Numerical Integration Using Rys Polynomials

HARRY F. KING AND MICHEL DUPUIS

Department of Chemistry, State University of New York, Buffalo, New York, 14214

Received August 11, 1975; revised October 27, 1975

We define and discuss the properties of manifolds of polynomials $J_n(t, x)$ and $R_n(t, x)$, called Rys polynomials, which are orthonormal with respect to the weighting factor $\exp(-xt^2)$ on a finite interval of t. Numerical quadrature based on Rys polynomials provides an alternative approach to the computation of integrals commonly encountered in molecular quantum mechanics. This gives rise to a curve fitting problem for the roots and quadrature weights as a function of the x parameter. We have used Chebyshev approximation for small x and an asymptotic expansion for large x. A modified Christoffel–Darboux equation applicable to Rys polynomials is derived and used to obtain alternative formulas for Rys quadrature weight factors.

I. INTRODUCTION

The definite integral

$$F_n(x) = \int_0^1 t^{2n} \exp(-xt^2)\, dt, \qquad n = 0, 1, 2... \tag{1}$$

is commonly encountered in molecular quantum mechanical calculations using Gaussian basis functions [1]. A linear combination of $F_m(x)$ all with the same x can be written as a single integral

$$I_n(x) = \sum_{m=0}^{n} c_m F_m(x) = \int_0^1 f_n(t) \exp(-xt^2)\, dt, \tag{2}$$

where $f_n(t)$ is an even polynomial of order $2n$ with coefficients c_m. Our purpose in this paper is to define the system of orthogonal polynomials that leads to an exact quadrature formula for $I_n(x)$, to record some properties of these polynomials, and to discuss the efficient computation of polynomial roots and associated quadrature weight factors.

$F_n(x)$ can be re-expressed in terms of the error function or the incomplete gamma function, but for reasonably small values of the argument it is readily evaluated by a nonalternating power series expansion in x as in Shavitt [1, Eqs. (25)–(30)], or

Copyright © 1976 by Academic Press, Inc.
All rights of reproduction in any form reserved.

by obtaining a polynomial approximation to $F_n(x)$ valid over a specified range of x values. Given $f_n(t)$ explicitly, i.e., given values of c_m, a computationally stable procedure for the evaluation of $I_n(x)$ is to first obtain $F_n(x)$, then generate the other F_m by downward recursion using

$$(2m - 1) F_{m-1}(x) = 2xF_m(x) + \exp(-x), \tag{3}$$

followed by the summation in (2). This is, essentially, the method followed in all standard Gaussian integral programs [2]. The motivation behind the present investigation is to bypass entirely the computation of c_m, which in typical quantum chemical applications is the really difficult part of the computation in terms of both programming effort and machine execution time. This strategy requires a complete reorganization of existing quantum chemical programs. Instead of manipulating algebraic expressions for c_m, we work with numerical values of the integrand in (2) evaluated at the zeros, t_α, of a Rys polynomial. In place of the usual subroutine for computing $F_n(x)$, this method requires one for the evaluation of $t_\alpha(x)$ and its associated weight factor. We defer to a later paper a discussion of the relative merits of the two approaches.

Here we attempt to establish the underlying mathematical relationships and computational methods. Section II defines the Rys polynomials, useful properties of which are given in Section III. The bulk of the paper, Section IV, is concerned with practical evaluation of roots and weights. This is done numerically in three steps. First, these functions of x are computed to high accuracy at a number of points. Then we attempt to fit these results by low-order polynomial approximations valid over finite intervals of x, and by asymptotic expansions for large x. The resulting values of the parameters in the various fitting functions are then stored once and for all as constants in a set of efficient subroutines. Finally, Section V records a number of analytical relationships between roots and weights and how they vary with respect to the x parameter.

II. Rys Polynomials

Let $p_n(t)$ denote a polynomial of order n. A system of polynomials is said to be orthogonal on the interval (a, b) with respect to the weight function $w(t)$, if [3]

$$\int_a^b w(t)\, p_n(t)\, p_m(t)\, dt = h_n \delta_{nm}. \tag{4}$$

In particular, they are orthonormal if $h_n = 1$. Familiar examples are the Legendre polynomials $P_n(t)$, for which $a = -1$, $b = 1$, $w = 1$, $h_n = (n + 1/2)^{-1}$, and

Hermite polynomials $H_n(t)$, for which $a = -\infty$, $b = \infty$, $w = \exp(-t^2)$, and $h_n = \pi^{1/2}2^n n!$.

Consider a manifold of polynomials, $J_n(t, x)$ orthonormal on the interval $-1 < t < 1$ or, alternatively, $R_n(t, x)$ orthonormal on (0, 1), both with respect to the weight function

$$w(t, x) = \exp(-xt^2), \tag{5}$$

where x is a real parameter. The fact that w is everywhere positive is sufficient to assure existence and uniqueness [3]. When the value of x is obvious from context, we simplify the notation to $J_n(t)$ or J_n. The R_n are chosen to be even polynomials of order $2n$. They are simply proportional to the even members of J_n.

$$R_n(t, x) = 2^{1/2}J_{2n}(t, x). \tag{6}$$

In recognition of the contributions of a colleague we call these Rys polynomials, or to be specific, J-type or R-type Rys polynomials.

In the limit $x \to 0$, J_n becomes a Legendre polynomial.

$$J_n(t, 0) = (n + 1/2)^{1/2} P_n(t). \tag{7}$$

In the limit of large x, J_n becomes a scaled Hermite polynomial in the sense discussed below. Replace $p_n(t)$ by $J_n(t, x)$ in (4) and make the substitutions $b = x^{1/2}$ and $u = bt$. It follows that

$$b^{-1} \int_{-b}^{b} J_n(u/b, x)\, J_m(u/b, x) \exp(-u^2)\, du = \delta_{nm}\,. \tag{8}$$

As b increases, (8) approaches the defining equation for normalized Hermite polynomials. Further analysis of (8) leads to the conclusion that the following function, which appears in the integrand of (8), converges uniformly to a Hermite function.

$$\lim_{x\to\infty} [x^{-1/4}J_n(x^{-1/2}u, x) \exp(-u^2/2)] = \pi^{-1/4}(2^n n!)^{-1/2} H_n(u) \exp(-u^2/2). \tag{9}$$

In particular, the zeros of the function on the left approach those of the Hermite polynomial as x increases. Later we make use of this result in computing the roots of $R_n(t, x)$.

