Journal of Computational Physics **156,** 257–271 (1999)
Article ID jcph.1999.6361, available online at http://www.idealibrary.com on IDEAL®

Integration by Cell Algorithm for Slater Integrals in a Spline Basis

Yanghui Qiu and Charlotte Froese Fischer

Vanderbilt University, Box 1679B, Nashville, Tennessee 37235
E-mail: cff@vuse.vanderbilt.edu

Received March 29, 1999; revised July 21, 1999

An algorithm for evaluating Slater integrals in a B-spline basis is introduced. Based on the piecewise property of the B-splines, the algorithm divides the two-dimensional (r_1, r_2) region into a number of rectangular cells according to our chosen grid and implements the two-dimensional integration over each individual cell using Gaussian quadrature. Over the off-diagonal cells, the integrands are separable so that each two-dimensional cell-integral is reduced to a product of two one-dimensional integrals. Furthermore, the scaling invariance of the B-splines in the logarithmic region of the chosen grid is fully exploited such that only some of the cell integrations need to be implemented. The values of given Slater integrals are obtained by assembling the cell integrals. This algorithm significantly improves the efficiency and accuracy of the traditional method that relies on the solution of differential equations and renders the B-spline method more effective when applied to multi-electron atomic systems.

Key Words: Slater integral; B-spline.

1. INTRODUCTION

Critical to the success of the many-electron atomic structure calculation is an accurate and efficient evaluation of the Slater integrals. The Slater integrals are integrations of a product of four atomic radial functions and a coupling weighting factor over the two-dimensional radial region, i.e.,

$$R^k(a, b; c, d) = \int_0^\infty \int_0^\infty \frac{r_<^k}{r_>^{k+1}} P_a(r_1) P_b(r_2) P_c(r_1) P_d(r_2)\, dr_1\, dr_2, \tag{1}$$

where P_a, P_b, P_c, and P_d are the radial functions of atomic orbitals, k is an integer, and $r_<, r_>$ are the $\min(r_1, r_2)$ and $\max(r_1, r_2)$, respectively. The radial function of orbital a is

0021-9991/99 $30.00

approximated by

$$P_a(r) \approx \sum_i a_i B_i(r), \tag{2}$$

where $B_i(r)$ is the basis function and a_i the expansion coefficient.

Splines as basis functions in many-electron atomic systems were introduced over 20 years ago when Altenberger-Siczek and Gilbert [1] investigated the two-electron helium atom. Under the spline basis, the Slater integral becomes

$$R^k(a, b; c, d) = \sum_i \sum_j \sum_{i'} \sum_{j'} a_i b_j c_{i'} d_{j'} R^k(i, j; i', j'), \tag{3}$$

where the Slater matrix element $R^k(i, j; i', j')$ is

$$R^k(i, j; i', j') = \int_0^\infty \int_0^\infty \frac{r_<^k}{r_>^{k+1}} B_i^{k_s}(r_1) B_j^{k_s}(r_2) B_{i'}^{k_s}(r_1) B_{j'}^{k_s}(r_2)\, dr_1\, dr_2. \tag{4}$$

In their work they used both cardinal splines and B-splines. Because of the apparently large number of numerical operations needed to evaluate this integral accurately, they finally concluded that B-splines were not suitable for atomic structure calculations.

However, there has recently been a renewed interest in the use of B-splines in complex atomic systems: Bottcher and Strayer [2] applied the B-splines to time-dependent problems, Johnson and co-workers to many-body perturbation theory [3–5], Fischer and co-workers [6–9] to Hartree–Fock calculations and continuum problems, Hansen and co-workers to orthogonal operators and Rydbergy series [10, 11], Decleva and co-workers [12–14] to multichannel continuum problems, and van der Hart to R-matrix theory [15]. More detailed discussions can be found in a recent review by Sapirstein and Johnson [16].

An important development was finding a method for evaluating the Slater matrix elements [8, 9]. In this approach, the Slater matrix elements $R^k(i, j; i', j')$ are rewritten as

$$R^k(i, j; i', j') = \int_0^\infty \frac{1}{r_1} B_i^{k_s}(r_1) Y_{jj'}^k(r_1) B_{i'}^{k_s}(r_1)\, dr_1, \tag{5}$$

where

$$Y_{jj'}^k(r_1) = r_1 \int_0^\infty B_j^{k_s}(r_2) \frac{r_<^k}{r_>^{k+1}} B_{j'}^{k_s}(r_2)\, dr_2. \tag{6}$$

The function $Y_{jj'}^k(r)$ can be computed by solving the following differential equation for the given boundary conditions

$$\begin{aligned} \frac{d^2 Y_{jj'}^k(r)}{dr^2} &= \frac{k(k+1)}{r^2} Y_{jj'}^k(r) - \frac{2k+1}{r} B_j^{k_s}(r) B_{j'}^{k_s}(r) \\ Y_{jj'}^k(0) &= 0 \\ \frac{d}{dr} Y_{jj'}^k(r) &= -\frac{k}{r} Y_{jj'}^k(r) + B_j^{k_s}(r) B_{j'}^{k_s}(r) \qquad \text{as } r \to +\infty. \end{aligned} \tag{7}$$

Then the Slater matrix element in Eq. (5) is obtained by direct integration. The Slater

integrals as a quadruple summation of the matrix elements are subsequently assembled. In order to avoid the two-point boundary value problem, Hartree [17] replaced the second-order differential equation by a pair of first-order equations that could be integrated outward from the origin and inward from large r, a method suitable for numerical integration. However, when used in spline-based applications, the special characteristics of the B-splines cannot readily be exploited. In this paper, we have introduced and implemented an algorithm which is designed specifically for the spline-based applications. It takes advantage of the piecewise property and scaling invariance of the B-splines and implements the integration in Eq. (4) by cells. Since the system and user time spent on the evaluation of the Slater matrix elements is much more significant than that spent on assembling the Slater integrals from the matrix elements, significant improvements in both efficiency and accuracy in evaluating the Slater integrals are observed under the current algorithm.

