

> restart;

## Maple 12

The procedure W3j is derived from the C program described by William J. Thompson.  
William J. Thompson, Angular Momentum: An Illustrated Guide to Rotational Symmetries for Physical Systems,  
John Wiley and Sons Inc., 1994

### Addition of Four Angular Momenta

Using Racah's formula to determine Wigner 3j coefficients

$$\begin{bmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{bmatrix} = (-1)^{j_1 - j_2 - m_3} \cdot N \cdot S$$

$$N = \sqrt{\frac{(j_3 + j_1 - j_2)!(j_3 - j_1 + j_2)!(j_1 + j_2 - j_3)!(j_3 - m_3)!(j_3 + m_3)!}{(j_1 + j_2 + j_3 + 1)!(j_1 - m_1)!(j_1 + m_1)!(j_2 - m_2)!(j_2 + m_2)!}}$$

$$S = \sum_{kmin}^{kmax} \frac{(-1)^{(kmin + j_2 + m_2)} (j_2 + j_3 + m_1 - k)!(j_1 - m_1 + k)!}{(k)!(j_3 - j_1 + j_2 - k)!(j_3 - m_3 - k)!(k + j_1 - j_2 + m_3)!}$$

where

kmin = maximum of [0 and j2 - j1 - m3] and kmax = minimum of [j3 - j1 + j2 and j3 - m3]

Clebsch-Gordan Coefficients, C-G, from Wigner 3j Coefficients

$$\langle j_1, j_2; m_1, m_2 | j_1, j_2; j_3, m_3 \rangle = (-1)^{-j_1 + j_2 - m_3} \sqrt{2j_3 + 1} \begin{bmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & -m_3 \end{bmatrix}$$

\*\*\*\* Note that we are using the following conventions \*\*\*\*

coefficient  $\{a, b, c, m_a, m_b, m_c\} \equiv \langle a, b; m_a, m_b | a, b, c, m_c \rangle$

C-G value  $[\pm N] \equiv \pm \sqrt{N}$

**This procedure determines the Wigner 3j coefficient using Racah's formula**

```

> W3j :=proc(j1, j2, j3, m1, m2, m3)
    local k, kmin, kmax, N, S, P;

    kmin := max(0, j2 - j1 - m3);
    kmax := min( j3 - j1 + j2, j3 - m3);

    P := ( -1 )(j1 - j2 - m3);

    N :=  $\sqrt{\frac{(j3 + j1 - j2)!(j3 - j1 + j2)!(j1 + j2 - j3)!(j3 - m3)!(j3 + m3)!}{(j1 + j2 + j3 + 1)!(j1 - m1)!(j1 + m1)!(j2 - m2)!(j2 + m2)!}}$ ;

    S :=  $\sum_{k=kmin}^{kmax} \frac{(-1)^{(k + j2 + m2)} \cdot (j2 + j3 + m1 - k)!(j1 - m1 + k)!}{(k)!(j3 - j1 + j2 - k)!(j3 - m3 - k)!(k + j1 - j2 + m3)!}$ ;

    return (P·N·S);

end proc:

```

**This procedure returns a signed squared coefficient.**

$$\pm a\sqrt{n} = \pm a^2 n = \pm N$$

```

> SQ :=proc(n)
    local sign, c;
    if (evalf(n) < 0) then sign := -1 else sign := 1 end if;
    c := simplify( (n·n) · sign );
    return (c);
end proc:

```

This procedure determines the value of the C-G coefficients; e.g.,  $\langle j_1, j_2; m_1, m_2 | j_1, j_2; j_{12}, m_{12} \rangle$

