JOURNAL OF CHEMICAL EDUCATION

Article

Cite This: J. Chem. Educ. 2018, 95, 1572−1578

pubs.acs.org/jchemeduc

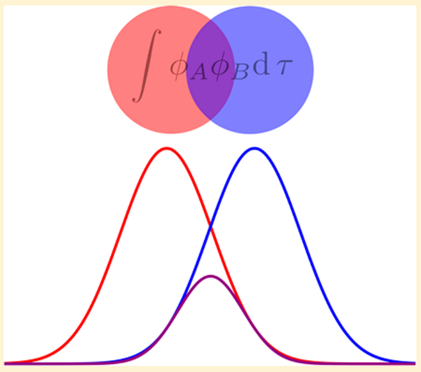
Student-Friendly Guide to Molecular Integrals

Kevin V. Murphy,*[†,‡] Justin M. Turney,[‡] and Henry F. Schaefer, III[‡]

[†]Department of Science, Valley City State University, Valley City, North Dakota 58072, United States and
[‡]Center for Computational Quantum Chemistry, University of Georgia, Athens, Georgia 30602, United States

Supporting Information

ABSTRACT: Preceding even the Hartree−Fock method, molecular integrals are the very foundation upon which quantum chemical molecular modeling depends. Discussions of molecular integrals are normally found only in advanced and technical texts or articles. The objective of the present article is to provide less experienced readers, or students in a physical/computational chemistry course, a thorough understanding of molecular integrals. Through a series of detailed Handouts, the student/reader can participate in the derivation of molecular integrals, and in turn implement them in computer code. Hartree−Fock theory is discussed in enough detail to motivate the molecular integrals and address such topics as the atomic orbital basis. An introduction to the programming language of choice, Python3, is provided, tailored toward developing the essential skills necessary for implementing molecular integrals. The article is intended to be useful not only to instructors of physical/computational chemistry, but also to any reader who has independently sought a primer on this elusive subject.



KEYWORDS: *Physical Chemistry, Quantum Chemistry, Computational Chemistry, Theoretical Chemistry, MO Theory, Graduate Education/Research, Upper-Division Undergraduate, Computer-Based Learning*

INTRODUCTION

The fledgling computational quantum chemist is often directed to implement (program) the Hartree−Fock (HF) method in computer code. This is done as an exercise to help her or him better understand this powerful method of constructing molecular orbitals from atomic orbitals. Being able to implement HF requires having access to "molecular integrals", quantities on which the HF method performs its iterative operations. To this end, the student may be given a set of text files, each filled with a long list of cryptic numbers for reading in by their program. Alternatively, the student is advised to use an extant quantum chemistry program to compute these integrals, so that they may call them into their code. Too often the origin and form of these fundamental quantities remain a mystery.

Many students and professional computational quantum chemists go about their work with only a vague idea of what molecular integrals are and how they are generated. A perusal of the literature does little to enlighten. To redress this unfortunate state of affairs, we offer the present article. The objective is to elucidate precisely what molecular integrals are, how their mathematical form is derived, and how to implement them in code. It is written to be accessible to the undergraduate who has completed, or is taking part in, the physical chemistry sequence. We hope it will be a useful resource for instructors of undergraduate or graduate-level physical and computational chemistry courses. Among other features, the present article distinguishes itself from the few other introductory treatments[1−4] of molecular integrals by presenting methods, including their derivation, that are applicable to orbitals of arbitrary angular momentum, for all atoms, of any molecule, with any geometry.

The molecular integrals that are derived and presented here are not the most computationally efficient; that is, they are not the fastest means of generating these numbers. These molecular integrals are the explicit, "block" equations proposed by Taketa, Huzinaga, and O-ohata in 1966.[5] There is a rich literature[6−11] dedicated to the purpose of efficiently computing molecular integrals, but unfortunately it is incomprehensible to all but the initiated. The point of this article is to provide basic understanding. When the reader has completed the exercises, they will be armed with the knowledge necessary to interpret the advanced literature. Additionally, in a future article we will address some of the more modern methods of molecular integral evaluation, such as the recursion schemes of the Obara—Saika method[6] and Head-Gordon—Pople method,[7] the quadrature of Dupuis and King,[8] and others. The present article establishes the fundamental knowledge and skills upon which a discussion of those methods will depend.

