Programs for generating Clebsch–Gordan coefficients of SU(3) in SU(2) and SO(3) bases

C. Bahri a,*, D.J. Rowe a, J.P. Draayer b

a Department of Physics, University of Toronto, Toronto, Ontario M5S 1A7, Canada
b Department of Physics and Astronomy, Louisiana State University, Baton Rouge, LA 70803, USA

Received 18 December 2003; accepted 24 January 2004

Abstract

Computer codes are developed to calculate Clebsch–Gordan coefficients of SU(3) in both SU(2)- and SO(3)-coupled bases. The efficiency of this code derives from the use of vector coherent state theory to evaluate the required coefficients directly without recursion relations. The approach extends to other compact semi-simple Lie groups. The codes are given in subroutine form so that users can incorporate the codes into other programs.

Program summary

Title of program: SU3CGVCS
Catalogue identifier: ADTN
Program summary URL: http://cpc.cs.qub.ac.uk/summaries/ADTN
Program obtainable from: CPC Program Library, Queen’s University of Belfast, N. Ireland
Licensing provisions: Persons requesting the program must sign the standard CPC non-profit use license
Computers for which the program is designed and on which it is operable: SGI Origin 2000, HP Apollo 9000, Sun, IBM SP, Pentium
Operating systems under which the program has been tested: IRIX 6.5, HP UX 10.01, SunOS, AIX, Linux
Programming language used: FORTRAN 77
Memory required to execute with typical data: On the HP system, it requires about 732 KBytes.
Disk space used for output: 2100+2460 bytes
No. of bits in a word: 32 bit integer and 64 bit floating point numbers.
No. of processors used: 1
Has the code been vectorized: No
No. of bytes in distributed program, including test data, etc.: 26 309
No. of lines in distributed program, including test data, etc.: 3969
Distribution format: tar gzip file

* Corresponding author. Present address: Department of Physics and Astronomy, Louisiana State University, Baton Rouge, LA 70803, USA.
E-mail addresses: bahri@physics.utoronto.ca, bahri@lsu.edu (C. Bahri), rowe@physics.utoronto.ca (D.J. Rowe), draayer@lsu.edu (J.P. Draayer).

0010-4655/ – see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.cpc.2004.01.005
Nature of physical problem: The group SU(3) and its Lie algebra $\mathfrak{su}(3)$ have important applications, for example, in elementary particle physics, nuclear physics, and quantum optics [1–3]. The code presented is particularly relevant for the last two fields. Clebsch–Gordan (CG) coefficients are required whenever the symmetries of many-body systems are used for the evaluation of matrix elements of tensor operators. Moreover, the construction of CG coefficients for SU(3) serves as a nontrivial prototype for larger compact semi-simple Lie algebras and even for non semi-simple Lie algebras. It is the simplest Lie algebra to have multiplicity in its outer products and a non-canonical subalgebra, i.e., SO(3).

Method of solution: Vector coherent state theory is first used to construct bases for the products of two irreducible representations (irreps) [4]. The bases are SU(2)-coupled so that SU(2)-reduced CG (or isoscalar factors) can be constructed naturally. The CG coefficients in the SO(3) bases are constructed subsequently from the overlaps between the SU(2) and SO(3) bases.

Restriction on the complexity of the problem: The programs are limited by computer memory and the maximum size of variable arrays. As dimension overflow conditions are possible, they are flagged and can be fixed by following the directions given as part of the error message.

Typical running time: The calculation time for a single SU(3) CG coefficient is very different for SU(2) and SO(3) bases. It varies between 7.3–54.1 ns in SGI Origin 2000, 0.81–5.48 ms in HP Apollo 9000, or 0.055–0.373 ms in Intel Pentium 4 for SU(2) bases while it is between 0.027–0.255 s in Intel Pentium 4 for SO(3) bases.

Unusual features of the program: Intrinsic bit functions: and, or, and shift, called iand, ior, and ishift, respectively, in FORTRAN, are used for packing and unpacking the labels for the irreps. Intrinsic logical btest is used to test the bit for the phase factor.

References:

© 2004 Elsevier B.V. All rights reserved.

PACS: 02.20.Qs; 02.20.Sv; 03.65.Fd; 21.60.Fw

Keywords: SU(3); SU(2); SO(3); Clebsch–Gordan coefficients; Wigner coefficients; Vector coupling coefficients; Coupling coefficients; Isoscalar factors

1. Introduction

Algorithms to generate the Clebsch–Gordan (CG) coefficients for SU(3) in canonical SU(2) and non-canonical SO(3) bases have been given recently within the framework of vector coherent state (VCS) theory [1]. The algorithms can be adapted to the calculation of CG coefficients for other semi-simple Lie groups/algebras. Thus, the algorithms for SU(3) serve as a non-trivial prototype for the application of VCS theory to the computation of CG coefficients more generally.

The group SU(3) and its Lie algebra $\mathfrak{su}(3)$ play central roles both in nuclear [2] and elementary particle physics [3]. Applications of SU(3) and other unitary groups are also useful in the analysis of quantum interferometers [4]. Applications in elementary particle physics normally require small SU(3) irreducible representations (irreps) and the Clebsch–Gordan (CG) coefficients for such applications are readily tabulated. But in nuclear physics as well as in quantum interferometry, the irreps can be very large and may approach classical asymptotic limits.

Furthermore, SU(3) occurs as a physically significant subgroup of many groups needed in physical theories. For example, the subgroup chains $U(\Omega) \supset SU(3) \supset SO(3)$ [2]—with $\Omega = (n+1)(n+2)/2$ for the $n$th oscillator shell—and $Sp(3, R) \supset SU(3) \supset SO(3)$ enter in the microscopic theory of nuclear collective motion [5]. The chain $U(6) \supset SU(3) \supset SO(3)$ is also used in the nuclear interacting boson model [6]. In such applications, one needs the matrices of SU(3) irreps and CG coefficients for reducing the tensor products of irreps. SU(3) CG coefficients are
also needed for computing matrix elements of various operators when such operators are expressible in terms of components of irreducible SU(3) tensors [7,8].

Despite a long series of mathematical articles on computing SU(3) irreps and their CG coefficients (see references in [1]), few computer codes are publicly available (cf. [9] and [10]). And among the available codes, there exists only one in a SO(3) basis [9].

The present article provides a user guide to a new computer code—both in the SU(2) and SO(3) bases—based on Ref. [1].

The algorithms on which the current code is based differ in two fundamental respects from previous algorithms. In the first place the basis states are constructed explicitly in a basis that reduces the SU(2) on Ref. [1], few computer codes are publicly available (cf. [9] and [10]). And among the available codes, there exists only one in a SO(3) basis [9].

The algorithms on which the current code is based differ in two fundamental respects from previous algorithms. In the first place the basis states are constructed explicitly in a basis that reduces the SU(2) ⊂ SU(3) subgroup. Construction of SU(2)-reduced CG coefficients is then very natural and efficient. Secondly, the desired CG coefficients are viewed in terms of transformations that reduce the subgroup chain SU(3) × SU(3) ⊂ SU(3); this is facilitated by the ease with which highest weight states for coupled irreps in the tensor product space of two irreps are identified in a VCS representation.

2. Mathematical preliminaries

Detailed derivations of various formulae in this section are given in Ref. [1]. Here, we give only some results that are directly relevant to the package.

2.1. Irreducible representations of the su(3) algebra and basis states

Diagonal operators $[\hat{C}_{11} - \hat{C}_{22}, \hat{C}_{22} - \hat{C}_{33}]$ together with raising operators $[\hat{C}_{ij}; i < j]$ and lowering operators $[\hat{C}_{ij}; i > j]$ form a standard basis for the su(3) algebra. A highest weight state $|\lambda, \mu\rangle$ for a su(3) irrep of highest weight $(\lambda, \mu)$ satisfies the equations

\begin{align}
\hat{C}_{ij}|\lambda, \mu\rangle &= 0, \quad i < j; \ i, j = 1, 2, 3, \tag{1} \\
(\hat{C}_{11} - \hat{C}_{22})|\lambda, \mu\rangle &= \lambda|\lambda, \mu\rangle, \tag{2} \\
(\hat{C}_{22} - \hat{C}_{33})|\lambda, \mu\rangle &= \mu|\lambda, \mu\rangle. \tag{3}
\end{align}

