ELSEVIER

Computer Physics Communications 111 (1998) 167–184

Computer Physics Communications

Maple procedures for the coupling of angular momenta II. Sum rule evaluation

S. Fritzsche [1], S. Varga, D. Geschke, B. Fricke

Fachbereich Physik, Universität Kassel, Heinrich-Plett-Str. 40, D-34132 Kassel, Germany

Received 20 November 1997

Abstract

In a previous paper (S. Fritzsche, Comput. Phys. Commun. 103 (1997) 51), we defined data structures to deal with typical expressions from Racah algebra within the framework of Maple. Such expressions arise very frequently in various fields, for instance, by treating composite wave functions and tensor operators in many-particle physics. Often, these Racah expressions are written in terms of Clebsch–Gordan coefficients and Wigner n–j symbols. Our previous set of Maple procedures mainly concerned numerical computations on such symbols, the simplification by special values as well as the use of recursion relations.

The full elegance of applying Racah algebra techniques in daily research work is, however, only revealed by the analytic simplification of more complex expressions. In practise, this even requires the major effort in dealing with these techniques. Its success closely depends on the knowledge of sum rules which typically include a number of dummy summation indices. The application of these sum rules is a rather straightforward task but may become very tedious for more difficult expressions due to the large number of symmetries of the Clebsch–Gordan coefficients and Wigner n–j symbols. We therefore extended the Racah program to facilitate sum rule evaluations in the given framework. A set of new and revised procedures now supports the evaluation of Racah algebra expressions by applying the orthogonality properties of the Wigner symbols and a variety of sum rules. More than 40 sum rules known from the literature and involving products of up to six Wigner n–j symbols have been implemented and are available for interactive use. The applicability of this new tool will be demonstrated by three examples from many-particle physics. © 1998 Elsevier Science B.V.

PACS: 3.65F; 2.90+p

Keywords: Angular momentum; Racah algebra techniques; Sum rule evaluation; Spherical tensor operators; Wigner n–j symbols

PROGRAM SUMMARY

Title of program: Racah

Catalogue identifier: ADHW

Program obtainable from: CPC Program Library, Queen's University of Belfast, N. Ireland

Licensing provisions: none

Computer for which the program is designed and others on which it is operable:
Computers: All computers with a license of the computer algebra

[1] E-mail: fritzsch@physik.uni-kassel.de

PII S0010-4655(98)00021-6

package Maple [1];*Installations:* University of Kassel (Germany)

Operating systems under which the program has been tested: AIX 3.2.5, Linux

Program language used: Maple V, Release 3 and 4

Memory required to execute with typical data: 4 MB

No. of lines in distributed program, etc.: About 4900 lines in addition to the source code of the Racah program as described previously in Ref. [2]

No. of bytes in distributed program, including test data, etc.: 467242

Distribution format: ASCII

Keywords: angular momentum, Racah algebra techniques, sum rule evaluation, spherical tensor operators, Wigner n–j symbols

Nature of the physical problem
Computer algebra (CA) is used to evaluate and to simplify typical expressions from Racah algebra which may also include the summation over dummy quantum numbers. A large variety of sum rules is implemented in a set of Maple commands for interactive use.

Method of solution
In a recent paper [2], we defined proper data structures to deal efficiently with expressions from Racah algebra and to enable the numerical computation of such expressions. The present extension of the program now aims at simplifying typical Racah expressions which may also include the summation over dummy indices. A simplification is attempted by the successive analyses of the various parts of a given Racah expression and by comparison with a set of sum rules in their standard form as found in the literature. The orthogonality relations which are known for the Wigner n–j symbols are also implemented; internally, however, these relations are treated as *special* sum rules. All equivalent symmetric forms of a given Racah expression are taken into account during the evaluation. More than 40 different sum rules are known to the package which will cover many applications in different fields.

Restrictions onto the complexity of the problem
The set of sum rules, which has been implemented in the Racah program, mainly refers to the monograph by Varshalovich et al. [3] on the theory of angular momentum. These sum rules either include a single Wigner n–j symbol or products with a different number of such symbols. In general, the complexity of Racah expressions increase as more Wigner symbols are involved in the product terms. Though we incorporated a large number of sum rules for the Wigner 3–j and 6–j symbols, only a selected set of those rules involving 9–j symbols have been implemented in the present version. So far, we have also considered only a smaller number of sum rules involving products of more than four Wigner n–j symbols of different kinds as well as those including more than a triple summation over dummy quantum numbers. The most complex sum rule currently involves the product of six 3–j symbols and a nine-fold summation. The application of this latter sum rule, however, does often not work very efficiently with respect to time.

The success in simplifying Racah algebra expressions critically depends on the fact that all equivalent symmetric forms of the expression are recognized internally. Thereby, the overall symmetry of a Racah expression is directly related to the symmetries of all the Wigner n–j symbols which are involved in the expression. Apart from the classical symmetries of the Wigner symbols, there is an extended range of symmetries due to Regge [4]; these symmetries, however, are of minor importance for most practical applications. Even though, in principle, the Racah package enables one to apply the full range of extended symmetries, limitations in computer time will often restrict the usage of the program to the classical ones.

Unusual features of the program
All commands of the Racah package are available for interactive work. As explained in Ref. [2], the program is based on data structures which are suitable for almost any complexity of Racah algebra expressions. More enhanced expressions are built up from simpler data structures. The simplification of any valid Racah expression can be attempted just by typing the command Racah_evaluate() at Maple's prompt. This will test all different rules which are known to the package. However, if one knows the *structure* of the sum rule in advance, i.e. the number and type of the Wigner n–j symbols involved, these rules can also directly be invoked by individual commands. This usually results in a faster evaluation, in particular if more complex expressions need to be simplified. In Appendix A, we summarize all new commands at user's level for quick reference (i.e. those commands which had not been described yet in Ref. [2]).

Typical running time
All examples of the long write-up require about 3 minutes on an IBM workstation.

References

[1] Maple is a registered trademark of Waterloo Maple Inc.
[2] S. Fritzsche, Comput. Phys. Commun. 103 (1997) 51.
[3] D.A. Varshalovich, A.N. Moskalev, V.K. Khersonskii, Quantum Theory of Angular Momentum (World Scientific, Singapore, 1988).
[4] T. Regge, Nuovo Cimento 10 (1958) 544.

LONG WRITE-UP

1. Introduction

The study of open-shell atoms first raised the question how the rotational symmetry of closed systems, i.e. the conservation of angular momentum, can efficiently be exploited in understanding many-particle systems. After the pioneering work of Wigner in the late thirties, Racah [1] developed a powerful machinery to deal with such systems. This machinery is known as *Racah algebra* today, and it became one of the fundamental concepts in the (quantitative) treatment of many-particle systems.

In studying such systems one often considers composite wave functions and composite tensor operators as they arise from many-body wave functions and operators which are symmetric in all participating particles. The aim of applying Racah algebra techniques is then to reduce the matrix elements of one- and two-body spherical tensor operators for many-particle wave functions. Typical examples are the tensor product of two different operators from either a single subsystem or the interaction among two individual subsystems. Thereby, Racah algebra is concerned with the analytic evaluation of the angular portion of matrix elements and, in particular, with the complications that arise from the coupling of various angular momenta. For this, these techniques may provide notational and computational simplifications of great elegance which are practically indispensable for describing complex many-particle systems. By contrast, the (often numerical) evaluation of the radial integration is straightforward and remains unaffected by the coupling of the angular momenta.

