SIAM J. SCI. COMPUT.
Vol. 25, No. 4, pp. 1416–1428

EFFICIENT STORAGE SCHEME FOR PRECALCULATED WIGNER 3*J*, 6*J* AND GAUNT COEFFICIENTS*

J. RASCH† AND A. C. H. YU†

Abstract. Efficient storage schemes are presented for storing Clebsch–Gordan, Wigner $3j$ and $6j$ symbols, as well as Gaunt coefficients, which are the integral over three spherical harmonics. Use is hereby made of the large number of symmetries which these symbols exhibit. Computer codes have been written and benchmarked against well-known published programs which usually use recursion relations for the evaluation. It is shown that our codes can be an order of magnitude or more faster in execution speed, maintaining full double precision accuracy.

Key words. vector coupling coefficients, Wigner $3j$, $6j$ symbols, Gaunt coefficients

AMS subject classifications. 31.15, 02.70

DOI. 10.1137/S1064827503422932

1. Introduction.

In many applications of physical and chemical interest, where a problem is decomposed using basis functions or a partial wave analysis, one encounters vector coupling coefficients, i.e., Clebsch–Gordan, Wigner $3j$ and $6j$ symbols, as well as Gaunt coefficients, which are the integral over three spherical harmonics (see (4.1), section 4). The problem is that these vector coupling coefficients need to be calculated thousands if not millions of times [1], [2], [3]. Depending on the complexity of the program, it is not uncommon that these coefficients are recalculated many times so that the question immediately arises of whether it is possible to precalculate and store them in memory or on disk in order to speed up the calculation.

All three coefficients we are going to consider here depend on 6 parameters. Any straightforward storage scheme in a six dimensional array would require prohibitively large memory requirements. Furthermore, it would be extremely wasteful since a large proportion would be zero or contain identical values due to the various symmetry properties these coefficients possess.

Due to the complexity of the current quantum chemistry and physics codes, larger and larger values of the angular momentum quantum numbers l and m that enter the coefficients are being calculated. The analytic expressions that are known for these coefficients, however, contain very large factorials that are notoriously difficult to evaluate on a computer. It is therefore not surprising that a large number of schemes have been devised to deal with this problem [6], [8], [11], [17], [18]. These schemes may give high accuracy but are not always the fastest. Any efficient storage scheme would therefore not only alleviate the problem of speed but simultaneously the problem of precision, since these coefficients could be calculated as accurately as possible and stored without the overhead of recalculating them. In the following we would like to present such schemes for the $3j$, $6j$, and Gaunt coefficients.

2. Wigner 3*j* symbols and Clebsch–Gordan coefficients.

Wigner $3j$ symbols and Clebsch–Gordan coefficients are very closely related by the formula (see,

*Received by the editors February 13, 2003; accepted for publication June 27, 2003; published electronically December 19, 2003. This research was supported by the Nuffield Foundation (NAL/00383/A) and the Royal Society (G503/22389/SM and HA/ESEP/JP).
http://www.siam.org/journals/sisc/25-4/42293.html
†Department of Applied Mathematics and Theoretical Physics, Queen's University Belfast, Belfast, BT7 1NN, Northern Ireland (j.rasch@qub.ac.uk, a.yu@qub.ac.uk). The research of the second author was supported by a Ph.D. award from the Department of Education and Learning (DEL).

e.g., [4])

$$\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \frac{(-)^{j_1-j_2-m_3}}{\sqrt{2j_3+1}} (j_1 m_1\, j_2 m_2 | j_3 - m_3)\,. \tag{2.1}$$

We will therefore in the following concentrate on Wigner 3j symbols since they exhibit more symmetry properties than Clebsch–Gordan coefficients, which also helps in the numerical evaluation.

A common expression for the 3j symbol is [4]

$$\begin{aligned} \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} &= \epsilon(j_1,j_2,j_3)\Delta(j_1,j_2,j_3)\delta_{m_1+m_2+m_3,0}(-)^{j_1-j_2-m_3} \\ &\times \sqrt{(j_1+m_1)!(j_1-m_1)!(j_2+m_2)!(j_2-m_2)!(j_3+m_3)!(j_3-m_3)!} \\ &\times \sum_{k=k_{\min}}^{k_{\max}} \frac{(-)^k}{k!(j_1+j_2-j_3-k)!(j_1-m_1-k)!(j_2+m_2-k)!} \\ &\times \frac{1}{(j_3-j_2+m_1+k)!(j_3-j_1-m_2+k)!}, \end{aligned} \tag{2.2}$$

where $k_{\max} = \min(j_1+j_2-j_3, j_1-m_1, j_2+m_2)$ and $k_{\min} = \max(-j_3+j_2-m_1, -j_3+j_1+m_2, 0)$ such that nowhere does a factorial of a negative number appear. The symbol Δ is here and in the following defined as the triangle coefficient

$$\Delta(j_1,j_2,j_3) = \sqrt{\frac{(j_1+j_2-j_3)!(j_1-j_2+j_3)!(-j_1+j_2+j_3)!}{(j_1+j_2+j_3+1)!}}, \tag{2.3}$$

where

$$\epsilon(j_1,j_2,j_3) = \begin{cases} 1 & \text{if } (j_1,j_2,j_3) \text{ form a triangle,} \\ 0 & \text{otherwise.} \end{cases} \tag{2.4}$$

Formula (2.2) is only valid if j_i and m_i are both integers or both half-integers for $i = 1, 2, 3$ and if

$$J = j_1 + j_2 + j_3 \tag{2.5}$$

is an integer. Equation (2.2) and to some extent (2.3) symptomatically show the difficulties encountered in evaluating these coefficients. Even moderately small j values go beyond the accurate representation of 32 and 64 bit machines; for example, for 32 bit machines, problems occur for as low as $j_1 = j_2 = 7$ such that $(7+7)! = 87178291200 > 2^{32} = 4294967296$. Similarly, for 64 bit machines one has $(11+11)! = 112400072777607680000 > 2^{64} = 18446744073709551616$. At the same time, (2.2) is ideal for symbolic algebra packages, which will evaluate the sum over rational numbers exactly and the square root to any precision specified.