In this main article, we provide an overview and preview of the structure and contents of the pedagogical documents (derivations, programming projects, and more), which are available as downloadable Supporting Information materials. Following the preview, we conclude this main article with a

Received: May 1, 2018
Revised: June 26, 2018
Published: July 19, 2018

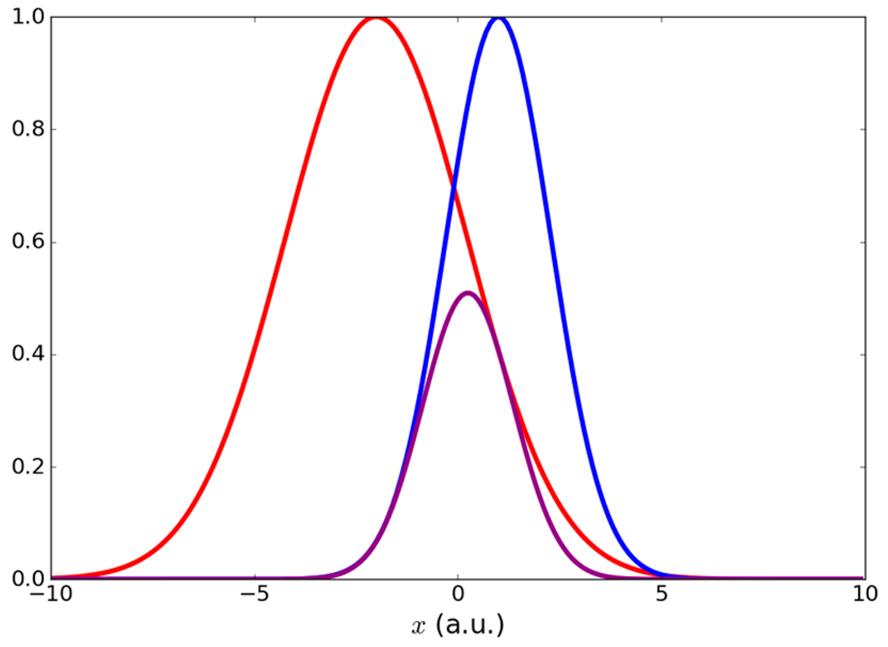


Figure 1. Product of two Gaussian functions (red and blue), each representing a distinct, atom-centered orbital, yields a third, intermediate Gaussian function (purple). This permits the evaluation of a molecular integral over two orbitals (e.g., eq 4) to be calculated with respect to a *single* intermediate center, rather than two separate centers. Details are provided in Handout 1.

brief discussion of the authors' experience using these materials with two undergraduate and two graduate students. Scaffolding techniques for instructors who seek to tailor the project to the level of their students are provided.

Quantum chemistry is a vast, deep subject. In the interest of clarity and space, we skim over fundamental but tangential details of quantum mechanics such as antisymmetry. Fortunately, most of these will have been addressed in a physical chemistry course. There are many excellent textbooks[12−14] and *J. Chem. Educ.* articles,[15−31] if review is needed.

PEDAGOGICAL CONTENT OVERVIEW

Molecular modeling based upon quantum mechanics is a complex topic. In this article we strive to provide a *comprehensible* and *comprehensive* introduction to the foundational quantities upon which all such modeling depends: the molecular integrals. Because of the detail necessary to explicate the topic in this way, the bulk of the article appears in separate documents of the Supporting Information (SI). We term five of these documents "Handouts", because they are designed to be ready for use by instructors as handouts to their students. The sixth Supporting Information file is a compilation of computer code, input files, and output files.

While it is recommended that the Handouts be read in order, it is at the instructor's (or general reader's) discretion to take advantage of their modular structure, selecting those pieces useful to their purposes; the Handouts have been written to be as independent of one another as possible, including only a few cross-references. For example, instructors of graduate courses in quantum chemistry that assume some programming experience on the part of the student may choose to bypass Handout 3, which is an introduction to programming. By contrast, instructors of an undergraduate course in computational or physical chemistry may assign only Handout 1 (background), parts of Handout 2 (derivation through the overlap integrals), and Handout 3 (a detailed introduction to programming, guiding through the implementation of the overlap integrals).

We now provide a brief preview of the contents of each Handout.

