INTERNATIONAL JOURNAL OF QUANTUM CHEMISTRY, VOL. 40, 745–752 (1991)

Two-Electron Repulsion Integrals Over Gaussian *s* Functions

PETER M.W. GILL, BENNY G. JOHNSON, AND JOHN A. POPLE
Department of Chemistry, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213

Abstract

We present an efficient scheme to evaluate the $[\mathbf{0}]^{(m)}$ integrals that arise in many ab initio quantum chemical two-electron integral algorithms. The total number of floating-point operations (FLOPs) required by the scheme has been carefully minimized, both for cases where multipole expansions of the integrals are admissable and for cases where this is not so. The algorithm is based on the use of a modified Chebyshev interpolation formula to compute the function $\exp(-T)$ and the integral $F_m(T) = \int_0^1 u^{2m}\exp(-Tu^2)\,du$ very cheaply.

Introduction

In retrospect, the 1986 paper [1] by Obara and Saika (OS) appears to have heralded a renaissance of interest in the efficient computation of the many two-electron repulsion integrals (ERIs) that are needed in most ab initio calculations of electronic structure. OS introduced an eight-term recurrence relation by which the ERIs between Gaussian basis functions of arbitrarily high angular momentum may be generated recursively from auxiliary *s*-type integrals, and in the years following, a number of algorithms [2–6] have been developed that successfully extend this approach by using other recurrence relations in addition to, or instead of, the original one.

The generation of ERIs using such algorithms involves two disjoint tasks. First, a small number of auxiliary *s*-type integrals ($[ss\,|\,ss]^{(m)}$ or $[\mathbf{0}]^{(m)}$) are formed, and, second, these are suitably combined together using the available recurrence relations. However, although considerable effort has recently focused on the optimization of the second of these tasks, the first has received scant attention. The predictable consequence of this neglect, as Hamilton and Schaefer have noted [3], is that the point has been reached at which the computation of integrals of comparatively low angular momentum, for example, highly contracted $(pp\,|\,pp)$, is now dominated by the formation of the requisite auxiliary *s*-type integrals.

In the present paper, we address this problem and describe a highly optimized scheme for computing $[\mathbf{0}]^{(m)}$ integrals.

A Statement of the Problem

Suppose that we have a primitive shell of Cartesian Gaussian functions on each of four centers **A**, **B**, **C**, and **D** and that their exponents and contraction co-

 CCC 0020-7608/91/060745-08$04.00

efficients are α, β, γ, and δ and D_A, D_B, D_C, and D_D, respectively. (For definitions of these and other terms, see [4]). Then, the fundamental integral in which we are interested is given by

$$[\mathbf{0}]^{(0)} = D_A D_B D_C D_D \iint e^{-\alpha|\mathbf{r}_1-\mathbf{A}|^2} e^{-\beta|\mathbf{r}_1-\mathbf{B}|^2} r_{12}^{-1} e^{-\gamma|\mathbf{r}_2-\mathbf{C}|^2} e^{-\delta|\mathbf{r}_2-\mathbf{D}|^2} d\mathbf{r}_1\, d\mathbf{r}_2 \tag{1}$$

where $r_{12} = |\mathbf{r}_1 - \mathbf{r}_2|$ and the six-dimensional integral is over all space.

First, we define three quantities associated with the shells on A and B:

$$\sigma_P = 1/(\alpha + \beta) \tag{2}$$

$$\mathbf{P} = (\alpha\mathbf{A} + \beta\mathbf{B})\sigma_P \tag{3}$$

$$U_P = (\pi\sigma_P)^{3/2} D_A D_B e^{-\alpha\beta\sigma_P|\mathbf{A}-\mathbf{B}|^2}, \tag{4}$$

and three associated with the shells on C and D:

$$\sigma_Q = 1/(\gamma + \delta) \tag{5}$$

$$\mathbf{Q} = (\gamma\mathbf{C} + \delta\mathbf{D})\sigma_Q \tag{6}$$

$$U_Q = (\pi\sigma_Q)^{3/2} D_C D_D e^{-\gamma\delta\sigma_Q|\mathbf{C}-\mathbf{D}|^2}. \tag{7}$$

It is important to note that all such shell-pair quantities can be evaluated within a comparatively cheap preliminary loop over all possible *pairs* of shells in the system. We therefore assume that all such quantities will have been calculated and stored before the main loop over all possible *quartets* of shells is entered.