2.1. Symmetries. In the following we summarize the various symmetries known for Wigner 3j symbols that are in common use. In particular they are

(1) invariant under any permutation of the columns (with the exception of a sign

change)

$$\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \begin{pmatrix} j_3 & j_1 & j_2 \\ m_3 & m_1 & m_2 \end{pmatrix} = \begin{pmatrix} j_2 & j_3 & j_1 \\ m_2 & m_3 & m_1 \end{pmatrix} \qquad \text{cyclic} \tag{2.6}$$

$$= (-)^J \begin{pmatrix} j_3 & j_2 & j_1 \\ m_3 & m_2 & m_1 \end{pmatrix} = (-)^J \begin{pmatrix} j_1 & j_3 & j_2 \\ m_1 & m_3 & m_2 \end{pmatrix} \tag{2.7}$$

$$= (-)^J \begin{pmatrix} j_2 & j_1 & j_3 \\ m_2 & m_1 & m_3 \end{pmatrix} \qquad \text{anticyclic;}$$

(2) invariant under space inflection, i.e.,

$$\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = (-)^J \begin{pmatrix} j_1 & j_2 & j_3 \\ -m_1 & -m_1 & -m_1 \end{pmatrix}; \tag{2.8}$$

(3) symmetric with respect to the additional symmetries based on the work of [5];
(4) zero for l_1, l_2, l_3 not fulfilling triangle relation;
(5) zero for $m_1 + m_2 + m_3 \neq 0$;
(6) zero for violating any one of the conditions

$$l_1 \geq |m_1|, \qquad l_2 \geq |m_2|, \qquad l_3 \geq |m_3|. \tag{2.9}$$

As can be seen, $3j$ symbols are zero for a wide range of parameter values. Any efficient storage scheme should take this into account.

The first two symmetries (1), (2) are well known and usually found and used in the literature and account for 12 symmetries. However, additional symmetries were found by Regge [5] showing that the $3j$ symbol has a total of 72 symmetries. These can best be displayed by the definition of a Regge symbol which assumes the properties of a magic square:

$$R := \boxed{\begin{array}{ccc} -j_1 + j_2 + j_3 & j_1 - j_2 + j_3 & j_1 + j_2 - j_3 \\ j_1 - m_1 & j_2 - m_2 & j_3 - m_3 \\ j_1 + m_1 & j_2 + m_2 & j_3 + m_3 \end{array}}. \tag{2.10}$$

In contrast to the usual definition of a magic square, we only require that all rows and columns have the same sum, but no conditions are placed on the diagonals (see section 2.2.1). The 72 symmetries now correspond to 3! row and 3! column interchanges plus a transposition of the matrix. For odd row or column permutations of the matrix the $3j$ symbol has to be multiplied by $(-)^J$. The permutation symmetries (1) and the space inflection (2) correspond to the permutation of columns and the exchange of the second and third row of the magic square (2.10). Tables of explicit formulas for the $3j$ symbol corresponding to the transposition of the square and the additional row interchanges can be found in [7]. Use of the magic square has been made to evaluate $3j$ symbols numerically [8]. More properties of magic squares are also given in section 2.2.1, where these properties are used for devising an efficient storage scheme. With these properties one has a unique one-to-one mapping between R and the corresponding $3j$ symbol [13].

2.2. Storing Wigner 3*j* symbols. Wigner $3j$ symbols with integer values for j_1, j_2, j_3 can be stored in a similar fashion as Gaunt coefficients (see section 4.2 for more details). However, the memory storage scheme which we devise in the following can also deal with half-integer values and is based on magic squares (2.10) as introduced in [5]. We shall first give a short summary of properties of magic squares which can be used for an efficient storage scheme.

2.2.1. Magic squares. We shall consider here only the 3×3 magic square. The usual condition is that the sum of all row and column entries adds up to the same number J. This would give 6 conditions for the 9 possible entries; however, only 5 of them are linearly independent. We will furthermore consider a semimagic square where we do not place any restraints on the diagonal which would give another 2 conditions. Since the sum J is itself a free variable we are left with $9-5+1=5$ independent variables. This is what we would expect since condition (5) eliminates one of the parameters m_i.

It is now possible to deduce further information about those 5 independent variables [8]. In particular, one can choose the smallest S and largest L entries as independent variables. Both have to be on the same row or column, which can be seen as follows: If one assumes that S and L are on a diagonal, then the row containing S and the column containing L share a common element C. It now follows that the addition of the row of S and column of L yields $J=L+C+(S+a)=S+C+(L-b)$, and therefore we have the contradiction that $a+b=0$ with positive a, b. This means S and L have to be on the same row or column. We can now put S and L at the first and second position of the first row. The remaining 3 variables can be arranged by requiring that $R_{22}<R_{32}$ or if $R_{22}=R_{32}$, then $R_{23}\le R_{33}$. We finally get

$$R=\boxed{\begin{array}{ccc} S & L & X+B-T \\ X & B & S+L-T \\ L+B-T & S+X-T & T \end{array}}. \tag{2.11}$$

With this arrangement and the fact that $S+X-T\ge S$ and $L\ge L+B-T$, we get the ordering

$$L\ge X\ge T\ge B\ge S, \tag{2.12}$$

which we will be exploiting in the following section for storing the Wigner 3j symbols.