In Handout 1, molecular integrals are motivated particularly with respect to their role in Hartree—Fock theory. We begin with a discussion of the Schrödinger equation

$$\hat{H}\Psi = E\Psi \tag{1}$$

which is solved on a computer for a molecule by recasting it into the Hartree—Fock—Roothan—Hall equation, which in matrix form is

$$\mathbf{FC} = \varepsilon\mathbf{SC} \tag{2}$$

The matrix **S** is the overlap integral matrix, and each matrix element is made up of an integral (rather, the integral's solution) of the following general form

$$S_{AB} = \int \phi_A \phi_B \, d\tau \tag{3}$$

where ϕ_A and ϕ_B are atomic orbitals. Each atomic orbital ϕ has the general form

$$\phi = \sum_p d_p \underbrace{x^l y^m z^n}_{\text{angular}} \underbrace{e^{-\alpha_p r^2}}_{\text{radial}} \tag{4}$$

The angular component of the atomic orbital ϕ has powers of angular momentum for each Cartesian coordinate. For example, $l = 0$, $m = 1$, and $n = 0$ describe a p_y orbital, one of three Cartesian p orbitals, while $l = 1$, $m = 0$, $n = 1$ describes a d_{xz} orbital, one of six Cartesian d orbitals. (Note that l, m, and n are related, but not identical, to the azimuthal and magnetic quantum numbers.) The radial component is expressed as a Gaussian function (e^{-r^2}) instead of the more accurate exponential function (e^{-r}), because Gaussian functions make evaluation of the integrals easier. They do so by allowing two atom-centered orbitals, ϕ_A and ϕ_B, to be integrated over with

One of the polynomials in Eq. 13 can be expanded as

$$(x - \vec{A}_x)^{l_A} = \{(x - \vec{P}_x) + (\vec{P}_x - \vec{A}_x)\}^{l_A} = (x_P - \vec{PA}_x)^{l_A}\,. \tag{15}$$

We have employed the x-component of the vector quantity $\vec{P}$, which is the third Gaussian center located between centers $\vec{A}$ and $\vec{B}$. Using the standard binomial expansion,

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k\,, \tag{16}$$

where $\binom{n}{k}$ is the binomial coefficient meaning "n choose k" or

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}\,, \tag{17}$$

we can recast Eq. 15.

Exercise.
Express the right-most side of Eq. 15 as a binomial expansion, with a binomial coefficient "l_A choose i."

Solution.

$$(x_P - \vec{PA}_x)^{l_A} = \sum_{i=0}^{l_A} \binom{l_A}{i} (\vec{PA}_x)^{l_A - i} (x_P)^i \tag{18}$$

[end of solution]

Figure 2. Example of an exercise appearing in Handout 2, which serves to further the derivation of the solution to the overlap integral S_{AB} (eq 3).

respect to a single, intermediate center, as Figure 1 suggests. By summing several such Gaussian functions, accuracy approaching that of the exponential function can be achieved. The parameters d_p and α_p are values found in basis set tables. (For the interested reader, the method by which these parameters are derived can be found in textbooks[2] and review articles.[33,34] Very briefly, for the simplest basis sets, an optimum value for α_p is found by maximizing the overlap between an orbital represented by a Gaussian function and an orbital represented by an exponential function, which the Gaussian function is intended to emulate. The contraction coefficients d_p are determined by fitting the exponential function curve to a sum of Gaussian functions, using least-squares methods. As will be discussed in Handout 1, summing over a larger number of Gaussian functions, that is, larger values of p in eq 4, gives a better likeness of an exponential function.)

The Fock matrix **F** is effectively the sum of three matrices (**T**, **V**, **G**), the matrix elements of which constitute the other three molecular integrals that we address. The elements of kinetic energy integral matrix **T** have the general form

$$T_{\mathrm{AB}} = \int \phi_{\mathrm{A}} \left(-\frac{1}{2}\nabla^2\right) \phi_{\mathrm{B}}\, \mathrm{d}\tau \tag{5}$$

dτ represents a general differential volume element; its exact form depends on the coordinate system chosen to solve the integral. The elements of electron−nuclear attraction integral matrix **V** have the general form

$$V_{\mathrm{AB}} = \int \phi_{\mathrm{A}} \left(\frac{-Z_{\mathrm{C}}}{r_{i\mathrm{C}}}\right) \phi_{\mathrm{B}}\, \mathrm{d}\tau \tag{6}$$

Indices i and C represent electron i and nucleus C, respectively, and r_{iC} is the distance between electron i and nucleus C. The elements of electron−electron repulsion integral matrix (or tensor) **G** have the general form

$$G_{\mathrm{ABCD}} = \int \phi_{\mathrm{A}} \phi_{\mathrm{B}} \left(\frac{1}{r_{ij}}\right) \phi_{\mathrm{C}} \phi_{\mathrm{D}}\, \mathrm{d}\tau \tag{7}$$

the solution of which requires an intermediate center over not just orbitals ϕ_{A} and ϕ_{B}, but an additional intermediate center over orbitals ϕ_{C} and ϕ_{D}. r_{ij} is the distance between electron i and electron j.