$$\langle j_1, j_2; m_1, m_2 | j_1, j_2; j_{12}, m_{12} \rangle = [\pm N]$$

$$[\pm N] \equiv \pm \sqrt{N}$$

```

> VCo := proc(j1, j2, j3, j4, j12, j123, J, M, v)
    local m1, m2, m3, m4, m12, m123, c1, c2, c3, s;
    s := " = ";
    if v = 0 then printf("|%a,%a,%a,%a;%a,%a,%a,%a>\n", j1, j2, j3, j4, j12, j123, J, M) end if;
    for m1 from -j1 by 1 to j1 do
        for m2 from -j2 by 1 to j2 do
            for m12 from -j12 by 1 to j12 do
                for m3 from -j3 by 1 to j3 do
                    for m123 from -j123 by 1 to j123 do
                        for m4 from -j4 by 1 to j4 do
                            if (m1 + m2) = m12 and (m12 + m3) = m123 and (m123 + m4) = M then
                                c1 := (-1)(-j1 + j2 - m12) ·  $\sqrt{2 \cdot j_{12} + 1}$  · W3j(j1, j2, j12, m1, m2, -m12);
                                c2 := (-1)(-j12 + j3 - m123) ·  $\sqrt{2 \cdot j_{123} + 1}$  · W3j(j12, j3, j123, m12, m3, -m123);
                                c3 := (-1)(-j123 + j4 - M) ·  $\sqrt{2 \cdot J + 1}$  · W3j(j123, j4, J, m123, m4, -M);
                                printf("          %s[%a]", s, SQ(c1·c2·c3) );
                                printf("|%a,%a,%a,%a;%a,%a,%a,%a> \n", j1, j2, j3, j4, m1, m2, m3, m4);
                                s := " + ";
                            end if;
                        end do;
                    end do;
                end do;
            end do;
        end do;
    end do;
    print( );
end proc:

```

This procedure returns the product of the CG coefficients and the state  $|j_1, j_2, j_3, j_4; m_1, m_2, m_3, m_4\rangle$

$\{j_1, j_2, j_{12}, m_1, m_2, m_{12}\} \{j_{12}, j_3, j_{123}, m_{12}, m_3, m_{123}\} \{j_{123}, j_4, J, m_{123}, m_4, M\} |j_1, j_2, j_3, j_4; m_1, m_2, m_3, m_4\rangle$

where  $j_{12} = j_1 + j_2$ ,  $j_{123} = j_{12} + j_3$  and  $J = j_{123} + j_4$   
 $m_{12} = \{j_{12}, j_{12} - 1, \dots, -j_{12}\} = m_1 + m_2$   
 $m_{123} = \{j_{123}, j_{123} - 1, \dots, -j_{123}\} = m_{12} + m_3$   
 $M = \{J, J - 1, \dots, -J\} = m_{123} + m_4$

```
> SCo := proc(j1, j2, j3, j4, j12, j123, J, M)
    local m1, m2, m3, m4, m12, m123, s;
    s := " = ";
    # printf("          =====\n");
    printf("|%a,%a,%a,%a;%a,%a,%a,%a>\n", j1, j2, j3, j4, j12, j123, J, M);
    for m1 from -j1 by 1 to j1 do
        for m2 from -j2 by 1 to j2 do
            for m12 from -j12 by 1 to j12 do
                for m3 from -j3 by 1 to j3 do
                    for m123 from -j123 by 1 to j123 do
                        for m4 from -j4 by 1 to j4 do
                            if (m1 + m2) = m12 and (m12 + m3) = m123 and (m123 + m4) = M then
                                printf("          %s{%a,%a,%a,%a,%a,%a,%a,%a}", s, j1, j2, j12, m1, m2, m12 );
                                printf("{%a,%a,%a,%a,%a,%a,%a,%a}", j12, j3, j123, m12, m3, m123);
                                printf("{%a,%a,%a,%a,%a,%a,%a,%a}", j123, j4, J, m123, m4, M);
                                printf("|%a,%a,%a,%a;%a,%a,%a,%a> \n", j1, j2, j3, j4, m1, m2, m3, m4);
                                s := " + ";
                            end if;
                        end do;
                    end do;
                end do;
            end do;
        end do;
    end do;
end proc;
```

**Main procedure. Add4j adds four angular momenta - Add4j(j1,j2,j3,j4)**  
**using the coupling scheme  $j_1 + j_2 = j_{12}$ ,  $j_{12} + j_3 = j_{123}$ , and  $j_{123} + j_4 = J$**