Basis states for the irrep with highest weight state $|\lambda, \mu\rangle$ are conveniently labelled by the quantum numbers of the subgroup chain

$$\text{SU}(3) \supset \text{SU}(2) \supset \text{U}(1)$$

where $\text{SU}(2) \equiv \text{SU}(2)_{23}$ is the subgroup with Lie algebra spanned by $[\hat{C}_{22} - \hat{C}_{33}, \hat{C}_{23}, \hat{C}_{32}]$ and $j$ is a multiplicity index. The index $j$ is associated with the grade of a state; it is assigned a value $j = 0$ for the so-called highest grade states, which are states that are annihilated by the raising operators $[\hat{e}_2 = \hat{C}_{12}, \hat{e}_3 = \hat{C}_{13}]$, and values $j = \frac{1}{2}, 1, \frac{3}{2}, \ldots$ for states of lower grade. The space of states of a given grade is invariant under SU(2)23. In particular, the highest grade states span a SU(2)23 irrep of spin $s = \mu/2$ with basis states $|((\lambda, \mu)0s m = s)\rangle$. Thus, the highest weight state $|\lambda, \mu\rangle$ is the state $|((\lambda, \mu)0sm = s)\rangle$. Basis states of lower grade, with $j > 0$, are obtained from the highest grade states $|((\lambda, \mu)0sm)\rangle$ by application of $2j$ lowering operators $[\hat{f}_i = \hat{C}_{1i}; i = 2, 3]$ to form the states

$$|((\lambda, \mu)jIN) = \frac{1}{K^{(j)}} \left[ \tilde{P}^j (\hat{f}) \otimes |((\lambda, \mu)0s)\rangle \right]_N$$

with

$$\tilde{P}^j (\hat{f}) = \frac{((\hat{f}_2)^{j+n}(\hat{f}_3)^{-n}}{(j+n)!/(j-n)!},$$

$$\sqrt{(j+n)!/(j-n)!}.$$
where $\tilde{P}_m^j$ is a spin-$j$ lowering tensor. The normalization factor $K(j \mu I)$, which takes values

$$K(j \mu I) = \sqrt{\frac{\lambda + \mu + 1)! \lambda!}{(\lambda + s + I - j + 1)!(\lambda + s - I - j)!}}$$

is derived by VCS methods [11].

The weights of states belonging to an irrep of highest weight $(\lambda \mu)$ lie on lattice points bounded by a hexagon whose sides are of length $\lambda$ and $\mu$ in Fig. 1. Among the range $I = |s - j|, |s - j| + 1, \ldots, s + j$, of $I$ values in Eq. (5) not all values occur; the excluded values are those for which $K(j \mu I)$ vanishes. Thus, the range for $I$ becomes

$$I = |s - j|, |s - j| + 1, \ldots, \min(s + j, s + \lambda - j).$$

The SU(2)-reduced matrix elements for the infinitesimal generators, defined by the Wigner–Eckart theorem

$$\langle (\lambda \mu) j' I' || \hat{X} || (\lambda \mu) j I \rangle = -\delta_{j', j} \delta_{I', I} \frac{1}{2\sqrt{s}} (2\lambda + \mu - 6 j),$$

$$\langle (\lambda \mu) j I || \hat{I} || (\lambda \mu) j' I' \rangle = \delta_{j', j} \delta_{I', I} \sqrt{I(I + 1)},$$

$$\langle (\lambda \mu) j I || \hat{e} || (\lambda \mu) j' I' \rangle = \delta_{j', j + \frac{1}{2}} \left( -\delta_{I', I + \frac{1}{2}} \sqrt{\frac{(\lambda + s - I - j)(s + I + j + 2)}{2I + 1} \sqrt{I + j - s + 1}} + \delta_{I', I - \frac{1}{2}} \sqrt{\lambda + s + I - j + 1} \sqrt{\frac{(s - I + j + 1)(I - j + s)}{2I + 1}} \right),$$

without the customary factor $\sqrt{2J_3 + 1}$, are then
\[
\langle (\lambda, \mu) | I \| f \rangle = \delta_{f, f'} \frac{1}{2} \left( \sqrt{\frac{(\lambda + s + I - j + 1)(s - I + j + 1)}{2I + 2}} \sqrt{I - j + s} \right.
+ \delta_{f, f'} \frac{1}{2} \sqrt{\lambda + s - I - j} \left. \sqrt{\frac{(s + I + j + 2)(I + j - s + 1)}{2I}} \right),
\]

(13)

The operator \( \hat{X} \) is a grading operator

\[
\hat{X} = -\frac{1}{2\sqrt{3}} (2\hat{C}_{11} - \hat{C}_{22} - \hat{C}_{33}),
\]

(14)

which in particle physics is also called strangeness. Together, the commuting generators \( \hat{X} \) and \( \hat{I}_0 \) span the Cartan subalgebra of SU(3). The operators \( \hat{e}_i \) = \( \hat{C}_{1i} \), \( i = 2, 3 \), which are simply the Hermitian conjugates of \( \hat{f}_i \), are raising operators.

### 2.2. Highest grade states in the tensor product space

A SU(3) irrep of highest weight \((\lambda_3, \mu_3)\) can have multiple occurrences in the tensor product of two SU(3) irreps \((\lambda_1, \mu_1) \otimes (\lambda_2, \mu_2)\). Such multiple occurrences of equivalent irreps are distinguished by a so-called outer multiplicity index \( \rho \). SU(3)-coupled basis states for a \((\lambda_1, \mu_1) \otimes (\lambda_2, \mu_2)\) tensor product space are then expanded

\[
|\rho(\lambda_3, \mu_3) j_3 N_3 \rangle = \sum_{j_1 j_2 N_1 N_2} \langle (\lambda_1, \mu_1) j_1 I_1 N_1; (\lambda_2, \mu_2) j_2 I_2 N_2|\rho(\lambda_3, \mu_3) j_3 N_3 \rangle
\times |(\lambda_2, \mu_2) j_2 I_2 N_2|\langle (\lambda_1, \mu_1) j_1 I_1 N_1 \rangle,
\]

(15)

where \( \langle (\lambda_1, \mu_1) j_1 I_1 N_1; (\lambda_2, \mu_2) j_2 I_2 N_2|\rho(\lambda_3, \mu_3) j_3 N_3 \rangle \) is a SU(3) CG coefficient which can be expressed as a product

\[
\langle (\lambda_1, \mu_1) j_1 I_1 N_1; (\lambda_2, \mu_2) j_2 I_2 N_2|\rho(\lambda_3, \mu_3) j_3 N_3 \rangle = \langle I_1 N_1; I_2 N_2|I_3 N_3 \rangle \times \langle (\lambda_1, \mu_1) j_1 I_1; (\lambda_2, \mu_2) j_2 I_2|\rho(\lambda_3, \mu_3) j_3 I_3 \rangle
\]

(16)

of a SU(2) CG coefficient \( \langle I_1 N_1; I_2 N_2|I_3 N_3 \rangle \) and a SU(2)-reduced CG coefficients for SU(3) \( \langle (\lambda_1, \mu_1) j_1 I_1; (\lambda_2, \mu_2) j_2 I_2|\rho(\lambda_3, \mu_3) j_3 I_3 \rangle \).

The \( J \) labels satisfy the following relation

\[
j_3 = j_1 + j_2 - \frac{1}{6}[2\lambda_1 + \mu_1 + 2\lambda_2 + \mu_2 - 2\lambda_3 - \mu_3].
\]

(17)

Computation of the reduced coupling coefficients starts with reduced coupling coefficients (expressed in curly brackets) for a non-orthogonal basis of highest grade states in the coupled space. These coefficients are easy to derive in a VCS representation in which the highest grade states are easily identified (details are given in Ref. [1]) and are given by

\[
\left\{ (\lambda_1, \mu_1) j_1 I_1; (\lambda_2, \mu_2) j_2 I_2 | \hat{i}(\lambda_3, \mu_3) 0 s_3 \right\} = \left(\frac{-1)^{2j_1}}{K_{\lambda_1}^{\lambda_1} j_1 I_1} K_{\lambda_2}^{\lambda_2} j_2 I_2 \right) \sum_{s_1, s_2, s_3} \left[ \begin{array}{ccc} s_1 & s_2 & s \\ j_1 & j_2 & j \\ I_1 & I_2 & s_3 \end{array} \right] a_{s_1},
\]

(18)

where \( j = j_1 + j_2 \) and

\[
\begin{bmatrix} s_1 & s_2 & s \\ j_1 & j_2 & j \\ I_1 & I_2 & s_3 \end{bmatrix} = \sqrt{(2s + 1)(2j + 1)(2I_1 + 1)(2I_2 + 1)} \begin{bmatrix} s_1 & s_2 & s \\ j_1 & j_2 & j \\ I_1 & I_2 & s_3 \end{bmatrix}
\]

(19)
is a unitary SU(2) 9-j-symbol; the transformation \( \{a_{si}\} \) is chosen such that the sums

\[
\sum_s s \left[ \begin{array}{ccc} s_1 & s_2 & s \\ j_1 & j_2 & j \\ I_1 & I_2 & s_3 \end{array} \right] a_{si} = 0
\]  

vanish for all values of \( I_1 \) and \( I_2 \) for which the corresponding \( K \)-factors, \( K^{(\lambda_1\mu_1)}_{j_1 I_1} \) and \( K^{(\lambda_2\mu_2)}_{j_2 I_2} \), vanish. This somewhat subtle procedure, cf. Ref. [1], ensures that the coefficients of Eq. (18) generate a linearly-independent set of highest grade states. This is necessary because, any state for which the \( K \)-factors vanishes is excluded from the representation space. Choosing \( \{a_{si}\} \) coefficients that satisfy Eq. (20) ensures that, when this happens, the states that are retained are linearly independent. This complication becomes unnecessary when none of the \( K \)-factors vanish for any \( j_1 I_1 \) and \( j_2 I_2 \); it is then possible to set \( a_{sji} = \delta_{sji} \), where \( k \) indexes the values of \( s \) that occur.

