Rotation matrices for real spherical harmonics: general rotations of atomic orbitals in space-fixed axes

Didier Pinchon[1] and Philip E. Hoggan[2]

October 16, 2006

(1) MIP, UMR 5640 CNRS.
University Paul Sabatier
118 route de Narbonne
31062 TOULOUSE Cedex, FRANCE.

(2) LASMEA, UMR 6602 CNRS.
University Blaise Pascal
24 avenue des Landais,
63177 AUBIERE Cedex, FRANCE.

Abstract

The angular factors of atomic orbitals are real spherical harmonics. This is independent of the choice of basis function. In the course of molecular electronic structure calculations, numerous rotations of real spherical harmonics are required in a suitably defined space-fixed co-ordinate system. The origin and axes are space-fixed and rotation matrices defined on a basis of spherical harmonics.

In this work, a highly compact expression and efficient evaluation of the rotation matrices are given for a real spherical harmonic basis. Relations to Gaunt coefficients are shown explicitly as are recurrence formulae for rotation matrices. This leads to extremely rapid and precise rotation algorithms.

The Wigner rotation matrices which are still often used in orbital rotations are shown to be completely surpassed by this approach. The present work is related to a method described by Kautz in the field of image processing but significant improvements have been made, especially in the study of structure and storage of the rotation matrices.

After complete testing using computer algebra, a numerical program was written in C. Numerical tests are cited in the closing sections of this work.

Keywords: atomic orbitals, rotations, real spherical harmonics, matrix structure, compact integer representation.

1 Introduction

The standard convention for atomic orbitals in quantum chemistry is to choose them to be real functions. Furthermore, since they are functions of independent variables, it is readily shown that they can be expressed as products of a radial factor and an angular factor. The angular factor will always be a spherical harmonic and the present work focusses on this angular variation. For conventional atomic orbitals, it is just a real spherical harmonic. A few special cases can be visualised very obviously (defined by the value of the azimuthal quantum number l integer or 0 with $lmax = n - 1$, n being the principal quantum number): if $l = 0$, the spherical harmonic is constant and the orbitals are spherically symmetric. If $l = 1$, the functions are axially symmetric and a suitable real basis is just the set of cartesian functions x, y and z. These are orthogonal and equivalent functions which can evidently be permuted indistinguishably with the exception of their orientation.
If $l = 2$, the set of cartesian functions of order l comprises six functions, whereas there can only be 5 $(2\,l + 1)$ independent real harmonics. The squared functions are, however, not all independent, since the equation describing a sphere provides a relationship between them. The two independent functions are determined from the orthogonality conditions and result in the choice of one symmetric and one anti-symmetric combination with respect to permutations of axes: $3z^2-1$ and x^2-y^2. The cartesian basis is over-defined for higher l and this procedure of determining the $2\,l + 1$ orthogonal real spherical harmonics may be accomplished with the appropriate definitions (*vide infra*).
Note that the value of the quantum number l determines the orbital angular momentum. The real spherical harmonics $S_{l,\,m}$are eigenfunctions of the magnitude operator $\mathbf{L}^2$:

$$\mathbf{L}^2\, S_{l,\,m} \;=\; l(l+1)\hbar\, S_{l,\,m}\;.$$

When m is non-zero, the real spherical harmonics are not eigenfunctions of the z-component operator $\mathbf{L_z}$ but this operator is nevertheless useful since the x and y orientations are interconverted by it.
The most frequently used rotation technique is that involving the Wigner rotation matrices as described, for example in [1]. The nature of these matrices suggests that it would be advantageous to investigate alternatives.
Previous work by Kautz has shown (in the context of image shading) [2] any general direct rotation in $\mathbb{R}^3$ may be accomplished by separate rotations about the z axis, then for rotation about y, a 90 degree rotation about x precedes the general rotation about z and then a rotation about x of -90 degrees in a real spherical harmonic basis. In the present work, a much simpler and more regular series of axial rotations and direct permutations of axes is proposed, leading to a very efficient set of half the number of matrix elements to store for rotation representation matrices as for those in Kautz' work. The structure of the elements is also elucidated fully and shown to be highly simplified. First, a rigorous discussion is made of rotation in a basis of real spherical harmonics, evidencing the simplest possible sparce matrix representations and suitable permutations of the cartesian axes. The aim of this work is to provide a highly efficient scheme for general rotations of the (real) atomic orbitals in a laboratory-fixed set of axes, in which the z axis is the quantisation axis.
Let us consider the rotation $R \in SO(3)$, the group of direct rotations in $\mathbb{R}^3$. We denote by $\hat{R}$ the operator defined on $L^2(S_2, d\Omega)$, where S_2 is the unit sphere with the area measure $d\Omega$, by

$$\hat{R}f(x,y,z) = f(R^{-1}(x,y,z)), (x,y,z) \in S_2\;. \tag{1}$$

For integers l and m with $l \geq 0$, $-l \leq m \leq l$, the normalized spherical harmonic function $Y_m^l(\theta, \phi)$

is defined by

$$Y_0^0(\theta,\phi) = \frac{1}{\sqrt{4\pi}}\ , \tag{2}$$

$$Y_m^l(\theta,\phi) = i^{m+|m|}\sqrt{\frac{(2l+1)(l-|m|)!}{4\pi(l+|m|)!}}\,P_l^{|m|}(\cos\theta)\,e^{im\phi}\ ,\ (l,m)\neq(0,0)\ , \tag{3}$$

where l et m are non-negative integers and $P_l(x)$ stands for the Legendre polynomial of degree l

$$P_l(x) = \frac{1}{2^l l!}\frac{d^l}{dx^l}(x^2-1)^l\ , \tag{4}$$

and $P_l^m(x)$ is the associated Legendre function defined for $m \geq 0$ by

$$P_l^m(x) = (-1)^m\sqrt{(1-x^2)^m}\frac{d^m}{dx^m}P_l(x),\ -1\leq x\leq 1\ . \tag{5}$$

For $l \geq 0$ and $-l \leq m \leq l$, the real spherical harmonic function $S_{m,l}(\theta,\phi)$ is defined by

$$S_{l,0}(\theta,\phi) = \sqrt{\frac{2l+1}{4\pi}}P_l(\cos\theta)\ , \tag{6}$$

$$S_{l,m}(\theta,\phi) = \begin{cases} \sqrt{\dfrac{(2l+1)(1-m)!}{2\pi(1+m)!}}P_l^m(\cos\theta)\cos(m\phi) & \text{for } m>0\ , \\ \sqrt{\dfrac{(2l+1)(1+m)!}{2\pi(1-m)!}}P_l^{-m}(\cos\theta)\sin(-m\phi) & \text{for } m<0\ . \end{cases} \tag{7}$$

Real spherical harmonics functions are related to spherical harmonics by the relations

$$S_{l,0}(\theta,\phi) = Y_0^l(\theta,\phi)\ , \tag{8}$$

$$S_{l,m}(\theta,\phi) = \begin{cases} \dfrac{1}{\sqrt{2}}\left[Y_m^l(\theta,\phi)+(-1)^m Y_{-m}^l(\theta,\phi)\right] & \text{for } m>0\ , \\ \dfrac{i}{\sqrt{2}}\left[(-1)^m Y_m^l(\theta,\phi)-Y_{-m}^l(\theta,\phi)\right] & \text{for } m<0\ . \end{cases} \tag{9}$$

Let us denote by E_l the subspace of $L^2(S_2,d\Omega)$ generated by the $2l+1$ (complex) spherical harmonic functions $Y_m^l, m=-l,\ldots,l$ or equivalently by the $2l+1$ real spherical harmonics $S_{l,m}, m=-l,\ldots,l$. $Y_l=[Y_m^l, m=-l,\ldots,l]$ and $S_l=[S_{l,m}, m=-l,\ldots,l]$ each constitute an orthonormal basis of E_l. Since E_l is invariant under rotation, a representation of $SO(3)$ of dimension $2l+1$ results. This representation is irreductible and all the irreductible representations of $SO(3)$ are obtained by varying l.

The specific problem addressed in the present work has already been studied in [3], [4]. It is the computation of the matrix of an arbitrary rotation on E_l in the basis of real spherical harmonics. Let us denote by $\mathbf{D}_l(\alpha,\beta,\gamma)$ the matrix representation of $\hat{R}$ on E_l in the basis Y_l and by $\Delta_l(\alpha,\beta,\gamma)$ in the basis S_l, when the rotation R is expressed as $R=R_Z(\gamma)R_Y(\beta)R_Z(\alpha)$ where R_Z and R_Y are rotation of axis $0z$ and $0y$ in $\mathbb{R}^3$ and α,β,γ the so-called Euler angles.

$$\mathbf{R} = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix}\begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}\ . \tag{10}$$

The matrix of a rotation $\hat{R}_Z(\alpha)$ on E_l with basis Y_l is the diagonal matrix of dimension $2l+1$ (with the diagonal elements $e^{-im\alpha}, m = -l, \ldots, l$).
If $\mathbf{C}_l$ denotes matrix representation of the change of basis between Y_l and S_l defined by $S_l = Y_l\mathbf{C}_l$, the matrix of $\hat{R}_Z(\alpha)$ in S_l, denoted by $\mathbf{X}_l(\alpha)$ is equal to

$$\mathbf{X}_l(\alpha) = \mathbf{C}_l^\star \text{ Diag}(e^{im\alpha}, m = -l, \ldots, l)\ \mathbf{C}_l\ . \tag{11}$$

where $\mathbf{C}^\star$ is the transpose conjugate matrix of $\mathbf{C}$. $\mathbf{C}$ is a complex unitary matrix : $\mathbf{C}^\star\mathbf{C} = \mathbf{I}_{2l+1}$.