R_n is an even polynomial in t

$$R_n(t, x) = \sum_{k=0}^{n} C_{kn}(x)\, t^{2k}. \tag{10}$$

For any fixed x value R_n is orthogonal to all other R_m and so is orthogonal to t^{2m} for $m < n$. Substitution of (10) into the orthogonality relation gives

$$C_{mm} \sum_{k=0}^{n} C_{kn} F_{m+k} = \delta_{mn}, \qquad m \leqslant n. \tag{11}$$

In matrix notation this becomes

$$\mathbf{C}^{\dagger}\mathbf{F}\mathbf{C} = \mathbf{I}, \tag{12}$$

where the elements of $\mathbf{C}$ below the diagonal are zero by definition and the elements of $\mathbf{F}$ are

$$(\mathbf{F})_{mk} = F_{m+k-2}(x). \tag{13}$$

Solving for the coefficients of the lowest two R-type polynomials gives

$$R_0 = F_0^{-1/2} \tag{14}$$

and

$$R_1 = (F_0 t^2 - F_1)[F_0(F_0F_2 - F_1^2)]^{-1/2}. \tag{15}$$

These are special cases of Szegö [3, Eq. (2.2.6)] which gives $R_n(t)$ as the determinant

$$R_n = \lambda_n \begin{vmatrix} F_0 & F_1 & \cdots & F_{n-1} & 1 \\ F_1 & F_2 & \cdots & F_n & t^2 \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ F_n & F_{n+1} & \cdots & F_{2n-1} & t^{2n} \end{vmatrix}, \tag{16a}$$

with normalization factor

$$\lambda_n = (|\mathbf{F}^n|\,|\mathbf{F}^{n-1}|)^{-1/2}, \tag{16b}$$

where $|\mathbf{F}^n|$ is the determinant of the finite, square $\mathbf{F}$ matrix of order $n + 1$.

Figure 1 shows R_3 for three different x values. Note the characteristic shift of the three nodes toward smaller t values as x increases. This shift is predicted in the treatise by Szegö [3, Theorem 6.12.2]. Shown by a broken line is the scaled Hermite polynomial which, according to (6) and (9), approximates $R_3(t, 10)$. The corresponding Hermite approximations are seriously in error for the lower x values, and are not shown.

Finally, we point out that Rys polynomials, like all orthogonal polynomials, multiply accord to

$$R_i R_j = \sum_k b_{ijk} R_k, \tag{17}$$

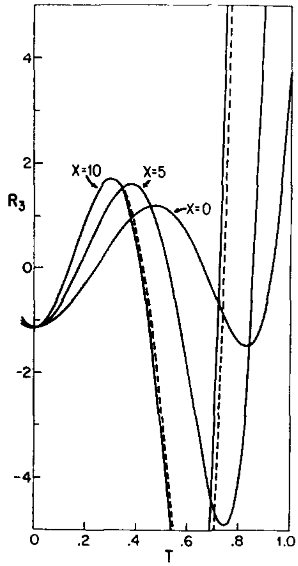

FIG. 1. Rys polynomial $R_3(t, x)$. The broken curve is the scaled Hermite polynomial proportional to $H_6(10^{1/2}t)$.

where b_{ijk} is invariant with respect to permutation of its indices

$$b_{ijk} = \int_0^1 R_i R_j R_k w \, dt, \tag{18}$$

and b_{ijk} is nonzero only if it its indices satisfy the "triangle inequality," i.e., if no index is greater than the sum of the other two. Because $R_i R_j$ is obviously a polynomial of order $i + j$ in the variable t^2, it follows that $b_{ijk} \neq 0$ when $k = i + j$.

III. Summation Orthogonality

The relationship between orthogonal polynomials and quadratures is a classical area of analysis [3–4]. We recapitulate the portion that pertains to the evaluation of the integral in (2). The central theorem is that Rys polynomials are orthogonal under summation,

$$\sum_{\alpha=1}^{n} R_i(t_\alpha, x) \, R_j(t_\alpha, x) \, W_\alpha(x) = \delta_{ij}, \tag{19}$$

where $2n > i + j$, $t_\alpha(x)$ is a positive root of R_n, and the W_α are appropriate quadrature factors. The fact that $w > 0$ assures that $W_\alpha > 0$. Another way to view these properties is to recognize that for any given n and x the quantity

$$U_{\beta\alpha} = R_{\beta-1}(t_\alpha)\, W_\alpha^{1/2} \tag{20}$$

can be regarded as an element of an n by n real unitary matrix $\mathbf{U}(x)$. This leads to a number of useful relationships. For example, from the diagonal elements of $\mathbf{U}^\dagger\mathbf{U}$ one obtains a Christoffel formula for the quadrature factors.

$$W_\alpha^{-1} = \sum_{i=0}^{n-1} R_i(t_\alpha)^2. \tag{21}$$

Proof of the assertions above is not difficult and can be found (paraphrased) in texts on orthogonal polynomials. [3–5]. Readers familiar with (27) given below can regard (19) as a quadrature formula for $R_i(t)\, R_j(t)$. Perhaps it is helpful to others to point out that if (19) is shown to be true in the special case $i = 0$, then the general case follows immediately from (17). The special case reduces to

$$\sum_\alpha R_j(t_\alpha)\, W_\alpha = 0, \tag{22}$$

when $0 < j < 2n$. One can construct a nonnull vector $\mathbf{W}$ with elements W_α to satisfy (22) for $0 < j < n$ because there always exists an n-dimensional vector orthogonal to $n - 1$ specified vectors. By induction one then shows that (22) is satisfied for $0 < j < 2n$ by substitution of $R_n(t_\alpha)\, R_{j-n}(t_\alpha)$ into (17). The W_α can be scaled according to

$$\sum_\alpha W_\alpha = F_0\,, \tag{23}$$

which satisfies (19) when $i = j = 0$.