2.2.2. Storing magic squares. The unique ordering of the Regge square (2.11) and its one-to-one mapping results in a corresponding 3j symbol which exhibits an ordering of the form $j_1\ge j_3\ge j_2$, and furthermore $m_2\ge 0$ and if $m_2=0$, then $m_3\ge 0$, although we will not make use of this in the following.

However, we will make use of the ordering (2.12). It allows for the creation of an ordered one dimensional array of 3j symbols which can be indexed with the help of these 5 identifying variables. In practice this means that a file can be created that contains the Wigner 3j symbols in consecutive order indexed by L,X,T,B,S.

In order to retrieve an individual 3j symbol with parameters j_1,j_2,j_3,m_1,m_2,m_3 from this array, one first has to calculate the corresponding Regge parameters L,X,T, B,S. From this an index $c(L,X,T,B,S)$ can be calculated according to

$$\begin{aligned} c &= \sum_{l=0}^{L-1}\sum_{x=0}^{l-1}\sum_{t=0}^{x-1}\sum_{b=0}^{t-1}\sum_{s=0}^{b-1}1+\sum_{x=0}^{X-1}\sum_{t=0}^{x-1}\sum_{b=0}^{t-1}\sum_{s=0}^{b-1}1+\sum_{t=0}^{T-1}\sum_{b=0}^{t-1}\sum_{s=0}^{b-1}1+\sum_{b=0}^{B-1}\sum_{s=0}^{b-1}1+\sum_{s=0}^{S-1}1+1 \\ &= \frac{1}{120}L(24+L(50+L(35+L(10+L))))+\frac{1}{24}X(6+X(11+X(6+X))) \\ &\quad +\frac{1}{6}T(2+T(3+T))+\frac{1}{2}B(B+1)+S+1. \end{aligned} \tag{2.13}$$

The total number of stored Regge squares for all indexed values up to L is

$$c_{\text{total}} = \sum_{l=0}^{L}\sum_{x=0}^{l-1}\sum_{t=0}^{x-1}\sum_{b=0}^{t-1}\sum_{s=0}^{b-1} 1 + 1$$

$$= \frac{1}{120}L(274 + L(225 + L(85 + L(15 + L)))) + 1. \tag{2.14}$$

In (2.13) and (2.14) it is understood that the sums are zero if any of the upper limits are negative, e.g., $\sum_{s=0}^{-1} 1 = 0$. Any $3j$ symbol can now be retrieved by calculating the corresponding Regge square, which in turn has to be reordered to the form (2.11) to extract the 5 numbers L, X, T, B, S. Then the index $c(L, X, T, B, S)$ can be calculated to retrieve the $3j$ symbol from memory.

Unfortunately, this storage scheme makes use of only 36 instead of 72 symmetries. The reason is that the sum over X, B, and T includes redundancies since we have not made use of the fact that the Regge square is ordered such that $R_{22} < R_{32}$ or if $R_{22} = R_{32}$, then $R_{23} \leq R_{33}$. However, extracting those redundancies and introducing additional bookkeeping will almost certainly reduce the effect of gaining a factor of 2 in memory saving and fast retrieval of the $3j$ symbol. Therefore we have not made use of it.

In Table 1 we have listed the required number of $3j$ symbols for typical values of L using this storage scheme. Also shown is the total number of nontrivial zeros which occur naturally and are not enforced by any known symmetry. They were first found and investigated by Biedenharn and Louck [9], [13].

TABLE 1

Shown is the storage required for storing Wigner $3j$ symbols. `N` *is the total number of the nontrivial zeros stored in the data file. The column labeled* `R` *shows the number of R symbols which have to be stored according to* (2.14). Σ *is the overall storage required for the* `R` *vector given in bytes, assuming that* 8 *bytes are required for a double precision number.*

L	`N`	`R`	Σ
20	328	53,130	425,040
25	635	142,506	1,140,048
30	930	324,632	2,597,056
35	1528	658,008	5,264,064
40	2092	1,221,759	9,774,072

3. Wigner $6j$ symbols and Racah coefficients. Whenever 3 angular momenta have to be coupled, the Wigner $6j$ symbols or the closely related Racah coefficients are encountered. Usually Wigner $6j$ symbols are used since they exhibit a higher symmetry than Racah coefficients. We shall refer to the vast literature for a detailed account of these coefficients and merely state the most important properties and the ways of evaluating them. Wigner $6j$ symbols can usually be defined in terms of Clebsch–Gordan coefficients, and all their properties can subsequently be derived from them. Since the relation between $6j$ and Racah coefficients can be written as

$$\begin{Bmatrix} j_1 & j_2 & j_3 \\ j_4 & j_5 & j_6 \end{Bmatrix} = (-)^{j_1+j_2+j_4+j_5} W(j_1 j_2 j_5 j_4; j_3 j_6)\,, \tag{3.1}$$

we will mainly concentrate on $6j$ symbols which exhibit higher symmetry properties than Racah coefficients. A typical formula for evaluation of Wigner $6j$ symbols is [4]