The matrix $\mathbf{X}_l(\alpha)$ has non-zero elements only on its diagonal and anti-diagonal and is the commutative product of l in-plane rotations of angles $-\alpha, -2\alpha, \ldots, -l\alpha$ on the planes of coordinates $(l, l+2), (l-1, l+3), \ldots, (1, 2l+1)$.

For example

$$\mathbf{X}_3(\alpha) = \begin{bmatrix} \cos 3\alpha & 0 & 0 & 0 & 0 & 0 & \sin 3\alpha \\ 0 & \cos 2\alpha & 0 & 0 & 0 & \sin 2\alpha & 0 \\ 0 & 0 & \cos\alpha & 0 & \sin\alpha & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -\sin\alpha & 0 & \cos\alpha & 0 & 0 \\ 0 & -\sin 2\alpha & 0 & 0 & 0 & \cos 2\alpha & 0 \\ -\sin 3\alpha & 0 & 0 & 0 & 0 & 0 & \cos 3\alpha \end{bmatrix} \tag{12}$$

The following is obtained:

$$\Delta_l(\alpha, \beta, \gamma) = \mathbf{X}_l(\alpha)\ \mathbf{D}_l(\beta)\ \mathbf{X}_l(\gamma)\ , \tag{13}$$

where $\mathbf{D}_l(\beta)$ is the matrix of $\hat{R}_Y(\beta)$ on E_l with the basis S_l.

In [4], the authors provide a method of computing $\mathbf{D}_l(\beta)$ for a given value of β with a two order recurrence on l. This computation by recurrence must be done whenever $\mathbf{D}_l(\beta)$ is required, for each distinct value of β.

In this work, the fact that a rotation in $\mathbb{R}^3$ with axis Oy could be obtained by first exchanging coordinates y and z , before a rotation with axis Oz and then reversing the exchange of coordinates y and z is evidenced and applied. Let us denote by J this symmetric operator on $\mathbb{R}^3$ and by J_l its representation on E_l with matrix $\mathbf{J}_l$ for the base S_l.

Therefore

$$\mathbf{D}_l(\beta) = \mathbf{J}_l\mathbf{X}_l(\alpha)\mathbf{J}_l\ , \tag{14}$$

and thus

$$\Delta_l(\alpha, \beta, \gamma) = \mathbf{X}_l(\alpha)\ \mathbf{J}_l\mathbf{X}_l(\beta)\mathbf{J}_l\mathbf{X}_l(\gamma)\ . \tag{15}$$

The problem is thus reduced to the computation of $\mathbf{J}_l$ that can be done once and for all and simultaneously for the required values of l with a maximal precision and stored before any computation of rotation matrices on the basis of real spherical harmonics.

In conclusion to this paragraph, note that although the matrix of J_l has few non-zero elements in the basis Y_l, this is not the case of $\mathbf{J}_l$. For example, for $l = 3$:

$$J_l Y_l = Y_l \begin{bmatrix} \frac{1}{8} & i\frac{\sqrt{6}}{8} & -\frac{\sqrt{15}}{8} & -i\frac{\sqrt{5}}{4} & \frac{\sqrt{15}}{8} & i\frac{\sqrt{6}}{8} & -\frac{1}{8} \\ -i\frac{\sqrt{6}}{8} & \frac{1}{2} & i\frac{\sqrt{10}}{8} & 0 & i\frac{\sqrt{10}}{8} & -\frac{1}{2} & -i\frac{\sqrt{6}}{8} \\ -\frac{\sqrt{15}}{8} & -i\frac{\sqrt{10}}{8} & -\frac{1}{8} & -i\frac{\sqrt{3}}{4} & \frac{1}{8} & -i\frac{\sqrt{10}}{8} & \frac{\sqrt{15}}{8} \\ i\frac{\sqrt{5}}{4} & 0 & i\frac{\sqrt{3}}{4} & 0 & i\frac{\sqrt{3}}{4} & 0 & i\frac{\sqrt{5}}{4} \\ \frac{\sqrt{15}}{8} & -i\frac{\sqrt{10}}{8} & \frac{1}{8} & -i\frac{\sqrt{3}}{4} & -\frac{1}{8} & -i\frac{\sqrt{10}}{8} & -\frac{\sqrt{15}}{8} \\ -i\frac{\sqrt{6}}{8} & -\frac{1}{2} & i\frac{\sqrt{10}}{8} & 0 & i\frac{\sqrt{10}}{8} & \frac{1}{2} & -i\frac{\sqrt{6}}{8} \\ -\frac{1}{8} & i\frac{\sqrt{6}}{8} & \frac{\sqrt{15}}{8} & -i\frac{\sqrt{5}}{4} & -\frac{\sqrt{15}}{8} & i\frac{\sqrt{6}}{8} & \frac{1}{8} \end{bmatrix} \tag{16}$$

and

$$J_l S_l = S_l \, \mathbf{J}_l = S_l \begin{bmatrix} 0 & 0 & 0 & \frac{\sqrt{10}}{4} & 0 & -\frac{\sqrt{6}}{4} & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\sqrt{6}}{4} & 0 & \frac{\sqrt{10}}{4} & 0 \\ \frac{\sqrt{10}}{4} & 0 & \frac{\sqrt{6}}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{4} & 0 & -\frac{\sqrt{15}}{4} \\ -\frac{\sqrt{6}}{4} & 0 & \frac{\sqrt{10}}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{\sqrt{15}}{4} & 0 & \frac{1}{4} \end{bmatrix} \tag{17}$$

The following lemma may also be proven.

Lemma 1.– *For every $l \geq 1$, every element x in $\mathbf{J}_l$ may be uniquely expressed as $x = a\sqrt{b}/c$ where a and b are relatively prime integers with $b > 0$ and c is a square-free positive integer, i.e. without squares in its prime factor decomposition.*

Proof.– Let us call pure quadratic numbers those satisfying $x^2 = q$ with q positive definite. From its definition, note that any real spherical harmonic function may be expressed by the product of a constant (i.e. a pure quadratic number divided by $\sqrt{\pi}$) by a homogeneous polynomial in x, y, z with rational coefficients. As the product of two pure quadratic numbers is a quadratic pure number, deduce that any element of $\mathbf{J}$ is obtained as the product of a pure quadratic number divided by π and a sum of integrals, expressed as follows:

$$I = \int x^\alpha y^\beta z^\gamma \, d\Omega \, . \tag{18}$$

Replacing x, y, z by their expression in spherical coordinates gives

$$I = \int_0^{2\pi} \cos^\alpha \phi \, \sin^\beta \phi \, d\phi \int_0^\pi \sin^{\alpha+\beta+1} \theta \, \cos^\gamma \theta \, d\theta \, . \tag{19}$$

Symmetries of the integrand in the first integral imply that it is zero if α or β is an odd integer. When α and β are both even integers, linearizing the product gives $\cos^\alpha \phi \, \sin^\beta$. Note that a non-zero term can only come from a constant and the first integral is thus the product of π by a rational number.

For a non-zero integral I, the second integral has the form

$$I_2 = \int_0^\pi \sin^{2k+1}\theta \, \cos^\gamma \theta \, d\theta \; . \tag{20}$$

Again by symmetry, $I_2 = 0$ if γ is an odd integer. For an even γ, replacing $\cos^2\theta$ by $1 - \sin^2\theta$ shows that I_2 is a linear combination of integrals such as:

$$\int_0^\pi \sin^{2n+1}\theta \, d\theta = \frac{2^{2n+1}(n!)^2}{(2n+1)!} \; , \; n \geq 1 \; , \tag{21}$$

with integer coefficients. Thus I is the product of a rational number by π which completes the proof.

Lemma 1 is strengthened by the following conjecture.

Conjecture.. *For every $l \geq 1$, every element x in $\mathbf{J}_l$ may be uniquely expressed as $x = a\sqrt{b}/2^n$ where $n \geq 0$, a is an integer, odd when $n \geq 1$, and b is a square-free positive integer.*

The validity of this conjecture has been checked with a computer algebra system for $1 \leq l \leq 40$. Furthermore, a special case will be proven below.

2 The block structure of $\mathbf{J}_l$

For odd l, $l = 2k+1$ with k, l integers, denote by $E_{xyz,k}$ the k-dimensional subspace of E_l generated by $S^l_{-l+2i-1}, i = 1, \ldots, k$. Basis functions of S_l divisible by xyz may be expressed as homogeneous polynomials in x, y, z and thus $E_{xyz,k}$ is the subspace of E_l of functions divisible by xyz. Of course $E_{xyz,k}$ is invariant under J_l. Let us denote by $\mathbf{A}_{xyz,k}$ the multiplication matrix by J_l in the basis $E_{xyz,k}$.
In a similar way, defining three $k+1$-dimensional subspaces of E_l, denoted by $E_{z,k+1}$, $E_{y,k+1}$ and $E_{x,k+1}$ for those functions divisible par z, y and x respectively, the appropriate bases are given in table 2

Name	Basis	Div. by	Dimension	Image under J_{2k+1}
$E_{xyz,k}$	$S^l_{-l+2i-1}, i = 1, \ldots, k$	xyz	k	$E_{xyz,k}$
$E_{z,k+1}$	$S^l_{2i-2}, i = 1, \ldots, k+1$	z	$k+1$	$E_{y,k+1}$
$E_{y,k+1}$	$S^l_{-l+2i-2}, i = 1, \ldots, k+1$	y	$k+1$	$E_{z,k+1}$
$E_{x,k+1}$	$S^l_{2i-1}, i = 1, \ldots, k+1$	x	$k+1$	$E_{x,k+1}$

Table 1: Orthogonal subspaces of E_{2k+1}.

