Whole Number Recursion Formulae for High Order Clebsch-Gordan Coupling Coefficients

David Ritchie

Whole Number Recursion Formulae for High Order Clebsch-Gordan Coupling Coefficients

David W. Ritchie

Inria Nancy – Grand Est, 54600 Villers-lès-Nancy, France

Abstract

Clebsch-Gordan (CG) coupling coefficients appear in many wave-based theories of matter and radiation. Here, novel whole number recursion formulae are presented for the exact calculation of arbitrarily high order CG coupling coefficients. Using an extended precision integer and rational number arithmetic library, these recursion formulae are shown to be up to four orders of magnitude faster than an exact prime factorisation approach. When coded in the C programming language using floating point "long double" or "double" precision, good numerical precision is obtained for the common case where the total magnetic quantum number, m_3, is zero for all CG coefficients up to order $J = 200$, or $J = 130$, respectively, where the ranges of the principal quantum numbers are restricted to $0 \leq j_1 \leq J$, $0 \leq j_2 \leq j_1$, and $(j_1 - j_2) \leq j_3 \leq (j_1 + j_2)$. In this case, the calculation is up to four orders of magnitude faster than the exact prime number factorisation method used here. A C program that demonstrates these calculations is available for download at *http://cgc.loria.fr*.

Keywords: Clebsch-Gordan Coefficient, $3j$ Symbol, Gaunt Coefficient, Rational Number Recursion, Extended Precision Arithmetic

1. Introduction

In quantum mechanics, Clebsch-Gordan (CG) coefficients describe how individual angular momentum states may be coupled to give a total angular momentum state of a system [1, 2]. In the literature, CG coefficients are sometimes also known as Wigner coefficients or vector coupling coefficients. CG coefficients are closely related to Wigner's $3j$ symbol [3] and Gaunt's coefficient [4, 5]. The need to calculate high order CG coefficients often

arises in applications where one wishes to calculate translations of spherical harmonic expansions [6] e.g. in particle scattering simulations [7, 8], fast multipole methods [9], and polar Fourier correlation techniques [10].

Explicit recursion formulae and closed-form factorial sum expressions for the CG coefficients have long been known [1, 2]. Schulten and Gordon developed recursion formulae suitable for the computer calculation of high order $3j$ symbols, thus allowing sequences of coefficients to be calculated where one of the principal quantum numbers changes and the others remain fixed [11]. However, in this approach intermediate results need to be re-normalised to prevent the recurrence from diverging. Such problems were largely resolved by Luscombe and Luban who introduced a non-linear recurrence technique, although it is necessary in their approach to protect against possible division by zero problems [12].

In order to avoid the risk of numerical instabilities, several authors calculate CG coefficients using explicit factorial sum formulae, in which the factorials are expressed as products of prime numbers [13, 14, 15, 16]. This allows many integer factors to be manipulated and ultimately cancelled symbolically. When coupled with a modern extended precision arithmetic library this also allows CG coefficients or $3j$ symbols with very large quantum numbers to be calculated accurately. However, this approach is computationally expensive, as only one CG coefficient or $3j$ symbol can be calculated at a time. Other authors have aimed to achieve a compromise between speed and accuracy by re-writing some of the factorial terms as binomial coefficients [14, 17, 18], or as triangle coefficients in Regge squares [19, 20, 21] in order to allow common factors to be re-used.

Here, novel yet simple two-term and three-term recursion formulae are presented for calculating high order CG coefficients rapidly and accurately. More specifically, a CG coefficient is expressed as a sum of products of three binomial coefficients, here called an un-normalised CG coefficient, and the remaining factorial product terms are treated as a normalisation factor. This separation of terms allows un-normalised CG coefficient coefficients to be calculated exactly in integer arithmetic using simple three-term recursion formulae that correspond to un-normalised forms of the classical ladder recursion formulae. While the calculation of high order coefficients must be done using an extended precision library, the recursion is guaranteed to be stable for arbitrarily large quantum numbers. This approach also allows the squares of the normalisation factors to be calculated exactly using two-term rational number recursion formulae. Thus the overall calculation involves

only one square root function call per CG coeffient.

Compared to previous work, the approach described here follows a similar intuition to the recursive binomial sum method of Tuzun *et al.* [20] for the calculation of $3j$ and $6j$ symbols. In their method, the recursions proceed outwards from low magnetic quantum number values, and hence their approach requires several different formulae to initialise the recursion and to move in different directions within the space of allowed magnetic quantum numbers. The main novelty of the present contribution is that the first recursion proceeds upwards from unity on the principal quantum numbers having maximal magnetic quantum numbers (i.e. so-called "top coefficients", as described below), and this is then followed by one or more "inward" ladder recursions on the magnetic quantum numbers. This approach is particularly well suited for calculating the common case in which $m_3 = 0$ as it allows all CG coefficients of interest to be reached using just two main recursion paths. In practical applications, the present approach is well suited to calculating lists of CG coefficients "on-the-fly" for different magnetic quantum numbers having the same principal quantum numbers, thus avoiding the need for large or complex in-memory storage schemes.