To address these solutions, as well as the structure of the matrices **S**, **T**, **V**, and **G** that they make up, we discuss the meaning of an atomic orbital basis. This is particularly important because molecular orbitals are formed from the atomic orbital basis, by iteratively solving the Hartree—Fock—Roothan—Hall equation (eq 2). We discuss matrix and tensor structures in Handouts 1 and 2.

The full, explicit derivation of the solutions to eqs 3, 5, 6, and 7, for the solutions are what we seek to implement in computer code, is provided in Handout 2. The equations are based on the original 1966 paper by Taketa, Huzinaga, and O-ohata,[1] and use notation developed by Cook.[32] The derivation entails Fourier transforms, which mitigate the challenge of the inverse operators present in eqs 6 and 7. The student (reader) is encouraged to actively participate in the derivation through guided exercises. For example, as the student (reader) derives the solution to the angular component of an overlap integral, they are given the question that appears in Figure 3. They are

14. Define another function for computing the the constant c_k. Referring to Eq. 21 of Handout 2, and calling the `binomial` function of the `special` module (part of the SciPy library), type in the following code:

```
def compute_ck(arguments):
  for i in range(l+1): # i from 0 up to and including l, hence +1
    #for loop over j
      if i + k == ?:
        ck += special.binom(l,k) * ? * a**(l-i) * ?
  return ?
```

Complete the code.

Figure 3. Example of a programming exercise that appears in Handout 3. Students are asked to complete the function that calculates the coefficient c_k, by typing in the provided code, and replacing the dark blue text with functioning code.

```
GAMESS results:

          ********************
          1 ELECTRON INTEGRALS
          ********************
 BARE NUCLEUS HAMILTONIAN INTEGRALS (H=T+V)
                     1          2
    1  HE 1  S   -2.598283
    2  H  2  S   -1.431829  -1.731826
 KINETIC ENERGY INTEGRALS
                     1          2
    1  HE 1  S    1.411763
    2  H  2  S    0.197443   0.760032

          --------------------
          2 ELECTRON INTEGRALS
          --------------------
   1   1   1   1  1.0      1.055712940    2   1   1   1  1.0
0.443964988
   2   2   1   1  1.0      0.590807309    2   1   2   1  1.0
0.224319340
   2   2   2   1  1.0      0.367410158    2   2   2   2  1.0
0.774605944

 FINAL RHF ENERGY IS      -2.8418364976 AFTER   8 ITERATIONS
```

```
Present results:

T =
[[ 1.411763  0.197443]
 [ 0.197443  0.760032]]

H = T + V =
[[-2.598283 -1.431828]
 [-1.431828 -1.731826]]

G =
[[[[ 1.055713  0.443965]
   [ 0.443965  0.590807]]

  [[ 0.443965  0.224319]
   [ 0.224319  0.367410]]]

 [[[ 0.443965  0.224319]
   [ 0.224319  0.367410]]

  [[ 0.590807  0.367410]
   [ 0.367410  0.774606]]]]

Energy (units Hartree) = -2.841836 (9 iterations)
```

Figure 4. Molecular integral (and Hartree—Fock energy) values produced by the methods detailed in this article and its associated Handouts (shown in the right panel) are, within numerical precision limits, identical to those produced by the widely used quantum chemistry software GAMESS (shown in the left panel). The present example is for HeH^+ (interatomic distance of 1.4632 Bohrs) in the STO-3G basis. Note that the "2 ELECTRON INTEGRALS" printed in the GAMESS output features only unique matrix elements, whereas the "Present results" output features all two-electron integral values making up **G** (the "matrix" of two-electron integrals); more information on the structure of **G** can be found in Handouts 1 and 2.

encouraged to cover up solutions, which follow a solid black line.