Even though the mathematical background of angular momentum theory is now well understood, the treatment and simplification of typical expansions as they arise in Racah algebra is often a very laborious task. Many standard techniques of Racah algebra, namely, result in rather complex expressions involving a multiple summation over products of Clebsch–Gordan coefficients and/or Wigner n–j symbols. Usually, these expressions can be simplified by a given set of sum rules; however, a lot of experience and effort is required to find the proper rule and to rewrite the Racah expression in that form that one can recognize possible simplifications. Apart from this sum rule evaluation, a variety of graphical methods have also been developed during the last decades [2–4] which may be very helpful in analytic work but, again, may become so elaborate that alternative help will be highly desirable.

In a previous paper by us ([5], hereafter referred to as paper I), we defined proper data structures to deal with Racah algebra expressions within the framework of Maple. A set of procedures were developed for the numerical computation of single Wigner n–j symbols and for even more complex expressions. We also introduced the term *Racah expression* with the intention of providing an (internal) data structure which makes an automatic evaluation and simplification of typical expressions possible. Particular attention was directed to defining structures which are flexible and powerful enough for most practical applications. A numerical evaluation of a Racah expression, however, can be carried out only if, in the Racah expression, all quantum numbers, apart from dummy summation variables, have *numerical* values. By contrast, the main strength of using Racah algebra techniques is that such expressions can often be simplified considerably by algebraic transformations. Therefore, we will now present an extension of our Maple program which supports the simplification by orthogonality properties and sum rules. A set of frequently applied sum rules has been implemented in the package as we will indicate below.

Useful relations and sum rules in Racah algebra are usually expressed in terms of the Wigner n–j symbols which possess a very high symmetry. During the last decades, much attention in the literature has been devoted to deriving and proving certain sum rules among these and other related symbols. Here, we are not concerned with such derivations. Instead, we will start from a set of known relations and present a computer algebra tool which facilitates handling of such expressions. Emphasize was put on developing an interactive tool for a rather wide range of applications. Computer algebra may partially replace graphical methods in the future since it allows to treat phases and weight factors in a much simpler way.

In the following two sections, we briefly outline the way of handling sum rule evaluation for Racah expressions. We will also give reasons for the benefit of developing such a computer algebra tool. However, we will not present these rules in their explicit form here but will refer the reader for all mathematical details to the literature. Reference to the monograph by Varshalovich et al. [6] is given in two tables, one table for sum rules involving the Wigner 3–j symbols and a second one for those rules involving only Wigner 6–j and 9–j symbols. We also explain how the program will be distributed. The power of sum rule evaluation is later demonstrated in Section 4 by three examples from many-particle physics. They will concern the recoupling of three angular momenta as well as the simplification of many-particle matrix elements and of a single Feynman–Goldstone diagram.

2. Sum rule evaluation

Many sum rules of the Racah algebra can be related to a subsequent application of the known coupling rule for two angular momenta (cf. paper I, Eq. (1)) or to the recoupling of three or more angular momenta. These rules can therefore often be expressed in terms of the Clebsch–Gordan coefficients, or more generally, in terms of overlap integrals of angular momentum eigenstates which belong to different sets of commuting operators. Even more frequently, however, these rules are represented in terms of Wigner n–j symbols. The properties of these symbols have been discussed previously; they may also be found in most standard texts about the coupling of angular momenta. These symbols are the basic quantities of the Racah program and will preferably be used in the text below. An example of a typical sum rule occurs for the recoupling of three angular momenta (as we will show below) which results in a complete contraction over all magnetic quantum numbers for products of four Wigner 3–j symbols.

Usually, Racah algebra techniques are mainly concerned with basis functions which have not been antisymmetrized with respect to the interchange of particle coordinates. Thus, the standard formalism is to be applied for matrix elements of wave functions that are simple products of single-particle functions whose angular momenta are coupled together according to some given coupling scheme. Antisymmetrization is sometimes considered within a subshell of equivalent particles but this should be done by making use of the cfp-coefficients, i.e. of coefficients of fractional parentage [7].

Useful compilations about relations in Racah algebra were given by Rotenberg and co-workers [8] and later, in much more comprehensive form, in the monograph by Varshalovich et al. [6]. We will mainly refer to this monograph in the course of this text. To simplify complex expressions from Racah algebra one usually starts from these compilations today. Here, we would like to remind the reader that, in our notation, a Racah expression might include products of an arbitrary number of Wigner n–j symbols ($n = 3, 6, 9$) of different kinds as well as of Kronecker and triangular $\delta(j_1, j_2 j_3)$ deltas. This has been reflected by the internal structure of such expressions in the Racah program as shown in paper I, Appendix B.

The major difficulty in practical work is to recognize the equivalence of a certain part of a Racah algebra expression, for instance with one side of a given sum rule, so that the expression as a whole could be rewritten in a simpler form. This requires a careful analysis of the full Racah expression. Since such expressions will usually include a summation over dummy indices, these summation indices not only have to be placed correctly in the Wigner n–j symbols but they must also contribute to a correct phase and weight of the overall expression. Moreover, a summation index may not occur in other Wigner symbols of the Racah expression which are not part of the considered rule.

Another difficulty for the simplification of general Racah expressions arises from the large number of symmetric forms of the total expression. The overall symmetry of a Racah expression as a whole namely is the direct result of the symmetries of the Wigner n–j symbols. Apart from the so-called classical symmetries of the Wigner n–j symbols, there are additional symmetries known due to Regge for the 3–j and 6–j symbols. The number of symmetric forms of the Wigner symbols have been listed in paper I, Table 1. These symmetries need to be exploited in order to simplify general Racah algebra expressions. In the Racah program, however, the distinction between the classical symmetries and those due to Regge is kept since the classical ones are by far more important for most practical purposes. To remind the reader of our notation in paper I, we consider the classical symmetries to be a subset of the Regge symmetries.

To simplify a given Racah expression, in practice one has to start with a certain part of the expression and then try to identify the equivalence of this part with one side of any sum rule by cycling through all symmetries of the Wigner n–j symbols. Once this equivalence has been proven, this part of the Racah expression can be replaced by the corresponding simpler structure (i.e., by the other side of this rule). Note that we characterize the *simplification of an expression* in the Racah program by reducing the number of summation indices and/or the number of Wigner n–j symbols. To this end computer algebra is well suited, in particular when it comes to cycling through all symmetric forms of a Racah expression and to identifying the algebraic equivalent parts. The Racah program basically serves these tasks and replaces equivalent parts of a given Racah expression.

Using graphical methods [3,4,6], each sum rule for the Wigner n–j symbols can be illustrated by a diagram which shows the coupling of the various angular momenta in this sum. Generally, a different number of summation variables corresponds to a different number of external lines in the diagrammatic representation. According to the two types of the angular momentum quantum numbers, two different kinds of summations appear:

(i) A summation over an angular momentum quantum number j. This will run over all possible integer or half-integer values which are consistent with one or more triangular conditions.