2. Methods

Racah's Form of the Clebsch-Gordan Coupling Coefficients

The CG coupling coefficients describe how the angular momentum state or wave-function of two coupled particles, $\psi_{j,m}(1,2)$, may be expressed as a sum of products of uncoupled states or wave-functions $\psi_{j_1,m_1}(1)$ and $\psi_{j_2,m_2}(2)$ according to

$$\psi_{j,m}(1,2) = \sum_{m_1,m_2} C \begin{pmatrix} j_1 & j_2 & j \\ m_1 & m_2 & m \end{pmatrix} \psi_{j_1,m_1}(1)\psi_{j_2,m_2}(2). \tag{1}$$

Racah's symmetric form for the CG coefficients is [22]

$$C \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \delta_{m_1+m_2,m_3}(2j_3+1)^{1/2} \left[\frac{(j_1+j_2-j_3)!(j_1-j_2+j_3)!(-j_1+j_2+j_3)!}{(j_1+j_2+j_3+1)!} \right]^{1/2} \times$$
$$\sum_k \frac{(-1)^k \left[(j_1+m_1)!(j_1-m_1)!(j_2+m_2)!(j_2-m_2)!(j_3+m_3)!(j_3-m_3)! \right]^{1/2}}{k!(j_1+j_2-j_3-k)!(j_1-m_1-k)!(j_2+m_2-k)!(j_3-j_2+m_1+k)!(j_3-j_1-m_2+k)!}, \tag{2}$$

where the summation on k ranges over all values for which the factorials are well-defined. The allowed values of the principal quantum numbers, j_n, must satisfy the triangle rule, $|j_1 - j_2| \leq j_3 \leq (j_1 + j_2)$, and the "magnetic" quantum numbers, m_i, must satisfy the conservation rule, $m_3 = m_1 + m_2$. Thus, the CG coefficients are essentially five-dimensional quantities, since one of the magnetic quantum numbers may always be deduced from the other two. With a suitable choice of phase convention, the CG coefficients may be treated as real numbers with magnitude less than or equal to unity.

CG coefficients are related to Wigner's $3j$ symbol according to

$$\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \frac{(-1)^{j_1 - j_2 - m_3}}{\sqrt{2j_3 + 1}} \, C \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix}. \tag{3}$$

The $3j$ symbol is often preferred in many applications because it gives equal status to the three principal quantum numbers, and it exhibits in a natural way their permutational symmetries. However, for the purposes of calculating high order $3j$ symbols, it is convenient to work with the CG coefficients, as this allows the phase factor that stands in front of the $3j$ symbol to be dropped. This choice is also convenient because it allows simple restrictions to be placed on the ranges of the principal quantum numbers to be calculated, knowing that any coefficients not calculated explicitly can be obtained using their column-wise permutational symmetries.

While CG coefficients having small quantum numbers can be calculated relatively easily in ordinary floating point arithmetic using Racah's form, Equation 2 is expensive to evaluate when any of the principal quantum numbers are large, and especially when the magnitudes of m_1 and m_2 are also small. Furthermore, accurate calculations with high quantum numbers become challenging due to the numerical instabilities that arise when manipulating high order factorials. Indeed, early implementations fail due to numerical overflow or underflow with principal quantum numbers greater than about $J = 6$ [23] or $J = 24$ [13].

The CG coefficients may also be calculated using well-known recursion formulae. However, calculating long recursion chains for high order coefficients can result in severe numerical instabilities. Nonetheless, since Racah was able to obtain this explicit form by studying the basic recursion relations that define the CG coefficients, it remains highly relevant to seek simple and stable recursion relations for their efficient and accurate calculation.

2.1. Whole Number Recursion Formulae for Un-normalised CG Coefficients

It is well known that binomial coefficients with whole-number arguments evaluate to whole numbers. Thus it follows that products and sums of binomial coefficients must also be whole numbers. Therefore, it is convenient to express Racah's explicit CG coefficient formula as a sum of products of binomial coefficients [14, 20]

$$C\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = N\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix}^{1/2} \times G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix}, \quad (4)$$

where

$$G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \delta_{m_1+m_2,m_3} \sum_k (-1)^k \binom{j_1+j_2-j_3}{k} \binom{j_1-j_2+j_3}{j_1-m_1-k} \binom{-j_1+j_2+j_3}{j_2+m_2-k} \quad (5)$$

and

$$N\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = (2j_3+1)\left[\frac{(j_1+m_1)!(j_1-m_1)!(j_2+m_2)!(j_2-m_2)!(j_3+m_3)!(j_3-m_3)!}{(j_1+j_2-j_3)!(j_1-j_2+j_3)!(-j_1+j_2+j_3)!(j_1+j_2+j_3+1)!}\right] \quad (6)$$

is the corresponding squared normalisation factor. It is readily confirmed by expanding the binomial coefficients in Equation 5 that Equation 4 corresponds to Racah's form (Equation 2). As it is well known that several three-term recursion formulae exist for the CG coefficients, and because the normalisation factor (Equation 6) provides only a scale factor, it follows that these recursion formulae must stem from the alternating sum of products in Equation 5. Thus, G may be considered as an un-normalised CG coefficient.