Some of the questions are quite simple, as suggested in Figure 2. Others are more involved, such as one directed to deriving the equation for the normalization constant *N*, an equation which is ubiquitous throughout the solutions to the molecular integrals. Regardless of an exercise's complexity, students are given detailed guidance and adequate information to solve the problem. Completion of the exercise serves to move the derivation forward. Most exercises are designed to take no more than about 15 min, though this will depend on the student's proficiency with what is largely algebraic mathematics.

Handout 3 is a gentle introduction to the Python programming language for those with little or no prior programming experience. It is tailored toward programming molecular integrals. Python was chosen over other languages for both its relative ease of use as well as its power. Once again, a series of exercises are included for the student (reader) to practice using their newfound knowledge. For example, Figure 3 shows a coding exercise wherein students are asked to practice their for loop skills, as well as hone their ability to translate mathematical equations into code. They must type in the provided block of code and replace the placeholder text that appears in dark blue with functioning code. The skills developed in Handout 3 prepare the reader for Handout 4.

The coding introduction of Handout 3 is followed by the full implementation of the molecular integrals in code in Handout 4. As with the other documents (except for Handout 3 which occasionally references Handout 2), Handout 4 can be utilized independently of the other Handouts. The construction of each molecular integral matrix (**S**, **T**, **V**, **G**) and its elements is treated in turn. While the instructions in Handout 4 are highly detailed, they are less overtly "guided" than those in Handout 3, requiring the application of deeper problem-solving strategies. Hints and tips are provided throughout, including directions for using special functions that perform certain mathematical operations with simple keywords, such as the calculation of factorials and double factorials. Figure 4 demonstrates, using the simple molecule HeH^+, that the derived and implemented equations produce values identical to those generated by the established quantum chemistry program GAMESS.[35]

Having calculated the molecular integrals, Handout 5 describes their deployment in the implementation of the HF method. This Handout is quite brief by comparison to the other handouts, highlighting the complexity of molecular integrals as compared to the HF "algorithm". Completion of Handouts 4 and 5 marks the development of a complete, independent quantum chemistry program.

The sixth Supporting Information file is a compilation of the authors' implementation of the programming projects. As with answers in the back of a mathematics textbook, the reader should use these with care. It is strongly advised to reference these codes (.py files) only as a last resort and only as needed. Otherwise the learning value is forfeit.

Additionally, Supporting Information file 6 includes molecule coordinate (.xyz) input files and output integral matrix files (in .txt format). For example, out.S.h2o.txt contains the "printed" calculated overlap integral matrix corresponding to the h2o.xyz input file. These input files are included as "test cases", so that reference to the respective output integral matrices enables one to ensure that her or his code generates the same, correct integral matrix.

■ RESULTS AND SCAFFOLDING TECHNIQUES FOR INSTRUCTORS

A sample of four students completed the central programming project, corresponding to Handout 4. This was preceded by a lecture addressing most of the content present in Handout 1 and parts of the derivations in Handout 2. Two of the four students were undergraduates who had just completed physical chemistry. The other two were recent graduates about to begin their doctoral studies. Every student completed the entire set of programming projects. The completion times for each student were approximately 8, 20, 35, and 40 h.

The two students who completed the project in the least amount of time worked independently and required no assistance. The first of these (8 h) was an undergraduate studying computer science and chemistry. The second student (20 h) had graduated with majors in chemistry and mathematics, and a minor in computer science. The student who completed the project in 35 h was a graduate and had a few, clarifying questions, but was otherwise independent; this student did consult online resources for coding guidance. The student had a background in chemistry and mathematics, and had acquired some previous coding experience. The student who completed the project in 40 h was an undergraduate studying chemistry, and had no experience in programming or formal background in mathematics. Several hours of close assistance from one of the authors was required to complete the project, though it should be noted that Handout 3 (introduction to programming) was not available to the student at this time. Other than the student who used online resources for coding guidance, no students indicated the need to consult outside references about molecular integrals, other than for personal interest in the subject.

The students indicated that the most difficult part of the programming projects was the complexity of the equations needing to be implemented, increasing the possibility of introducing errors that would later require debugging. One student suggested that maximally functionalizing code (i.e., writing separate functions for frequently used tasks), and having reference integral matrices to compare against (as we have provided in Supporting Information file 6), helped reduce this difficulty. The students reported finding the lecture and programming project instructions clear and helpful. The predominant reported benefit was the clarity the exercises brought to their previously vague and abstract conceptions of molecular integrals. One student also wrote, "It made the sheer number of equations computational quantum chemists use more viscerally real, and demonstrated the importance of efficiency in code." The students reflected that the experience helped them develop a more intuitive understanding of each molecular integral type, especially two-electron integrals, and their relation to atomic and molecular features like orbital position, orientation, and type (e.g., s, p, d). They described having a greater appreciation for the use of Gaussian functions to approximate Slater-type orbitals. The HF method itself, which some of them had previously implemented, "made more sense" as a result of these exercises.