$$|j_1 - j_2| \leq j_{12} \leq (j_1 + j_2)$$

$$m_{12} = \{j_{12}, j_{12} - 1, \dots, -j_{12}\}$$

$$|j_{12} - j_3| \leq j_{123} \leq (j_{12} + j_3)$$

$$m_{123} = \{j_{123}, j_{123} - 1, \dots, -j_{123}\}$$

$$|j_{123} - j_4| \leq J \leq (j_{123} + j_4)$$

$$M = \{J, J - 1, \dots, -J\}$$

```
> Add4j := proc(j1, j2, j3, j4, v)
    local c, j12, j123, J, M;
    printf("\n   There are %a |j1,j2,j3,j4;j12,j123,J,M> states where,\n\n", (2*j1 + 1) * (2*j2 + 1)
    * (2*j3 + 1) * (2*j4 + 1));
    c := 0 : # counter
    for j12 from (j1 + j2) by -1 to |j1 - j2| do
        for j123 from (j12 + j3) by -1 to |j3 - j12| do
            for J from (j123 + j4) by -1 to |j4 - j123| do
                for M from J by -1 to -J do
                    c := c + 1 : printf("%a.", c);
                    if v = 1 then SCo(j1, j2, j3, j4, j12, j123, J, M) end if;
                    VCo(j1, j2, j3, j4, j12, j123, J, M, v);
                end do;
            end do;
        end do;
    end do;
end proc;
```

### Example:

Add4j(j1,j2,j3,j4,v)

where v={ 1 or 0} 1 = print CG coefficients & values, 0 = print CG values only

$$> \text{Add4j}\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0\right);$$

There are 16  $|j_1, j_2, j_3, j_4; j_{123}, J, M\rangle$  states where,

1.  $|1/2, 1/2, 1/2, 1/2; 1, 3/2, 2, 2\rangle$   
 $= [1] |1/2, 1/2, 1/2, 1/2; 1/2, 1/2, 1/2, 1/2\rangle$
2.  $|1/2, 1/2, 1/2, 1/2; 1, 3/2, 2, 1\rangle$   
 $= [1/4] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, 1/2, 1/2\rangle$   
 $+ [1/4] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, 1/2, 1/2\rangle$   
 $+ [1/4] |1/2, 1/2, 1/2, 1/2; 1/2, 1/2, -1/2, 1/2\rangle$   
 $+ [1/4] |1/2, 1/2, 1/2, 1/2; 1/2, 1/2, 1/2, -1/2\rangle$
3.  $|1/2, 1/2, 1/2, 1/2; 1, 3/2, 2, 0\rangle$   
 $= [1/6] |1/2, 1/2, 1/2, 1/2; -1/2, -1/2, 1/2, 1/2\rangle$   
 $+ [1/6] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, -1/2, 1/2\rangle$   
 $+ [1/6] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, 1/2, -1/2\rangle$   
 $+ [1/6] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, -1/2, 1/2\rangle$   
 $+ [1/6] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, 1/2, -1/2\rangle$   
 $+ [1/6] |1/2, 1/2, 1/2, 1/2; 1/2, 1/2, -1/2, -1/2\rangle$
4.  $|1/2, 1/2, 1/2, 1/2; 1, 3/2, 2, -1\rangle$   
 $= [1/4] |1/2, 1/2, 1/2, 1/2; -1/2, -1/2, -1/2, 1/2\rangle$   
 $+ [1/4] |1/2, 1/2, 1/2, 1/2; -1/2, -1/2, 1/2, -1/2\rangle$   
 $+ [1/4] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, -1/2, -1/2\rangle$   
 $+ [1/4] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, -1/2, -1/2\rangle$
5.  $|1/2, 1/2, 1/2, 1/2; 1, 3/2, 2, -2\rangle$   
 $= [1] |1/2, 1/2, 1/2, 1/2; -1/2, -1/2, -1/2, -1/2\rangle$
6.  $|1/2, 1/2, 1/2, 1/2; 1, 3/2, 1, 1\rangle$   
 $= [-1/12] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, 1/2, 1/2\rangle$   
 $+ [-1/12] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, 1/2, 1/2\rangle$   
 $+ [-1/12] |1/2, 1/2, 1/2, 1/2; 1/2, 1/2, -1/2, 1/2\rangle$   
 $+ [3/4] |1/2, 1/2, 1/2, 1/2; 1/2, 1/2, 1/2, -1/2\rangle$
7.  $|1/2, 1/2, 1/2, 1/2; 1, 3/2, 1, 0\rangle$   
 $= [-1/6] |1/2, 1/2, 1/2, 1/2; -1/2, -1/2, 1/2, 1/2\rangle$   
 $+ [-1/6] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, -1/2, 1/2\rangle$   
 $+ [1/6] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, 1/2, -1/2\rangle$   
 $+ [-1/6] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, -1/2, 1/2\rangle$   
 $+ [1/6] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, 1/2, -1/2\rangle$   
 $+ [1/6] |1/2, 1/2, 1/2, 1/2; 1/2, 1/2, -1/2, -1/2\rangle$