2. SLATER INTEGRALS IN B-SPLINES

2.1. *Definition of B-Splines*

Following de Boor [18], we divide the interval [0, R] into segments. The endpoints of these segments are given by a knot sequence t_i, $i = 1, 2, \ldots, n + k_s$. The B-splines of order k_s, $B_i^{k_s}(r)$, are a set of piecewise polynomials defined on the knot sequence recursively by the relations

$$B_i^1(r) = \begin{cases} 1, & t_i \le r < t_{i+1} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

and

$$B_i^{k_s}(r) = \frac{r - t_i}{t_{i+k_s-1} - t_i} B_i^{k_s-1}(r) + \frac{t_{i+k_s} - r}{t_{i+k_s} - t_{i+1}} B_{i+1}^{k_s-1}(r). \tag{9}$$

The function $B_i^{k_s}(r)$ is a piecewise polynomial of degree $k_s - 1$ inside the interval $t_i \le r < t_{i+k_s}$ which vanishes outside this interval. The sum at any point r of all of the B-splines is unity [19], i.e.,

$$\sum_i B_i^{k_s}(r) = 1. \tag{10}$$

The set of B-splines of order k_s on the knot sequence t_i forms a complete basis for piecewise polynomials of degree $k_s - 1$ on the interval spanned by the knot sequence. The knots defining our grid have k_s-fold multiplicity at the endpoints, namely $t_1 = t_2 = \cdots = t_{k_s} = 0$ and $t_{n+1} = t_{n+2} = \cdots = t_{n+k_s} = R$. When multiple knots are encountered, limiting forms of the above recursive definition of the B-splines must be used. For $k_s > 1$, the B-splines generally vanish at their endpoints. However, at $r = 0$ the first B-spline is equal to 1 with all others vanishing and at $r = R$ the last B-spline shows the same behavior as the first B-spline at the origin. The two end B-splines are generally related to the boundary conditions of the problem under investigation [20]. A set of B-splines of order 5 in the region [0, 1] is shown in Fig. 1.

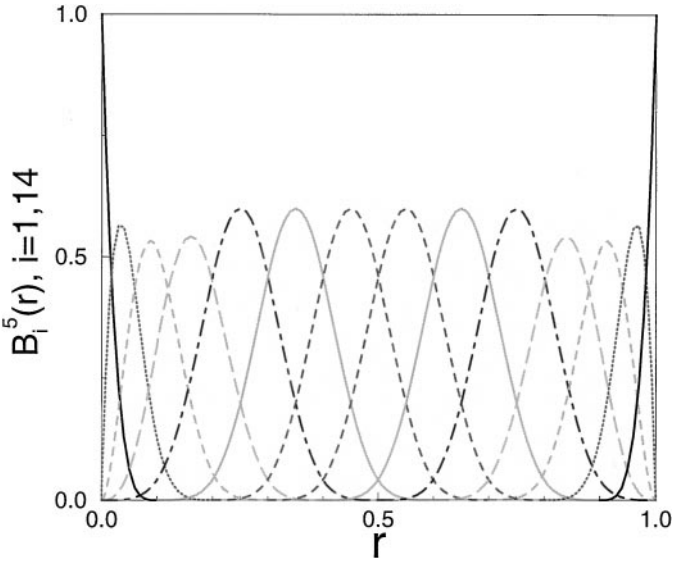

FIG. 1. The set of B-splines of order 5 in the region [0, 1] on the knot sequence $t_1 = t_2 = \cdots = t_5 = 0$, $t_i = t_{i-1} + 0.1$ for $i = 6, 7, \cdots, 15$ and $t_{15} = t_{16} = \cdots = t_{19} = 1$.

2.2. *A General Grid for Atomic Systems*

One of the advantages of a spline basis is that the choice of grid may be tailored to the problem under investigation. A different type of grid can easily be used in different regions. In atomic physics calculation, a general grid for both bound and continuum states for arbitrary nuclear charges was introduced by Fischer [7, 20] and subsequently applied successfully in the calculation of photoionization [12, 21]. In the grid definition, four different parameters are introduced. They are the step size $h = 2^{-m}$ with m an integer, the maximum step size h_{max}, the maximum range R, and the order of splines k_s. The grid points are defined through the array t_i such that

$$\begin{aligned} t_i &= 0, \qquad \text{for } i = 1, \ldots, k_s \\ t_{i+1} &= t_i + h, \qquad \text{for } i = k_s, \ldots, k_s + m \\ t_{i+1} &= t_i(1 + h), \qquad \text{for } 1 \le t_{i+1} - t_i < h_{max} \\ t_{i+1} &= t_i + h_{max}, \qquad \text{for } t_i < ZR \\ r_i &= t_i/Z, \qquad \text{for all } i, \end{aligned}$$

where r is the radial coordinate and Z is the nuclear charge. The rules for setting such a knot sequence can be found in Ref. [20]. Briefly speaking, since the Hamiltonians of atomic systems and the corresponding orbital wavefunctions scale approximately with respect to the nuclear charge Z, the knot sequence is defined for the variable $t = Zr$. For the continuum calculation, the logarithmic grid is shown to be a good choice for small r, except near the origin, where a constant grid can avoid the large numbers of small intervals. Meanwhile, the logarithmic grid results in a very large step size for large r, which can be larger than the wavelength of the oscillatory continuum wavefunctions, therefore a constant grid in the large r region is used.

2.3. *Symmetry of the Slater Matrix Elements*

The Slater matrix elements have two types of symmetries—they are invariant under the exchange of certain indices and are scaling invariant under coordinate displacement within the exponential region.

2.3.1. Exchange symmetry of the Slater matrix elements. From the definition of the Slater matrix elements, it is trivial to show that the exchange of indices i and i', j and j', and $\{i, i'\}$ and $\{j, j'\}$ does not affect the result of the Slater matrix element, i.e.,

$$\begin{aligned} R^k(i, j; i', j') &= R^k(i', j; i, j') \\ &= R^k(i, j'; i', j) \\ &= R^k(j, i; j', i') \\ &= R^k(j', i; j, i') \\ &= R^k(j, i'; j', i) \\ &= R^k(j', i'; j, i) \\ &= R^k(i', j'; i, j). \end{aligned}$$

Therefore, only about 1/8 of the Slater matrix elements need to be evaluated.

