Theor Chem Account (2008) 120:185–189
DOI 10.1007/s00214-007-0295-5

REGULAR ARTICLE

A new algorithm of two-electron repulsion integral calculations: a combination of Pople–Hehre and McMurchie–Davidson methods

Kazuya Ishimura · Shigeru Nagase

Received: 26 November 2006 / Accepted: 26 February 2007 / Published online: 18 April 2007

Abstract A new algorithm of two-electron repulsion integral (ERI) calculations has been developed. In this algorithm, Cartesian axes are rotated to make several coordinate components zero or constant using the Pople–Hehre algorithm, and ERIs are evaluated at the rotated coordinate by the McMurchie–Davidson algorithm. The new algorithm applicable to $(ss|ss)$ to $(dd|dd)$ ERIs is implemented in the quantum chemistry program GAMESS. Test calculations show that the new algorithm reduces the computational times by 10–40%, as compared with the Pople–Hehre and the Rys quadrature algorithms.

Keywords Two-electron repulsion integral · Axis-switch · Recurrence relation

1 Introduction

In quantum chemistry, it is an important subject to calculate two-electron repulsion integrals (ERIs) at a high speed, because ERIs are fundamental in all *ab initio* self-consistent field (SCF) and post-SCF calculations. The generation of ERIs is especially time consuming in widely used direct SCF calculations because ERIs are generated in every SCF iteration. Since Boys proposed the use of Gaussian functions as basis functions in 1950 [1], many algorithms have been developed to reduce the computational time [2–17].

In the Pople–Hehre algorithm [2], Cartesian axes are rotated so that several coordinate components become zero or constant. The Floating-Point Operation (FLOP) count of the fourth and second power terms is reduced. The Rys quadrature developed by Dupuis, Rys, and King [3–5] is based on a set of orthogonal polynomials and is used especially for ERIs including high angular momentum functions. In the McMurchie–Davidson algorithm [6], the efficient recurrence relation derived from Hermite polynomials was adopted to generate all ERIs from $(ss|ss)$ type integrals. Obara and Saika [7] developed an efficient algorithm of a vertical recurrence relation (VRR) in which required ERIs are generated from actually needed and auxiliary ERIs of lower angular momentum functions. Head-Gordon and Pople (HGP) [8] derived a horizontal recurrence relation (HRR) to make several terms in VRR always vanish. Gill et al. [9] derived another relation by combining the MD and HGP algorithms. Lindh et al. [10] derived an algorithm from the Rys quadrature and HRR. Ishida [11] developed the accompanying coordinate expansion (ACE) algorithm based on the Rys quadrature, which can treat up to $(hh|hh)$ ERIs [12]. A different algorithm was proposed by Gill and Pople [13], and was named PRISM. In PRISM, the path from $(ss|ss)^{(m)}$ integrals to required ERIs is selected to minimize the FLOP count. The bra and ket transformations in the algorithms such as MD and HGP as well as the bra and ket contractions depend on the degrees of bra and ket contractions of the primitive functions.

In this paper, we report a new algorithm of ERI calculations by combining the PH and MD algorithms because MD can use all advantages of PH. The new algorithm is considerably faster than the original PH and Rys, as demonstrated by test calculations.

2 Methods

An unnormalized primitive Gaussian function $\varphi_a(\mathbf{r})$ with the exponent α at point $\mathbf{A}$ is given by

Contribution to the Mark S. Gordon 65th Birthday Festschrift Issue.

K. Ishimura · S. Nagase (✉)
Department of Theoretical Molecular Science,
Institute for Molecular Science, Myodaiji, Okazaki,
Aichi 444-8585, Japan
e-mail: nagase@ims.ac.jp

$$\varphi_a(\mathbf{r}) = (x - A_x)^{a_x}\left(y - A_y\right)^{a_y}(z - A_z)^{a_z} \times \exp\left[-\alpha_A(\mathbf{r} - \mathbf{A})^2\right]. \quad (1)$$

A primitive ERI can be written by

$$[\mathbf{ab}|\mathbf{cd}] = \iint \varphi_a(\mathbf{r}_1)\varphi_b(\mathbf{r}_1)r_{12}^{-1}\varphi_c(\mathbf{r}_2)\varphi_d(\mathbf{r}_2)d\mathbf{r}_1 d\mathbf{r}_2. \quad (2)$$