$$\begin{Bmatrix} j_1 & j_2 & j_3 \\ j_4 & j_5 & j_6 \end{Bmatrix} = \Delta(j_1,j_2,j_3)\Delta(j_3,j_4,j_5)\Delta(j_1,j_5,j_6)\Delta(j_2,j_4,j_6) \times \sum_{k=k_{\min}}^{k_{\max}} \frac{(-)^k}{(k-j_1-j_2-j_3)!(k-j_1-j_5-j_6)!(k-j_2-j_4-j_6)!(k-j_3-j_4-j_5)!} \times \frac{(k+1)!}{(j_1+j_2+j_4+j_5-k)!(j_1+j_3+j_4+j_6-k)!(j_2+j_3+j_5+j_6-k)!}, \tag{3.2}$$

where each triplet satisfies the triangle conditions and $k_{\min} = \max(j_1+j_2+j_3, j_1+j_5+j_6, j_2+j_4+j_6, j_3+j_4+j_5)$, $k_{\max} = \min(j_1+j_2+j_4+j_5, j_1+j_3+j_4+j_6, j_2+j_3+j_5+j_6)$, and Δ is defined as the triangle coefficient (2.3).

3.1. Symmetries. Again let us summarize the most important symmetries of Wigner 6j symbols which will be the basis for an efficient storage scheme.

1. Wigner 6j symbols are left invariant under any permutation of the columns

$$\begin{Bmatrix} j_1 & j_2 & j_3 \\ j_4 & j_5 & j_6 \end{Bmatrix} = \begin{Bmatrix} j_3 & j_1 & j_2 \\ j_6 & j_4 & j_5 \end{Bmatrix} = \begin{Bmatrix} j_2 & j_3 & j_1 \\ j_5 & j_6 & j_4 \end{Bmatrix} \quad \text{cyclic} \tag{3.3}$$

$$= \begin{Bmatrix} j_3 & j_2 & j_1 \\ j_6 & j_5 & j_4 \end{Bmatrix} = \begin{Bmatrix} j_1 & j_3 & j_2 \\ j_4 & j_6 & j_5 \end{Bmatrix} = \begin{Bmatrix} j_2 & j_1 & j_3 \\ j_5 & j_4 & j_6 \end{Bmatrix} \quad \text{anticyclic.} \tag{3.4}$$

2. They are invariant under the exchange of the upper and lower arguments in each of any two columns, i.e.,

$$\begin{Bmatrix} j_1 & j_2 & j_3 \\ j_4 & j_5 & j_6 \end{Bmatrix} = \begin{Bmatrix} j_1 & j_5 & j_6 \\ j_4 & j_2 & j_3 \end{Bmatrix} = \begin{Bmatrix} j_4 & j_2 & j_6 \\ j_1 & j_5 & j_3 \end{Bmatrix} = \begin{Bmatrix} j_4 & j_5 & j_3 \\ j_1 & j_2 & j_6 \end{Bmatrix}. \tag{3.5}$$

3. Symmetries based on the work by Regge [12] account for a total of 144 symmetries.
4. The Wigner 6j symbols are zero unless one can draw a tetrahedron with sides of lengths $j_1, j_2, j_3, j_4, j_5, j_6$ and such that the sum of the lengths of the sides of each triangular face sum to an integer.

The first two symmetries (1), (2) are the main symmetries of these coefficients which are usually found and described in the literature. They account for a total of 24 symmetries corresponding to the symmetries of a tetrahedron, where each j value corresponds to the edge of such a tetrahedron. An additional 6 symmetries were found such as (see [12] for more details)

$$\begin{Bmatrix} j_1 & j_2 & j_3 \\ j_4 & j_5 & j_6 \end{Bmatrix} = \begin{Bmatrix} j_1 & \frac{-j_2+j_3+j_5+j_6}{2} & \frac{j_2-j_3+j_5+j_6}{2} \\ j_4 & \frac{j_2+j_3-j_5+j_6}{2} & \frac{j_2+j_3+j_5-j_6}{2} \end{Bmatrix}, \tag{3.6}$$

which can be superimposed on the 24 tetrahedral symmetries giving a total of 144 symmetries as found by Regge [12]. These 144 symmetries can easily be demonstrated by rewriting the 6j symbol in the following 3×4 array (see [8], [9], [14]):

$$R = \begin{bmatrix} -j_3+j_4+j_5 & j_2+j_4-j_6 & j_1+j_5-j_6 & j_1+j_2-j_3 \\ -j_2+j_4+j_6 & j_3+j_4-j_5 & j_1-j_2+j_3 & j_1-j_5+j_6 \\ -j_1+j_5+j_6 & -j_1+j_2+j_3 & j_3-j_4+j_5 & j_2-j_4+j_6 \end{bmatrix}. \tag{3.7}$$

The symmetries correspond now to $3! \cdot 4! = 144$ row and column permutations.

It is furthermore possible to represent the $6j$ symbol as a 4×4 magic square; however, this mapping is no longer a one-to-one correspondence as in the case of the $3j$ symbol (see [8], [13], [14]).

3.2. Storing Wigner $6j$ coefficients. As in the case of the Wigner $3j$ symbols, we will be making use of the symmetries of the Regge symbol (3.7) for devising an efficient storage scheme for Wigner $6j$ symbols.