2.3.2. Scaling laws of the Slater matrix elements. The grid with an exponentially increasing interval length results in several properties that can be exploited to avoid redundant calculations. Suppose the splines discussed lie entirely within the exponential region. Since the splines are normalized such that the sum of the values at any point in the range [0, R] is equal to unity, a simple displacement invariance applies. Let the left-most knot defining $B_i^{k_s}(r)$ be t_i and let $r = t_i + s$. Then

$$B_i^{k_s}(t_i + s) = B_{i+1}^{k_s}((1 + h)(t_i + s)) = B_{i+1}^{k_s}(t_{i+1} + s(1 + h)). \tag{11}$$

From the displacement invariance of the splines, several scaling laws exist [22]. The ones relevant to the Slater matrix elements are

$$\left\langle B_{i+1}^{k_s}(r)|r^k|B_{j+1}^{k_s}(r)\right\rangle = (1 + h)^{1+k}\left\langle B_i^{k_s}(r)|r^k|B_j^{k_s}(r)\right\rangle \tag{12}$$

$$R^k(i + 1, j + 1; i' + 1, j' + 1) = (1 + h)R^k(i, j; i', j'). \tag{13}$$

However, the most valuable properties to the current integration by cell method are the scaling laws over individual cells, i.e.,

$$\int_{r_{iv+1}}^{r_{iv+2}} B_{i+1}^{k_s}(r)B_{j+1}^{k_s}(r)r^k\,dr = (1 + h)^{1+k}\int_{r_{iv}}^{r_{iv+1}} B_i^{k_s}(r)B_j^{k_s}(r)r^k\,dr \tag{14}$$

and

$$\begin{aligned} &\int_{r_{iv+1}}^{r_{iv+2}}\int_{r_{jv+1}}^{r_{jv+2}} \frac{r_<^k}{r_>^{k+1}} B_{i+1}^{k_s}(r_1)B_{j+1}^{k_s}(r_2)B_{i'+1}^{k_s}(r_1)B_{j'+1}^{k_s}(r_2)\,dr_1\,dr_2 \\ &\quad = (1 + h)\int_{r_{iv}}^{r_{iv+1}}\int_{r_{jv}}^{r_{jv+1}} \frac{r_<^k}{r_>^{k+1}} B_i^{k_s}(r_1)B_j^{k_s}(r_2)B_{i'}^{k_s}(r_1)B_{j'}^{k_s}(r_2)\,dr_1\,dr_2. \end{aligned} \tag{15}$$

A detailed proof of the scaling laws over a cell is given in Appendix A of Ref. [23].

3. INTEGRATION BY CELL ALGORITHM

The distinguishing feature of the B-splines is their piecewise property. Every spline is a positive polynomial over a finite range. If k_s order B-splines are chosen, the spline $B_i^{k_s}$ is non-trivial only in the range from knot i to knot $i + k_s$, i.e., the spline $B_i^{k_s}$ is non-zero only in intervals MAX$(1, i - k_s + 1)$, MAX$(1, i - k_s + 1) + 1, \ldots,$ MIN(i, nv), where nv is the total number of intervals. This feature leads to an important constraint such that at any interval iv in the one-dimensional region, the number of splines whose values are not zero is always k_s. In fact the non-zero ones are the splines $iv, iv + 1, \ldots,$ and $iv + ks - 1$. Other splines at interval iv simply vanish. When this property is applied to the Slater matrix elements

$$R^k(i, j; i', j') = \int_0^\infty \int_0^\infty \frac{r_<^k}{r_>^{k+1}} B_i^{k_s}(r_1) B_j^{k_s}(r_2) B_{i'}^{k_s}(r_1) B_{j'}^{k_s}(r_2)\, dr_1\, dr_2, \tag{16}$$

$R^k(i, j; i', j') = 0$, if either $|i' - i| \geq k_s$ or $|j' - j| \geq k_s$. Moreover, the integration contributing to $R^k(i, j; i', j')$ extends over only the cells in which $B_i^{k_s}(r_1)$ and $B_{i'}^{k_s}(r_1)$, and $B_j^{k_s}(r_2)$ and $B_{j'}^{k_s}(r_2)$ overlap. Because of the symmetry of the Slater matrix elements, we can assume without loss of generality that $i \leq i'$, $j \leq j'$, and $i \leq j$. The area over which the integrand contributes to the Slater matrix element $R^k(i, j; i', j')$ is illustrated in Fig. 2. The area is a block of cells stretching from interval $i - k_s + 1$ to i' in the r_1 direction and from interval $j - k_s + 1$ to j' in the r_2 direction. The integration by cell algorithm exploits this feature thoroughly. Since there are only k_s splines which are not zero along the r_1 or r_2 direction in each cell, we can integrate all the non-trivial $\{i, i', j, j'\}$ combinations over each individual cell first. The Slater matrix elements are then obtained as a summation of the cell integrals.

3.1. *Integration over the Off-Diagonal Cells*

Over the off-diagonal cells the integrand in the Slater matrix element is separable and the integration limits are not coupled. Therefore, the two-dimensional integral is reduced

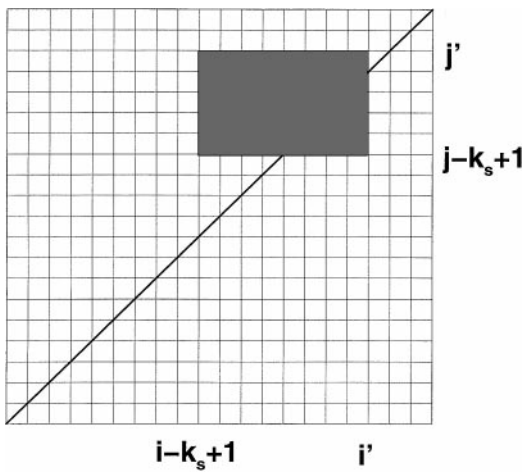

FIG. 2. Illustration of the area over which the integrand of the Slater matrix element $R^k(i, j; i', j')$ is non-trivial. The area is a block of cells stretching from interval $i - k_s + 1$ to i' in the r_1 direction and from interval $j - k_s + 1$ to j' in the r_2 direction.