Within the shell-quartet loop, we define four parameters:

$$R^2 = |\mathbf{P} - \mathbf{Q}|^2 \tag{8}$$

$$\vartheta^2 = 1/(\sigma_P + \sigma_Q) \tag{9}$$

$$T = \vartheta^2 R^2 \tag{10}$$

$$U = U_P U_Q \tag{11}$$

As Boys has shown [7], the $[\mathbf{0}]^{(0)}$ integral can be reduced to a much simpler one-dimensional integral related to the incomplete gamma function. In terms of (9)–(11), it may be expressed as follows:

$$[\mathbf{0}]^{(0)} = U(2\vartheta^2)^{1/2}\left(\frac{2}{\pi}\right)^{1/2}\int_0^1 e^{-Tu^2}\,du. \tag{12}$$

In most calculations, we must generalize this formulation to

$$[\mathbf{0}]^{(m)} = U(2\vartheta^2)^{m+1/2}\left(\frac{2}{\pi}\right)^{1/2}\int_0^1 u^{2m} e^{-Tu^2}\,du \tag{13}$$

and evaluate the set of $[\mathbf{0}]^{(m)}$ $(0 \le m \le L)$ for some given L. The computation of such sets should be as efficient as possible, since, in the course of a large direct ab initio calculation [8], it represents a major proportion of the total work performed.

An Efficient Scheme to Compute the $[\mathbf{0}]^{(m)}$

We first consider the shell-quartet parameters in (8)–(11). Of these, the most expensive to compute is R^2, which, if evaluated by a straightforward application of (8), will involve three subtractions, three multiplications, and two additions, i.e., eight floating-point operations (FLOPS). This cost can be reduced, however, if the quantities

$$\omega = \alpha|\mathbf{A} - \mathbf{B}|/(\alpha + \beta) \tag{14}$$

$$\mu = (\mathbf{A} - \mathbf{B}) \cdot (\mathbf{B} - \mathbf{Q})/|\mathbf{A} - \mathbf{B}| \tag{15}$$

$$\nu = |\mathbf{B} - \mathbf{Q}|^2 - \mu^2 \tag{16}$$

have been precomputed. It is easy to show that, in terms of these, R^2 becomes

$$R^2 = (\omega + \mu)^2 + \nu, \tag{17}$$

which costs only three FLOPS (two additions and a multiplication).

As (9) stands, ϑ^2 costs two FLOPS (a division and an addition), and this cannot be reduced. However, it will turn out later that a scaled version of ϑ^2 is more useful than ϑ^2 itself and, if the scaling has been precomputed, no extra cost is incurred; thus:

$$\frac{\vartheta^2}{2\Delta} = \frac{1}{2\Delta} \Big/ (\sigma_P + \sigma_Q), \tag{18}$$

where 2Δ is an interpolation interval to be defined later.

Similarly, we prefer (for reasons that will later become apparent) T to be likewise scaled. Again, this can be achieved for the same cost as (10):

$$\frac{T}{2\Delta} = \frac{\vartheta^2}{2\Delta} R^2. \tag{19}$$

If T is large enough, we can take advantage of the asymptotic formula

$$\left(\frac{2}{\pi}\right)^{1/2} \int_0^1 u^{2m} e^{-Tu^2} du \sim \frac{(2m-1)!!}{(2T)^{m+1/2}}, \tag{20}$$

which leads to the very simple result

$$[\mathbf{0}]^{(m)} \sim (2m-1)!! \frac{U}{R^{2m+1}}, \tag{21}$$

from which we see that (in the large-T limit) the $[\mathbf{0}]^{(m)}$ integrals are closely related to multipole interactions between the AB and CD shell-pairs.

For smaller values of T, we propose that an efficient way to evaluate the $[\mathbf{0}]^{(m)}$ is to rewrite (13) as

$$[\mathbf{0}]^{(m)} = (2m-1)!! U \left(\frac{\vartheta^2}{2\Delta}\right)^{m+1/2} G_m(T), \tag{22}$$

where

$$G_m(T) = \frac{(4\Delta)^{m+1/2}}{(2m-1)!!}\left(\frac{2}{\pi}\right)^{1/2}\int_0^1 u^{2m}e^{-Tu^2}du, \tag{23}$$

then to compute $G_L(T)$ using an efficient interpolation scheme and to calculate the remaining $G_m(T)$ $(0 \le m < L)$ using the downward recurrence relation

$$G_m(T) = \left(\frac{2}{\pi}\right)^{1/2}\frac{(4\Delta)^{m+1/2}}{(2m+1)!!}\exp(-T) + \left(\frac{T}{2\Delta}\right)G_{m+1}(T). \tag{24}$$

Of course, (24) uses $\exp(-T)$, and we propose that this, too, be found using an efficient interpolation scheme.