Finally, we wish to record some useful corollaries of (19). If $f_m(t)$ is an even polynomial of order $2m < 4n$ there exist coefficients a_i such that

$$f_m(t) = \sum_{i=0}^{m} a_i(x)\, R_i(t, x) \tag{24}$$

$$a_i = \int_0^1 f_m(t)\, R_i(t)\, w(t)\, dt, \tag{25}$$

$$a_i = \sum_{\alpha=1}^{n} f_m(t_\alpha)\, R_i(t_\alpha)\, W_\alpha\,, \tag{26}$$

where t_α is a root of R_n. In particular, when $i = 0$ Eqs. (25)–(26) give the n-point Rys formula which is exact for f_m

$$a_0/R_0 = I_m(x) = \sum_{\alpha=1}^{n} f_m(t_\alpha)\, W_\alpha(x). \tag{27}$$

IV. Rys Roots and Weights

Because of the functional form of $f_n(t)$ commonly encountered in molecular quantum mechanics, we prefer to evaluate directly the quantity u_α which is related to the root of an R-type Rys polynomial according to

$$u_\alpha = t_\alpha^{\,2}/(1 - t_\alpha^{\,2}). \tag{28}$$

Note that there is negligible round off error in obtaining t_α from a given u_α even for t_α close to unity. Practical applications require that $u_\alpha(x)$ and $W_\alpha(x)$ be computed accurately and efficiently for any given n and x. We have computed many $u_\alpha(x)$

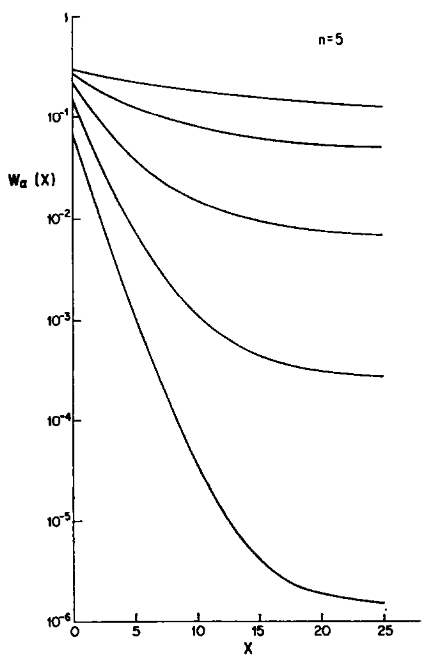

FIG. 2. Quadrature weight factors for five-point formula based on the roots of $R_5(t, x)$. Note that W_5 varies by five powers of 10 over this x interval.

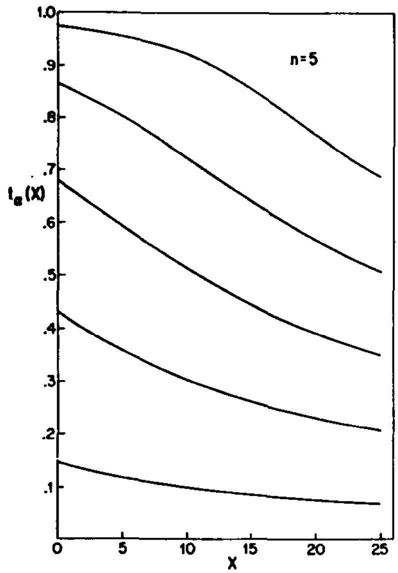

FIG. 3. Zeroes of $R_5(t, x)$.

TABLE I

Rys Roots and Weights for $n = 5$

x	α	$t_\alpha(x)$				$W_\alpha(x)^a$			
0.0	1	0.14887	43389	81631	21088	0.29552	42247	14752	87017(0)
0.0	2	0.43339	53941	29247	19080	0.26926	67193	09996	35509(0)
0.0	3	0.67940	95682	99024	40623	0.21908	63625	15982	04400(0)
0.0	4	0.86506	33666	88984	51073	0.14945	13491	50580	59315(0)
0.0	5	0.97390	65285	17171	72008	0.66671	34430	86881	37594(1)
5.0	1	0.12061	64790	67479	80274	0.22404	70675	36327	28532(0)
5.0	2	0.35999	36089	78937	69402	0.12394	61261	90236	70786(0)
5.0	3	0.59183	18425	24405	76114	0.38986	36709	17377	18353(1)
5.0	4	0.80234	18319	01655	82444	0.76362	12053	30385	45742(2)
5.0	5	0.95672	72697	52466	32751	0.10965	36738	90797	59497(2)
10.0	1	0.10123	93950	75509	96865	0.18293	17078	96803	42357(0)
10.0	2	0.30488	75738	73516	11244	0.80914	97211	11935	30740(1)
10.0	3	0.51182	43809	11261	54433	0.15223	89283	14807	32218(1)
10.0	4	0.72225	74026	79194	15721	0.11413	65256	49064	74907(2)
10.0	5	0.92103	91128	95102	92651	0.35452	41067	44066	12318(4)

[a] Number in parenthesis is negative power of 10.

and $W_\alpha(x)$ functions and have tried to represent them using Chebyshev expansion and other curve fitting techniques. We start with a FORTRAN program, DUBROOT, which was not written with efficiency in mind, but which is general and achieves an accuracy of about 25 significant figures when run in double precision on a CDC 6400.

For a specified n and x, DUBROOT evaluates the **F** matrix using Shavitt [1, Eqs. (25)–(30)] for F_{2n} followed by downward recursion using (3). Then (12) is solved by Schmidt orthogonalization. The t_α are obtained by root search, and the W_α from (21). Results for $n = 5$ are displayed in Figs. 2 and 3 and a few accurate values are reported in Table I. Note that roots are ordered so that $t_{\alpha+1} > t_\alpha > 0$.

