone, the range of application of the algorithm can be enhanced. However, for the highest order rules it is difficult to avoid ultimately using multiple precision arithmetic (as adopted in the well-known Stroud and Secrest [17] calculations). Indeed, in principle, one would prefer that the precision used to compute the rules should exceed the precision under which they are to be applied in practice.

REFERENCES

1. GOLUB, G. H., AND ROBERTSON, T. N. A generalized Bairstow algorithm. *Commun. ACM 10*, 6 (June 1967), 371–373.
2. GOLUB, G. H., AND VAN LOAN, C. F. *Matrix Computations*. Oxford University Press, New York, 1983.
3. GOLUB, G. H., AND WELSCH, J. H. Calculation of Gauss quadrature rules. *Math. Comput. 22* (1969), 221–230.
4. GRADSHTEYN, I. S., AND RYZHIK, I. M. *Tables of Integrals, Series and Products*. Translated from the Russian by Alan Jeffrey. Academic Press, Orlando, Fla., 1965.
5. KAHANER, D. K., AND MONEGATO, G. Nonexistence of extended Gauss–Laguerre and Gauss–Hermite quadrature rules with positive weights. *Z. Angew. Math. Phys. 29*, (1978), 983–986.
6. KRONROD, A. S. *Nodes and Weights for Quadrature Formulae. Sixteen Place Tables*. Nauka, Moscow, 1964; English translation by the Consultants Bureau, New York, 1965.
7. KRYLOV, V. I. *Approximate Calculation of Integrals*. Translated from the first Russian edition, 1959, by A. H. Stroud. Macmillan, New York, 1962.
8. MONEGATO, G. A note on extended Gaussian quadrature rules. *Math. Comput. 30*, (1976), 812–817.
9. MONEGATO, G. Stieltjes polynomials and related quadrature rules. *SIAM Rev. 24*, (1982), 137–158.
10. PATTERSON, T. N. L. The optimal addition of points to quadrature formulae. *Math. Comput. 22*, (1968), 847–856, and On some Gauss and Lobatto based quadrature formulae, *ibid.*, *22*, (1968), 877–881.
11. PATTERSON, T. N. L. Algorithm 468—Algorithm for automatic numerical integration over a finite interval. *Commun. ACM 16*, 11 (Nov. 1973), 694–699.
12. PIESSENS, R., AND BRANDERS, M. A note on the optimal addition of abscissas to quadrature formulas of the Gauss and Lobatto type. *Math. Comput. 28*, (1974), 135–139.
13. SMITH, F. J. An algorithm for summing orthogonal polynomial series and their derivatives with application to curve-fitting and interpolation. *Math. Comput. 19*, (1965), 33–36.
14. STIELTJES, T. J. *Correspondance d'Hermite et de Stieltjes*, Vol. II. Gauthier-Villars, Paris, 1905, 439–441.
15. STROUD, A. H., AND SECREST, D. *Gaussian Quadrature Rules*. Prentice Hall, Englewood Cliffs, N.J., 1966.
16. STRUBLE, G. W. Orthogonal polynomials: Variable-signed weight functions. *Numer. Math. 5*, (1963), 88–94.
17. SZEGÖ, G. Uber gewisse orthogonale Polynome, die zu einer oszillierenden Belegungsfunktion gehören. *Math. Ann. 110*, (1934), 501–513.

Received May 1987; revised December 1987; accepted October 1988