$$\begin{aligned}
8. & |1/2, 1/2, 1/2, 1/2; 1, 3/2, 1, -1\rangle \\
&= [-3/4] |1/2, 1/2, 1/2, 1/2; -1/2, -1/2, -1/2, 1/2\rangle \\
&+ [1/12] |1/2, 1/2, 1/2, 1/2; -1/2, -1/2, 1/2, -1/2\rangle \\
&+ [1/12] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, -1/2, -1/2\rangle \\
&+ [1/12] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, -1/2, -1/2\rangle
\end{aligned}$$

$$\begin{aligned}
9. & |1/2, 1/2, 1/2, 1/2; 1, 1/2, 1, 1\rangle \\
&= [-1/6] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, 1/2, 1/2\rangle \\
&+ [-1/6] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, 1/2, 1/2\rangle \\
&+ [2/3] |1/2, 1/2, 1/2, 1/2; 1/2, 1/2, -1/2, 1/2\rangle
\end{aligned}$$

$$\begin{aligned}
10. & |1/2, 1/2, 1/2, 1/2; 1, 1/2, 1, 0\rangle \\
&= [-1/3] |1/2, 1/2, 1/2, 1/2; -1/2, -1/2, 1/2, 1/2\rangle \\
&+ [1/12] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, -1/2, 1/2\rangle \\
&+ [-1/12] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, 1/2, -1/2\rangle \\
&+ [1/12] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, -1/2, 1/2\rangle \\
&+ [-1/12] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, 1/2, -1/2\rangle \\
&+ [1/3] |1/2, 1/2, 1/2, 1/2; 1/2, 1/2, -1/2, -1/2\rangle
\end{aligned}$$

$$\begin{aligned}
11. & |1/2, 1/2, 1/2, 1/2; 1, 1/2, 1, -1\rangle \\
&= [-2/3] |1/2, 1/2, 1/2, 1/2; -1/2, -1/2, 1/2, -1/2\rangle \\
&+ [1/6] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, -1/2, -1/2\rangle \\
&+ [1/6] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, -1/2, -1/2\rangle
\end{aligned}$$

$$\begin{aligned}
12. & |1/2, 1/2, 1/2, 1/2; 1, 1/2, 0, 0\rangle \\
&= [1/3] |1/2, 1/2, 1/2, 1/2; -1/2, -1/2, 1/2, 1/2\rangle \\
&+ [-1/12] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, -1/2, 1/2\rangle \\
&+ [-1/12] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, 1/2, -1/2\rangle \\
&+ [-1/12] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, -1/2, 1/2\rangle \\
&+ [-1/12] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, 1/2, -1/2\rangle \\
&+ [1/3] |1/2, 1/2, 1/2, 1/2; 1/2, 1/2, -1/2, -1/2\rangle
\end{aligned}$$

$$\begin{aligned}
13. & |1/2, 1/2, 1/2, 1/2; 0, 1/2, 1, 1\rangle \\
&= [-1/2] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, 1/2, 1/2\rangle \\
&+ [1/2] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, 1/2, 1/2\rangle
\end{aligned}$$

$$\begin{aligned}
14. & |1/2, 1/2, 1/2, 1/2; 0, 1/2, 1, 0\rangle \\
&= [-1/4] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, -1/2, 1/2\rangle \\
&+ [-1/4] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, 1/2, -1/2\rangle \\
&+ [1/4] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, -1/2, 1/2\rangle \\
&+ [1/4] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, 1/2, -1/2\rangle
\end{aligned}$$

$$\begin{aligned}
 15. & |1/2, 1/2, 1/2, 1/2; 0, 1/2, 1, -1\rangle \\
 &= [-1/2] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, -1/2, -1/2\rangle \\
 &+ [1/2] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, -1/2, -1/2\rangle
 \end{aligned}$$

$$\begin{aligned}
 16. & |1/2, 1/2, 1/2, 1/2; 0, 1/2, 0, 0\rangle \\
 &= [1/4] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, -1/2, 1/2\rangle \\
 &+ [-1/4] |1/2, 1/2, 1/2, 1/2; -1/2, 1/2, 1/2, -1/2\rangle \\
 &+ [-1/4] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, -1/2, 1/2\rangle \\
 &+ [1/4] |1/2, 1/2, 1/2, 1/2; 1/2, -1/2, 1/2, -1/2\rangle
 \end{aligned}$$

(1)

>