Large X Approximation

Equations (6) and (9) imply that, as x becomes very large, the roots of J_{2n} and R_n decrease as $x^{-1/2}$, i.e.

$$\lim_{x\to\infty} [x^{1/2}t_\alpha(x)] = r_{\alpha n}\,, \tag{29}$$

where $r_{\alpha n}$ is a positive root of H_{2n}. Similarly

$$\lim_{x\to\infty} [x^{1/2}W_\alpha(x)] = w_{\alpha n}\,, \tag{30}$$

where $w_{\alpha n}$ is the corresponding weight factor for the $2n$-point Gauss–Hermite quadrature formula. For sufficiently large x, $t_\alpha \approx r_{\alpha n}x^{-1/2}$. To avoid a spurious singularity in u, this approximation should not be used unless x is at least greater than $r_{\alpha n}^2$. For example, this minimum x value is 2.7, 11.8, and 29.0 for $\alpha = n = 2$, 5, and 10 respectively. We show later that x has to be roughly three times greater than r_{nn}^2 before a "large x expansion" becomes very useful.

As a further guide to the behavior of these functions at large x consider the case $n = 1$. Then

$$t_\alpha = (F_1/F_0)^{1/2}, \tag{31}$$

$$W_\alpha = F_0\,, \tag{32}$$

and

$$u_\alpha = F_1/(F_0 - F_1). \tag{33}$$

Use the asymptotic expansion for F_m [1]

$$F_0(x) = 1/2(\pi/x)^{1/2} - e^{-x}[(2x)^{-1} - (2x)^{-2} + 3(2x)^{-3} \cdots], \tag{34}$$

and

$$F_1(x) = (4x)^{-1}(\pi/x)^{1/2} - e^{-x}[(2x)^{-1} + (2x)^{-2} - (2x)^{-3} + \cdots]. \tag{35}$$

If one drops the exponential terms in (34)–(35), he recovers the limiting expressions (29)–(30). This is a general result, for all n, so let

$$W_\alpha = x^{-1/2} w_{\alpha n} + e^{-x} Q_w , \tag{36}$$

and

$$u_\alpha = r_{\alpha n}^2/(x - r_{\alpha n}^2) + e^{-x} Q_u . \tag{37}$$

In the case $n = 1$ the Q-type correction factors are

$$Q_w = -(2x)^{-1} + (2x)^{-2} - 3(2x)^{-3} + \cdots, \tag{38}$$

and

$$Q_u = -(x/\pi)^{1/2}[x^{-1} + x^{-2} + 6(2x)^{-3} + \cdots] + O(e^{-x}). \tag{39}$$

The analysis for higher n is difficult, but it shows that in the limit $x \to \infty$ the leading term in Q is x^s, where s is an integer or half integer that increases with n. This suggests plotting $\log Q$ versus $\log x$. Figure 4 shows some results for $n = 3$ obtained using accurate numerical values of u_α and W_α. The slopes of the lines

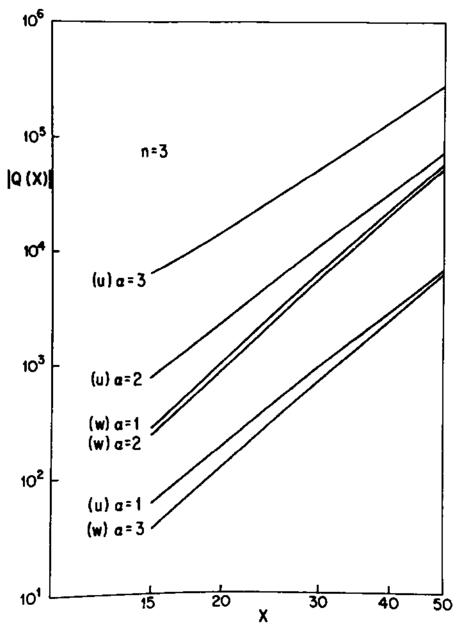

FIG. 4. Q-type correction factors for use in Eqs. (36)–(37). In the case $n = 3$, all Q_u are negative. Q_w is positive except when $\alpha = 1$.

are slowly varying over the important interval of x and generally are not close to integer or half integer values. We tried fitting $s(x)$ by a polynomial, but found that, with usually one extra term, we could do as well fitting Q by a sum of integer powers of x. We are interested in $Q(x)$ only for x values such that Qe^{-x} is small but not negligible compared with the W_α or u_α function being approximated. Table II gives the value of x that satisfies

$$Q_w e^{-x} = \gamma_w W_\alpha \tag{40}$$

for four values of γ_w. If, for example, one requires W_1 to 15 significant figures for $n = 5$ then Table II tells us that Q_w is needed to only five significant figures when

TABLE II

Value of x Corresponding to a Specified Value of γ_w

		$\gamma_w = W_\alpha^{-1} e^{-x} \mid Q_w \mid$			
n	α	10^{-1}	10^{-5}	10^{-10}	10^{-15}
1	1	1.4	9.8	20.9	32.2
2	1	2.9	15.8	28.8	41.2
2	2	6.4	18.4	31.2	43.6
3	1	4.5	20.4	34.4	47.4
3	3	11.1	24.2	37.8	50.7
4	1	6.2	24.6	39.6	53.2
4	4	15.4	29.4	43.6	57.0
5	1	7.8	29.0	44.6	58.8
5	5	19.6	34.4	49.2	63.0
6	1	9.5	33.2	49.6	64.0
6	6	23.8	39.2	54.6	68.6
7	1	11.1	37.4	54.4	69.4
7	7	28.2	44.1	59.9	74.4
8	1	12.8	41.5	59.2	74.5
8	8	32.3	48.8	65.1	79.9
9	1	14.4	45.6	63.8	79.5
9	9	36.5	53.5	70.1	85.2
10	1	16.0	49.6	68.5	84.5
10	10	40.6	58.1	75.1	90.6