In general, the coefficients (elements of the unitary 9-j-symbols) for the unknown \( \{a_{si}\} \) parameters form a rectangular matrix of the size \( m \times n \), where \( m \) is the number of equations that result from the excluded values of \( I_1 \) and \( I_2 \) (values for which the \( K \)-factors vanish) and \( n \) is the number of \( s \) values. The number \( m \) of equations can exceed the number of \( n \) unknowns. However, the equations are not all linearly independent. After applying the Gaussian elimination method, i.e. by subtracting one row of the equations from the preceding ones with appropriate factors, the coefficient matrix is transformed so that the non-zero elements of the matrix form an upper trapezoidal shape. Then, the set \( \{a_{si}\} \) can be obtained by using the so-called backsubstitution method. The number of solutions for \( \{a_{si}\} \) is precisely the same as the number of outer multiplicity \( \rho \).

Due to a partially stretched coupling of the 9-j-symbol in Eq. (19), the general expression of the 9-j-symbol as a sum over four running labels reduces to a sum over two labels only. The explicit expression for this special 9-j-symbol is given in Appendix C.

The CG coefficients (expressed in round brackets) are obtained by orthonormalizing Eq. (18) using Gram–Schmidt procedures to obtain the so-called seed coefficients

\[
\left( (\lambda_1\mu_1) j_1 I_1 ; (\lambda_2\mu_2) j_2 I_2 \| \rho (\lambda_3\mu_3) 0 s_3 \right) = \sum_i C_i^{(\lambda_1\mu_1)\lambda_3}(\lambda_2\mu_2) ; j_1 I_1 ; (\lambda_2\mu_2) j_2 I_2 \| i (\lambda_3\mu_3) s_3 \right). 
\]  

(21)

The arbitrariness of the Gram–Schmidt orthogonalization procedure is in principle immaterial. However, it is conventional and useful to choose CG coefficients such that the reduced matrix elements of the infinitesimal generators (elements of the \( \text{su}(3) \) Lie algebra) are non-zero only for \( \rho = 1 \). This is achieved in this code by first defining the \( \rho = 1 \) CG coefficients to be the linear combinations

\[
(\cdot j_1 I_1 ; j_2 I_2 \| \rho = 1) = N_0 \left[ a (\cdot j_1 I_1 ; j_2 I_2 \| s_{\text{max}}) + b (\cdot j_1 I_1 ; j_2 I_2 \| s_{\text{max}} - 1) \right],
\]  

(22)

with

\[
a = \sqrt{\mu_3 + 1} \left[ \frac{\lambda_3}{\lambda_3 + \mu_3 + 1} + \mu_3 \frac{\lambda_3 + \mu_3 + 1}{\lambda_3} \right],
\]

\[
b = \sqrt{\mu_3 + 1} \left[ - (\mu_3 + 2) \frac{\lambda_3}{\lambda_3 + \mu_3 + 1} + \lambda_3 \frac{\lambda_3 + \mu_3 + 1}{\lambda_3} \right].
\]  

(23)

\( (\cdot j_1 I_1 ; j_2 I_2 \| \rho) \) is a shorthand notation for \( ((\lambda_1\mu_1) j_1 I_1 ; (\lambda_2\mu_2) j_2 I_2 \| \rho (\lambda_3\mu_3) 0 s_3) \) and \( N_0 \) is a normalization constant (see Appendix A). Note that, although such a choice is required only for the \( (\lambda\mu) \otimes (11) \rightarrow (\lambda\mu) \) coefficients, it is convenient to make it for all coefficients. The \( \rho \geq 1 \) coefficients are subsequently orthogonalized to the \( \rho = 1 \) coefficients, i.e.

\[
(\cdot j_1 I_1 ; j_2 I_2 \| \rho) = N_0 \left[ (\cdot j_1 I_1 ; j_2 I_2 \| \rho) - \sum_{\sigma = 1}^{\rho - 1} (\cdot j_1 I_1 ; j_2 I_2 \| \sigma) \right].
\]
of a non-orthonormal Elliott basis

\[
\sum_{j_1^I, I_2^I} (\cdot j_1^I; \cdot j_2^I_2 \| \cdot j_1^I; \cdot j_2^I_2 \| \cdot j_1^I; \cdot j_2^I_2 \| \cdot j_1^I).
\]

(24)

It is important to note that the shift tensors [12] naturally resolve the outer multiplicity and that they satisfy the
CG coefficient criterion for the infinitesimal generators outlined above. This particular choice is implemented in
Refs. [9,11].

2.3. SU(2) bases

Once the seed coefficients are determined, other CG coefficients can be derived in a straightforward manner by
stepping down with the SU(3) lowering operators making use of the fact that matrix elements of these operators are
known in analytic form in the canonical SU(2)-coupled basis (cf. Eq. (13)). As shown, for example, in Ref. [11],
this gives the general reduced CG coefficients

\[
\left(\rho(\lambda \mu | \lambda \mu_{j j}) \right) = \frac{K_{j1}^{(\lambda \mu_{j j})} K_{j1}^{(\lambda \mu_{j j})}}{K_{j1}^{(\lambda \mu_{j j})}} \sum_{I_1, I_2} U \left( \frac{1}{2} j_1 I_1 j_2 - j_1 I_1 j_2 - j_1 I_1 j_2 \right)
\]

\[
\times \left( \cdot j_1^I; \cdot j_2^I_2 \right) \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \cdot j_1^I; \cdot j_2^I_2 \right)
\]

(25)

where \( U \left( j_1 j_2 j_3 \right) \) is a Racah \( U \)-coefficient. Note that \( j_3 = j_1 - j_1^I + j_2 - j_1^I \). In Eq. (25), Racah coefficients and the \( 9 j \)-symbols are of the stretched type.

2.4. SO(3) bases

An orthonormal SO(3) basis \( \{(\lambda \mu | \alpha L M) \} \) for a SU(3) irrep of highest weight \( \lambda \mu \) is defined by a linear transformation

\[
| \lambda \mu | \alpha L M \rangle = \sum_{K} | \lambda \mu | K L M \rangle \tilde{K}_{K \alpha}^{(\lambda \mu)} (L)
\]

(26)

of a non-orthonormal Elliott basis \( \{(\lambda \mu | K L M) \} \). The need for an index \( \alpha \) to resolve the multiplicity in the
SU(3) \( \downarrow \) SO(3) branching rule is a hallmark of a non-canonical subgroup reduction. The \( \tilde{K}_{K \alpha}^{(\lambda \mu)} \) matrix is determined by expanding the overlap matrix

\[
S_{K K'}^{(\lambda \mu)} = \langle \lambda \mu K L M | \lambda \mu K' L M \rangle = \sum_{\alpha} K_{K \alpha}^{(\lambda \mu)} (L) K_{K' \alpha}^{\star (\lambda \mu)} (L),
\]

(27)

and (arbitrarily) choosing the \( K_{K \alpha}^{(\lambda \mu)} (L) \) matrix to be the Hermitian square root of \( S_{K K'}^{(\lambda \mu)} \). The matrix \( K_{K \alpha}^{(\lambda \mu)} (L) \) is then defined by the equation

\[
\sum_{K} K_{K \alpha}^{(\lambda \mu)} (L) K_{K \beta}^{\star (\lambda \mu)} (L) = \delta_{\alpha \beta}.
\]

(28)