to a product of two one-dimensional integrals. Assume $iv < jv$. Then

$$\begin{aligned} R^k(i, j; i', j'; iv, jv) &= \int_{r_{jv}}^{r_{jv+1}} \int_{r_{iv}}^{r_{iv+1}} \frac{r_<^k}{r_>^{k+1}} B_i^{k_s}(r_1) B_j^{k_s}(r_2) B_{i'}^{k_s}(r_1) B_{j'}^{k_s}(r_2)\, dr_1\, dr_2 \\ &= \int_{r_{iv}}^{r_{iv+1}} r_1^k B_i^{k_s}(r_1) B_{i'}^{k_s}(r_1)\, dr_1 \int_{r_{jv}}^{r_{jv+1}} \frac{1}{r_2^{k+1}} B_j^{k_s}(r_2) B_{j'}^{k_s}(r_2)\, dr_2 \\ &= r^k(i, i'; iv) \times r^{-(k+1)}(j, j'; jv), \end{aligned} \tag{17}$$

where

$$r^k(i, i'; iv) = \int_{r_{iv}}^{r_{iv+1}} r_1^k B_i^{k_s}(r_1) B_{i'}^{k_s}(r_1)\, dr_1, \tag{18}$$

$$r^{-(k+1)}(j, j'; jv) = \int_{r_{jv}}^{r_{jv+1}} \frac{1}{r_2^{k+1}} B_j^{k_s}(r_2) B_{j'}^{k_s}(r_2)\, dr_2. \tag{19}$$

Although there are $n_v(n_v - 1)$ off-diagonal cells, we only need to calculate and store the n_v two-dimensional moment arrays $r^k(i, i'; iv)$ and $r^{-(k+1)}(j, j'; jv)$ for later assembly. The arrays of $r^k(i, i'; iv)$ and $r^{-(k+1)}(i, i'; iv)$ with $i, i' = 1, \ldots, k_s$ and $iv = 1, \ldots, n_v$ can be evaluated effectively using Gaussian quadrature with k_s Gaussian points. Note that because

$$r^k(i, i'; iv) = r^k(i', i; iv) \tag{20}$$

$$r^{-(k+1)}(i, i'; iv) = r^{-(k+1)}(i', i; iv), \tag{21}$$

only $r^k(i, i'; iv)$ and $r^{-(k+1)}(i, i'; jv)$ such that $i \le i'$ need to be evaluated. Moreover, the evaluation can be further reduced by using the scaling law in Eq. (14) in the logarithmic grid region. The storage of $r^k(i, i'; iv)$ and $r^{-(k+1)}(i, i'; jv)$ takes $n_v k_s^2$ locations in memory, ignoring symmetry, though only about half these need to be computed.

3.2. *Integration over the Diagonal Cells*

Over the diagonal cells the two coordinates in the integrand (the integration limits) of the Slater matrix elements are coupled. Moreover, the integrand is not continuous across the cell diagonal where $r_1 = r_2$. Although there are only n_v cells, the diagonal cell integration is the most CPU intensive part in the evaluation of the Slater matrix element and it is also the major barrier towards achieving highly accurate matrix elements.

We separate the rectangular cell into two triangles so that the integrand inside each triangle is continuous. The integration over the rectangular cell is then a summation of integrations over the two triangles, which are symmetric with respect to index exchanges, i.e.,

$$\begin{aligned} R^k(i, j; i', j'; iv) &= \int_{r_{iv}}^{r_{iv+1}} \int_{r_{iv}}^{r_{iv+1}} \frac{r_<^k}{r_>^{k+1}} B_i^{k_s}(r_1) B_j^{k_s}(r_2) B_{i'}^{k_s}(r_1) B_{j'}^{k_s}(r_2)\, dr_1\, dr_2 \\ &= \int_{r_{iv}}^{r_{iv+1}} B_i^{k_s}(r_1) B_{i'}^{k_s}(r_1)\, dr_1 \cdot \left\{ \frac{1}{r_1^{k+1}} \int_{r_{iv}}^{r_1} r_2^k B_j^{k_s}(r_2) B_{j'}^{k_s}(r_2)\, dr_2 \right. \\ &\quad \left. + r_1^k \int_{r_1}^{r_{iv+1}} \frac{1}{r_2^{k+1}} B_j^{k_s}(r_2) B_{j'}^{k_s}(r_2)\, dr_2 \right\} \\ &= R_\Delta^k(i, j, i', j'; iv) + R_\Delta^k(j, i, j', i'; iv) \end{aligned} \tag{22}$$

where

$$R^k_\Delta(i,\, j,\, i',\, j';\, iv) = \int_{r_{iv}}^{r_{iv+1}} B_i^{k_s}(r_1) B_{i'}^{k_s}(r_1)\, dr_1 \frac{1}{r_1^{k+1}} \int_{r_{iv}}^{r_1} r_2^k B_j^{k_s}(r_2) B_{j'}^{k_s}(r_2)\, dr_2. \qquad (23)$$

We use Gaussian quadrature again to do the integration over the triangular cell in Eq. (23). The key for an effective evaluation is to choose Gaussian grid points such that the available B-spline values used in the one-dimensional integration can be exploited and the new B-spline values needed to achieve the required accuracy are minimized. We know that n point Gaussian integration has $2n-1$ degrees of accuracy. It is obvious that at least a k_s by k_s Gaussian grid with respect to both coordinates is needed to achieve the highest algebraic accuracy for the integrals in Eq. (23) for $k=0$. Therefore, we apply the original Gaussian points used in the evaluation of one-dimensional integrals, $r^k(i, i'; iv)$ and $r^{-(k+1)}(j, j'; jv)$, to the two-dimensional integration with respect to coordinate r_1. In the two-dimensional integration regarding coordinate r_2, we also use k_s Gaussian points. A graphical representation of the chosen Gaussian points for $k_s = 8$ is shown in Fig. 3. The star is the original Gaussian point used in the one-dimensional integration and the circle is the Gaussian point used in current triangular cell integration. With this two-dimensional grid, the difficulty of the cell integrals near the origin because of the singularity of the integrand is minimized and uniformly accurate results for all the Slater integrals can be obtained. The evaluation can be further reduced by using the scaling law in Eq. (15) in the logarithmic grid region. The full storage of $R^k_\Delta(i, j, i', j'; iv)$ takes $n_v k_s^4$ locations in memory, ignoring symmetry ($i \le i'$, $j \le j'$, $i \le j$) though now only about $1/8$ of the values need to be computed. Practical calculation demonstrates the effectiveness of current choice of grid points (see later section).