TABLE III

Value of x Corresponding to a Specified Value of γ_u

		$\gamma_u = u_\alpha^{-1} e^{-x} \mid Q_u \mid$			
n	α	10^{-1}	10^{-5}	10^{-10}	10^{-15}
1	1	3.3	13.0	24.8	36.5
2	1	5.2	17.4	30.4	42.8
2	2	6.5	17.8	30.6	43.0
3	1	7.2	21.8	35.8	48.8
3	3	10.0	22.6	36.2	49.0
4	1	9.3	26.2	40.8	54.5
4	4	13.7	27.2	41.4	54.8
5	1	11.4	30.4	45.8	59.8
5	5	17.2	31.6	46.6	60.4
6	1	13.5	34.6	50.8	65.2
6	6	21.0	36.2	51.6	65.8
7	1	15.7	38.8	55.6	70.3
7	7	24.8	40.7	56.7	71.2
8	1	17.8	43.0	60.4	75.6
8	8	28.6	45.2	61.6	76.4
9	1	19.9	47.1	65.0	80.6
9	9	32.4	49.5	66.4	81.6
10	1	22.1	51.2	69.7	85.6
10	10	36.2	53.9	71.2	86.6

$x > 44.6$; and when $x > 58.8$ the Q factor is not needed at all. For each n only the maximum and minimum α values are included in Table II; the intermediate x_α values are bracketed by those given. Similarly, Table III gives this information for u_α. Note that for a given γ and a given $n > 2$ the values of x_α in Table III are bracketed by the corresponding x_1 and x_n in Table II. In other words, these tables indicate that of all the u_α and W_α for a given n, the approach to the "large x limit" is most rapid for W_1 and slowest for W_n.

We fit each W_α and u_α individually using (36) or (37) and

$$Q(x) \cong \bar{Q} = \sum A_k(x/x_{\min})^k, \tag{41}$$

where k takes on all integer values from $k_{\min}$ to $k_{\max}$. Note that computer time for the evaluation of (41) would not be reduced significantly by eliminating some intermediate k values. The optimal range for k depends upon: the interval of x for which the approximation is to be used, the desired accuracy, the value of n and, to a lesser extent, upon α. The best fit usually contains some positive and some negative powers of x. We want to minimize the maximum percent error in the object function, $G = W(x)$ or $G = u(x)$, over the interval $x_{\min}$ to $x_{\max}$. This means that we can tolerate a much larger error in Q near the upper end of the interval than near $x_{\min}$. Our procedure has been to generate a list of points $x_{\min} < x_1, x_2, \ldots, x_N < x_{\max}$ with the density of points being greatest at the lower end of the interval. Typically, $N = 100$. With a little experience we can assure that there are several points between each pair of nodes in the error function $G - \bar{G}$. The coefficients A_k are varied by a Newton–Raphson method to minimize

$$v_p = \sum_j [(G_j - \bar{G}_j)/G_j]^{2p}, \tag{42}$$

where $\bar{G}_j = \bar{G}(x_j)$ is the approximation using (41) and either (36) or (37). When $p = \infty$ we have truly minimized the maximum error. We find no important changes when we go beyond $p = 3$, so we first obtain an initial guess by solving

TABLE IV

Accurate Expansions of Q_g Using Equation (41)

n	α	g	$x_{\min}$	$x_{\max}$	$k_{\min}$	$k_{\max}$
2	1	u	15.0	45.0	−2	4
2	2	u	15.0	45.0	−2	3
3	1	u	15.0	20.0	−3	6
3	2	u	15.0	20.0	−3	5
3	3	u	15.0	20.0	−2	5
3	1–3	u	20.0	50.0	−1	4
4	1–4	u	20.0	50.0	−3	5
4	1–4	u	35.0	54.0	4	6
4	1–4	w	20.0	25.0	−1	7
4	1–4	w	25.0	35.0	0	6
4	1–4	w	35.0	54.0	4	6
5	1–5	u	25.0	40.0	0	8
5	1–5	u	40.0	59.0	3	6
5	1–5	w	25.0	40.0	0	9
5	1–5	w	40.0	59.0	6	8

TABLE V

Dependence of Accuracy on Choice of powers of x in Eq. (41) for the Expansion of Q_u in the Case $\alpha = 3$, $n = 4$, $x_{\min} = 20$, $x_{\max} = 50$[a]

No. terms	$k_{\min}$	$k_{\max}$	Accuracy
6	2	7	1.6×10^{-11}
7	−1	5	3.8×10^{-12}
7	0	6	2.7×10^{-12}
7	1	7	5.6×10^{-12}
8	−2	5	7.7×10^{-13}
8	−1	6	3.1×10^{-13}
8	0	7	2.7×10^{-13}
8	1	8	1.0×10^{-12}
9	−4	4	8.7×10^{-13}
9	−3	5	1.1×10^{-14}
9	−2	6	1.1×10^{-13}
9	−1	7	1.4×10^{-13}

[a] Accuracy is defined to be the maximum relative error in $u_3(x)$ over the specified x interval.

TABLE VI

Coefficients A_k, for Three of the Q_u Expansions Listed in Table V

k	7 terms	8 terms	9 terms
−3			−3 60330.8009
−2			26 66745.0661
−1			−86 26611.4368
0	89608.0562	1 68624.1120	160 47760.7410
1	−5 02825.5741	−9 82641.5417	−190 18848.3430
2	11 38724.1481	23 83919.6630	150 10447.9190
3	−13 89982.2603	−31 80032.9514	−80 17947.7993
4	10 43029.9687	25 82378.7123	29 46420.1272
5	−4 78047.3780	−12 69829.7610	−7 61331.5377
6	− 14203.0246	2 11324.2759	
7		− 27438.5727	

the weighted least squares problem ($p = 1$), and then refine the resulting **A** vector by minimizing v_4. This procedure has been carried out hundreds of times for various n, α, $x_{\min}$, $x_{\max}$, $k_{\min}$, and $k_{\max}$. (Each case takes several seconds on the CDC 6400.) Table IV reports the range of k and x values for $n < 6$. For all expansions reported in Table IV the approximation is accurate throughout the x interval to at least one part in 10^{13}. Reducing the expansion length by one term results, typically, in the loss of one significant figure in the accuracy of the approximation, but some experimentation is required in order to know which term to drop. For example, see Table V. Note that the best eight-term fit is a polynomial of order seven, but the best nine-term fit is a fifth-order polynomial with three terms in inverse powers of x. Coefficients for some of these examples are given in Table VI.