The subspace $E_{x,k+1}$ is invariant under J_l, its matrix representation being denoted by $\mathbf{A}_{x,k+1}$. On the other hand J_l is an isometry of $\mathbf{E}_{z,k+1}$ onto $\mathbf{E}_{z,k+1}$. Let us denote by $\mathbf{A}_{z,k+1}$ its matrix representation in the previously defined basis.
Therefore, E_{2k+1} is the direct sum of the four orthogonal subspaces

$$E_{2k+1} = E_{xyz,k} \oplus E_{z,k+1} \oplus E_{y,k+1} \oplus E_{x,k+1} \; . \tag{22}$$

Now, the permutation σ of $[1, 2, \ldots, 4k+3]$ is introduced, defined by $\sigma([1, 2, \ldots, 4k+3]) = [\{2*i, i = 1, \ldots, 2k+1\}, \{2*i-1, i = 1, \ldots, 2k+2\}]$ (even indices and then odd indices) and its corresponding permutation matrix $\mathbf{Q}_{2k+1}$ defined by $[\mathbf{Q}_{2k+1}]_{i,j} = 1$ if $j = \sigma(i)$ and 0 elsewhere.

In the permuted basis, J_{2k+1} has a block structure as follows

$$\mathbf{J}_{2k+1} = \mathbf{Q}_{2k+1}^T \begin{bmatrix} \mathbf{A}_{xyz,k} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_{y,k+1} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{z,k+1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{x,k+1} \end{bmatrix} \mathbf{Q}_{2k+1} \ , \tag{23}$$

where the block matrices satisfy the following identities

$$\mathbf{A}_{xyz,k}^2 = I_k, \qquad \mathbf{A}_{xyz,k}^T = \mathbf{A}_{xyz,k} \ , \tag{24}$$

$$\mathbf{A}_{x,k+1}^2 = I_{k+1}, \qquad \mathbf{A}_{x,k+1}^T = \mathbf{A}_{x,k+1} \ , \tag{25}$$

$$\mathbf{A}_{y,k+1}\mathbf{A}_{y,k+1}^T = I_{k+1}, \qquad \mathbf{A}_{z,k+1}^T = \mathbf{A}_{y,k+1} \ . \tag{26}$$

For even l, $l = 2k$, four subspaces of E_l are also introduced, comprising functions divisible by zy , xz, xy or indivisible by x,y or z. The name, basis and dimension of these subspaces are given in table 2.

Name	Basis	Div. by	Dimension	Image by J_{2k}
$E_{yz,k}$	$S^l_{-l+2i-1}, i = 1, \ldots, k$	yz	k	$E_{yz,k}$
$E_{xz,k}$	$S^l_{2i-2}, i = 1, \ldots, k$	xz	k	$E_{xy,k}$
$E_{xy,k}$	$S^l_{-l+2i-2}, i = 1, \ldots, k$	xy	k	$E_{xz,k}$
$E_{1,k+1}$	$S^l_{2i-1}, i = 1, \ldots, k+1$	—	$k+1$	$E_{1,k+1}$

Table 2: Orthogonal subspaces of E_{2k}.

The following decomposition is obtained

$$E_{2k} = E_{yz,k} \oplus E_{xz,k} \oplus E_{xy,k} \oplus E_{1,k+1} \ . \tag{27}$$

and the permutation matrix $\mathbf{Q}_{2k}$ for the permutation $\sigma([1, 2, \ldots, 4k+1]) = [\{2*i, i = 1, \ldots, 2k\}$, $\{2*i - 1, i = 1, \ldots, 2k + 1\}]$ allows the block structure of $\mathbf{J}_{2k}$ to be obtained, as follows:

$$\mathbf{J}_{2k} = \mathbf{Q}_{2k}^T \begin{bmatrix} \mathbf{A}_{yz,k} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_{xy,k} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{xz,k} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{1,k+1} \end{bmatrix} \mathbf{Q}_{2k} \ , \tag{28}$$

where the block matrices satisfy the following identities

$$\mathbf{A}_{yz,k}^2 = I_k, \qquad \mathbf{A}_{yz,k}^T = \mathbf{A}_{yz,k} \ , \tag{29}$$

$$\mathbf{A}_{1,k+1}^2 = I_{k+1}, \qquad \mathbf{A}_{1,k+1}^T = \mathbf{A}_{1,k+1} \ , \tag{30}$$

$$\mathbf{A}_{xy,k}\mathbf{A}_{xy,k+}^T = I_k, \qquad \mathbf{A}_{xz,k}^T = \mathbf{A}_{xy,k} \ . \tag{31}$$

With a view to pre-computing the matrices $\mathbf{J}_l$, the properties of the blocks allow storage of the upper (or lower) triangular parts including the diagonal, of the symmetric diagonal blocks and only one of the two non diagonal blocks. That is $2k^2 + 2k + 1 = \frac{1}{2}l^2 + l + 1$ elements for $l = 2k$ and $2(k + 1)^2 = \frac{1}{2}(l + 1)^2$ elements for $l = 2k + 1$. This is highly efficient.

Up to a value of $l = 40$, the computation of the blocks of $\mathbf{J}_l$ is a straight forward job using a linear algebra package in any algebra computer system. Exact values for the coefficients of $\mathbf{J}_l$ are obtained in this way. A similar method would be less appropriate within a numerical program and may be prone to numerical instabilities. It is also possible to compute matrices $\mathbf{J}_l$ by recurrence on l as seen in the next paragraph.

3 Computing $\mathbf{J}_l$ by recurrence

The decomposition theorem, for $l \geq 1$,

$$S_{1,s} S_{l,m} = \sum_{\mu=-l-1}^{l+1} \left[\begin{array}{ccc} l+1 & l & 1 \\ \mu & m & s \end{array} \right]_{\mathbb{R}} S_{l+1,\mu} + \sum_{\mu=-l+1}^{l-1} \left[\begin{array}{ccc} l-1 & l & 1 \\ \mu & m & s \end{array} \right]_{\mathbb{R}} S_{l-1,\mu} \; , \tag{32}$$

where $s \in \{-1, 0, 1\}$ and the real Gaunt coefficients are given by

$$\left[\begin{array}{ccc} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \end{array} \right]_{\mathbb{R}} = \int S_{l_1,m_1} S_{l_2,m_2} S_{l_3,m_3} \, d\Omega \; . \tag{33}$$

They are given a matrix form for each s in $\{-1, 0, 1\}$ by:

$$v \, \mathbf{B}_l = \mathbf{B}_{l+1} \, \mathbf{G}_v^l \; + \; \mathbf{B}_{l-1} \, [\mathbf{G}_v^{l-1}]^T \; , \tag{34}$$

where $\mathbf{B}_l$ is the row matrix of the $2l+1$ real spherical harmonic basis $S_{l,-l-1+i}, i = 1..2l+1$, v belongs to the set $\{x, y, z\}$ and the $(2l+3) \times (2l+1)$ matrix $\mathbf{G}_v^l$ is defined by

$$[\mathbf{G}_v^l]_{i,j} = (-1)^{s(v)} \frac{2\sqrt{\pi}}{\sqrt{3}} \left[\begin{array}{ccc} l+1 & l & 1 \\ i-l-2 & j-l-1 & s(v) \end{array} \right]_{\mathbb{R}} \; , \tag{35}$$

where $s(y) = -1, s(z) = 0$ and $s(x) = 1$.
This is because

$$S_{1,-1} = -\frac{\sqrt{3}}{2\sqrt{\pi}} \, y, \; S_{1,0} = \frac{\sqrt{3}}{2\sqrt{\pi}} \, z, \; S_{1,1} = -\frac{\sqrt{3}}{2\sqrt{\pi}} \, x \; . \tag{36}$$

As an example, $l = 2$, gives:

$$\mathbf{G}_x^2 = \left[\begin{array}{ccccc} -\frac{\sqrt{42}}{14} & 0 & 0 & 0 & 0 \\ 0 & -\frac{\sqrt{7}}{7} & 0 & 0 & 0 \\ \frac{\sqrt{70}}{70} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\sqrt{105}}{35} & 0 \\ 0 & 0 & -\frac{\sqrt{210}}{35} & 0 & \frac{\sqrt{70}}{70} \\ 0 & 0 & 0 & -\frac{\sqrt{7}}{7} & 0 \\ 0 & 0 & 0 & 0 & -\frac{\sqrt{42}}{14} \end{array} \right] \tag{37}$$

$$\mathbf{G}_y^2 = \left[\begin{array}{ccccc} 0 & 0 & 0 & 0 & -\frac{\sqrt{42}}{14} \\ 0 & 0 & 0 & -\frac{\sqrt{7}}{7} & 0 \\ 0 & 0 & -\frac{\sqrt{210}}{35} & 0 & -\frac{\sqrt{70}}{70} \\ 0 & \frac{\sqrt{105}}{35} & 0 & 0 & 0 \\ \frac{\sqrt{70}}{70} & 0 & 0 & 0 & 0 \\ 0 & \frac{\sqrt{7}}{7} & 0 & 0 & 0 \\ \frac{\sqrt{42}}{14} & 0 & 0 & 0 & 0 \end{array} \right] \tag{38}$$

$$\mathbf{G}_z^2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ \frac{\sqrt{7}}{7} & 0 & 0 & 0 & 0 \\ 0 & \frac{2\sqrt{70}}{35} & 0 & 0 & 0 \\ 0 & 0 & \frac{3\sqrt{35}}{35} & 0 & 0 \\ 0 & 0 & 0 & \frac{2\sqrt{70}}{35} & 0 \\ 0 & 0 & 0 & 0 & \frac{\sqrt{7}}{7} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{39}$$

Using the conditions that lead to vanishing Gaunt coefficients and an explicit expression for non-zero coefficients such as given in [1], a simple explicit expression for matrices $\mathbf{G}_x^l$, $\mathbf{G}_y^l$ and $\mathbf{G}_z^l$ results.