While one must be careful of making generalizations from a very small sample, it is reasonable to infer that students with more computer programming experience, especially those having formally studied computer science, will complete the projects significantly faster than those without it. More formal experience in mathematics probably helps as well, though the magnitude of this effect is difficult to parse.

The materials making up this article, including the five modular Handouts, have been designed to be adopted to varying extents by instructors in their undergraduate or graduate courses, as suits the needs of their course. It would be very difficult for the undergraduate physical chemistry instructor to teach the full suite of materials contained herein, even if the course should have a computational chemistry focus. Because the overlap integrals are the easiest to understand and implement, one suggestion to the instructor is to assign implementation of just the **S** matrix. This is especially so because Handout 3 guides the reader with *no* programming experience through the implementation of this integral matrix via an especially detailed set of exercises. The reading and programming exercises of Handout 3 should be preceded by a reading of Handout 1, as well as completion of Handout 2 and its exercises through section one (Overlap Integrals), the presentation and mathematics of which have been written at a level suitable for undergraduate physical chemistry students. If students are directed to implement the HF code (Handout 5), which is a relatively straightforward exercise with ample suggestions provided therein, the instructor can provide the code or values for **T**, **V**, and **G**.

Particularly for undergraduate courses, the instructor might consider scaffolding the material by providing incomplete code to the students, for them to fill in an assignment. This technique is demonstrated in Handout 3. There are at least two ways of utilizing this strategy:

1. Within a function, provide the beginning of a block of code that reveals a pattern, for example, the code for the *x*-component of the overlap integral. Students can observe the pattern and replicate it for the *y*- and *z*-components.
2. Provide largely completed code where there are just a few empty functions that need to be written. This might entail the student implementing a handful of equations (such as the coefficient c_k; see Handout 3 or Handout 4), where each equation requires only a few lines of computer code.

By these methods, students are exposed to the logic of molecular integrals and how they are built in the atomic orbital basis, and are given the opportunity to practice basic coding skills without the steeper demand of generating the code entirely from scratch.

The kinetic energy integrals, **T**, are a fairly direct extension of the overlap integrals in **S**, and do not typically require much more work to implement. An optional, "bonus" exercise for undergraduate students, or perhaps requirement for students in nonspecialist graduate-level physical/computational chemistry courses, may be the additional completion of **T**. The instructor should consider assigning the derivation of **T** as provided in section two of Handout 2 (less than two pages), as well as the programming instructions on page five of Handout 4.

There is a relatively large difficulty gap between coding the integrals of **S** and **T** and coding the two potential energy integrals (electron−nuclear attraction **V** and electron−electron repulsion **G**), due to the complexity of the matrix element equations. Assignment of these latter integrals, then, would be better suited to graduate students specializing in computational chemistry. Their contents make up the remainder of Handouts 2 and 4, such that derivation and implementation of the complete set of molecular integrals depend on completing Handouts 1 through 4, with Handout 3 being excluded for courses assuming some prior programming experience. Completion of a quantum chemistry code, through Hartree−Fock, is accomplished by assigning Handout 5.

In the authors' experience, the materials *can* be effectively utilized in their entirety by graduate students that are pursuing a specialization in computational chemistry, either as part of a graduate-level course or independently. Indeed, as just reported, our experience shows that even well-prepared undergraduate students are capable of successfully completing the instructional materials provided. Independent studies by a graduate student will certainly be facilitated by access to knowledgeable peers and/or the student's advisor.

The following additional techniques should be considered for use by instructors in their courses, whether undergraduate, nonspecialist graduate, or specialist (computational chemistry) graduate:

1. Students should have access to reference integral matrices for several molecules. That is, for a few specific .xyz files ("test cases") students should also be given the correct integrals. Students can use these test cases to directly compare the output of their code to the correct molecular integral values, which is especially valuable when debugging. During debugging, patterns often appear in the integral matrices that make it possible to deduce, or at least narrow down, the location of the offending code. To this end, .xyz files and the correct integrals for H_2, HeH^+, HF (hydrogen fluoride), and H_2O are packaged as a .zip file in Supporting Information file 6.
2. Some students may benefit from working in pairs, though this has not been assessed. A common practice in software development is pair programming, in which two programmers work together at one computer.[36] For the current project, one student would write the code while the other reviews each line of code as it is typed in. The two students should switch roles frequently.