Small x Approximation

The x interval from zero up to where (36) or (37) becomes useful is broken up into several smaller ones. Within each we obtain polynomial approximations for $W_\alpha(x)$ and $u_\alpha(x)$. It is helpful to have exact Taylor series coefficients for comparison, and a few are given below:

$$n = 1$$

$$W_1(x) = 1 - x/3 + x^2/10 - x^3/42 + \cdots, \tag{43}$$

$$u_1(x) = 1/2 - x/5 + 8x^2/175 - 129x^3/28000 + \cdots, \tag{44}$$

$$n = 2$$

$$W_\alpha(x) = 2^{-1}(1 - r/18) - 6^{-1}(1 + 13r/270)\,x + 20^{-1}(1 + 503r/5346)\,x^2 \cdots, \tag{45}$$

$$u_\alpha(x) = 4^{-1}(6 + r) - 18^{-1}(6 + r)\,x + 693^{-1}(20 + 139r/45)\,x^2 - \cdots. \tag{46}$$

Here r is given by $r^2 = 30$ where $\alpha = 1$ or 2 corresponds to the negative or positive root respectively. It is well known that a finite Taylor series expansion is an excellent approximation to the object function in the immediate vicinity of a given point, but Chebyshev polynomial expansion tends to minimize the maximum error over a specified interval [6, 7].

Let $T_n(y)$ be the Chebyshev polynomial of order n on the interval $a < x < b$

$$T_n(y) = \cos(n\theta), \tag{47}$$

where y has the standard range $-1 < y < 1$,

$$y = \cos\theta = 2(x - x_0)/(b - a) \tag{48}$$

and x_0 is the center of the interval

$$x_0 = (a + b)/2. \tag{49}$$

We evaluate the object function $G(x)$ at the roots of $T_n(y)$, usually $n = 32$ or 48. Using standard methods based on the orthogonality of Chebyshev polynomials under summation [6, 7] the object function is approximated by a linear combination of $T_j(y)$ which fits G at each of the n points.

$$\bar{G} = \sum_{j=0}^{n-1} B_j T_j . \tag{50}$$

After the first few terms, successive B_j coefficients typically fall off by a factor of ten or more as shown in Table VII. The series (50) is truncated at the desired

TABLE VII

Chebyshev Polynomial Expansion of $u_\alpha(x)$ in the Vicinity of $x = 12.5$ in the Case $n = 5$, $\alpha = 3$[a]

j	$a = 10.0$, $b = 15.0$, B_j	$a = 11.25$, $b = 13.75$, B_j		$a = 10.0$, $b = 15.0$, $g_j(m = 11)$	$a = 11.25$, $b = 13.75$, $g_j(m = 8)$
0	2.98(01)[b]	2.95(01)	0	2.94150 68446 542(01)	2.94150 68446 542(01)
1	−5.38(02)	−2.67(02)	1	−2.13014 52165 350(02)	−2.13014 52165 514(02)
2	3.43(03)	8.51(04)	2	1.08648 98274 890(03)	1.08648 98275 510(03)
3	−1.72(04)	−2.15(05)	3	−4.39602 14734 500(05)	−4.39602 13339 000(05)
4	7.96(06)	5.01(07)	4	1.64329 07881 000(06)	1.64329 07200 000(06)
5	−1.39(07)	−4.26(09)	5	−2.21706 49400 000(08)	−2.21738 67000 000(08)
6	−1.02(08)	−1.59(10)	6	−1.33348 04400 000(09)	−1.33351 60000 000(09)
7	−4.52(10)	−4.13(12)	7	−5.81410 15000 000(11)	−5.53898 00000 000(11)
8	−6.77(12)	−3.89(14)	8	−9.02339 80000 000(13)	−8.34500 00000 000(13)
9	1.11(11)	2.26(14)	9	7.88430 80000 000(13)	
10	3.98(13)	4.49(16)	10	2.13926 00000 000(14)	
11	−5.89(14)	−2.81(17)	11	−2.52880 00000 000(15)	
12	−6.53(15)	−1.73(18)			
13	−1.13(16)	−2.19(20)			
14	5.26(17)				
15	5.55(18)				
16	−8.28(20)				

[a] The table illustrates to what extent convergence is improved by cutting the (a, b) interval in half. The coefficients are defined by Eqs. (50)–(51).

[b] Number in parenthesis is negative power of 10.

tolerance, which we choose to be one part in 10^{13}, and rearranged into a single polynomial of order m in the variable $x - x_0$,

$$\bar{G} = \sum_{j=0}^{m} g_j(x - x_0)^j. \tag{51}$$

The resulting g_j coefficients are also shown in Table VII. All computations up to this point are carried out in double precision. The g_j are then stored and used in single precision. Note that the maximum error in approximation (51) is less than the sum of neglected B coefficients and is thus very nearly equal to $| B_{m+1} |$. The value of g_m is a poor indication of the error.

From Table VII, one sees that reducing the size of the (a, b) interval by a factor of two does not reduce the number of terms in (51) by a comparable factor. Thus, small intervals are obviously more efficient in terms of computer time for evaluating $\bar{G}$, but also more costly in terms of the number of coefficients to be stored. Table VIII represents one compromise between these two considerations. The table lists the value of m in (51) for $n < 7$. For example, Table VIII tells us that a 15th order polynomial is required to approximate W_4 over the interval $x = 5$ to 10 to one part in 10^{13} when $n = 4$.