It is well known that the matrix elements of the angular momentum ladder operators define the form of the CG coefficients up to an arbitrary phase factor. For example, using the result ([24], Eq.s 3.5.36 – 3.5.38)

$$\hat{J}_{\pm}\psi_{j,m}(\cdot) = \left[(j(j+1) - m(m\pm 1)\right]^{1/2}\psi_{j,m\pm 1}(\cdot) \quad (7)$$

together with the orthogonality of the CG coefficients leads to the CG "lad-

der" recursion relations ([24], Eq. 3.7.49)

$$\begin{aligned}
&\left[(j_3(j_3+1)-m_3(m_3\pm 1)\right]^{1/2} C\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3\pm 1\end{pmatrix} = \\
&\left[(j_1(j_1+1)-m_1(m_1\mp 1)\right]^{1/2} C\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1\mp 1 & m_2 & m_3\end{pmatrix} + \\
&\left[(j_2(j_2+1)-m_2(m_2\mp 1)\right]^{1/2} C\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2\mp 1 & m_3\end{pmatrix}.
\end{aligned} \tag{8}$$

By substituting un-normalised CG coefficients (Equation 4) into Equation 8, and by pulling out the leading factors of the factorials that have changed, each normalisation factor may be brought into an identical form and hence cancelled to give

$$\begin{aligned}
&\left[(j_3\mp m_3)(j_3\pm m_3+1)\right]^{1/2} G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3\pm 1\end{pmatrix}\left[\frac{(j_3\pm m_3+1)}{(j_3\mp m_3)}\right]^{1/2} = \\
&\left[(j_1\pm m_1)(j_1\mp m_1+1)\right]^{1/2} G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1\mp 1 & m_2 & m_3\end{pmatrix}\left[\frac{(j_1\mp m_1+1)}{(j_1\pm m_1)}\right]^{1/2} + \\
&\left[(j_2\pm m_2)(j_2\mp m_2+1)\right]^{1/2} G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2\mp 1 & m_3\end{pmatrix}\left[\frac{(j_2\mp m_2+1)}{(j_2\pm m_2)}\right]^{1/2},
\end{aligned} \tag{9}$$

and hence

$$\begin{aligned}
(j_3\pm m_3+1)\, G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3\pm 1\end{pmatrix} &= (j_1\mp m_1+1)\, G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1\mp 1 & m_2 & m_3\end{pmatrix} \\
&\quad + (j_2\mp m_2+1)\, G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2\mp 1 & m_3\end{pmatrix}.
\end{aligned} \tag{10}$$

Since all quantities in Equation 10 are obviously whole numbers, it follows that these recursions can be calculated using exact integer arithmetic. Here, it is convenient to combine the two formulae in Equation 10 to give a single

ladder recursion formula for a fixed value of m_3:

$$\begin{aligned}
&[j_3(j_3+1) - j_1(j_1+1) - j_2(j_2+1) - 2m_1m_2]\, G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \\
&(j_1 - m_1 + 1)(j_2 + m_2 + 1)\, G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 - 1 & m_2 + 1 & m_3 \end{pmatrix} + \\
&(j_1 + m_1 + 1)(j_2 - m_2 + 1)\, G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 + 1 & m_2 - 1 & m_3 \end{pmatrix}.
\end{aligned} \tag{11}$$

Equation 11 here corresponds to Equation (53) of Tuzun *et al.* [20]. The normalised form of this recursion is given by Rose (1957, page 44):

$$\begin{aligned}
&[j_3(j_3+1) - j_1(j_1+1) - j_2(j_2+1) - 2m_1m_2]\, C\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \\
&[(j_1 - m_1 + 1)(j_1 + m_1)(j_2 + m_2 + 1)(j_2 - m_2)]^{1/2}\, C\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 - 1 & m_2 + 1 & m_3 \end{pmatrix} + \\
&[(j_1 + m_1 + 1)(j_1 - m_1)(j_2 - m_2 + 1)(j_2 + m_2)]^{1/2}\, C\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 + 1 & m_2 - 1 & m_3 \end{pmatrix}.
\end{aligned} \tag{12}$$

It is also possible to obtain recursion formulae for the principal quantum numbers. For example, Rose (1957, page 224) gives the following recursion formula for j_3:

$$\begin{aligned}
&\left[m_1 - m_3\left(\frac{j_1(j_1+1) - j_2(j_2+1) + j_3(j_3+1)}{2j_3(j_3+1)}\right)\right] C\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \\
&\left[\frac{(j_3^2 - m_3^2)(-j_1 + j_2 + j_3)(j_1 - j_2 + j_3)(j_1 + j_2 - j_3 + 1)(j_1 + j_2 + j_3 + 1)}{4j_3^2(2j_3 - 1)(2j_3 + 1)}\right]^{1/2} \times \\
&C\begin{pmatrix} j_1 & j_2 & j_3 - 1 \\ m_1 & m_2 & m_3 \end{pmatrix} + \\
&\left[\frac{((j_3+1)^2 - m_3^2)(-j_1 + j_2 + j_3 + 1)(j_1 - j_2 + j_3 + 1)(j_1 + j_2 - j_3)(j_1 + j_2 + j_3 + 2)}{4(j_3+1)^2(2j_3 + 1)(2j_3 + 3)}\right]^{1/2} \times \\
&C\begin{pmatrix} j_1 & j_2 & j_3 + 1 \\ m_1 & m_2 & m_3 \end{pmatrix}.
\end{aligned} \tag{13}$$

From this, it is straight-forward but rather tedious to derive the following whole number recursion formula for the un-normalised CG coefficients

$$(2j_3+1)\big[(m_1-m_2)j_3(j_3+1)-m_3\big(j_1(j_1+1)-j_2(j_2+1)\big)\big]\,G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} =$$
$$(j_3+1)\big[(-j_1+j_2+j_3)(j_1-j_2+j_3)(j_1+j_2+j_3+1)\big]\,G\begin{pmatrix} j_1 & j_2 & j_3-1 \\ m_1 & m_2 & m_3 \end{pmatrix} +$$
$$j_3\big[\big((j_3+m_3)(j_3-m_3)+(2j_3+1)\big)(j_1+j_2-j_3)\big]\,G\begin{pmatrix} j_1 & j_2 & j_3+1 \\ m_1 & m_2 & m_3 \end{pmatrix}. \tag{14}$$

Again, this is much simpler than the normalised form (Equation 13).