(ii) A summation over some magnetic quantum number m_j, i.e. some projection of an angular momentum onto the quantization axis. Such an index runs over all correspondingly allowed values $-j, -j+1, \dots, +j$.

In a Racah expression as defined in paper I, one could specify for a given summation index (in principle) an special *index range equation*; such ranges of indices other than indicated above, however, do not appear for any index in the presently implemented sum rules and are thus not well supported.

In the literature, compilations of sum rules are often divided due to these different kinds of summation variables into two parts. One part includes those products involving Wigner 3–j symbols and a second one products involving only 6–j and/or 9–j symbols. We will here follow this line and will give a brief account of the sum rules which have been implemented in the Racah program in the next two subsections.

2.1. Sum rules involving products of 3–j symbols

Varshalovich et al. [6] list a total of about thirty sum rules involving Wigner 3–j symbols. These rules are most conveniently represented in terms of the Wigner symbols rather than in terms of the Clebsch–Gordan coefficients because the 3–j symbols fulfill simpler symmetry properties without that additional weights are occurring. For a selected number of these sum rules, Varshalovich et al. also display the corresponding representation in terms of Clebsch–Gordan coefficients. Within the Racah program we need not really distinguish between these two representations. Although the internal setup of the Racah program always uses the Wigner n–j symbols, Racah expressions which are written in terms of Clebsch–Gordan coefficients can be simplified just as easily. This can be seen from the examples in Section 4. All Clebsch–Gordan coefficients which appear in a Racah expression are automatically transformed in a Wigner 3–j symbol taking into account the proper phase and weight factors.

One example of a sum rule involving the products of four Wigner 3–j symbols arise from the recoupling of three angular momenta,

$$\sum_{m_1 m_2 n_1 n_2 n_3} (-1)^S \, [j_3] \begin{pmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{pmatrix} \begin{pmatrix} j_1 & l_2 & l_3 \\ m_1 & n_2 & -n_3 \end{pmatrix} \begin{pmatrix} l_1 & j_2 & l_3 \\ -n_1 & m_2 & n_3 \end{pmatrix} \begin{pmatrix} l_1 & l_2 & j_3 \\ n_1 & -n_2 & m_3 \end{pmatrix}$$

Table 1
List of sum rules involving Wigner 3–j symbols as implemented in the Racah package

Sum rules involving	Comment	Ref. [6]
one 3–j symbol	One such rule is included with a single summation over a magnetic quantum number.	Eq. (12.1:2)
two 3–j symbols	Two orthogonality relations for the Wigner 3–j symbols are included in the Racah program each having a double summation.	Eqs. (12.1:3–4)
three 3–j symbols	One such rule is included with a triple summation over magnetic quantum numbers.	Eq. (12.1:6)
two 3–j symbols and one 6–j symbol	One such rule having a double summation over one angular momentum and one magnetic quantum number is included. This rule results in a single sum involving the product of two Wigner 3–j symbols with a single summation over a magnetic index.	Eq. (12.1:5)
four 3–j symbols	Two of these rules are included representing a single 6–j symbol in terms of four 3–j symbols. They express both a complete contraction over the magnetic quantum numbers and have either a six-fold or a five-fold summation (cf. Eq. (1)), respectively. Two further sum rules known for products of four Wigner 3–j symbols have not yet been incorporated in the Racah package.	Eq. (12.1:8)
six 3–j symbols	One such sum rule representing a single 9–j symbol in terms of six 3–j symbols is included. This again is achieved by a complete contraction over all magnetic quantum numbers, i.e. by a nine-fold summation. Sometimes, however, a corresponding Racah expression is simplified in the program in a two-step evaluation to a triple sum involving Kronecker deltas which need to be evaluated explicitly.	Eq. (10.2:17)

A short comment indicates the number of sum rules of this type as well as the number of dummy summation indices involved in these rules. For all further mathematical details, reference is given to the monograph by Varshalovich et al. [6], Chapter 12; for instance, a notation like (12.1:3–4) refers to Eqs. (3) and (4) in Chapter 12, Section 1 of this monograph.

$$= \begin{Bmatrix} j_1 & j_2 & j_3 \\ l_1 & l_2 & l_3 \end{Bmatrix} \tag{1}$$

with $S = l_1 + l_2 + l_3 + n_1 + n_2 + n_3$. In this particular situation, a sum over all magnetic quantum numbers apart from one (m_3) for a product of four Wigner 3–j symbols appear. As seen from the right-hand side of (1), this term defines an invariant quantity with respect to a rotation of the coordinates, i.e. a quantity which is independent of the special choice of the quantization axis.

Table 1 shows a list of sum rules involving Wigner 3–j symbols which have been implemented in the Racah program. For all further mathematical details, however, we refer (in the third column of this table) for each sum rule separately to the book of Varshalovich et al.. Though we do not repeat all the formulas, the different entries in this table reflect – at least to a certain extent – the structure of the source code of the Racah program as well as the sequence in which the different rules are internally "probed". In almost all these sum rules each magnetic summation index m_j usually appears with opposite signs in two of the Wigner 3–j symbols. In addition, these sum rules often include the phase factors $(-1)^{j-m_j}$ to ensure that the overall expression is invariant under coordinate rotations [6].

2.2. *Sum rules involving products of only 6–j and/or 9–j symbols*

Sums involving only products of 6–j and/or 9–j symbols typically represent quantities which are invariant under rotations. In this case, summations appear over different angular momenta. From the sum rules given by Varshalovich et al., only a selected set of rules involving Wigner 9–j symbols have currently been incorporated in the Racah program. Reference to these rules is given in Table 2. The Wigner 9–j symbols have particular importance for complex systems as they represent the full contractions over products of six Wigner 3–j symbols.

Table 2
List of sum rules involving only Wigner 6–j and/or 9–j symbols as implemented in the Racah package; the same notation as in Table 1 is used