Miscellaneous Comments

A useful test of the various approximations is to compute the sum

$$\sum_{\alpha=1}^{n} t_\alpha^{2m} W_\alpha = F_m, \qquad m < 2n. \tag{52}$$

We computed roots and weights in single precision using (41) or (51) and compared the sum (52) with F_m computed in double precision for x at closedly spaced intervals for all $m < 2n$. The disagreement was never greater than four parts in 10^{13}. Incidently, the terms with high α dominate the sum (52) when m is large, even though their W_α values are small. Thus it is the per cent error not the absolute error in W_α that is important.

An exact quadrature formula analogous to (27) can be developed using the roots of J_n. The advantage would be that for odd n one root is at $t = 0$ for all x. For example, to compute $I_6(x)$ using (27) requires the computation of four roots of R_4 and associated weight factors whereas a formula based on the roots of J_7 would require computation of only three roots and four weights. The disadvantage is that, unlike the R_4 formula, the J_7 roots could not also be used for the exact computation of an $I_7(x)$ integral. In practice the slight advantage of the "odd n" formula does not seem sufficiently great to warrent storing coefficients for the additional polynomial approximations.

TABLE VIII

Degree m, of Chebyshev Polynomial Approximations to Rys $u_\alpha(x)$ and $W_\alpha(x)$ Functions[a]

			x interval						
n	α	func	0, 1	1, 5	5, 10	10, 15	15, 20	20, 25	25, 30
1	1	u	9	14	14	12			
1	1	w	9	13	13	12			
2	1	u	8	12	14	12			
2	2	u	8	13	13	12			
2	1	w	9	13	13	11			
2	2	w	9	14	14	13			
3	1	u	7	11	12	12	11		
3	2	u	7	11	12	12	11		
3	3	u	7	11	13	13	11		
3	1	w	8	12	12	12	10		
3	2	w	9	13	13	12	10		
3	3	w	9	14	15	14	12		
4	1	u	7	10	10	12	12		
4	2	u	6	10	10	12	12		
4	3	u	7	10	11	11	12		
4	4	u	7	10	12	12	12		
4	1	w	7	11	11	11	11		
4	2	w	9	13	13	11	12		
4	3	w	9	14	14	13	12		
4	4	w	10	15	15	14	13		
5	1	u	7	9	9	11	11	10	
5	2	u	7	10	10	11	11	11	
5	3	u	6	10	10	11	11	11	
5	4	u	5	9	11	11	12	11	
5	5	u	6	10	11	12	12	11	
5	1	w	7	10	10	10	11	10	
5	2	w	8	12	12	11	11	10	
5	3	w	9	13	13	12	11	10	
5	4	w	9	14	15	14	12	12	
5	5	w	10	15	16	15	14	13	
6	1	u	6	9	9	9	11	11	10
6	2	u	6	9	10	9	11	11	10
6	3	u	6	9	10	9	11	11	11
6	4	u	6	9	10	10	11	10	11
6	5	u	6	9	10	10	11	11	11
6	6	u	5	9	10	10	11	11	11
6	1	w	7	10	10	9	10	10	10
6	2	w	8	11	11	11	10	10	10
6	3	w	9	13	13	12	10	11	10
6	4	w	9	14	14	13	12	12	11
6	5	w	9	14	15	14	13	12	11
6	6	w	10	15	16	15	15	14	12

[a] All approximations are accurate to at least one part in 10^{13}.

V. Alternative Weight Formulas

For low n, special formulas can replace some of the approximations for the W_α and u_α discussed above. In the case $n = 1$ it is simpler to compute $F_1(x)$ and use (3), (31) and (32). For $n = 2$ the weights can be computed from the roots according to:

$$W_2 = ((F_1 - F_0)\, u_1 + F_1)(1 + u_2)/(u_2 - u_1), \tag{53}$$

and

$$W_1 = F_0 - W_2, \tag{54}$$

which follow from (52) and (28). Similar formulas apply when $n = 3$, and in all these cases round off error is less than one decimal figure.

We have searched for, but have not yet found, a generally useful formula for computing weights from roots for higher n. Equations analogous to (53), based on (52), become numerically unstable for higher n. Methods based on the Christoffel–Darboux equation have been employed for the classical orthogonal polynomials [3–5]. That analysis requires a slight modification for R-type Rys polynomials because of the omission of all the J_n of odd order. The appropriate analysis is presented below. Some important questions concerning computational accuracy remain to be investigated, so we presently recommend use of the W_α fitting functions discussed above. As can be seen from Table VIII, however, a suitable weight formula would save considerable computer time and core storage.

Since t^2 is a linear combination of R_0 and R_1 it follows from (17) that $t^2 R_n(t)$ is a linear combination of R_{n-1}, R_n, and R_{n+1}. This can be expressed as a recursion formula

$$\beta_{n+1} R_{n+1} = (t^2 - \beta_n')\, R_n - \beta_n R_{n-1}, \tag{55}$$

where

$$\beta_n = \int_0^1 t^2 R_n R_{n-1} w\, dt, \qquad \beta_0 = 0, \tag{56}$$

and

$$\beta_n' = \int_0^1 t^2 R_n^2 w\, dt. \tag{57}$$

These same coefficients arise in the differentiation of R_n with respect to the x parameter. Note that $(\partial R_n/\partial x)$ is an even polynomial of order $2n$ and so is a linear combination of R_k, $k \leqslant n$.

$$\frac{\partial R_n(t')}{\partial x} = \sum_{k=0}^{n} R_k(t') \int_0^1 R_k \left(\frac{\partial R_n}{\partial x}\right) w\, dt. \tag{58}$$

Note that

$$\frac{\partial}{\partial x}\int_0^1 R_k R_n w\, dt = 0, \tag{59}$$

so

$$0 = \int \left(\frac{\partial R_k}{\partial x}\right) R_n w\, dt + \int R_k \left(\frac{\partial R_n}{\partial x}\right) w\, dt + \int R_k R_n \left(\frac{\partial w}{\partial x}\right) dt. \tag{60}$$