2.2. Rational Recursion Formulae for Top CG Coefficients in (j_1, j_2, j_3)

In order to calculate a list of CG coefficients for all principal quantum numbers, it is convenient to assume, without loss of generality, that $j_1 \geq j_2$ and that $j_1 + j_2 \geq j_1 - j_2 \geq j_3$. If this is not the case, then it can always be achieved by applying up to two column permutations. From Equation 5 it can be seen that when $j_1 - m_1 = 0$ or $j_2 + m_2 = 0$, the only allowed value of k is $k = 0$, and therefore the summation must evaluate to a positive whole number. I call a CG coefficient having $k = 0$ a "top" coefficient, and denote it as such with a subscript of zero. Here, it is convenient to begin with $m_2 = -j_2$, and $m_3 = 0$, which entails setting $m_1 = j_2$ so that the condition $m_1 + m_2 = m_3$ is always satisfied. Furthermore, because when $k = 0$, a top CG coefficient must be positive, it is also convenient to work with their squares,

$$C_0\begin{pmatrix} j_1 & j_2 & j_3 \\ j_2 & -j_2 & 0 \end{pmatrix}^2 = N_0\begin{pmatrix} j_1 & j_2 & j_3 \\ j_2 & -j_2 & 0 \end{pmatrix} \times\, G_0\begin{pmatrix} j_1 & j_2 & j_3 \\ j_2 & -j_2 & 0 \end{pmatrix}^2. \tag{15}$$

This allows the calculation of square roots to be deferred. Furthermore, by substituting $m_2 = -j_2$ and $m_1 = j_2$ into Equation 5, it can be seen that the recursion may be initialised using the value of a single binomial coefficient,

$$\begin{aligned} G_0\begin{pmatrix} j_1 & j_2 & j_3 \\ j_2 & -j_2 & 0 \end{pmatrix} &= \binom{j_1-j_2+j_3}{j_1-j_2} \\ &= \frac{(j_1-j_2+j_3)!}{(j_1-j_2)!(j_3)!}. \end{aligned} \tag{16}$$

Although calculating the corresponding normalisation factor using Equation 6 appears to be rather more daunting at first sight, as shown below an entire sequence of squared coefficients, C_0^2, may be obtained rather efficiently. The corresponding normalisation factors may then be extracted using Equations 15 and 16. The C_0^2 coefficients are calculated from Equation 2 with the substitutions $m_1 = j_2$, $m_2 = -j_2$, and $m_3 = 0$ to give

$$C_0 \begin{pmatrix} j_1 & j_2 & j_3 \\ j_2 & -j_2 & 0 \end{pmatrix}^2 = \frac{(2j_3+1)(j_1-j_2+j_3)!(2j_2)!}{(j_1+j_2-j_3)!(j_1+j_2+j_3+1)!(j_3-j_1+j_2)!} \frac{(j_1+j_2)!}{(j_1-j_2)!}. \tag{17}$$

Equation 17 may be used to develop simple two-term rational number recursion formulae in j_1, j_2, and j_3 for all of the top coefficients in m_2 when $m_3 = 0$. Pulling factors in the usual way gives

$$C_0 \begin{pmatrix} j_1 & j_2 & j_3+1 \\ j_2 & -j_2 & 0 \end{pmatrix}^2 = \frac{(2j_3+3)}{(2j_3+1)} \frac{(j_1-j_2+j_3+1)(j_1+j_2-j_3)}{(j_1+j_2+j_3+2)(j_3-j_1+j_2+1)} C_0 \begin{pmatrix} j_1 & j_2 & j_3 \\ j_2 & -j_2 & 0 \end{pmatrix}^2. \tag{18}$$

In a similar manner, one obtains

$$C_0 \begin{pmatrix} j_1 & j_2+1 & j_3 \\ j_2+1 & -j_2-1 & 0 \end{pmatrix}^2 = \frac{(2j_2+1)(2j_2+2)(j_1+j_2+1)(j_1-j_2)}{(j_1-j_2+j_3)(j_1+j_2-j_3+1)(j_1+j_2+j_3+2)(j_3-j_1+j_2+1)} C_0 \begin{pmatrix} j_1 & j_2 & j_3 \\ j_2 & -j_2 & 0 \end{pmatrix}^2. \tag{19}$$

Furthermore, assuming that $j_1 \geq j_2$, then the range of j_3 is restricted to $(j_1 - j_2) \leq j_3 \leq (j_1 + j_2)$, and so whenever j_2 is incremented the lower and upper ranges of j_3 must be decremented and incremented, respectively. Therefore, another useful relation for a practical implementation is

$$C_0 \begin{pmatrix} j_1 & j_2 & j_3 \\ j_2 & -j_2 & 0 \end{pmatrix}^2 = \frac{(2j_2)(2j_2-1)(j_1+j_2)(j_1-j_2+1)}{(j_1-j_2+j_3+1)(j_1-j_2+j_3+2)(j_1+j_2-j_3)(j_1+j_2-j_3-1)} \times \frac{(2j_3+1)}{(2j_3+3)} C_0 \begin{pmatrix} j_1 & j_2-1 & j_3+1 \\ j_2-1 & -j_2+1 & 0 \end{pmatrix}^2. \tag{20}$$