Lemma 2.– *For $l \geq 1$, the non-zero elements of $\mathbf{G}_x^l$ and $\mathbf{G}_y^l$ are given by :*

$$\begin{aligned} \left[\mathbf{G}_x^l\right]_{2+k,k} = \left[\mathbf{G}_x^l\right]_{2l+2-k,2l+2-k} &= \left[\mathbf{G}_y^l\right]_{2l+2-k,k} = -\left[\mathbf{G}_y^l\right]_{2+k,2l+2-k} = \\ &\frac{\sqrt{k(k+1)}}{2\sqrt{(2l+1)(2l+3)}}\,, 1 \leq k \leq l-1\,, \\ \left[\mathbf{G}_x^l\right]_{k,k} = \left[\mathbf{G}_x^l\right]_{2l+4-k,2l+2-k} &= \left[\mathbf{G}_y^l\right]_{k,2l+2-k} = -\left[\mathbf{G}_y^l\right]_{2l+4-k,k} = \\ &-\frac{\sqrt{(2l+2-k)(2l+3-k)}}{2\sqrt{(2l+1)(2l+3)}}\,, 1 \leq k \leq l\,, \\ \left[\mathbf{G}_x^l\right]_{l+2,l+2} = \left[\mathbf{G}_y^l\right]_{l+2,l} &= \frac{\sqrt{2l(l+1)}}{2\sqrt{(2l+1)(2l+3)}}\,, \\ \left[\mathbf{G}_x^l\right]_{l+3,l+1} = \left[\mathbf{G}_y^l\right]_{l+1,l+1} &= -\frac{\sqrt{2(l+1)(l+2)}}{2\sqrt{(2l+1)(2l+3)}}\,. \end{aligned}$$

Lemma 3.– *For $l \geq 1$, $\mathbf{G}_z^l$ is given by :*

$$\begin{aligned} \left[\mathbf{G}_z^l\right]_{k+1,k} &= \frac{\sqrt{k(2l+2-k)}}{\sqrt{(2l+1)(2l+3)}}\,, 1 \leq k \leq 2l+1\,, \\ \left[\mathbf{G}_z^l\right]_{i,j} &= 0 \textit{ elsewhere.} \end{aligned}$$

Theorem.– *For $l \geq 2$, the following equations are satisfied*

$$\mathbf{G}_x^l \mathbf{J}_l = \mathbf{J}_{l+1} \mathbf{G}_x^l\,, \tag{40}$$
$$\mathbf{G}_z^l \mathbf{J}_l = \mathbf{J}_{l+1} \mathbf{G}_y^l\,, \tag{41}$$
$$\mathbf{G}_y^l \mathbf{J}_l = \mathbf{J}_{l+1} \mathbf{G}_z^l\,. \tag{42}$$

Proof.– For $x = s$, (34) is

$$x\ \mathbf{S}_l = \mathbf{S}_{l+1}\ \mathbf{G}_x^l\ +\ \mathbf{S}_{l-1}\ [\mathbf{G}_x^{l-1}]^T\,. \tag{43}$$

Applying the operator J_l (for exchange of y and z) on both sides of this equation gives

$$x\ \mathbf{S}_l \mathbf{J}_l = \mathbf{S}_{l+1}\ \mathbf{J}_{l+1}\ \mathbf{G}_x^l\ +\ \mathbf{S}_{l-1}\ \mathbf{J}_{l-1} [\mathbf{G}_x^{l-1}]^T\,. \tag{44}$$

Expanding the left-hand side of this equation again using (43) gives

$$\left[\mathbf{S}_{l+1}\,\mathbf{G}_x^l\;+\;\mathbf{S}_{l-1}\,[\mathbf{G}_x^{l-1}]^T\right]\mathbf{J}_l=\mathbf{S}_{l+1}\,\mathbf{J}_{l+1}\,\mathbf{G}_x^l\;+\;\mathbf{S}_{l-1}\,\mathbf{J}_{l-1}[\mathbf{G}_x^{l-1}]^T\;. \tag{45}$$

Now identify right-hand factors of $\mathbf{S}_{l+1}$ and $\mathbf{S}_{l-1}$ on both sides which gives (40) and

$$[\mathbf{G}_x^{l-1}]^T\mathbf{J}_l=\mathbf{J}_{l-1}\,[\mathbf{G}_x^{l-1}]^T\;, \tag{46}$$

which is equivalent to (40) with $l-1$ instead of l.

Identities (41) and (42) are proven in a similar way.

As proven in lemma 3, $\mathbf{G}_z^l$ without its first and last rows is a diagonal matrix, denoted $\hat{\mathbf{G}}_z^l$ of dimension $2l+1$. By equation (42), $\mathbf{G}_y^l\mathbf{J}_l[\hat{\mathbf{G}}_z^l]^{-1}$ is just $\mathbf{J}_{l+1}$ without its first and last columns. Since $\mathbf{J}_{l+1}$ is a symmetric matrix, all the elements of the first and last columns (except the corners) may be recovered in the first and last rows of $\mathbf{G}_y^l\mathbf{J}_l[\hat{\mathbf{G}}_z^l]^{-1}$.

Values in the corners of $\mathbf{J}_{l+1}$ are given by the following lemma.

Lemma 4.– *For* $l\geq 1$

$$[\mathbf{J}_l]_{1,1}=[\mathbf{J}_l]_{1,2l+1}=[\mathbf{J}_l]_{2l+1,1}=0,\;\;[\mathbf{J}_l]_{2l+1,2l+1}=2^{1-l}\;. \tag{47}$$

Proof.– For $l=2k$ (resp. $l=2k+1$), the first element of $\mathbf{S}_l$ belongs to $E_{xy,k}$ (resp. $E_{y,k+1}$) and, as $E_{xy,k}\cup J_l(E_{xy,k})=\{0\}$ (resp. $E_{y,k+1}\cup J_l(E_{y,k+1})=\{0\}$), gives $\mathbf{J}_l]_{1,1}=0$.
A similar argument leads to $[\mathbf{J}_l]_{1,2l+1}=[\mathbf{J}_l]_{2l+1,1}=0$.

For $l\geq 1$, we have

$$S_{l,l}=K\sin^l\theta\;\cos l\phi\;\text{ with }K=\frac{(-1)^l}{2^l l!}\sqrt{\frac{(2l+1)!}{2\pi}}\;. \tag{48}$$

Therefore

$$S_{l,l}=K\;\Re[\sin^l(\cos l\phi+i\sin l\phi)]=K\;\Re[\sin^l\theta(\cos\phi+i\sin\phi)^l]=K\;\Re[(x+iy)^l]\;. \tag{49}$$

Applying operator J gives

$$J(S_{l,l})=K\;\Re[(x+iz)^l]=K\;\Re[(\sin\theta\cos\phi+i\cos\theta)^l]\;. \tag{50}$$

So

$$\begin{aligned}[\mathbf{J}_l]_{2l+1,2l+1}&=\int S_{l,l}J(S_{l,l})\;d\Omega\\&=K^2\int_0^{2\pi}\int_0^{\pi}\sin^{l+1}\theta\;\cos l\phi\;\Re[(\sin\theta\cos\phi+i\cos\theta)^l]\;d\theta\;d\phi\;.\end{aligned}$$

In the binomial expansion inside the real part, only the term in $\cos^l\phi$ gives a non-zero contribution because

$$\int_0^{2\pi}\cos l\phi\;\cos^k\phi\;d\;phi=0\;\text{ for }0\leq k<l\;. \tag{51}$$

So

$$[\mathbf{J}_l]_{2l+1,2l+1}=K^2\;\int_0^{2\pi}\cos l\phi\;\cos^l\phi\;d\phi\;\int_0^{2\pi}\sin^{2l+1}\theta\;d\theta\;. \tag{52}$$

The result then follows from identity (21) and

$$\int_0^{2\pi} \cos l\phi \, \cos^l \phi \, d\phi = \frac{\pi}{2^{l-1}} \ , \ l \geq 1 \ . \tag{53}$$

A sketch of the recurrence algorithm for computing $\mathbf{J}_l$ is given in figure 1.