The first term in (60) vanishes when $k < n$, so then (60) becomes

$$\int_0^1 R_k \left(\frac{\partial R_n}{\partial x}\right) w\, dt = \int_0^1 t^2 R_k R_n w\, dt, \qquad k < n. \tag{61}$$

The right side of (61) equals β_n when $k = n - 1$, and equals zero when $k < n - 1$. The first two terms in (60) are identical when $k = n$ in which case the third term is seen to be the negative of β_n'. It follows from (58)–(61) that

$$\partial R_n/\partial x = (1/2)\,\beta_n' R_n + \beta_n R_{n-1}\,. \tag{62}$$

From (55) one obtains

$$\begin{aligned}\beta_{n+1}&[R_{n+1}(t)\, R_n(\tau) - R_{n+1}(\tau)\, R_n(t)]\\ &= (t^2 - \tau^2)\, R_n(t)\, R_n(\tau) + \beta_n[R_n(t)\, R_{n-1}(\tau) - R_n(\tau)\, R_{n-1}(t)].\end{aligned} \tag{63}$$

Repeated application of (63) yields a modified form of the Christoffel–Darboux equation applicable to Rys polynomials.

$$\beta_{n+1}[R_{n+1}(t)\, R_n(\tau) - R_{n+1}(\tau)\, R_n(t)] = (t^2 - \tau^2) \sum_{k=0}^{n} R_k(t)\, R_k(\tau). \tag{64}$$

Let $t = t_\alpha$, i.e., a root of R_n, and let $\tau \to t$, then (64) becomes

$$\beta_{n+1} R_{n+1}(t_\alpha)\, R_n'(t_\alpha) = -2t_\alpha \sum_{k=0}^{n-1} R_k(t_\alpha)^2, \tag{65}$$

where

$$R_n' \equiv \partial R_n/\partial t. \tag{66}$$

From (55), (65), and (21), we obtain the weight formula

$$W_\alpha = 2t_\alpha[\beta_n R_{n-1}(t_\alpha)\, R_n'(t_\alpha)]^{-1}. \tag{67}$$

Since we know u_α as a function of x we can eliminate either R_{n-1} or R_n' from (67) using Eq. (70) derived below. First note that

$$R_n(t_\alpha) = 0. \tag{68}$$

Differentiate (68) with respect to x to obtain

$$R_n'(t_\alpha)(\partial t_\alpha/\partial x) + (\partial R_n/\partial x) = 0. \tag{69}$$

Substitution of (62) into (69) gives the useful relationship

$$R_n'(t_\alpha)(\partial t_\alpha/\partial x) + \beta_n R_{n-1}(t_\alpha) = 0. \tag{70}$$

Note from (28) that

$$t_\alpha^2 = u_\alpha(1 + u_\alpha)^{-1}, \tag{71}$$

and

$$2t_\alpha(\partial t_\alpha/\partial x) = u_\alpha'(1 + u_\alpha)^{-2}, \tag{72}$$

where u' is the derivative with respect to x. Equations (67), (70), and (72) yield the alternative weight formula

$$W_\alpha = -4u_\alpha(1 + u_\alpha)[R_n(t_\alpha)]^2/u_\alpha'. \tag{73}$$

Write R_n in terms of its roots

$$R_n(t) = C_{nn} \prod_{\beta=1,n} (t^2 - t_\beta^2). \tag{74}$$

Differentiate (74) with respect to t, substitute into (73) and use (71) to obtain

$$W_\alpha = -K_n \left[(1 + u_\alpha)^{n-1} \Big/ \prod_{\beta\neq\alpha} (u_\alpha - u_\beta)\right]^2 \Big/ u_\alpha', \tag{75}$$

where $K_n(x)$ is independent of α

$$K_n = \left[C_{nn}^{-1} \prod_\alpha (1 + u_\alpha)\right]^2. \tag{76}$$

Given appropriate fitting functions for $u_\alpha(x)$ for a specified n, one could calculate u_α and u_α' for a specified x. Equation (75) yields W_α to within an unknown scale factor. Finally, one would require a fitting function for $K_n(x)$ or, alternatively, one could compute $F_0(x)$ and scale the W_α so as to satisfy (23). No information about roots or weights for other values of n are required.

VI. Summary

Rys polynomials, like all sequences of polynomials orthogonal with respect to a positive weight factor, possess useful orthogonality properties with respect to summation over the roots of a higher Rys polynomial. Quadrature based on these properties provides a practical alternative to existing computational methods for a wide class of molecular integrals (2). Calculation of $u_\alpha(x)$ and $W_\alpha(x)$ replaces the traditional $F_m(x)$. Unfortunately no simple relationship analogous to the recursion relation (3) is known for the roots and weights, but satisfactory and highly accurate approximations have been developed. An improved formula for quadrature weight factors, however would be useful.

The fitting functions described in Section IV have been incorporated into subroutines that form part of programs to be submitted to the Quantum Chemistry Program Exchange [2]. Listings of these subroutines, DUBROOT, and any other program used in the present investigation can be obtained by writing to the authors.

References

1. I. Shavitt, *in* "Methods in Computational Physics," (B. Alder S. Fernbach, and M. Rotenburg, Eds.) Vol. 2, p. 1. Academic Press, New York/London, 1963.
2. For example, programs No. 47, 199, 236, and 241. Quantum chemistry program exchange, Bloomington, Indiana.
3. G. Szegö, "Orthogonal Polynomials, Amer. Math. Soc. New York, New York 1959.
4. A. H. Stroud and D. Secrest, "Gaussian Quadrature Formulas," Prentice Hall, Englewood Cliffs, New Jersey, 1966.
5. V. I. Krylov, "Approximate Calculation of Integrals," (A. H. Stroud, transl.), MacMillian, New York, 1962.
6. T. J. Rivilin, "The Chebyshev Polynomials," Wiley, New York, 1974.
7. C. Lanczos, *in* "Tables of Chebyshev Polynomials, Applied Math. Series 9," U.S. National Bureau of Standards, U.S. Government Printing Office, Washington, D. C., 1952.