Finally, by noting that when $j_2 = 0$ we must have $j_3 = j_1$. It is then easily found that

$$C_0 \begin{pmatrix} j_1 & 0 & j_1 \\ 0 & 0 & 0 \end{pmatrix}^2 = 1. \tag{21}$$

From this starting point, Equations 18, 19, and 20 may be used to derive the following two-term upward recursion formulae for top coeficients having $m_3 = 0$:

$$C_0 \begin{pmatrix} j_1 & j_2 & j_1 - j_2 \\ j_2 & -j_2 & 0 \end{pmatrix}^2 = \frac{j_1 + j_2}{4(j_1 - j_2) + 6} C_0 \begin{pmatrix} j_1 & j_2 - 1 & j_1 - j_2 + 1 \\ j_2 - 1 & -j_2 + 1 & 0 \end{pmatrix}^2 \tag{22}$$

and

$$C_0 \begin{pmatrix} j_1 & j_2 & j_3 \\ j_2 & -j_2 & 0 \end{pmatrix}^2 = \frac{(2j_3 + 1)(j_1 - j_2 + j_3)(j_1 + j_2 - j_3 + 1)}{(2j_3 - 1)(j_2 - j_1 + j_3)(j_1 + j_2 + j_3 + 1)} C_0 \begin{pmatrix} j_1 & j_2 & j_3 - 1 \\ j_2 & -j_2 & 0 \end{pmatrix}^2. \tag{23}$$

Since Equations 22 and 23 each involve multiplications by a numerator and denominator of similar magnitude, it may be expected that these two-term recursions should be rather stable numerically.

2.3. Calculating Runs of CG Coefficients for Fixed m_3 by Ladder Recursion

Substituting $m_1 \to (m_1 + 1)$ and $m_2 \to (m_2 - 1)$ into Equation 11 gives a form suitable for whole number downward recursion on m_1 and upward recursion on m_2 while holding m_3 fixed:

$$\begin{aligned} &(j_1 - m_1)(j_2 + m_2)\, G \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \\ &\left[j_3(j_3 + 1) - j_2(j_2 + 1) - j_1(j_1 + 1) - 2(m_1 + 1)(m_2 - 1) \right] G \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 + 1 & m_2 - 1 & m_3 \end{pmatrix} \\ &- (j_1 + m_1 + 2)(j_2 - m_2 + 2)\, G \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 + 2 & m_2 - 2 & m_3 \end{pmatrix}. \end{aligned} \tag{24}$$

The corresponding two-term rational number recursion relation for the normalisation factor is found from Equation 6 to be

$$N\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \frac{(j_1 - m_1)(j_2 + m_2)}{(j_1 + m_1 + 1)(j_2 - m_2 + 1)} N\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 + 1 & m_2 - 1 & m_3 \end{pmatrix}. \tag{25}$$

While ladder recursion using Equation 24 may be used with any value of m_3, it is most efficient (and more stable when using floating point arithmetic) when $m_3 = 0$ because in this case the length of the recursion may be halved by using the symmetry relation [2] (p38)

$$G\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & 0 \end{pmatrix} = (-1)^{j_1+j_2-j_3} G\begin{pmatrix} j_1 & j_2 & j_3 \\ \overline{m}_1 & \overline{m}_2 & 0 \end{pmatrix}. \tag{26}$$

After a first "run" of CG coefficients for $m_3 = 0$ has been calculated using Equation 24, further ladder runs in (m_1, m_2) may be calculated for $m_3 \pm 1$ from the coefficients of the preceding run using Equation 10. When starting a ladder run for $m_3 \neq 0$, the normalisation factor at the top of the new run may be calculated from that of the previously calculated adjacent run. Letting $T(m_1, m_2, m_3)$ denote the largest allowed m_1 and smallest allowed m_2 for the desired value of $m_3 = m_1 + m_2$, then if $T(m_1, m_2, m_3) = T(m_1 \pm 1, m_2, m_3 \mp 1)$ set

$$N\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \frac{(j_1 \pm m_1)(j_3 \pm m_3)}{(j_1 \mp m_1 + 1)(j_3 \mp m_3 + 1)} N\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 \mp 1 & m_2 & m_3 \mp 1 \end{pmatrix}. \tag{27}$$

Otherwise, it must be the case that $T(m_1, m_2, m_3) = T(m_1, m_2 \pm 1, m_3 \mp 1)$, so set

$$N\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} = \frac{(j_2 \pm m_2)(j_3 \pm m_3)}{(j_2 \mp m_2 + 1)(j_3 \mp m_3 + 1)} N\begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 \mp 1 & m_3 \mp 1 \end{pmatrix}. \tag{28}$$

The normalisation factors for the rest of the ladder may then be calculated using Equation 25.