From the triangle condition it is clear that all entries in the R symbol (3.7) are nonnegative. Furthermore, every entry can be written as [8]

$$R_{ij} = \alpha_i - \beta_j, \tag{3.8}$$

where

$$\begin{aligned} &\alpha_1 = j_1 + j_2 + j_4 + j_5, \qquad \alpha_2 = j_1 + j_3 + j_4 + j_6, \qquad \alpha_3 = j_2 + j_3 + j_5 + j_6, \\ &\beta_1 = j_1 + j_2 + j_3, \qquad \beta_2 = j_1 + j_5 + j_6, \qquad \beta_3 = j_2 + j_4 + j_6, \qquad \beta_4 = j_3 + j_4 + j_5. \end{aligned} \tag{3.9}$$

Furthermore, as shown in [8] each α and β is a nonnegative integer and $\alpha_i \geq \beta_j$ for all i, j. Together with the symmetry property for the elements of the R symbol

$$R_{ij} + R_{kl} = R_{il} + R_{kj}, \tag{3.10}$$

one can represent the R symbol as [14]

$$R = \boxed{\begin{array}{cccc} R_{11} & R_{11}+\Delta_1 & R_{11}+\Delta_2 & R_{11}+\Delta_3 \\ R_{11}+\Delta_4 & R_{11}+\Delta_1+\Delta_4 & R_{11}+\Delta_2+\Delta_4 & R_{11}+\Delta_3+\Delta_4 \\ R_{11}+\Delta_5 & R_{11}+\Delta_1+\Delta_5 & R_{11}+\Delta_2+\Delta_5 & R_{11}+\Delta_3+\Delta_5 \end{array}}, \tag{3.11}$$

where the Δ_i are integer differences.

This representation enables us now to define a unique ordering. If we order the α's and β's such that

$$\begin{aligned} \alpha_1 \geq \alpha_2 \geq \alpha_3, \\ \beta_1 \geq \beta_2 \geq \beta_3 \geq \beta_4, \end{aligned} \tag{3.12}$$

then the first row and last column of the R symbol give us 6 numbers, i.e.,

$$R = \boxed{\begin{array}{cccc} S & B & T & X \\ & & & L \\ & & & E \end{array}}, \tag{3.13}$$

which are ordered such that

$$E \geq L \geq X \geq T \geq B \geq S. \tag{3.14}$$

With this ordering we can now apply the same method as in section 2.2.2 except that instead of looping over 5 parameters we loop over 6. Correspondingly, the index

$c(E,L,X,T,B,S)$ which is needed for the retrieval needs to be calculated according to

$$\begin{aligned} c &= \sum_{e=0}^{E-1}\sum_{l=0}^{e-1}\sum_{x=0}^{l-1}\sum_{t=0}^{x-1}\sum_{b=0}^{t-1}\sum_{s=0}^{b-1}1 + \sum_{l=0}^{L-1}\sum_{x=0}^{l-1}\sum_{t=0}^{x-1}\sum_{b=0}^{t-1}\sum_{s=0}^{b-1}1 + \sum_{x=0}^{X-1}\sum_{t=0}^{x-1}\sum_{b=0}^{t-1}\sum_{s=0}^{b-1}1 \\ &\quad + \sum_{t=0}^{T-1}\sum_{b=0}^{t-1}\sum_{s=0}^{b-1}1 + \sum_{b=0}^{B-1}\sum_{s=0}^{b-1}1 + \sum_{s=0}^{S-1}1 + 1 \\ &= \frac{1}{720}E(120+E(274+E(225+E(85+E(15+E))))) \\ &\quad + \frac{1}{120}L(24+L(50+L(35+L(10+L)))) + \frac{1}{24}X(6+X(11+X(6+X))) \\ &\quad + \frac{1}{6}T(2+T(3+T)) + \frac{1}{2}B(B+1) + S + 1. \end{aligned} \tag{3.15}$$

The total number of stored Regge symbols for $6j$ coefficients for all parameters up to the value E is

$$\begin{aligned} c_{\text{total}} &= \sum_{e=0}^{E}\sum_{l=0}^{e-1}\sum_{x=0}^{l-1}\sum_{t=0}^{x-1}\sum_{b=0}^{t-1}\sum_{s=0}^{b-1}1 + 1 \\ &= \frac{1}{720}E(1764+E(1624+E(735+E(175+E(21+E))))) + 1. \end{aligned} \tag{3.16}$$

Table 2 shows the number of $6j$'s for given value of E and the required memory. It should be noted that all 144 symmetries have been taken into account. Also shown is the total number of nontrivial zeros which occur naturally and are not enforced by any known symmetry. They were first found and investigated by Biedenharn and Louck [9], [13].

TABLE 2

Shown is the storage required for storing Wigner $6j$ symbols. `N` *is the total number of the nontrivial zeros stored in the data file. The column labeled* `R` *shows the number of R symbols which have to be stored according to* (3.16). Σ *is the overall storage required for the* `R` *vector given in bytes, assuming that* 8 *bytes are required for a double precision number.*

E	`N`	`R`	Σ
20	164	230,230	1,841,840
25	279	736,281	5,890,248
30	448	1,947,792	15,582,336
35	800	4,496,388	35,971,104

4. Gaunt coefficients. We will make use of the following definition (see, for example, [4]):

$$\begin{aligned} Y\begin{smallmatrix} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \end{smallmatrix} &= \int Y_{l_1,m_1}(\Omega)Y_{l_2,m_2}(\Omega)Y_{l_3,m_3}(\Omega)\,d\Omega \\ &= \sqrt{\frac{(2l_1+1)(2l_2+1)(2l_3+1)}{4\pi}}\begin{pmatrix} l_1 & l_2 & l_3 \\ 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \end{pmatrix}, \end{aligned} \tag{4.1}$$

where $Y_{l_1,m_1}(\Omega)$ is a spherical harmonic and $d\Omega = \sin\vartheta\,d\vartheta\,d\varphi$ is the element of the solid angle. Although Gaunt coefficients can be calculated by evaluating the two $3j$

symbols separately, a more combined expression can also be given [15] as follows.