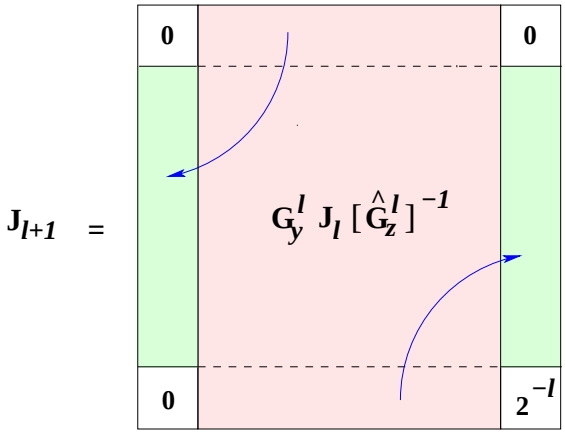

Figure 1: Sketch of the recurrence algorithm for $\mathbf{J}_l$.

4 Programs

Three main tasks have been assigned to a computer algebra system for achieving the goals of this study :

1. Check the correctness of all formulae in this document,
2. Generate a file to be used by numerical C functions that contains the exact values of entries in matrices $\mathbf{J}_l$ for $1 \leq l \leq l_{\max}$ for a given maximum value $l_{\max}$ of l.
3. Test the precision of the implemented C function that computes rotation matrices by producing exact expected results.

The Maple system was used here but certainly any other general computer algebra system would also be suitable.

As pointed out below, for a given value of l, the number of coefficients to be stored is $\frac{1}{2}l^2 + l + 1$ for even l and $\frac{1}{2}(l+1)^2$ for odd l. Following the conjecture, checked indeed for used values of l, each coefficient in $\mathbf{J}_l$ has a canonical form $a\sqrt{b}/2^n$. The values of a, b and n are sequentially stored in a file as integers, thus without any loss of information. Appendix programs may now produce files with numerical values of the coefficients with the chosen precision (usual double precision or extended precision) to be read by C functions.

One convenient method of computing rotation matrices with exact entries is to used rotation angles α, β, γ with rational sines and cosines. Given a rational number t, an angle $\theta \in [0, \pi)$ is well defined by $\tan\frac{\theta}{2} = t$, $\cos\theta = \frac{1-t^2}{1+t^2}$, $\sin\theta = \frac{2t}{1+t^2}$. As polynomials in $\cos\theta$ and $\sin\theta$, the values of $\cos k\theta$ and $\sin k\theta$, for any integer k, are also rational numbers and thus every element in any $\mathbf{J}_l$ has a

closed form representation. However this method is very time-consuming and it may be efficiently replaced by simply computing $\mathbf{J}_l$ using floating point numbers with arbitrarily large, but fixed, extended precision within the computer algebra system.

For a given dimension $n \geq 3$, let us denote by $\mathcal{M}(n, \mathbb{R})$ the set of real $n \times n$ matrices and by $\mathcal{O}(n, \mathbb{R})$ its subset of orthogonal matrices. Let $\mathbf{A} \in \mathcal{O}(n, \mathbb{R})$ and $\mathbf{B} \in \mathcal{M}(n, \mathbb{R})$ an approximation of $\mathbf{A}$ obtained by a numerical procedure. The error may be estimated by considering the distance $||\mathbf{A} - \mathbf{B}||$ between $\mathbf{A}$ and $\mathbf{B}$ where $||.||$ is a norm on $\mathcal{M}(n, \mathbb{R})$. The Frobenius norm is used here, defined by

$$||\mathbf{A}||_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{A}_{i,j}^2 = \mathrm{Tr}(A^T A) \; . \tag{54}$$

For $\mathbf{A} \in \mathcal{O}(n, \mathbb{R})$, $||\mathbf{A} - \mathbf{B}||_F = ||\mathbf{A}^T\mathbf{B} - \mathbf{I}_n||_F$.
When $\mathbf{A} = \Delta_l(\alpha, \beta, \gamma)$, the number $e = ||\mathbf{A} - \mathbf{B}||_F$ is related to quadratic mean σ of the absolute errors on the entries of the C-computed matrix $\mathbf{B}$ by the relation $e = (2l + 1)\sigma$. As an example, figure 4 shows the histogram of absolute errors on the entries of $\mathbf{B}$ for $l = 20, \alpha = 0.1, \beta = 0.2$ and $\gamma = 0.3$. The mean is $m = 1.527 \times 10^{-19}$ and the quadratic mean $\sigma = 4.636 \times 10^{-17}$.

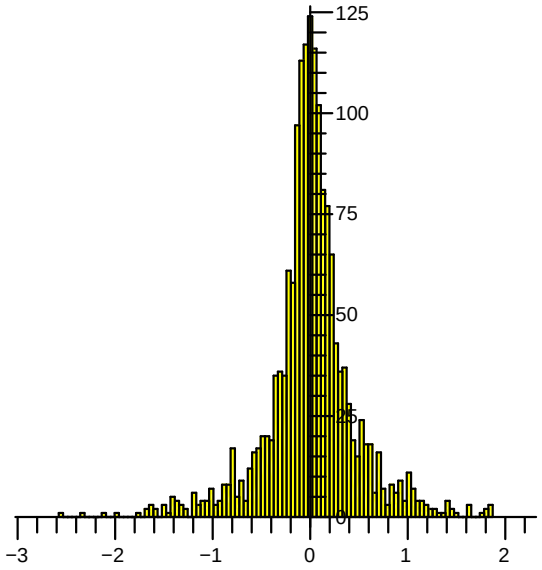

Figure 2: Histogram for absolute errors on the 1681 entries of $\Delta_{20}(0.1, 0.2, 0.3)$ computed by the C function. The unit along the x-axis is 10^{-16}.

Figures 3 and 4 show the values of the error distance e when computing $\Delta_l(\alpha, \beta, 0)$ with the C function for $l = 10$ and $-\pi \leq \alpha \leq \pi$, $0 \leq \beta \leq \pi$. A total of 1000 trials have been done by choosing α and β at random with an uniform distribution in their definition interval with the extra condition that they must be exactly representable numbers in usual double precision format.
Although suspected in [5], no numerical instability is observed when the angles correspond to a point near the poles on the unit sphere, that is when β is close to 0 or π.

$l_{\max}$	5	10	15	20
Time	3.010 sec.	20.31 sec.	70.58 sec.	181.2 sec.

Table 3: Computation time for 10^5 calls of C rotation function.

Table 4 shows the time for 10^5 calls of the rotation C function to compute $\hat{\mathbf{R}}_l(\alpha, \beta, \gamma), 1 \leq l \leq l_{\max}$ for different values of $l_{\max}$ and random values of angles α, β, γ. The program was run on a workstation equipped with an Intel Pentium 4 processor at 3.40 GHz.

5 Discussion

A well-established procedure for rotation of real spherical harmonics in space-fixed axes is that using so-called Wigner rotation matrices. This work investigates alternative techniques, based on successive axial rotations about the cartesian axes and suitable permutations of these axes. An efficient method, due to Kautz [2] used in the field of image rotation has been improved and made more systematic in the present work. The result is a highly compact form for the rotation matrices which have a very simple structure. In fact, it is the simplest possible and each element of the upper triangle may be stored in the form of three integers. Furthermore, the entire matrix may be pre-calculated for a given maximum l value of the spherical harmonics. This is clearly advantageous compared with the Wigner matrix strategy and tests have shown the resulting algorithm to be very fast. Indeed, with so few instructions to execute it is difficult to imagine a faster procedure for a given precision.
Rigorous relationships are also given to the Gaunt coefficients and there is a section describing recursive procedures for generating the matrices from previously calculated elements for lower l values. This obviously enables a progressive construction of matrices with no superfluous evaluations. Furthermore, this matrix evaluation is carried out once and for all and simultaneously for all the values of l contrary to previous two-term recurrence approaches for one l value at a time (see [4]).
The structure of the rotation matrices presented here has been completely characterized. They are shown to comprise blocks of non-zero elements of a simple exact form.

6 Conclusion

A new technique for evaluating, storing and implementing rotation matrices in a real spherical harmonic basis has been put forward. The relationship to previous work is emphasized, as are the advantages of that described herein.
This work paves the way for highly efficient algorithms required in the rotation of atomic orbitals in space-fixed axes for molecular electronic calculations in quantum chemistry. Indeed, the rotation matrix structure evidenced here has been checked using computer algebra and a program written in C generated to evaluate the elements and carry out the rotations to high accuracy. The structural information (regarding matrix elements) allows the storage of the upper triangle of rotation matrix elements in the form of three integers per element. These matrices and the requisite integers are generated once and for all when the atomic orbitals are input, since the maximum value of l involved is known at this stage. A table of timings for hundred thousand calls of the rotation matrix shows how fast this is for the high precision and numerical stability studied in the closing sections of this work.

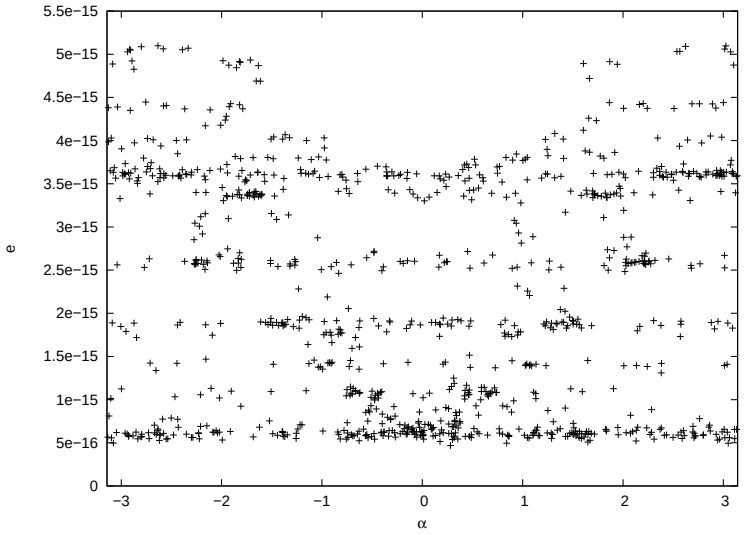

Figure 3: Error distances of C-computed $\Delta_l(\alpha, \beta, 0)$ for $l = 10$ – Projection on α-axis.