2.4. Implementation in C and the GNU MP Multiple Precision Library

Equations 22 and 23 were used to program the calculation of the squares of all normalised top coefficients, C_0^2, in the C programming language. More specifically, functions were written to perform the calculation in C "double precision" (64-bit) and "long double precision" (80-bit) arithmetic, as well as exact rational arithmetic in which the numerators and denominators are accumulated separately using the "MPQ" family of arbitrary length rational arithmetic functions from the Gnu Multiple Precision (GMP) software library [25]. In all cases, top coefficients are calculated using nested iterations over $0 \leq j_1 \leq J$, $0 \leq j_2 \leq j_1$, and $(j_1 - j_2) \leq j_3 \leq (j_1 + j_2)$, where $m_3 = 0$, and where J is a given upper limit on the iteration on j_1. In a similar manner, the exact calculation of the un-normalised CG coefficients, G_0 (Equation 16), was implemented using the "MPZ" arbitrary length integer arithmetic functions from the GMP library. The exact calculations of C_0^2 and G_0 allow exact rational values of N_0 (Equation 15) to be obtained for arbitrarily high order top CG coefficients. The values of G_0 and N_0 may then be used to seed the ladder recursions on the magnetic quantum numbers (Equations 24, 25, 27, and 28). These recursions have been implemented in C double and long double percision arithmetic, and also using the GMP MPQ (exact rational) and MPZ (exact integer) arithmetic library functions. A test program consisting of C code for all of the above calculations is available for download at *http://cgc.loria.fr*.

2.5. Relative Error Calculations

In order to compare the accuracy of CG coefficients calculated in floating point arithmetic, C_F, with those calculated by exact whole number recursion, C_E, the relative error, E_R, was calculated using

$$E_R = |C_F - C_E|/C_E. \tag{29}$$

It is well known that some CG coefficients may be "accidentally" zero [26]. In such cases, the error is taken to be $E = |C_F|$.

It is sometimes useful to consider the deviation from orthonormality of the CG coefficients as another estimate of numerical error. For fixed j_1, j_2, j_3, and m_3, the orthogonality condition is given by

$$S^2 = \sum_{m_1,m_2} C \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix}^2 = 1, \tag{30}$$

where the sum is taken over one of m_1 or m_2, since the other is fixed by the condition $m_1 + m_2 = m_3$. The relative normalisation error, E_N, is then taken to be $E_N = |S - 1|$.

3. Results and Discussion

Table 2 shows the execution times in seconds for calculating the top coefficients up to various maximum orders $j_1 = J$. Note that calculating coefficients up to $j_1 = J$ means that the maximum value of j_3 is $2J$. The columns labeled "MPQ/s", "D80/s", and "D64/s" show the execution times in seconds for exact rational, long double, and double precision calculations, respectively. This table shows that the calculation is very fast and that only a modest amount of memory is required to store a list of all top coefficients to high order. The remaining CG Coefficients were then calculated using the above approach for the commonly encountered case in which m_3 is restricted to zero, and also for the general case in which coefficients are calculated for all allowed magnetic quantum numbers. Table 2 shows execution times and maximum errors for the first case for all coefficients having for $j_1 \leq J = 200$ and $m_3 = 0$. In order to illustrate the speed-up provided by the new recursion formulae, the "SYM/s" column shows execution times for CG coefficients calculated by the exact symbolic prime factor summation method of calculating Equation 2 [15]. This table shows that the MPQ recursion is almost 10^3 times faster than the symbolic summation method for $J = 100$, and around 3.8×10^3 faster for $J = 200$. Furthermore, the long double calculation is at least an additional order of magnitude faster than the exact MPQ calculation, while still giving only relatively small maximum errors of $O(10^{-11})$ and $O(10^{-17})$ or better for E_R and E_N, respectively. Working in double precision is seen to be slightly faster than long double, and still gives reasonable maximum error values of $O(10^{-9})$ and $O(10^{-14})$, respectively. However, the double precision calculation fails beyond $J = 130$ due to numerical overflows in the normalisation factor exponents.

Overall, Table 2 shows that in the common case where the total magnetic quantum number (m_3) is zero, the recursions presented here are numerically very stable when implemented in ordinary double or long double precision floating point arithmetic, and hence may be calculated in hardware. In this case, for J=200, CG coefficients are calculated at rates of 1.0×10^3/s by the symbolic prime factor method (SYM), 4.0×10^6/s by rational number recursion (MPQ), and 6.1×10^7/s by recursion in long double precision (D80).

In other words, the rational number recursion and the long double calculations are respectively at least three and four orders of magnitude faster than symbolic factorisation.

Table 3 shows the corresponding calculation times and relative errors for calculating CG coefficients for *all* allowed values of the magnetic quantum numbers. This table shows similar coefficient calculation rates to those in Table 2 (e.g. for J=30, SYM: 1.2×10^4/s; MPQ: 1.2×10^6/s; D80: 3.6×10^7/s). However, in this case, the total number of CG coefficients grows much more rapidly than when m_3 is restricted to zero, and it becomes impractical to calculate exact execution times for the prime factorisation method (estimated at several CPU-days for e.g. J=100). More importantly, since many more ladder recursions over the magnetic quantum numbers are now required, Table 3 shows that the maximum errors for the floating point calculations rise rapidly with increasing J. Hence, the MPQ method is recommended beyond about $J = 30$ when CG coefficients with $m_3 \neq 0$ are required.

Given that for arbitrary magnetic quantum numbers the total number of CG coefficients grows as $O(J^5)$, storing all pre-calculated coefficients up to $J = 100$ (4,273,412,111 coefficients) or $J = 200$ (132,320,314,221 coefficients), for example, would require several tens or hundreds of gigabytes of memory, respectively. This would exceed the available memory in all but the largest of contemporary computers. On the other hand, since the number of top coefficients grows only as $O(J^3)$ (Table 1), it would be quite feasible to store exact pre-calculated top coefficients to very high order, and to calculate the rest on-the-fly when needed. It would be straight-forward to adapt the C code in the provided test program in order to do this in a particular application.