The bra part, a product of two primitive functions, can be written by

$$\exp\left[-\alpha_A(\mathbf{r} - \mathbf{A})^2 - \alpha_B(\mathbf{r} - \mathbf{B})^2\right] = L_P \exp\left[-\alpha_P(\mathbf{r} - \mathbf{P})^2\right] \quad (3)$$

where point **P** is defined as

$$\mathbf{P} = (\alpha_A\mathbf{A} + \alpha_B\mathbf{B})/\alpha_P, \quad (4)$$

$$\alpha_P = \alpha_A + \alpha_B, \quad (5)$$

and the constant L_P is

$$L_P = \exp\left[-\alpha_A\alpha_B(\mathbf{A} - \mathbf{B})^2/\alpha_P\right]. \quad (6)$$

In the same manner, the ket part in Eq. 2 can be written as follows:

$$\exp\left[-\alpha_C(\mathbf{r} - \mathbf{C})^2 - \alpha_D(\mathbf{r} - \mathbf{D})^2\right] = L_Q \exp\left[-\alpha_Q(\mathbf{r} - \mathbf{Q})^2\right], \quad (7)$$

$$\mathbf{Q} = (\alpha_C\mathbf{C} + \alpha_D\mathbf{D})/\alpha_Q, \quad (8)$$

$$\alpha_Q = \alpha_C + \alpha_D, \quad (9)$$

$$L_Q = \exp\left[-\alpha_C\alpha_D(\mathbf{C} - \mathbf{D})^2/\alpha_Q\right]. \quad (10)$$

Primitive functions are usually contracted in the following way,

$$\phi_a(\mathbf{r}) = \sum_k^K D_{ak}\varphi_{ak} \quad (11)$$

where D_{ak} is a contraction coefficient and K is the degree of contraction. A contracted ERI can be written from Eqs. 2 to 3,

$$(\mathbf{ab}|\mathbf{cd}) = \sum_i^{K_A}\sum_j^{K_B}\sum_k^{K_C}\sum_l^{K_D} D_{ai}D_{bj}D_{ck}D_{dl}\left[\mathbf{a}_{Ai}\mathbf{b}_{Bj}|\mathbf{c}_{Ck}\mathbf{d}_{Dl}\right]. \quad (12)$$

In the Pople–Hehre (PH) algorithm [2], Cartesian axes are rotated to make z- and y- directions along **AB** and perpendicular to **AB** and **CD**, respectively, as shown in Fig. 1, which are named axes-2 in the original paper. Point **P** lies on the **AB** line and depends on exponents α_A and α_B. In the rotated coordinate, PA_x, PA_y, PB_x, and PB_y are zero and PQ_x and PQ_y become constant at any point **P**. QC_y, and QD_y are zero at any point **Q**.

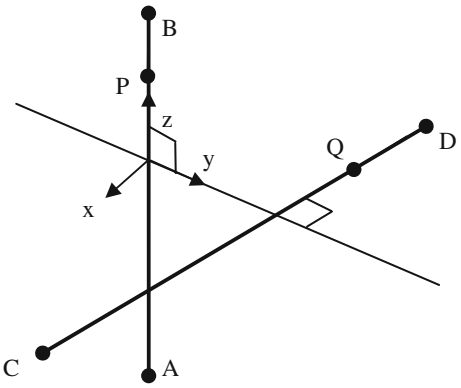

Fig. 1 Rotated Cartesian axes

In the McMurchie–Davidson (MD) algorithm [6], the bra part of the primitive [**ab**|**cd**] (=[**ab0**|**cd0**]) is formed from the following recurrence relation,

$$\left[(\mathbf{a} + \mathbf{1}_i)\mathbf{bp}\right| = p_i\left[\mathbf{ab}(\mathbf{p} - \mathbf{1}_i)\right| + (P_i - A_i) \times \left[\mathbf{abp}\right| + (2\alpha_P)^{-1}\left[\mathbf{ab}(\mathbf{p} + \mathbf{1}_i)\right| \quad (13)$$

where subscript i denotes a Cartesian direction (x, y, or z) and $\mathbf{1}_i$ is the unit vector of the i-direction. The ket part is also formed in the following way,

$$\left|(\mathbf{c} + \mathbf{1}_i)\mathbf{dq}\right] = q_i\left|\mathbf{cd}(\mathbf{q} - \mathbf{1}_i)\right] + (Q_i - C_i)\left|\mathbf{cdq}\right] + \left(2\alpha_Q\right)^{-1}\left|\mathbf{cd}(\mathbf{q} + \mathbf{1}_i)\right]. \quad (14)$$