3.3. *Assembly of the Cell Integrals*

Once all the cell integrals $r^k(i, i'; iv)$, $r^{-(k+1)}(j, j'; jv)$, and $R^k_\Delta(i, j, i', j'; iv)$ are prepared, the Slater matrix elements in Eq. (16) can be assembled straightforwardly. A

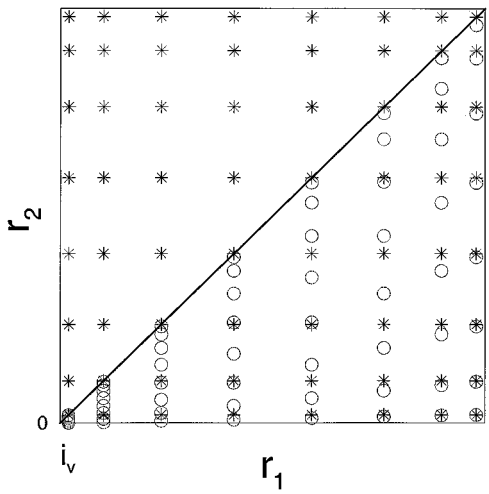

FIG. 3. Graphical representation of the Gaussian points for $k_s = 8$. The star is the original Gaussian point used in the one-dimensional integration and the circle is the Gaussian point used in current two-dimensional cell integration.

FORTRAN 90 code for evaluating the cell integrals and assembling the Slater matrix elements from the cell integrals can be found in Ref. [23]. The computational effort of putting the pieces together is almost trivial. Since the Slater matrix elements need more space to store than the cell integrals, it is more efficient to store the cell integrals directly. When need for the Slater matrix elements arises, we can assemble them immediately.

Once all of the Slater matrix elements are assembled, the value of any Slater integral in Eq. (3) can be easily obtained.

3.4. *An Alternative to the Diagonal-Cell Integration*

The integration over a diagonal cell in Eq. (22) can be implemented in an alternative way. The cell integral can be rewritten as

$$R^k(i,j;i',j';iv) = \int_{r_{iv}}^{r_{iv+1}} dr_1 r_1^k B_i^{k_s}(r_1) B_{i'}^{k_s}(r_1) \int_{r_{iv}}^{r_{iv+1}} dr_2 \frac{1}{r_2^{k+1}} B_j^{k_s}(r_2) B_{j'}^{k_s}(r_2)$$
$$+ \int_{r_{iv}}^{r_{iv+1}} dr_1 B_i^{k_s}(r_1) B_{i'}^{k_s}(r_1) \int_{r_{iv}}^{r_1} dr_2 B_j^{k_s}(r_2) B_{j'}^{k_s}(r_2) \left\{ \frac{r_2^k}{r_1^{k+1}} - \frac{r_1^k}{r_2^{k+1}} \right\}, \tag{24}$$

where the first term in Eq. (24) is a product of two one-dimensional integrals which are evaluated during the off-diagonal cell integration in Eq. (17) and the second term is a coupled two-dimensional integral over a triangular cell. The second term has a nice property such that the integrand is zero along the hypotenuse of the cell and becomes most significant at the low-right corner of the cell (Fig. 4). To exploit this property, we make a coordinate

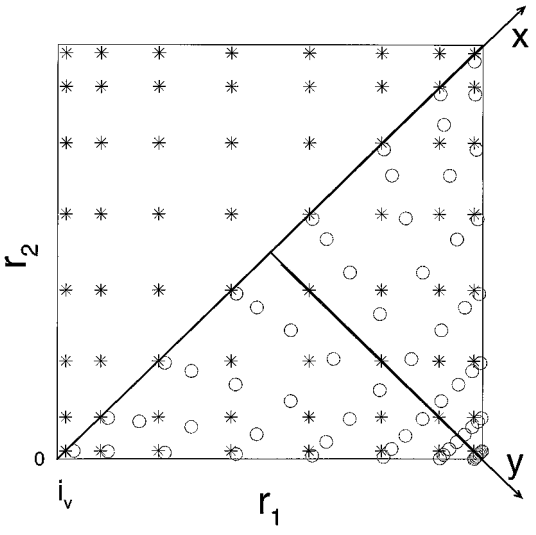

FIG. 4. Graphical representation of the Gaussian points for $k_s = 8$. The stars represent the points that would be used for two-dimensional integration over the square using Gaussian quadrature in each dimension and the circles represent the points used in the current two-dimensional cell integration over the triangle.

rotation

$$x = \frac{r_1 + r_2}{\sqrt{2}}$$
$$y = \frac{r_1 + r_2}{\sqrt{2}}, \tag{25}$$

where the new coordinates x and y are always positive in the integration area. The second term in Eq. (24) is then transformed as

$$R^k(i, j; i', j'; iv)|_{term2} = \sqrt{2} \int_0^{\frac{r_{iv+1} - r_{iv}}{\sqrt{2}}} dy \int_{\sqrt{2} r_{iv} + y}^{\sqrt{2} r_{iv+1} - y} dx B_i^{k_s}\left(\frac{x+y}{\sqrt{2}}\right) B_{i'}^{k_s}\left(\frac{x+y}{\sqrt{2}}\right)$$
$$\cdot B_j^{k_s}\left(\frac{x-y}{\sqrt{2}}\right) B_{j'}^{k_s}\left(\frac{x-y}{\sqrt{2}}\right)\left\{\frac{(x-y)^k}{(x+y)^{k+1}} - \frac{(x+y)^k}{(x-y)^{k+1}}\right\}. \tag{26}$$

Again we use two-dimensional Gaussian quadrature to perform the integration over the triangular cell. In Fig. 4 a graphical representation of the adapted optimal Gaussian points is shown. The stars represent the original Gaussian points that would be used in a two-dimensional integration over the square and the circles represent the Gaussian point used in the current triangular cell integration over the triangle. With this two-dimensional grid, greater weight of integration is located at the low-right corner of the triangular cell where the integrand is most significant. This approach however has drawbacks. Note that the transformation results in a singularity in the integrands when $x = y$ (or $r_2 = 0$) and also when $x = -y$ (or $r_1 = 0$). The present scheme does not pay special attention to these singularities of the integrand at the origin. Moreover, the B-spline itself as a function of one variable r_1 or r_2 becomes a function of two variables x and y, which leads to more computational effort during the integration. As the results in the next section will show, this approach is not as accurate.