Sum rules involving	Comment	Ref. [6]
one 6–j symbol	Two such sum rules with single summations are included.	Eqs. (12.2:3–4)
one 9–j symbol	Two such rules are included having a single summation each. A further sum rule due to Rotenberg et al. [8], Eq. (3.8), and involving a triple summation has not yet been incorporated in the Racah program.	Eqs. (12.2:5–6)
two 6–j symbols	One orthogonality relation for two Wigner 6–j symbols is included. Furthermore, we incorporated one sum rule with a single summation and one rule with a triple summation.	Eqs. (12.2:7–8) and (12.2:15)
one 6–j symbol and one 9–j symbol	Two of these sum rules are included having a single summation each. Both rules evaluate themselves to a simple product of two 6–j symbols.	Eqs. (12.2:9–10)
two 9–j symbols	One orthogonality relation for two Wigner 9–j symbols is included. Furthermore, a rather similar rule having an additional phase and a double summation has also been incorporated in the Racah package.	Eqs. (12.2:13–14)
three 6–j symbols	Two such sum rules are included having a single summation. They give rise either to a single 9–j symbol or to a product of two 6–j symbols.	Eqs. (12.2:18–19)
two 6–j symbols and one 9–j symbol	Three of such sum rules having a double summation are included and one sum rule having a triple summation. By using Racah_evaluate( ), however, a corresponding Racah expression is often simplified in a two-step evaluation applying sum rules for one 6–j and one 9–j symbol, and followed by a sum rule for three 6–j symbols.	Eqs. (12.2:24–26) and (12.2:28)
one 6–j symbol and two 9–j symbols	Three of these sum rules are included, one having double summation and two having a triple summation. By using Racah_evaluate( ), however, a corresponding Racah expression is simplified in a two-step evaluation using sum rules for two 9–j symbols, and followed by a sum rule for two 6–j and one 9–j symbols.	Eqs. (12.2:27) and (12.2:29–30)
three 9–j symbols	One such rule having a triple summation is included.	Eq. (12.2:31)
four 6–j symbols	Five of those sum rules can be applied; two of them have a single summation, two a double summation, and one rule even a triple summation. Only the two rules with a single summation needed to be implemented explicitly. The other three sum rules can be applied using different other sum rules in a two step evaluation.	Eqs. (12.2:32–35) and (12.2:38)
three 6–j symbols and one 9–j symbol	One such rule with a double summation is included and two of them with a triple summation. The latter two, however, can be simplified using alternative sum rules in a two-step evaluation.	Eqs. (12.2:36) and (12.2:39–40)
two 6–j symbols and two 9–j symbols	Four of those sum rules have been included. One has a double summation and three of them have a triple summation.	Eqs. (12.2:41–43) and (12.2:37)
one 6–j symbol and three 9–j symbols	One such rule having a triple summation over dummy angular momenta is included. Evaluation by this rule works well if the procedure Racah_usesumrulesforonew6jthreew9j( ) is invoked independently. By using Racah_evaluate( ), instead, it may need a long time before the rule is really applied[a].	Eq. (12.2:44)

[a] In general, it is rather difficult to estimate the execution time for the evaluation by different rules. An evaluation is attempted by the program if the type and number of Wigner n–j symbols as well as the number of formal summation indices are appropriate. Sum rules with a simpler structure are applied first, i.e. they are applied in the order listed in these tables.

For instance, these symbols form the transformation matrix which has to be applied when changing from a LS-coupling basis into a jj-coupling basis. By contrast, Wigner 9–j symbols are somewhat less important if they occur within sum rules including different kinds of Wigner n–j symbols. To test whether a certain sum rule has been implemented in the Racah program it is usually enough to enter the more complex side of the sum rule and to invoke the command Racah_evaluate( ). This will be shown in Section 4.

3. Extended setup of the Racah package

The general setup of the Racah program need not be explained once more since it has been described in paper I. There, we outlined the application of the program within the framework of Maple and also showed some simpler examples for the syntax of the program. We saw that all commands form a hierarchical structure where each procedure by itself could be used as individual command for interactive work as well as a basic element to built up procedures at some higher level of the hierarchy. The input and output data were handled by the procedures as logical objects which could have, in principle, an arbitrary complex data structure.

For the current extension of the program which enables the analytic evaluation due to a set of internally known sum rules, we basically need to describe only the command Racah_evaluate( ). This command attempts simplifying a specified Racah expression in a series of individual steps by invoking a large subset of procedures for different sum rules. In total, however, we added more than 20 new procedures to the Racah program so that the whole package now contains about 80 individual commands.

The command Racah_evaluate( ) is described in Appendix A; this expands our description in paper I. Again, we follow the style of the Maple handbook by Redfern [9]. In this appendix, we also list the names of all individual subprocedures which can be invoked separately with the same list of arguments as Racah_evaluate( ) if the type of the required sum rule is known in advance. The names of these subprocedures are typically rather long but they obviously reflect the type of the implemented rules. The prefix Racah_ to the names of all procedures of the Racah program shows clearly that they are not part of the inherent functionality of Maple V.

In the program, the sum rules are grouped together in accordance with the number of Wigner n–j symbols which are involved in the product terms. For an equal number of such symbols, we further distinguish the type of the Wigner symbols as seen in Table A.1 in the appendix. The simplification of Wigner 3–j symbols, thereby, has a higher priority than the simplification of Wigner 6–j symbols and these again a higher priority than the Wigner 9–j symbols. All sum rules with an equal number and type of Wigner n–j symbols are implemented within *one and the same* procedure of the Racah package even if a different number of dummy summation indices appear.

The Racah program will now be distributed as the ASCII file `racah2`. This file also contains the whole source code of paper I in which we fixed a few minor bugs. As in the previous version, the source file `racah2` lists all procedures in alphabetic order. To use these procedures as interactive commands, it is most convenient to load the whole program just by entering

```
> read racah2;
```

at the beginning of each session. As in our previous version we did not define global variables which would have to be kept in mind.

We will finally remind the user of the Racah program that all variables and expressions which appear like angular momentum quantum numbers must represent either integer or half-integer values and must fulfill proper coupling conditions. This word of caution is required since, in many instances, the validity of some expression cannot easily be checked during the process of evaluation. Failures in the coupling conditions, however, may lead to syntax errors in some later step of the evaluation or to incorrect results.

By using the new feature of sum rule evaluation, we are now able to demonstrate the power of this CA tool to Racah's algebra in solving a few simpler problems from many-particle physics.

4. Test examples from many-particle physics

We show the usefulness of the program by three examples which could be taken as *standard* problems in teaching many-particle quantum mechanics. Indeed, these or similar examples are found in textbooks on both atomic and nuclear structure theories. The evaluation of a single Feynman–Goldstone diagram in our third example will demonstrate further that such a computer algebra tool is useful not only for simple problems but also for active research where one has to deal with Racah algebra techniques. In the following we assume that the Racah program has been loaded into an interactive Maple session.

Let us first consider a quantum system described by three angular momentum operators $\boldsymbol{j}_1$, $\boldsymbol{j}_2$, $\boldsymbol{j}_3$, and the corresponding z-components j_{1z}, j_{2z}, and j_{3z}, respectively. Then, the six commuting operators $\{j_1^2, j_{1z}, j_2^2, j_{2z}, j_3^2, j_{3z}\}$ of this system have the product vectors $|j_1m_1, j_2m_2, j_3m_3\rangle \equiv |j_1m_1\rangle\,|j_2m_2\rangle\,|j_3m_3\rangle$ as eigenvectors. For the composite system we can furthermore define the total angular momentum operators

$$\boldsymbol{J} = \boldsymbol{j}_1 + \boldsymbol{j}_2 + \boldsymbol{j}_3\,, \qquad J_z = j_{1z} + j_{2z} + j_{3z}\,.$$

Using these and similar operators for just two angular momenta, there are three other sets of commuting operators where the three one-particle angular momenta are coupled together. One set, for example, is given by $\{j_1^2, j_2^2, j_3^2, J_{12}^2, J^2, J_z\}$ with the corresponding eigenvectors $|(j_1j_2)J_{12}j_3; JM\rangle$. In this notation, we omit the particle indices and assume the standard convention for carrying out Racah algebra, i.e. that the quantum state for particle 1 is coupled with the quantum state of particle 2, and this composite state with the one of particle 3 in exactly that order in which we read from left to right.