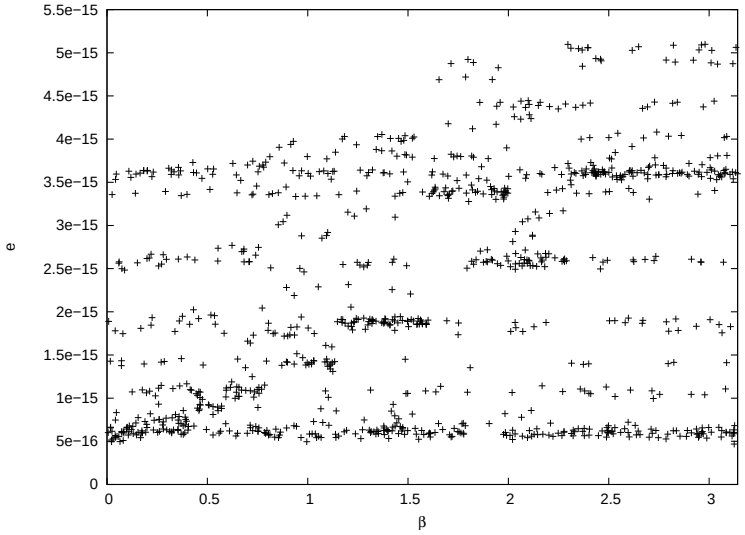

Figure 4: Error distances of C-computed $\Delta_l(\alpha, \beta, 0)$ for $l = 10$ – Projection on β-axis.

References

[1] D.A. Varshalovitch, A.N. Moskalev, and V.K. Khersonskii. *Quantum theory of Angular momentum techniques*. World Scientific, 1989. 514 p.

[2] J. Kautz, P.-P. Sloan, and J. Snyder. Fast, arbitrary BRDF shading for low-frequency lighting using spherical harmonics. In *Proc. 13th Eurographics Workshop on Rendering, Nicosia, Cyprus, 2002*, 2002. 7 p.

[3] J. Ivanic and K. Ruedenberg. Rotation matrices for real spherical harmonics.Direct determination by recursion. *J. Phys.Chem.*, 100:6342–6347, 1996.

[4] M. A. Blanco, M. Flórez, and M. Bermejo. Evaluation of the rotation matrices in the basis of real spherical harmonics. *J. Mol. Struc. (Theochem)*, 419:19–27, 1997.

[5] J. Křivánek, J. Konttinen, S. Pattanaik, and K. Bouatouch. Fast approximation to spherical harmonic rotation. Technical Report 1728, IRISA, Rennes, France, 2005. 14 p.

[6] Y. Kosmann-Schwartzbach. *Groupes et symétries*. Ed. École Polytechnique, 2003. 141 p.

[7] E.O. Steinborn and K. Ruedenberg. Rotation and translation of regular and irregular solid spherical harmonics. *Adv. Quant. Chem.*, 7:1–81, 1973.

[8] C.H. Choi, J. Ivanic, M.S. Gordon, and K. Ruedenberg. Rapid and stable determination of rotation matrices between spherical harmonics by direct recursion. *J. Chem. Phys.*, 111:8825–8831, 1999.

[9] V. Devanathan. *Angular momentum techniques in quantum machanics*. Kluwer Acad. Publ., 2002. 242 p.

[10] J. Křivánek, J. Konttinen, K. Bouatouch, S. Pattanaik, and J. Zára. Fast approximation to spherical harmonic rotation. In *Proc. SCCG'06, 22nd Spring Conf. on Computer Graphics, Časť-Papiernička, Sloviaka*. ACM Press, New York, NY, USA, 2006.

[11] J.R. Ivarez Collado, J. Fernández Rico, R. López, M. Paniagua, and G. Ramírez. Rotation of real spherical harmonics. *Comput. Phys. Commun.*, 52:323–331, 1989.

Appendix – Maple verification programs

A first Maple commands file gives the definition of complex and real spherical harmonics.

spherical.mpl

```
# spherical.mpl
# Authors: DP & PH
# Copyright (c) CNRS, 2006

# Integration on the unit sphere
intSpherik := proc(f)
    local g;
    g := int(f,phi=0..2*Pi);
    simplify(int(g*sin(theta),theta=0..Pi));
end:

# Associated Legendre functions of the 1st kind
associatedLegendre := proc(l,m)
    if l=0 then RETURN(1); fi;
    if m<0 then ERROR("m should be non negative!"); fi;
    (-1)^m*1/(2^l*l!)*(sin(theta))^m*
       subs(x=cos(theta), diff( (x^2-1)^l ,seq(x,i=1..(l+m))) );
end:

# Complex spherical harmonics
YY := proc(l,m)
   local mm,x,res;
   if l=0 and m=0 then RETURN(1/(4*Pi)); fi;
   mm:=abs(m);
   res :=I^(m+mm)*
         simplify(sqrt((2*l+1)*(l-mm)!/(l+mm)!/4/Pi)*1/2^l/l!)*
         sin(theta)^mm*subs(x=cos(theta), diff( (x^2-1)^l,x$(l+mm)))*
         exp(I*m*phi);
end:

# Real spherical harmonics
SS := proc(l,m)
    local cte;
    if m=0 then
      RETURN(sqrt((2*l+1)/(4*Pi))*P(l,cos(theta)));
    fi;
    cte := sqrt((2*l+1)*(l-abs(m))!/(l+abs(m))!/2/Pi);
    if m>0 then
      RETURN( cte*associatedLegendre(l,m,theta)*cos(m*phi) );
    fi;
    cte*associatedLegendre(l,-m,theta)*sin(-m*phi);
end:
```

Only the mathematical definition of real Gaunt coefficients will be used below.

gaunt.mpl

```
# real Gaunt coefficients
# mathematical definition
realGaunt := proc(l1,l2,l3,m1,m2,m3)
   intSpherik(SS(l1,m1)*SS(l2,m2)*SS(l3,m3));
end:
```

The main functions to deal with rotation matrices have been collected in the following Maple command file.