Our algorithm for $[\mathbf{0}]^{(m)}$ generation is now almost completely defined. It remains only to discuss the interpolation scheme that we have implemented.

Modified Chebyshev Interpolation

Suppose that we wish to develop interpolation formulae over some finite domain for a function f whose values and derivatives $f^{(m)}(x)$ $(m = 0,1,2,\ldots)$ are known at a set of points $x = X_j$. Ultimately, we seek an nth-degree polynomial in x:

$$A_n(x;j) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n, \tag{25}$$

to approximate $f(x)$ over the jth interpolation interval because (25) can be evaluated very efficiently (in just n adds and n multiplies) as

$$A_n(x) = a_0 + x(a_1 + x(a_2 + \ldots + xa_n)). \tag{26}$$

For convenience, we will focus on one interpolation interval $(X - \Delta, X + \Delta)$ and we will develop the theory in terms of the normalized variable:

$$t = \frac{x - X}{\Delta}. \tag{27}$$

Clearly, any nth-degree polynomial in t is equivalent to another in x.

The most obvious choice for $A_n(t)$ is the Taylor polynomial:

$$A_n^{Taylor}(t) = f(X) + (t\Delta)f^{(1)}(X) + \frac{(t\Delta)^2}{2!}f^{(2)}(X) + \ldots + \frac{(t\Delta)^n}{n!}f^{(n)}(X), \tag{28}$$

and McMurchie and Davidson [9], Harris [10], and Head-Gordon and Pople [2] have all advocated the use of Taylor interpolation to compute $F_m(T)$ [7]. However, although (28) is very accurate near the middle of the interpolation interval, it becomes much poorer near the endpoints and, for small Δ, its maximum error is

$$E_n^{Taylor} \approx \frac{\Delta^{n+1}}{(n+1)!}f^{(n+1)}(X). \tag{29}$$

There usually exist other nth-degree polynomials whose maximum error is less than this. The best of these, the "min-max" polynomial [11], is generally awk-

ward to compute, but the least-squares polynomial $A_n^{Cheby}(x)$, which minimizes the integral

$$I = \int_{-1}^{+1} \frac{[A_n^{Cheby}(t) - f(X + t\Delta)]^2}{(1 - t^2)^{1/2}}\, dt \tag{30}$$

is almost as good [11] and, as we now demonstrate, is easy to compute.

We expand $A_n(t)$ as a sum of Chebyshev polynomials:

$$A_n^{Cheby}(t) = a_0T_0(t) + a_1T_1(t) + \ldots + a_nT_n(t), \tag{31}$$

and then minimize (30) (using the orthogonality of the T_i) to obtain

$$a_k \int_{-1}^{+1} \frac{T_k^2(t)}{(1 - t^2)^{1/2}}\, dt = \int_{-1}^{+1} \frac{T_k(t)}{(1 - t^2)^{1/2}} f(X + t\Delta)\, dt. \tag{32}$$

The left integral is elementary. Assuming that the right integral cannot be evaluated in closed form, we expand $f(X + \Delta t)$ in series to obtain

$$a_k = \left(\frac{2 - \delta_{k0}}{\pi}\right) \int_{-1}^{+1} \frac{T_k(t)}{(1 - t^2)^{1/2}} \sum_{m=0}^{\infty} \frac{(t\Delta)^m}{n!} f^{(m)}(X)\, dt. \tag{33}$$

If we interchange the order of integration and summation, we obtain

$$a_k = \left(\frac{2 - \delta_{k0}}{\pi}\right) \sum_{m=0}^{\infty} \frac{\Delta^m f^{(m)}(X)}{m!} \int_{-1}^{+1} \frac{t^m T_k(t)}{(1 - t^2)^{1/2}}\, dt. \tag{34}$$

These "Chebyshev-moment" integrals vanish unless $m = k, k + 2, k + 4, \ldots,$ when

$$a_k = \left(\frac{2 - \delta_{k0}}{\pi}\right) \sum_{m=0}^{\infty} \frac{\Delta^{k+2m} f^{(k+2m)}(X)}{(k + 2m)!} \int_{-1}^{+1} \frac{t^{k+2m} T_k(t)}{(1 - t^2)^{1/2}}\, dt. \tag{35}$$

The nonvanishing integrals are well known:

$$\int_{-1}^{+1} \frac{t^{k+2m} T_k(t)}{(1 - t^2)^{1/2}}\, dt = \frac{\pi(k + 2m)!}{2^{k+2m} m!\,(k + m)!}, \tag{36}$$

and substitution of (36) into (35) yields a final expression for the a_k:

$$a_k = (2 - \delta_{k0}) \left(\frac{\Delta}{2}\right)^k \sum_{m=0}^{\infty} \left(\frac{\Delta}{2}\right)^{2m} \frac{f^{(k+2m)}(X)}{m!\,(k + m)!}, \tag{37}$$

which involves only the known quantities $f^{(m)}(X)$ and Δ.