[**p**|**q**] (=[**00p**|**00q**]) necessary to generate [**ab**|**cd**] is evaluated by

$$\left[\mathbf{p}|\mathbf{q}\right] = (-1)^q\left[\mathbf{p} + \mathbf{q}\right]. \quad (15)$$

Furthermore, [**r**](= $[\mathbf{r}]^{(0)}$) integrals can be obtained from $[\mathbf{0}]^{(\mathrm{m})}$(= $(ss|ss)^{(\mathrm{m})}$) integrals,

$$[\mathbf{r}]^{(m)} = (Q_i - P_i)[\mathbf{r} - \mathbf{1}_i]^{(m+1)} - (r_i - 1)[\mathbf{r} - \mathbf{2}_i]^{(m+1)} \quad (16)$$

where $\mathbf{2}_i$ is twice the unit vector in the i-direction and $[\mathbf{0}]^{(\mathrm{m})}$ is generated from

$$[\mathbf{0}]^{(m)} = D_aD_bD_cD_dL_PL_Q2^{m+1}\pi^{5/2} \times \frac{\alpha_P^{m-1}\alpha_Q^{m-1}}{\left(\alpha_P + \alpha_Q\right)^{m+1/2}}F_m(T) \quad (17)$$

where

$$F_m(T) = \int_0^1 t^{2m}\mathrm{e}^{-Tt^2}\mathrm{d}t \quad (18)$$

$$T = \frac{\alpha_P\alpha_Q}{\alpha_P + \alpha_Q}(\mathbf{PQ})^2. \quad (19)$$

When T is small, $F_m(T)$ for the largest m value is evaluated using Taylor or Chebyshev expansions and the others are evaluated using the following recurrence relation [18],

$$F_m(T) = \frac{1}{2m+1}\left[2T\,F_{m+1}(T) + \exp(-T)\right]. \tag{20}$$

When T is large, $F_0(T)$ is evaluated using the following equation,

$$F_0(T) = \sqrt{\pi/4T}, \tag{21}$$

and the others are evaluated using the following recurrence relation,

$$F_{m+1}(T) = (2m+1)F_m(T)/2T. \tag{22}$$

We developed the PH + MD algorithm by combining the PH and MD algorithms. Cartesian axes are rotated using the PH algorithm, as described above, and then ERIs are calculated using MD algorithm at the rotated coordinate. The calculated ERIs are transformed to the original coordinate. The second rotation in the original PH algorithm, axes-1, is not exploited in the PH + MD algorithm because the rotation is not useful for contracted basis functions. The pseudocode is shown in Fig. 2. Cartesian axes are first rotated, as shown in Fig. 1, and the quadruple loop starts at the rotated coordinate. The outer loop is over **c** and **d**, and the inner loop is over **a** and **b**. In the quadruple loop, $[\mathbf{0}]^{(\mathrm{m})}$ and the related integrals such as $(P_z - A_z)[\mathbf{0}]^{(\mathrm{m})}$, $(P_z - B_z)[\mathbf{0}]^{(\mathrm{m})}$, $(2\alpha_P)^{-1}[\mathbf{0}]^{(\mathrm{m})}$, and $(Q_z - P_z)[\mathbf{0}]^{(\mathrm{m})}$ are calculated and accumulated. Calculations of integrals including terms such as $(P_x - A_x)$ and $(Q_y - P_y)$ are skipped because PA_x, PA_y, PB_x, and PB_y are all zero and PQ_x and PQ_y are constant at the rotated coordinate. After the loop, the bra-contracted integrals $(\mathbf{0}]^{(m)}$ are obtained. In the double loop, (**r**] and the related integrals such as $(P_z - A_z)$(**r**], $(Q_x - C_x)$(**r**], and $(2\alpha_Q)^{-1}$(**r**] are calculated and accumulated. Calculations of integrals including terms such as $(P_x - A_x)$ and $(Q_y - C_y)$ are again skipped because PA_x, PA_y, PB_x, PB_y, QC_y, and QD_y are all zero. After the loop, the contracted integrals (**r**) are obtained. The (**ab**|**cd**) integrals are calculated from (**r**) using Eqs. 13–15 in the rotated coordinate and finally transformed into the original coordinate. $(sp, sp|sp, sp)$ and $(dd|dd)$ integrals have $90K^4$ and $830K^4$ constant values and $250\,K^2 + 380$ and $3900\,K^2 + 31000$ zero values, respectively (K is the degree of contraction).