4. RESULTS AND DISCUSSIONS

We have implemented the two variants of the integration-by-cell algorithm for the Slater matrix elements in FORTRAN 90 and assembled the Slater integrals thereafter. The traditional algorithm based on solving the differential equation is also implemented for comparison. The Slater integral package we developed is divided into 4 modules.

define_grid This first module gets input parameters for the grid and sets up the knots for splines according to our grid scheme.

define_spline This second module initializes the splines and their derivatives and evaluates the hydrogenic matrix elements (the one-electron Hamiltonian operator in spline basis).

set_Slater_matrix_elements This third module sets up the Slater matrix elements for a given k with both the integration-by-cell algorithm and the traditional one.

test_Slater_integrals The fourth module determines a set of hydrogenic orbitals in terms of the spline basis and calculates a series of Slater integrals from the Slater matrix elements, all of the same k.

The first two modules are set up for general applications whereas the last two implement and test the current algorithm.

Extensive tests were performed for the evaluation of the Slater matrix elements and the Slater integrals for a variety of parameters but only a few sets are given here. The results are obtained using a grid with $Z = 1$, $h = 1/8$, and $1/4$, $R = 160$ a.u., and $k_s = 4$ and 8 where $nv = 52$ for $h = 1/8$ and $nv = 27$ for $h = 1/4$.

The calculations were performed on a Sun workstation (CPU, UltraSPARC 143 MHz; RAM, 64 MB) in double precision where the fractional part consists of 52 bits for an accuracy of 15 significant digits. The user and system time is returned using the system function **DTIME**(). We find the two schemes of the integration-by-cell algorithm are almost equally efficient in terms of the user and system time. However, the results evaluated with the second scheme mentioned in Subsection 3.4 are much less accurate for the *ns* Slater integrals such as $F^0(1s, 1s)$ and $G^0(1s, 2s)$ where the radial functions of the integrands are localized near the origin. For example, for the above parameters and $k_s = 8$, the value of $F^0(1s, 1s)$ evaluated with the second scheme is about 3 orders of magnitude less accurate. The reason for this is obvious. The Slater matrix integrand has a singularity at the origin. The difficulty of the cell integrals caused by the singularity at the origin is minimized by the chosen Gaussian grid points which are densely populated near the origin in the first scheme of the integration-by-cell algorithm while it is not treated with care in the second scheme.

In the following, we only compare the results evaluated with the first scheme of the integration-by-cell algorithm and those with the traditional method of solving the differential equations. The comparison of the time in setting up the Slater matrix elements is shown in Table I where t_1 is the time with the traditional method and t_2 is the time with the first scheme of the integration-by-cell algorithm including the time for evaluating the splines at the new points. We find that the integration by cell method is several times faster in

TABLE I

User and System Time in Seconds for Setting Up All of the Slater Matrix Elements for k_s = 4 and 8 on a Sun Workstation (CPU, UltraSPARC 143 MHz; RAM, 64 MB)

$h = 1/8$				$h = 1/4$			
k	t_1	t_2	t_1/t_2	k	t_1	t_2	t_1/t_2
	$k_s = 4$				$k_s = 4$		
0	1.1	.11	10.	0	.33	.041	7.9
1	1.1	.10	11.	1	.32	.038	8.3
2	1.1	.10	11.	2	.32	.038	8.3
3	1.1	.11	11.	3	.32	.039	8.3
4	1.1	.11	11.	4	.32	.039	8.3
6	1.1	.11	11.	6	.32	.039	8.3
	$k_s = 8$				$k_s = 8$		
0	10.	1.5	6.9	0	3.0	.74	4.0
1	10.	1.5	6.9	1	3.0	.74	4.0
2	10.	1.5	6.9	2	3.0	.74	4.0
3	10.	1.5	6.9	3	3.0	.74	4.0
4	10.	1.5	6.9	4	3.1	.74	4.1
6	10.	1.5	6.9	6	3.0	.74	4.0

Note. k, the order of the Slater integrals; k_s, the order of the B-splines; h, the starting step size of the grid; t_1, the time with the traditional method; t_2, the time with the integration by cell method.

TABLE II

Comparison of the Accuracy of Some F^k and G^k Integrals for Different Order of B-Splines k_s