4. Conclusion

Novel whole number recursion formulae have been presented for the exact calculation of CG coupling coefficients to arbitrarily high order. Depending on the highest principal quantum number, these recursion formulae have been shown to be up to four orders of magnitude faster than an exact prime factorisation approach. In the approach presented here, CG coefficients can be calculated exactly by rational number recursion at a rate of around 10^6 per second on one CPU core of a contemporary workstation. Calculating CG coefficients for the common case in which $m_3 = 0$ involves only relatively short recursion runs, and this can be performed in C long double or even double precision with reasonable numerical precision up to order $J = 200$,

or $J = 130$, respectively. On the other hand, if accurate CG coefficients with $m_3 \neq 0$ are required, then the use of exact whole number recursion is recommended beyond about $J = 30$. To circumvent the need for the very large computer memory necessary to store all CG coefficients to high order, it is proposed to pre-calculate and store only a modest number of top coefficients, from which the rest may be calculated on-the-fly, as required.

5. Competing Interests

The author declares no competing interests.

References

[1] A. R. Edmonds, Angular Momentum in Quantum Physics, Princeton University Press, New Jersey, 1957.

[2] M. E. Rose, Elementary Theory of Angular Momentum, Wiley, New York, 1957.

[3] L. C. Biedenharn, J. C. Louck, Angular Momentum in Quantum Physics, Addison-Wesley, Reading, MA, 1981.

[4] Y.-L. Xu, Fast evaluation of Gaunt coefficients: recursive approach, Journal of Computational and Applied Mathematics 85 (1997) 53–65.

[5] D. Sébilleau, On the computation of integrated products of three spherical harmonics, Journal of Physics A: Mathematical and General 31 (1998) 7157–7168.

[6] M. Danos, L. C. Maximon, Multipole matrix elements of the translation operator, Journal of Mathematical Physics 6 (1) (1965) 766–778.

[7] D. Sébilleau, New recurrence relations for matrix elements of the propagator, Journal of Physics: Condensed Matter 7 (1995) 6211–6220.

[8] Y.-L. Xu, Efficient evaluation of vector translation coefficients in multiparticle light scattering theories, Journal of Computational Physics 139 (1998) 137–165.

[9] M. A. Epton, B. Dembart, Multipole translation theory for the three-dimensional Laplace and Helmholtz equations, SIAM Journal on Scientific Computing 16 (1995) 865–897.

[10] D. W. Ritchie, High-order analytic translation matrix elements for real-space six-dimensional polar Fourier correlations, J. Appl. Cryst. 38 (2005) 808–818.

[11] K. Schulten, R. G. Gordon, Recursive evaluation of $3j$ and $6j$ coefficients, Computer Physics Communications 11 (1976) 269–278.

[12] J. H. Luscombe, M. Luban, Simplified recursive algorithm for Wigner $3d$ and $6j$ symbols, Physics Review E 57 (1998) 7274–7277.

[13] A. J. Stone, C. P. Wood, Root-rational-fraction package for exact calculation of vector-coupling coefficients, Computer Physics Communications 21 (1980) 195–205.

[14] S.-T. Lai, Y.-N. Chiu, Exact computation of the 3-j and 6-j symbols, Computer Physics Communication.

[15] D. Ritchie, High Performance Algorithms for Molecular Shape Recognition, Habilitation à diriger des recherches, Université Henri Poincaré - Nancy I (Apr. 2011).
URL `https://tel.archives-ouvertes.fr/tel-00587962`

[16] H. T. Johansson, C. Forssén, Fast and accurate evaluation of Wigner $3j$, $6j$, and $9j$ symbols using prime factorization and multiword integer arithmetic, SIAM J. Sci. Comput. 38 (2016) A376–A384.

[17] I. I. Guseinov, A. Özmen, U. Atav, H. Yüksel, Computation of Clebsch-Gordan and Gaunt coefficients using binomial coefficients, Journal of Computational Physics 122 (1995) 343–347.

[18] L. Wei, Unified approach for exact calculation of angular momentum coupling and recoupling coefficients, Computer Physics Communications 120 (2–3) (1999) 222–230.

[19] C. C. J. Roothaan, New algorithms for calculating $3n$-j symbols, International Journal of Quantum chemistry 27 (1993) 13–24.

[20] R. E. Tuzun, P. Burkhardt, D. Secrest, Accurate computation of individual and tables of 3-j and 6-j symbols, Computer Physics Communications 112 (1998) 112–148.

[21] C. C. J. Roothaan, S.-T. Lai, Calculation of $3n-j$ symbols by Labarthe's method, International Journal of Quantum chemistry 63 (1997) 57–64.

[22] G. Racah, Theory of complex spectra. II, Physical Review 9–10 (1942) 438–462.

[23] J. J. P. Stewart, Mopac, http://openmopac.net/manual/FORTRAN_CG.html, accessed: 04-Aug-2017.

[24] J. J. Sakurai, Modern Quantum Mechanics, Addison-Wesley, Reading, MA, 1994.