This choice of an orthonormal basis has the advantage that it approaches the Elliott basis and, hence, the physically
meaningful rotor-model basis, in the limit of large \( 2 \alpha + \mu \) in which the Elliott basis states become orthonormal [13].
Seed CG coefficients in this basis are derived from the SU(2) seeds of Eqs. (18) and (24) by means of the identity
\[
\left(\langle \lambda_1 \mu_1 | \alpha_1 L_1 M_1 \rangle ; \langle \lambda_2 \mu_2 | \alpha_2 L_2 M_2 \rangle \right)_{\rho \lambda_3 \mu_3} = \sum_{j_1 i_1 N_1, j_2 i_2 N_2} \left(\langle \lambda_1 \mu_1 | \alpha_1 L_1 M_1 \rangle j_1 i_1 N_1 \left| \langle \lambda_2 \mu_2 | \alpha_2 L_2 M_2 \rangle j_2 i_2 N_2 \right. \right) \\
\times \left(\langle \lambda_1 \mu_1 | j_1 i_1 N_1 \rangle L_1 N_1 | \lambda_2 \mu_2 | j_2 i_2 N_2 \rangle L_2 N_2 \right)_{\rho \lambda_3 \mu_3} (i_1 N_1, i_2 N_2 | s_3 s_3), \tag{29}
\]
where \(\langle \rho \lambda_3 \mu_3 \rangle\) is a highest weight state in the space of the \((\lambda_1 \mu_1) \otimes (\lambda_2 \mu_2)\) tensor product irrep and \((\langle \lambda \mu \rangle a LM | \langle \lambda \mu \rangle j IN \rangle)\) is the overlap between the SO(3) and SU(2) basis states. Reduced CG coefficients are then given generally by
\[
\left(\langle \lambda_1 \mu_1 | \alpha_1 L_1 \rangle ; \langle \lambda_2 \mu_2 | \alpha_2 L_2 \rangle \rho \lambda_3 \mu_3 \right)_{\alpha_3 L_3} = \sqrt{\frac{8 \pi^2}{2L + 1}} \sum_{M_1, M_2} \left(\langle \lambda_1 \mu_1 | \alpha_1 L_1 M_1 \rangle ; \langle \lambda_2 \mu_2 | \alpha_2 L_2 M_2 \rangle \rho \lambda_3 \mu_3 \right) \\
\times \sum_{K_3 \geq 0} \left(\langle L_1 M_1 ; L_2 M_2 | L_3 K_3 \rangle \tilde{K}_{K_3 \alpha_3}^{(\lambda_3 \mu_3)}(L) \right), \tag{30}
\]
The \(S^{(\lambda_\mu) L}_{K K'}\) matrix is determined from the expression
\[
S^{(\lambda_\mu) L}_{K K'} = \int \mathcal{D}_{K K'}^L(\Omega) \langle \lambda \mu | R(\Omega) | \lambda \mu \rangle d\Omega \\
= 8 \pi^2 \sum_{\nu=0}^{\mu} \frac{1}{\lambda + \nu + L + 1} \left(\frac{\mu}{\nu}\right) I\left(\mu - \nu, \frac{1}{2}(\mu + K)\right) I\left(\mu - \nu, \frac{1}{2}(\mu + K')\right) \\
\times \sum_{n=0}^{L-K} d(K, L, K', n) S\left(\lambda + \nu, L, \frac{1}{2}(K - K' + 2n)\right), \tag{31}
\]
where \(\mathcal{D}_{K K'}^L(\Omega)\) and \(R(\Omega)\) are a Wigner rotation matrix and a rotation, respectively, for \(\Omega \in SO(3)\). The functions \(d, I,\) and \(S\) are
\[
d(M, J, N, p) = (-1)^{M-N+p} \left(\frac{2 J}{J + M}\right)^{1/2} \left(\frac{2 J}{J + N}\right)^{-1/2} \left(\frac{J - M}{p}\right) \left(\frac{J + M}{J + N - p}\right), \tag{32}
\]
\[
I(p, q, \sigma) = \frac{1}{2^{p+q}} \sum_{n=1}^{p} (-1)^n \left(\frac{p}{\sigma - n}\right) \left(\frac{q}{n}\right), \tag{33}
\]
\[
S(p, q, \sigma) = \sum_{n=1}^{p} (-1)^n \left(\frac{p}{\sigma + n}\right) \left(\frac{p + q}{\sigma + n}\right)^{-1}, \tag{34}
\]
with non-vanishing values of \(I\) and \(S\) functions for integer arguments. The \(d\) function is expressed as a product of binomial coefficients, instead of factorials as in Ref. [1], for computational efficiency. The \(I\) function differs from that of Ref. [1] by a factor \(2 \pi (-i)^q\).

The overlaps between the SU(2)- and SO(3)-bases are given by
\[
\langle \langle \lambda \mu \rangle a LM | \langle \lambda \mu \rangle j IN \rangle \rangle = \sum_{K \geq 0} \tilde{K}_{K a}^{(\lambda \mu) K L M}(\lambda \mu) j \langle K \rangle \langle N \rangle, \tag{35}
\]
where
\[
\langle (\lambda \mu) K L M | (\lambda \mu) j IN \rangle = \sqrt{\frac{2L + 1}{8 \pi^2}} \int \mathcal{D}_{K M}^{L+}(\Omega) \langle \lambda \mu | R(\Omega) \rangle | (\lambda \mu) j IN \rangle d\Omega. \tag{36}
\]
The latter expression is evaluated to give

\[
\langle (\lambda \mu) KLM | (\lambda \mu) jIN \rangle = (-1)^j 2\pi (-i)^{j+N} 2^{j+1} \sqrt{\frac{2L+1}{2}} \times \sum_m C_m^{(\lambda \mu)} (jI) \frac{1}{2J + L + 1} \left(\frac{2J}{2j}\right)^{\frac{1}{2}} d(s, s, m, 0) I(s + m, s - m, s + \frac{1}{2} K) \\
\times \sum_p (-1)^p d(j + m, I, N, p) I(2I + N - j - m - 2p, j + m - N + 2p, I + \frac{1}{2} M) \\
\times \sum_q d(K, L, M, q) S \left(2J - 2j, 2j + L, \frac{1}{2}(K - M + 2q + 2j)\right),
\] (37)

where \(2J = \lambda + s - m\) and

\[
C_m^{(\lambda \mu)} (jI) = (-1)^{I + j + \frac{\lambda}{2} - \frac{\lambda}{2}} \sqrt{(2J + 1)(2I + 1)} \left\{ \begin{array}{c} \frac{\lambda + \mu - 2j}{2} \\ \frac{s - m}{2} \\ \frac{j}{2} \\ \frac{I}{2} \end{array} \right\}.
\] (38)

with \(s = \mu/2\) as usual.

It is worth noting that the 6\(j\)-symbol in Eq. (38) is of the stretched type.

3. Computational method

The implementation of the above formulae for computer codes is very straightforward. Nevertheless, several techniques are employed to increase the accuracy and efficiency of the performance that are particularly relevant for large SU(3) irreps.

The angular momenta of SU(2) take half integer values; therefore, twice their values are stored and integer arithmetic is used as often as possible, e.g., for the coupling and recoupling coefficients of SU(2) as in Eqs. (18), (25), (29), and (30).

Factorials involving large numbers enter frequently into the formulae and it is important to compute them accurately. They are calculated in floating point arithmetic and their arithmetic operations are executed with logarithms to preserve accuracy. One must also be concerned about the accuracy of the alternating sums that diminishes rapidly as the number of terms in the sums get larger, specifically in evaluation of the \(I\) and \(S\) functions.

The evaluation is central to the computation of CG coefficients and it is important to evaluate them rapidly and accurately. Thus, we evaluate them by means of recursion relations (details are given in Appendix B). The \(I\) function satisfies the following recursion relation

\[
I(p, q, \sigma) = \frac{1}{4} \left[ I(p - 1, q - 1, \sigma) - I(p - 1, q - 1, \sigma - 2) \right] \\
\] (39)

for \(p \geq q\) with a starting condition

\[
I(p, 0, \sigma) = \frac{1}{2^p} \left( \frac{p}{\sigma} \right).
\] (40)

For \(p < q\) we apply the symmetry of Eq. (33), i.e.

\[
I(p, q, \sigma) = (-1)^\sigma I(q, p, \sigma). \\
\] (41)
Similarly, the $S$ function satisfies the recursion relation

$$S(p, q, \sigma) = S(p-1, q+1, \sigma) - S(p-1, q+1, \sigma+1)$$

with a starting condition

$$S(0, q, \sigma) = \left( \frac{q}{\sigma} \right)^{-1}.$$  

The drawback with this approach is that the $S$ function depends on the succeeding values of $q$ and $\sigma$ instead of the preceding ones. To circumvent this problem, a different recursion relation is used for $q > \sigma$, i.e.

$$S(p, q, \sigma) = \frac{q-\sigma}{p+q} S(p, q-1, \sigma) + \frac{p}{p+q} S(p-1, q, \sigma).$$

Further reductions in computational time can be achieved by generating and using lookup tables for factorials, binomial coefficients, $I$, and $S$ functions.

As all the numbers are real, the phase factors in the algorithms are obtained from logical operations; for example, $(-1)^{M-N+p}$ is expressed in the following algorithm (in FORTRAN)

```fortran
if(btest(M - N + p, 0)) then
    phase = -1
else
    phase = +1
end if
```

where $btest(var, n)$ is a logical function which tests the $n$th bit of the integer variable $var$ with $0 \leq n \leq 31$. If the $n$th bit is 1 (on), the test succeeds; i.e. the logical function $btest$ return a $true$. value. The bit is read from the right to the left; thus, the bit $n = 0$ corresponds to the rightmost bit in the bit representation of an integer.

For book-keeping of the irrep labels, especially for the seed coefficients (which give the couplings to the highest grade states in the tensor product space), four integers corresponding to the labels of $SU(2)$ subirreps \{2$j_1$, 2$I_1$, 2$j_2$, 2$I_2$\} are stored as packed labels. We use `and`, `or`, and `shift` functions to achieve this task. The label packing function can be expressed as

$$pklabel = \text{pack}(a, b, c, d) = \text{shift}(a, 24) \text{ or } \text{shift}(b, 16) \text{ or } \text{shift}(c, 8) \text{ or } d$$

and the unpacking counterpart for the variable $pklabel$ is

$$a = \text{shift}(pklabel, -24) \text{ and } n8bits$$
$$b = \text{shift}(pklabel, -16) \text{ and } n8bits$$
$$c = \text{shift}(pklabel, -8) \text{ and } n8bits$$
$$d = \text{shift}(pklabel, 0) \text{ and } n8bits$$

where $n8bits = 255 = 11111111_2$. Using this method, one can save the storage for the labels by a factor of four, and in most applications the values for $j$ and $I$ do not reach 255.