$$
\begin{aligned}
(4.2)\quad Y^{\,l_1\ \ l_2\ \ l_3}_{m_1\ m_2\ m_3} &= \Delta(l_1,l_2,l_3)\,\delta_{\mathrm{mod}(2(l_1+l_2+l_3),2),0}\,\delta_{m_1+m_2+m_3,0}(-)^{L+l_3+m_1-m_2}\\
&\times\sqrt{\frac{(2l_1+1)(2l_2+1)(2l_3+1)}{4\pi}}\,\frac{L!}{(L-l_1)!(L-l_2)!(L-l_3)!}\\
&\times\sqrt{(l_1+m_1)!(l_1-m_1)!(l_2+m_2)!(l_2-m_2)!(l_3+m_3)!(l_3-m_3)!}\\
&\times\sum_{k=k_{\min}}^{k_{\max}}\frac{(-)^k}{k!(k+l_3-l_1-m_2)!(k+l_3-l_2+m_1)!(k+l_3-l_2+m_1)!}\\
&\times\frac{1}{(l_1-m_1-k)!(l_2+m_2-k)!},
\end{aligned}
$$

where $L=\frac{1}{2}(l_1+l_2+l_3)$, $k_{\min}=\max(-l_3+l_1+m_2,-l_3+l_2-m_1,0)$, and $k_{\max}=\min(l_2+m_2,l_1-m_1,l_1+l_2-l_3)$ such that nowhere a factorial of a negative number appears. The Kronecker delta $\delta_{\mathrm{mod}(2(l_1+l_2+l_3),2),0}$ in (4.2) ensures that only even numbers of $l_1+l_2+l_3$ give a nonzero contribution. The symbol $\Delta(l_1,l_2,l_3)$ is defined in (2.3).

4.1. Symmetries. The symmetries of the Gaunt coefficients are essentially the same as the Wigner $3j$ symbols and can most easily be derived from (4.1). However, due to the fact that they are a product of two $3j$ symbols, they have additional symmetry properties; in particular, Gaunt coefficients are

1. left invariant under any permutation of the columns

$$
(4.3)\quad Y^{\,l_1\ \ l_2\ \ l_3}_{m_1\ m_2\ m_3} = Y^{\,l_3\ \ l_1\ \ l_2}_{m_3\ m_1\ m_2} = Y^{\,l_2\ \ l_3\ \ l_1}_{m_2\ m_3\ m_1} \qquad \text{cyclic}
$$

$$
(4.4)\quad = Y^{\,l_3\ \ l_2\ \ l_1}_{m_3\ m_2\ m_1} = Y^{\,l_1\ \ l_3\ \ l_2}_{m_1\ m_3\ m_2} = Y^{\,l_2\ \ l_1\ \ l_3}_{m_2\ m_1\ m_3} \qquad \text{anticyclic;}
$$

2. invariant under space inflection, i.e.,

$$
(4.5)\quad Y^{\,j_1\ \ j_2\ \ j_3}_{m_1\ m_2\ m_3} = Y^{\,j_1\ \ \ j_2\ \ \ j_3}_{-m_1\ -m_2\ -m_3};
$$

3. symmetric with respect to the Regge symmetries as inherited for the $3j$ symbols [5];
4. zero for l_1, l_2, l_3 not fulfilling the triangle relation;
5. zero for violating any one of the conditions

$$
(4.6)\quad l_1\geq|m_1|,\qquad l_2\geq|m_2|,\qquad l_3\geq|m_3|;
$$

6. nonzero only for an even sum of the l_i, i.e.,

$$
(4.7)\quad J=l_1+l_2+l_3=2n,\quad n\in\mathbf{N}\quad\Leftrightarrow\quad l_1+l_2+l_3=0\ \mathrm{mod}\ 2.
$$

In particular, condition (4.7) comes from the first $3j$ symbol in (4.1) with $m_i=0,\ i=1,2,3$, together with the antisymmetry property (2.7).

4.2. Storing Gaunt coefficients. The fact that makes Gaunt coefficients more attractive for storing is that they exhibit more symmetry properties than $3j$ symbols. Furthermore, two $3j$ symbols plus a square root coefficient (see (4.1)) can be stored in one go. The storage scheme we propose here can also be used for Wigner $3j$ symbols with integer values. However, the scheme devised in section 2.2 is more generic and can deal with half-integer values.

The cyclic and anticyclic permutation properties (4.3) and (4.4) make it possible to store only Gaunt factors where $l_1 \geq l_2 \geq l_3$. Furthermore, by using rule (4.5) we can always arrange $m_3 \geq 0$. It is advantageous to use m_3 instead of m_1 since the latter has a wider range and can force $m_3 > l_3$, which leads to redundancies. We can now use l_1, l_2, l_3, m_3 to form a pointer pointing to an object which has only two indices left: m_1, m_2. Although this object appears to be a matrix, it can be stored as a vector depending on m_2, where the value of m_1 is fixed by symmetry (4.5). The value of m_2 is hereby in the range $-l_2 \leq m_2 \leq \min(l_1 - m_3, l_2)$ as required by relation (4.6).

In the following we will use the syntax of the programming language C which has the concept of pointers and is therefore more adapted for the problem at hand. We can now define a one dimensional array of pointers `p[]` indexed by $c(l_1, l_2, l_3, m_3)$ as given by

$$
\begin{aligned}
(4.8) \quad c &= \sum_{i=0}^{l_1} \sum_{j=0}^{i} \sum_{k=0}^{j} k + \sum_{j=0}^{l_2} \sum_{k=0}^{j} k + \sum_{k=0}^{l_3} k + m_3 + 1 \\
&= \frac{1}{24} l_1 (6 + l_1 (11 + l_1 (6 + l_1))) + \frac{1}{6} l_2 (2 + l_2 (3 + l_2)) + \frac{1}{2} l_3 (l_3 + 1) + m_3 + 1.
\end{aligned}
$$

Each entry of this pointer `p[c]` points to the beginning of a vector indexed by m_2 that stores the actual Gaunt coefficient. All these vectors can be stored in a consecutive array `y[]`.