For small Δ, the maximum error of (31) is

$$E_n^{Cheby} \approx a_{n+1} \approx \frac{\Delta^{n+1}}{2^n(n + 1)!} f^{(n+1)}(X). \tag{38}$$

Comparing (38) with (29) reveals that nth-degree Chebyshev interpolation is 2^n times more accurate than is nth-degree Taylor interpolation. Moreover, by rearranging (38), we obtain a formula for the greatest Δ that will give rise to a

Chebyshev interpolation error less than some tolerance ε:

$$\Delta = \left[\frac{2^n(n+1)!\,\varepsilon}{\max|f^{(n+1)}(X)|}\right]^{1/(n+1)}. \tag{39}$$

In our algorithm for computing $[\mathbf{0}]^{(m)}$ integrals, we choose the grid

$$X_j = (2j+1)\Delta \qquad (j = 0,1,\ldots), \tag{40}$$

so that the index j to the interpolation table (INT) can be found, very cheaply, as

$$j = \text{INT}\left(\frac{x}{2\Delta}\right). \tag{41}$$

As a compromise between expense and accuracy, we have chosen to use the *cubic* Chebyshev interpolation formula:

$$\begin{aligned} f(x) &\approx a_0T_0(t) + a_1T_1(t) + a_2T_2(t) + a_3T_3(t) \\ &= a_0 + a_1\left(\frac{x-X}{\Delta}\right) + a_2\left[2\left(\frac{x-X}{\Delta}\right)^2 - 1\right] + a_3\left[4\left(\frac{x-X}{\Delta}\right)^3 \right. \\ &\quad \left. - 3\left(\frac{x-X}{\Delta}\right)\right] \\ &= f_0 + \frac{x}{2\Delta}\left(f_1 + \frac{x}{2\Delta}\left(f_2 + \frac{x}{2\Delta}f_3\right)\right), \end{aligned} \tag{42}$$

to compute $G_m(T)$ and $\exp(-T)$. Thus, having formed a_0, a_1, a_2, and a_3 using (37), the f_i given by

$$f_0 = (a_0 - a_2) + (a_1 - 3a_3)\left(\frac{-X}{\Delta}\right) + 2a_2\left(\frac{-X}{\Delta}\right)^2 + 4a_3\left(\frac{-X}{\Delta}\right)^3 \tag{43}$$

$$f_1 = (2a_1 - 6a_3) + 8a_2\left(\frac{-X}{\Delta}\right) + 24a_3\left(\frac{-X}{\Delta}\right)^2 \tag{44}$$

$$f_2 = 8a_2 + 48a_3\left(\frac{-X}{\Delta}\right) \tag{45}$$

$$f_3 = 32a_3 \tag{46}$$

may be computed and stored. It is then possible, given $x/(2\Delta)$ and using (42), to interpolate $f(x)$ in just three adds and three multiplies, i.e., in six FLOPS.

Computational Notes

We summarize our scheme for computing the $[\mathbf{0}]^{(m)}$ $(0 \le m \le L)$ in Figure 1. The number of FLOPS contributed by each of the component steps to the work performed in the innermost contraction loop is also shown. We note the following points:

(1) When $L = 0$, steps (7) and (8) may be skipped.

(1) $R^2 = (\omega + \mu)^2 + \nu$ *3 FLOPs*

(2) $\frac{\theta^2}{2\Delta} = \frac{1}{2\Delta} / (\sigma_P + \sigma_Q)$ *2 FLOPs*

(3) $\frac{T}{2\Delta} = \frac{\theta^2}{2\Delta} R^2$ *1 FLOP*

(4) **IF** $\left[\frac{T}{2\Delta}\right] < \left[\frac{T_{crit}}{2\Delta}\right]$ **THEN** *1 FLOP*

(5) $j = \mathrm{INT}\left[\frac{T}{2\Delta}\right]$ *1 FLOP*

(6) $G_L(T) = g_0(j,L) + \frac{T}{2\Delta}\left[g_1(j,L) + \frac{T}{2\Delta}\left[g_2(j,L) + \frac{T}{2\Delta} g_3(j,L)\right]\right]$ *6 FLOPs*