As an example, we describe the $(ps|ss)$ generation. In the quadruple loop, $[\mathbf{0}]^{(0)}$ and $[\mathbf{0}]^{(1)}$ are calculated for each primitive function, and $(2\alpha_P)^{-1}[\mathbf{0}]^{(1)}$, $(Q_z - P_z)(2\alpha_P)^{-1}[\mathbf{0}]^{(1)}$, and $(P_z - A_z)[\mathbf{0}]^{(0)}$ are calculated and accumulated for the bra part. In the double loop, $(2\alpha_P)^{-1}$(**x**], $(2\alpha_P)^{-1}$(**y**], $(2\alpha_P)^{-1}$(**z**], and $(P_z - A_z)$(**0**] are calculated and accumulated for the ket part. After the loop, $(ps|ss)$ integrals are calculated using

```
rotate Cartesian axes
do c shells
  do d shells
    do a shells
      do b shells
        calculate F_m(T)
        calculate and accumulate [0]^(m) and the related integrals
      enddo b
    enddo a
    calculate and accumulate (r]^(m) and the related integrals
  enddo d
enddo c
calculate (ab|cd)
rotate (ab|cd) to original Cartesian axes
```

Fig. 2 Pseudocode of PH + MD algorithm

the following recurrence relations derived from Eqs. 13 to 15,

$$(p_x s|ss) = (2\alpha_P)^{-1}(\mathbf{x}), \tag{23}$$

$$(p_y s|ss) = (2\alpha_P)^{-1}(\mathbf{y}), \tag{24}$$

$$(p_z s|ss) = (P_z - A_z)(\mathbf{0}) + (2\alpha_P)^{-1}(\mathbf{x}). \tag{25}$$

At the end, $(ps|ss)$ integrals are transformed to the original coordinate.

3 Results

Tables 1 and 2 show the FLOP count parameters for $(sp, sp|sp, sp)$ and $(dd|dd)$, respectively. The FLOP count is given by

$$\mathrm{FLOPs} = xK^4 + yK^2 + z. \tag{26}$$

Table 3 shows the FLOP counts of $(sp, sp|sp, sp)$ for several K. The parameters x and y for K^4 and K^2 become very small in the PH + MD algorithm, compared with the original algorithms, because of zero components, constant components, and efficient recurrence relations. Instead, the zeroth power parameter z becomes large because the operations of the last recurrence relations, Eqs. 13–15, are performed. This PH + MD algorithm is especially suited for the moderate degree of contraction, for instance, STO-3G and 6–31G(d). The FLOP count of the PH + MD algorithm is not the least of all the algorithms. However, more than 80% operations of the algorithm are performed in do loops. This is a significant advantage for

Table 1 FLOP count parameters for $(sp, sp|sp, sp)$

	x	y	z
PH + MD	180	1,100	5,330
PH [2]	220	2,300	4,000
MD [6]	1,500	1,700	0
Dupuis, Rys, and King [3–5]	1,056	30	800
Head-Gordon and Pople [8]	1,400	30	800
Gill et al. [9]	450	1,300	1,700
Lindh et al. [10]	753	30	800
ACE-RR [17]	155	774	4,548

Table 2 FLOP count parameters for $(dd|dd)$

	x	y	z
PH + MD	850	11,860	173,500
MD [6]	27,300	24,000	0
Dupuis, Rys, and King [3–5]	30,900	220	0
Head-Gordon and Pople [8]	14,600	30	11,300
Gill et al. [9]	2,450	25,800	28,900
Lindh et al. [10]	10,255	30	11,300
ACE-b3 k3 [12]	327	2,281	163,000

Table 3 FLOP counts for $(sp, sp|sp, sp)$

	$K = 1$	$K = 2$	$K = 3$	$K = 6$
PH + MD	6,610	12,610	29,810	278,210
PH [2]	6,520	16,720	42,520	371,920
MD [6]	3,200	30,800	136,800	2,005,200
Dupuis, Rys, and King [3–5]	1,886	17,816	86,606	1,370,456
Head-Gordon and Pople [8]	2,230	23,320	114,470	1,816,280
Gill [9]	3,450	14,100	49,850	631,700
Lindh [10]	1,583	12,968	62,063	977,768
ACE-RR [17]	5,477	10,124	24,069	233,292

practical calculations because of the sequential access to fast memory.