Although by construction this array of Gaunt coefficients should not contain any redundancies in the form of consecutive zeros, this is not the case. This is due to the fact that the $3j$ symbols contain additional zeros which are not related to any symmetry properties as pointed out by [13]. However, the storage method as given above can easily cater to such redundancies. Whilst storing the Gaunt coefficients it is easy to test whether they are zero for a whole set of m_2 values. In this case one simply skips filling the storage vector `y[]` and lets the pointer `p[c]` become the `NULL` pointer. As can be seen in Table 3, about 20% of memory can be saved. We found that in practice all zeros could be eliminated this way.

Even though we have used C to store the coefficients, the above algorithm can also be implemented in Fortran, where the pointer `p[c]` becomes an `integer` array holding the index of the array holding the Gaunt coefficients, i.e., `y(p(c))`. Since in this case additional index arithmetic is required, we found that the C program is on average 2.7% faster than the equivalent Fortran 77 program (see Table 4). In Table 3 we have listed the required number of Gaunt coefficients for typical values of l_1. It should be noted that it is difficult to compare Tables 1 and 3. This is because there is no direct one-to-one correspondence between L and l_1. In fact, all symbols up to $l_1 = l_2 = l_3 = L/2$ are stored. Above this value and especially for $m_1 \approx l_1$ the L values are exceeded so that these symbols would have to be recalculated.

5. Benchmarking of the Wigner $3j$ and $6j$ symbols retrieval and calculation routines. We have used published computer codes, which calculate Wigner $3j$ and $6j$ symbols, to benchmark our storage routine against. Those computer codes

TABLE 3

Shown is the storage required for storing Gaunt coefficients. The column labeled `Y` *shows the number of Gaunt coefficients which have to be stored after all the zero elements have been removed (see text for more details).* $\texttt{Y}^0$ *includes those zero elements.* `p` *shows the number of pointer entries needed.* Σ *is the overall storage required for both* `Y` *and* `p` *vectors given in bytes, assuming that* 8 *bytes are required for a double precision number and* 4 *bytes for a pointer variable.*

l_1	$\texttt{Y}^0$	`Y`	`p`	Σ
20	121,794	102,962	10,627	866,204
25	334,336	280,828	23,726	2,341,528
30	785,402	657,154	46,377	5,442,740
35	1,611,164	1,343,720	82,216	11,078,624
40	3,043,735	2,533,071	135,752	20,807,576

TABLE 4

The time for retrieving Gaunt coefficients. `N` *is the total number of retrieved Gaunt coefficients;* `C` *and* `Fortran` *denote the timings obtained in seconds for our programs written in C and Fortran 77, respectively;* `Recursion` *denotes the timing in seconds for the recursion algorithm* [11].

`L`	`N`	`Fortran`	`C`	`Recursion`
20	17145051	1.51	1.47	56.21
25	59708376	6.22	6.07	195.62
30	167613776	18.19	17.68	548.39
35	404219376	44.19	43.00	1318.59
40	870793301	95.32	92.80	2855.81

are the highly accurate $3j$ and $6j$ routines of Schulten and Gordon [10], [11]. We found that for the range of values tested here, close to machine precision was obtained for those recursion routines. They are therefore ideal candidates for comparison since our stored $3j$ and $6j$ routines provide these symbols exactly to 16 digits in double precision. There are other methods [8], [17]; unfortunately, with most of them computer codes have not been published, and attempts to contact the authors proved unsuccessful.

In our test we have assumed that the $3j$ routines are called with unpredictable parameter values for $j_1, j_2, j_3, m_1, m_2, m_3$ by looping over all 6 parameters up to a chosen maximum value of j_1 including half-integer values. This also tests for the ability of the routines to return zeros for the cases where the symmetry properties in section 2.1 are violated. We feel that this accounts for the most typical accessing of these subroutines in real codes of physical interest [1], [2], [3]. A similar methodology has been applied to the $6j$ routines as well. For all benchmarks a standard PC with a 500 MHz Intel Celeron processor and 256 MB of RAM was used.

5.1. Wigner $3j$ symbols. In Table 5 we compare our program for calculating $3j$ symbols written in C and Fortran 77, respectively, with the recursion routines due to Schulten and Gordon [11]. We found that on average the CPU time for the C program is about 12–16% faster than the Fortran77. The reason is explained in the previous section and is due to the fact that the C program has the concept of pointers. Also, both our programs are much faster than the recursion method which is written in Fortran 77. The factor column `F` shows how many times our C program is faster than the recursion method. We achieve a maximum of 41 times faster!

5.2. Wigner $6j$ symbols. In Table 6 a comparison is made between our storage routine for retrieving Wigner $6j$ symbols written in C and the recursion routines due to Schulten and Gordon [11]. Again we see a clear increase for our program.