4. Program structures

In addition to the sample main programs as drivers that will be described below, the SU3CGVCS package consists of the following main subprograms: `cgu3hw`, `cgu3`, and `cgu3o3`, with several important supporting
The subprograms: homgauss, orthon, klw, kmat, and ovrlap. The subprograms for the coupling and recoupling coefficients of SU(2) are also given in the package: dwr3, drr3, d9jr3, and d9jr3sp.

\texttt{attn(liter, *)}

This subprogram gives a warning message \texttt{litter} with alternative return. If it is an error message, the \texttt{entry error(liter)} is invoked and the program terminates.

\texttt{readfact}

The subprogram generates lookup tables for the logarithms of factorials and of binomial coefficients. The generation of the lookup tables is only done once in the main program.

\texttt{readtab}

The subprogram generates lookup tables for $I$- and $S$-functions, using recursive relations as outlined in Section 3. This subprogram is only needed for the calculations of CG coefficients in SO(3) bases and has to be called after the calling of \texttt{readfact}.

\texttt{dwr3(j1t, j2t, j3t, m1t, m2t, m3t)}

This function computes the CG coefficients of SU(2) (or SO(3)) $(j_1 m_1 ; j_2 m_2 | j_3 m_3 )$. The argument parameters for \texttt{dwr3} are integer variables $2j_1, 2j_2, 2j_3, 2m_1, 2m_2, \text{and } 2m_3$.

\texttt{drr3(j1t, j2t, l2t, l1t, j3t, l3t)}

This function computes the Racah coefficients of SU(2) (or SO(3)) $W(j_1 j_2 l_2 l_1 ; j_3 l_3 )$ with the argument parameters for the function $2j_1, 2j_2, 2l_2, 2l_1, 2j_3, \text{and } 2l_3$.

\texttt{d9jr3(j1t, j2t, j3t, j4t, j5t, j6, j7t, j8t, j9t)}

This function computes the 9$j$-symbols with the argument parameters for the function $2j_1, 2j_2, 2j_3, 2j_4, 2j_5, 2j_6, 2j_7, 2j_8, \text{and } 2j_9$.

\texttt{d9jr3sp(j1t, j2t, j3t, j4t, j5t, j6, j7t, j8t, j9t)}

This function computes the special stretched 9$j$-symbols in Eq. (C.2) for which $j_7 = j_1 + j_4$. (The function \texttt{d9jr3} is actually never used in the package because only the stretched 9$j$-symbols are required.)

\texttt{u3mult(lm1, mu1, lm2, mu2, lm3, mu3, kromax, *)}

This subroutine gives the multiplicity of the outer SU(3) product $(\lambda_1 \mu_1) \otimes (\lambda_2 \mu_2) \rightarrow (\lambda_3 \mu_3)$, kromax. The first six entries for the argument are the SU(3) irrep labels. An alternative return branch is activated when the coupling is not allowed, i.e. kromax = 0. The multiplicity can also be obtained indirectly from the only allowed couplings of the 9$j$-symbols in Eq. (18).

\texttt{homgauss(nrow, ncol, nmax, dmtx, dsol)}

This subroutine determines the solutions of \texttt{nrow} simultaneous homogeneous linear equations using the Gaussian elimination procedure. Initially, \texttt{nrow} is the number of equations. The value of \texttt{nrow} returned is the number of linearly independent equations. The number of independent variables is denoted by \texttt{ncol}. The coefficients of the variables are passed as \texttt{nrow x ncol} matrix \texttt{dmtx(*,*)} whose physical dimension is \texttt{nmax}. In return, \texttt{dmtx(*,*)} is an upper triangular matrix. The solutions of the variables are stored in \texttt{dsol(*,*)} whose physical dimension is \texttt{nmax}.

\texttt{orthon(nsize, nvec, nmax, dvec)}

This subroutine orthonormalizes \texttt{nvec} vectors of an array \texttt{dvec(*,*)} of length \texttt{nsize} using the Gram–Schmidt procedure. \texttt{dvec(i,j)} corresponds to the \texttt{i}th component of the \texttt{j}th vector. The physical dimension of \texttt{dvec(nmax,*)} is \texttt{nmax}.

\texttt{klw(lm, mu, jt, It)}

This function computes the factor $K_{ji}^{ji}$ as in Eq. (7). The arguments of the function \texttt{klw} are the integer variables $\lambda, \mu, 2j, \text{and } 2I$. 

This subroutine calculates the CG coefficients of SU(3) in SU(2)-bases
\( \text{cgu3}(\lambda \mu, \mu_1, \mu_2, \mu_3, n_{\text{hw}}, \text{ifu3e}, \text{dcgu3e}, n_{\text{max}}) \)

The first six entries for the argument are the irrep labels for SU(3). This subroutine calls the functions \( \text{d9jr3sp} \) and the subroutines \( \text{orthon}, \text{u3mult}, \) and \( \text{attn} \) in addition to the standard math functions.

\( \text{cgu3}(\lambda_1, \mu_1, \lambda_2, \mu_2, \lambda_3, \mu_3, j_1, I_1, j_2, I_2, j_3, I_3, n_{\text{hw}}, \text{ifu3e}, \text{dcgu3e}, n_{\text{max}}) \)

This subroutine calculates the CG coefficients of SU(3) in SU(2)-bases \( (|\lambda_1 \mu_1\rangle J_1 I_1; |\lambda_2 \mu_2\rangle J_2 I_2; |\lambda_3 \mu_3\rangle J_3 I_3) \) using Eq. (25). In addition to the parameters defined as for \( \text{cgu3hw} \), the other parameters correspond to \( J_1, I_1, J_2, I_2, J_3, I_3 \) and the CG coefficients are stored in an array \( \text{dcgu3} (*) \). The actual dimension of \( \text{dcgu3}(*) \) is determined by the multiplicity of the outer product \(|\lambda_1 \mu_1\rangle \otimes |\lambda_2 \mu_2\rangle \rightarrow |\lambda_3 \mu_3\rangle \). Calling this subroutine must be preceded by the call of \( \text{cgu3hw} \). This subroutine calls the functions \( \text{klw}, \text{d9jr3sp} \) and the subroutine \( \text{u3mult} \).

\( \text{cgu3o3}(\lambda_1, \mu_1, \lambda_2, \mu_2, \lambda_3, \mu_3, L_1, L_2, L_3, n_{\text{hw}}, \text{ifu3e}, \text{dcgu3e}, \text{dcgu3o3}, n_{\text{max}}) \)

This subroutine calculates the CG coefficients of SU(3) in SO(3)-bases \( (|\lambda_1 \mu_1\rangle a_1 L_1; |\lambda_2 \mu_2\rangle a_2 L_2; |\lambda_3 \mu_3\rangle a_3 L_3) \) using Eq. (30). In addition to the parameters as in \( \text{cgu3hw} \), the other parameters correspond to \( L_1, L_2, L_3 \) and the CG coefficients are stored in an array \( \text{dcgu3o3}(*,*,*,*) \). The four arguments in the array \( \text{dcgu3o3}(*,*,*,*) \) correspond to the running multiplicity indices for \( a_1, a_2, a_3, \) and \( \rho \). Calling this subroutine must be preceded by the call of \( \text{cgu3hw} \). This subroutine calls the functions \( \text{ovrlap}, \text{d9jr3sp} \) and the subroutines \( \text{kmat} \) and \( \text{u3mult} \).

5. Using the program

Details of the installation and the executing procedures are given in the README file, which is also distributed along with the source codes.

The programs \( \text{cgu3drv} \) and \( \text{cgu3o3drv} \) are the drivers for the calculations of the reduced CG coefficients for SU(3) in the SU(2) and SO(3) bases, respectively. In addition to their function as sample programs, they are also used to test the overall performance of the package. The structure for both main programs are very similar. Upon initializing the code, the user is requested to choose from among the following options:

1. Direct calculation of a specific reduced CG coefficient \( (|\lambda_1 \mu_1\rangle a_1 L_1; |\lambda_2 \mu_2\rangle a_2 L_2; |\lambda_3 \mu_3\rangle a_3 L_3) \) with \( \rho = 1, 2, \ldots, \rho_{\text{max}} \), where \( a_i = j_i I_i \) for the SU(2) bases or \( a_i L_i \) for the SO(3) bases.
(2) The set of reduced CG coefficients for all \( \omega_3 \) with fixed \( \omega_1 \) and \( \omega_2 \). The total time \( T_{tot} \) is the time for computing all coefficients while \( T_{ave} \) is the average time per coefficient. The performance, corresponding to the speed of the calculations, is inversely proportional to the time.

(3) The set of all reduced CG coefficients for a fixed value of \( \omega_3 \).

(4) The set of all reduced CG coefficients within \( (\lambda_1 \mu_1) \otimes (\lambda_2 \mu_2) \rightarrow \bigoplus \rho (\lambda_3 \mu_3) \).