The two types of eigenvectors are often denoted as the "uncoupled" and "coupled" eigenvectors of the system and are related to each other via an expansion in terms of the Clebsch–Gordan coefficients,

$$\begin{aligned}
|(j_1j_2)J_{12}j_3; JM\rangle &= \sum_{M_{12}m_3} |(j_1j_2)J_{12}M_{12}, j_3m_3\rangle\,\langle (j_1j_2)J_{12}M_{12}j_3m_3\,|\,(j_1j_2)J_{12}j_3; JM\rangle \\
&= \sum_{m_1m_2M_{12}m_3} |j_1m_1, j_2m_2, j_3m_3\rangle\,\langle j_1m_1j_2m_2\,|\,(j_1j_2)J_{12}M_{12}\rangle \\
&\quad\times\,\langle (j_1j_2)J_{12}M_{12}j_3m_3\,|\,(j_1j_2)J_{12}j_3; JM\rangle\,,
\end{aligned} \tag{2}$$

where we have used the coupling of two angular momenta (cf. paper I, Eq. (1)) in two successive steps. The latter "overlap" matrix element in (2) represents a Clebsch–Gordan coefficient which does not depend on the angular momentum quantum numbers j_1 and j_2; these quantum numbers will be therefore omitted below. The expansion (2) can also be written in terms of the Wigner 3–j symbols as internally dealt with by the Racah program. But we need not rewrite here this expression explicitly since the program recognizes both Wigner 3–j symbols and Clebsch–Gordan coefficients. Note that for the relative phase between these quantities we shall apply the Condon–Shortley phase convention [10].

Two other sets of "coupled" eigenvectors are $|j_1(j_2j_3)J_{23}; JM\rangle$ and $|(j_1[j_2]j_3)J_{13}; JM\rangle$. The transformation between any two set of these eigenvectors can be constructed explicitly; this gives rise to the so-called *recoupling coefficients* [11]. For the two set of eigenvectors $\{|(j_1j_2)J_{12}j_3; JM\rangle\}$ and $\{|j_1(j_2j_3)J_{23}; JM\rangle\}$ this transformation is given by

$$|j_1(j_2j_3)J_{23}; JM\rangle = \sum_{J_{12}} |(j_1j_2)J_{12}j_3; JM\rangle\,\langle (j_1j_2)J_{12}j_3; JM\,|\,j_1(j_2j_3)J_{23}; JM\rangle\,. \tag{3}$$

For symmetry reasons, one may find simple arguments that the recoupling coefficients $\langle (j_1j_2)J_{12}j_3; J\,|\,j_1(j_2j_3)J_{23}; J\rangle = \langle (j_1j_2)J_{12}j_3; JM\,|\,j_1(j_2j_3)J_{23}; JM\rangle$ should not depend on the magnetic quantum number M. A typical textbook problem therefore is showing this independence on M explicitly for the transformation between states of three angular momenta with different coupling order.

This can simply be achieved by the Racah program. We first use the expansion (2) for the eigenvectors $|(j_1j_2)J_{12}j_3; JM\rangle$ and a corresponding expansion for $|j_1(j_2j_3)J_{23}; JM\rangle$, i.e. for the left and right side of the recoupling coefficient in Eq. (3). Then, to enter the right-hand side, we type the following lines at Maple's prompt:

```
> CGright1 := Racah_set(ClebschGordan,j2,m2,j3,m3,J23,M23):
> CGright2 := Racah_set(ClebschGordan,j1,m1,J23,M23,J,M):
> wright := Racah_add(Racahexpression,CGright1,CGright2):
> wright := Racah_add(sum,{m1,m2,m3,M23},wright):
```

As in paper I, we terminate these lines with a colon to avoid the reply of all the underlying list structures by Maple. The eigenstates of the three subsystems $\{|j_1m_1\rangle\}$, $\{|j_2m_2\rangle\}$, and $\{|j_3m_3\rangle\}$ are assumed to be orthonormal, i.e. the matrix elements between the uncoupled states are diagonal in all the j and m quantum numbers. In particular, we have $\langle j_1m_1, j_2m_2, j_3m_3 \mid j_1m_{1'}, j_2m_{2'}, j_3m_{3'}\rangle = \delta_{m_1m_{1'}}\delta_{m_2m_{2'}}\delta_{m_3m_{3'}}$, and can thus use the same summation variables m_1, m_2, and m_3 for the expansion of the eigenvectors on both sides of the recoupling coefficient in (3). Therefore, the left-hand side is entered by

```
> CGleft1 := Racah_set(ClebschGordan,J12,M12,j3,m3,J,M):
> CGleft2 := Racah_set(ClebschGordan,j1,m1,j2,m2,J12,M12):
> wleft := Racah_add(Racahexpression,CGleft1,CGleft2):
> wleft := Racah_add(sum,{m1,m2,m3,M12},wleft):
```

The full recoupling coefficient $\langle (j_1j_2)J_{12}j_3; JM \mid j_1(j_2j_3)J_{23}; JM\rangle$ is now simply the *product* of these two expansions

```
> rcc := Racah_add(Racahexpression,wleft,wright):
```

but before we carry out the evaluation of this expression we would first like to print it in terms of Wigner 3–j symbols,

```
> rcc := Racah_print(rcc):
--->
                          SUM{M23,m1,m2,m3,M12}

            (-J12 + 2 j3 - 2 M - 2 j1 - M12 - M23 + J23)
        (-1)

                                    1/2            1/2
             (2 J + 1) (2 J12 + 1)     (2 J23 + 1)

                          w3j(J12,j3,J,M12,m3,-M)

                         w3j(j1,j2,J12,m1,m2,-M12)

                         w3j(j2,j3,J23,m2,m3,-M23)

                          w3j(j1,J23,J,m1,M23,-M)
```

In order to show the independence on the M quantum number by means of sum rules, we must *evaluate* this Racah expression applying sum rules involving products of three or four Wigner 3–j symbols; in fact, we need not know in advance the sum rule which will apply during the simplification.

```
> Racah_evaluate(rcc):
> Racah_print(''):

--->
                 (-2 J12 + j3 - 2 M - 3 j1 - 2 J23 + J - j2)
            (-1)

                               1/2           1/2
                    (2 J12 + 1)     (2 J23 + 1)

                        delta(J,J)  delta(M,M)

                        w6j(J12,j3,J,J23,j1,j2)
```

From this evaluation we immediately obtain a result which can easily be rewritten to

$$(-1)^{j_1+j_2+j_3+J}\ [J_{12}, J_{23}]^{1/2} \begin{Bmatrix} j_1 & j_2 & J_{12} \\ j_3 & J & J_{23} \end{Bmatrix} \tag{4}$$

as it is displayed, for instance, in Ref. [13]. As might have been expected from Eq. (1), the recoupling coefficient `rcc` indeed simplifies to a Wigner 6–j symbol with no explicit dependence on M anymore, i.e. upon the choice of the quantization axis.