It appears that the recursion routine for Wigner $6j$ symbols is faster than for

TABLE 5

CPU time required by different programs to calculate the 3j symbols; `N` *is the total number of calling the 3j symbol subroutines. The columns labeled* `Fort3j` *and* `C3j` *show the timings in seconds for our 3j subroutines written in Fortran 77 and C, respectively; the column labeled* `Recur3j` *shows the timings for the recursion method* [11]. *The column labeled* `F` *denotes the comparison speed in retrieving Wigner 3j symbols between our C routine and the recursion method.*

J	N	Fort3j	C3j	Recur3j	F
20	4084101	10.39	8.71	354.12	40.66
25	11881376	31.72	26.86	1052.59	39.19
30	28629151	80.41	68.79	2612.15	37.97
35	60466176	177.81	154.34	5676.78	36.78
40	115856201	356.53	313.09	11035.64	35.25

TABLE 6

CPU time required by different programs to calculate the Wigner 6j symbols. `N` *is the total number of calling the 6j symbol subroutines. The column labeled* `C6j` *shows the timing in seconds for our 6j subroutines. The column labeled* `Recur6j` *shows the timing in seconds for the recursion method* [11]. *The column labeled* `F` *denotes the comparison speed in retrieving Wigner 6j symbols between our C routine and the recursion method.*

J	N	C6j	Recur6j	F
20	4641791	4.22	14.48	3.43
25	16199001	13.83	53.80	3.89
30	45549416	39.02	160.8	4.12
35	109992786	92.30	414.18	4.49

Wigner 3j symbols. This is largely due to the fact that the triangle conditions for Wigner 6j symbols are more restrictive, and therefore the recursion sequences to be calculated tend to be shorter.

6. Conclusion. We have shown that with the help of the large numbers of symmetries that Wigner 3j, 6j and Gaunt coefficients exhibit, very efficient storage routines can be devised. Retrieving those Wigner 3j and 6j symbols outperforms the calculation of those symbols by more than an order of magnitude. Furthermore, these Wigner 3j and 6j symbols can be precalculated by using MuPAD [16], which is a symbolic algebra package, so that no loss of machine precision occurs in the retrieval of these numbers. In addition we have shown that computer codes written in C outperform the same codes in Fortran by about 16% due to the use of pointer arithmetic. All computer codes can be easily modified to provide quadruple precision numbers. We intend to publish the computer codes elsewhere.

REFERENCES

[1] S. KELLER, C. T. WHELAN, H. AST, H. R. J. WALTERS, AND R. M. DREIZLER, *Relativistic distorted-wave Born calculations for (e,2e) processes on inner shells of heavy atoms*, Phys. Rev. A, 50 (1994), p. 3865–3877.

[2] J. RASCH, M. ZITNIK, L. AVALDI, C. T. WHELAN, G. STEFANI, R. J. ALLEN, AND H. R. J. WALTERS, *Theoretical and experimental investigation of the triple differential cross sections for electron impact ionization of Kr(4p) and Xe(5p) at 1keV impact energy*, Phys. Rev. A, 56 (1997), pp. 4644–4655.

[3] P. MARCHALANT, J. RASCH, C. T. WHELAN, D. H. MADISONN, AND H. R. J. WALTERS, *First and second Born calculation of (e, 2e) excitation-ionization of helium*, J. Phys. B, 32 (1999), pp. L705–L710.

[4] A. R. EDMONDS, *Angular Momentum in Quantum Mechanics*, Princeton University Press, Princeton, NJ, 1974.

[5] T. Regge, *Symmetry properties of Clebsch-Gordan's coefficients*, Nuovo Cimento, 10 (1958), pp. 544–545.
[6] K. Venkatesh, *Symmetries of the* 3*j coefficient*, J. Math. Phys., 19 (1978), pp. 2060–2063.
[7] K. Venkatesh, *Symmetries of the* 6*j coefficient*, J. Math. Phys., 19 (1978), pp. 1973–1974.
[8] R. E. Tuzun, P. Burkhardt, and D. Secrest, *Accurate computation of individual and tables of* 3*j and* 6*j symbols*, Comput. Phys. Comm., 112 (1998), pp. 112–148.
[9] L. C. Biedenharn and J. C. Louck, *Angular Momentum in Quantum Physics*, Vol. 8, Addison–Wesley, Reading, MA, 1981.
[10] K. Schulten and R. G. Gordon, *Semiclassical approximations to* 3*j- and* 6*j-coefficients for quantum-mechanical coupling of angular momenta*, J. Math. Phys., 16 (1975), pp. 1971–1975.
[11] K. Schulten and R. G. Gordon, *Recursive evaluation of* 3*j and* 6*j coefficients*, Comput. Phys. Comm., 11 (1976), pp. 269–278.
[12] T. Regge, *Symmetry properties of Racah coefficients*, Nuovo Cimento, 11 (1959), pp. 116–117.
[13] L. C. Biedenharn and J. C. Louck, *The Racah-Wigner Algebra in Quantum Theory*, Vol. 9, Addison–Wesley, Reading, MA, 1981.
[14] L. A. Shelepin, *On the symmetry of the Clebsch-Gordon coefficients*, Soviet Physics JETP, 19 (1964), pp. 702–705.
[15] A. Liberato de Brito, *FORTRAN program for the integral of three spherical harmonics*, Comput. Phys. Comm., 25 (1982), pp. 81–85.
[16] B. Fuchssteiner, *MuPAD*, University of Paderborn, Paderborn, Germany, http://www.mupad.de/.
[17] D. F. Fang and J. F. Shriner Jr., *A computer program for the calculation of angular-momentum coupling coefficients*, Comput. Phys. Comm., 70 (1992), pp. 147–153.
[18] S. T. Lai and Y. N. Chiu, *Exact computation of the* 3*j and* 6*j symbols*, Comput. Phys. Comm., 61 (1990), pp. 350–360.