F^k/G^k	Exact value	Difference 1	Difference 2	t_3(milliseconds)
		(a) $k_s = 4$		
$F^0(1s, 1s)$	5/8	−4.8(−09)	−3.0(−11)	2.3
$F^0(1s, 2s)$	17/81	−4.6(−10)	−2.4(−11)	4.1
$F^0(1s, 2p)$	59/243	1.6(−10)	−1.6(−11)	4.0
$F^0(2s, 2s)$	77/512	−1.3(−09)	−6.8(−11)	2.1
$F^0(2s, 2p)$	83/512	−6.8(−10)	−4.5(−11)	4.1
$F^0(2p, 2p)$	93/512	−6.6(−10)	−2.5(−11)	2.4
$F^0(4s, 4s)$	19541/524288	−3.8(−09)	−5.3(−10)	2.1
$F^0(4s, 4p)$	19943/524288	−3.2(−09)	−4.6(−10)	4.0
$F^0(4s, 4d)$	20693/524288	−1.3(−09)	−3.6(−10)	4.0
$F^0(4s, 4f)$	21743/524288	1.2(−10)	−2.8(−10)	4.1
$F^0(4p, 4p)$	20413/524288	−3.0(−09)	−3.9(−10)	2.2
$F^0(4p, 4d)$	21239/524288	−1.7(−09)	−2.9(−10)	4.0
$F^0(4p, 4f)$	22373/524288	−1.1(−12)	−2.1(−10)	3.9
$F^0(4d, 4d)$	22373/524288	−1.8(−09)	−1.9(−10)	2.2
$F^0(4d, 4f)$	23759/524288	−3.8(−10)	−1.2(−10)	3.9
$F^0(4f, 4f)$	26333/524288	−5.6(−10)	−4.3(−11)	2.2
$G^0(1s, 2s)$	16/729	−7.3(−10)	2.5(−12)	2.4
$G^0(2p, 3p)$	96768/9765625	−3.8(−10)	7.0(−12)	2.2
$G^0(2p, 4p)$	560/177147	−2.0(−10)	4.1(−12)	2.2
$G^1(1s, 2p)$	112/2187	−2.5(−10)	1.9(−12)	2.3
$G^1(2s, 2p)$	45/512	−9.9(−10)	−3.6(−11)	2.1
$G^1(2p, 3s)$	92016/9765625	−1.0(−09)	1.8(−11)	2.2
$G^1(2p, 3d)$	1824768/48828125	4.0(−11)	4.1(−12)	2.2
$G^1(2p, 4s)$	5168/1594323	−5.5(−10)	1.3(−11)	2.1
$G^1(2p, 4d)$	19120/1594323	−2.1(−10)	3.3(−12)	2.2
$F^2(4f, 4f)$	103275/3670016	−1.0(−09)	−3.7(−11)	2.4
$G^2(2p, 3p)$	110592/9765625	−8.9(−10)	2.3(−11)	2.2
$G^2(2p, 4p)$	2128/531441	−5.2(−10)	1.3(−11)	2.1
$G^2(2p, 4f)$	4784/1594323	−6.7(−11)	1.7(−12)	2.2
$G^3(2p, 3d)$	1064448/48828125	−3.6(−10)	1.0(−12)	2.2
$G^3(2p, 4d)$	3920/531441	−2.7(−10)	2.7(−12)	2.2
$F^4(4f, 4f)$	69003/3670016	−1.0(−09)	−2.8(−11)	2.3
$G^4(2p, 4f)$	1040/531441	−4.9(−11)	1.3(−12)	2.3
$F^6(4f, 4f)$	7293/524288	−6.9(−10)	−2.1(−11)	2.2
		(b) $k_s = 8$		
$F^0(1s, 1s)$	5/8	−4.2(−15)	4.4(−16)	12
$F^0(1s, 2s)$	17/81	−1.4(−15)	1.4(−16)	21
$F^0(1s, 2p)$	59/243	−1.9(−15)	−1.4(−16)	21
$F^0(2s, 2s)$	77/512	−7.2(−16)	5.6(−17)	12
$F^0(2s, 2p)$	83/512	−4.7(−16)	1.9(−16)	21
$F^0(2p, 2p)$	93/512	−8.9(−16)	0.0(+00)	12
$F^0(4s, 4s)$	19541/524288	−1.6(−14)	−6.8(−16)	12
$F^0(4s, 4p)$	19943/524288	−1.1(−14)	−5.1(−16)	21
$F^0(4s, 4d)$	20693/524288	−2.8(−15)	−2.8(−16)	21
$F^0(4s, 4f)$	21743/524288	8.8(−16)	−6.2(−17)	21
$F^0(4p, 4p)$	20413/524288	−8.9(−15)	−3.5(−16)	12
$F^0(4p, 4d)$	21239/524288	−3.3(−15)	−1.7(−16)	21

TABLE II—*Continued*

F^k/G^k	Exact value	Difference 1	Difference 2	t_3(milliseconds)
		(b) $k_s = 8$		
$F^0(4p, 4f)$	22373/524288	4.0(−16)	−9.7(−17)	21
$F^0(4d, 4d)$	22373/524288	−2.2(−15)	−6.9(−17)	12
$F^0(4d, 4f)$	23759/524288	−8.3(−17)	3.5(−17)	21
$F^0(4f, 4f)$	26333/524288	6.2(−17)	2.8(−17)	12
$G^0(1s, 2s)$	16/729	−6.6(−17)	2.8(−17)	12
$G^0(2p, 3p)$	96768/9765625	−1.1(−16)	5.2(−18)	12
$G^0(2p, 4p)$	560/177147	−7.0(−17)	1.3(−18)	12
$G^1(1s, 2p)$	112/2187	−5.3(−15)	2.1(−17)	12
$G^1(2s, 2p)$	45/512	−8.2(−14)	0.0(+00)	12
$G^1(2p, 3s)$	92016/9765625	−3.0(−15)	2.1(−17)	12
$G^1(2p, 3d)$	1824768/48828125	−6.9(−14)	−2.1(−17)	12
$G^1(2p, 4s)$	5168/1594323	−8.5(−16)	0.0(+00)	12
$G^1(2p, 4d)$	19120/1594323	−9.0(−15)	1.7(−18)	12
$F^2(4f, 4f)$	103275/3670016	−6.3(−14)	3.1(−17)	12
$G^2(2p, 3p)$	110592/9765625	−5.3(−16)	0.0(+00)	12
$G^2(2p, 4p)$	2128/531441	−2.7(−16)	−7.8(−18)	12
$G^2(2p, 4f)$	4784/1594323	−2.9(−16)	−1.3(−18)	12
$G^3(2p, 3d)$	1064448/48828125	−9.7(−17)	−1.7(−17)	12
$G^3(2p, 4d)$	3920/531441	−1.1(−16)	−4.3(−18)	12
$F^4(4f, 4f)$	69003/3670016	−7.8(−16)	3.8(−17)	12
$G^4(2p, 4f)$	1040/531441	−1.9(−17)	−1.7(−18)	12
$F^6(4f, 4f)$	7293/524288	−7.4(−16)	3.5(−17)	12

Note. The exact values of the Slater integrals are from Ref. [9]. Difference 1, the difference of the exact value and the one evaluated with the traditional method; Difference 2, the difference of the exact value and the one evaluated with the integration by cell method. t_3, the user and system time of assembling the Slater integral from the Slater matrix elements.

evaluating the Slater matrix elements than the traditional method by solving the differential equations for all the choices of k_s and the two cases of h (1/8 and 1/4). Since accuracy is also one of our major concerns, we compare the differences between the exact value and the results evaluated with the two methods in Table II, where Difference 1 is the deviation between the exact value and the Slater integral assembled from the Slater matrix elements which are evaluated with the traditional method and the input parameter $Z = 1$; Difference 2 is the deviation between the exact value and the Slater integral assembled from the Slater matrix elements with the integration-by-cell method and the same input parameters as Difference 1; and t_3 is the user and system time of assembling the Slater integral from the Slater matrix elements. We see that all of the Slater integrals obtained by the integration by cell method are systematically at least as accurate as the one obtained by the traditional method. Moreover, unlike the traditional method where some of the Slater integrals, such as $F^0(4s, 4s)$, $F^2(4f, 4f)$, are significantly less accurate than others, the integration by cell method gives uniformly accurate results for all the Slater integrals. When k_s increases the difference between the exact value of the Slater integrals and the one evaluated with the integration by cell method decreases. When $k_s = 8$, both values converge.