Our second example concerns the two-body interaction matrix elements between coupled states of two particles,

$$\langle ab; J_{ab} M_{ab} \,|\, g_{12} \,|\, cd; J_{cd} M_{cd} \rangle \tag{5}$$

and we take for an interaction $g_{12} \equiv g(\boldsymbol{r}_1, \boldsymbol{r}_2)$ which is symmetric in the particle indices. Such matrix elements arise very frequently in atomic and nuclear structure, for instance. To be more precise, here we choose the instantaneous Coulomb repulsion $g_{12} = 1/r_{12} \equiv 1/|\boldsymbol{r}_1 - \boldsymbol{r}_2|$ as it appears among the electrons in (non-relativistic) atomic and molecular computations. To find an expression for the interaction matrix (5), we start from the *radial-angular* decomposition of the "uncoupled" matrix elements [14],

$$\begin{aligned} \left\langle ab \left| \frac{1}{r_{12}} \right| cd \right\rangle &\equiv \left\langle n_a l_a j_a m_a(1)\, n_b l_b j_b m_b(2) \left| \frac{1}{r_{12}} \right| n_c l_c j_c m_c(1)\, n_d l_d j_d m_d(2) \right\rangle \\ &= \sum_{LM} (-1)^{L-M+j_a-m_a+j_b-m_b} \begin{pmatrix} j_a & L & j_c \\ -m_a & M & m_c \end{pmatrix} \begin{pmatrix} j_b & L & j_d \\ -m_b & -M & m_d \end{pmatrix} X^L(abcd)\,, \end{aligned} \tag{6}$$

with the so-called *effective interaction strength* $X^L(abcd)$ given by

$$\begin{aligned} X^L(abcd) &= \delta(j_a, j_c, L)\ \delta(j_b, j_d, L)\ \Pi^e(l_a, l_c, L)\ \Pi^e(l_b, l_d, L) \\ &\quad \times (-1)^L\ \langle j_a \| \boldsymbol{C}^{(L)} \| j_c \rangle \langle j_b \| \boldsymbol{C}^{(L)} \| j_d \rangle\ R^L(abcd)\,. \end{aligned} \tag{7}$$

Here, we use the notation $\delta(j_1, j_2, j_3)$ to represent a symbol which is equal to 1 if j_1, j_2, and j_3 satisfy the triangular condition and is zero otherwise. The reduced matrix elements of the normalized spherical harmonics are

$$\langle j_a \| \boldsymbol{C}^{(L)} \| j_c \rangle = (-1)^{j_a+1/2}\ [j_a, j_c]^{1/2} \begin{pmatrix} j_a & j_c & L \\ 1/2 & 1/2 & 0 \end{pmatrix}, \tag{8}$$

and the coefficients

$$\Pi^e(l_a, l_c, L) = \begin{cases} 1 & \text{even,} \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

reflect the conservation of parity for the one-particle states which are solutions to a given central field potential. Moreover, $R^L(abcd)$ denotes a pure radial integral [14] which we need not discuss in the present context.

We can use the decomposition of the "uncoupled" matrix elements (6) and an expansion in terms of Clebsch–Gordan coefficients as shown in the previous example in order to find an expression for the interaction matrix element (5),

$$\begin{aligned} \langle ab; J_{ab}M_{ab} \,|\, g_{12} \,|\, cd; J_{cd}M_{cd} \rangle &= \sum_{m_a m_c, m_b m_d} \left\langle ab \left| \frac{1}{r_{12}} \right| cd \right\rangle \langle J_{ab}M_{ab} \,|\, j_a m_a j_b m_b \rangle \, \langle j_c m_c j_d m_d \,|\, J_{cd}M_{cd} \rangle \\ &= \sum_L X^L(abcd) \sum_{M m_a m_c m_b m_d} \langle J_{ab}M_{ab} \,|\, j_a m_a j_b m_b \rangle \, \langle j_c m_c j_d m_d \,|\, J_{cd}M_{cd} \rangle \\ &\quad \times (-1)^{L-M+j_a-m_a+j_b-m_b} \begin{pmatrix} j_a & L & j_c \\ -m_a & M & m_c \end{pmatrix} \begin{pmatrix} j_b & L & j_d \\ -m_b & -M & m_d \end{pmatrix}. \end{aligned} \tag{10}$$

Again, we can try to simplify the sum over the magnetic quantum numbers by means of the Racah package. We enter this sum in a manner similar to our first example and assign it to the variable `coupledg12`. By using the command Racah_evaluate(), we then obtain

```
> Racah_evaluate(coupledg12):
> Racah_print(''):

--->
                         (-Mab - Mcd + Jcd - jc - jb - L)
                    (-1)

                                       1/2
                          (2 Jab + 1)
                          --------------
                                       1/2
                          (2 Jcd + 1)

                  delta(Jcd,Jab)  delta(Mcd,Mab)

                        w6j(jc,jd,Jcd,jb,ja,L)
```

From this reply of Maple, we find the matrix element (5) for a scalar interaction between coupled two-particle states to be given by

$$\langle ab; J_{ab}M_{ab} \,|\, g_{12} \,|\, cd; J_{cd}M_{cd} \rangle = \sum_L (-1)^{j_c+j_d+J_{cd}} \begin{Bmatrix} j_c & j_d & J_{cd} \\ j_b & j_a & L \end{Bmatrix} X^L(abcd)\, \delta_{J_{ab}J_{cd}}\, \delta_{M_{ab}M_{cd}}. \tag{11}$$

The same result can be found in various textbooks about the structure of many-particle quantum systems, for instance by Heyde [12], Chapter 3.

The simplification of even more complex Racah expressions is frequently required in the calculation of Feynman–Goldstone diagrams in various forms of many-body perturbation theory. For a closed-shell atom and assuming a Hartree–Fock basis, for example, there are just two Feynman–Goldstone diagrams which represent

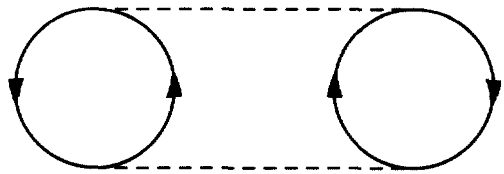

Fig. 1. One of the Feynman–Goldstone diagrams which occur in the second-order correlation contributions of atoms.

the second-order correlation contribution to the total energy of the atom [4]. One of these diagrams is shown in Fig. 1 and can be written as

$$D = \frac{1}{2} \sum_{abrs} \frac{\left\langle ab \left| \frac{1}{r_{12}} \right| rs \right\rangle \left\langle rs \left| \frac{1}{r_{12}} \right| ab \right\rangle}{(\epsilon_a + \epsilon_b - \epsilon_r - \epsilon_s)} . \tag{12}$$

The matrix elements in the nominator of this expression are those between "uncoupled" two-electron states (6). In this algebraic form (12), the indices $a \equiv (n_a l_a j_a m_a)$, $b \equiv (n_b l_b j_b m_b)$ run over all *occupied* one-particle states whereas the indices $r \equiv (n_r l_r j_r m_r), \ldots$ run over all *unoccupied* states in the Hartree–Fock determinant. In general, the corresponding one-particle energies $\epsilon_a = \epsilon_{n_a l_a j_a}$ do not depend on the magnetic quantum numbers.