(7) $\exp(-T) = e_0(j) + \frac{T}{2\Delta}\left[e_1(j) + \frac{T}{2\Delta}\left[e_2(j) + \frac{T}{2\Delta} e_3(j)\right]\right]$ *6 FLOPs*

(8) $G_m(T) = \left[\frac{2}{\pi}\right]^{1/2} \frac{(4\Delta)^{m+1/2}}{(2m+1)!!} \exp(-T) + \left[\frac{T}{2\Delta}\right] G_{m+1}(T)$ *3L FLOPs*

(9a) $[\mathbf{0}]^{(m)} = (2m-1)!!\, U_Q U_P \left[\frac{\theta^2}{2\Delta}\right]^{m+1/2} G_m(T)$ *1 SQUARE ROOT + 2(L + 1) FLOPs*

ELSE

(9b) $[\mathbf{0}]^{(m)} = (2m-1)!!\, U_Q \frac{U_P}{R^{2m+1}}$ *1 SQUARE ROOT + (L + 1) FLOPs*

END IF

Figure 1. An optimized scheme for the generation of $[\mathbf{0}]^{(m)}$ integrals from shell-pair data. The number of floating-point operations (FLOPS) that each step contributes to the cost of the innermost contraction loop [assuming that constants like $1/(2\Delta)$ have been precomputed] is shown in italics. See text for definitions of symbols.

(2) When $L = 1$, three FLOPS may be saved by interpolating $G_0(T)$ rather than generating it recursively from the $G_1(T)$ value, i.e., steps (7) and (8) may be replaced by a second step like (6).

(3) The multiplications by $(2m - 1)!!$ and U_Q in steps (9a) and (9b) may be factored out of the innermost contraction loop, and, for this reason, they do not contribute to the FLOP-cost of this loop.

(4) In limited computer memory, it may be impossible to store all the μ and ν values. When this is so, two alternatives are available. One of these is to evaluate μ and ν inside the loop over CD shells but outside the loop over

AB shells. The R^2 values can then be computed as in step (1). Alternatively, the R^2 values can simply be found in eight FLOPs using Eq. (8) instead of Eq. (17). In this case, μ and ν values are never needed. It can be shown that the first alternative becomes more efficient than the second when the number of primitives on A and B is large enough.

(5) By adding the FLOP costs of the individual steps in Figure 1, it is easy to show that the cost, in the innermost contraction loop, of forming a set of $[\mathbf{0}]^{(m)}$ $(0 \leq m \leq L)$ from shell-pair data is

	If $T < T_{\text{crit}}$	If $T \geq T_{\text{crit}}$
$L = 0$	$16 + S$	$8 + S$
$L = 1$	$24 + S$	$9 + S$
$L \geq 2$	$(5L + 22) + S$	$(L + 8) + S$

where S represents a square root evaluation. It is clear from this analysis that (not surprisingly) the multipole formula (21) is much less expensive than is the interpolation formula (22) and that, whenever T is sufficiently large, the former should be used.

Acknowledgment

This work was supported by the National Science Foundation under grant CHEM-8918623.

Bibliography

[1] (a) H. B. Schlegel, J. Chem. Phys. **77**, 3676 (1982). (b) S. Obara and A. Saika, J. Chem. Phys. **84**, 3963 (1986).
[2] M. Head-Gordon and J. A. Pople, J. Chem. Phys. **89**, 5777 (1988).
[3] T. P. Hamilton and H. F. Schaefer III, Chem. Phys. **150**, 163 (1991).
[4] P. M.W. Gill, M. Head-Gordon, and J. A. Pople, Int. J. Quantum Chem. Symp. **23**, 269 (1989).
[5] P. M.W. Gill, M. Head-Gordon, and J. A. Pople, J. Phys. Chem. **94**, 5564 (1990).
[6] P. M.W. Gill and J. A. Pople, Int. J. Quantum Chem.
[7] S. F. Boys, Proc. R. Soc. Lond. A**200**, 542 (1950).
[8] (a) J. Almlöf, K. Faegri, and K. Korsell, J. Comput. Chem. **3**, 385 (1982). (b) M. Head-Gordon and J. A. Pople, Chem. Phys. Lett. **153**, 503 (1988). (c) S. Saebo and J. Almlöf, Chem. Phys. Lett. **154**, 83 (1989).
[9] L. E. McMurchie and E. R. Davidson, J. Comput. Phys. **26**, 218 (1978).
[10] F. E. Harris, Int. J. Quantum Chem. **23**, 1469 (1983).
[11] F. Scheid, in *Theory and Problems of Numerical Analysis* (McGraw-Hill, New York, 1988).

Received November 20, 1990
Accepted for publication December 11, 1990