It is important to note that once all of the Slater matrix elements are prepared the user and system time t_3 in calculating the Slater integrals is trivial compared to the time needed

in evaluating the matrix elements. It is obvious from the Tables I, II that $t_3 \ll t_1$ and $t_3 \ll t_2$ for a given k_s. Therefore, the effectiveness in the evaluation of the Slater matrix elements demonstrated by the integration by cell method indeed significantly improves the evaluation of the Slater integrals.

5. CONCLUSION

We have developed an algorithm for evaluating Slater integrals in a spline basis (B-spline). The algorithm divides the two-dimensional radial region (r_1, r_2) into a number of rectangular cells according to our chosen grid and implements the two-dimensional integration over each individual cell using Gaussian quadrature in each dimension. Over the off-diagonal cells, the two-dimensional cell-integrals are reduced to a product of two one-dimensional integrals. Over each diagonal cell, the rectangular cell integral is transformed into two triangular cell integrals to overcome the discontinuity of the integrand and the available B-spline values used in the one-dimensional integration are reused. Furthermore, the scaling invariance of the B-splines in the logarithmic region of the chosen grid is fully exploited. The values of the Slater matrix elements and the given Slater integrals are obtained by assembling the cell integrals. This algorithm significantly improves the efficiency and accuracy of the traditional method of solving differential equations and renders the B-spline methods much more effective when applied to multi-electron atomic systems.

ACKNOWLEDGMENTS

This work was supported by the Division of Chemical Sciences, Office of Basic Energy Sciences, Office of Energy Research, U.S. Department of Energy.

REFERENCES

1. A. Altenberger-Siczek and T. L. Gilbert, Spline bases for atomic calculations, *J. Chem. Phys.* **64**, 432 (1976).
2. C. Bottcher and M. R. Strayer, Relativistic theory of fermions and classical field on a collocation lattice, *Ann. Phys. (N.Y.)* **175**, 64 (1987).
3. W. R. Johnson and J. Sapirstein, Computation of second-order many-body corrections in relativistic atoms, *Phys. Rev. Lett.* **57**, 1126 (1986).
4. W. R. Johnson, M. Idrees, and J. Sapirstein, Second-order energies and third-order matrix elements of alkali atoms, *Phys. Rev. A* **35**, 3218 (1987).
5. W. R. Johnson, S. A. Blundell, and J. Sapirstein, Finite basis sets for the Dirac equation constructed from B splines, *Phys. Rev. A* **35**, 3218 (1988).
6. C. Fischer and M. Idrees, Spline algorithms for continuum functions, *Comput. Phys.* **3**, 53 (1989).
7. C. Fischer and M. Idrees, Spline methods for resonances in photoionization cross sections, *J. Phys. B* **23**, 679 (1990).
8. C. Fischer and W. Guo, Spline algorithms for the Hartree–Fock equation for the helium ground state, *J. Comput. Phys.* **90**, 486 (1990).
9. C. Fischer, W. Guo, and Z. Shen, Spline methods for multiconfiguration Hartree–Fock calculations, *Int. J. Quantum. Chem.* **42**, 849 (1992).
10. Y. T. Shen, M. Landtman, and J. E. Hansen, Ab-initio calculation of orthogonal parameters for complex configurations using B-splines, *J. Phys. B* **23**, L121 (1990).
11. H. van der Hart and J. E. Hansen, Calculation of double-excited states in He, N^{5+} and N^{3+} using B-splines, *J. Phys. B* **25**, 41 (1992).

12. P. Decleva, A. Lisini, and M. Venuti, Multichannel continuum states by a least-squares approach in a spline basis: Application to He and H^- photoionization, *J. Phys. B* **27**, 4867 (1994).
13. M. Venuti, P. Decleva, and A. Lisini, Accurate multichannel continuum states by a general configuration interaction expansion in a B-spline basis: Application to He photoionization, *J. Phys. B* **29**, 5315 (1996).
14. M. Venuti and P. Decleva, Convergent multichannel continuum states by a general configuration interaction expansion in a B-spline basis: Application to H^- photodetachment, *J. Phys. B* **30**, 4839 (1997).
15. H. van der Hart, B-spline methods in R-matrix theory for scattering in two-electron systems, *J. Phys. B* **30**, 453 (1997).
16. J. Sapirstein and W. R. Johnson, The use of basis splines in theoretical atomic physics, *J. Phys. B* **29**, 5213 (1996).
17. C. Fischer, *The Hartree–Fock Method for Atoms* (Wiley, 1977) New York.
18. C. deBoor, *A Practical Guide to Splines* (Springer-Verlag, New York, 1978).
19. A. S. Umar, J. Wu, M. R. Strayer, and C. Bottcher, Basis-spline collocation method for the lattice solution of boundary value problems, *J. Comput. Phys.* **93**, 426 (1991).
20. T. Brage, C. F. Fischer, and G. Miecznik, Non-variational, spline-Galerkin calculations of resonance positions and widths, and photodetachment and photoionization cross sections for H^- and He, *J. Phys. B* **25**, 5289 (1992).
21. J. H. Xi and C. Fischer, Cross section and angular distribution for the photodetachment of $He^-(1s2s2p^4P^o)$ below the He($n = 4$) threshold, *Phys. Rev. A* **53**, 3169 (1996).
22. C. F. Fischer, Concurrent vector algorithms for spline solutions of the helium pair equation, *Int. J. Supercom. Appl.* **5**, 5 (1991).
23. Yanghui Qiu, *Integration by Cell Algorithm for Slater Integrals in a Spline Basis*, Thesis, Vanderbilt University, May 1999 (unpublished).