Regardless of the materials that are selected by an instructor, it is believed that students (and the general reader) will gain a deeper appreciation and understanding of computational quantum chemistry by actively engaging these pedagogical resources.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available on the ACS Publications website at DOI: 10.1021/acs.jchemed.8b00255.

Handout 1, background to molecular integrals and HF theory (PDF)

Handout 2, derivation of the molecular integrals, with exercises (PDF)

Handout 3, coding in Python: The essential skills, with exercises (PDF)

Handout 4, implementation of the molecular integrals: Programming project (PDF)

Handout 5, using the molecular integrals: HF programming project (PDF)

Test case input and output files, and authors' Python implementation (ZIP)

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: kevin.murphy.1@vcsu.edu.

ORCID

Justin M. Turney: 0000-0003-3659-0711

Henry F. Schaefer, III: 0000-0003-0252-2083

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

We acknowledge support from the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences, under Contract DE-SC0018412, and the U.S. National Science Foundation, Grant CHE-1661604.

■ REFERENCES

(1) Szabo, A.; Ostlund, N. *Integral Evaluation with 1s Primitive Gaussians. Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*; McGraw-Hill: New York, 1989; pp 410−416.

(2) Quinn, C. M. *Computational Quantum Chemistry: An Interactive Introduction to Basis Set Theory*; Academic Press: New York, 2002.

(3) Orsini, G. Exploring Do-It-Yourself Approaches in Computational Quantum Chemistry: The Pedagogical Benefits of the Classical Boys Algorithm. *J. Chem. Educ.* **2015**, *92* (11), 1853−1859.

(4) Goings, J. J. A (Hopefully) Gentle Guide to the Computer Implementation of Molecular Integrals Over Gaussian Basis Functions. http://joshuagoings.com/2017/04/28/integrals/ (accessed June 4, 2018).

(5) Taketa, H.; Huzinaga, S.; O-ohata, K. Gaussian-Expansion Methods for Molecular Integrals. *J. Phys. Soc. Jpn.* **1966**, *21* (11), 2313−2314.

(6) Obara, S.; Saika, A. Efficient Recursive Computation of Molecular Integrals Over Cartesian Gaussian Functions. *J. Chem. Phys.* **1986**, *84* (7), 3963−3974.

(7) Head-Gordon, M.; Pople, J. A. A Method for Two-Electron Gaussian Integral and Integral Derivative Evaluation Using Recurrence Relations. *J. Chem. Phys.* **1988**, *89* (9), 5777−5786.

(8) King, H. F.; Dupuis, M. Numerical Integration Using Rys Polynomials. *J. Comput. Phys.* **1976**, *21* (2), 144−165.

(9) Gill, P. W. Molecular Integrals Over Gaussian Basis Functions. In *Advances in Quantum Chemistry*; Sabin, J. R., Zerner, M. C., Eds.; Academic Press: New York, 1994; Vol. 25, pp 141−205.

(10) Helgaker, T.; Jorgensen, P.; Olsen, J. *Molecular Integral Evaluation. Molecular Electronic-Structure Theory*; John Wiley & Sons: Chichester, 2000; pp 336−428.

(11) Helgaker, T.; Taylor, P. R. Gaussian Basis Sets and Molecular Integrals. In *Modern Electronic Structure Theory*; Yarkony, D., Ed.; World Scientific Publishing: River Edge, NJ, 1995; Vol. *2*, pp 725−856.

(12) McQuarrie, D. A.; Simon, J. D. *Physical Chemistry: A Molecular Approach*, 1st ed.; University Science Books: CA, 1997; pp 70−243.

(13) McQuarrie, D. A. *Quantum Chemistry*, 2nd ed.; University Science Books: CA, 2008; pp 97−432.

(14) Levine, I. *Quantum Chemistry*, 7th ed.; Prentice Hall: New York, 2014; pp 142−187.

(15) Page, T. R.; Boots, C. A.; Freitag, M. A. Quantum Chemistry: Restricted Hartree—Fock SCF Calculations Using Microsoft Excel. *J. Chem. Educ.* **2008**, *85* (1), 159.