Using the radial–angular decomposition (6) of the matrix elements from the previous example, we obtain for this "diagram"

$$D = \frac{1}{2} \sum_{n_a l_a j_a, \ldots, L_1 L_2} \frac{X^{L_2}(abrs)\, X^{L_1}(rsab)}{(\epsilon_a + \epsilon_b - \epsilon_r - \epsilon_s)} \mathcal{M} , \tag{13}$$

with

$$\mathcal{M} = \sum_{m's, M's} (-1)^S \begin{pmatrix} j_r & L_1 & j_a \\ -m_r & M_1 & m_a \end{pmatrix} \begin{pmatrix} j_s & L_1 & j_b \\ -m_b & -M_1 & m_b \end{pmatrix}$$
$$\times \begin{pmatrix} j_a & L_2 & j_r \\ -m_a & M_2 & m_r \end{pmatrix} \begin{pmatrix} j_b & L_2 & j_s \\ -m_b & -M_2 & m_s \end{pmatrix} \tag{14}$$

and $S = L_1 - M_1 + L_2 - M_2 + j_a - m_a + j_b - m_b + j_r - m_r + j_s - m_s$. Following the lines of the two previous examples, it is now a straightforward task to enter the expression $\mathcal{M}$ and to "simplify" it with the Racah program. We assign the sum as a whole to the variable M,

```
> w1 := Racah_set(w3j,jr,L1,ja,-mr,M1,ma):
> w2 := Racah_set(w3j,js,L1,jb,-ms,-M1,mb):
> w3 := Racah_set(w3j,ja,L2,jr,-ma,M2,mr):
> w4 := Racah_set(w3j,jb,L2,js,-mb,-M2,ms):
> M := Racah_set(Racahexpr):
> M := Racah_add(njsymbol,w1,w2,w3,w4,M):
> M := Racah_add(phase,L1-M1+L2-M2+ja-ma+jb-mb+jr-mr+js-ms,M):
> M := Racah_add(sum,{ma,mb,mr,ms,M1,M2},M):
> Racah_print(M):

--->
                          SUM{mr,M2,mb,ms,ma,M1}

          (L1 - M1 + L2 - M2 + ja - ma + jb - mb + jr - mr + js - ms)
      (-1)
```

```
                    w3j(jr,L1,ja,-mr,M1,ma)

                    w3j(js,L1,jb,-ms,-M1,mb)

                    w3j(ja,L2,jr,-ma,M2,mr)

                    w3j(jb,L2,js,-mb,-M2,ms)
```

and evaluate it by the command Racah_evaluate().

```
> M := Racah_evaluate(M);
> Racah_print(M):
--->
                           SUM{M2,mb,ms,M1}

           (2 L1 - 2 M1 + 2 L2 - M2 + ja + jb - mb + js - ms - jr)
       (-1)

                                  1
                              --------
                              2 L1 + 1

         delta(L1,L2)  delta(-M1,M2)  triangle(L1,ja,jr)

                      w3j(js,L1,jb,-ms,-M1,mb)

                      w3j(jb,L2,js,-mb,-M2,ms)
```

Here, a first step of the evaluation does not yield the final result. In order to proceed, we simplify the Kronecker δ-factors and then re-evaluate the expression

```
> M := Racah_simplifydeltas(M):
> M := Racah_evaluate(M):
> Racah_print(M):
--->

                              SUM{M1}

                 (3 L1 + 3 L2 + ja + jb - jr - js)
             (-1)

                                 1
                            -----------
                                      2
                            (2 L1 + 1)

delta(L1,L2) triangle(L1,ja,jr) delta(L1,L2) delta(M1,M1) triangle(L1,jb,js)
```

In this case, the summation over M1 cannot be evaluated automatically

```
> Racah_simplifydeltas(M);

The following tdelta cannot be simplified automatically; tdelta =
['delta#', M1, M1]
```

since the *range of* M_1 *values* is not being recognized by the Racah program itself. Simplications like that one should be done interactively instead.

Summarizing the different steps of the evaluation, the *angular reduction* of the diagram D thus gives rise to

$$D = \frac{1}{2} \sum_{n_a l_a j_a, \ldots, L} \frac{(-1)^{j_a+j_b-j_r-j_s}}{[L]} \frac{X^L(abrs)\, X^L(rsab)}{(\epsilon_a + \epsilon_b - \epsilon_r - \epsilon_s)} \delta(L, j_a, j_c)\, \delta(L, j_b, j_d) \,, \tag{15}$$

In the present case, we could have obtained the same result more elegantly using graphical rules for the coupling of the angular momenta. But not every reader will be familiar with these rules. Moreover, this example already demonstrates that simplifying complex expressions from Racah algebra is a straightforward matter. This advantage remains the same for more elaborate situations where graphical methods might become awkward and rather error-prone with respect to the weight and phase factors.

5. Outlook

The Racah program will be developed further in the future. There are different possible directions which will help to make this CA tool more powerful for the study of many-particle systems. Apart from the Wigner symbols and Clebsch–Gordan coefficients, the spherical harmonics $Y_{lm}(\vartheta, \varphi)$ and the reduced matrix elements of the rotation operator $d^{(j)}_{mm'}(\beta)$ play an inherent role in Racah algebra. The spherical harmonics are the well-known eigenfunctions of the orbital angular momentum operator. The close relation between them and the Wigner n–j symbols can, for instance, easily be seen from the Clebsch–Gordan expansion of products of two or more spherical harmonics with the same angular arguments [6]. Moreover, there appear the vector and some types of tensor spherical harmonics in many derivations concerning atomic and nuclear angular dependent properties. We have now started to implement the spherical harmonics into the Racah program, i.e. into the previously established structure of a Racah expression. Special commands are designed for numerical computations on spherical harmonics, their graphical representation as well as for various useful expansions.

The spherical harmonics $Y_{lm}(\vartheta, \varphi)$ and the reduced matrix elements $d^{(j)}_{mm'}(\beta)$ depend not only on different quantum numbers but also on a set of continuous angular variables. Therefore, different types of integrals over these variables may also appear during some evaluation. Often, this integration can be carried out analytically and then again results in entities from Racah algebra. Thus, such integral representations should also be included in some appropriate form in our forthcoming work.

In Section 4, our third example showed the close interconnection of atomic many-body perturbation theory with the application of Racah algebra techniques. Beside of the angular parts of the matrix elements and diagrams one would also like to treat the *radial parts* within a similar framework. In such cases, the gradient of the spherical harmonics (as they appear in the matrix elements) depends on the structure of the associated radial functions. This must be reflected in an appropriate data structure during further course of developing the Racah program.

Acknowledgements

This work has been supported by the Deutsche Forschungsgemeinschaft (DFG).

Appendix A. Further commands to the Racah package

For quick reference, we add here new and revised procedures of the Racah package to the description of paper I, Appendix C. We will only present those procedures in more detail which are important for interactive work whereas all other procedures at some hidden level in the program will remain unexplained. The whole package now contains a total of about 80 subprograms.

The presentation follows, again, the style of *The Maple Handbook* by Redfern [9]. Reference to the internal representation of all data structures as they appear as input and output of the commands were given in paper I, Appendix B.

Racah_evaluate(wexpr)

Attempts to simplify wexpr which can be either of type wnj or Racahexpr.

Output: A Racahexpr is returned if the simplification was successful and a [NULL] list otherwise.

Argument options: (Racahexpr,*Regge*) to apply also all symmetric forms due to Regge [15] to the Wigner n–j symbols. This optional argument, however, can result in a very time- and memory-consuming evaluation process.