The input and output samples are given in Appendices D and E.

6. Program limitations and modification

The restriction of the arrays is determined by the main program because almost all of the array variables are passed as argument parameters for the subprograms. Nevertheless, some arrays are expressed explicitly with their dimensions in the subprograms kmat and cgu3o3. In kmat, the arrays \( xk1(MXK, MXK) \), \( xk2(MXK, MXK) \), \( e(MXK) \), and \( y(MXK) \) are used as temporary storage for the calculations of the eigenvalues and eigenvectors of the \( S \) - and \( K \) -matrices. \( MXK \) is the dimension of the matrices, and in the published code \( MXK = 10 \). Similarly, in cgu3o3, \( xkbar1(MXK, MXK) \), \( xkbar2(MXK, MXK) \), and \( xkbar3(MXK, MXK) \) are the temporary storage for \( \bar{K}_1 \), \( \bar{K}_2 \), and \( \bar{K}_3 \), respectively, for Eqs. (30) and (35). \( MXK = 10 \) puts a limit of \( \alpha \leq 10 \) where \( \alpha \) is the inner multiplicity label of the number of occurrences \( L \) in \( (\lambda, \mu) \).

Another important issue concerns the look-up tables for binomial coefficients. Since the binomial coefficients are stored in a look-up table array, one may encounter errors for very large irreps. This problem is especially likely to arise in the calculation of the reduced CG coefficients in the SO(3) bases. The limitation of the present code is \( \lambda + \mu + L \leq NMBIN \) where \( NMBIN \) is the number of all binomial coefficients in the lookup table.

For moderate size of the calculations that we are interested in, one does not need to change the dimensions. In any case, a message will be flagged whenever any dimension related problems arise in the routines.

In solving Eq. (20), one may encounter accuracy problems in finding linearly independent solutions due to the machine error of the computer. To fix this problem, should it occur, one must change the constant parameter \( \text{ZERO} \) in the subprogram homgauss. In the test we were performing, the error arose in the evaluation of a few SU(3) tensor products, e.g. \( (108 \ 6) \otimes (6 \ 4) \rightarrow (99 \ 13), (101 \ 6), (102 \ 4) \) in HP Apollo 9000 but not in other machines. This problem is really machine-dependent.

Improvements can be made to the package by various means. One may utilize the symmetry properties of the irreps, especially in evaluating information for irreps of highest weight \( (\lambda, \mu) \) when \( \lambda < \mu \). Another avenue is to substitute asymptotic expressions for the coefficients for large representations when they are known to be sufficiently accurate. And last but not least, one can store the results of expensive computations, especially the overlaps between SU(2) - and SO(3) -bases overlaps, into a binary tree. One can then retrieve already calculated values from the tree. For this purpose one can incorporate the so-called WSTREE package [15].

7. Performance characteristics

Several tests were performed to check the limitations and efficiency of the algorithms with emphasis on the calculation of CG coefficients for the coupling \( (A \mu) \otimes (pq) \rightarrow (A' \mu') \) with \( A \) and \( A' \) of the order of 100. Several scenarios were investigated and the results are shown in Table 1.
involving the adjoint representation
larger groups like SU(4) ≃ equivalent to 9j
Appendix A. The resolution of the outer multiplicity ρ for the reduced Clebsch–Gordan coefficients involving the adjoint representation

Table 1
The computing performance of SU3CGVCS for large irreps. The CG coefficients are evaluated for all \((\lambda\mu')\) in the SU(3) coupling product \((\lambda\mu) \otimes (pq) \rightarrow \bigoplus_{\rho(\lambda')\mu'} \rho(\lambda'\mu')\). The minimum performance normally occurs for non-stretched couplings, while the maximum performance is achieved for near stretched couplings. The values of \((\lambda'\mu')\) are given for the minimum and maximum performance. The performance of the Draayer–Akiyama code [9] is set to be 1. Better performance is characterized by larger numbers.

<table>
<thead>
<tr>
<th>((\lambda, \mu))</th>
<th>((p, q))</th>
<th>min</th>
<th>max</th>
<th>ave</th>
</tr>
</thead>
<tbody>
<tr>
<td>(108, 6)</td>
<td>(6, 4)</td>
<td>0.545</td>
<td>(100, 8)</td>
<td>4.044</td>
</tr>
<tr>
<td>(108, 6)</td>
<td>(4, 6)</td>
<td>0.467</td>
<td>(99, 8)</td>
<td>3.892</td>
</tr>
<tr>
<td>(6, 108)</td>
<td>(6, 4)</td>
<td>0.687</td>
<td>(2, 106)</td>
<td>3.129</td>
</tr>
<tr>
<td>(6, 108)</td>
<td>(6, 0)</td>
<td>1.651</td>
<td>(104, 11)</td>
<td>2.984</td>
</tr>
<tr>
<td>(108, 6)</td>
<td>(0, 6)</td>
<td>1.474</td>
<td>(103, 7)</td>
<td>3.409</td>
</tr>
<tr>
<td>(6, 108)</td>
<td>(6, 0)</td>
<td>0.957</td>
<td>(7, 107)</td>
<td>2.647</td>
</tr>
<tr>
<td>(6, 108)</td>
<td>(6, 0)</td>
<td>1.550</td>
<td>(11, 107)</td>
<td>2.015</td>
</tr>
<tr>
<td>(108, 0)</td>
<td>(6, 4)</td>
<td>1.428</td>
<td>(98, 6)</td>
<td>4.592</td>
</tr>
<tr>
<td>(108, 0)</td>
<td>(6, 4)</td>
<td>1.374</td>
<td>(98, 4)</td>
<td>4.466</td>
</tr>
<tr>
<td>(0, 108)</td>
<td>(6, 4)</td>
<td>1.745</td>
<td>(100, 106)</td>
<td>3.492</td>
</tr>
<tr>
<td>(0, 108)</td>
<td>(6, 4)</td>
<td>2.064</td>
<td>(100, 104)</td>
<td>3.675</td>
</tr>
</tbody>
</table>

8. Concluding remarks

The routines in the package are simple to use and they execute efficiently. The efficiency is a crucial point for various applications in many-body physics and quantum interferometry. This efficiency is attributed to the minimal usage of recursion relations (in fact, the seed coefficients are calculated directly).

The routines have been written such that they are easily incorporated into new and into existing programs. From the tests done for the given sample programs, the code is generally two to three times faster than the Draayer–Akiyama code [9] except for relatively few special cases. In a few situations, the results are more than four times faster. The Draayer–Akiyama code is faster for SU(3) couplings that are far from stretched couplings.

Further improvements are achieved primarily in the accuracy of the calculations. With the new techniques in the evaluation of alternating series, the usage of extended (quadruple) precision floating point numbers can be minimized.

The package can be extended easily to include the SU(3) Racah and recoupling coefficients (known as 9j, equivalent to 9j-symbols for SU(2)). These coefficients are needed for computation of the CG coefficients of larger groups like SU(4) ∼ SO(6), Sp(3) and Sp(3, R), using a similar algorithm.

Acknowledgements

The authors thank one of the referees for his/her careful and constructive comments, particularly in regard to combinatorial identities (with Ref. [16]) and critiques to improve speed and accuracy.

Appendix A. The resolution of the outer multiplicity \(\rho\) for the reduced Clebsch–Gordan coefficients involving the adjoint representation

The outer product \((\lambda\mu) \otimes (11) \rightarrow \rho(\lambda\mu)\) has a multiplicity of two for non-zero \(\lambda\) and \(\mu\). It is natural to identify \(\rho = 1\) with the coupling to elements of the algebra. Thus, the reduced matrix elements of the infinitesimal
generators are non-zero only for $\rho = 1$; specifically

$$\langle (\lambda \mu)_{0} s | \hat{\epsilon} | (\lambda \mu)_{1/2}, s \pm \frac{1}{2} \rangle = \langle (\lambda \mu)_{1/2}, s \pm \frac{1}{2} | (11)0_{2} \| \rho = 1(\lambda \mu)0_{s} \rangle \sqrt{2C_{2}^{(\lambda \mu)}} \quad (A.1)$$

with $s = \mu/2$ and

$$C_{2}^{(\lambda \mu)} = \frac{2}{3}(\lambda^{2} + \mu^{2} + \lambda \mu + 3\lambda + 3\mu). \quad (A.2)$$

It follows from Eq. (A.1) that the explicit expression for the matrix elements of $\hat{\epsilon}$, given by Eq. (13), defines the $\rho = 1$ reduced CG coefficients, and hence gives the $a = C_{s+1/2}^{(\lambda \mu)}$ and $b = C_{s-1/2}^{(\lambda \mu)}$ coefficients of Eq. (21) as solutions of the equation

$$\langle (\lambda \mu)_{1/2}, s \pm \frac{1}{2} | (11)0_{2} \| \rho = 1(\lambda \mu)0_{s} \rangle = a \{ (\lambda \mu)_{1/2}, s \pm \frac{1}{2} | (11)0_{2} \| s + \frac{1}{2}(\lambda \mu)0_{s} \} \quad + b \{ (\lambda \mu)_{1/2}, s \pm \frac{1}{2} | (11)0_{2} \| s - \frac{1}{2}(\lambda \mu)0_{s} \}. \quad (A.3)$$