[25] T. Granlund *et al.*, GNU MP: The GNU multiple precision arithmetic library (2013).
URL `http://gmplib.org/`

[26] T. A. Heim, J. Hinze, A. R. P. Rau, Some classes of 'notrivial zeros' of angular momentum addition coefficients, Journal of Physics A: Mathematical and Theoretical 42 (2009) 175203.

Table 1: Top CG coefficient calculation times ($m_3 = 0$). MPQ: exact whole number recursion calculation time; D80: long double precision recursion calculation time; D64: double precision recursion calculation time.

J	No. Coeffs	MPQ/s	D80/s	D64/s
10	506	0.00	0.00	0.00
20	3,311	0.00	0.00	0.00
50	45,526	0.04	0.00	0.00
100	348,551	0.36	0.00	0.00
150	1,159,076	1.23	0.01	0.01
200	2,727,101	3.26	0.03	0.02

Table 2: CG coefficient calculation times and relative errors for all CG coefficients for $m_3 = 0$ to order $j_1 = J$. Columns are labeled as SYM: exact symbolic factorial sum calculation time; MPQ: exact whole number recursion calculation time; D80: long double recursion calculation time; E_R80: the largest relative error between the calculation in long double precision and the exact MPQ calculation; E_N80: the largest normalisation error when using long double precision arithmetic; D64: double recursion calculation time; E_R64: the largest relative error between the calculations in double precision and the MPQ calculation; E_N64: the largest normalisation error when using double precision calculation. Calculation times are in seconds on one core of a 2.33 GHz Intel E5410 processor. The double precision calculation fails with an exponent overflow in the normalisation factor at $(j_1 = 140, j_2 = 140, j_3 = 268)$.

J	No. Coeffs	SYM/s	MPQ/s	D80/s	E_R80	E_N80	D64/s	E_R64	E_N64
10	5,786	0.19	0.00	0.00	1.1e-16	2.2e-19	0.00	6.3e-15	6.7e-16
20	71,071	2.79	0.02	0.00	3.2e-16	4.9e-19	0.00	6.3e-13	1.0e-15
30	327,856	16.07	0.08	0.01	9.8e-16	1.0e-18	0.00	5.9e-12	1.9e-15
40	988,141	68.54	0.25	0.02	7.2e-16	1.3e-18	0.01	9.7e-12	4.9e-15
50	2,343,926	187.44	0.59	0.04	1.8e-14	3.5e-18	0.03	1.0e-10	1.0e-14
60	4,767,211	484.55	1.21	0.08	1.8e-13	5.1e-18	0.05	1.2e-10	1.1e-14
70	8,709,996	1,123.45	2.20	0.14	1.9e-13	7.9e-18	0.10	1.2e-10	1.3e-14
80	14,704,281	2,342.09	3.71	0.24	2.2e-13	7.9e-18	0.17	2.2e-10	2.1e-14
90	23,362,066	4,686.93	5.87	0.39	2.2e-13	7.9e-18	0.27	6.7e-10	2.9e-14
100	35,375,351	8,681.99	8.87	0.58	3.5e-13	1.6e-17	0.41	6.7e-10	2.9e-14
110	51,516,136	15,158.65	12.92	0.86	3.9e-13	1.9e-17	0.59	9.1e-10	2.9e-14
120	72,636,421	27,082.28	18.20	1.19	3.0e-12	1.9e-17	0.83	2.1e-09	3.2e-14
130	99,668,206	42,233.32	24.98	1.65	3.0e-12	2.3e-17	1.21	2.1e-09	5.0e-14
140	133,623,491	64,360.78	33.43	2.22	3.0e-12	2.3e-17	–	–	–
150	175,594,276	95,805.43	43.88	2.88	9.7e-12	2.5e-17	–	–	–
160	226,752,561	140,055.47	56.99	3.72	9.7e-12	3.0e-17	–	–	–
170	288,350,346	199,763.60	72.07	4.77	2.9e-11	3.3e-17	–	–	–
180	361,719,631	279,728.38	90.28	5.94	2.9e-11	4.2e-17	–	–	–
190	448,272,416	385,406.59	111.99	7.35	2.9e-11	4.2e-17	–	–	–
200	549,500,701	532,533.66	137.28	9.05	2.9e-11	4.2e-17	–	–	–

Table 3: CG coefficient calculation times and relative errors for all CG coefficients up to order $j_1 = J$, for all allowed values of m_1, m_2, and $m_3 = m_1 + m_2$. Column headings are the same as for Table 2. This table is truncated after $J = 80$ due to the very long calculation times for the symbolic factorial sum method (column SYM).

J	No. Coeffs	SYM/s	MPQ/s	D80/s	E_R80	E_N80	D64/s	E_R64	E_N64
10	74,162	0.62	0.08	0.00	4.2e-16	3.3e-19	0.00	6.3e-13	8.9e-16
20	1,763,223	33.92	1.47	0.05	2.6e-11	6.5e-19	0.05	7.3e-08	1.6e-15
30	12,067,184	463.72	10.03	0.33	2.7e-06	1.5e-17	0.37	9.9e-03	2.5e-10
40	48,226,045	3,233.98	41.56	1.49	6.3e-01	5.4e-09	1.50	1.1e+03	6.6e-02
50	142,519,806	15,104.24	125.90	4.30	1.1e+05	1.4e+00	–	–	–
60	347,068,467	53,892.01	311.50	–	–	–	–	–	–
70	738,632,028	159,035.55	673.90	–	–	–	–	–	–
80	1,423,410,489	407,127.04	1321.73	–	–	–	–	–	–