Additional information: A Racahexpr is considered to be simplified if the number of summation indices and/or the number of Wigner n–j symbols is reduced. ♣ The argument wexpr is converted into a Racahexpr independent of the given type of the argument. Then, the procedure normally invokes Racah_evaluateRacahexpr() to perform the attempted evaluation. ♣ The procedure mainly attempts three ways of simplification by

(i) Applying a set of *special values* for the Wigner n–j symbols.
(ii) Using the orthogonality properties for sums of products of two Wigner n–j symbols of the same type.
(iii) Using a variety of sum rules and incomplete orthogonality relations which can be found in the literature. A brief compilation of these rules is displayed in Section 2 Tables 1 and 2.

Whereas special-value evaluation applies to individual Wigner n–j symbols, the methods in steps (ii) and (iii) requires analyzing the whole Racahexpr including all summation indices, the dependence on variables in the remaining parts of Racahexpr as well as the overall phase. ♣ If a simplification is found and replaced by the program, the Racah expression might be returned before the same rule or other sum rules are "probed" again. Therefore, some Racah expressions may require a two-step evaluation or even several steps using this command to find the final answer. ♣ A list of important special values can be found by Edmonds [16] and in the monograph by Varshalovich et al. [6]. ♣ The simplification of complex Racahexpr can be both very time- and memory-consuming.

See also: Racah_add(), Racah_compute(), Racah_delete().

Racah_useorthogonality(Racahexpr)

Attempts to simplify a Racahexpr by using the orthogonality relations of two Wigner n–j symbols of the same type.

Output: A Racahexpr is returned if the simplification was successful and a [NULL] list otherwise.

Argument options: (Racahexpr,*Regge*) to apply also all symmetric forms due to Regge [15] to the Wigner n–j symbols. This optional argument, however, can result in a very time- and memory-consuming evaluation process.

Additional information: There are two orthogonality relations for the Wigner 3–j symbols, and one relation for each, the Wigner 6–j and 9–j symbols. These formulas are shown in detail as in-line comments in the Maple procedures. ♣ In the standard application, the procedure considers all basic symmetric forms of the Wigner n–j symbols and "compares" them with some internal representation of the corresponding orthogonality relation. ♣ If an orthogonality is found for one type of Wigner n–j symbols, the simplification is carried out

Table A.1
Subprograms to the commands Racah_evaluate( ) and Racah_usesumrules( )

Racah_usesumrulesforonew3j( ) Racah_usesumrulesforonew9j( )	Racah_usesumrulesforonew6j( )
Racah_usesumrulesfortwow3j( ) Racah_usesumrulesforonew6jonew9j( )	Racah_usesumrulesfortwow6j( ) Racah_usesumrulesfortwow9j( )
Racah_usesumrulesforthreew3j( ) Racah_usesumrulesforthreew6j( ) Racah_usesumrulesforthreew9j( )	Racah_usesumrulesfortwow3jonew6j( ) Racah_usesumrulesfortwow6jonew9j( )
Racah_usesumrulesforfourw3j( ) Racah_usesumrulesforthreew6jonew9j( ) Racah_usesumrulesforonew6jthreew9j( )	Racah_usesumrulesforfourw6j( ) Racah_usesumrulesfortwow6jtwow9j( )
Racah_usesumrulesforsixw3j( )	

Each subprogram attempts the simplification of a given Racah expression by one or a few some rules, the type of which is simply indicated by the name of the corresponding procedure. They all return (analogue to the command Racah_usesumrules( )) a valid Racah expression if the simplification was successful and a [NULL] list otherwise. Similar to Racah_usesumrules( ), each subprogram may be invoked separately with the optional argument list (Racahexpr,*Regge*) (cf. Tables 1 and 2 in Section 2).

and the simplified Racahexpr returned. The procedure does not look for further simplifications for the same type of Wigner n–j symbols but this could easily be done by applying the procedure again to the result of the previous step.

See also: Racah_usespecialvalues(), Racah_usesumrules().

Racah_usesumrules(Racahexpr)

Attempts to simplify a Racahexpr by using various known sum rules for products of Wigner n–j symbols in this expression.

Output: A Racahexpr is returned if the simplification was successful and a [NULL] list otherwise.

Argument options: (Racahexpr,*Regge*) to apply also all symmetric forms due to Regge [15] to the Wigner n–j symbols. This optional argument, however, can result in a very time- and memory-consuming evaluation process.

Additional information: The command invokes a set of procedures which classify the *simplification* into groups of products of Wigner n–j symbols as indicated by the names of the corresponding subprocedures. This modular structure easily allows to add further rules for simplification. ♣ The formulas of all sum rules are shown in detail by Varshalovich et al. [6] (see Tables 1 and 2) and as in-line comments in the Maple procedures. ♣ The procedure considers all basic symmetric forms of the Wigner n–j symbols and "compares" them with the internal representation of the corresponding sum rule. ♣ If a simplification due to one type of a given sum rule is found, this rule is applied and the simplified Racahexpr is returned. The procedure does not search for further simplifications by the same rule but this could easily be done by applying the same procedure again to the result of the previous step. ♣ In Table A.1, the subprograms of Racah_usesumrules() are listed (not alphabetically but) by the number and type of Wigner n–j symbols. For an equal number of Wigner n–j symbols we report the 3–j symbols before the 6–j symbols and these again before the 9–j symbols.

See also: Racah_useorthogonality(), Racah_usespecialvalues().

References

[1] G. Racah, Phys. Rev. 61 (1941) 186; 62 (1942) 438; 63 (1943) 367.
[2] A.P. Yutsis, I.B. Levinson, V.V. Vanagas, The theory of Angular Momentum (Israel Program for Scientific Translation, Jerusalem, 1962).
[3] E. El-Baz, B. Castel, Graphical Methods of Spin Algebras in Atomic, Nuclear, and Particle Physics (Marcel Decker, New York, 1972).
[4] I. Lindgren, J. Morrison, Atomic Many-Body Theory, 2nd ed. (Springer, Berlin, 1986).
[5] S. Fritzsche, Comput. Phys. Commun. 103 (1997) 51.
[6] D.A. Varshalovich, A.N. Moskalev, V.K. Khersonskii, Quantum Theory of Angular Momentum (World Scientific, Singapore, 1988).
[7] U. Fano, G. Racah, Irreducible Tensor Sets (Academic Press, New York, 1959).
[8] M. Rotenberg, R. Bivins, N. Metrapolis, J.K. Wooten jr., The $3-j$ and $6-j$ Symbols (The Technology Press, Cambridge, MA, 1959).
[9] D. Redfern, The Maple Handbook (Springer, New York, Berlin, 1994).
[10] E.U. Condon, G.H. Shortley, The Theory of Atomic Spectra (Cambridge Univ. Press, Cambridge, 1935).
[11] A. De-Shalit, I. Talmi, Nuclear Shell Theory (Academic Press, New York, 1963).
[12] K.L.G. Heyde, The Nuclear Shell Model (Springer, Berlin, 1994).
[13] A. Messiah, Quantum Mechanics, Vol. II (North-Holland, Amsterdam, 1974) Section 13.5.
[14] I.P. Grant, in: Methods in Computational Chemistry, Vol. 2, S. Wilson, ed. (Plenum, New York, 1989) p. 1.
[15] T. Regge, Nuovo Cimento 10 (1958) 544.
[16] A.R. Edmonds, Angular Momentum in Quantum Mechanics (Princeton Univ. Press, New York, 1957).