With the explicit form of the non-orthogonal reduced coupling coefficients given by Eq. (18), this equation becomes

$$\langle (\lambda \mu)_{1/2}, s \pm \frac{1}{2} | (11)0_{2} \| 1(\lambda \mu)0_{s} \rangle = -\frac{1}{\sqrt{\lambda + \mu + 1}} \{ a \left[ \begin{array}{ccc} \mu/2 & 1/2 & (\mu + 1)/2 \\ 1/2 & 0 & 1/2 \\ (\mu + 1)/2 & 1/2 & \mu/2 \end{array} \right] + b \left[ \begin{array}{ccc} \mu/2 & 1/2 & (\mu + 1)/2 \\ 1/2 & 0 & 1/2 \\ (\mu - 1)/2 & 1/2 & \mu/2 \end{array} \right] \}, \quad (A.4)$$

$$\langle (\lambda \mu)_{1/2}, s \pm \frac{1}{2} | (11)0_{2} \| 1(\lambda \mu)0_{s} \rangle = -\frac{1}{\sqrt{\lambda + \mu + 1}} \{ a \left[ \begin{array}{ccc} \mu/2 & 1/2 & (\mu - 1)/2 \\ 1/2 & 0 & 1/2 \\ (\mu + 1)/2 & 1/2 & \mu/2 \end{array} \right] + b \left[ \begin{array}{ccc} \mu/2 & 1/2 & (\mu - 1)/2 \\ 1/2 & 0 & 1/2 \\ (\mu - 1)/2 & 1/2 & \mu/2 \end{array} \right] \}. \quad (A.5)$$

This leads to the matrix equation

$$\left( \begin{array}{c} -\frac{2(\mu + 2)}{\mu + 1} \\ \sqrt{2(\lambda + \mu + 1)/\mu + 1} \end{array} \right) = \frac{1}{\sqrt{\lambda + \mu + 1}} \left( \begin{array}{cc} \lambda + \mu + 1 & \mu + 1 \\ \mu + 1 & \lambda + \mu + 1 \end{array} \right) \left( \begin{array}{c} a \\ b \end{array} \right), \quad (A.6)$$

which, to within a normalization constant $N$, has solution

$$a = N \sqrt{\mu + 2} \left[ \frac{\lambda}{\sqrt{\lambda + \mu + 1}} + \mu \sqrt{\frac{\lambda + \mu + 1}{\lambda}} \right],$$

$$b = N \sqrt{\mu} \left[ -(\mu + 2) \frac{\lambda}{\sqrt{\lambda + \mu + 1}} + \mu \sqrt{\frac{\lambda + \mu + 1}{\lambda}} \right]. \quad (A.7)$$

**Appendix B. Evaluation of $I(p, q, \sigma)$ and $S(p, q, \sigma)$ functions (Eqs. (33) and (34))**

The evaluation of $I(p, q, \sigma)$ and $S(p, q, \sigma)$ functions is critical in obtaining meaningful results of the computation for CG coefficients in SO(3) bases. To illustrate this importance, Fig. B.1 shows how the accuracy diminishes rapidly, for some cases, as the arguments of $I$ and $S$ functions get larger.

Due to the accuracy problem arising in the alternating sums in Eqs. (33) and (34), the functions are expressed differently.
Fig. B.1. The accuracy of double precision computation of the $I(p,q,\sigma)$ function (hence, $I(p,q,\sigma)$) for $\sigma = 64$ with respect to extended (quadruple) precision one, using (a) direct sum in Eq. (B.2), (b) Eqs. (B.5) and (B.7), (c) recursive relation of Eq. (B.2), and (d) recursive relation of (b), i.e. Eq. (B.9).

The accuracy of the computation increases if the number of terms in an alternating sum decreases. First, let us apply this principle to evaluate $I$ function,

$$I(p, q, \sigma) := \frac{1}{2^{p+q}} I(p, q, \sigma)$$ \hspace{1cm} (B.1)

with

$$I(p, q, \sigma) = \sum_{n=\max(0,\sigma-p)}^{\min(q,\sigma)} (-1)^n \left( \begin{array}{c} p \\ \sigma - n \end{array} \right) \left( \begin{array}{c} q \\ n \end{array} \right).$$ \hspace{1cm} (B.2)

The last expression is given to emphasize that $I$ is an integer.
The binomial expansion of both sides of the identity

\[(1 + x)^p (1 - x)^q = (1 - x^2)^p (1 - x)^{q-p}, \quad (B.3)\]

with respect to \(x\), leads to

\[\sum_{mn} (-1)^n \binom{p}{m} \binom{q}{n} x^{m+n} = \sum_{mn} (-1)^{m+n} \binom{p}{m} \binom{q-p}{n} x^{2m+n}. \quad (B.4)\]

An evaluation of \(x^\sigma\) term in both sums gives the result for \(I(p, q, \sigma)\) and it can be expressed as

\[I(p, q, \sigma) = (-1)^\sigma \min\{p, [\sigma/2]\} \sum_{n=\max(0, [\sigma+p-q)/2\}] \binom{p}{n} \binom{q-p}{\sigma-2n} \quad (B.5)\]

for \(q \geq p\). Here, the limits of the sum are explicitly written to indicate the advantages of the latter expression. The floor \([n/2]\) and ceiling \([n]\) integer functions are defined as

- \([n/2]\) = \([n/2]\) = \(n/2\), \(n \) even,
- \([n/2]\) = \((n - 1)/2\), \(n \) odd,
- \([n/2]\) = \((n + 1)/2\), \(n \) odd.

The expression in Eq. (B.5) is identical to that of Ref. [16] under the change of variable \(n \leftrightarrow \sigma - n\), i.e.

\[I(p, q, \sigma) = \sum_{n} (-1)^n \binom{p}{\sigma-n} \binom{q-p}{2n-\sigma} \quad (B.6)\]

For \(q < p\), using Eq. (41), one obtains

\[I(p, q, \sigma) = \sum_{n=\max(0, [\sigma-p+q)/2\}] \binom{p-q}{n} \binom{q}{\sigma-2n} \quad (B.7)\]

In so doing the computation effort for directly computing the individual sums can be cut in half as it can be seen from the limits of the sums. This integer sum can be calculated to a machine precision as long as each term does not exceed \(F\), the largest integer that can be expressed in floating point. (In double precision floating point for a 32-bit machine (i.e. 8-byte words), \(F\) occupies 15 decimal digits; while in extended (quadruple) precision (16-byte words), \(F\) occupies 32 decimal digits.)

Based on the recursion relation of binomial coefficients,

\[\binom{m}{n} = \binom{m-1}{n} + \binom{m-1}{n-1} \quad (B.8)\]

computation time is improved in the evaluation of \(I\) function. Using Eq. (B.7), one obtains

\[I(p, q, \sigma) = I(p-1, q-1, \sigma) - I(p-1, q-1, \sigma-2) \quad (B.9)\]

or

\[I(p, q, \sigma) = \frac{1}{4}[I(p-1, q-1, \sigma) - I(p-1, q-1, \sigma-2)] \quad (B.10)\]

Various implementations of the expressions of \(I(p, q, \sigma)\) have been tested and the results for the accuracy of double precision floating point numbers against extended (quadruple) precision ones are given in Fig. B.1.
The $S(p, q, \sigma)$ function in Eq. (34) as a sum of binomial divisions (case 1/1 in Ref. [16]) can be written as a sum of binomial products (case 2/0):

$$S(p, q, \sigma) = \sum_{n=0}^{\min(p, p+q-\sigma)} (-1)^n \binom{p}{n} \binom{p+q}{\sigma+n}^{-1} \binom{p+q-\sigma}{p+q}^{-1}$$

provided $q \geq \sigma$. Except for the condition, the advantage of the latter expression is that the function can be expressed as a division of two integers $S(p, q, \sigma)/T(p, q, \sigma)$ where $S$ is the sum and $1/T$ is the factor in front of the sum. Note that $T$ is the number of possible partitions $p+q$ in 3 spaces (the trinomial expansion coefficient); hence, it is also an integer.

Substituting $\binom{p+q-\sigma-n}{p-n}$ in Eq. (B.11) with its recursion relation, Eq. (B.8), leads to the recursion relation Eq. (44),

$$S(p, q, \sigma) = \frac{q-\sigma}{p+q} S(p, q-1, \sigma) + \frac{p}{p+q} S(p-1, q, \sigma).$$

Notice that the relation does not hold anymore for $q = \sigma$. An alternative recursion relation, i.e. Eq. (42) can be obtained by substituting $\binom{p}{j}$ in Eq. (34) with its recursion relation.