(16) Ge, Y. Let Students Derive, by Themselves, Two-Dimensional Atomic and Molecular Quantum Chemistry from Scratch. *J. Chem. Educ.* **2016**, *93* (12), 2033−2039.

(17) Ge, Y.; Rittenhouse, R. C.; Buchanan, J. C.; Livingston, B. Using a Spreadsheet To Solve the Schrödinger Equations for the Energies of the Ground Electronic State and the Two Lowest Excited States of H_2. *J. Chem. Educ.* **2014**, *91* (6), 853−859.

(18) Halpern, A. M.; Glendening, E. D. Exploring the Nature of the H_2 Bond. 1. Using Spreadsheet Calculations to Examine the Valence Bond and Molecular Orbital Methods. *J. Chem. Educ.* **2013**, *90* (11), 1452−1458.

(19) Halpern, A. M.; Glendening, E. D. Exploring the Nature of the H_2 Bond. 2. Using Ab Initio Molecular Orbital Calculations To Obtain the Molecular Constants. *J. Chem. Educ.* **2013**, *90* (11), 1459−1462.

(20) Parson, R. Visualizing the Variation Principle: An Intuitive Approach to Interpreting the Theorem in Geometric Terms. *J. Chem. Educ.* **1993**, *70* (2), 115−119.

(21) Zúñiga, J.; Bastida, A.; Requena, A. Using the Screened Coulomb Potential to Illustrate the Variational Method. *J. Chem. Educ.* **2012**, *89* (2), 1152−1158.

(22) Goodfriend, P. L. Simple Perturbation Example for Quantum Chemistry. *J. Chem. Educ.* **1985**, *62* (3), 202−204.

(23) Aebersold, D. Integral Equations in Quantum Chemistry. *J. Chem. Educ.* **1975**, *52* (7), 434.

(24) Lucas, J. M.; Mota, F.; Novoa, J. J. Theoretical Study of the Vibrational-Rotational Spectra of Diatomic Molecules: A Quantum Chemistry Experiment. *J. Chem. Educ.* **1986**, *63* (10), 919−920.

(25) Weininger, S. J. The Molecular Structure Conundrum: Can Classical Chemistry be Reduced to Quantum Chemistry? *J. Chem. Educ.* **1984**, *61* (11), 939−944.

(26) Li, W.-K.; Blinder, S. M. Introducing Relativity into Quantum Chemistry. *J. Chem. Educ.* **2011**, *88* (1), 71−73.

(27) El-Issa, H. D. The Particle in a Box Revisited. *J. Chem. Educ.* **1986**, *63* (9), 761−764.

(28) Pettitt, B. A. The Morse Oscillator and Second-Order Perturbation Theory. *J. Chem. Educ.* **1998**, *75* (9), 1170.

(29) Besalú, E.; Martí, J. Exploring the Rayleigh-Ritz Variational Principle. *J. Chem. Educ.* **1998**, *75* (1), 105−107.

(30) Ma, N. L. Quantum Analogies on Campus. *J. Chem. Educ.* **1996**, *73* (11), 1016−1017.

(31) Hall, P. G. An Introduction to Quantum-Mechanical Operators. *J. Chem. Educ.* **1966**, *43* (1), 38−39.

(32) Cook, D. B. *Handbook of Computational Quantum Chemistry*, 2nd ed.; Dover: New York, 2005; pp 217−251.

(33) Jensen, F. Atomic Orbital Basis Sets. *WIREs Comput. Mol. Sci.* **2013**, *3* (3), 273−295.

(34) Pye, C. C.; Mercer, C. J. On the Least-Squares Fitting of Slater-Type Orbitals with Gaussians: Reproduction of the STO-NG Fits Using Microsoft Excel and Maple. *J. Chem. Educ.* **2012**, *89* (11), 1405−1410.

(35) Schmidt, M. W.; Baldridge, K. K.; Boatz, J. A.; Elbert, S. T.; Gordon, M. S.; Jensen, J. H.; Koseki, S.; Matsunaga, N.; Nguyen, K. A.; Su, S.; Windus, T. L.; Dupuis, M.; Montgomery, J. A. General Atomic and Molecular Electronic Structure System. *J. Comput. Chem.* **1993**, *14*, 1347−1363.

(36) Williams, L.; Kessler, R. R.; Cunningham, W.; Jeffries, R. Strengthening the Case for Pair Programming. *IEEE Software* **2000**, *17* (4), 19−25.