rotation.mpl

```
# rotation.mpl
# MEPOM - Didier Pinchon
# Version 1.0 , 1 September 2006
# rotation matrices for real spherical harmonics
with(linalg):
read("spherical.mpl"):
read("gaunt.mpl"):
with(linalg):

# real spherical harmonics with variables x,y,z
Sxyz := proc(l,m)
  local leq,res;
  leq := {cos(theta)=z/r,sin(theta)=sqrt(x^2+y^2)/r,cos(phi)=x/sqrt(x^2+y^2),
   sin(phi)=y/sqrt(x^2+y^2)};
  res := simplify(r^l*subs(leq,expand(SS(l,m))));
  res := simplify(subs(r=sqrt(x^2+y^2+z^2),res));

end:

Sbasis := proc(l) [seq(Sxyz(l,m),m=-l..l)]  end:

# complex spherical harmonics with variables x,y,z
Yxyz := proc(l,m)
  local leq,res;
  leq := {cos(theta)=z/r,sin(theta)=sqrt(x^2+y^2)/r,cos(phi)=x/sqrt(x^2+y^2),
   sin(phi)=y/sqrt(x^2+y^2)};
  res :=simplify(r^l*subs(leq,expand(convert(YY(l,m),trig))));
  res := simplify(subs(r=sqrt(x^2+y^2+z^2),res));
end:

Ybasis := proc(l) [seq(Yxyz(l,m),m=-l..l)] end:

setCoeffs := proc(f,lv)
   local g,lc;
   # Maple "coeffs" does not work
   lc := {seq(coeff(f,lv[1],i),i = 0..degree(f,lv[1]))};
   if nops(lv)=1 then RETURN(lc); fi;
   {seq(op(setCoeffs(g,lv[2..-1])), g = lc)};
end:
```

```
linCoeffs := proc(f,basis)
  local g,leq,i,sol,dim;
  dim := nops(basis);
  g := add(c[i]*basis[i],i=1..dim)- expand(f);
  leq:= setCoeffs(g,[x,y,z]);
  sol := solve(leq,{seq(c[i],i=1..dim)});
  [seq(combine(subs(sol,c[i])),i=1..dim)];
end:

# Matrix of a linear operator in a space given a base
opMatrix := proc(opf,basis)
   local mat,i,j,dim,lco;
   dim := nops(basis);
   mat := matrix(dim,dim,0);
   for j to dim do
     lco := linCoeffs(opf(basis[j]), basis);
     for i to dim do
       mat[i,j] := simplify(lco[i]);
     od;
   od;
   mat;
end:

# Matrix of a linear operator between two spaces given two bases fam and basis
opMatrix2 := proc(opf,fam,basis)
   local mat,i,j,dim,dim2,lco;
   dim := nops(basis);
   dim2 := nops(fam);
   mat := matrix(dim,dim2,0);
   for j to dim2 do
     lco := linCoeffs(opf(fam[j]), basis);
     for i to dim do
       mat[i,j] := simplify(lco[i]);
     od;
   od;
   mat;
end:

# Change of base matrix
changeBase := proc(bas1,bas2) opMatrix2(a -> a, bas1,bas2); end:

# Permutation matrix
permutationMat := proc(ll)
  local dim,mat,i;
  dim := nops(ll);
  mat := matrix(dim,dim,0);
  for i to dim do
    mat[i,ll[i]]:=1;
  od;
  mat;
end:
```

```
# The elementary rotations
rotXY := proc(f,alpha) subs({x=cos(alpha)*x+sin(alpha)*y,y=-sin(alpha)*x+cos(alpha)*y},f) end:
rotXZ := proc(f,alpha) subs({x=cos(alpha)*x+sin(alpha)*z,z=-sin(alpha)*x+cos(alpha)*z},f) end:
rotYZ := proc(f,alpha) subs({y=cos(alpha)*y+sin(alpha)*z,z=-sin(alpha)*y+cos(alpha)*z},f) end:

# Symmetry z <-> y and its matrix in S-basis
symYZ := proc(f) subs({y=z,z=y},f); end:
mkMatJ := proc(l) opMatrix(symYZ, Sbasis(l)); end:

# The special permutation matrix
mkMatQ := proc(l) permutationMat([seq(2*i,i=1..l),seq(2*i+1,i=0..l)]); end:

# Invariant subspaces and symmetric subspaces of E^l by symYZ and block matrices
mkBxyz := proc(k) [seq(SS(2*k+1,-2*k-2+2*i),i=1..k)]; end:
mkExyz := proc(k) [seq(Sxyz(2*k+1,-2*k-2+2*i),i=1..k)]; end:
mkAxyz := proc(k) opMatrix(symYZ, mkExyz(k)); end:

mkBx := proc(k) [seq(SS(2*k-1,-1+2*i),i=1..k)]; end:
mkEx := proc(k) [seq(Sxyz(2*k-1,-1+2*i),i=1..k)]; end:
mkAx := proc(k) opMatrix(symYZ, mkEx(k)); end:

mkBz := proc(k) [seq(SS(2*k-1,-2+2*i),i=1..k)]; end:
mkEz := proc(k) [seq(Sxyz(2*k-1,-2+2*i),i=1..k)]; end:
mkAz := proc(k) opMatrix2(symYZ, mkEz(k), mkEy(k)); end:

mkBy := proc(k) [seq(SS(2*k-1,-2*k+2*i-1),i=1..k)]; end:
mkEy := proc(k) [seq(Sxyz(2*k-1,-2*k+2*i-1),i=1..k)]; end:
mkAy := proc(k) opMatrix2(symYZ, mkEy(k), mkEz(k)); end:

mkByz := proc(k) [seq(SS(2*k,-2*k+2*i-1),i=1..k)]; end:
mkEyz := proc(k) [seq(Sxyz(2*k,-2*k+2*i-1),i=1..k)]; end:
mkAyz := proc(k) opMatrix(symYZ, mkEyz(k)); end:

mkBxz := proc(k) [seq(SS(2*k,2*i-1),i=1..k)]; end:
mkExz := proc(k) [seq(Sxyz(2*k,2*i-1),i=1..k)]; end:
mkAxz := proc(k) opMatrix2(symYZ, mkExz(k), mkExy(k)); end:

mkBxy := proc(k) [seq(SS(2*k,-2*k+2*i-2),i=1..k)]; end:
mkExy := proc(k) [seq(Sxyz(2*k,-2*k+2*i-2),i=1..k)]; end:
mkAxy := proc(k) opMatrix2(symYZ, mkExy(k), mkExz(k)); end:

mkB1 := proc(k) [seq(SS(2*k-2,2*i-2),i=1..k)]; end:
mkE1 := proc(k) [seq(Sxyz(2*k-2,2*i-2),i=1..k)]; end:
mkA1 := proc(k) opMatrix(symYZ, mkE1(k)); end:

# Gaunt matrices
# s*E^l == E^{l+1} C_s^{l} + E^{l-1} D_s^{l} with s in {x,y,z}
mkGauntMat := proc(l,symb)
  local fam, bas, mat,m1;
  if not(symb=`x` or symb=`y` or symb=`z`) then
     ERROR("bad second argument!");
  fi;
  fam := Sbasis(l);
  bas := [op(Sbasis(l+1)), seq(f*(x^2+y^2+z^2), f=Sbasis(l-1))];
```

```
  mat := opMatrix2(f -> symb*f, fam, bas);
  m1 := submatrix(mat,1..2*l+3,1..2*l+1);
  evalm(m1);
end:

# Equivalent definition with real Gaunt coefficients
mkGauntMat2 := proc(l,symb)
   local s,mat,i,j;
   mat := matrix(2*l+3,2*l+1,0);
   if symb=‘x‘ then s:=1; elif symb=‘y‘ then s:=-1; elif symb=‘z‘ then s:=0; else
      ERROR("bad second argument!");
   fi;
   for i to 2*l+3 do
      for j to 2*l+1 do
         mat[i,j] := (-1)^s*2*sqrt(Pi)/sqrt(3)*realGaunt(l+1,l,1,i-l-2,j-l-1,s);
         mat[i,j] := simplify(combine(mat[i,j]));
      od;
   od;
   mat;
end:

# Rotation with z-axis
mkMatZ := proc(l,a)
  local mat,i;
  mat := matrix(2*l+1,2*l+1,0);
  for i to l do
     mat[i,i] := cos((l+1-i)*a);
     mat[i,2*l+2-i] := sin((l+1-i)*a);
     mat[2*l+2-i,i] := - sin((l+1-i)*a);
     mat[2*l+2-i,2*l+2-i] := cos((l+1-i)*a);
  od;
  mat[l+1,l+1]:=1;
  mat;
end:
```

The worksheet **rotation.mws** that follows is for checking the main formulas of the paper.

Rotation matrices for real spherical harmonics: general rotations of atomic orbitals in space-fixed axes

Verifications

Didier Pinchon and Philip E. Hoggan

October 16,2006

```
> restart;
> read("spherical.mpl");
> read("rotation.mpl");
```

Example: matrices of J_l in basis Y_l and S_l (16),(17)

```
> l:=3:
> basY := Ybasis(l):
> basS := Sbasis(l);
```

$$basS := \left[1/8\,\frac{\sqrt{70}y\left(-3\,x^2+y^2\right)}{\sqrt{\pi}}, 1/2\,\frac{\sqrt{105}zyx}{\sqrt{\pi}}, 1/8\,\frac{\sqrt{42}y\left(-4\,z^2+x^2+y^2\right)}{\sqrt{\pi}}, -1/4\,\frac{\sqrt{7}z\left(-2\,z^2+3\,x^2+3\,y^2\right)}{\sqrt{\pi}}, 1/8\,\frac{\sqrt{42}x\left(-4\,z^2+x^2+y^2\right)}{\sqrt{\pi}}, -1/4\,\frac{\sqrt{105}z\left(-x^2+y^2\right)}{\sqrt{\pi}}, 1/8\,\frac{\sqrt{70}x\left(-x^2+3\,y^2\right)}{\sqrt{\pi}}\right]$$

```
> mC := evalm(changeBase(basS,basY));
```

$$mC := \begin{bmatrix} -1/2\,i\sqrt{2} & 0 & 0 & 0 & 0 & 0 & -1/2\,\sqrt{2} \\ 0 & 1/2\,i\sqrt{2} & 0 & 0 & 0 & 1/2\,\sqrt{2} & 0 \\ 0 & 0 & -1/2\,i\sqrt{2} & 0 & -1/2\,\sqrt{2} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1/2\,i\sqrt{2} & 0 & 1/2\,\sqrt{2} & 0 & 0 \\ 0 & -1/2\,i\sqrt{2} & 0 & 0 & 0 & 1/2\,\sqrt{2} & 0 \\ -1/2\,i\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 1/2\,\sqrt{2} \end{bmatrix}$$

```
> matJY := evalm(opMatrix(f-> subs({y=z,z=y},f), basY));
  matJS := evalm(opMatrix(f-> subs({y=z,z=y},f), basS));
  map(f->simplify(f), evalm(mC &* matJS &*
          transpose(map(f -> conjugate(f),mC))-matJY));
```

$$matJY := \begin{bmatrix} 1/8 & 1/8\,i\sqrt{6} & -1/8\sqrt{15} & -1/4\,i\sqrt{5} & 1/8\sqrt{15} & 1/8\,i\sqrt{6} & -1/8 \\ -1/8\,i\sqrt{6} & 1/2 & 1/8\,i\sqrt{10} & 0 & 1/8\,i\sqrt{10} & -1/2 & -1/8\,i\sqrt{6} \\ -1/8\sqrt{15} & -1/8\,i\sqrt{10} & -1/8 & -1/4\,i\sqrt{3} & 1/8 & -1/8\,i\sqrt{10} & 1/8\sqrt{15} \\ 1/4\,i\sqrt{5} & 0 & 1/4\,i\sqrt{3} & 0 & 1/4\,i\sqrt{3} & 0 & 1/4\,i\sqrt{5} \\ 1/8\sqrt{15} & -1/8\,i\sqrt{10} & 1/8 & -1/4\,i\sqrt{3} & -1/8 & -1/8\,i\sqrt{10} & -1/8\sqrt{15} \\ -1/8\,i\sqrt{6} & -1/2 & 1/8\,i\sqrt{10} & 0 & 1/8\,i\sqrt{10} & 1/2 & -1/8\,i\sqrt{6} \\ -1/8 & 1/8\,i\sqrt{6} & 1/8\sqrt{15} & -1/4\,i\sqrt{5} & -1/8\sqrt{15} & 1/8\,i\sqrt{6} & 1/8 \end{bmatrix}$$