### Appendix C. Expressions for stretched-Racah coefficients and 9j-symbols

Many Racah coefficients and 9j-symbols used in this paper are of the stretched types. The stretched Racah coefficients (6j-symbols) used in (25) and (38) can be expressed

$$W(j_1 j_2 J j_3; j_1 + j_2 j_23) = \frac{(2j_1)!(2j_2)!(j_1 + j_2 + j_3 + J + 1)!}{(2j_1 + 2j_2 + 1)!(j_1 + j_2 - j_3 + J)!} \frac{(j_1 + j_2 - j_3 - J)!}{(j_1 + j - j_3)(j_1 - j_2 + j_3 + J)!} \frac{(j_1 + j_2 + j_3 - J)!}{(j_1 + j + j_2)!(j_1 + j + j_2 + j_3 - J)!} \frac{(-j_1 + j + j_3 + j_2)!(j_2 - j + j_3)!}{(-j_2 + j + j_3 + j_2)!(j_2 - j + j_3 + j_2)!} \frac{(j_2 + j - j_3)!}{(j_2 + j - j_3)!} \frac{(j_2 + j + j_3)!}{(j_2 + j + j_3)!} \frac{(j_2 + j + j_3)!}{(j_2 + j + j_3)!} \frac{1}{(j_2 + j + j_3)!}$$

(C.1)
after applying the classical symmetries.

The stretched 9j-symbols used in (18) are expressed [14]

$$\left\{ \begin{array}{c} j_1 \ j_2 \ j_12 \\ j_3 \ j_4 \ j_34 \\ j_1 + j_3 \ j_24 \ J \end{array} \right\} = \Delta(j_34, J, j_1 + j_3) \times \Delta(j_1, j_2, j_12) \Delta(j_3, j_4, j_34) \Delta(j_2, j_4, j_24) \Delta(j_12, j_34, J) \times \sqrt{\frac{(2j_1)!(2j_3)!(j_1 + j_2 - j_3)!(j_2 + j_4 - j_3)!}{(2j_1 + 2j_3 + 1)!} \frac{(j_1 + j_2 + j_3 + J)!}{(j_1 + j_2 + j_3 + J)!} \frac{(j_2 + j_4 + J + 1)!}{(j_2 + j_4 + J + 1)!} \frac{(j_4 + j_34 - j_3)!(j_12 + j_34 - J)!(j_1 + j_3 + j_24 + J + 1)!}{(j_4 + j_34 + J + 1)!} \frac{(-1)^m+\pi(2j_12 - m)!(2j_4 - n)!}{m!n!(j_12 + j_34 - J - m)!(j_2 + j_4 - j_24 - n)!} \frac{(J + j_34 - j_1 + j_24 + m + m)!}{(j_4 - j_12 - j_3 + J + m - n)!}.$$ \hspace{1cm} (C.2)
where
\[
\Delta(j_1, j_2, j_3) = \sqrt{\frac{(j_1 + j_2 - j_3)! (j_1 - j_2 + j_3)! (-j_1 + j_2 + j_3)!}{(j_1 + j_2 + j_3 + 1)!}}.
\] (C.3)

Appendix D. Input files

D.1. Input file for SU(2) bases

3 option
108 6 6 4 118 2 SU(3) irreps
4 4
n
n

D.2. Input file for SO(3) bases

1 option
108 6 6 4 118 2 SU(3) irreps
10 4 12
n
n

Appendix E. Output files

E.1. Output file for SU(2) bases

*** ------------------------------------------------ ***
*** Clebsch Gordan coefficient of SU(3) ***
*** in SU(2) bases ***
*** ------------------------------------------------ ***
((lm1 mu1) j1 I1; (lm2 mu2) j2 I2 || rh (lm3 mu3) j3 I3)

*** Choose the following options to calculate:
  1) specific CG
  2) all 3rd weights
  3) all 1st and 2nd weights
  4) all weights
***> option 3

Enter the irreps: lm1,mu1, lm2,mu2, lm3,mu3::

Enter: 2*j3,2*I3::

((lm1 mu1) j1I1t; (lm2 mu2) j2I2t || rh (lm3 mu3) j3I3t)
108 6 0 6 6 4 4 2 1 118 2 4 4 0.0011073
<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0 6</td>
<td>4 4</td>
<td>1</td>
<td>4 4</td>
<td>-0.0017086</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 6</td>
<td>4 6</td>
<td>1</td>
<td>4 4</td>
<td>0.0015660</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 6</td>
<td>4 8</td>
<td>1</td>
<td>4 4</td>
<td>-0.0008923</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 5</td>
<td>3 1</td>
<td>1</td>
<td>4 4</td>
<td>0.0068558</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 5</td>
<td>3 3</td>
<td>1</td>
<td>4 4</td>
<td>-0.0105787</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 5</td>
<td>3 5</td>
<td>1</td>
<td>4 4</td>
<td>0.0119951</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 5</td>
<td>3 7</td>
<td>1</td>
<td>4 4</td>
<td>-0.0091875</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 7</td>
<td>3 3</td>
<td>1</td>
<td>4 4</td>
<td>-0.0100445</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 7</td>
<td>3 5</td>
<td>1</td>
<td>4 4</td>
<td>0.0113894</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 7</td>
<td>3 7</td>
<td>1</td>
<td>4 4</td>
<td>-0.0070097</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 4</td>
<td>2 2</td>
<td>1</td>
<td>4 4</td>
<td>-0.0211309</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 4</td>
<td>2 4</td>
<td>1</td>
<td>4 4</td>
<td>0.0415002</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 4</td>
<td>2 6</td>
<td>1</td>
<td>4 4</td>
<td>-0.0471887</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 6</td>
<td>2 2</td>
<td>1</td>
<td>4 4</td>
<td>-0.0581731</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 6</td>
<td>2 4</td>
<td>1</td>
<td>4 4</td>
<td>0.0712472</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 6</td>
<td>2 6</td>
<td>1</td>
<td>4 4</td>
<td>-0.0558115</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 8</td>
<td>2 4</td>
<td>1</td>
<td>4 4</td>
<td>0.0449907</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 8</td>
<td>2 6</td>
<td>1</td>
<td>4 4</td>
<td>-0.0339130</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 3</td>
<td>1 3</td>
<td>1</td>
<td>4 4</td>
<td>0.0635334</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 3</td>
<td>1 5</td>
<td>1</td>
<td>4 4</td>
<td>-0.1331113</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 5</td>
<td>1 3</td>
<td>1</td>
<td>4 4</td>
<td>0.1626470</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 5</td>
<td>1 5</td>
<td>1</td>
<td>4 4</td>
<td>-0.2102149</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 7</td>
<td>1 3</td>
<td>1</td>
<td>4 4</td>
<td>0.2491470</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 7</td>
<td>1 5</td>
<td>1</td>
<td>4 4</td>
<td>-0.2146751</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
C. Bahri et al. / Computer Physics Communications 159 (2004) 121–143

3 9 1 5 1 4 4 -0.1040981
4 2 0 4 1 4 4 -0.1850649
4 4 0 4 1 4 4 -0.3762170
4 6 0 4 1 4 4 -0.5305188
4 8 0 4 1 4 4 -0.5581637

More j,I’s?
Total: 30 nodes.

More (lm,mu)’s?

E.2. Output file for SO(3) bases

*** ------------------------------------------------ ***
*** Clebsch Gordan coefficient of SU(3) ***
*** in SO(3) bases ***
*** ------------------------------------------------ ***
((lm1 mu1)al1 L1; (lm2 mu2)al2 L2 || rh (lm3 mu3)al3 L3)

*** Choose the following options to calculate:
1) specific CG
2) all 3rd weights
3) all 1st and 2nd weights
4) all weights

***> option 1
Enter: lm1,mu1, lm2,mu2, lm3,mu3::

Enter: L1, L2, L3::

((lm1 mu1)al1 L1; (lm2 mu2)al2 L2 || rh (lm3 mu3)al3 L3)
108 6 1 10 6 4 1 4 1 118 2 1 12 -0.1235100
1 10 1 4 1 2 12 -0.0036753
1 10 2 4 1 1 12 -0.0113518
1 10 2 4 1 2 12 -0.0273097
1 10 3 4 1 1 12 -0.0001060
1 10 3 4 1 2 12 -0.0005038
\begin{verbatim}
2 10  1 4  1  1 12  -0.0159115
2 10  1 4  1  2 12  -0.0766655
2 10  2 4  1  1 12  -0.1175689
2 10  2 4  1  2 12  -0.0052723
2 10  3 4  1  1 12  -0.0030052
2 10  3 4  1  2 12  0.0383229
3 10  1 4  1  1 12  0.0001144
3 10  1 4  1  2 12  -0.0163565
3 10  2 4  1  1 12  0.0031166
3 10  2 4  1  2 12  -0.1222128
3 10  3 4  1  1 12  0.0818798
3 10  3 4  1  2 12  -0.0034110
4 10  1 4  1  1 12  0.0000000
4 10  1 4  1  2 12  0.0000665
4 10  2 4  1  1 12  0.0000001
4 10  2 4  1  2 12  0.0020672
4 10  3 4  1  1 12  -0.0000014
4 10  3 4  1  2 12  0.0569243
\end{verbatim}

Total: 24 nodes.

More L’s?

More (lm,\mu) s?

References

    J.D. Louck, Am. J. Phys. 38 (1970) 3;