The PH + MD algorithm derived for $(ss|ss)$ to $(dd|dd)$ was implemented in the quantum chemistry package GAMESS [19] by adding 39,000 lines. Test calculations were performed for PH + MD and compared with the PH and Rys algorithms available in GAMESS using a 3.0 GHz Pentium4 machine. PH is used for ERIs including s and sp functions and Rys is used for ERIs including d functions. The computational times for the Fock matrix generation in the first direct SCF iteration are shown in Table 4 for taxol ($C_{47}H_{51}NO_{14}$, no symmetry) and luciferin ($C_{11}H_8N_2O_3S_2$, no symmetry). STO-3G, 6–31G, and 6–31G(d) basis sets were used for taxol and 6–31G, 6–31G(d), and aug-cc-pVDZ were used for luciferin. An integral prescreening threshold of 10^{-9} was adopted

Table 4 Computational times of PH + MD, PH, and Rys algorithms (sec)

	PH + MD	PH	PH and Rys
Taxol			
STO-3G	69.9	85.7	
6–31G	379.4	475.4	
6–31G(d)	1, 361.8		2, 015.2
Luciferin			
6–31G	8.6	9.6	
6–31G(d)	39.3		62.2
aug-cc-pVDZ	1, 154.5		2, 014.9

in all calculations. As shown in Table 4, the PH + MD algorithm reduces computational times by 10–20% in comparison with the PH algorithm for the STO-3G and 6–31G basis sets. This is because the FLOP count of $(sp, sp|sp, sp)$ is 25–30% reduced in PH + MD though the FLOP counts of $(ss|ss)$ and $(sp, s|ss)$ are almost the same in PH + MD and PH. It is interesting that PH + MD reduces the computational times by 32–43% in comparison with PH and Rys for 6–31G(d) and aug-cc-pVDZ. This is owing to the fact that ERIs of sp and d functions such as $(d, sp|d, sp)$ and $(dd|sp, sp)$ are generated very fast. Only for primitive $(dd|dd)$ calculations PH + MD is somewhat slower than Rys. The number of primitive $(dd|dd)$ ERIs is usually very small, so that the disadvantage is negligible in practical calculations. Due to the small x coefficient for quartic work, the strength of this new approach must be even more apparent for heavy atoms with valence or core d electrons, whose basis sets would contain contracted d functions.

4 Conclusion

A new algorithm to calculate $(ss|ss)$ to $(dd|dd)$ ERIs is developed by combining the PH and MD algorithms and implemented in the GAMESS program. The FLOP count is highly reduced because of the axis-switch in PH and the recurrence relation in MD. The PH + MD algorithm is considerably faster than the PH and Rys algorithms, especially for the basis sets including d functions such as 6–31G(d) and aug-cc-pVDZ.

Acknowledgements We thank Dr. M. W. Schmidt for valuable discussion. This work was supported by the Grant-in-Aid for the Next Generation Super Computer Project (Nanoscience Program) and Scientific Research on Priority Area from the Ministry of Education, Culture, Sports, Science, and Technology of Japan.

References

1. Boys SF (1950) Proc R Soc Lond A 200:542
2. Pople JA, Hehre WJ (1978) J Comput Phys 27:161

3. King HF, Dupuis M (1976) J Comput Phys 21:144
4. Dupuis M, Rys J, King HF (1976) J Chem Phys 65:111
5. Rys J, Dupuis M, King HF (1983) J Comput Chem 4:154
6. McMurchie LE, Davidson ER (1978) J Comput Phys 26:218
7. Obara S, Saika A (1986) J Chem Phys 84:3963
8. Head-Gordon M, Pople JA (1988) J Chem Phys 89:5777
9. Gill PMW, Head-Gordon M, Pople JA (1990) J Phys Chem 94:5564
10. Lindh R, Ryu U, Liu B (1991) J Chem Phys 95:5889
11. Ishida K (1996) Int J Quantum Chem 59:209
12. Ishida K (1998) J Comput Chem 19:923
13. Gill PMW, Pople JA (1991) Int J Quantum Chem 40:753
14. Schlegel HB (1982) J Chem Phys 77:3676
15. Hamilton TP, Schaefer HFIII (1991) Chem Phys 150:163
16. Ten-no S (1993) Chem Phys Lett 211:259
17. Kobayashi M, Nakai H (2004) J Chem Phys 121:4050
18. Gill PMW, Johnson BG, Pople JA (1991) Int J Quantum Chem 40:745
19. Schmidt MW, Baldridge KK, Boatz JA, Elbert ST, Gordon MS, Jensen JH, Koseki S, Matsunaga N, Nguyen KA, Su S, Windus TL, Dupuis M, Montgomery JA (1993) J Comput Chem 14:1347