$$matJS := \begin{bmatrix} 0 & 0 & 0 & 1/4\sqrt{10} & 0 & -1/4\sqrt{6} & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/4\sqrt{6} & 0 & 1/4\sqrt{10} & 0 \\ 1/4\sqrt{10} & 0 & 1/4\sqrt{6} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1/4 & 0 & -1/4\sqrt{15} \\ -1/4\sqrt{6} & 0 & 1/4\sqrt{10} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1/4\sqrt{15} & 0 & 1/4 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Block structure of J_l

```
> checkBlocks := proc(l)
  local k,matJ, matQ, mA;
  k := iquo(l,2);
  matJ := mkMatJ(l);
  matQ := mkMatQ(l);
  matJ := evalm(matQ &* matJ &* transpose(matQ));
  if type(l,odd) then # equation (23)
    mA := mkAy(k+1);
    if equal(mkAxyz(k), submatrix(matJ, 1..k, 1..k)) and
       equal(mA, submatrix(matJ,k+1..2*k+1,2*k+2..3*k+2)) and
       equal(transpose(mA), submatrix(matJ,2*k+2..3*k+2,k+1..2*k+1)) and
       equal(mkAx(k+1), submatrix(matJ, 3*k+3..4*k+3, 3*k+3..4*k+3))
                 then RETURN(true);
    else
      RETURN(false);
    fi;
  else # l even, equation (28)
    mA := mkAxy(k);
    if equal(mkAyz(k), submatrix(matJ, 1..k, 1..k)) and
       equal(mA, submatrix(matJ,k+1..2*k,2*k+1..3*k)) and
       equal(transpose(mA), submatrix(matJ,2*k+1..3*k,k+1..2*k)) and
       equal(mkA1(k+1), submatrix(matJ, 3*k+1..4*k+1, 3*k+1..4*k+1))
                 then RETURN(true);
    else
      RETURN(false);
    fi;
  fi;
end:



> for l from 2 to 9 do print(l, checkBlocks(l)); od;
```

2, *true*
3, *true*
4, *true*
5, *true*
6, *true*
7, *true*
8, *true*
9, *true*

Matrix Gz (lemma 3)

```
> checkGz := proc(l)
  local mat,k,mat0;
  mat := matrix(2*l+3,2*l+1,0);
  mat0 := matrix(2*l+3,2*l+1,0);
  for k to 2*l+1 do
    mat[k+1,k] :=   sqrt(k*(2*l+2-k))/sqrt((2*l+1)*(2*l+3));
  od;
  mat := map(f -> simplify(f),evalm(mat - mkGauntMat(l,z)));
  equal(mat,mat0)
end:



> seq(checkGz(l),l=1..6);
```

$$true,\ true,\ true,\ true,\ true,\ true$$

Matrix Gy (lemma 2)

```
> checkGy := proc(l)
  local mat,mat0,k;
  mat := matrix(2*l+3,2*l+1,0);
  mat0 := matrix(2*l+3,2*l+1,0);
  for k from 1 to l-1 do
     mat[2*l+2-k,k] := sqrt(k*(k+1)/((2*l+1)*(2*l+3)))/2;
  od;
  mat[l+2,l] := sqrt(2*l*(l+1)/(2*l+1)/(2*l+3))/2;
  mat[l+1,l+1] := -sqrt(2*(l+1)*(l+2)/(2*l+1)/(2*l+3))/2;
  for k from 1 to l do
     mat[k,2*l+2-k] := -sqrt((2*l+2-k)*(2*l+3-k)/((2*l+1)*(2*l+3)))/2;
  od;
  for k from 1 to l do
     mat[2*l+4-k,k] := sqrt((2*l+2-k)*(2*l+3-k)/(2*l+1)/(2*l+3))/2;
  od;
  for k from 1 to l-1 do
     mat[2+k,2*l+2-k] := - sqrt(k*(k+1)/(2*l+1)/(2*l+3))/2;
  od;
  mat := map(f -> simplify(f),evalm(mat - mkGauntMat(l,y)));
  equal(mat,mat0);
end:

> seq(checkGy(l),l=1..6);}
```

$$true,\ true,\ true,\ true,\ true,\ true$$

Matrix Gx (lemma 2)

```
> checkGx := proc(l)
  local mat,mat0,k;
  mat := matrix(2*l+3,2*l+1,0);
  mat0 := matrix(2*l+3,2*l+1,0);
  for k from 1 to l do
     mat[k,k] := -sqrt((2*l+2-k)*(2*l+3-k)/(2*l+1)/(2*l+3))/2;
  od;
  mat[l+2,l+2] := sqrt(2*l*(l+1)/(2*l+1)/(2*l+3))/2;
  mat[l+3,l+1] := -sqrt(2*(l+1)*(l+2)/(2*l+1)/(2*l+3))/2;
  for k from 1 to l-1 do
     mat[2*l+2-k,2*l+2-k] := sqrt(k*(k+1)/((2*l+1)*(2*l+3)))/2;
  od;
  for k from 1 to l-1 do
     mat[2+k,k] := sqrt(k*(k+1)/(2*l+1)/(2*l+3))/2;
  od;
  for k from 1 to l do
     mat[2*l+4-k,2*l+2-k] := - sqrt((2*l+2-k)*(2*l+3-k)/(2*l+1)/(2*l+3))/2;
  od;
  mat := evalm(mat - mkGauntMat(l,x));
  equal(mat,mat0);
end:

seq(checkGx(l),l=1..6);
```

$$true,\ true,\ true,\ true,\ true,\ true$$

Gaunt matrices for l=2. Equation (37), (38), (39).

```
> Gx := mkGauntMat(2,x);
```

$$Gx := \begin{bmatrix} -1/14\sqrt{42} & 0 & 0 & 0 & 0 \\ 0 & -1/7\sqrt{7} & 0 & 0 & 0 \\ \frac{1}{70}\sqrt{70} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/35\sqrt{105} & 0 \\ 0 & 0 & -1/35\sqrt{210} & 0 & \frac{1}{70}\sqrt{70} \\ 0 & 0 & 0 & -1/7\sqrt{7} & 0 \\ 0 & 0 & 0 & 0 & -1/14\sqrt{42} \end{bmatrix}$$

```
> Gy := mkGauntMat(2,y);
```

$$Gx := \begin{bmatrix} 0 & 0 & 0 & 0 & -1/14\sqrt{42} \\ 0 & 0 & 0 & -1/7\sqrt{7} & 0 \\ 0 & 0 & -1/35\sqrt{210} & 0 & -\frac{1}{70}\sqrt{70} \\ 0 & 1/35\sqrt{105} & 0 & 0 & 0 \\ \frac{1}{70}\sqrt{70} & 0 & 0 & 0 & 0 \\ 0 & 1/7\sqrt{7} & 0 & 0 & 0 \\ 1/14\sqrt{42} & 0 & 0 & 0 & 0 \end{bmatrix}$$

```
> Gz := mkGauntMat(2,z);
```

$$Gx := \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1/7\sqrt{7} & 0 & 0 & 0 & 0 \\ 0 & \frac{2}{35}\sqrt{70} & 0 & 0 & 0 \\ 0 & 0 & \frac{3}{35}\sqrt{35} & 0 & 0 \\ 0 & 0 & 0 & \frac{2}{35}\sqrt{70} & 0 \\ 0 & 0 & 0 & 0 & 1/7\sqrt{7} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Recurrence algorithm for J_l

```
> nextJl := proc(mJ, mnextJ)
  local l, mJ1, mat0, Ghatz,Gy, mat, i, j;
  l := iquo(rowdim(mJ),2);
  mJ1 := matrix(2*l+3, 2*l+3, 0);
  mat0 := matrix(2*l+3, 2*l+3, 0);
  Gy := mkGauntMat(l,y);
  Ghatz := submatrix(mkGauntMat(l,z),2..2*l+2,1..2*l+1);
  mat := map(f- -> simplify(f), evalm(Gy &* mJ &* inverse(Ghatz)) );
  for i from 1 to 2*l+3 do
    for j from 1 to 2*l+1 do
      mJ1[i,j+1] := simplify(mat[i,j]);
    od;
  od;
  for i from 1 to 2*l+1 do
    mJ1[i+1,1] := mJ1[1,i+1];
    mJ1[i+1,2*l+3] := mJ1[2*l+3,i+1];
  od;
  mJ1[2*l+3,2*l+3] := 1/2\symbol{94}l;
  equal(map(f -> simplify(f), evalm(mJ1 - mnextJ)), mat0);
end:

> mJ := mkMatJ(1):
  for i from 1 to 9 do
    mJ1 := mkMatJ(i+1):
    print(i, nextJl(mJ,mJ1));
    mJ := mJ1:
  od:
```

1, *true*
2, *true*
3, *true*
4, *true*
5, *true*
6, *true*
7, *true*
8, *true*
9, *true*